SPECIAL

Scalable Policy-awarE Linked Data arChitecture for prIvacy, trAnsparency and compLiance

Deliverable 3.1

Initial setup of policy-aware Linked Data architecture and engine

Document version: 1.0

SPECIAL DELIVERABLE

Name, title and organisation of the scientific representative of the project's coordinator:

Mr Philippe Rohou t: +33 4 97 15 53 06 f: +33 4 92 38 78 22 e: philippe.rohou@ercim.eu

GEIE ERCIM, 2004, route des Lucioles, Sophia Antipolis, 06410 Biot, France
Project website address: http://www.specialprivacy.eu/

Project	
Grant Agreement number	731601
Project acronym:	SPECIAL
Project title:	Scalable Policy-awarE Linked Data arChitecture for prIvacy, trAnsparency and compLiance
Funding Scheme:	Research & Innovation Action (RIA)
Date of latest version of DoW against which the assessment will be made:	17/10/2016
Document	
Period covered:	M1-M6
Deliverable number:	D3.1
Deliverable title	Initial setup of policy-aware Linked Data architecture and engine
Contractual Date of Delivery:	30-06-2017
Actual Date of Delivery:	30-06-2017
Editor (s):	Uroš Milošević (TF)
Author (s):	Bert Van Nuffelen (TF)
Reviewer (s):	Rigo Wenning (ERCIM), Philip Raschke (TUB), Sabrina Kirrane (WU)
Participant(s):	ERCIM, WU, CeRICT, TUB, ULD, DTAG, TR, PROX
Work package no.:	WP3
Work package title:	Big Data Policy Engine
Work package leader:	TF
Distribution:	PU
Version/Revision:	V1.0
Draft/Final:	Final
Total number of pages (including cover):	14

Disclaimer

This document contains description of the SPECIAL project work and findings.

The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any responsibility for actions that might occur as a result of using its content.

This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the SPECIAL consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 28 Member States of the Union. It is based on the European Communities and the Member States cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors (http://europa.eu/).

SPECIAL has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731601.

Table of Contents

1 Introduction

This document accompanies the deployment of a development environment based on the Big Data Europe (BDE) platform with extensions. SPECIAL will extend the BDE architecture and engine with policy, transparency and compliance mechanisms. By doing so, it will leverage the existing Big Data processing capabilities, thus affording SPECIAL the opportunity to focus specifically on policies, transparency and compliance and the robustness of the proposed components.

This document reports on the initial deployment of the SPECIAL platform, which corresponds to the Big Data Europe platform. In doing so, the SPECIAL consortium can familiarise itself with the environment and the way of working it embodies.

2 BDE Platform

2.1 Overview

Big Data Europe is a European funded project undertaking foundational work in cross community big data management by building innovative products and services based on semantically interoperable, large-scale data assets. For that, it created the BDE's Integrator Platform[1] (BDI): a platform to get started with common big data technologies with a minimal amount of effort. This platform forms the foundational basis on top of which we will build the SPECIAL platform.

BDI offers a set of basic building blocks which facilitate the creation of applications, also called data processing pipelines. Generic building blocks, including a graphical UI to manage the pipelines, and an interaction monitor to observe the operational behavior, interact with core big data processing components such as Apache Spark[2], Hadoop HDFS[3] and Apache Flink[4]. As the data processing landscape is very active and fast moving, the BDI platform is designed for flexibility. It therefore easily supports a wide range of data management challenges and is flexible regarding the used data processing technologies.

Technology-wise, BDI is an Open Source[5] platform based on Docker[6], currently a common virtualisation solution. It can work on a local development machine, or on hundreds of nodes connected in a swarm. The platform can be run in house, or can be hosted by vendors like Amazon Web Services or Microsoft Azure. The base Docker platform is enriched with a layer of supporting services, helping setup and maintenance.

2.2 Architecture

The architecture is illustrated in Figure 1 - BDE Platform architecture. We can distinguish three layers in BDI: hardware, resource manager and applications. This layering offers a separation of concerns without sacrificing flexibility nor execution performance.

[1] https://www.big-data-europe.eu/platform/

[2] https://spark.apache.org/

[3] https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

[4] https://flink.apache.org/

[5] https://github.com/big-data-europe

[6] https://www.docker.com/

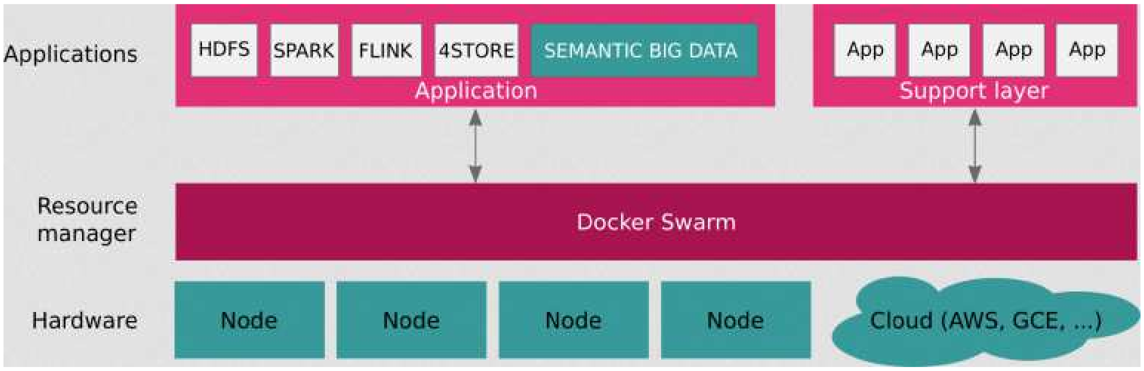


Figure 1 - BDE Platform architecture

The hardware layer offers the computational elements on which the application solution is being executed. They are CPUs, memory, disks and network. On physical hardware, these are organised in well-defined groups, called machines, for instance, a laptop. The BDI platform abstracts whether the hardware is physically available or not. It may also be a cluster of virtual machines.

Within application solution design, typically some effort is needed to layout the application setup on the available hardware configuration. This is a laborious effort, and often does not consume the available resources efficiently. Therefore, a more global resource view is preferred, because the usage of resources can then be allocated dynamically on a needs basis by a resource manager. BDI has chosen for Docker Swarm as the resource manager.

Docker Swarm is a clustering and scheduling tool that acts directly on top of the hardware layer. As earlier mentioned, hardware nodes can be servers available in a data-center or hosted by a cloud service provider.

It turns a pool of hardware nodes, configured as Docker hosts, into a single, virtual Docker host. Because Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts. The Docker Swarm offers most of the features required by the platform i.e. scalability, interlinking the containers, networking between different containers, resource management between containers, load balancing, fault tolerance, failure recovery and log based monitoring.

On top of Swarm, applications can be deployed easily through Docker as a single container, or through Docker Compose as a collection (cluster) of communicating containers. An application running on the BDI platform can be viewed as a pipeline consisting of multiple components, which are wired together to solve a specific Big Data problem. Each component in the pipeline is available and packaged as a Docker container.

To encourage technology reuse and streamline the development of a component, the BDI platform provides base Docker images which are preconfigured and integrate well with the generic supporting components.

The BDI architecture is designed around the need for easing (and automating) the roll-out of an application. The importance of this activity cannot be underestimated, as the majority of the big data processing tasks go through a long initial phase of experimentation in which one searches for the best approach to achieve the application’s objective. Therefore, the ability to go from the developer workbench to a roll-out has to be quick, but trustworthy. This quality is valuable for the SPECIAL project.

Docker service virtualisation brings developers and operational teams together. Already during the development process developers consider and integrate operational aspects such as indicating the required open ports and the dependencies between components, making the step towards a runnable, reproducible solution much smaller. In addition, Docker facilitates the flexibility and reuse of components. By correctly applying the layered design of Docker, one can build upon another's work and vice versa. A base Docker image offers a template implementation for a specific technology. This template can easily be extended by a developer with his/her own, custom, implementation to solve a problem. Should some base technology not be supported yet, a new base image can be built.

2.3 Supported Components

To encourage technology reuse and facilitate the development of a component, the BDI provides support for a set of components, mainly by providing base Docker images. This selection of technologies is, however, by no means fixed. In the fast-moving space of Big Data, new technologies rise whilst others fall out of grace. The BDI environment is designed with this dynamicity in mind. Current supported components are listed below and can also be found online[7].

The below listed components and the core BDI generic components are capable of handling two of the 'Vs of big data' challenges – velocity and volume. Other extensions, in particular the semantification (see section 2.4), address other challenges.

Computational frameworks

- Flink, Spark, Storm

Data storage

- Hadoop, Hue HDFS File Browser, Cassandra, CouchDB, HBase, Hive, MongoDB, Redis, Virtuoso

Data acquisition

- Flume

Message passing

- Kafka, Rabbit MQ

Search engines

- Elasticsearch, Solr

Distributed Key/Value Stores

- Zookeeper

[7] https://www.big-data-europe.eu/bdi-components/

Semantic components

- DEEREDCAT, FOX, GeoTriples, Limes, Silk, SEMAGROW engine, Sextant, Strabon, UnifiedViews

2.4 Semantic interoperability

A common challenge in data processing is semantic interoperability. BDI offers additional components that address this challenge. As SPECIAL wants to address policy management with a strong semantic foundation, these extensions are likely to play a key role. The Big Data Europe project's effort in this area is twofold and includes:

1. a Semantic Data Lake, and
2. a Semantic Analytics Stack.

2.4.1 Ontario, the BDI Semantic Data Lake

Variety (another Big-Data 'V') in Big Data is usually less adequately addressed in Big Data frameworks due to its inherent complexity. To address this challenge better BDE involves Linked Data approaches for data representation and querying. Linked Data is a proven approach to deal with all the facets of data integration of heterogeneous collections of data sources. From this the concept of the Semantic Data Lake emerges: a range of processing tools for the semantic web and knowledge graphs in combination with Big Data processing and storage tools making each data item in the Semantic Data Lake accessible under machine processable semantics.

The BDI Semantic Data Lake is called Ontario[8]. It extends a Data Lake with a Semantics. A Data Lake requires that the data remains in its raw format and its original location in the lake. This enables processing to take place over that data, e.g., text mining, image processing or machine learning. The Data Lake only knows the schema on read, meaning that a schema is formed describing the portion of data needed for a particular computation. The Data Lake can call for an extract, transform, load (ETL) procedure, shifting the *transform* to the end of the overall process. The data is extracted from the source and loaded to the Data Lake, to be transformed during a certain computation.

A Semantic Data Lake, enables that each contributed data source is accessible through the common semantic layer. Figure 2 - Semantic Layer on top of the Data Lake illustrates the approach for 3 kinds of data: on the left *Sem Data* denotes data which is available as Linked Data, in the middle (*Sem Annotated Data)* is data annotated with a Linked Data interpretation, and thirdly there is any other form of data (denoted with *Non Sem Data*). For the latter a Semantic lifting has to happen by adding mappings to the Data Lake.

[8] The BigDataEurope Platform – Supporting the Variety Dimension of Big Data, 17th International Conference on Web Engineering (ICWE 2017), To be published, https://www.researchgate.net/publication/312597089_The_BigDataEurope_Platform_-_Supporting_the_Variety_Dimension_of_Big_Data

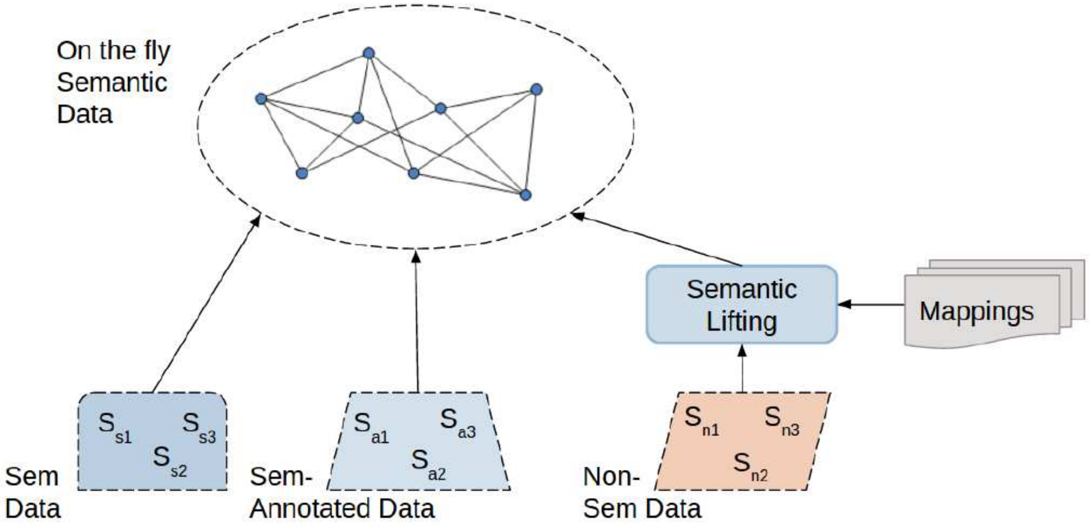


Figure 2 - Semantic Layer on top of the Data Lake

This approach satisfies the objectives of a Data Lake by

1. preventing any sort of preprocessing that physically transforms the original data into a unified format, and
2. enabling the processing of the data in its diversity.

At the same time a uniform data access layer is provided to the data. This does not imply a global unique data model. A unique data model can be achieved, but it is no requirement. Moreover the Semantic Data Lake can support multiple data models at once, allowing the data in the best possible way being accessed by the users.

2.4.2 SANSA-Stack, the BDI's Semantic Analytics Stack

The SANSA-Stack, or Semantic Analytics Stack[9] combines distributed analytics and semantic technologies into a scalable analytics stack which allows RDF data processing. The reasoning behind such as stack is that performing analytics is better done with regard to the semantics of the data and not the raw data representation. The data scientist then explores the data in a semantic way instead of combining several raw data items in a variety of formats. This library can process massive RDF data at scale and perform learning and prediction for the data. It is also useful for large triple stores. This is in addition to answering queries from the data. Although BDI supports both Spark and Flink, SPECIAL's focus is Spark at the moment.

Figure 3 - Architecture of the SANSA-Stack depicts the components of the SANSA-Stack, where an RDF layer offers a basic API to read and write native RDF data. On top are the Querying and Inference layers.

[9] http://sansa-stack.net/

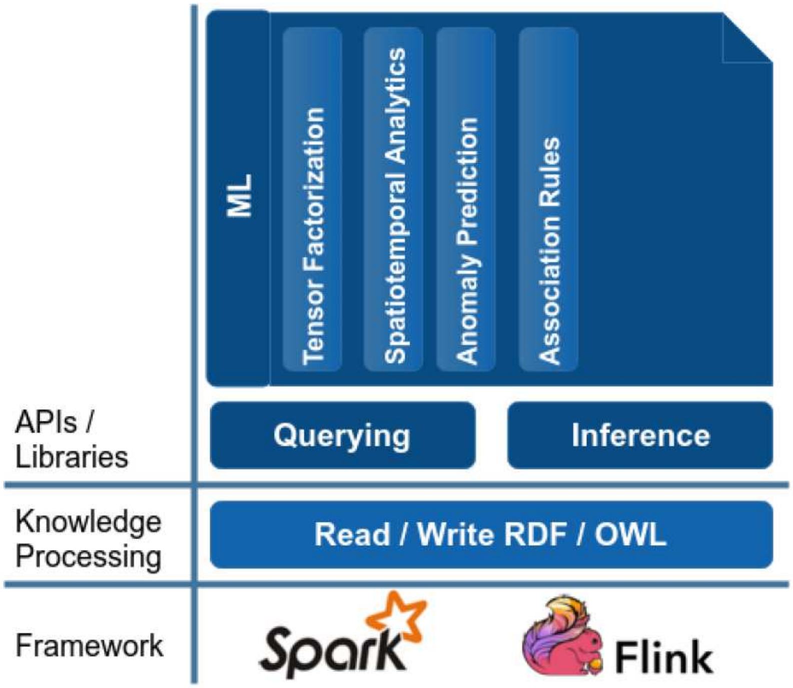


Figure 3 - Architecture of the SANSA-Stack[10]

[10] http://sansa-stack.net/user-guide/

3 SPECIAL Platform

The SPECIAL platform will be an extension of the BDI platform. BDI does not impose any restrictions to the data processing components with respect to the data that is being processed, how it is processed and how it is being disseminated. Within SPECIAL the work is to extend and adapt the BDI platform and some components so that the restrictions and enforced abilities by the EU General Data Protection Regulation (GDPR) are supported out-of-the box. This means that the components need to be hardened against unwanted access and information discovery to protect the rights of the data subjects whose data is being processed.

Additionally, SPECIAL tries to leverage existing policy information into the data flow, thus recording environmental information at collection time with the information. This is more constraint than the semantic lifting of arbitrary data in the data lake. SPECIAL will therefore not only develop the semantic lifting further, but also develop ways how to register, augment and secure semantically lifted data.

Figure 4 – Initial SPECIAL Architecture presents a first outlook of the future SPECIAL platform architecture. The light blue boxes are currently offered by BDI. The SPECIAL hardening layer, in green, depicts the adaptions and extensions to the BDI platform to make the data processing secure. The Policy Manager, in yellow, is a component designed by SPECIAL to handle all aspects of data policies.

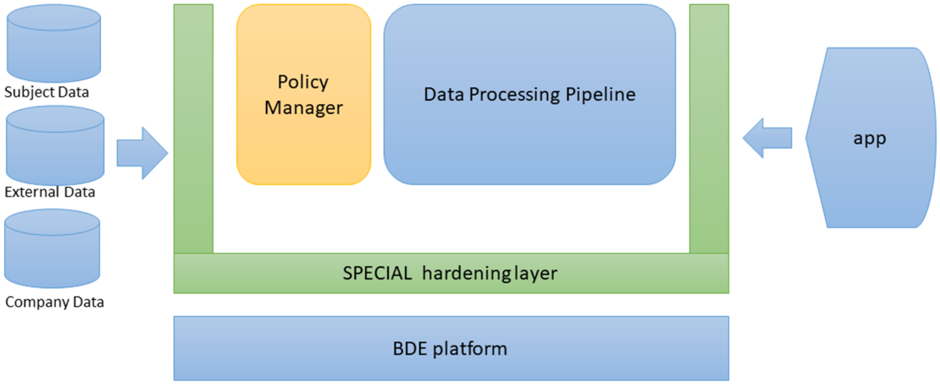


Figure 4 – Initial SPECIAL Architecture

4 Initial deployment of the SPECIAL platform

The initial deployment of the SPECIAL platform is the deployment of the BDI platform on the SPECIAL projects' shared environment. This allows partners to familiarise themselves with the core aspects of the platform.

4.1 Deployment Environment

The shared environment is provided by partner T-Labs in the form of an OpenStack[11] environment. OpenStack is a software platform for cloud computing. It allows Virtual Machines to be created on top of a physical hardware.

To instantiate the BDI platform, a Terraform configuration has been created. Terraform[12] is a tool for building, changing and versioning infrastructures based on a configuration (code) which can be stored in a source control system. It is cloud vendor neutral and most cloud software systems support it. It has as main advantage that the setup can be reproduced automatically with limited human intervention and shorted deployment manuals. In general, improving the automation from source code development to deployment in a production setting using descriptive approaches will facilitate the usage and impact of the applications that are being build.

The SPECIAL BDI Terraform description is found at the project's Github page[13].

4.2 SPECIAL resource management layer

The deployed resource management layer requires four nodes:

1. A large instance Manager
2. A large instance Worker
3. A tiny instance Bastion
4. A small instance Consul

The resource description and their interaction are depicted in Figure 5 - Initial SPECIAL Resource Manager configuration.

[11] https://www.openstack.org/

[12] https://www.terraform.io/

[13] https://github.com/specialprivacy

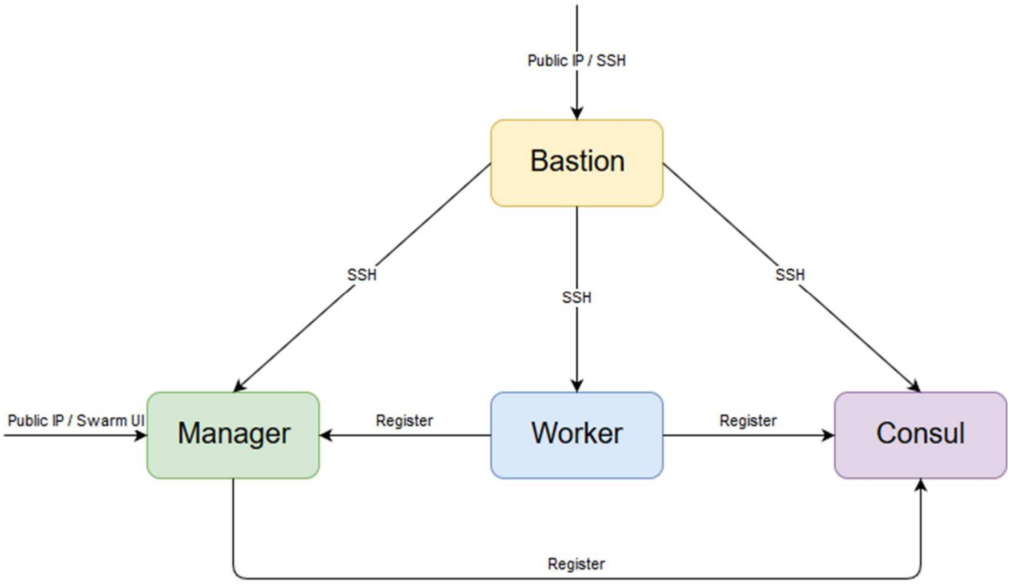


Figure 5 - Initial SPECIAL Resource Manager configuration

The **Worker** is a Docker Swarm agent. Data processing components will run on them. Depending on the application needs additional worker nodes can be added. This can be done dynamically.

The **Manager** is the Docker Swarm[14] manager. It manages the workload of all workers. This virtual machine is also an access point for the Swarm UI application.[15] Swarm UI is a generic BDI component supporting the administration of the Swarm. When the setup is primarily used for experimentation and evaluation purposes, a manager node can also get the worker role assigned to increase the resource usage efficiency. This choice has been made for the initial SPECIAL platform.

The **Bastion** is the access point for SSH access through the Internet. Its security measures help prevent malicious attacks. The bastion allows the administrator to connect to the Swarm directly and check information about managing the environment.

Consul is a service discovery tool, which is needed for Docker Swarm. All nodes need to register with the discovery tool, which helps discover services within the infrastructure.

4.3 SPECIAL application layer

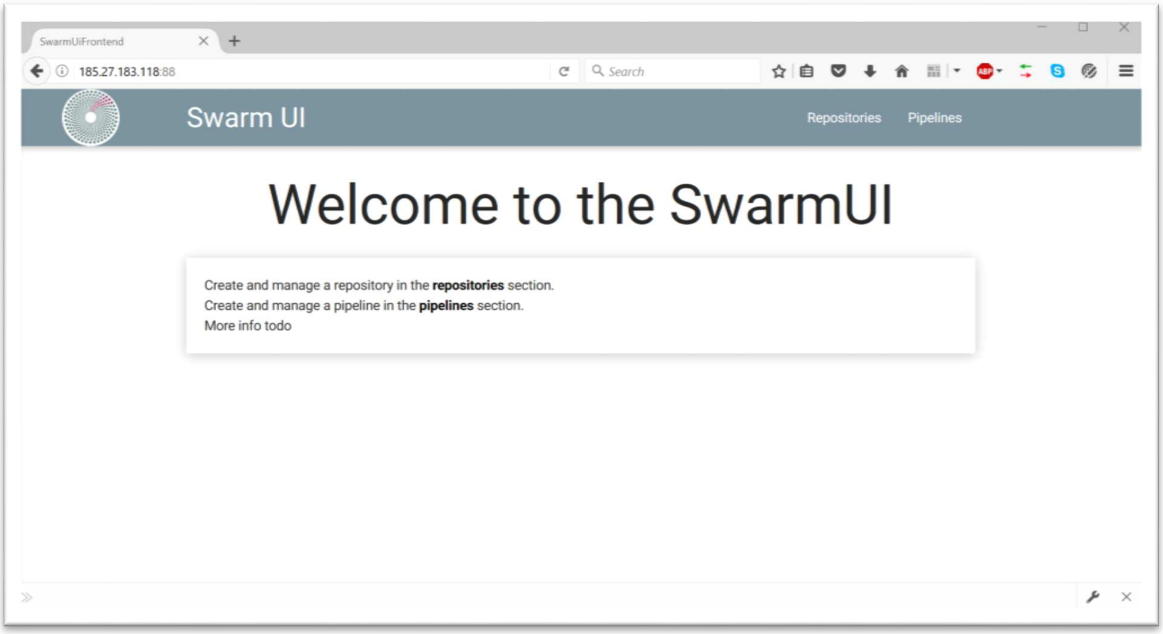
The Swarm UI facilitates deployment of pipelines on the swarm cluster with a limited amount of manual intervention. The following set of screenshots illustrates the Swarm UI functionality and its usage.

From the welcome screen, see Figure 6 - Swarm UI welcome screen, a user can explore the available repositories and launched pipelines. When starting from scratch, a user has to register first a repository which contains the docker-compose file of the pipeline. Afterwards, the

[14] https://docks.docker.com/engine/swarm/

[15] https://github.com/big-data-europe/app-swarm-ui

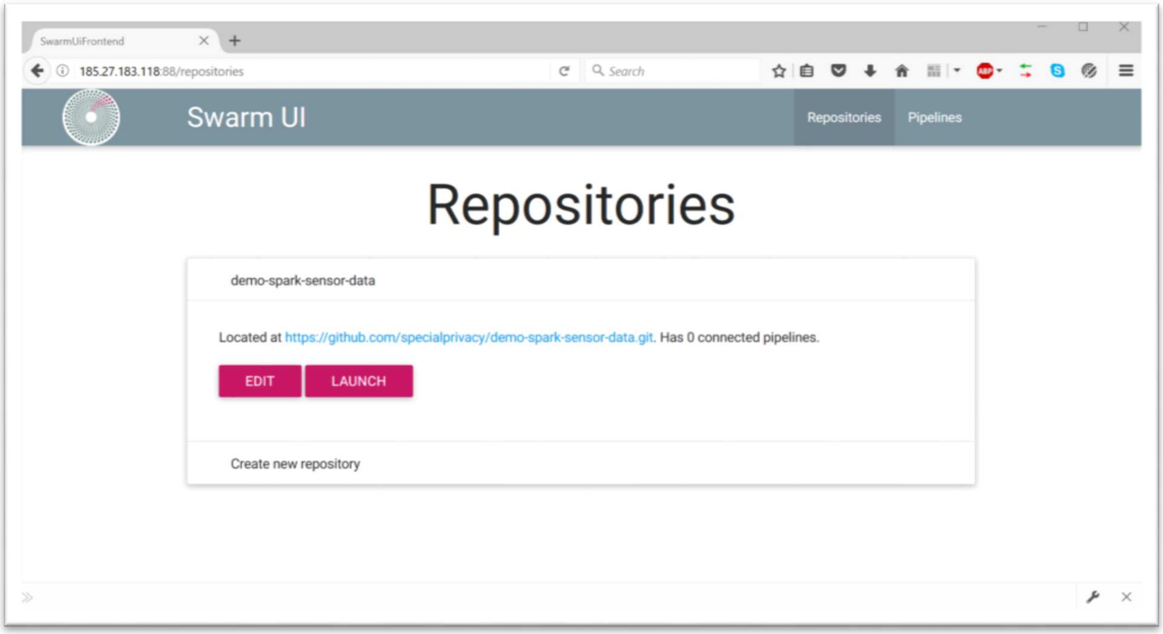
repository can be launched (See Figure 7 - Swarm UI repository overview). Only then the user can see the launched pipelines in the pipeline section. The user can then start and manage the pipelines, for instance dynamically increase the number of running instances for a component, or stop it. For demonstration purposes a demo Spark application is preconfigured. Figure 8 - Swarm UI pipeline management shows the management UI for the demo application.



Welcome to the SwarmUI

Create and manage a repository in the **repositories** section.
Create and manage a pipeline in the **pipelines** section.
More info todo

Figure 6 - Swarm UI welcome screen



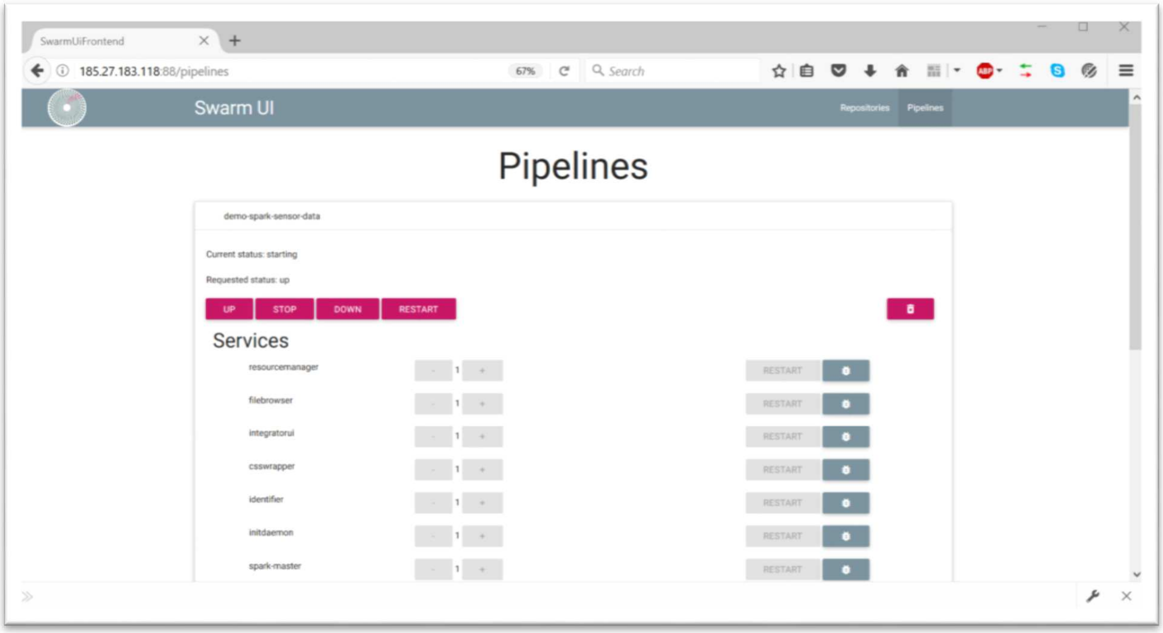
Repositories

demo-spark-sensor-data

Located at https://github.com/specialprivacy/demo-spark-sensor-data.git. Has 0 connected pipelines.

Create new repository

Figure 7 - Swarm UI repository overview



Pipelines

demo-spark-sensor-data

Current status: starting

Requested status: up

Services

resourcemanager	1
filebrowser	1
integratorui	1
csswrapper	1
identifier	1
initdaemon	1
spark-master	1

Figure 8 - Swarm UI pipeline management

5 Next Steps and Conclusion

Now that the initial version of the SPECIAL platform has been deployed, the work to include the security, authorisation and policy management components and adaptations has commenced. This setup allows the early evaluation of the developments in:

- T3.2- Scalable secure data, policy and provenance/events representation
- T3.3- Authentication and authorisation (Security & Enforcement)
- T3.4- Policy management and synthesis components
- T3.5- Transparency and compliance components.

Two tasks are defined (T3.2 and T3.3) in which the following should be tackled. Firstly, both the data and metadata should be represented in a secure and efficient way. By combining encryption and compression mechanisms, the volume of the data stored on disk is reduced while at the same time data is protected from unauthorised access. Furthermore, various indexing strategies will be considered based on the predominant access policies and access requests. Secondly there is the authentication and authorisation aspect. The data stored and processed in the SPECIAL engine is only accessible to authorised individuals. Data subjects, on the other hand, can only access data which concerns them, and data processors and controllers have a high-level view with respect to personal data compliance. The authentication and authorisation mechanisms are required to limit access to system components, data and metadata.

The creation of a SPECIAL Policy Manager is an important step (T3.4 and T3.5). As policy data and metadata will be compressed and encrypted, it is necessary to develop specific components for creating, managing and querying policies, and for attaching them to data. Such components are also responsible for the automated synthesis of policies for derived data. Moreover, within the context of a Policy Manager, transparency and compliance, components are responsible for recording data usage events and the associated provenance metadata. They also check that data sharing and processing actions comply with the applicable policies. Additionally, to build this component, a common language, vocabulary and rules must be decided on. Considerations regarding usage of the subject policy manager include:

1. changes in policy IDs,
2. changes in policy preferences of the data subject,
3. exported data: both how and which data, and
4. exploitation of more data.