D3.1 – EW-Shopp
Components Specification and Design

<table>
<thead>
<tr>
<th>Deliverable n:</th>
<th>3.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date:</td>
<td>29 December 2017</td>
</tr>
<tr>
<td>Status:</td>
<td>Final</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0</td>
</tr>
<tr>
<td>Authors:</td>
<td>M. Ciavotta (UNIMIB), F. De Paoli (UNIMIB), A. Košmerlj (JSI), A. Marguglio (ENG), N. Nikolov (SINTEF), M. Zvan (BT)</td>
</tr>
<tr>
<td>Contributors:</td>
<td>F. Rodriguez (JOT), P. Orel (BB), D. Jerončič (CE), O. Melnyk (ME), S. Dalgard (SINTEF)</td>
</tr>
<tr>
<td>Reviewers:</td>
<td>F. Perales (JOT), D. Čreslovnik (CE)</td>
</tr>
<tr>
<td>Distribution:</td>
<td>Public</td>
</tr>
</tbody>
</table>

Grant n. 732590 - H2020-ICT-2016-2017/H2020-ICT-2016-1
# History of Changes

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
<th>Revised by</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>10/10/2017</td>
<td>Definition of the Table of Contents</td>
<td>Flavio de Paoli</td>
</tr>
<tr>
<td>0.2</td>
<td>15/11/2017</td>
<td>Contribution to BC4 Section</td>
<td>Francisco Rodriguez</td>
</tr>
<tr>
<td>0.3</td>
<td>20/11/2017</td>
<td>Contribution on the state of the art, Data Analyzer and Qminer</td>
<td>Aljaž Košmerlj</td>
</tr>
<tr>
<td>0.4</td>
<td>22/11/2017</td>
<td>Contribution to BC1 Section</td>
<td>Davy Jerončič, Darko Dujič</td>
</tr>
<tr>
<td>0.5</td>
<td>24/11/2017</td>
<td>Contribution to BC3 Section</td>
<td>Olga Melnyk</td>
</tr>
<tr>
<td>0.6</td>
<td>24/11/2017</td>
<td>SINTEF’s contribution to several sections</td>
<td>Nikolay Nikolov</td>
</tr>
<tr>
<td>0.7</td>
<td>27/11/2017</td>
<td>Contribution to BC1 Section</td>
<td>Patricija Filipič Orel</td>
</tr>
<tr>
<td>0.8</td>
<td>29/11/2017</td>
<td>Chapter 1, Chapter 2 and Chapter 3 completed</td>
<td>Michele Ciavotta</td>
</tr>
<tr>
<td>0.9</td>
<td>04/12/2017</td>
<td>Chapter 4 and Chapter 6 completed</td>
<td>Michele Ciavotta, Flavio De Paoli</td>
</tr>
<tr>
<td>0.10</td>
<td>06/12/2017</td>
<td>Integration strategy section added</td>
<td>Michele Ciavotta</td>
</tr>
<tr>
<td>0.11</td>
<td>07/12/2017</td>
<td>Summary and conclusions added</td>
<td>Michele Ciavotta</td>
</tr>
<tr>
<td>1.0</td>
<td>22/12/2017</td>
<td>Revision according to reviewers’ comments and finalization of the deliverable</td>
<td>Michele Ciavotta, Flavio De Paoli</td>
</tr>
</tbody>
</table>
Executive Summary

EW-Shopp aims to support e-commerce, retail and marketing industries in improving their efficiency and competitiveness through providing the ability to perform predictive and prescriptive analytics over integrated and enriched large datasets through the use of open and flexible solutions.

This document is intended to be the first step towards the construction of the EW-Shopp ecosystem, a platform that seeks to significantly reduce integration time and to improve the quality of analytics by providing cutting-edge tools and access to data as a service, with particular reference to weather and event information.

We propose a reference architecture specifically designed to handle Big Data volumes. In order for this to be possible, we have carried out a thorough requirement gathering phase that takes into account the best practices in creating Big Data architectures and the expectations of the partners involved in business cases. As a result, we were able not only to define the platform architecture, but also to set up a unified common language that, we believe, will have clear benefits in the next stages of the project.

In addition, we propose an initial reference implementation by selecting one or more solutions for each reference component. In particular, embracing the lean development approach already introduced in Deliverable D4.1, we detail only the core of the platform, namely the tools on which a strong convergence exists among the consortium members (the minimum viable product or MVP). The rest of the platform is constituted by suggestion of tools and/or technologies that at this moment seem to be good candidates but that may be replaced as the project develops.
# Table of context

History of Changes........................................................................................................... 2
Summary............................................................................................................................ 3
List of figures .................................................................................................................... 6
List of tables...................................................................................................................... 7
Acronyms.......................................................................................................................... 8

## Chapter 1

Introduction ...................................................................................................................... 10
1.1 Objectives and Scope .............................................................................................. 11
1.2 Relationship to Other Deliverables ........................................................................ 11
1.3 Document Structure ............................................................................................... 12

## Chapter 2

Reference Solutions ........................................................................................................ 13
2.1 Big Data Reference Architectures ........................................................................... 13
2.1.1 NIST Reference Architecture ............................................................................ 13
2.1.2 BDVA Reference Model ..................................................................................... 16
2.2 Data Wrangling ....................................................................................................... 17
2.3 Analytics ................................................................................................................. 19
2.4 Visualization and Reporting .................................................................................... 20

## Chapter 3

Driving Requirements .................................................................................................... 22
3.1 Elicitation Methodology .......................................................................................... 22
3.2 Requirements Definition Survey ............................................................................ 23
3.2.1 Ceneje ................................................................................................................ 24
3.2.2 Big Bang ............................................................................................................. 25
3.2.3 Browsetel ........................................................................................................... 26
3.2.4 Measurence ........................................................................................................ 28
3.2.5 JOT ...................................................................................................................... 29
3.3 Architectural Requirements .................................................................................... 31
3.3.1 Functional Requirements .................................................................................. 31
3.3.2 Non-functional Requirements .......................................................................... 37

## Chapter 4

Architecture Design ........................................................................................................ 39
4.1.1 Envisioned Workflow and Actors ...................................................................... 39
4.2 Architectural Specifications ..................................................................................... 40
4.2.1 The Dimension Reduction Approach ................................................................. 43
4.2.2 Core Data Services .......................................................................................... 44
4.2.3 Platform Services ............................................................................................ 46
4.2.4 Corporate Services ......................................................................................... 50
4.2.5 Crosscutting Services ..................................................................................... 51
4.3 Control Flow ...................................................................................................... 51
4.4 Data Format ......................................................................................................... 54
4.5 Mapping the Reference Architecture on the Business Cases.............................. 55
  4.5.1 BC1 – Integration for Consumer Journey ....................................................... 55
  4.5.2 BC3 – Location Scouting ............................................................................... 56
  4.5.3 BC4 – Campaign-driven Purchasing Intentions .............................................. 57
Chapter 5 Reference Implementation ....................................................................... 59
  5.1 Overview ........................................................................................................... 59
  5.2 Integration Strategy ............................................................................................ 61
  5.3 Core Data Services ............................................................................................ 61
    5.3.1 EventRegistry ............................................................................................... 61
    5.3.2 Weather API ............................................................................................... 62
    5.3.3 Products and Other Knowledge Bases ............................................................. 63
  5.4 Platform Services ............................................................................................... 64
    5.4.1 Data Wrangler (DW) .................................................................................... 64
    5.4.2 Data Analyzer (DA) ..................................................................................... 67
    5.4.3 Data Reporter (DR) ..................................................................................... 69
  5.5 Corporate Services ............................................................................................. 71
    5.5.1 Cluster Configuration .................................................................................... 71
    5.5.2 Data Flow Manager ..................................................................................... 72
    5.5.3 Enrichment and analysis database ................................................................. 72
  5.6 Crosscutting services .......................................................................................... 73
    5.6.1 Security e privacy ......................................................................................... 73
    5.6.2 Monitoring and Auditing Services ................................................................. 73
  5.7 Single-site Configuration ..................................................................................... 75
Chapter 6 Conclusions .............................................................................................. 76
References ............................................................................................................... 77
List of figures

Figure 1: NIST Reference Architecture ................................................................. 16
Figure 2: BDVA Reference Model .............................................................. 16
Figure 3: Requirements definition process ..................................................... 23
Figure 4: Mapping EW-Shopp platform components on NIST reference architecture for Big Data ......................... 41
Figure 5: EW-Shopp logical reference architecture ............................................ 42
Figure 6: EW-Shopp reference Dataflow pipeline ............................................. 44
Figure 7: Data Wrangler architecture .............................................................. 47
Figure 8: Data Analyzer architecture .............................................................. 48
Figure 9: Data Reporter architecture .............................................................. 49
Figure 10: Generic control sequence for Data Wrangling ................................. 52
Figure 11: Generic control sequence for Data Analytics ..................................... 53
Figure 12: Generic control sequence for Report Visualization ......................... 53
Figure 13: BC1 Architecture ......................................................................... 56
Figure 14: BC3 Architecture ......................................................................... 57
Figure 15: BC4 Architecture ......................................................................... 58
Figure 16: ASIA Architecture ......................................................................... 66
Figure 17: Data Wrangler Component Specification ......................................... 67
Figure 18: Data Analyzer Component Specification .......................................... 68
Figure 19: Data Reporter Component Specification ........................................... 69
Figure 20: Dataset creation ............................................................................ 70
Figure 21: Querying the dataset ...................................................................... 70
Figure 22: Big Data Runtime Component Specification ..................................... 71
Figure 23: Monitoring system ........................................................................ 74
Figure 24: Self-contained Architecture and Component Specification .................. 75
### List of tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE 1:</td>
<td>REQUIREMENT F.1</td>
<td>32</td>
</tr>
<tr>
<td>TABLE 2:</td>
<td>REQUIREMENT F.2</td>
<td>32</td>
</tr>
<tr>
<td>TABLE 3:</td>
<td>REQUIREMENT F.3</td>
<td>32</td>
</tr>
<tr>
<td>TABLE 4:</td>
<td>REQUIREMENT F.4</td>
<td>32</td>
</tr>
<tr>
<td>TABLE 5:</td>
<td>REQUIREMENT F.5</td>
<td>33</td>
</tr>
<tr>
<td>TABLE 6:</td>
<td>REQUIREMENT F.6</td>
<td>33</td>
</tr>
<tr>
<td>TABLE 7:</td>
<td>REQUIREMENT F.7</td>
<td>33</td>
</tr>
<tr>
<td>TABLE 8:</td>
<td>REQUIREMENT F.8</td>
<td>33</td>
</tr>
<tr>
<td>TABLE 9:</td>
<td>REQUIREMENT F.9</td>
<td>34</td>
</tr>
<tr>
<td>TABLE 10:</td>
<td>REQUIREMENT F.10</td>
<td>34</td>
</tr>
<tr>
<td>TABLE 11:</td>
<td>REQUIREMENT F.11</td>
<td>34</td>
</tr>
<tr>
<td>TABLE 12:</td>
<td>REQUIREMENT SP.1</td>
<td>35</td>
</tr>
<tr>
<td>TABLE 13:</td>
<td>REQUIREMENT SP.2</td>
<td>35</td>
</tr>
<tr>
<td>TABLE 14:</td>
<td>REQUIREMENT SP.3</td>
<td>35</td>
</tr>
<tr>
<td>TABLE 15:</td>
<td>REQUIREMENT SP.4</td>
<td>36</td>
</tr>
<tr>
<td>TABLE 16:</td>
<td>REQUIREMENT SP.5</td>
<td>36</td>
</tr>
<tr>
<td>TABLE 17:</td>
<td>REQUIREMENT SP.6</td>
<td>37</td>
</tr>
<tr>
<td>TABLE 18:</td>
<td>REQUIREMENT NF.1</td>
<td>37</td>
</tr>
<tr>
<td>TABLE 19:</td>
<td>REQUIREMENT NF.2</td>
<td>37</td>
</tr>
<tr>
<td>TABLE 20:</td>
<td>REQUIREMENT NF.3</td>
<td>38</td>
</tr>
<tr>
<td>TABLE 21:</td>
<td>REQUIREMENT NF.4</td>
<td>38</td>
</tr>
<tr>
<td>TABLE 22:</td>
<td>REQUIREMENT NF.5</td>
<td>38</td>
</tr>
<tr>
<td>TABLE 23:</td>
<td>EW-SHOPP ENVISIONED ACTORS</td>
<td>39</td>
</tr>
</tbody>
</table>
# Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVP</td>
<td>Minimum Viable Product</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>BDVA</td>
<td>European one called Big Data Value Association</td>
</tr>
<tr>
<td>NBD-PWG</td>
<td>NIST Big Data Public Working Group</td>
</tr>
<tr>
<td>ETL</td>
<td>Extract, Transform and Load</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>DPU</td>
<td>Data Processing Unit</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>BI</td>
<td>Business Intelligence</td>
</tr>
<tr>
<td>ROI</td>
<td>Return Of Investment</td>
</tr>
<tr>
<td>EAN</td>
<td>European Article Number</td>
</tr>
<tr>
<td>TSV</td>
<td>Tab Separated Values</td>
</tr>
<tr>
<td>AWS</td>
<td>Amazon Web Services</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>KML</td>
<td>Keyhole Markup Language</td>
</tr>
<tr>
<td>OLAP</td>
<td>Online analytical processing</td>
</tr>
<tr>
<td>NAS</td>
<td>Network-attached storage</td>
</tr>
<tr>
<td>CSV</td>
<td>Comma Separated Values</td>
</tr>
<tr>
<td>PPC</td>
<td>Price Paid per Click</td>
</tr>
<tr>
<td>GDPR</td>
<td>General Data Protection Regulation</td>
</tr>
<tr>
<td>SSO</td>
<td>Single Sign On</td>
</tr>
<tr>
<td>DoA</td>
<td>Description of Action</td>
</tr>
<tr>
<td>DW</td>
<td>Data Wrangler</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>DA</td>
<td>Data Analyzer</td>
</tr>
<tr>
<td>DR</td>
<td>Data Reporter</td>
</tr>
<tr>
<td>BDR</td>
<td>Big Data Runtime</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol and RDF Query Language</td>
</tr>
<tr>
<td>KG</td>
<td>Knowledge Graph</td>
</tr>
<tr>
<td>KB</td>
<td>Knowledge Base</td>
</tr>
<tr>
<td>BC</td>
<td>Business Case</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>PETs</td>
<td>Privacy-Enhancing Technologies</td>
</tr>
<tr>
<td>CEP</td>
<td>Complex Event Processor</td>
</tr>
<tr>
<td>MARS</td>
<td>Meteorological Archival and Retrieval System</td>
</tr>
<tr>
<td>ECMWF</td>
<td>European Centre for Medium-Range Weather Forecasts</td>
</tr>
</tbody>
</table>
Chapter 1  Introduction

For some years now we have been witnessing the flourishing and reinforcement of Big Data methodologies in academia and especially in industry. This success can in part be explained by the promise of Big Data supporters to be able to define and deliver more accurate and efficient data-driven decision-making processes. This is not the only asset, but it is what we are most interested in highlighting within the EW-Shopp project.

The term "Data Analytics" is too often used to define a generic data driven decision process and therefore covers heterogeneous activities such as cleaning, linking, enrichment, application of analytic models up to business intelligence and visualization. In this document, we propose to design the EW-Shopp platform from its components and their relationships. We present a reference architecture obtained from a rigorous process of requirements elicitation that took into account both literature and best practices as well as the needs expressed by involved stakeholders.

A considerable amount of work has already been done within the consortium (presented in document D4.1 [1]) to outline the strategy for the implementation of business case pilots. In that particular document, not only the functional requirements associated with business cases (the implementation and reporting of which in marketed services is one of the core aspects of the project) have been collected and refined, but also the development guidelines based on lean methodology [2] have been defined. In particular, the build-measure-learn feedback loop is also presented. In a nutshell, the methodology envisions for each pilot a starting phase with Minimum Viable Product (MVP) meant to reduce the time-to-market. The MVP is going to be enriched as the project progresses and the vision matures. Pilots are therefore seen as playgrounds for experimenting approaches and technologies that eventually will be included in the final marketed services.

Three stages of the development are foreseen:

- **First stage** is building the integrated data platform from heterogeneous sources as the key inputs.
- In the **second stage** the technical partners will need to present is analytical expertise to model these data to extract relevant analytical patterns, predictive strength and business sense.
- The **last stage** will be the development of marketed services focused on target segments on top of the integrated platform.

At the time this document is written, we are at the beginning of the first phase and it is the right time to design a unified, complete and well-founded reference architecture. The declared objective is to provide a point of reference that will remain as steady as possible for the entire duration of the project and that can thus lead its evolution on the basis of a solid and agreed basis. To do this, as we will see, we started from the functional requirements collected in D4.1 trying to read in them the
answers to the architectural questions that we asked ourselves. It was immediately evident, however, that these requirements were not sufficient to define a modern, efficient, responsive and scalable platform. As a consortium, we have therefore decided to undertake an in-depth study of the problems that we want to tackle. The results of such teamwork are detailed in this document.

1.1 Objectives and Scope

With the work the consortium has done on component design, the processes and outcomes of which are described in this document, we aim to achieve the following two objectives:

1. Define a reference architecture. This deliverable provides a reference architecture that aims at being general and flexible enough to be successful applied in the pilots. The architecture describes the core components and their relationships in terms of data flow. Moreover, it aims at setting a common language among the partners. To make this possible, we have referred to the pilot descriptions, the partners’ experience, and the best practices of the field of Big Data. Furthermore, the effort required to define precisely the pilots, especially in terms of processes, data flow and workflow, has spawned awareness in the consortium members about the complexity of the problem and the need of a common reference. Finally, it is important to note that defining a reference architecture does not conflict with the lean methodology as we describe the components according to their general functionality, without imposing specific solutions.

2. Propose an initial implementation of the platform. This deliverable also provides an early stage implementation of the reference architecture, whose components can be divided into two groups; the first one contains tools upon which there is a general agreement among the partners, the second one features components that are either left unspecified or are concrete tool to be considered as an initial choice (subject to be further refinement). Such an approach is necessary for two reasons. First of all, following the lean methodology the pilots need freedom to experiment in order to make the platform evolve. Secondly, part of the ecosystem must be adaptable to the needs of individual pilots, which might require the use of specific technologies.

1.2 Relationship to Other Deliverables

This document is based on the work done in Deliverable D4.1 [1]. In both documents, we seek to provide the groundwork for realizing the vision of EW-Shopp, but this is done in a largely complementary way. The D4.1 outlines the development strategy and defines the various pilots as well as their functionalities. The document presents an initial set of requirements, the main elements of a baseline platform and a first version of the data flow. Here, the requirements have been taken up, expanded, evaluated and selected for their impact on architecture. The architecture has been detailed and reorganized. In addition, a clear strategy for handling large scale datasets has been outlined. Finally, the data and control flows have been defined.
This document must also be considered linked to D2.2 [3] to be released in M12 as well. In fact, D2.2 presents the first version of the EW-Shopp platform deployment in the cloud environment provided by SINTEF. The deployment is accompanied by a description of the work done and the various software tools involved. The objectives of the two documents therefore overlap in part. In the end, however, D2.2 responds to the specific need to present the current version of the platform while this document describes a more general and complete architecture and the process that led to its definition.

1.3 Document Structure

This document is structured as follows. In Chapter 2 we present the reference architectures for Big Data as well as the main solutions for data wrangling, data analytics, business intelligence and reporting. The requirements that have guided the definition of our reference architecture are presented and discussed in Chapter 3. The EW-Shopp reference platform is detailed in Chapter 4, where both the reference data flow and control flow are also presented. Once the reference architecture has been defined, we have decided to present a possible implementation (in embryonic form) of the EW-Shopp platform in Chapter 5. Finally, Chapter 6 concludes the document.
Chapter 2  Reference Solutions

The purpose of this chapter is to provide an exhaustive overview of the methodologies and technologies commonly adopted to create data preparation, analytics and reporting platforms capable of managing large volumes of data.

2.1 Big Data Reference Architectures

In recent years, we have witnessed the proliferation of references for Big Data architectures. The aim of these proposals was to fill the gap between industry and academia by seeking to establish shared objectives and taxonomies. In the particular field of Big Data architectures, industry has assumed the lead and driving role for the entire sector, proposing and experimenting various approaches. Many of the solutions that have been developed with this "trial and error" approach have been jealously guarded for a long time, but today we are witnessing a change in trend, especially due to initiatives of the open source community. From many quarters, it was deemed necessary to re-organize the know-how of the sector and to propose a structured and as complete as possible vision of a software architecture that can be defined as Big Data. Both research institutions and large companies have therefore proposed their architectural vision. Government initiatives joined forces. Among these, certainly worth mentioning are the U. S. under the National Institute of Standards and Technology (NIST), and the European one called Big Data Value Association (BDVA).

The NIST Big Data Public Working Group (NBD-PWG), born in 2013, and the BDVA, born in 2015 share similar ambitions. In fact, their primary objective is to foster the participation of Academia, Industry and Government in the creation of a community of interests for the adoption of the Big Data in many sectors. Both initiative, moreover, aims at developing consensus on definition taxonomies, reference models, and standards.

2.1.1 NIST Reference Architecture

The work of the NBD-PWG results in the NIST Big Data Interoperability Framework, which addresses the most important topics of Big Data, that is:

- Definition and taxonomies
- Use cases and general requirements
- Security and privacy
- Architectural survey
• Reference Architecture
• Standards Roadmap

Volumes 5 [4] and 6 [5] of the NIST Big DATA Interoperability Framework, which propose a survey of the most proposed architectures and a reference architecture, respectively, fall within the scope of this project.

As far as volume 5 is concerned, the survey covered nine reference architectures:

1. ET Strategies
2. Microsoft
3. University of Amsterdam
4. IBM
5. Oracle
6. Pivotal
7. SAP
8. 9Sight
9. LexisNexis

The work has been systematic and meticulous. In particular, for each specific implementation a task has been carried out to identify the common components, to which reference has been made by means of a single taxonomy. Three macro-components have been identified:

• Big Data Management and Storage: provides Big Data capabilities to structured and unstructured data.
  o Structured, semi-structured and unstructured data
  o Volume, variety, velocity, and variability
  o SQL and NoSQL
  o Distributed file system

• Big Data Analytics and Application Interfaces
  o ETL (Extract, Transform and Load) and Mining
  o Analytics
  o Report and Visualization
  o Governance

• Big Data Infrastructure: infrastructure and system support with capabilities of integrating internal or external sources.
Figure 1 depicts the Reference Architecture proposed by the NIST, which is expected to be a vendor-neutral, technology- and infrastructure-agnostic conceptual model of a Big Data architecture. It is the results of an elicitation and synthesis process based on the major proposals from the Industry and the Academia.

The main components of the NIST Reference Architecture are:

- **Data provider**: The data providers are the actors that provide the input datasets or information feeds to the Big Data ecosystem.

- **System orchestration**: Its purpose in the architecture is set up and manage the other components to implement one or more workloads. The System Orchestrator may also monitor the workloads and system to confirm that the Quality of Service (QoS) requirements are fulfilled in each phase of the data flow pipeline, and may actually elastically assign and provision additional physical or virtual resources.

- **Big Data Application Provider**: This component of the architecture is the one the implements the business logic and the added-value functionalities that are executed by the underlying architecture layers. The Big Data Application Provider can, in particular, provides support for the following activities/functionalties: Collection, Preparation, Analytics, Visualization, and Access.

- **Big Data Framework Provider**: This is the macro-component, made up of several hierarchically organized layers, that provides the infrastructure (virtual and physical resources, storage buckets, networks, etc.) and the tools for the management of high volumes of data (SQL, NoSQL, file systems, etc.) and to process them efficiently, often in a distributed way (Batch, Interactive, Streaming workloads).

- **Data Consumer**: as for the data provider, it can be an actual user or another system. It is the entity that interacts with the platform (e.g., it can execute Search and Retrieve tasks, or implement Analytics models) and consumes its outcomes (e.g., Reporting, Visualization, Download).

More details regarding the NIST Reference Architecture can be found in [5]
Similarly to what happened in the U.S., European politics have seen the importance of promoting a common space of public-private aggregation with the aim of fostering growth through the development of a Data-driven economy. This led to the stipulation of a public-private partnership between the European Commission (public partner) and the Big Data Value Association (private partner) as signed in October 2014.

Recently, BDVA has released the forth version of its research agenda [6], which summarizes the work done by several task forces in the definition of priorities and, a Reference Model (see Figure 2).
The Figure above represents graphically two different points of view, called Horizontal and Vertical Concerns in the BDVA lingo. The horizontal concerns regard the different aspects of the envisioned data processing chain, from data ingestion to reporting and visualization tasks. Interestingly, it although a clear mapping between the BDVA and NIST architectures can be identified, the BDVA reference model focuses more on the data, whereas the NIST architecture is more process and component centric. The inclusion of data protection and data management as first-class citizen of the architecture proves it. Unlike the NIST architecture, the BDVA seems to consider Things, Sensors and Actuators as the main data providers and consumers. The data centrality in the BDVA platform is clearly evident in what are called vertical concerns. In this context, the architecture makes explicit reference to data types and their semantics, and to data sharing platforms.

Finally, we would like to point out that in the platform proposed by BDVA there are considerations relating to the engineering, development and operations processes, which are completely absent in the NIST architecture as well as the need for standardization. This, although falling within the objectives of NIST, does not appear in its reference architecture as it is focused on components, data flows and processes. More details on the BDVA reference model can be found in [6].

### 2.2 Data Wrangling

Data Wrangling (preparation, curation, and integration) activities still consume up to the majority of the resources invested in data exploitation actions. The EW-Shopp integration architecture aims to deliver the best top-to-bottom solution to integrate business data with data coming from different sectors, by using shared systems of identifiers for Core Data. Recently, several software solutions appeared in the market, which provide transformation toolboxes and data flow management agent to realize suitable wrangling pipelines.

**OpenRefine**\(^1\) is an open source desktop tool (since 2010) previously known as Google Refine. This software solution provides the tooling necessary to perform cleanup and transformation activities on data provided in tabular form. An OpenRefine project consists of a single data table that can be filtered and enriched using data from external data sources. Data reconciliation using Wikidata is also supported. OpenRefine presents a web user interface served by a web server running locally on the user’s machine. This software’s main limitations are related to the size of the datasets it can handle, the number of external sources and data monitoring capabilities.

**Trifacta**\(^2\) is a commercial suite of applications for the exploration and preparation (transformation and enrichment) of raw datasets to make them suitable for the subsequent analysis phase. The declared objective is to provide non-skilled users with specific intelligent tools to prepare datasets for different analysis types. To do this, advanced techniques of machine learning, parallel processing, and human-machine interaction are employed. The company that develops these tools was born in 2012 as a joint effort of researchers from the universities of Berkeley, Washington and Stanford. The suite consists of three software solutions with increasingly advanced features that allow the

---

\(^1\) [http://openrefine.org](http://openrefine.org)

\(^2\) [https://www.trifacta.com](https://www.trifacta.com)
processing of large volumes of data. Since March 2017, Google has been providing a cloud version of Trifacta branded as **Cloud Dataprep**[^3].

**UnifiedViews**[^4] [7] is an open-source tool for creating ETL pipelines that natively supports data in RDF (Resource Description Framework). UnifiedViews provides the abstractions and toolboxes needed to define, execute, and monitor Data Processing Unit (DPU) tasks. The DPUs supported by UnifiedViews are:

- Importing raw data from external sources
- Transformation of data into RDF
- Cleaning the data
- Interlinking with other data sources
- Conflict resolution
- It also allows the sharing of transformation results.

**Karma**[^5] is a data integration tool whose main objective is to provide rapid and intuitive tools for data integration for users. Datasets can be generated from different sources such as spreadsheets, delimited text files, XML, JSON, KML, and Web APIs. Moreover, Karma is open source and versions compatible with Windows Linux and Mac operating systems are available. Karma developers claim to offer some innovative features that distinguish this tool from others:

- **Usability:** guaranteed by a graphical interface that gives users the possibility to refine a semantic model previously generated in automatic mode. In this way, the complexity of the association rules is hidden from end users.
- **Flexibility:** It supports a variety of structured data sources that can always be analysed and annotated semantically. These range from simple tables to the most complex structured XML, JSON, and KML files. Data integration service is also supported for Web APIs. Karma offers the possibility of combining different ontologies to associate their data with standard vocabulary and ensure more accurate and accurate data integration.
- **Responsiveness and Big Data:** karma uses only a subset of the data provided by users to complete its services, and thanks to this strategy it offers a responsive graphical interface and reduced calculation times; in addition, it provides many a batch execution mode for the integration of large datasets;

This last consideration is particularly interesting in the scope of EW-Shopp as it outlines an approach that share many similarities to the one designed and implemented in the project. More details can be found in Chapters Chapter 4 and Chapter 5.

**DataGraft**[^6] is a web-powered tool that features a collation of services for managing the process of transforming data from tabular format into a graph format and deploying the data onto a database.

[^3]: [https://cloud.google.com/dataprep](https://cloud.google.com/dataprep)
[^4]: [https://unifiedviews.eu](https://unifiedviews.eu)
[^6]: [https://datagraft.net](https://datagraft.net)
The transformation supports both cleaning and mapping steps. Transformation steps on rows (add, drop, filter, duplicate detection etc.), columns (add, drop, rename, merge etc.) and entire dataset (sort, aggregate etc.) are provided together with visualization of the result after each step. More details of DataGraft can be found in Section 5.4.1.

### 2.3 Analytics

There is a plethora of different data analytics tools and architectures available today. In general, the differences between them are often minute and the choice comes down to personal preferences. Here we provide an overview of some of the most known and widely-used analytics platforms.

**Weka** is one of most well-known machine-learning suites. It was developed and is maintained by the University of Waikato, New Zealand and is widely spread due to its comprehensive collection of data pre-processing and modelling techniques. The main areas where it is used are education and research, where it is popular also due to its free availability through the GNU General Public License\(^8\). It supports all stages of machine learning including pre-processing, modelling and visualization. It is developed in Java programming language and offers a GUI front end.

**Orange** follows the motto of making data mining “fruitful and fun”. Developed and maintained at the University of Ljubljana it is a machine-learning suite with a strong support for visual programming. The user can build a solution by visually drawing a flowchart on the Orange canvas using elements from the Orange library, which produces a functional program in the background. These functionalities make it popular in education. Research users typically use it directly as a Python library.

**R** differs from the previous platforms in the sense that its roots are more in the area of statistics and not machine learning. It is an open source programming language and an environment for statistical computing and graphics. It is widely used throughout research and business and has one of the most extensive libraries of tools. As can be expected, the area of statistical modelling is covered best.

**Python** is a general-purpose scripting programming language and not a data analytics platform. Nevertheless, its selection of modules supporting analysis of data make it a popular choice. Most functionalities of the previously described platforms are freely available through modules such as:

- **Scikit** – general purpose analytics and modelling,
- **TensorFlow, Theano** – powerful libraries for modelling using (deep) neural networks,
- **Pandas** – a library offering versatile data structures for data analysis,
- **Matplotlib, Plotly** – libraries for data visualization.

**RapidMiner** is a commercial data science software platform developed by the company with the same name. As a general-purpose suite, it has support for data preparation, modelling and

---

\(^7\) [http://www.cs.waikato.ac.nz/~ml/weka](http://www.cs.waikato.ac.nz/~ml/weka)

\(^8\) [https://www.gnu.org/licenses/gpl.html](https://www.gnu.org/licenses/gpl.html)

\(^9\) [https://orange.biolab.si](https://orange.biolab.si)

\(^10\) [https://www.r-project.org](https://www.r-project.org)

\(^11\) [https://www.python.org](https://www.python.org)
visualization and is used for academic as well as business applications. Its design follows the open core model with the free basic version being limited to 1 logical processor and 10,000 data rows. It is developed in Java and offers a GUI front end. Its suite of models and algorithms can be extended using either Python or R languages.

**Qminer**\(^{13}\) is an open source data analytics platform written in C++, especially designed for processing of large-scale datasets in real time. It supports both structured and unstructured data including text, geospatial data, networks and time streams. The QMiner library contains an extensive array of analytics tools covering classification, regression, and preprocessing, among the others. More details on Qminer can be found in Section 5.4.2.

## 2.4 Visualization and Reporting

Big Data applications require new forms of processing to enable enhanced process optimization, insight discovery and decision making. Effective data visualization is a key part of the discovery process in the era of Big Data, since visualization is an important approach to helping Big Data get a complete view of data and discover data values, by placing them in a visual context. Patterns, trends and correlations that might go undetected in text-based data can be exposed and recognized easier with data visualization software.

Data visualization is a Business Intelligence (BI) tool aims to create reports; dashboards and data visualizations to present actionable information to users that can make informed business decisions.

The potential benefits of business intelligence tools, as Data visualization, include: accelerating and improving decision-making, optimizing internal business processes, improving collaboration and information sharing, providing self-service capabilities to end users, increasing Return Of Investment (ROI), saving time and reducing burden on IT [8].

The outcome of all visualization tools is a certain number of charts/reports provided, being valuable by the end user and making possible to emphasize a specific feature of the data.

EW-Shopp project aims to take advantage of all these benefits through the developing of business services, which provide insights resulting from the analysis and visualization of large amount of consumer and market data, in multiple languages, collected by different business partners and enriched with information about weather and events.

According to EW-Shopp project needs, the visualization and data navigation tier has to provide **end-users simplified data access and services** within a Big Data visualization platform and **insightfully display** of geospatial (transport/traffic), weather, network (linked) data with multiple attributes (multivariate data) and/or a temporal dimension.

---

\(^{12}\) [https://rapidminer.com](https://rapidminer.com)

\(^{13}\) [http://qminer.ijs.si](http://qminer.ijs.si)
Several solutions exist in the market, but with different dashboarding capabilities, API availability, licensing, and pricing. Most relevant solutions are: Tableau\(^{14}\), Google data studio\(^{15}\), Datahero\(^{16}\), Plotly saas\(^{17}\), Raw\(^{18}\), Visualizefree\(^{19}\), Datawrapper\(^{20}\), Infogram\(^{21}\), Chartblocks\(^{22}\), Zoho reports\(^{23}\).

Looking at project context and after comparing all those different Big Data visualization tools, discovering strengths and weaknesses, the Knowage Suite (a tool developed by ENGINEERING) can be considered a good choice in EW-Shopp project thanks to its features.

Knowage\(^{24}\) is an open source business analytics suite that combines traditional data and Big Data sources into valuable and meaningful information. It represents the evolution of SpagoBI, the first suite released by ENGINEERING and with a ten-year history. It is a suite of autonomous and combinable products that allows everyone to define their own functional perimeter, with the possibility of extending it at a later date. Actually, it is available in two versions: Knowage Community Edition (CE), with open source license, ensuring full utilization of analytics functionalities and full operativeness of the final user; Knowage Enterprise Edition (EE), with a subscription model, facilitating management operations for the installed park and ensuring the professional services required for use in business settings.

Knowage provides advanced self-service capabilities that give autonomy to the end-user, who is able to build his own analysis, get insights on data and turn them into actionable knowledge for effective decision-making processes. The software is flexible because it adopts open standards and can be used in various environments without considerable requirements. Its modular approach, scalable architecture and open standards ensure easy customization and the development of user-friendly solutions.

To summarize, the main Knowage characteristics are the following:

- It is a pure Open Source Model
- It is released at no license fee
- It presents “no software and vendor lock-in”, the right to use the software is separated from the purchase of professional services
- It embraces innovation and the results coming from research activities and contributions
- It is offered also as reference implementations of FIWARE GE specifications\(^{25}\)
- It has enabled connectors: the platform can gather information from multifarious data sources by means of some connectors that are ready to be used off-the-shelf

\(^{14}\) [https://www.tableau.com](https://www.tableau.com)
\(^{15}\) [https://datastudio.google.com](https://datastudio.google.com)
\(^{16}\) [https://datahero.com](https://datahero.com)
\(^{17}\) [https://plot.ly](https://plot.ly)
\(^{18}\) [http://rawgraphs.io](http://rawgraphs.io)
\(^{19}\) [https://visualizefree.com](https://visualizefree.com)
\(^{20}\) [https://www.datawrapper.de](https://www.datawrapper.de)
\(^{21}\) [https://infogram.com](https://infogram.com)
\(^{22}\) [https://www.chartblocks.com](https://www.chartblocks.com)
\(^{23}\) [https://www.zoho.eu](https://www.zoho.eu)
\(^{24}\) [https://www.knowage-suite.com](https://www.knowage-suite.com)
Chapter 3     Driving Requirements

The main objective of this section in the scope of the document is to elicit, gather and organize the core requirements to be addressed in EW-Shopp Reference Architecture as well as in its actual implementation as a first step towards the realization of the project vision.

3.1 Elicitation Methodology

In order to collect the requirements that will drive the definition of the EW-Shopp Reference architecture, the following steps have been carried out (see Figure 3):

i. **Review of the relevant State of the Art in Big Data Architecture:** In Chapter 2 of this document, the state of the art analysis of both the two main reference architectures and the main data transformation, analytics and reporting tools has been reported. As the EW-Shopp project is focused on defining large-scale data transformation and analytics processes, the state of the art of data storage and data processing has been deliberately neglected. From this analysis, we mainly extracted a list of components that (sometimes under different names) are present in the Big Data platforms. Some of these were not considered because they relate to real time analysis systems. Others, however, such as those relating to data ingestion, are not explicitly included among the requirements, because we have tried to focus exclusively on data transformation, enrichment, analysis and reporting of data.

ii. **Collection of requirements defined in other deliverables:** Deliverable D4.1 outlines the development strategy and defines the various pilots as well as their functionalities. The document presents an initial set of functional requirements, the main elements of a baseline platform and a first version of the data flow. The requirements that can be found in D4.1 mainly concern the business logic behind the pilot; however, important requirements as well as a first definition of the components to be involved in the definition of the platform are presented.

iii. **Development of surveys for the stakeholders:** In order to actively involve the consortium stakeholders, i.e., the EW-Shopp business partners, in the requirements elicitation a survey have been developed. The questions of this survey mainly concern the current state of the data transformation and analytics process, the size of the datasets to be processed, the presence or not of Big Data solutions on premises. In Section 3.2, the reader can find the survey questions and the related answers.

iv. **Elicitation of architectural requirements from (i), (ii), and (iii):** Once these three parallel steps were completed, two sets, one of functional and the other of non-functional requirements were identified (see Sections 3.3.1 and 3.3.2).
v. **Prioritization of the requirements.** Finally, a prioritization process has been carried out. The goal of this process is to identify top priority requirements. Consequently, both functional and non-functional requirements have been divided into two subsets, namely must, that is requirements of primary importance, and should, containing those requirements whose non-fulfilment would not affect the success of the project.

![Diagram of requirements definition process]

**Figure 3: Requirements definition process**

### 3.2 Requirements Definition Survey

Below are the requests for clarification addressed to the stakeholders involved in the implementation of Business cases. In particular, we asked to define and describe the current platform (where present), the possible deployment of the entire ecosystem and the expected workflow.

1. *Describe here the current business case platform (pre EW-Shopp), in particular try to detail:*
   - *The current data transformation processes (either batch or streaming)*
   - *The data sources (+ external APIs)*
   - *The approximate dataset sizes*
   - *The current Big Data architecture (if any) – Analytics, Processing tools, database*

2. *Describe the envisioned deployment for the entire ecosystem (current business case platform + EW-Shopp tools). In particular, specify whether the system will be deployed on premises, entirely on the cloud or use a hybrid configuration.*

3. *Define the specific workflow for the case of use. In particular, an effort is needed to define the frequency of use for the tools made available by the project. Possible examples (among others) of workflows are:*
• **Fully static:** The EW-Shopp components are used (once or rarely) to define the transformation and enrichment process, and validate the analytical algorithms. Once a satisfactory result has been achieved, the defined transformation and analytics models are implemented in a stable fashion within the business case platform, giving it added value.

• **Dynamic:** Tools are used on a regular basis to continuously improve the data flow on the business case platform.

• **Fully interactive:** In this case the size of the datasets could be small and a Big Data platform unnecessary. The main objective of the analyses is to gain knowledge about the correlation between data and events and weather.

## 3.2.1 Ceneje

### Current platform

Ceneje gathers market and product data from partners and stores them in an SQL database where it is transformed and merged to suit Ceneje business needs. All consumer activity is gathered and stored in a NoSQL database. Following datasets will be available:

• **Ceneje dataset – Consumer data:** Purchase intent: A collection of user journey data – page views, search terms, redirects to sellers and similar. Data is logged to local databases and data from 1.1.2015 is provided. Local databases consist of SQL databases and NoSQL databases. For more information please see Deliverable D2.1 [9] Section 6.1.

• **Ceneje dataset – Market data:** A collection of seller quotes for products. For more information please see Deliverable D2.1, Section 6.16.

• **Ceneje dataset – Product data:** A collection of product attributes (varying from generic such as name, EAN (European Article Number), brand, categorization and color to more specific as dimensions or technical specifications). Data is collected from more than a thousand online stores in 5 countries and then automatically and manually merged into an organized dataset. For more information please see Deliverable D2.1, Section 6.9.

All data will be available in TSV and JSON format on AWS (Amazon Web Services) servers. Based on these properties, the approximate dataset sizes are:

• **Ceneje dataset – Consumer data:** Purchase intent: > approx.1 TB, updated with 50MB daily

• **Ceneje dataset – Market data:** approx. 200 GB, updated with 3 GB daily

• **Ceneje dataset – Product data:** approx. 100 GB, updated with 100MB daily

Current Big Data architecture: Ceneje has no Big Data architecture. All current data is on on-premises hardware, archived data has been moved either to the cloud (AWS) or to a NAS.

### Envisioned deployment

Ceneje prefers an on-premises platform deployment but can also settle with a cloud deployment. Ceneje will be flexible according to EW-Shopp specifications and other partners’ needs. The final
deployment architecture and resources will be customized depending on the needs and functionalities of the core EW-Shopp modules: DataGraft/ASIA/ABSTAT (For data basis enrichment and manipulation), QMINER (analytics) and Knowage (for reporting and visualization).

**Workflow**

Ceneje Business case consists of two services. For both cases, daily TSVs and JSONs will be sent to EW-Shopp with the previous day information. Then, the requested information will be different for BS1 and BS2.

- **BS1:** The main objective of this service is to know products of interest for a certain user. Whenever a Partner wants to predict product of users’ interest, knowing the effect of weather and events, he will ask EW-Shopp for this service.
- **BS2:** The main objective of this service is to know categories of interest for a certain B2B partner. Whenever a Partner wants to predict interesting categories, knowing the effect of weather and events, he will ask EW-Shopp for this service.

Therefore, based on the number of products and user interactions Ceneje business services will interact with the EW-Shopp platform in dynamic and fully interactive approach (depending on the date, season, current weather data, weather forecast and so on).

### 3.2.2 Big Bang

**Current platform**

Big Bang stores all data (customer purchase history data, customer intent and interaction data, product attributes, product and categories sell-out data) in relational databases on MS SQL Server. The datasets available for EW-Shopp project will be:

- **Big Bang dataset – Consumer data – Customer purchase history:** A collection of sell-out data matched with customer baskets in a defined timeframe. Dataset is available from 2013 and can be provided in total or per pick-up location. Data is logged to local databases that consist of SQL databases and NoSQL databases. For more information please see Deliverable D2.1. Section 6.4.

- **Big Bang dataset – Consumer data – Customer Intent and Interaction:** A collection of user journey data (page views, page events, search terms, redirects to channels, product views, etc.). Dataset is available since 2013. Data is logged to Google Analytics database. For more information please see Deliverable D2.1. Section 6.5.

- **Big Bang dataset – Product and Categories data – Product Attributes and Sales Data:** A collection of detailed product specifications for products which are included in Big Bang selling portfolio (from generic to specific technical details). Dataset can be provided in total or per store location. Data is logged to local SQL databases. For more information please see Deliverable D2.1. Section 6.15.

All data will be available in SQL format, on Big Bang serves and can be transformed to various output files.

Based on these properties, the approximate dataset sizes are:
• **Big Bang dataset – Consumer data – Customer purchase history:** approx. 800 MB-1.2 TB, updated with 100 MB daily.

• **Big Bang dataset – Consumer data – Customer Intent and Interaction:** approx. 100 GB, updated with 1 GB daily.

• **Big Bang dataset – Product and Categories data:** approx. 75 GB, updated with 50 daily.

**Current Big Data architecture:** Big Bang system has a Big Data architecture based on relational database systems, mostly working with MySQL database engine with custom import/export tools and tools for OLAP (Online analytical processing) analysis. All current data is on on-premises hardware, archived data has been moved either to the cloud or to a NAS (Network-attached storage).

**Envisioned deployment**
Big Bang prefers an on-premises platform deployment but can also settle with a cloud deployment. Big Bang will be flexible according to EW-Shopp specifications and other partners’ needs. The final deployment architecture and resources will be customized depending on the needs and functionalities of the core EW-Shopp modules: DataGraft/ASIA/ABSTAT (For data basis enrichment and manipulation), QMINER (analytics) and Knowage (for reporting and visualization).

**Workflow**
Big Bang Business case consist of two services for which data in agreed format will be sent to EW-Shopp with the previous day information.

The requested information will be as follows:

• **BS1:** The main objective of this service is to determine influential factors to support Decision Making i.e. which activities should be implemented by channels and content for specific categories of interest.

• **BS2:** This service supports the analysis and the visualization of the possible impact of weather, events, marketing activities, price changes etc. on search and sell-out of categories and products.

Therefore, based on the number of products and user interactions Big Bang business service will interact with the EW-Shopp platform in dynamic and fully interactive approach (depending on the date, season, current weather data, weather forecast and so on).

### 3.2.3 Browsetel

**Current platform**
Current data transformation processes: With the goal of generating suitable datasets to be enriched with weather and events information and the further processing, several transformation processes are developed within BT IT architecture.

Data Sources: With the aim of generating the most complete and real dataset possible, BT will collect traffic related data within the following dataset:
• **BT Dataset – Contact and Consumer Interaction Data**: dataset, of a proprietary structure, in CSV format, collecting information related to customers’ interactions and other events within the BT system (e.g. agents’ login / logout, agents spent time in certain status, agents time of joining a campaign, etc.). It covers information related to Slovenian region in English language. For more information please see Deliverable D2.1. Section 6.7.

Based on these properties, the approximate dataset sizes are:

• **BT Dataset – Contact and Consumer Interaction Data**: > 600 GB, updated with 5 GB per month;

Current Big Data architecture: Actual BT system has a Big Data architecture based on relational database systems, mostly working with MySQL database engine with custom import/export tools and tools for OLAP analysis like Mondrian. BT is also experimenting with Apache Nifi. With the aim of facilitating the scalability of the Core Data basis and their business partners’ access to the information, BT is providing implementation of its solutions on the premises, cloud or hybrid platforms, enabling integration of business partner already existing systems (via standard or proprietary developed APIs).

**Envisioned deployment**

BT platform deployment will be enabled in a premise or cloud platform and will be flexible according to EW-Shopp specifications. The final deployment architecture and resources will be customized depending on the needs and functionalities of the core EW-Shopp modules: DataGraft/ASIA/ABSTAT (For data basis enrichment and manipulation), QMINER (analytics) and Knowage (for reporting and visualization).

**Workflow**

BT Business case is divided in three different services. For all cases, csv will be sent to EW-Shopp. Period and data set for data updating will be defined by BT.

• BS1: The main objective of this service is to know the best moment to launch a campaign based on the characteristics of the service, climatic factors, and events that may influence the objectives set for that campaign. This best moment can be defined depending on the expected goals to be achieved: interactions related to a certain campaign, product, etc. Whenever BT wants to launch a new campaign, knowing the effect of weather and events, will ask EW-Shopp for this service.

• BS2 and BS3: This service supports the analysis and the visualization of the possible impact of events on online searches. The goal will be to predict how and when an event can affect user behavior. BT user will set up the how he wants to be warned and then the platform will show an alert in the period that BT user specified. Analytics (and visualization) can be performed on a basis of BT defined period.

Therefore, based on the amount of campaigns the company used to manage and the fact that they are dealing with all possible categories, BT business services will interact with the EW-Shopp platform between a dynamic and fully interactive approach (depending on the date, weather forecast and so on).
3.2.4 Measurence

Current Platform
Measurence’s sensor platform comprises of Wi-Fi sensors, sensor manager and sensor monitoring system. Wi-Fi sensors, which are placed in/out of physical locations (i.e. small and big shops, out-of-home advertising, etc.) passively collect and anonymize Wi-Fi signals transmitted by nearby smartphones. After anonymization and filtering from anomalies, unnecessary or private data and static devices, data about Wi-Fi signals, namely time and signal strength of probe requests from devices is processed by Measurence’s Wi-Fi Analytic Pipeline.

The Wi-Fi Analytic Pipeline is a composed by a series of micro-services developed in Scala with the Akka framework\(^{26}\) and by a job procedure realized in Scala [10] that is run on an Apache Spark\(^{27}\) cluster and that translates the raw data into our analytics. In other words, Measurence algorithms estimate the flow of people across sensors and locations, people traffic inside and outside a location, and loyalty and engagement metrics. All described above raw and processed data after anonymization step is stored in the database.

Measurence provides to the clients (retailers) the information about traffic of people in and out of their physical location. The pilot data set contains information about half-hourly, daily and weekly number of visitors and pedestrians (walkby) for two shops and the daily number of receipts (private data of the shop) for one of the location. So Measurence does not provide to the partners/pilot during the project the raw data for individual devices, but already processed data – aggregated numbers.

Six months of Measurence data (csv-file) contains the data set of traffic (half-hourly, daily and weekly number of people) in/out of the particular location ~ 0.5 Mb.

Envisioned Development
EW-Shopp platform will add to the data an additional layer of intelligence: weather and events data. During the pilot Measurence expects to build the analytical model and consider/find different correlations between number of receipts/visitors/walkby and different weather attributes like temperature, precipitation, wind etc. Measurence will use both its own algorithms / facilities and EW-Shopp modules: DataGraft (for data basis enrichment and manipulation), QMINER (analytics) and Knowage (for reporting and visualization) to find the most effective way of data integration and delivering of the analytic to the customers. The training (pilot) data set is csv file but we plan to use the APIs provided by the EW-Shopp ecosystem for the data integration process.

Workflow

1. historical analysis – static approach. Measurence business case expects to study the behavior of customers in particular location using historical data of traffic of people and weather/events. Since the behavior of people might be different in different type of stores that evidently depends not only on daily deviations of weather conditions but also seasonality, physical location and characteristics of products sold by the shop, this kind of historical analysis should be done for every Measurence customer/location which decide to

\(^{26}\) [https://akka.io](https://akka.io)
\(^{27}\) [https://spark.apache.org](https://spark.apache.org)
use EW-Shopp service. Also, we should study how the forecast of the weather depends on the real weather and how it may influence on the people behavior and accuracy of the forecast of people traffic itself. The output of this study should look like “formula” in the form of “decision tree” that will be applied to the daily data in order to predict next week data.

2. weekly/daily data – interactive approach. As soon as the historical analysis is done, Measurence makes a list of physical coordinates/addresses for the locations/stores which desire to use EW-Shopp service (at the beginning these are two pilot locations) and sends a request to EW-Shopp platform on weekly (or daily) basis. EW-Shopp ecosystem response (ideally through API) contains needed attributes of past week actual weather and forecast for a coming week. Measurence applies “formula” to the integrated data and shows the traffic forecast on its dashboard.

3.2.5 JOT

Current platform

Current data transformation processes: With the goal of generating suitable datasets to be enriched with weather and events information and the further processing, several transformation processes are developed within JOT IT architecture. The main ones are: Automation of data extraction based on platform APIs, definition and selection of campaigns and parameters containing relevant information, selection of linking attributes for external data integration (like data, location, etc.)

Data Sources: With the aim of generating the most complete and real dataset possible, JOT will collect web traffic related data from three data sources. Two of them will contain the information regarding JOT marketing campaigns and the other one will contain twitter trends.

- **JOT Dataset - Consumer data: Traffic source (Google):** This dataset is in CSV format and it collects information related to different countries and different languages (Spain and Germany for the pilot). It contains all the information related to searches made through the Google platform. In order to access the data for the campaigns launched on the Google AdWords platform, Google provides an API which allows us to consult all the data we need quickly and intuitively. Each row features different fields among which there are the number of clicks, impressions, and the location. The Dataset is created daily meaning that every day the dataset contains only the newly generated data. For more information please see Deliverable D2.1, Section 6.19.

- **JOT Dataset - Consumer data: Traffic source (Bing):** This dataset is also in CSV format and it collects data gathered from different countries and in different languages. It contains all the information related to searches made in Microsoft Bing platform, showing different fields as the number of clicks or the location. Data is updated daily that means every day the dataset contains only the newly generated data. For more information please refer to D2.1, Section 6.18. As for the Bing Ads platform, the way we obtain the data is very similar to the Google

---

28 https://developers.google.com/adwords/api/docs/reference/v201607/ManagedCustomerService
AdWords platform as outlined above, given that they have an API to obtain the same data that is analogous.\(^{29}\)

- **JOT Dataset - Market data:** Twitter trends: This dataset has a CSV format, contains Open data and focuses on trending topics as available through Twitter APIs. Data is updated daily that means every day the dataset contains only the data newly generated. For more information please see D2.1, Section 6.20. API REST is used to download all the information needed.

Based on these properties, the approximate dataset sizes:

- **JOT Dataset - Consumer data:** Traffic source (Google): > 3 TB, updated with 4 GB daily
- **JOT Dataset - Consumer data:** Traffic source (Bing): 1 TB, updated with 1.5 GB daily
- **JOT Dataset - Market data:** Twitter trends: 2 million registers, updated with 24,000 new registers daily

Current Big Data architecture: Nowadays JOT has no Big Data architecture. With the aim of facilitating the scalability of the Core Data basis and the access to the information, JOT has migrated the data storage process from on-premises hardware to the cloud (Microsoft Azure).

**Envisioned deployment**

JOT platform deployment will be in a cloud platform (Azure), and it will be flexible according to EW-Shopp specifications. The final deployment architecture and resources will be customized depending on the needs and functionalities of the core EW-Shopp modules: DataGraft/ASIA/ABSTAT (For data basis enrichment and manipulation), QMINER (analytics) and Knowage (for reporting and visualization).

**Workflow**

JOT Business case is divided in four different services, so the frequency of use can be split in two, depending on the Business Service used. For all cases, daily csv will be sent to EW-Shopp with the information about the campaign performance on the previous day. Then, the requested information will be different for BS1 and BS2, BS3 and BS4.

- **BS1:** The main objective of this service is to know the best moment to launch a campaign based on the characteristics of the service, climatic factors, and events that may influence the objectives set for that campaign. This best moment can be defined depending on the expected goals to be achieved: number of clicks, average position, PPC (price paid per click). Whenever JOT wants to launch a new campaign, knowing the effect of weather and events, will ask EW-Shopp for this service.
- **BS2, BS3 and BS4:** This service supports the analysis of the possible impact of events on online searches. The goal will be to predict how and when an event can affect user behavior. JOT user will set up how he wants to be warned and then the platform will show an alert in the period that JOT user specified.

Therefore, based on the amount of digital campaigns the company used to manage and the fact that they are dealing with all possible categories, JOT business services will interact with the EW-Shopp platform between a dynamic and fully interactive approach (depending on the date, weather forecast and so on)

### 3.3 Architectural Requirements

The harvested requirements are presented and described in this section. Compared to a classic classification into functional and non-functional requirements, we have preferred to make the functional requirements of security and privacy explicit as they are deemed of strategic importance as recognized at European level by the General Data Protection Regulation - GDPR [11].

It is important to note that the requirements collected are those that have a direct impact on the architecture of the EW-Shopp ecosystem. You can easily realize that they have been left specially at a high level because they have to answer architectural questions:

1. **What functionality should the platform support?** The project is defined by its precise objectives, which concern only certain problems of Big Data architectures. It is important to define what falls within the scope of the project and therefore needs to be analyzed in detail, and what can be achieved simply by following best practices.

2. **To which areas should the identified functions be attributed?** In other words, we want to define architecture by identifying its components based on their responsibilities.

3. **How do architecture components interact?** The requirements have served us to reason about the relationships between the components, so that we can design their integration while keeping the objectives of each component separated as much as possible.

4. **What are the users of the system, and how do they interact with it?** The definition of the workflow and data flow further allows you to specify the system architecture.

These questions have guided us in identifying the main components, dividing them into groups. They also allowed us to clarify many points of the project by focusing on important issues such as Big Data and security management. Finally, this work has allowed us to create a unique language and also to better define the contours of the associated pilot projects.

### 3.3.1 Functional Requirements

Below are listed, in tabular form, the main functional requirements to be met by the platform. Each one of them is associated with a unique id, a short textual description and a level of stringency. Stringency can take on two values, namely Must and Should, meaning that the achievement of a particular requirement cannot be compromised as it is considered a fundamental condition for the success of the project (must). Other requirements, on the other hand, are desiderata, i.e. they describe functionality that the system should possess in order to fully express its potential but whose lack does not affect the success of the project.
Table 1: Requirement F.1

<table>
<thead>
<tr>
<th>Title</th>
<th>Data wrangling</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.1</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The platform must provide a (possibly open source) tool for the data transformation process. The tool will allow the user to interactively build data transformations. Among the activities supported by the tool there is the transformation of the user data into RDF knowledge graphs.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Must</td>
</tr>
</tbody>
</table>

Table 2: Requirement F.2

<table>
<thead>
<tr>
<th>Title</th>
<th>Data annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.2</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The platform must provide a (possibly open source) tool for the data annotation, data linking, and data enrichment. This software component must be able to help the user during the semantic annotation and enrichment of the dataset by incorporating information available openly or only to consortium members.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Must</td>
</tr>
</tbody>
</table>

Table 3: Requirement F.3

<table>
<thead>
<tr>
<th>Title</th>
<th>Core Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.3</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The data enrichment process must support the EW-Shopp Core Data sources, which provide information about Events, Weather and Products (at least). Such data sources can be accessed and consumed in the form of both remote and local APIs.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Must</td>
</tr>
</tbody>
</table>

Table 4: Requirement F.4

<table>
<thead>
<tr>
<th>Title</th>
<th>Data Analytics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.4</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The system must provide a (possibly open source) tool for the data analysis activity. The selected tool must provide efficient and scalable implementations</td>
</tr>
</tbody>
</table>
of the main analysis and machine learning algorithms.

<table>
<thead>
<tr>
<th>Req. Stringency</th>
<th>Must</th>
</tr>
</thead>
</table>

**Table 5: Requirement F.5**

<table>
<thead>
<tr>
<th>Title</th>
<th>Business Intelligence and Reporting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.5</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The system must provide a (possibly open source) tool for the business intelligence and reporting activity. The selected tool must provide support to business intelligence, open standards, Big Data processing engine, and NoSQL and distributed databases.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Must</td>
</tr>
</tbody>
</table>

**Table 6: Requirement F.6**

<table>
<thead>
<tr>
<th>Title</th>
<th>Data Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.6</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The system must guarantee support to the most common data format (e.g., specified within the RFC standards) to represent data tables</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Must</td>
</tr>
</tbody>
</table>

**Table 7: Requirement F.7**

<table>
<thead>
<tr>
<th>Title</th>
<th>Interface-service separation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.7</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The components of the reference architecture should provide a clear separation between user interface and service. The consortium, moreover, will favor communication based on RESTful APIs.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Should</td>
</tr>
</tbody>
</table>

**Table 8: Requirement F.8**

<table>
<thead>
<tr>
<th>Title</th>
<th>Data flow management</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>F.8</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>The system should provide a (possibly open source) automation tool to support</td>
</tr>
</tbody>
</table>
data flow (a.k.a. data logistics) among the different components of EW-Shopp ecosystem.

| Req. Stringency | Must |

**Table 9: Requirement F.9**

<table>
<thead>
<tr>
<th>Title</th>
<th>Data Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>F.9</td>
</tr>
<tr>
<td>Req. Description</td>
<td>The system must feature a persistent layer capable to handle data coming in different formats. At least column and graph based databased must be supported. Such layer may consist of either different federated solutions dedicated to a particular database type or a single multi-model distributed database. This component must provide suitable data access API (for table and RDF) compliant with the standards commonly recognized.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Must</td>
</tr>
</tbody>
</table>

**Table 10: Requirement F.10**

<table>
<thead>
<tr>
<th>Title</th>
<th>Data processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>F.10</td>
</tr>
<tr>
<td>Req. Description</td>
<td>The system should provide a (possibly open source) distributed processing tool to support data flow (a.k.a. data logistics) among the different components of EW-Shopp ecosystem.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Must</td>
</tr>
</tbody>
</table>

**Table 11: Requirement F.11**

<table>
<thead>
<tr>
<th>Title</th>
<th>Cloud ready</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>F.11</td>
</tr>
<tr>
<td>Req. Description</td>
<td>The software tools selected to be part of the EW-Shopp system should preferably be web-based in order to be ready for possible future deployment in cloud environments.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Should</td>
</tr>
</tbody>
</table>
Security and Privacy requirements

Security and privacy management is one of the main issues in the Big Data area. A debate is currently taking place within the scientific community and industry to define methodologies and architectures capable of guaranteeing high safety standards [12]. States are also moving in this direction, as evidenced by the recent legislative intervention of the European community in the area of privacy management with regard to sensitive datasets (GDPR [11]).

This section presents those requirements that explicitly relate to security management and privacy in the EW-Shopp system. They concern security aspects, authentication, encryption of communication between the various actors (software or human) involved in the system. During the collection, definition and selection of requirements, the importance of clearly defining these requirements was recognized, for this reason a separate section of this document has been dedicated to them.

Table 12: Requirement SP.1

<table>
<thead>
<tr>
<th>Title</th>
<th>Authentication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>SP.1</td>
</tr>
<tr>
<td>Req. Description</td>
<td>Authentication is the process of confirming the identity of an actor in order to avoid possibly malicious access to the system resources and services. Authentication can be defined as the set of actions a software system has to implement in order to grant user the permission to execute an operation on one or more resources. The EW-Shopp ecosystem should support the most common authentication technologies and standards.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Must</td>
</tr>
</tbody>
</table>

Table 13: Requirement SP.2

<table>
<thead>
<tr>
<th>Title</th>
<th>Authorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>SP.2</td>
</tr>
<tr>
<td>Req. Description</td>
<td>Authorization is the process that implements the access control by specifying access privileges to resources. In particular, suitable access policies should be configured within the system in order to permit selective access to resources in form of data and programs.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Must</td>
</tr>
</tbody>
</table>

Table 14: Requirement SP.3

<table>
<thead>
<tr>
<th>Title</th>
<th>Encryption</th>
</tr>
</thead>
</table>
Encryption is the process of encoding messages in order to set up secure communication channels. Encryption, generally is based on the concept of decoding key, which is a piece of information able to decode the messages. Without the key, the message results unintelligible. EW-Shop consortium is committed to employ modern encryption mechanisms as HTTPS, SFTP, SSL, and TLS.

**Table 15: Requirement SP.4**

<table>
<thead>
<tr>
<th>Title</th>
<th>Anonymization</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>SP.4</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>With the term &quot;anonymization&quot; we mean the process of data transformation/sanitization aimed at encoding or removing sensitive information from a dataset. Anonymization is required to obfuscate and make unintelligible personal information that would allow the identification of a person, and important business information that constitutes the competitive advantage of a certain company. The overall EW-Shopp architecture must offer a proper anonymization task/component in completion with the GDPR requirements.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Must</td>
</tr>
</tbody>
</table>

**Table 16: Requirement SP.5**

<table>
<thead>
<tr>
<th>Title</th>
<th>Single Sign On</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. ID</strong></td>
<td>SP.5</td>
</tr>
<tr>
<td><strong>Req. Description</strong></td>
<td>EW-Shopp ecosystem should provide a Single Sign On (SSO) to the wrangling, analyzing and reporting solutions. Single sign-on is a functionality of an access control system that allows a user to execute the authentication process once, getting access rights valid for multiple software systems or computer resources to which she/he is enabled.</td>
</tr>
<tr>
<td><strong>Req. Stringency</strong></td>
<td>Should</td>
</tr>
</tbody>
</table>
Table 17: Requirement SP.6

<table>
<thead>
<tr>
<th>Title</th>
<th>Monitoring and Auditing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>SP.6</td>
</tr>
<tr>
<td>Req. Description</td>
<td>The EW-Shopp ecosystem should feature suitable mechanisms for monitoring, tracing, and auditing the processes in execution on corporate data. Those three terms refer to the property of tracking (and analyzing) the events happening in the system.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Should</td>
</tr>
</tbody>
</table>

3.3.2 Non-functional Requirements

In software engineering, whilst functional requirements are used to define the behavior the EW-Shopp ecosystem must have, non-functional requirements specify the criteria to be used to judge the operation of the system. In layman’s terms whereas functional requirements define what a system should do, the non-functional requirements specify how the system should be doing it.

Table 18: Requirement NF.1

<table>
<thead>
<tr>
<th>Title</th>
<th>Scalability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>NF.1</td>
</tr>
<tr>
<td>Req. Description</td>
<td>The EW-Shopp approach should be scalable, that is it must be able to store and process large amounts of data. Ideally the system should be able to process increasingly large dataset with a negligible impact on performances; however, a result in which gigabyte-size databases can be transformed and enriched in the order of minutes time can be considered satisfactory outcome for the project.</td>
</tr>
<tr>
<td>Req. Stringency</td>
<td>Must</td>
</tr>
</tbody>
</table>

Table 19: Requirement NF.2

<table>
<thead>
<tr>
<th>Title</th>
<th>Adaptability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req. ID</td>
<td>NF.2</td>
</tr>
<tr>
<td>Req. Description</td>
<td>Integration between the various components of the platform must take place in such a way that they result being weakly coupled. In this way, the final architecture must be flexible so that it can be possible to replace any of the tools with an equivalent one without this having a considerable impact on the</td>
</tr>
</tbody>
</table>
The proposed system must exhibit extensibility properties. This means that it must be designed in such a way to allow in future to add new solutions and new tasks in the process of value creation from data.

The system should exhibit a high degree of usability. In particular, whenever possible intuitive and reactive graphical interfaces should be available to guide the user in the definition of the data grinding process as well as in operations as monitoring and maintenance.

The system should be as much as possible elastic, that is the resource allocated to the it should vary according to the needs. In particular, it would be advisable to implement the necessary measures to allocate resources to system components only for the amount of time strictly necessary.
Chapter 4  Architecture Design

This chapter aims to present and discuss the reference architecture of EW-Shopp by highlighting and discussing, where necessary, the process that led to its definition.

4.1.1 Envisioned Workflow and Actors

This section is devoted to the definition of the workflows that can be implemented on the EW-Shopp system. In particular, the requirements collected show the need for tools to design the chain of transformations that business data must undergo to create value for project partners. Furthermore, once these changes have been defined, we want the partners to have the tools at their disposal for their deployment and execution. This means that the platform must support two types of workflow:

- **Design time**: defining the linking and enrichment process
- **Runtime**: the execution of this process in production.

In the table below the actors involved in operating the EW-Shopp architecture are described:

<table>
<thead>
<tr>
<th>Actor</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyst</td>
<td>The data analyst is the actor in charge to interact with the system to define linking and enrichment process. She/he must also be an expert in data analytics to be able to design suitable predictive and prescriptive algorithms to be applied on the enriched dataset.</td>
</tr>
<tr>
<td>Reporter</td>
<td>From the requirements, it emerges that the EW-Shopp ecosystem must provide an application capable of carrying out business intelligence and visualization activities on a professional level. We believe that special skills are needed to use this tool. For this reason, we have identified in the user of this tool an actor that we have called data reporter.</td>
</tr>
<tr>
<td>Big Data Engineer</td>
<td>Once the transformation process of the business data is defined, it must be carried out on a platform capable of processing large amounts of data. This is generally made up of many components and its use requires specific knowledge. For this reason, we believe that one of the actors involved should be an engineer experienced in Big Data processing.</td>
</tr>
</tbody>
</table>
4.2 Architectural Specifications

The first and foremost of the consortium’s objectives in defining reference architecture is to provide guidelines for the development of the pilots and their subsequent refinement in marketed services. Consequently, the most significant requirement to be satisfied is adaptability, i.e. the architecture that we are about to describe must be able to apply without changes to all the planned pilots. This is indeed challenging as pilots are characterized by use scenarios, workflow and resources that may vary from one another; nevertheless, they all plan to exploit the datasets provided by the project partners (hereinafter referred to as Core Data) and specific solutions for data transformation, analytics and reporting presented in the project Description of Action (DoA) as well as in deliverable D4.1 [1].

For this reason, the various components of the reference architecture have been grouped into three macro areas:

- **Core Data Services**: These components provide access to data made accessible by partners or freely available. These datasets are used in both data linking and enrichment processes.

- **Platform Services**: data preparation, analytics and visualization services. They are located at the top of the reference stack for Big Data architectures. The focus of the project is on these services, their integration and exploitation.

- **Corporate Services**: These components refer to tools for data ingestion, storage and processing, data flow and security management. In general, these components are found at the lowest levels of Big Data architectures and, moreover, in our case they depend heavily on the technologies present at customers' premises.

As mentioned above, the first two groups are the central focus of the project and will therefore be described extensively in this document. As far as the Corporate Services group is concerned, we will limit ourselves to indicating the main components and describing their functionalities in order to provide the reader with a comprehensive overview.

Figure 4 depicts the reference architecture for Big Data proposed by NIST in which we highlighted the components of the EW-Shopp Platform (within the green rectangle) distinguishing them from all the other components (in light red) that are considered as Corporate services. As you can see, the EW-Shopp ecosystem includes three macro components that relate to the preparation, analytics and visualization processes (req. 0, F.2, F.4, F.5). EW-Shopp Corporate, on the other hand, presents the components needed to store large volumes of data, process them in a distributed and scalable way, data ingestion systems, data flow orchestration, and security, i.e. authentication, authorization, encryption and anonymization (req. F.8, F.9, F.10, SP.1, SP.2, SP.3, SP.4, SP.5, SP.6). Core Data services are not included in the figure as they have no counterpart in the NIST architecture.
On the other hand, the focus in Figure 5 is exclusively on the components contributing to the EW-Shopp reference architecture. Unlike the solutions proposed by NIST or BDVA, the architecture that we describe in this document intentionally exclude the infrastructure layer, data ingestion and access layer. While these components will be considered in the implementation phase we believe that presenting and discussing them here would have added no value to the architecture specification, which aims to be especially focused on the large-scale execution of data preparation, analytics, and reporting tools belonging to the application layer (i.e. the NIST Big Data Application Provider).

Using the same color scheme used in Figure 4 as an interpretation guide, we specified the platform components at application level in Figure 5. They are:

- **Data Wrangler (DW)**, which is the component that is designed to enable the user to define at design time the transformations of data cleaning, linking and enrichment. Such data preparation processes will then be carried out by the component referred to as Big Data Runtime.

- **Data Analyzer (DA)**, which is the component that provides a set of predefined tools for predictive and prescriptive algorithms on enriched data.

- **Data Reporter (DR)**, which is the component that allows the user to visualize and analyze from a business point of view the outcomes produced by the Analyzer data.

More details on this component and its implementation can be found in Section 4.2.3 and Section 5.4. As regards the so-called Corporate services, these are gathered in a single macro-component called **Big Data Runtime (BDR)**. This part of the architecture provides the capability to execute transformation operations defined by the Data Wrangler and analytics operations of the Data Analyzer on large amounts of data (req. NF.1, NF.5). In addition, this component provides the data reporter with specific APIs for data access. Within the Big Data Runtime, we recognize (using the terminology proposed by NIST) a sub-component in charge of **Data Storage** and another dedicated to the **Processing** of such data (req. F.8, F.9, F.10). Moreover, we included in the picture a
component responsible for orchestrating the processes to be performed and the data flow (System Orchestration).

The architecture foresees the presence of crosscutting services to enforce appropriate security and monitoring policies (Security Services).

Unlike the previous picture, Figure 5 includes an area (in light blue) that groups together the components for the management of the Core Data (req. F.3). These are accessible via generic REST APIs (req. F.7, F.11, NF.2, NF.3). With no ambition for completeness, the figure shows the APIs for events, weather and products data. Other enrichment sources may include free datasets such as DBpedia.\(^{30}\)

Clearly, since the image in question does not constitute a deployment diagram, having presented the three logical sections of the architecture separately does not imply that they will be deployed independently and/or geographically apart. On the contrary, as we will see later on. The architecture, the control flow, and the dataflow pipeline have been conceived in such a way as to realize different typologies of deployments, from the completely distributed one in which the three logical regions are also physically separated and distant, to the centralized one in which the entire architecture is implemented within the same facility.

---

\(^{30}\) [http://dbpedia.org/ontology/]
4.2.1 The Dimension Reduction Approach

The EW-Shopp system aims to support e-commerce, retail and marketing industries in improving their efficiency and competitiveness through providing the ability to perform predictive and prescriptive analytics over integrated and enriched large datasets using open and flexible solutions (req. F.4, F.10). In addition, these tools must provide a responsive graphical user interface to guide the user in designing data transformations (req. F.7, F.9, NF.4). These observations together with the confidentiality requirement (req. SP.4) lead us to believe that the data transformation design phase has to be carried out on a small and possibly anonymized subset of the initial data. For this reason, we have introduced into the platform a specialized component, named Sampler, whose task is to properly generate this data subset.

The idea of reducing the size of the dataset to be able to handle it more easily is not new [13][14][15][16][17] and is commonly referred to as Dimension Reduction or Big Data Reduction. The approach is rather simple in its general lines; it consists in reducing the size of the dataset by identifying a possible compact representation of it. In this way, the data transformation and data analytics operations in EW-Shopp can be designed and tested on a smaller set than the original data, which should ensure greater responsiveness and efficiency for the applications involved without negatively affecting accuracy.

Figure 6 graphically illustrates how the reduction approach is implemented within the EW-Shopp architecture. First of all, we point out that only the Data Wrangler and the Data Analyzer are affected by this methodology as the data reporter will visualize and inspect data of small dimensions which were obtained as a result of the Data Analyzer results.

As far as the Data Wrangler and Data Analyzer are concerned, the operations associated with the Dimension Reduction Approach are the following:

1. **Creating a reduced dataset (called Sample in the diagram).** Such sample may depend on the particular operation to be carried out (preparation or analytics)

2. **The user operates the application working on the sample.** In the particular case of the Data Wrangler, it also receives in input a collection of recommendations to guide the user in the process of table annotation.

3. **The application generates a machine-readable description of the user’s operations**

4. **The model (of transformation or analytics) is executed on the initial dataset by the Big Data runtime component.**

It is important to note that this choice does not reduce the applicability and genericity of the EW-Shopp solution as, where possible and desired by the user (e. g. in cases where the data to be transformed is manageable and does not require anonymization), the sampling component can be excluded. In this mode, the user is free to work directly against the original dataset.
4.2.2 Core Data Services

The purpose of this section is to introduce the first group of components that make up the EW-Shopp architecture, i.e. those that provide access to the Core Data. Since such Core Datasets can vary depending on the particular deployment we focus here only on those that are considered indispensable by the consortium members. We therefore only refer to the three main datasets, i.e. weather, events and products.

Please note that the Core Data and services guaranteeing access to them have been covered extensively in D1.1 [18]. That is why we will only briefly introduce them in this document to ensure that architecture can be defined as exhaustively as possible.

Weather

This service, in simple terms, is intended to provide access to weather information related to a certain location and time. In addition, the service must also be able to provide weather forecasts for a 240-hour period in the future. For the first 90 hours, the time step is 1 hour, from hours 90 to 144 the time step is 3 hours and from hours 144 to 240 the time step is 6 hours.

The weather information, as said, is also related to spatial information. This means that it can be queried providing in input the opposite corners of a reduced Gaussian grid, which is regular in longitude and almost regular in latitude.
Events
The purpose of this component is to provide access to information on past or planned events. Events are obtained and continuously updated by collecting and processing web communication channels (such as newspapers and social media). Integration of events, within a marketing campaign strategy is strongly connected to the events’ characteristics where each of events can be described as a public assembly for the purpose of celebration, education, marketing or reunion. Also, other classifications are possible. For example, based on their type, events can be classified as:

- Sports events,
- Social / life–cycle events,
- Corporate events,
- Education and career events,
- Entertainment events,
- Political events,
- Fund raising/ cause related events;

By their dispersion level, events can be defined as:

- Global events with impact to local population (e.g. Olympics, World Cups, Tour de France, Dubrovnik Summer Festival),
- Regional events (large local events with great (international) participation, e.g. Marathon Franja, Metal Camp Tolmin),
- Local events;

Products
This component provides an API for access to GFK’s product catalog. GFK provides product data solutions for retailers, resellers, wholesalers, manufacturers and distributors; from product technical specifications and images, to marketing text, merchandizing functions, rich content and more. The product catalog consisting of more than 20 million products, structured and optimized for SEO, across 56 markets and 30 languages. Category/product data include structured datasets with a number of dimensions from product attributes, life cycle, price development, though consumer interaction data within the purchase funnel model divided to interest, research, intent and buy behavior.

To retrieve specific features of products and categories, expressive queries on products and categories using web-based data access methods must be supported. To this end, products and categories should be represented in native web-compliant languages, like the ones proposed in the Linked Data paradigm, where, using a SPARQL [19] endpoint, queries can be made to provide for data extraction. As a result, product data will be published as Knowledge Graphs31 (KGs) using RDF-based standard languages.

Other Reference Datasets
In addition to the presented services, there may be components that provide access to other datasets with geographic information and other Linked Open Data are reference data, which can be

used as bridges to join business and external data sources. Those datasets will be also provided with a SPARQL endpoint.

Especially important for the project is the access to geographical information, including spatial entities at different levels of resolution, e.g., states, regions, cities, districts, locations and attractions, using standard systems of identifiers, and shared geospatial reference systems and vocabularies. There are very common and very used KGs such as GeoNames\(^{32}\), LinkedGeoData\(^{33}\), and DBpedia\(^{34}\) that can be used as base datasets for those services.

### 4.2.3 Platform Services

In this section, we present the components that form the core of the EW-Shopp platform. These tools can be located at the application level, i.e. at the highest level of the Big Data stack. Using the terminology proposed by NIST [5], they cover the areas of Data preparation, Analytics and Visualization. These components are called Data Wrangler, Data Analyzer and Data Reporter respectively.

**Data Wrangler (DW)**

Data to be imported will typically be in tabular format with varying quality with respect to missing values, invalid values, duplicated records, etc. Improving quality through data cleaning is important when different datasets are connected in the final data graph. When data quality is sufficient, the data can be transformed by selecting the important columns and rows through a filtering process. This can reduce the amount of data and make the dataset easier to store and query. The dataset will further be enriched with data from other domains like event/weather data. Depending on the application area, the dataset can be converted/mapped into graph of data. The mapping must be done according to an ontology and vocabulary for the data set. The mapping/annotation step is typically the most time consuming. The data is stored in an appropriate format for the receiving applications. A sample dataset is used for specifying and assessing the transformation and mapping steps using online tools. A transformation model is generated for batch execution on Big Data Runtime against the full dataset.

Figure 7 shows a component diagram of the Data Wrangler tool with component interactions in terms of data exchanging. The image represents the most general case, i.e. where the data to be transformed is very large and managed by the corporate area of EW-Shopp. In this scenario, we apply the Dimension Reduction Approach presented in Section 4.2.1. The components of every logical area of the platform are involved. We have on the left the Data Wrangler Sampler, the summarizer and the engine that are components that interact with the Big Data Runtime to generate the sample to be used to define the transformations, to realize a set of suggestions for the table annotation process based on summarizations of existing knowledge bases or previous annotations, and finally to perform on a large scale the transformations defined by the user, respectively. The transformation process is defined by the user through a graphical interface that interacts with a backend service.

---

\(^{32}\) [http://www.geonames.org/ontology/documentation.html](http://www.geonames.org/ontology/documentation.html)

\(^{33}\) [http://linkedgeodata.org/ontology](http://linkedgeodata.org/ontology)

\(^{34}\) [http://dbpedia.org/ontology](http://dbpedia.org/ontology)
Both for the definition of the entire process (which requires the transformation of the work dataset) and for its actual execution on the business dataset, the Data Wrangler process may need data for disambiguation, linking and enrichment. These data are provided by the Core Data services.

Figure 7: Data Wrangler architecture

**Data Analyzer (DA)**

The role of the Data Analyzer (DA) component is to support all data analysis and modelling needs of the platform. It receives enriched data from the data wrangling stage and produces descriptive statistics offering insight into the data as well as operational data models that support business functionality. Its outputs are visualized by the reporter component.

The main working models of the Data Analyzer are Learning and Prediction, respectively:

- **Learning** – is used when a model has to be built from historical data. Typically, this is a batch job procedure where a large dataset is provided to the analyzer, which processes it and outputs a model performing the analytic function. The latter can be a prediction of some value, such as the number of page views within a certain category of products on a web store, or a most likely category for a given new product. Learning consists:
  - **Data analysis** – This includes several types of analytics tasks. For instance, *classification* of products into categories based on their properties and descriptions, *clustering* of similar products from different sources (stores) and *duplicate detection, time series analysis* of sale and web click through logs with identification of factors that impact trends etc.
  - **Modelling specification** – Platform users are companies that value their internal data and may not be prepared or able to share it outside. The analyzer component needs to be able to form a *modelling specification* using just a sample of the data. This includes the modelling algorithm as well as features used and how to compute them from the data and any other information needed. The specification can then be transported from the platform to the analyzer instance on user premises where the model can be built on the entire dataset following the specification.
• **Deployable models** – Models built by the analyzer answer business needs of the platform user. For use they have to be integrated into the workflow. Their design needs to allow for loose coupling (req. NF.2, NF.3) and quick deployment.

• **Prediction** – This is the production mode of the Data Analyzer component, used for operational deployment of the learning mode output. The model built in the learning stage is given new examples to process. For the two examples from the previous paragraph, these would be data about the weather and event context for an upcoming date with the target web store category and descriptions of new products submitted to a web store. The model processes the new examples and returns the predicted value.

Figure 8 shows the component diagram of the Data Analyzer where the interactions in terms of data exchanging among the elements are highlighted. The image represents the most general case, i.e. where the data to be processed is very large and managed by the corporate area of EW-Shopp. In this scenario, we apply the Dimension Reduction Approach presented in Section 4.2.1. For this reason, the components of the Corporate and Platform logical area are involved. On the left-hand side of the figure there are the Data Analyzer Sampler and the Data Analyzer engine that are components that interact with the Big Data Runtime to generate the sample to be used to define the analytics and to perform on a large scale some user-defined analytics, respectively. The analytics process is defined by the user through a web graphical interface that permits the user to define algorithms and test them on local data.

![Data Analyzer architecture](image)

**Figure 8: Data Analyzer architecture**

**Data Reporter (DR)**
Data Reporter (DR) component, as shown in Figure 5, follows the Wrangler (Transformation and Enrichment) and Analyzer (Analytics) components in the logical workflow foreseen in the EW-Shopp platform. This component should perform different tasks, but one of the most important is to provide H2M (Human to Machine) capabilities, to support users with getting the analytics results in a user-friendly and extremely concise structure, with respect to the volume of processed data, in
In order to extract operational knowledge from the value derived from the analysis. For instance, these insights may support the management team in the decision-making process, by exploiting the past trends and highlighting some favorable conditions. This scenario will be carried out mainly in BC1, BC2 and BC4. The scope should be to develop ad hoc dashboards to highlight specific aspects of the involved data, and make them available and constantly updated to the interested stakeholders. Moreover, in BC3, or other similar scenario, enriched data would be made available to stakeholders outside the platform. Such data may in turn become input for other systems in a M2M (Machine to Machine) collaborative scenario, where the Data Reporter output format will facilitate this knowledge transfer by adopting standard formats (e.g. XML, CSV, etc.).

The main expected features of the Data Reporter component follow:

- to give the opportunity to work with large data sources and traditional ones;
- to allow the creation of dashboards (with different graphic representations) providing an easier and concise data analysis reading;
- to make these cockpits available to users both within the platform and through external link;
- to export offline report in standard formats (e.g. XML, CSV, etc.).

Finally, Figure 9 shows the main components involved in the definition of the business intelligence and visualization tool: the data reporter. Unlike the other two tools that, together, define the heart of the EW-Shopp platform, the data reporter is not based on the Dimension Reduction Approach of the original dataset, presented in Section 4.2.1. This is not necessary because the data to be handled by this piece of software are generally small in size as a result of the Analytics process. In the more general scenario, these data must be moved from the corporate to the platform side of the EW-Shopp ecosystem. To do this, we have a component called the Data Access API which provide access to the analytics dataset. Once the data has been accessed, the user can perform business intelligence operations and design the final report through a graphical web interface that communicates with a service tier.

![Figure 9: Data Reporter Architecture](Image)
4.2.4 Corporate Services

As mentioned above, in the reference architecture we impose a logical separation between components by dividing them into platform services and corporate services. This structure has been conceived to make the architecture more flexible and adaptable to different deployment scenarios. In our opinion, this is necessary in order to implement the lean methodology in the development cycles of marketed services.

We have introduced the platform components in the previous section, in this section we focus on the services of the corporate section. In the more general case, these components are physically distant from the platform components and interact with them via APIs. In the early stages of pilot development, we do not rule out the possibility that the interaction between the components of the two logic areas can be carried out manually by means of one or more human users.

**Big Data Runtime (BDR)**

The Big Data Runtime (BDR) component’s role in the EW-Shopp solution is to perform batch data cleaning, integration, enrichment and data transformation on large amounts of input data. The objective of the workflows is to eventually expose the data in a central database for performing analytics.

To fulfil the aforementioned role in the EW-Shopp platform, the Big Data Runtime component needs to support the following main functionalities:

- **Deployment in a cluster configuration** – in order to handle the scale of data, the Big Data Runtime component has to be able to distribute the individual workflow steps over a cluster of virtual machines (VMs) or containers. The cluster has to be deployed on the premise of the EW-Shopp platform user to maintain the confidentiality of their data. The cluster also needs to provide access to the input data to all members of the cluster, for example by deploying data are on a reachable network location or on a mounted shared file system.

- **Execution of the data flow** – The Big Data Runtime executes a dataflow pipeline consisting of the steps specified by the users. To provide an overview of the execution and feedback on the results, the workflow is defined and displayed through a user interface. The data workflows need to be reused when new data comes in the shared input data steps. The data workflow steps need to be able to be scaled up and down, automatically, or semi-automatically, according to their load.

- **Data Storage** – The Big Data Runtime deploys the output of the data workflow in a central database, which is accessible through the network, or deployed on the same cluster.

The tools and services for providing analytics and reporting functions to the EW-Shopp consume the data produced by the data workflow executed by the Big Data Runtime (BDR) through the central database.
4.2.5 Crosscutting Services

The section aims at introducing the crosscutting services, that is those components that are in charge of implementing functionalities that span over several logical areas of the architecture.

Security Services

EW-Shopp reference architecture encompasses also security and privacy solutions as their enforcement assume paramount importance in Big Data; for this reason, we approach those aspects focusing on suitable Privacy-Enhancing Technologies (PETs) [20] to implement suitable Authentication, Authorization and Encryption mechanisms as stated by requirements SP.1, SP.2, SP.3, SP.4. Specifically, seeking for flexibility we suggest the adoption of security protocols and tools implementing the role-based access control model. It binds the authentication process to depend on the actor’s role. Moreover, the authentication and authorization mechanisms must be integrated with the customer’s IT platform.

Securing communication assumes a paramount importance, as no trustworthy authentication and authorization mechanism can be built in a distributed platform without the establishment of a secure communication channel. For this reason, EW-Shopp consortium is committed to employ modern encryption mechanisms to create secure channels over untrustworthy communication means.

It is important to consider that security services must have competence over the entire architecture, regardless of its deployment. For this reason, if the system is distributed, then with a separation between platform and corporate services, it will be necessary to provide an adequate security system in both sub-systems and ensure that they can interact transparently through some sort of security proxy.

Monitoring and Auditing Services

Finally, Corporate services should include a set of components responsible for monitoring the execution of the data transformation process (req. SP.6). These services are not among the core elements of the platform as their absence would not limit the functionality of the system but are nonetheless very important for its day-by-day operability. In fact, a system that does not allow for the timely detection of anomalies in processes or security breaches is certainly not operational.

For this to be possible, the system must be observable, i.e. its various components must be properly instrumented to generate audit events of adequate granularity. These events are usually recorded in several log files (but a message queue system can also be used for this purpose) and analyzed by a Complex Event Processor (CEP) [21].

4.3 Control Flow

Figure 10 illustrates a generic control flow associated with the data preparation process. The image entails both the design time and runtime phase. During the design time phase the authenticated and authorized user (a Data Analyst) designs the transformation process the dataset must undergo. To
do this, the user generates a sample dataset according to specific domain criteria and manipulates it by interacting with the Data Wrangler Web App. Once the process is fully defined it can be downloaded in a suitable format from the Data Wrangler backend to EW-Shopp Corporate. Here the runtime data flow can be set up and fired by loading and executing the generated transformation model into the Data Wrangler Engine. This component is in charge of executing the transformation (cleaning, linking, enrichment) against the overall Big Data set. The results are saved back in the data storage. Please notice that, in the figure, the Corporate workflow management tool represents a suitable automation tool implemented within the Big Data runtime.

![Diagram](image)

**Figure 10: Generic control sequence for Data Wrangling.**

Similarly, the control flow for the data analytics process is graphically represented in Figure 11. Once again, the system user must have previously logged in and been authorized. Afterwards, the user can then use a sample, generated by the Sampler from the enriched dataset, as a playground for the definition and configuration of the analytics models. These models can cover the various domains of data analytics including predictive and prescriptive methods. The user interacts with the system through a suitable graphical interface. After satisfactory results have been achieved, the analytical models can be downloaded to the Corporate platform and executed on a large scale on the complete enriched dataset.
Figure 11: Generic control sequence for Data Analytics

Regarding the execution flow of the data reporter, this slightly differs from the other two tools. In fact, the data reporter does not require the use of a reduced set of the enriched data but is able to work on the results of the analysis tool as these are generally small-sized. In Figure 12 we see how in essence the user can only interact with the graphical interface of the data reporter to generate visualizations and reports. We point out that since the definition of the Data Reporter architecture (Figure 9) we have adopted the option to let it interact with the data storage solution by means of suitable APIs. If this was not possible, we could still opt for a fallback solution in which the user uploads a file containing a dump of analytics data onto the Platform.

Figure 12: Generic control sequence for Report Visualization
4.4 Data Format

To enable simple data exchange, data are kept within a CSV format (req. F.6) following further rules:

**CSV file format rules**

- Data fields are separated by a single character (usually “Comma”, can be “Tab” or “Pipe” (|)),
- Each record is kept in a separate line, starting in its own line, but can span multiple lines,
- The last record is not followed by “Carriage Return”,
- The first line of the file should include the Header with the list of the column names used in the file (optional, but strongly recommended – enables self-documenting of the file),
- The Header list should be delimited in the same way as the data within the file. This helps guard against the data fields being transposed in the data when it is loaded, which can lead to getting wrong answers when you query the data.
- Enclosing character (typically double quotes) must be used when required, such as when the delimiter appears in a field;

More strict compliance according to [RFC 4180](https://www.rfc-editor.org/rfc/rfc4180) may be considered.

The system is supporting international character encoding:

**Character set**

- UNICODE UTF8,
- With BOM Header;

File Names of uploaded files are standardized:

**File Names Convention**

The uploaded file name contains:

- Company GUID,
- Date (Some file systems change the dates when copying file) in a form YYYYMMDDhhmms,
- Batch GUID if there will be multiple uploads through the day;

Each data is described by its global identity ([GUID]). Standard column names can have `$EWS_` prefix.

Each data access and manipulation is recorded as **Audit trail log** (req. SP.6) what enables tracking of the processes related to entities such as customer, user, sales object, campaign, etc.
4.5 Mapping the Reference Architecture on the Business Cases

The platform may be instantiated differently in different business cases (for example, likely, BC3 does not need to have the analytics part available), for this reason, we ask the representatives to imagine a specific deployment profile and data flows for their case.

Based on the feedback obtained from the partners involved in the definition and implementation of the pilot projects, we have endeavored to imagine the future implementation of the pilot in terms of the reference architecture presented in this chapter.

However, a brief disclaimer is necessary: at the time when this document is written we are still six months away from the actual realization of the pilots, this means that many aspects may be adjusted. Nonetheless, the effort made at this stage, we believe, is not just vain intellectual exercise but a fundamental starting point define a common language (components and relations) and a ground basis architecture around which the pilots will be built.

4.5.1 BC1 – Integration for Consumer Journey

In its pilot phase, the EW-Shopp project foresees three pilot projects:

- Pilot I – Enrichment of purchase information for web platforms,
- Pilot II – Integrated platform for category and marketing optimizations (B2B),
- Pilot III – External data access API and decision-making systems supporting customized campaigns;

All pilots are described in the project deliverable D4.1 Chapter 5. (Technical and syntactic interoperability requirements). Results of the EW-Shopp projects should reflect in the availability to offer the EW-Shopp services to different industries and their end-customers.

BC1 enables the EW-Shopp partners (BB, CE and BT) integration of their EW-Shopp Corporate Services with:

- EW-Shopp Platform Services on one side
- Their business partners (industries) environment on the other side (via the Local Information System integration API);

as described in chapter 4.2.4.

All EW-Shopp BC1 solutions are provided by the programming infrastructure and interfaces defined within other chapters of this document. The solution architecture is presented in Figure 13 to describe the platform (already sketched in D4.1.) for business case 1 (BC1) in detail by using the nomenclature and components detailed in Section 4.2.
In addition, we have broken down the data flow and highlighted the need to manage the corporate (i.e. business) data generated by the linking and enrichment process and those resulting from the analytics phase. As far as the Platform area components are concerned, an initial analysis of the pilots did not lead to a real need to include the Data Reporter tool. A component specific to the use case will ensure access to data for customers. This decision can certainly be questioned in the coming months, during which the pilots will be implemented.

4.5.2 BC3 – Location Scouting

Figure 14 refers to business case 3 (BC3) and this was derived from the textual description and an initial representation contained in document D4.1. We have therefore refined it by defining the data flow and we have described it in terms of the reference architecture. The figure shows two sections dedicated to the data preparation. In the first, which is part of the Corporate area of the system, the data coming from the sensors are collected, cleaned and filtered; in the second, the resulting dataset is enriched with weather information, products and events. To complete this second process, the Data Wrangler and Core Data are used (light green and light blue, respectively).

The data collected and enriched are stored in Apache Cassandra\(^\text{35}\), a columnar NoSQL database [22] [23], particularly suitable for handling huge amounts of data arranged in table form. Cassandra natively provides data partitioning and replication capabilities. As for the data processing component, the company currently uses another open source application from the Apache community: Spark. Apache Spark\(^\text{36}\) is one of the most widely used general purpose processing engines in the world. It enjoys a high reputation in the Big Data world and is currently used by the world’s leading companies to process their corporate data. Apache Spark allows distributed

---

\(^{35}\) [http://cassandra.apache.org](http://cassandra.apache.org)

\(^{36}\) [https://spark.apache.org](https://spark.apache.org)
processing of large amounts of data, has a flexible and comprehensive programming model, and includes a library with many machine learning algorithms.

The data once linked and enriched are subject to analytics algorithms. As for this phase, at the time this document is written, we expect it will be managed by means of software and algorithms specifically implemented by the company. Similarly, the business intelligence and visualization phase will be directly managed by components created by Measurence. In particular, there are APIs and a specially built web dashboard.

![Figure 14: BC3 Architecture](image)

### 4.5.3 BC4 – Campaign-driven Purchasing Intentions

As in the case of the two previous business cases, we have refined the architecture presented in deliverable D4.1 [1] in the light of a greater awareness of the elements making up a Big Data architecture. In order to do this, as in previous cases, we have availed ourselves of the collaboration of the company most involved in the business case: JOT. We have therefore succeeded in outlining the data flow, deployment (the Microsoft cloud: Azure[^37]) and the requirements in terms of platform components that are currently expected to be part of the business case architecture. The result is shown graphically in Figure 15.

Jot collects data from the APIs made provided by Google and Microsoft via a download agent. At this stage, data are collected, cleaned and aggregated. They are then loaded into a data store in Azure.

[^37]: [https://azure.microsoft.com](https://azure.microsoft.com)
Once the data has been transferred to the cloud, it can be processed using one of the tools provided by Microsoft (Hadoop\textsuperscript{38}, Storm\textsuperscript{39}, Spark\textsuperscript{40}, among the others) or by the recommended component of the EW-Shopp consortium. For further details, please refer to Section 5.5 of this document.

The data flow continues with the transformation and enrichment process, which makes use of the Data Wrangler and the Core Data (used both in definition and execution of the process). As with the other business cases, it is expected that the volume of data to be processed will be considerable and, therefore, it will be necessary to use a sample of data. Similarly, the Data Analyzer is applied on a sample of enriched data to define the analytics process that will then be applied to the entire dataset.

Finally, the Data Reporter accesses the results produced by the Data Analyzer generating a report that, in turn, is used to optimize the campaign.

\textbf{Figure 15: BC4 Architecture}
Chapter 5  Reference Implementation

This chapter provides a brief description of the initial system implementation. Details on the implementation are described in EW-Shopp Deliverable D2.2 [3].

5.1 Overview

As mentioned earlier in this document, EW-Shopp ecosystem is logical divided into three areas, Core Data, Platform, and Corporate, respectively, plus a set of crosscutting services.

The EW-Shopp Core Data are dataset that can be exploited in the linking and enriching phases of the dataflow pipeline. These data are made available via standardized APIs by special services that can be deployed locally or accessible through the Web. Weather and Events information are served by JSI by means of the EventRegistry\(^{41}\) and a Python API\(^{42}\) whereas Product data and other freely available dataset will be served via suitable APIs (such as SPARQL endpoints [19]). Please refer to Section 5.3 for more details.

The EW-Shopp Platform mainly features three functional components:

- **Data Wrangler** – This component is implemented by means of integration of three software solutions: DataGraft, ASIA and ABSTAT. Data transformations are developed online using DataGraft\(^{43}\) online tools used for data cleaning, transformation and enrichment using a sample data set. DataGraft uses ArangoDB\(^{44}\) for storage of transformed sample data. A model is exported as a Docker\(^{45}\) image that can be executed on the runtime framework with the complete data set as input. ASIA\(^{46}\) is a tool for semantic enrichment of data tables. It aims at guiding the users of the EW-Shopp ecosystem in integrating business data with the Core Data. Semantic reconciliation algorithms are implemented into the interface to help users map the data schema to shared vocabularies and ontologies, and link data to shared systems of identifiers. Data enrichment widgets exploit these links to shared systems of identifiers to ease the extraction of additional data from third-party sources and their fusion into the original tabular data. ABSTAT\(^{47}\) is a tool to profile KGs represented in RDF based on linked data summarization mechanisms. The profiles extracted by ABSTAT describe the content of KGs, using abstraction (schema-level patterns) and statistics. The profiles results in annotation suggestions that help users understand the content of the KGs used in the

---

\(^{41}\) [http://eventregistry.org](http://eventregistry.org)
\(^{42}\) [https://github.com/JozefStefanInstitute/weather-data](https://github.com/JozefStefanInstitute/weather-data)
\(^{43}\) [https://www.datagraft.io](https://www.datagraft.io)
\(^{44}\) [https://www.arangodb.com](https://www.arangodb.com)
\(^{45}\) [https://www.docker.com](https://www.docker.com)
\(^{46}\) [https://github.com/UNIMIBInside/ASIA](https://github.com/UNIMIBInside/ASIA)
\(^{47}\) [https://github.com/siti-disco-unimib/abstat](https://github.com/siti-disco-unimib/abstat)
platform (e.g., linked product data), support ASIA semantic reconciliation algorithms, and provide data quality insights.

- **Data Analyzer** – EW-Shopp Data Analysis consist of Learning and Prediction processes, which are provided by QMiner data analytic platform. While the Learning process defines an analytics model based on existing examples, the Prediction process uses this model processes for returning the predicted value of new examples. It is noteworthy that its role in the platform is not specific to QMiner and it could be interchanged with a different analytics component. Thus, in the text any QMiner mention is interchangeable with the more general term: Analyzer.

- **Data Reporter** – Data Reporter refers to the delivery of interactive dashboards or static reports by using a module of the Knowage Suite\(^\text{48}\), that combines traditional data and Big Data sources into valuable and meaningful information. The Knowage Suite is developed using a modular approach, where each module is designed for a specific analytical domain. They can be used individually as complete solution for a certain task, or combined with one another to ensure full coverage of user’ requirements, allowing to build a tailored product.

The Corporate services, on the other hand, consists of:

- **Big Data Runtime**– This macro-component targets processing of datasets that cannot be processed by the cloud services provided due to data volume or data confidentiality. It is an execution environment for transformation and analytics models. To handle data owners that requires in house storage, the platform is delivered as a set of Docker\(^\text{49}\) images deployed on a scalable host machine providing data access through a network filesystem.

Crosscutting concerns relate the management of security policies and monitoring (which embrace more than one area altogether):

- **Monitoring** – supporting the monitoring of all the events produced in the dataflow pipeline is crucial for any production environment. Implementation of system monitors can be done on different layers. In order to maximize the usage and minimize the implementation complexity, the usage of standard SYSLOG messaging logging (permitting the consolidation of logging data from different sources in a central repository) and REDIS services is foreseen and recommend.

- **Security and privacy** – Security (authentication, authorization, encryption) and privacy management (data confidentiality) are two features of paramount importance that cannot be implemented in just one of the logical areas of the EW-Shopp system. The consortium considers that it is necessary to create an ecosystem that is globally safe and to take this objective into account from the earliest stages of the project. For this reason, we believe that it is necessary to unify (where possible) security techniques and protocols and to create a service capable of managing, in a transparent way to the user, roles and permissions for the resource access.

---

\(^{48}\) [https://www.knowage-suite.com/site/product/enterprise-reporting](https://www.knowage-suite.com/site/product/enterprise-reporting)

\(^{49}\) [https://www.docker.com](https://www.docker.com)
5.2 Integration Strategy

As far as the integration of the various components that contribute to the creation of the EW-Shopp ecosystem is concerned, we must distinguish three main aspects:

- Integration of the components that form the EW-Shop platform, namely Data Wrangler, Data Analyzer, and Data Reporter.
- Integration of the components comprising the Data Wrangler,
- Integration between the various areas of the ecosystem, i.e. EW-Shopp Platform, EW-Shopp Corporate, and EW-Shopp Core Data.

In the first case, we chose to perform a loose integration in which mainly the various tools communicate exclusively through a common database. In this way, we have the benefit of greatly decoupling the components and thus ensuring maximum flexibility during both deployment and use (req. NF.2, NF.3). In addition, the components can be developed in parallel in complete isolation.

As for the structure of the Data Wrangler, this consists of a web app (GUI) that interacts with a set of services (Grafterizer\(^50\) and DataGraft, mainly). A component called ASIA has been added to the GUI to provide the user with a tool for the table annotation process (more details are available in Section 4.2.3). ASIA is one of the components that are being developed within the project and is currently fully integrated in the graphical interface of the Data Wrangler. Integration was not only technological but also methodological in nature.

Finally, integration between the various logical areas of the architecture is essentially achieved through the use of REST APIs. This is particularly evident in the case of Core Data that can be consumed by both the Platform and corporate areas. The integration between the corporate and platforming parties depends on the particular business cases. Here we will simply say that at the moment we plan to use a proxy system that integrates the security aspect and REST API to implement data exchange (such as Sample files, transformation and analytics models). Concerning data access for the data reporter at the moment we consider that this can be implemented using the JDBC\(^51\) API.

5.3 Core Data Services

5.3.1 EventRegistry

Event data is provided by the Event Registry\(^52\) – a global media monitoring platform developed and maintained by JSI. Event Registry collects and processes articles from more than 100,000 news sources from all over the world in 15 languages. Through semantic analysis these articles are annotated with concepts mentioned in the text. Based on these semantic annotations the articles

\(^{50}\) https://github.com/datagraft/grafterizer
\(^{51}\) http://www.oracle.com/technetwork/java/javase/jdbc/index.html
\(^{52}\) http://eventregistry.org
are then clustered into events, i.e. groups of articles about the same real-world happening. In this text, we will commonly use the term “event” interchangeably for the actual event and the group of articles it is represented by.

The Event Registry website offers a wide array of options for manual exploration of event data with a powerful search engine and various visualizations. The data is also available through an open-source Python API\(^{53}\). The API supports all the functionality of available on the website and has been made available to all project partners without restrictions. The data format used by the API to return data is JSON. Data about the following data types can be requested:

- **events**,  
- **news articles** in the events,  
- **concepts** mentioned in the articles and detected by the semantic analysis,  
- topical **categories** into which the events and articles are classified,  
- and **sources** (i.e. news publishers) of the articles.

The full data model for all data types returned is available in the online documentation\(^{54}\), data models of a subset of the most relevant data types is detailed in deliverable D1.1 [18].

### 5.3.2 Weather API

Weather data is provided by the European Centre for Medium-Range Weather Forecasts (ECMWF\(^{55}\)) an intergovernmental organization supported by 34 states that acts as both a research institute and a 24/7 operational service, producing and disseminating numerical weather predictions. ECMWF is one of the leading meteorological institutions in the world with a decades-long history of weather data collection. They have granted the project access to the Meteorological Archival and Retrieval System\(^{56}\) (MARS), which is the main repository of meteorological data at ECMWF. It contains petabytes of operational and research data ranging back as far as the 1950s, as well as data from special ECMWF projects.

MARS archive is a very rich data source that far exceeds the project needs. Its dataset of operational data contains the complete global state of 145 weather parameters twice per day (12 o’clock and midnight). The weather parameters include measurements such as surface temperature, wind speed, cloud coverage, precipitation type and amount, snow depth etc.

Besides the current state at a given time, the archive also contains the forecast made at that time for the next 240 hours in 125 time points.

The data is available over a RESTful API as well as a Python API. The data is returned in the GRIB (GRidded Binary) format commonly used in meteorology. To avoid overheads and enable efficient work an interface API\(^{57}\) will be developed by JSI for weather data access in the project. This interface|

---

\(^{53}\) [https://github.com/EventRegistry/event-registry-python](https://github.com/EventRegistry/event-registry-python)  
\(^{55}\) [https://www.ecmwf.int](https://www.ecmwf.int)  
\(^{56}\) [https://www.ecmwf.int/en/faq/what-mars](https://www.ecmwf.int/en/faq/what-mars)  
\(^{57}\) [https://github.com/JozefStefanInstitute/weather-data](https://github.com/JozefStefanInstitute/weather-data)
API will transform the raw weather data into a format suitable for integration with the project datasets.

### 5.3.3 Products and Other Knowledge Bases

With regard to the Product data and any other knowledge bases (e.g., GeoName, DBpedia, etc.) that may be relevant both in the linking and enrichment phases, we will proceed according to the guidelines presented in this section.

We have identified two main ways to query these knowledge bases:

- **Full-text search** on countries, administrative divisions, cities, and related information. The access on full-text search is enabled by the interlinking services and geocoding functionality. Traditional indexing techniques for text searching can be exploited to implement this feature;

- **Spatial-query** (e.g. retrieve the name of the location nearest to a given coordinates), enabled by reverse geocoding functionality. Spatial queries, on the other hand, leverages different kind of index, designed specifically for spatial data (such as Quadtree, Octree and R-tree).

In order to maximize the maintainability and reliability, the consortium is inclined to use ElasticSearch\(^{58}\) as indexing engine, since it can manage both full-text and spatial queries (along with other types of queries). ElasticSearch is an open source distributed engine that can be used for indexing, searching and analyzing data. It implements inverted indexes for full-text queries, while it adopts BKD-tree indexes [24] for spatial queries. Another important feature of ElasticSearch is that it scales to handle millions of events per second, while automatically managing how indexes and queries are distributed.

We intend to provide indexes on European countries, administrative divisions and cities that can be found in GeoNames, which are the interesting entities for the EW-Shopp project. In addition, we will store some other useful information that can be used for Data. We will index also coordinates about countries, administrative divisions and cities. This scenario may require the execution of complex spatial queries, in the case ElasticSearch falls short for this duty, we will also evaluate the need of developing a GIS (Geographic Information System), storing data in a spatial database (e.g. PostgreSQL + PostGIS).

With regard to Product data or other datasets that do not require spatial queries, they can be accessed and exploited by means of services that expose a standardized and possibly semantic APIs (such as a SPARQL endpoint). There are several alternatives on the market that can be used for this purpose, such as OpenLink Virtuoso\(^{59}\) and D2RQ\(^{60}\).

---

\(^{58}\) [https://www.elastic.co/](https://www.elastic.co/)

\(^{59}\) [https://virtuoso.openlinksw.com](https://virtuoso.openlinksw.com)

\(^{60}\) [http://d2rq.org](http://d2rq.org)
5.4 Platform Services

5.4.1 Data Wrangler (DW)

The Data Wrangler, whose main functions have been explained in Section 4.2.3, is realized through the integration and collaboration of three independent pieces of software, namely DataGraft, ASIA, and ABSTAT. In particular, DataGraft and ASIA have been integrated at code level, merging the two original repositories together. ABSTAT, on the other hand, collaborates with ASIA through a set of REST APIs. In the following, these three tools are presented separately.

DataGraft

DataGraft is an online tool that consists of GUI operated services for managing the process of transforming data and deploying the data onto a database.

- **Grafterizer** is used for transforming data from tabular format into a graph format. The transformation supports both cleaning and mapping steps. Transformation steps on rows (add, drop, filter, duplicate detection etc.), columns (add, drop, rename, merge etc.) and entire dataset (sort, aggregate etc.) are provided together with visualization of the result after each step. The mapping to project ontology is based on column information after the cleaning steps. The result is formatted as edge and value collections ready to be uploaded to ArangoDB. Grafterizer provides a GUI on top of the Grafter library that’s implemented in Clojure [25] – a language that runs on a Java virtual machine. This makes it easy to generate JAR files that can be executed at customers premise. This will be used to scale transformation execution on Big Data Runtime for Big Datasets.

- **DataGraft** has been updated to include a module called **DBMS manager**, which implements the data storage in ArangoDB. The DBMS manager accesses databases for storage of the transformed data. The manager uses full access privileges for administration of the collections in the database. Additional users with restricted access are handled through the user administration in ArangoDB.

- **ArangoDB** databases, collections, and graphs are handled in DataGraft using registration of ArangoDB assets, which act as the other assets on the platform (e.g., SPARQL [19] endpoints for RDF data). The ArangoDB asset supports create, read, update and delete of collections for a specific database. This is used to integrate the flow from transforming data to uploading the result into collections in the database. The module provides information about the collections when connected to ArangoDB. All information in the ArangoDB assets is synchronized directly with the ArangoDB server through the DBMS manager module. This is done to avoid complicated synchronization when the collections are accessed directly.

- **Query of collections** is supported using ArangoDB Query Language (AQL) through DataGraft query assets or directly using ArangoDB interfaces. AQL is similar to SQL in its purpose when it comes to reading and modifying collection data. Operations such as creating and dropping databases and collections are not supported using AQL.

Support for ArangoDB in DataGraft online tools is implemented in EW-Shopp.
**ASIA**

**ASIA** (Assisted Semantic Interpretation and Annotation of web sources tool) is a tool for the semantic enrichment of data in tabular formats, developed by UNIMIB. Joining tabular data that does not use the same record ids or some other identifying values is not straightforward, it requires to create links from the table values to a shared system of identifiers. ASIA aims to help users in creating these links, by means of especially designed semantic reconciliation algorithms. In this context, the entity linking is performed at two different levels:

- **Schema-level linking**: linking table schema values (i.e. the header of a table) to shared vocabularies and ontologies;

- **Instance-level linking**: linking data values to shared systems of identifiers.

Schema-level and instance-level links are created by ASIA as **annotations** for the table. Users can create schema-level annotation through the ASIA interface, by validating ASIA suggestions about classes and properties to be used. If a user specifies a different class (or property), ASIA suggests classes (or properties) that syntactically match the user input (autocomplete functionality). Otherwise, the instance-level annotations are expected to be created by ASIA automatically, as the large size of the dataset often impedes to validate the values singularly.

Table annotations underpins two different functionalities of ASIA:

- **Generation of knowledge graphs from a tabular data set**: the schema-level annotations are transformed into executable data transformation to publish tabular data as a knowledge graph; data values will be used to create new instances and populate the graph.

- **Enrichment of tabular data with third-party data**: instance-level annotations are used to facilitate enrichment of business data with data from these reference knowledge graphs (e.g., the link to a product in the products datasets supports the retrieval of the product brand, stored in the same dataset) or from third-party data by using links as bridges (e.g., the link to a GeoNames location can be used to retrieve the GeoNames location identifier, that is required for retrieving events from the Event Registry).

The ASIA interface is developed as component of Grafterizer. The ASIA Backend is developed and maintained by UNIMIB. All linking services implemented in the ASIA Backend are made accessible via REST APIs.

ASIA has been designed with a layered architecture. The Business Logic is developed within the ASIA Backend component, while the Presentation Layer contains the ASIA Frontend component. We describe in the following the two main components of ASIA, depicted in Figure 16:

- **ASIA Backend**, which contains the logic related to the annotation suggestion service and that is responsible for the intercommunication with third-party services (like ABSTAT); in particular, ASIA Backend is currently integrated with ABSTAT service via API; as a result, the Backend provides to ASIA, the real-time autocomplete service offered by ABSTAT as well as statistics computed by the same tool.

- **ASIA Frontend**, which is the frontend customized and integrated within Grafterizer.
**ABSTAT**

**ABSTAT** (Linked Data Summaries with ABstraction and STATistics) [26] is a framework developed by UNIMIB to support users and machines in better understanding big and complex RDF datasets. Given an RDF dataset and, optionally, an ontology ABSTAT computes a summary that provides an abstract but complete description of the dataset content. Summaries are published and made accessible via web interfaces, in such a way that the information they contain can be consumed by human users and machines (via APIs). ABSTAT makes also use of a minimization mechanism to keep summaries complete but as small as possible.

ABSTAT summaries provide answers to questions like: what types of resources are described in a dataset? What properties are used to link the resources? What types of resources are linked and by means of what properties? How many resources have a certain type and how frequent is the use of a given property? In practice, ABSTAT summaries describe the use of vocabularies in datasets.

ABSTAT is a backend infrastructure that aims at supporting different tasks:

- **Data understanding.** ABSTAT summaries provide a complete overview of the content of a dataset; this feature was proved to be useful to support, for example, SPARQL query formulation.

- **Vocabulary matching for table annotation.** Summaries record rich statistics about the usage of vocabularies/ontologies in datasets. Thus, we can use summaries to provide types and properties that match a string, for example, the header of a column in a table that we want to publish in RDF reusing existing vocabularies. In addition, statistics provide valuable information to algorithms aimed at suggesting the best properties and types to use when transforming a tabular data into RDF.
Figure 17: Data Wrangler Component Specification

The role of ABSTAT in EW-Shopp platform is to provide suggestions to ASIA for the table annotation process. For this reason, ABSTAT is placed twice in the figure, as it is at the same time the component that provides suggestions about the Core Data (summarizing information of Products, GeoNames, etc.), and suggestion about business data (leveraging information of previous usages).

As regards the Sampler Component, it has not been implemented in this version of the platform as it is strongly business-case dependent and therefore its actual implementation is entrusted to the realization of the single pilots.

It must be noted that the Core Data section of the architecture (graphically represented by the blue box in the figure) does not reflect the actual deployment of the data sources, which means that even if the weather and event APIs refer to a remote source (the EventRegistry), other sources (such as the product data source), may be deployed in the corporate infrastructure or left remote on case-by-case evaluation.

5.4.2 Data Analyzer (DA)

The Data Analyzer component is realized by QMiner\(^{61}\) - an open source data analytics platform developed and maintained by JSI. QMiner was designed for processing of large-scale data streams in real time. It supports both structured and unstructured data including text, geospatial data, networks and time streams.

The QMiner library contains an extensive array of analytics tools covering classification (support vector machine, neural networks, logistic regression), regression (ridge regression, recursive linear regression), clustering (k-means, data partitioning), preprocessing (multi-dimensional scaling, principal component analysis) and others. There are also a number tools directly supporting feature extraction from unstructured data sources such as text.

---

\(^{61}\) http://qminer.ijs.si
QMiner core is implemented in C++ programming language to ensure processing efficiency. It has a custom built internal dataset representation which provides out-of-the-box support for indexing, querying and aggregating using a simple query language. Dataset structure is defined using a simple JSON specification. Feature extraction working on top of the data layer is specified in the same fashion making the setup easily transferrable between QMiner instances. This is useful in situations when final model may be developed on a different instance as the one that has access to all the data as described in Section 4.2.3

For ease of use, QMiner is wrapped in a JavaScript library and is deployed as a node.js package. This makes installation easy through the node package manager. The JavaScript interface allows for fast development and rapid prototyping. Once the models are built they are simple to expose to other processes as simple REST services using one of the many node packages for setting up HTTP servers. This way QMiner models can be easily integrated into other workflows as was specified in the requirements in Section 3.3. Though it does not have a custom graphical user interface, QMiner can be managed in real time using the node.js console or an interactive computing platform such as Jupyter.

![Figure 18: Data Analyzer Component Specification](image)

The two main modes of operation of the Analyzer component are learning and prediction.

**Learning** is typically a batch operation performed by processing a large dataset of historical data. There are alternative approaches to the batch approach, but we do not foresee the need for them in the platform. Should such a need arise in future, it is possible to modify this stage without affecting the prediction stage, thus minimizing the impact of the change.

The dataset is processed by QMiner, which builds a model. Since QMiner parses the data into its own internal representation due to optimization for the learning algorithm, the source of the data (e.g. file on disk or a table in a database) is not important. The only thing needed is the specification of the learning problem, where the target variable and the input features are specified. This also allows the development of the model on a separate sample due to hardware or data sensitivity constraints and then transferring the developed specification to a different QMiner instance (for example on data owner’s premises) where the final learning is run.

---

62 [https://www.npmjs.com/package/qminer](https://www.npmjs.com/package/qminer)
63 [https://jupyter.org](https://jupyter.org)
Conceptually, the output model is a function which maps the input features’ values into the values of the target variable. This function is encoded into a QMiner module which performs this transformation. It can be serialized and transferred to other QMiner instances or saved to disk.

In the **Prediction** mode, the model built using the learning phase is loaded into a QMiner instance which can then receive new examples and perform prediction. The new examples are represented as sets of values of the same features as were used to build the model and the prediction is the value of the target variable as computed by the model function. Prediction is made available for querying to the external processes. Most commonly a simple server is set up using a REST API for access.

The control flow for the Data Analyzer is presented in Section 4.3, Figure 11.

### 5.4.3 Data Reporter (DR)

![Figure 19: Data Reporter Component Specification](image)

The Knowage Suite (developed by ENGINEERING) has been selected as the main background component able to realize the foreseen Data Reporter module in the overall EW-Shopp platform. Knowage is an Open Source business analytics suite that combines traditional data and large data sources into valuable and meaningful information. It merges the innovation coming from the community with the experience and practices from the enterprise-level solutions.

The Knowage Suite is composed of several modules, each one designed for a specific analytical domain. They can be used individually as complete solution for a certain task, or combined with one another to ensure full coverage of user’ requirements, allowing to build a tailored product. Knowage gives the opportunity to work with Big Data sources and traditional ones, federating data sets to build different analysis such as: static reports, maps, network views, interactive cockpits and data/text mining models. Moreover, the user can freely explore his own data using a drag & drop query builder or having immediate insights thanks to advanced visualizations. Advanced data visualization has to show general phenomenon producing insights, before going in depth. With respect to Big Data, this approach becomes central, because the end user needs to be driven towards relevant facts without being swamped with details.

In Knowage, the concept of *dataset* is essential to manage information on the cockpit, OLAP and analysis. This information can have different nature: a DB query, a CSV file or a NoSQL DB collection.
Here below, Figure 20 and Figure 21 show some basic database operations (Creation and Querying, respectively) can be performed both through the GUI web interface (a REST API is available as well).

**Figure 20: Dataset creation**

**Figure 21: Querying the dataset**
5.5 Corporate Services

The envisioned implementation of the Big Data Runtime component contains a set of tools and services that satisfy the platform design, described in Section 4.2.4. In this section, we outline the different components and their role in the implementation. More details on the current implementation and deployment of EW-Shopp ecosystem, inclusive of the Big-Data Runtime, are reported in Deliverable D2.2 [3].

5.5.1 Cluster Configuration

The cluster configuration in EW-Shopp will be implemented with the help of Docker containers, instead of VMs. Thereby, each of the data workflow steps, which produce the integrated and enriched output data, will be packaged in a Docker image, and the necessary number of instances of the image will be deployed in containers. Similarly, to VMs, Docker containers are able to share a single kernel and application libraries. However, containers are more resource-efficient than VMs and perform isolated tasks faster. Furthermore, Docker containers can work in a heterogeneous environment, making them much more flexible than VMs, which need to be coordinated by a hypervisor. The operating environment of Docker is additionally easy to deploy and install on a
proprietary cluster of servers, which is one of the requirements to integrate within the infrastructure of EW-Shopp platform users.

In order to enable easy independent horizontal scaling for each workflow step, the EW-Shopp platform will make use of a container orchestration engine (e.g., Docker Swarm\textsuperscript{64}, Rancher\textsuperscript{65}, Kubernetes\textsuperscript{66}). Container orchestration systems allow for seamless scaling up and down of individual images, which will be used by the Dataflow orchestration component to control the load on the dataflow steps.

\textbf{5.5.2 Data Flow Manager}

Dataflow pipelines will be implemented and manager by the Data Flow Manager component, which belongs to the Big Data Runtime and provides functionalities related to 1) specifying data flow on a high level, and 2) scaling up and down the flow steps (req. NF.1, NF.5). The data flows will consume the input data and transform it – e.g., cleaning up data using Grafterizer, converting the data using its ArangoDB mapper component, as well as other dataset-specific operations that can be customized in the data flow interface. For example, a data pipeline could consist of the following steps:

1. Calling a script to unzip input files stored in a folder and storing the unzipped data in another folder,
2. Cleaning up the output of the previous step by using a pre-defined self-sustained executable generated by Grafterizer
3. Transforming the cleaned-up data to ArangoDB collections using the ArangoDB mapper component
4. Storing the resulting data in the shared ArangoDB database

The Data Flow Manager will be implemented using a data flow automation system, such as Apache NiFi\textsuperscript{67}. The file system that is shared within the cluster will be implemented using a distributed clustered file system, such as HDFS\textsuperscript{68}, GlusterFS\textsuperscript{69}, or Ceph\textsuperscript{70}. It will be used to store input samples (extracted from a corporate database or copied there from other sources), or share intermediate results between workflow steps to be consumed by steps downstream.

\textbf{5.5.3 Enrichment and analysis database}

The Enrichment and Analysis Database implements the central database where the results of the data workflows will be stored. The component will be implemented using ArangoDB, which is a

\textsuperscript{64} https://docs.docker.com/engine/swarm
\textsuperscript{65} http://rancher.com
\textsuperscript{66} https://kubernetes.io
\textsuperscript{67} https://nifi.apache.org
\textsuperscript{68} https://hortonworks.com/apache/hdfs
\textsuperscript{69} https://www.gluster.org
\textsuperscript{70} http://ceph.com
multi-model database that provides document, key-value, and graph data storage and access functionalities.

The Enrichment and Analysis Database is used by the Data Analyzer component, implemented by QMiner to produce analytics models and by the Data Reporter component, implemented by Knowage to output reports.

More details on the Enrichment and Analysis Database and other connected components can be found in deliverable D2.2 [3].

5.6 Crosscutting services

5.6.1 Security and privacy

As mentioned above, project stakeholders are particularly sensitive to security and privacy issues. For this reason, we aim at developing an integrated identity management system that can be used in the various logical areas of architecture.

It is important to note, however, that platform services implement their own authentication and authorization technology (see Deliverable D2.2 for more details). In addition, the services of the Corporate area are highly dependent on the particular business case and therefore it is possible that they also implement solutions that vary from case to case.

We therefore expect that it will be necessary a single sign-on component implementing an identity federation mechanism capable of securely proxying and transparently integrating the various security technologies to the user. In this way, the system would guarantee the Single Sign-On property.

Examples of technologies that could be used to integrate the various security systems are Kerberos ticket-granting tickets (TGT) [27], Active Directory [28], Security Assertion Markup Language (SAML) [29] and OpenID [30].

5.6.2 Monitoring and Auditing Services

By analyzing the logging message, the severity level, facility and the message content, system administrators can use the SYSLOG protocol [31] capabilities and analyses to implement a simple system monitoring tool. SYSLOG servers, in fact, features sophisticated message logging functionalities as well as different routing mechanisms ranging from logging into a text file to distributed logging.

Besides messaging logging functionality, EW-Shopp project aims to support also tracing of key system events and variables to a cluster server (a.k.a. distributed tracing system).
There are several possible alternatives available for the creation of the tracing system. Today, a possible simple implementation can be achieved by using REDIS\textsuperscript{71}, which is an open-source in-memory database implementing a distributed key-value store with optional durability. It supports different kinds of data structure abstractions, such as strings, lists, maps, sets, sorted sets, hyperlogs, bitmaps and spatial indexes. REDIS also supports trivial-to-setup master-slave asynchronous replication, with very fast non-blocking first synchronization, auto-reconnection with partial resynchronization on net split.

EW-Shopp project, foresees to provide support for distributed tracing by means of a solution powered by REDIS. Such a system will support both single value variables and more complex list variables which may be used for storing the entries of repeated events. System variables will be written into the system with expiration period so no additional functionality is needed for deleting the values.

Implementation of system monitoring capabilities based on REDIS server consists of writing a suitable REDIS Client that checks the known system variables, analyzes their values and triggers predefined activities on the base of a project-dependent scripting.

While the SYSLOG and REDIS components collect information about the processed events within the EW-Shopp ecosystem (regardless of the generation location, Platform or Corporate Services) implementing what we refer to as monitoring infrastructure, the Monitoring Agent (see to Figure 23 for more details) is providing a script-controlled event processing phase. Reporting is enabled via Nagios\textsuperscript{72} agents, which is an open source solution that provides different channels of event status alarming (e.g. via Browser, SMS or e-mail) to the operations personal, or write triggers so they will become a part of general escalation procedures.

\textsuperscript{71} https://redis.io
\textsuperscript{72} https://www.nagios.org
5.7 Single-site Configuration

When the full dataset can be managed directly (i.e., Big Data processing solutions are not required), a simplified architecture can be envisaged (see Figure 24). In this configuration, DataGraft will be used to register and manage the Enrichment and Analysis database using ArangoDB assets and the DBMS manager module discussed in Section 5.4.1. The dataflow pipelines can also be implemented in the infrastructure managed by the Data Wrangler component (Grafterizer), which consumes input data directly from the Corporate DB and stores it in the locally accessible Enrichment and Analysis database. DataGraft can be used to directly implement the data workflows, but in case it is needed, a shared Big Data infrastructure can be made available to all components of the EW-Shopp platform. The Data Analyzer and Data Reporter components also have direct access to the Enrichment and Analysis database.

![Figure 24: Self-contained Architecture and Component Specification](image-url)
Chapter 6  Conclusions

The EW-Shopp ecosystem aims to support e-commerce, Retail and Marketing industries in improving their efficiency and competitiveness through providing the ability to perform predictive and prescriptive analytics over integrated and enriched large datasets using open and flexible solutions.

With this document, we have tried to pursue several objectives at once. Mainly, we have outlined a reference architecture specifically designed to handle large amounts of data. This architecture is based on the principle of the dataset Dimension Reduction to work. We have tried to explain to the reader how, by appropriately reducing the size of the dataset, it is possible to design a data enrichment process that is both precise and manageable. The same principle underpins the process of defining analytics algorithms. The proposed architecture natively supports this approach but can also work directly on business data as long as it is not too large.

Our second objective was to raise awareness among consortium members about the inherent difficulties associated with the project and the possible solutions that could be adopted. We have therefore attempted to create a unified common language that, we believe, will have clear benefits in the next stages of the project.

Finally, we have tried to propose an initial implementation of the architecture by identifying one or more tools for each component of the reference architecture. In particular, at this stage we have embraced the lean development approach already introduced in Deliverable D4.1. In our case this has meant that only the centerpiece of the platform, namely the tools already identified and on which an agreement exists among the consortium members, is presented as stable and detailed. The rest of the platform, on the other hand, is constituted by tools and/or technologies that at this moment seem to be good candidates but that can be replaced as the project develops. In addition, business cases can evolve independently and have individual requirements. This could lead to the adoption of different tools to cover the same functionalities. Nevertheless, we believe that the reference architecture and its implementation, however incomplete it may be, will be able to guide the development process that will take place in the coming months.
References


