



Collaborative Network for Industry, Manufacturing, Business and Logistics in Europe



D3.1

Core Platform Infrastructure

Project Acronym	NIMBLE	
Project Title	Collaboration Network for Industry, Manufacturing, Business and Logistics in Europe	
Project Number	723810	
Work Package	WP3 Core Business Services for the NIMBLE Platform	
Lead Beneficiary	IBM	
Editor	Benny Mandler	IBM
Reviewers	Johannes Innerbichler	SRFG
	Wernher Behrendt	SRFG
	Suat Gonul	SRDC
Contributors	EB	
Dissemination Level	PU	
Contractual Delivery Date	30/04/2017	
Actual Delivery Date	30/04/2017	
Version	V1.0	

Abstract

This is the accompanying documentation of the first prototype version of the NIMBLE integrated platform. The integrated platform took technological pieces developed in different tasks and connected them into a cohesive solution that can benefit external stakeholders. The platform offers companies the opportunity to easily register their identities and catalogues, and then enables all participants to search through this information and to establish business relationships among the entities. The business relations include creating a private communication link between them, and enabling selective data sharing among entities.

NIMBLE in a Nutshell

NIMBLE is the collaboration Network for Industry, Manufacturing, Business and Logistics in Europe. It will develop the infrastructure for a cloud-based, Industry 4.0, Internet-of-Things-enabled B2B platform on which European manufacturing firms can register, publish machine-readable catalogues for products and services, search for suitable supply chain partners, negotiate contracts and supply logistics. Participating companies can establish private and secure B2B and M2M information exchange channels to optimise business work flows. The infrastructure will be developed as open source software under an Apache-type, permissive license. The governance model is a federation of platforms for multi-sided trade, with mandatory interoperation functions and optional added-value business functions that can be provided by third parties. This will foster the growth of a net-centric business ecosystem for sustainable innovation and fair competition as envisaged by the Digital Agenda 2020. Prospective NIMBLE providers can take the open source infrastructure and bundle it with sectorial, regional or functional added value services and launch a new platform in the federation. The project started in October 2016 and will last for 36 months.

Document History

Version	Date	Comments
V0.1	15/03/2017	Initial skeleton version
V0.2	03/04/2017	Add deployment information (SRFG)
V0.3	13/04/2017	Ready for internal review
V0.4	27/04/17	Integrated internal review comments by Hannes
V0.5	28/04/17	Integrated internal review comments by Suat
V0.6	30/04/17	Integrated internal review comments by Wernher
V1.0	30/04/2017	Finalize and integrate all contributions

Table of Contents

Abstract.....	2
NIMBLE in a Nutshell.....	2
Document History	3
List of Figures	5
List of Tables	6
Acronyms	7
1 Introduction	8
2 High level picture – Main components and interactions	8
3 Components deep dive	10
3.1 Nimble Portal Component	11
3.1.1 Frontend Micro-Service.....	11
3.2 Identity Component	12
3.2.1 Identity Micro-service	12
3.3 Catalogue Component	12
3.3.1 Search Micro-service	13
3.3.2 Publish Micro-service	13
3.3.3 Catalogue Micro-service.....	14
3.4 Collaboration Component.....	14
3.4.1 Communication Micro-service	14
3.4.2 Negotiation Micro-service.....	14
3.4.3 Matchmaking Micro-Service	15
3.4.4 Business Process Micro-service.....	15
3.4.5 Product Lifecycle Micro-service	16

3.4.6	Data Sharing Service.....	16
3.5	IoT Component	16
3.5.1	IoT Data Processing Micro-service	16
3.5.2	IoT Analytics Micro-service	17
3.6	The cloud run-time.....	17
3.6.1	The run-time environment:.....	17
4	What is being demonstrated.....	17
5	Integrated platform installation & configuration.....	21
5.1	Public Cloud deployment	21
5.2	Private Cloud deployment.....	23
6	Applied Technologies	24

List of Figures

Figure 1: Main NIMBLE platform components.....	9
Figure 2: Core components - a microservices view.....	10
Figure 3: The NIMBLE git repository	11
Figure 4: Registration front-end of the NIMBLE platform.....	18
Figure 5: Product category selection	19
Figure 6: An example Excel-based product category template	19
Figure 7: UI of Apache Marmotta back-end service for search	20
Figure 8: Design of a business process by using the available platform templates	20
Figure 9: An intermediate step of execution of a business process where the user is expected to approve or decline the incoming configuration update request.....	21
Figure 10: NIMBLE BlueMix account.....	22
Figure 11: CF applications running in the NIMBLE space.....	22
Figure 12: Containers running in the NIMBLE space.....	23

Figure 13: BlueMix services running in the Nimble space23

Figure 14: Deployment Diagram for hosting NIMBLE on premise24

List of Tables

Table 1: Acronyms table7

Table 2: Applied technologies in NIMBLE25

Acronyms

Table 1: Acronyms table

Acronym	Meaning
API	Application Programming Interface
CF	Cloud Foundry
GUI	Graphical User Interface
IaaS	Infrastructure as a Service
IDM	Identity Management
IoT	Internet of Things
JSON	Java Script Object Notation
NIMBLE	Collaboration Network for Industry, Manufacturing, Business and Logistics in Europe
PaaS	Platform as a Service
REST	Representational State Transfer
SDK	Software Development Kit
SPARQL	SPARQL Protocol And RDF Query Language
UAA	User Account and Authentication

1 Introduction

This document accompanies the demonstration of the first integrated NIMBLE platform. In this prototype, an initial set of core capabilities planned within the platform are already in place. Contributions from different tasks have made it into this version of the platform, thus providing a baseline for showing NIMBLE platform capabilities. The platform will go through iterative phases of continuous development deployment and integration, until the final version of the platform which will also include a set of advanced services that go beyond the basic B2B functionality initially offered.

The actual deliverable D3.1 is the working prototype and this report provides the necessary background information concerning components and their interactions. It is not intended to be a full-fledged design document. More detailed information is provided in the NIMBLE architecture document (D2.1: Platform Architecture Specification and Component Design), and individual components will be detailed in their own deliverables.

2 High level picture – Main components and interactions

Figure 1 presents an overall view of the NIMBLE integrated platform. The figure shows the main components which are a part of the platform, and the main interactions between the various components. At its core the NIMBLE platform is a customization of an openly available PaaS infrastructure (Cloud Foundry¹), making it more suitable as a platform to serve the B2B / supply chain domain, considering especially the security requirements and potential scalability, extension and federation needs towards vertical sectors / countries. Thus, as can be seen in the figure, NIMBLE is a cloud platform, with specific capabilities that make it easier for business partners to interact with each other.

Most of the components operate within the cloud environment. Note, that as a development step towards the fast creation of a first platform prototype a local “cloud” based environment was set up and used as a staging service. The ultimate vision, presented herein, is for the core components to be hosted in the cloud.

The front-end is the main access point into the NIMBLE platform for end-users. Developers may interact with the platform using the API (which will be documented in deliverable: D2.3 - Design of an Open API for the NIMBLE Platform). In addition, the federation interaction will be made possible via the API as well. The front-end integrates various components for interacting with the platform applications and back-end. The interworking between all components forms the integrated NIMBLE platform. Figure 1 shows the core components of the NIMBLE back-end. The front-end provides direct access to particular features of the back-end components. NIMBLE platform components are deployed within the cloud run-time either as cloud applications or cloud services², depending on the requirements and mode of interaction

¹ <http://cloudfoundry.org/index.html>

² Cloud applications are stateless, have REST based connectivity, and rely on cloud services for middleware capabilities such as data store and messaging. Cloud services are state-full and are visible only to cloud applications

with each such component. Some back-end services may be external but connected to the cloud via the built-in service brokering capability.

Security components, such as the Identity Management, are also deployed as cloud applications, enabling other NIMBLE components to use it. External entities may use this component as well.

The core components are all deployed as cloud applications and communicate with required services using the cloud binding mechanism. There is a messaging and communication component, deployed as a cloud application, which enables flexible and dynamic interaction among different platform components, and external entities as well.

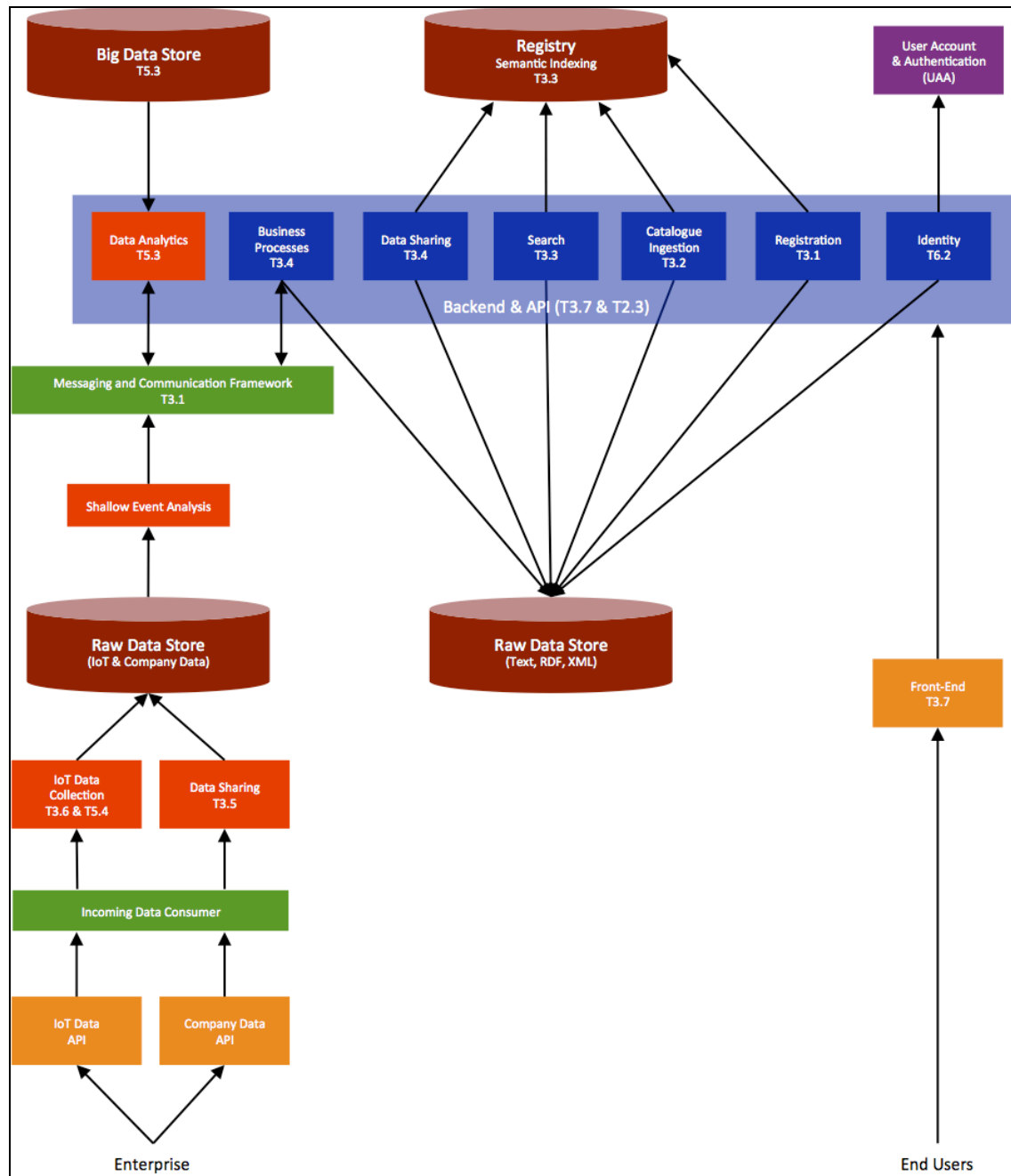


Figure 1: Main NIMBLE platform components

The data management component has a portion that is deployed as a cloud application, to enable the easy interaction with the rest of the components, and it uses a state-full back-end deployed as a cloud service. Only internal platform applications can bind to the data management service, ensuring that no external entity can access the data store directly.

3 Components deep dive

Figure 2 presents a deeper dive into the core platform infrastructure. In the figure, micro-services with similar contexts are grouped in component groups. Each group and micro-service is introduced and its responsibilities, inter-service communication and data-stores are explained.

Applied technologies as a part of the micro-services are also referred to and more details are provided in section 6.

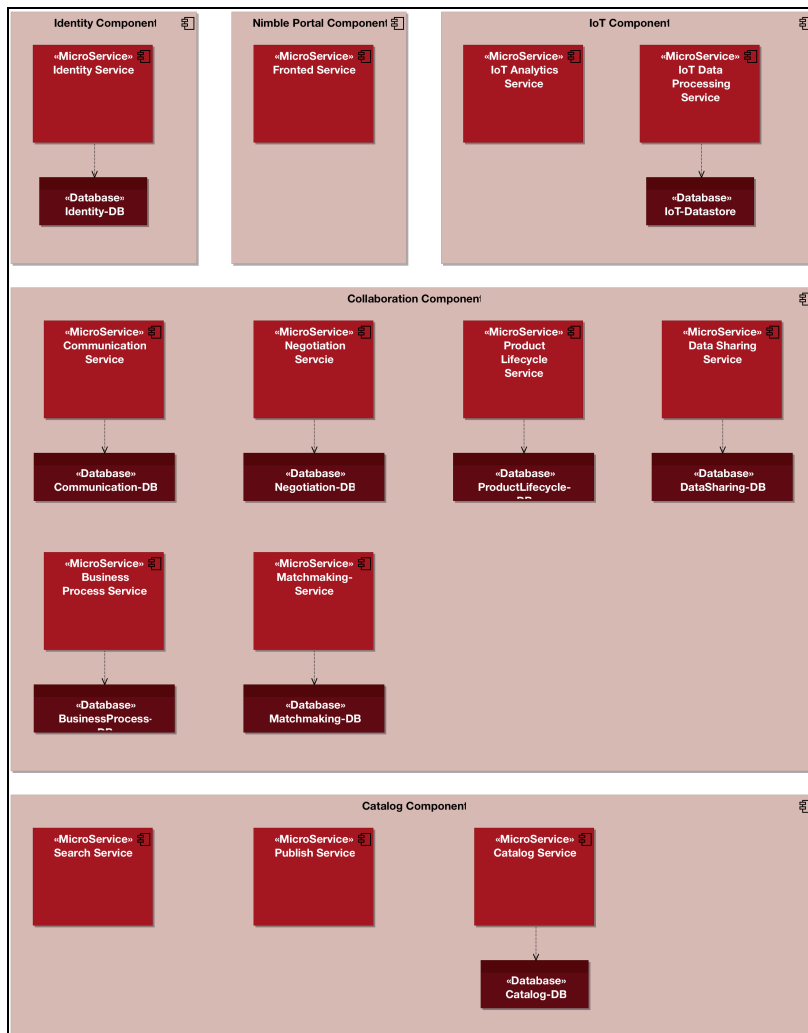


Figure 2: Core components - a microservices view

Code for the open source components can be found in: <https://github.com/nimble-platform>

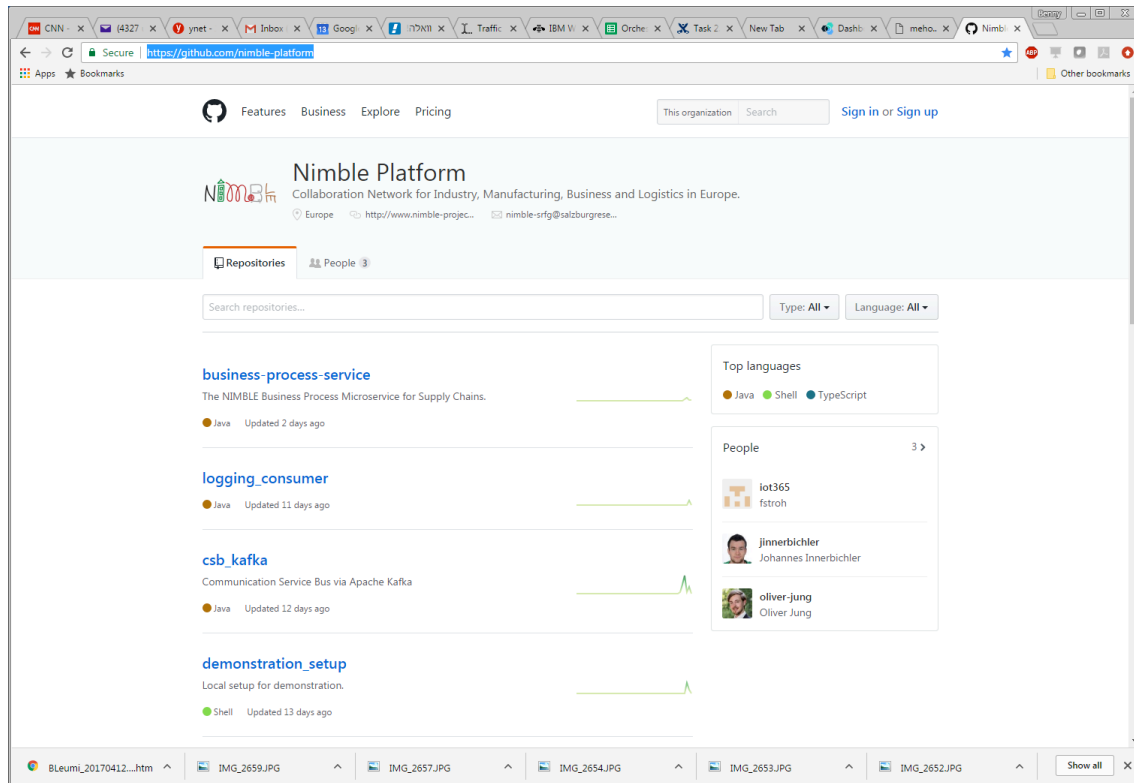


Figure 3: The NIMBLE git repository

3.1 Nimble Portal Component

This component group provides central services for user interactions. The main responsibility is to provide user interfaces and delegate tasks further to specific micro-services.

3.1.1 Frontend Micro-Service

ServiceID: frontend-service

GitHub: <https://github.com/nimble-platform/frontend-service>

This micro-service provides the graphical user interface. Each request from the user is delegated to the target micro-services (e.g. registration requests are delegated to the Identity Micro-service).

Responsibilities:

- Provides GUI (HTML5 / Javascript)
- Central point for user interactions / requests
- Delegates requests to designated micro-services

Inter-Service Communication:

- Identity Micro-service
- Search Micro-service
- Publish Micro-service
- Communication Micro-service

- Negotiation Micro-service
- Matchmaking Micro-service
- IoT Analytics Micro-service

3.2 Identity Component

Identities on the platform are administered by this component group, which plays a vital role in terms of security. Micro-services communicate with the UAA and the Identity Management component of the Microservice infrastructure in order to perform authentication and authorisation on the platform. Identities are defined as entities, which perform certain actions on the platform (i.e. users, companies and autonomous agents) based on their designated roles.

3.2.1 Identity Micro-service

This micro-service administers identities on the platform.

ServiceID: identity-service

GitHub: <https://github.com/nimble-platform/identity-service>

Responsibilities:

- Receives requests for CRUD operations of user entities.
- Receives requests for CRUD operations of company entities.
- Handles logins of users.
- Communicates with the UAA & Identity Management back-end service (i.e. Cloud Foundry UAA)
- Stores user and company related data.

Inter-Service Communication:

- UAA and Identity Management

Back-end Service:

- Cloud Foundry UAA

Data stores:

- *Identity-DB: Stores user and company related data.*

3.3 Catalogue Component

Products and user services on the platform are administrated by Microservices within this component group. With the vast amount of companies' information anticipated in the NIMBLE platform there is a need for users to be able to locate existing information easily. For this reason, NIMBLE contains a catalogue component which holds semantically enhanced service descriptions making it easier to discover services based on various criteria.

Each newly registered catalogue in the platform gets registered with this component as a part of its registration process. An end-user can pose semantic queries from the front-end to look for required information.

3.3.1 Search Micro-service

ServiceID: catalog-search-service

GitHub: <https://github.com/nimble-platform/catalog-search-service>

Search functionalities are provided by this micro-service. It communicates with the Catalogue Micro-service to get information about products and user services. It also communicates with the data stores under the hood to access to the data concerning product / service directly and to the semantic schemas for semantic search purposes.

Responsibilities:

- Handles search requests.
- Pre-process search requests.
- Searches through data stored in the Catalogue-Service.
- Manages caching of search results

Inter-Service Communication:

- Catalogue-Service

Data Stores:

- Apache Marmotta
- PostgreSQL

3.3.2 Publish Micro-service

ServiceID: catalog-publish-service

GitHub: <https://github.com/nimble-platform/catalog-service-srdc>

Provides administration of product and services as well as categories used for their semantic annotation. The publishing component interacts with the data stores to get details about product categories maintained as taxonomies.

Responsibilities:

- Receives requests for retrieval of product category details
- Receives CRUD requests for products and services.
- Pre-process CRUD operations.
- Delegates requests further to Catalogue-Micro-service.
- Resets cache of search micro-service.

Inter-Service Communication:

- Catalogue Micro-service
- Search Micro-service

Data Stores:

- Apache Marmotta
- PostgreSQL

3.3.3 Catalogue Micro-service

ServiceID: catalog-service

GitHub: <https://github.com/nimble-platform/catalog-service-srdc>

Stores products and services persistently by utilising Apache Marmotta (for storing semantic representations of the published items as a whole in a triple store as well as for selective storage of desired fields of the items in a free text-engine). For the time being, it also uses a relational database (i.e. PostgreSQL) for structured representation of items

Responsibilities:

- Stores products and user services persistently.
- Provides search capabilities.
- Provides functionalities for CRUD actions for products and user services.

Data Stores:

- Apache Marmotta
- PostgreSQL

3.4 Collaboration Component

Collaboration between users or companies is handled by micro-services in this group.

3.4.1 Communication Micro-service

ServiceID: communication-service

GitHub: https://github.com/nimble-platform/csb_kafka

Communication with or between entities on the platform is handled by this micro-service.

Responsibilities:

- Handles communication between users (or companies) in the form of messages.
- Handles communication between platform and users (or companies) in the form of notifications.
- Stores communication history persistently.

Data stores:

- **Communication-DB:** Stores historical communication.

3.4.2 Negotiation Micro-service

ServiceID: negotiation-service

Negotiation between companies is augmented by functionalities of this micro-service.

Responsibilities:

- Monitors negotiation between entities.
- Augments the negotiation process.
- Stores historical actions and events.

Data stores:

- **Negotiation-DB:** Stores *historical* negotiation actions and events.

3.4.3 Matchmaking Micro-Service

ServiceID: matchmaking-service

This micro-service provides functionalities for matching companies, which are likely to fulfil each other's requirements.

Responsibilities:

- Collects appropriate data for matchmaking from other micro-services, which includes
 - Historical communication data from Communication Micro-service
 - Historical negotiation data from Negotiation Micro-service
 - Metadata for company from Identity Micro-service
 - Data from products and user services of related company from Search Micro-service
- Performs matchmaking based on collected data.
- Stores matchmaking results for further analysis.

Inter-Service Communication:

- Communication Micro-service
- Negotiation Micro-service
- Identity Micro-service
- Search *Micro-service*

Data stores:

- **Matchmaking-DB:** Stores historical data about matchmaking.

3.4.4 Business Process Micro-service

ServiceID: business-process-service

GitHub: <https://github.com/nimble-platform/business-process-service>

This service provides the definition of communication workflows among multiple supply chain partners and the execution of the designed process through the Camunda Business Process Management (BPM) Platform. During the design process, it interacts with the Search and Matchmaking Micro-services to find the most suitable partner for a particular role in the process. During the execution phase, it interacts with the Communication Micro-service to setup a data sharing channel between the data sender and data receiver and transmit the data through this channel.

Responsibilities:

- Provides capabilities for the definition of business processes
- Stores the designed processes persistently
- Provides execution of the business processes designed
- Setup of data-sharing channels.

Inter-Service Communication:

- Matchmaking Micro-service
- Negotiation Micro-service

- Search Micro-service
- Communication Micro-service

Back-end Service

- Camunda BPM

Data stores:

- **BusinessProcess-DB:** Modelled business models are stored in this database.

3.4.5 Product Lifecycle Micro-service

ServiceID: product-lifecycle-service

This micro-service manages and analyses product lifecycle data. This microservice will be elaborated at later development stages of the platform.

Responsibilities:

- Retrieving and storing lifecycle data.
- Search through lifecycle data.
- Analyse lifecycle data.

Data stores:

- **ProductLifecycle-DB:** Stores product lifecycle data.

3.4.6 Data Sharing Service

ServiceID: data-sharing-service

Policies and rights-management for data sharing is handled by this micro-service. This micro-service does not store any actual data that is being shared. This microservice will be elaborated at later development stages of the platform.

Responsibilities:

- Handling of data privacy issues.
- Managing policies and rights management for data sharing.

Data stores:

- **DataSharing-DB:** Stores metadata necessary for managing data-sharing channels. Data, that is being shared, is not stored in this database.

3.5 IoT Component

Communication with IoT devices and platforms is performed by micro-services in this group.

3.5.1 IoT Data Processing Micro-service

ServiceID: iot-dataprocessing-service

This micro-service receives IoT data, pre-processes it and stores it persistently. This microservice will be elaborated at later development stages of the platform.

Responsibilities:

- Receives IoT data (e.g via MQTT)
- Performs first pre-processing of data.
- Stores data persistently.

Data stores:

- **IoT-Datastore:** Stores IoT data for further analysis.

3.5.2 IoT Analytics Micro-service

ServiceID: iot-analytics-service

Performs analytics on collected IoT data. This microservice will be elaborated at later development stages of the platform.

Responsibilities:

- Performs data analysis on collected IoT data.
- Provides insights for other micro-services.

Inter-Service Communication:

- IoT Data Processing Micro-service

3.6 The cloud run-time

The cloud run-time hosts NIMBLE entities and makes applications available to the external end-users. The NIMBLE platform cloud run-time consists of a customized version of the Cloud Foundry PaaS within IBM BlueMix. The cloud run-time hosts NIMBLE applications as well as the services required for their operation. The run-time provides binding mechanisms for NIMBLE applications to connect to the infrastructure services they require.

3.6.1 The run-time environment:

- The cloud is currently hosted by IBM's cloud, BlueMix in the European region (dedicated NIMBLE space)
- BlueMix Cloud Foundry instance managed with cf-cli (go executable)

4 What is being demonstrated

The intention of this demonstration is to provide a first glimpse of the core capabilities of the platform. The flow of the demonstration is as follows:

1. Register a company and several employees that will use the platform
2. Upload a catalogue of products/services that are semantically annotated so that automated selection of products/variants is possible
3. Perform a search to find a suitable supply chain partner based on uploaded catalogues

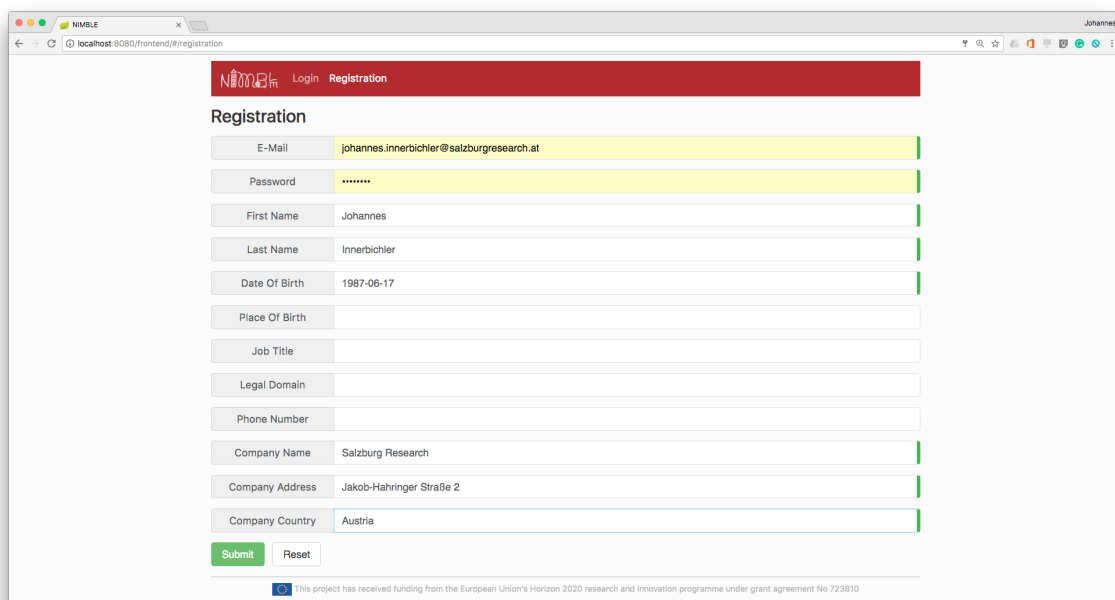
4. Define a simple business process between two firms (i.e. a supplier sending materials to a producer)
5. Execute that simple process between all involved parties (supplier, buyer, logistics-provider)
6. Establish a private information channel between at least two partners where production-related data is exchanged (e.g. monitoring the transport of some goods)
7. Provide a seamless record of all data and information flows that happened between the trading partners, on the platform.

NIMBLE components which comprise the demonstration:

- Front-End: shall be used for registration and invoking catalogue publish and query
- Catalogue component: used for publishing and searching through catalogues.
- Collaboration component: used to establish a business relation between two entities, as well as exchanging information.
- Identity component – used for login, id management and access control.

The combined capabilities of all NIMBLE components are demonstrated. Naturally not all capabilities of all components are demonstrated, but rather the major capabilities and interactions are shown.

1. The full flow of registering a new entity into the platform is demonstrated. This flow includes a user interacting with the front-end, while in the background the NIMBLE security component is invoked.



The screenshot shows a web browser window with the URL `localhost:8080/frontend/#/registration`. The page has a red header with the NIMBLE logo and 'Login Registration' links. The main content area is titled 'Registration' and contains a form with the following fields: E-Mail (johannes.innerbichler@salzburgresearch.at), Password (masked with dots), First Name (Johannes), Last Name (Innerbichler), Date Of Birth (1987-06-17), Place Of Birth, Job Title, Legal Domain, Phone Number, Company Name (Salzburg Research), Company Address (Jakob-Hahninger Straße 2), and Company Country (Austria). At the bottom of the form are 'Submit' and 'Reset' buttons. A footer note states: 'This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723810'.

Figure 4: Registration front-end of the NIMBLE platform

2. The full flow of publishing a new catalogue is demonstrated. This entails interacting with the catalogue publishing component for downloading an Excel-based product category template and storing the information provided through the template in its back-

end data store. As only the backend services are available but not the user interfaces at the time of writing this deliverable, envisioned user interfaces are demonstrated as mock-ups.

Register your products to NIMBLE

Choose the category of your product

Construction Technology >> Tile,Panel >> Natural Stone >> Mosaic (eCI@ss)☐ Select

Construction Technology >> Tile,Panel >> Natural Stone (eCI@ss)☐ Select

Construction Technology >> Tile,Panel >> Natural Stone >> Tile for fireplace (eCI@ss)☐ Select

Construction Technology >> Tile,Panel >> Natural Stone >> Floor tile (eCI@ss)☐ Select

Not relevant? Have a look at all categories

Figure 5: Product category selection

	A	B	C	D	E	F	G	H	I	J	K
1	Property Name	Name	Description	Rotation speed	Voltage	FLA_460_V	FLA_460_V	Product type	GTIN	Manufacturer name	name of supplier
2	Property Data Type	STRING	STRING	STRING	STRING	REAL_NUMBE	REAL_NUMBE	STRING_TRANS	STRING	STRING	STRING
3	Property Unit										
4		20N Ser	Elektrim Perf	1715	208-230/4	1.4	12.9	Electric motor		Elektrim Motors	Elektrim Motors
5		20N Ser	Elektrim Perf	3540	208-230/4	2.1	20	Electric motor		Elektrim Motors	Elektrim Motors
6		20N Ser	Elektrim Perf	3525	208-230/4	2.7	25.4	Electric motor		Elektrim Motors	Elektrim Motors
7											
8											
9											
10											
11											
12											

Information

Product Properties

Property Details

Allowed Values for ...

Figure 6: An example Excel-based product category template

3.
- The full flow of searching through the catalogues for a desired entity will be demonstrated. This entails interacting with the catalogue searching component which will invoke the necessary queries on its back-end data store.

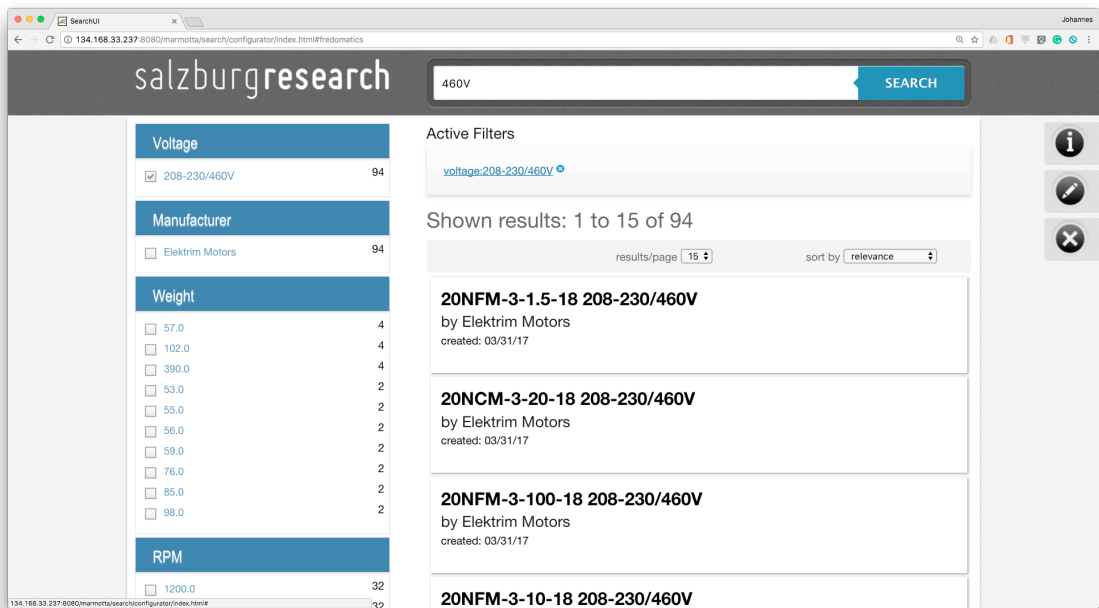


Figure 7: UI of Apache Marmotta back-end service for search

4. The full flow of establishing a simple business process is demonstrated. This entails designing and executing the business process, and have it pass through the platform.

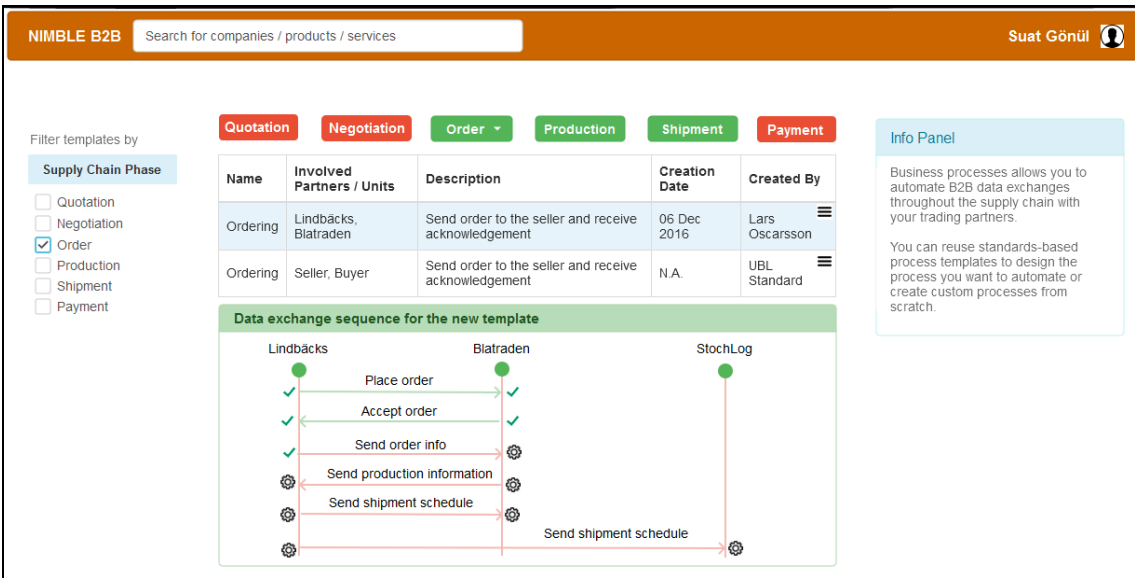


Figure 8: Design of a business process by using the available platform templates

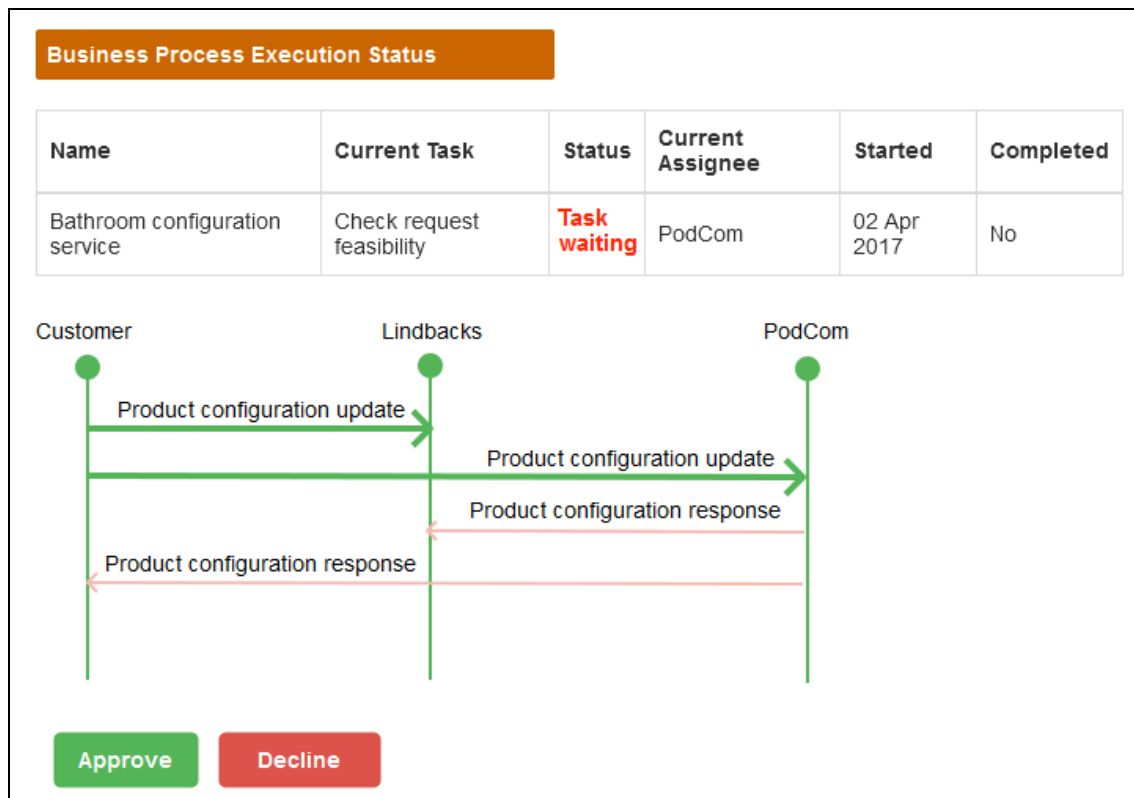


Figure 9: An intermediate step of execution of a business process where the user is expected to approve or decline the incoming configuration update request

5. The full flow of creating a private data channel among partners will be demonstrated. That entails invoking the communication bus component via the front-end to create a private topic for the partners to use, and exercising the communication component by having the partners exchange information through the created topic.
6. The seamless capture of all platform traffic will be demonstrated as well, by having all information exchanged between partners through the platform diverted to the platform logging facility.

5 Integrated platform installation & configuration

The ultimate long-term deployment target for the NIMBLE platform is a public cloud. As a step along the way, to ease and expedite the development of different NIMBLE components and deployments thereof, we have set up a private cloud deployment, making use of readily available open source components. Hand in hand with the deployment and integration of the different components within the local cloud environment we are working on deploying and migrating all components to be deployed on the cloud.

5.1 Public Cloud deployment

The public cloud chosen for the deployment of the NIMBLE platform is IBM's BlueMix. BlueMix is composed of three types of cloud offerings, namely IaaS (VMs), Cloud Foundry

applications, and a container service. The NIMBLE platform will be deployed over the PaaS portions of the cloud, namely as Cloud Foundry applications and services, and within the container service, using Docker containers.

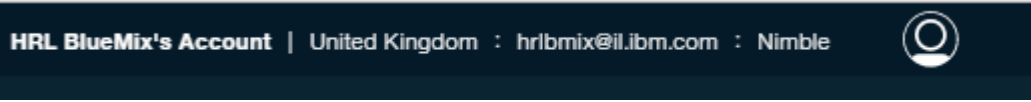


Figure 10: NIMBLE BlueMix account

For the deployment of the NIMBLE platform we have created under the HRL BlueMix account a BlueMix organization named hrlbmix@il.ibm.com, and within that account we created a space for the NIMBLE platform called Nimble (see Figure 10). The NIMBLE space is hosted in the UK region of the cloud. In BlueMix, organizations are created to enable collaboration among team members and to facilitate the logical grouping of project resources. In particular, the NIMBLE platform is supported by a dedicated Nimble space, which is used to group a set of related applications, services, and team members.

A snapshot taken from the BlueMix dashboard of the current state of the NIMBLE space can be seen in the following figures. The deployed Cloud Foundry applications can be seen in Figure 11; instantiated and deployed Cloud Foundry services being used by applications can be seen in Figure 13; while NIMBLE Docker containers deployed in the cloud can be seen in Figure 12.

A screenshot of the IBM Bluemix Apps dashboard. The top navigation bar shows "HRL BlueMix's Account | United Kingdom : hrlbmix@il.ibm.com : Nimble". Below the navigation bar, there's a section for "All Apps (16)" with a "Create App" button. The main content area is titled "Cloud Foundry Apps 20,750 GB/128 GB Used" and contains a table of applications. The table has columns: NAME, ROUTE, MEMORY (MB), INSTANCES, RUNNING, STATE, and ACTIONS. The applications listed are: config-server, csb_service, gateway-proxy, GetStartedJava, hystrix-dashboard, IoT SRDC Application, java-hello-world, JTest, service-discovery, sqhweb, testsql, and testTextileIoT. All applications are in a "Running" state.

NAME	ROUTE	MEMORY (MB)	INSTANCES	RUNNING	STATE	ACTIONS
config-server	nimble-config-server.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
csb_service	csb-service-tribeless-supertemptation.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
gateway-proxy	nimble-gateway-proxy.eu-gb.mybluemix.net	1024	1	1	Running	Refresh Stop Delete
GetStartedJava	getstartedjava-impressive-flagella.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
hystrix-dashboard	nimble-hystrix-dashboard.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
IoT SRDC Application	iot-srdc-application.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
java-hello-world	java-hello-world.eu-gb.mybluemix.net	1024	1	1	Running	Refresh Stop Delete
JTest	jtest.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
service-discovery	nimble-service-discovery.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete
sqhweb	sqhweb.eu-gb.mybluemix.net	1024	1	1	Running	Refresh Stop Delete
testsql	testsql.eu-gb.mybluemix.net	1024	1	1	Running	Refresh Stop Delete
testTextileIoT	testTextileIoT.eu-gb.mybluemix.net	512	1	1	Running	Refresh Stop Delete

Figure 11: CF applications running in the NIMBLE space

Containers 1 GB/1 GB 1/2 Public IPs Requested 1 Used					
NAME	INSTANCES	IMAGE	CREATED	STATUS	ACTIONS
infra_config-server_1	1	semantic_mediator_container/nimble-con	3/31/2017 8:19 AM	Running	↺ ⋮
infra_service-discovery_1	1	semantic_mediator_container/nimble-ser	3/31/2017 8:19 AM	Running	↺ ⋮
marmotta-backend	1	semantic_mediator_container/marmotta-t	3/30/2017 4:04 PM	Running	↺ ⋮
marmotta-db	1	semantic_mediator_container/postgre:late	3/30/2017 4:01 PM	Running	↺ ⋮

Figure 12: Containers running in the NIMBLE space

IBM Bluemix Apps			
Services 62/640 Used			
NAME	SERVICE OFFERING	PLAN	ACTIONS
AlchemyAPI-fz		free	⋮
API Connect-c1	API Connect	Essentials	⋮
availability-monitoring-auto	Availability Monitoring	Lite	⋮
Cloudant NoSQL DB-1f	Cloudant NoSQL DB	Lite	⋮
Compose for PostgreSQL-de	Compose for PostgreSQL	Standard	⋮
Compose for PostgreSQL-o3	Compose for PostgreSQL	Standard	⋮
config-service	user-provided	n/a	⋮
csb_message_hub	Message Hub	Standard	⋮
discovery-service	user-provided	n/a	⋮
IoT SRDC Application-cloudantNoSQLDB	Cloudant NoSQL DB	Lite	⋮
IoT SRDC Application-iotf-service	Internet of Things Platform	Standard	⋮
testTextileIoT-cloudantNoSQLDB	Cloudant NoSQL DB	Lite	⋮
testTextileIoT-iotf-service	Internet of Things Platform	Standard	⋮
Twilio-4i	user-provided	n/a	⋮

Figure 13: BlueMix services running in the Nimble space

5.2 Private Cloud deployment

General infrastructure components and Microservices can be deployed to Docker machines³ hosted on-premises. This enables individual services to be hosted inside a private network, without having each service publicly accessible on the Internet. Figure 14 depicts the deployment diagram for hosting the infrastructure locally using containerized applications. Single components are pushed to Dockerhub⁴ and can then be downloaded to hosting servers. This leads to a small set of software dependencies installed on the hosting operating system (i.e. Docker and Docker Compose). The infrastructure was deployed in the Salzburg Research premises on a single server (Thinkcentre 4x2GHz 8GB RAM).

³ <https://docs.docker.com/machine/>

⁴ <https://hub.docker.com/u/nimbleplatform/>

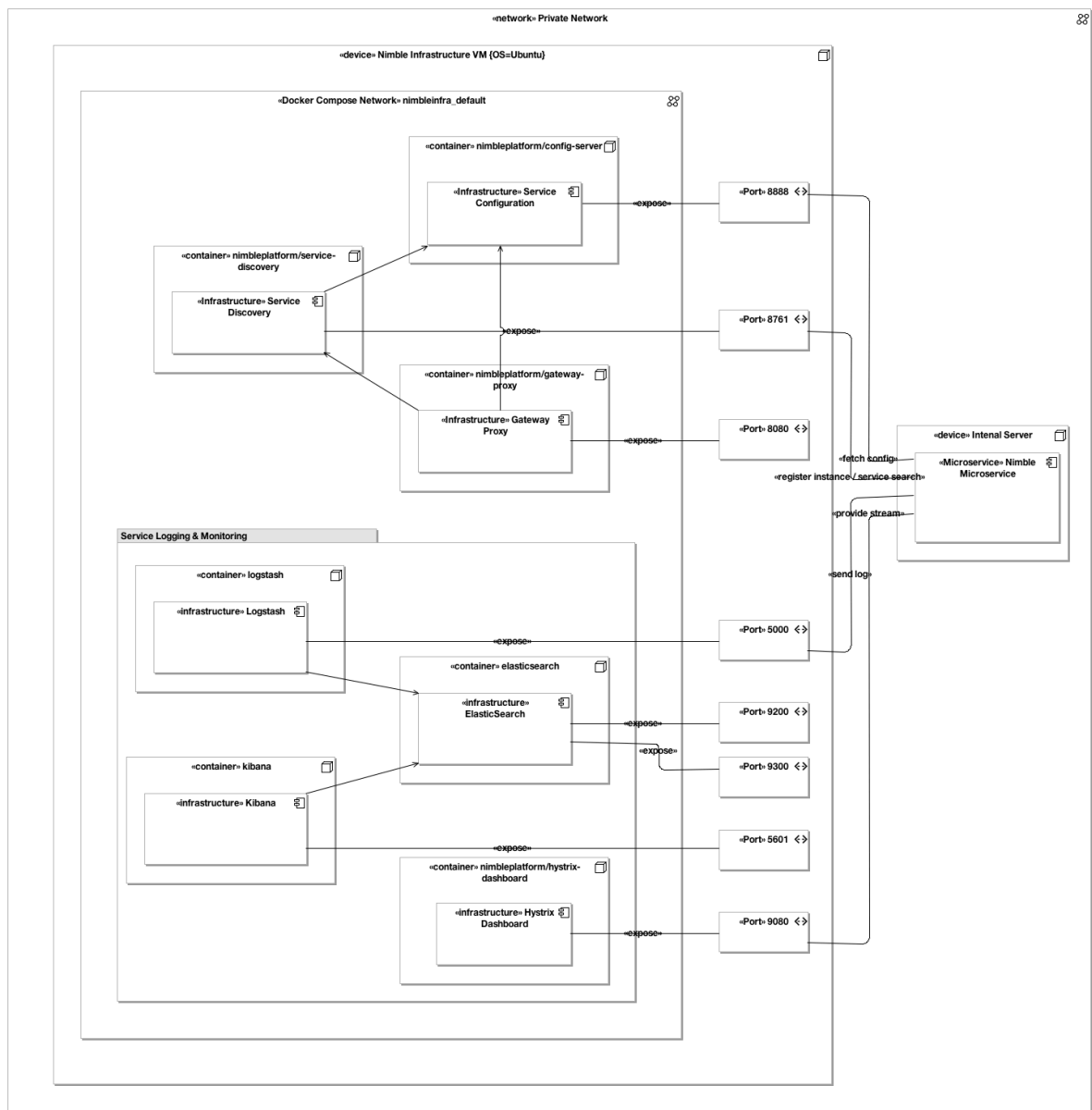


Figure 14: Deployment Diagram for hosting NIMBLE on premise

6 Applied Technologies

The following table summarizes applied technologies and frameworks in NIMBLE:

Table 2: Applied technologies in NIMBLE

Technology	Description	Usage
Spring Boot	Spring Boot is a framework for building web applications. It is built on top of the Spring Framework and follows a zero-configuration principle.	The major set of microservices are build using Spring Boot as an application framework.
Spring Cloud	Functionalities for building and integrating microservices are provided by Spring Cloud. It mainly aggregates components of the Netflix Open Source Software (Netflix OSS) project and makes them easily be integrated with Spring Boot applications.	Components of the underlying microservice infrastructure are heavily using modules from Spring Cloud (e.g. Service Discovery, Configuration Server and Gateway Proxy).
Spring Cloud Security	Standardized security mechanisms are implemented using Spring Cloud Security. It provides out-of-the-box integration of security modules to Spring Cloud applications.	Authentication and authorization between microservices are realized by using Spring Cloud Security, which supports OAuth2 and OpenID Connect and communicates with the authentication server (i.e. Cloud Foundry UAA).
ELK Stack	Logs can be streamed to Logstash, which stores them persistently in Elastic Search. visualizations are done using Kibana, hence the ELK stack.	The ELK stack is used to aggregate log output of distributed microservices in order to centrally perform analysis of generated log output.
Cloud Foundry UAA	The Cloud Foundry User Account and Authentication (UAA) is a multi tenant identity management service, available as a stand alone OAuth2 server issuing tokens for clients.	Cloud Foundry UAA acts as identity and authentication server issuing OpenID Connect tokens.
Camunda BPM	Camunda BPM is an open source platform for business process management.	Camunda BPM is used for the definition and execution of business processes (e.g. supply chain process).
Apache Marmotta	Apache Marmotta is an open implementation of a linked data platform.	Apache Marmotta will be mainly used to store catalog data and perform product-search queries.
Apache Solr	Apache Solr is a free-text indexing tool providing advanced search and navigation capabilities on the indexed data.	Apache Marmotta uses Apache Solr for its semantic search cores composed semantic features of indexed items.
Docker	Docker is an open-source solution for application deployment, consisting of prebuilt images running inside a container.	Docker will be used for intermediate development releases and on-premises deployment.
PostgreSQL	PostgreSQL is an open-source database system for object-relational data.	PostgreSQL will mainly be used as a database technology, in order to have a homogeneous setup.
Apache Kafka	Open Source messaging infrastructure	Mainly used for private communication among components and entities.