D2.2 DATA INTEROPERABILITY AND DATA MODEL DESIGN

Deliverable No.:	D2.2
Deliverable Title:	Data interoperability and data model design
Project Acronym:	Fandango
Project Full Title:	FAke News discovery and propagation from big Data and Artificial iNtelliGence Operations
Grant Agreement No.:	780355
Work Package No.:	2
Work Package Name:	Data Access, Interoperability and user requirements
Responsible Author(s):	Sindice (Lead), ENG, LIVETECH, VRT, CERTH, CIVIO, UPM, ANSA
Date:	14.02.2019
Status:	V1.1
Deliverable type:	REPORT
Distribution:	PUBLIC

REVISION HISTORY

VERSION	DATE	MODIFIED BY	COMMENTS
V0.1	18.10.2018	Jeferson Zanim (Siren/Sindice)	First draft.
V0.2	17.12.2018	Jeferson Zanim (Siren/Sindice)	Data model update, Elasticsearch inclusion and architecture update.
V0.3	21.12.2018	Theodoros Semertzidis (CERTH)	Data model specifications, Neo4J description, typographic corrections.
V0.4	27.12.2018	Jeferson Zanim (Siren/Sindice)	Consolidation and updates to conceptual data model.
V0.5	28.12.2018	Livetech	Data model update, elastic search.
V1.0	31.12.2018	Jeferson Zanim (Siren/Sindice)	Final consolidation and update for complete version.
V1.1	14.02.2019	Jeferson Zanim (Siren/Sindice)	Revised to match other partners feedback.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ABBREVIATION	DESCRIPTION
H2020	Horizon 2020
EC	European Commission
WP	Work Package
EU	European Union

EXECUTIVE SUMMARY

This document is a deliverable of the FANDANGO project funded by the European Union's Horizon 2020 (H2020) research and innovation programme under grant agreement No 780355. It is a public report that describes the data interoperability and data model design for FANDANGO.

The main goal of this deliverable is to define the data model for storing data from various sources and providing the conventions in storing data within the data lake paradigm of FANDANGO, ultimately describing how data is curated in different steps of the process.

Data lakes require data integration solutions that can work with structured and unstructured data, likely with schema-less data storage, and with streams of data that should be processed in near real-time. It stores any kind of data in their raw formats as well as results of the analysis in predefined data models allowing the software stack to have direct access to both the original (raw) data and to the processing results, thus being able to simplify and combine data analytics and shorten time to market for new products and services. In other words, data lake requires a completely different approach to data integration and newer technology stack when compared to traditional data warehouses.

Therefore, this document describes the different data ingestions and integrations currently designed, based on the proposed architecture. For each of those, ownership of source and target repository, type of data, access control, persistence period and purpose are asserted.

As the data lake evolves so will its documentation, becoming more descriptive and precise during the lifespan of the project. Therefore, this deliverable is also greatly complemented by content defined on sections of deliverables *D2.1 - Data lake integration plan*, *D3.1 - Data model and components* and/or *Project Progress Periodic Reports* to define more detailed data structures available in each repository and its conventions.

1. INTRODUCTION

FANDANGO's goal is to aggregate and verify different typologies of news data, media sources, social media and open data to detect fake news and provide a more efficient and verified form of communication for European citizens.

To achieve such goal, several different approaches must be used in conjunction to collect a large volume of data. The collection of these datasets is essential to ensure that the Machine Learning algorithms can process the inputs into meaningful information and provide high quality interactions with the user that allows real-time analysis for investigation and validation purposes. Solutions like Spark, which will be used for fast processing of machine learning and graph analysis, needs to work in conjunction with Elasticsearch, that is focused on semantic and statistical computation. Since these solutions use different technologies and require distinct implementations, it is essential that a clear data model is established to ensure compatibility of the data.

Another essential point of the data model is addressing the essential needs of the users, which are described through user stories in D2.3. This is clearly visualized through the conceptual data model, which the abstraction level more closely represents the data structure that users will be interacting with.

Therefore, this deliverable defines a clear data model that will be used across the FANDANGO solution to ensure data can be transferred across different systems, establishing implementation standards for each of its components that utilize structure data that needs to be shared within the system and with the users.

2. ARCHITECTURE OVERVIEW

The main goals of a data model are to support the development of software solution by providing the definition and standard format of the data that will be used. This is necessary to ensure consistency and compatibility of data across systems. If the same data structures are used to store and access data, then the different components of the solution can interoperate in an efficient manner.

To define the data lake data model, it is crucial to analyse the overall architecture of the solution and how data will be collected and processed across different environments. Therefore, the initial architecture overview in Figure 1 serves as base to describe the different parts of the solution.

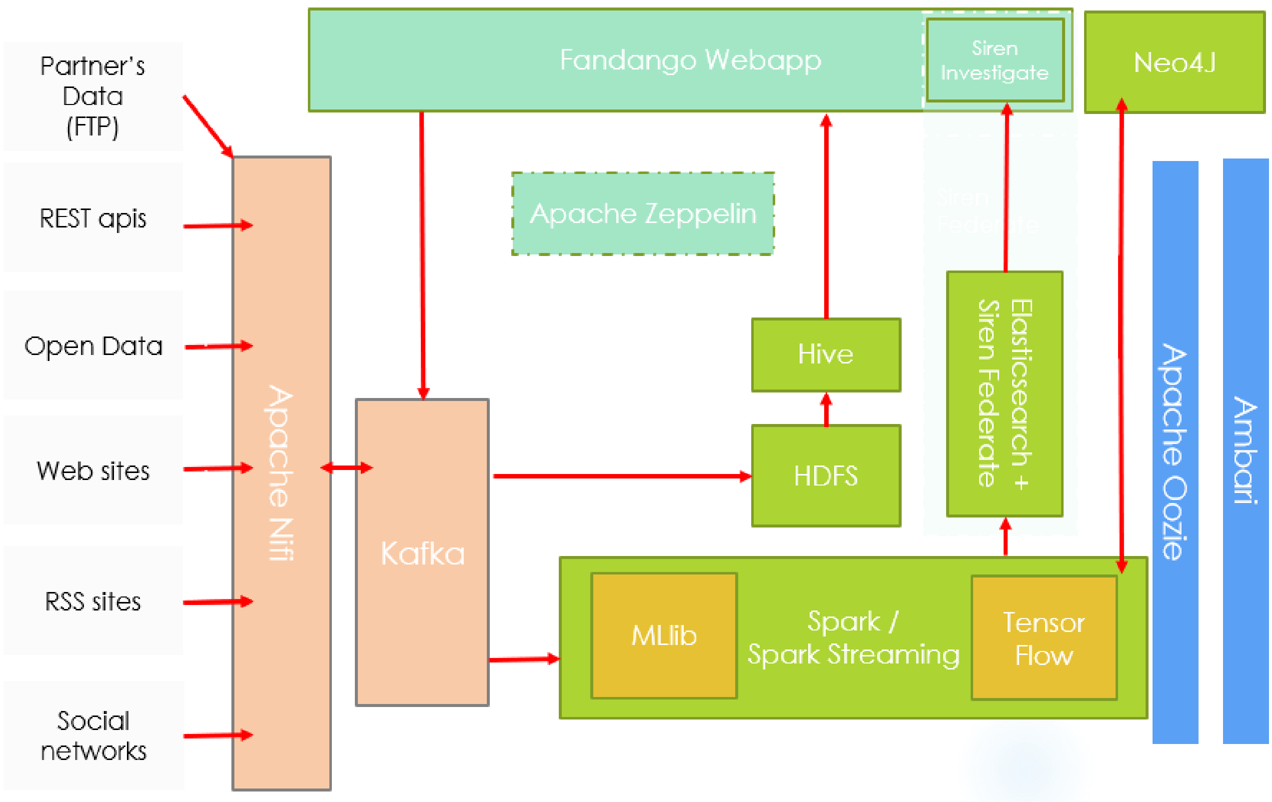

Figure 1 - Architecture Overview

FANDANGO's features to support journalist in fake-news detection and verification, as well as scoring the news with different trustworthiness scores, requires the development of several different big data processing and analyzing techniques. To optimize the solution and better comply to software quality standards, such as: Functional Suitability, Reliability, Operability, Performance Efficiency, Security, Compatibility, Maintainability and Transferability, FANDANGO relies on well-established products that were brought together to form the proposed architecture. The components of the architecture, which needs to be integrated are described on Table 1 – Architecture Components Overview.

SOFTWARE	DESCRIPTION
NiFi	Data flow ingestion tool, open source, distributed and scalable, to model real-time preprocessing workflow from several different sources. It hosts the crawlers.
Kafka	Publish-subscribe distributed messaging system, that grants high throughput and back pressure management. This is the tool FANDANGO uses to connect the different components
Spark	Fast, in-memory, distributed and general engine for large-scale data processing with machine learning (Mllib), graph processing (GraphX), SQL (Spark SQL) and streaming (Spark Streaming) features. This is the core processing engine of the project.

HDFS	The Hadoop distributed file system, open source, reliable, scalable, chosen as storage.
Elasticsearch + Siren Federate	Distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Siren Federate plugin is added to Elasticsearch to allow data set semi-joins and seamless integration with different data sources.
Hive	Query engine (SQL-like language) on HDFS (and HBase) with JDBC/ODBC interfaces.
Oozie	Workflow scheduler. Used to schedule retraining of models.
Ambari	it acts as both a workflow engine and a scheduler. In this case, its main role is to manage the scheduling of Spark jobs and the creation of Hive tables.
Siren	Investigative Intelligence UI with connectivity to Elasticsearch, whose aim is to allow reporting, investigative analysis and alerting to users based on the indexed contents.
Rest APIs, RSS, Web Sites, Open Data, Social network	Data sources of the FANDANGO project. Specific crawlers will connect to these sources of data to get the information needed to verify the news.
FTP	The File Transfer Protocol (FTP) is a standard network protocol used for the transfer of computer files between a client and server on a computer network. In our Architecture it is where Users can place files that will be than ingested in the data lake.
Zeppelin	The notebook dedicated to data scientists, to run in REPL mode scripts and algorithms on data stored in Hadoop.
Web App	Access point to FANDANGO Infrastructure. The journalist will use the FANDANGO Web application to insert news and verify the fakeness of certain publications.
Docker	Docker is used to run software packages called "containers". Containers are isolated from each other and bundle their own application, tools, libraries and configuration files; they can communicate with each other through well-defined channels.
Kubernetes	Kubernetes is an open-source container-orchestration system for automating deployment, scaling and management of containerized applications

Table 1 – Architecture Components Overview

While the overall FANDANGO solution requires all aforementioned components to work in conjunction, only some of those components will store and share structured data across the platform, thus requiring a clear data model structure in order to achieve such goals.

The components highlighted, in green, on Figure 2 identify the specific parts of FANDANGO architecture where structured data will be stored and shared, either with temporary or long-term persistence. These components are essential for long-term sustainability of the solution and will handle the majority of the data required for machine learning processes, knowledge-graph analysis, semantic interpretation and user interaction.

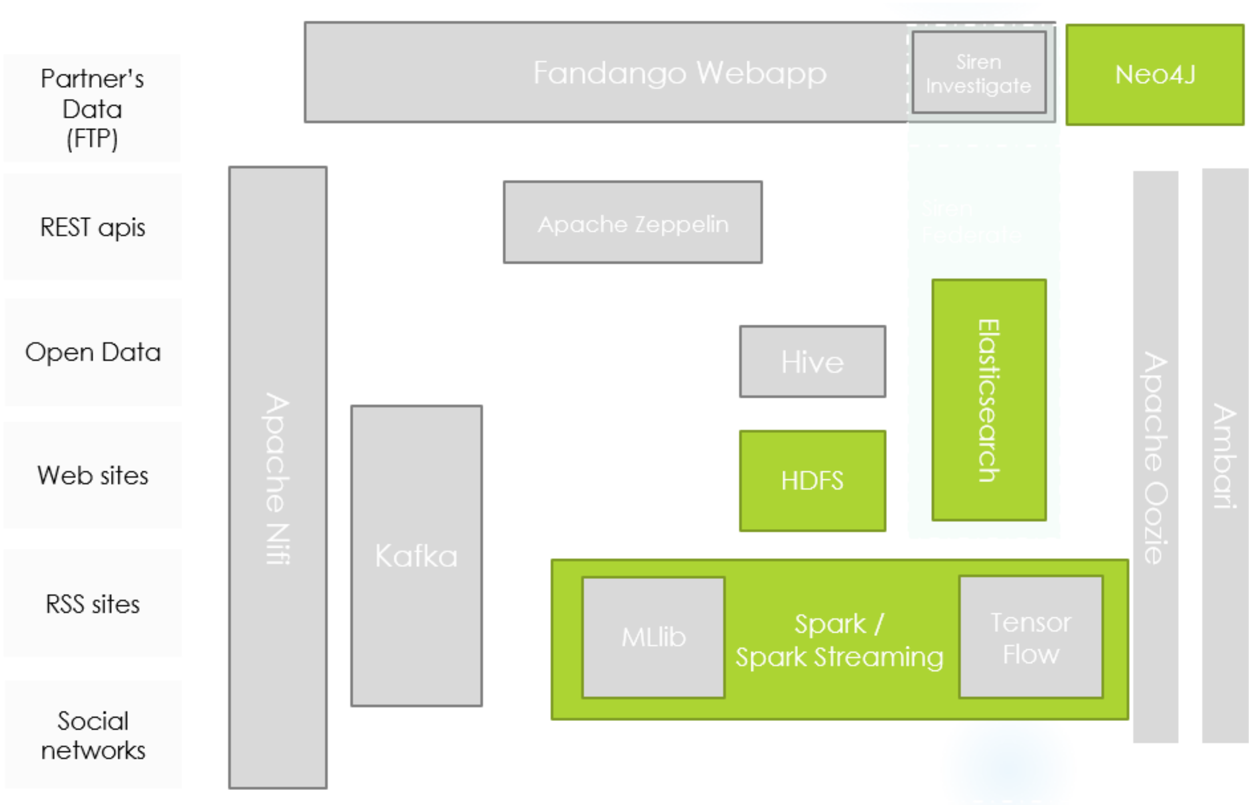

Figure 2 – Structured Data Components

Further sections of this document will detail the logical data model in each of these components, defining the entities that will be used to share data across these systems to enhance the collaboration process between the technical partners and provide better visibility of the solution implementation across the project and external parties.

3. Conceptual Data Model

FANDANGO's data contains several different entities that are used to store information about each of the concepts required to deliver the expected user functionality. A conceptual data model was created so that the overall approach of how the data is handled can be understood by all the parties involved. Figure 3, seen below, contains the main concepts and the main relationships among them to represents the semantics of the data model.

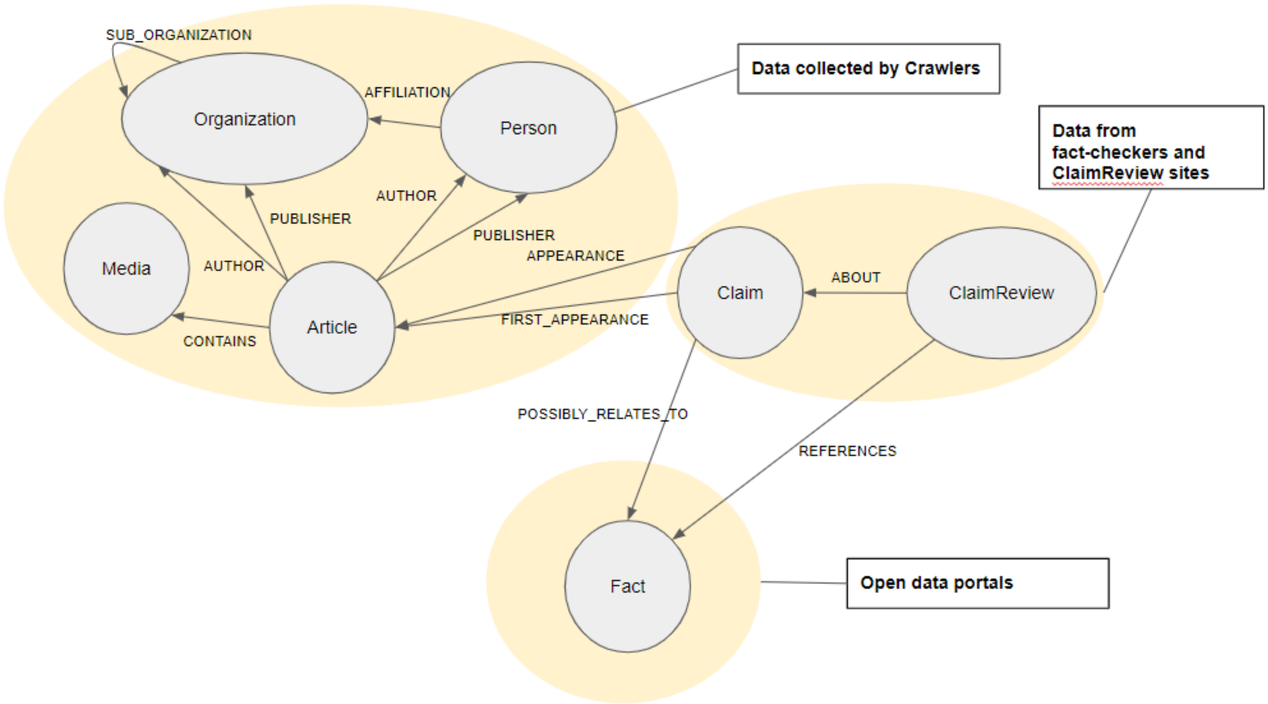

Figure 3 - Conceptual Data Model

Since this conceptual data model represents how the users perceive and interact with the data in the real world, it drives the implementation of each logical data model across the system and ensures that the overall vision is kept, and data can be easily interoperated in the platform.

FANDANGO's conceptual data model was modelled to extend characteristics of standard implementations of schema.org. Each entity in the conceptual schema extends a schema.org entity to improve compatibility with external systems and make the platform scalable and reusable. The list of extended entities can be found in Table 2.

FANDANGO ENTITY	EXTENDED SCHEMA.ORG ENTITY
Organization	Thing > Organization
Person	Thing > Person
Article	Thing > CreativeWork > Article
Media	Thing > CreativeWork > MediaObject
Claim	Thing > CreativeWork > Claim
ClaimReview	Thing > CreativeWork > Review > ClaimReview
Fact	Thing > CreativeWork

Table 2 - Extensions of schema.org entities

Additionally, these structures were specially designed to address the user stories defined on D2.3, like "Search for data sources relevant to statements", "Related articles", "Who is the publisher?", "Who is the author of a news item?", "Is the photo/video new? Has it been tampered with?" and "Third-party factchecking”. These are the primary purpose of the functional design of FANDANGO and, therefore, carefully mapped into the data model to ensure user expectations are met.

4. Logical Data Model

FANDANGO’s conceptual data model is implemented across multiple systems that will be responsible storing, processing and making the data available internally and externally. Therefore, each of the systems involved will have its own logical data model implemented and curated by the responsible partner. Re-use is expected across partners and each system will have a unified data model to avoid data redundancy. Each systems’ implementation will be described in the following section to detail entities and fields as well as how the relationships are structure within the system capabilities and data types.

4.1. Elasticsearch

Elasticsearch is a highly scalable open-source full-text search and analytics engine. It enables FANDANGO to store, search, and analyze big volumes of data quickly and in near real time. It is generally used as the underlying engine/technology that powers applications that have complex search features and requirements, like analyzing large volumes of news data and providing statistical analysis to identify trends and behaviors associated to it.

In FANDANGO’s architecture, Elasticsearch plays a central role as it will be the main repository for the processed data that users will utilize through the web app.

Elasticsearch is the main data storage for Siren Platform, being an essential part of its functionality. Among the distinct capabilities leveraged by Siren platform are the relevance analysis, text search and analytical functions. Complementary, Siren Federate enhances Elasticsearch capabilities and include the possibility of relational analysis through its distributed join features, which is essential for FANDANGO’s use cases.

Data Model Overview

As Elasticsearch is the main target data repository, where the final data is stored for user analysis after being processed, its data model resembles the conceptual data model in its entities as it is directly based on the way users will interact with FANDANGO’s web app.

While entity-relationship structures are often not used on Elasticsearch implementations, since Elasticsearch does not natively provides a way of connecting data in different data sets, in this case, we can benefit from an ontological data model as Elasticsearch is enhanced by Siren Federate’s join capabilities. Therefore, Figure 4 represents the logical data model that will be implemented in Elasticsearch.

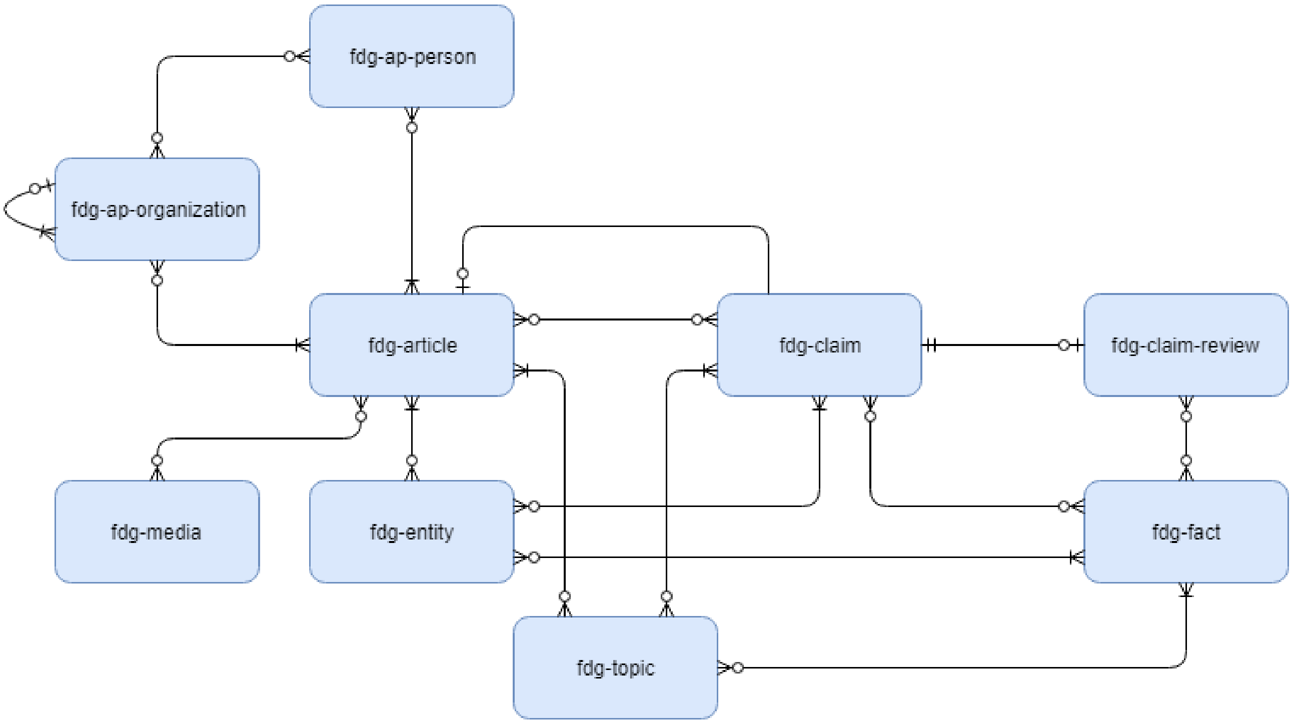

Figure 4 - Elasticsearch Logical Data Model

Differently from traditional relational databases, Elasticsearch allow data fields to contain arrays of different types of data, eliminating the need of joining tables or entities in case of many-to-many relationships when used in conjunction with Siren Federate. This dynamic simplifies the data model and makes it easier for systems to be implemented in a way that is intuitive to the users.

Table 3 describes each of the entities that will be available in Elasticsearch within the Siren data model.

ENTITY	OWNER	DESCRIPTION
fdg-ap-person	Siren	A person, such as a real individual or a fictional persona, like a pseudonym.
fdg-ap-organization	Siren	An organization, such as a school, a NGO, a newspaper, etc.
fdg-article	Siren	An article, such as a news article, piece of investigative report or social media publications. Newspapers, magazines, and social networks have articles of many different types and this is intended to cover them all.
fdg-media	Siren	A media object, such as an image, video, or audio object embedded in an article. Note that a creative work may have many media objects associated with it on the same item.

fdg-claim	Siren	A specific, factually-oriented claim that could be the reviewed in a ClaimReview. Ideally, a Claim description includes enough contextual information to minimize the risk of ambiguity or inclarity. In practice, many claims are better understood in the context in which they appear or the interpretations provided by claim reviews.
fdg-claim-review	Siren	A fact-checking review of claims made (or reported) in some creative work.
fdg-fact	Siren	An Open Data creative work containing data, published by a recognized institution to make information available publicly.
fdg-entity	Siren	An entity that can be mentioned in articles, claims and facts, such as: a person, a place, a landmark or an event.
fdg-topic	Siren	A matter dealt with in an article, claim or fact publication.

Table 3 – Siren Elasticsearch Entities Description

Entity specification

A detailed view of each of the entities is provided below, identifying fields, types, primary keys, foreign keys. Additionally, a description of the purpose of the field and an example of the data contained is added facilitate comprehension.

Data types and structures are specific to Elasticsearch platform and, while the compatibility with schema.org standard is maintained whenever possible, some of the types were modified according to the needs of FANDANGO and the capabilities of the software in question.

Entity: fdg-ap-person			
Field	Type	Example	Description
identifier	integer (PK)	595	Unique number identifier of the item.
name	text	“John Smith”	The name of the person.
url	text	"https://personpage.org/”	URL of the person.
nationality	text	”Swedish”	Nationality of the person.
bias	text		

jobTitle	text	”Financial Manager”	The job title of the person.
gender	text	“Not Specified”	Gender of the person.
affiliation	array of integer (FK fdg-ap-organization)	[10002, 356, 88, 8432]	An organization that this person is affiliated with.

ENTITY: FDG-AP-ORGANIZATION			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	integer (PK)	202	Numeric identifier of an organization.
name	text	“New Org”	The name of the organization.
url	text	"https://orgpage.org/”	URL of the organization.
nationality	text	”irish”	Nationality of the organization.
bias	text		
parentOrganization	integer (FK - AP_Organization)	77	The larger organization that this organization is a sub-organization of, if any.

ENTITY: FDG-ARTICLE			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	integer (PK)	42537	Numeric identifier of an article.
headline	text	“News article headline”	Headline of the article.
articleBody	text	“Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod”	The actual body of the article.
dateCreated	date	2017-12-15	The date on which the article was created, or

			the item was added to a DataFeed.
dateModified	date	2017-12-29	The date on which the Article was most recently modified, or when the item's entry was modified within a DataFeed.
datePublished	date	2017-12-28	Date of first broadcast/publication.
author	array of integer (FK - fdg-ap-person or fdg-ap-organization)	[2931, 5723]	The author of this content.
publisher	array of integer (FK - fdg-ap-person or fdg-ap-organization)	[9845, 1298]	The publisher of the article.
calculatedRating	number	0.898	Rating of trustworthiness calculated by FANDANGO for the article.
calculatedRatingDetail	JSON ({ string:number, ...})	{“publishRating”: 0.99, “authorRating”: 0.873}	Rating of trustworthiness calculated by FANDANGO broken down into different evaluated categories.
about	array of text (FK - Topic)	[“economy”, “brexit”, “eu”]	Topics to which the article talks about.
mentions	array of integer (FK - Entity)	[283, 18289, 84933]	Entities mentioned by the article.
contains	array of integer (FK - Media)	[45821, 239, 5945]	Media files like audio, videos and images that are part of the article.

ENTITY: FDG-MEDIA			
FIELD	TYPE	EXAMPLE	DESCRIPTION

identifier	integer (PK)	202	Numeric identifier of a media object.
contentUrl	text	"https://orgpage.org/”	URL for actual bytes of the media object, for example the image file or video file.
contentSize	text	“15.3 MB”	File size in (mega/kilo) bytes.
uploadDate	date	2017-12-29	Date when this media object was uploaded to this site.
encodingFormat	text	video/mp4	Media type typically expressed using a MIME format.
type	text	77	Type of the media. Valid values are “audio”, “video” or “image”.
calculatedRating	number	0.898	Rating of trustworthiness calculated by FANDANGO for the media.

ENTITY: FDG-CLAIM			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	integer (PK)	762	Numeric identifier of a claim.
appearance	array of integer (FK - Article)	[9765, 34]	Indicates an occurence of a claim in some article.
firstAppearance	integer (FK - Article)	387487	Indicates the first known occurence of a claim in some article.
text	text	“”	The textual content of this claim.
dateCreated	date	2017-12-15	The date on which the claim was created, or the

			item was added to a DataFeed.
dateModified	date	2017-12-29	The date on which the claim was most recently modified, or when the item's entry was modified within a DataFeed.
datePublished	date	2017-12-28	Date of first broadcast/publication.
calculatedRating	number	0.898	Rating of trustworthiness calculated by FANDANGO for the claim.
calculatedRatingDetail	JSON ({ string:number, ...})	{"publishRating": 0.99, "authorRating": 0.873}	Rating of trustworthiness calculated by FANDANGO broken down into different evaluated categories.
about	array of text (FK - Topic)	["economy", "brexit", "eu"]	Topics that the claim talks about.
mentions	array of integer (FK - Entity)	[283, 18289, 84933]	Entities mentioned by the claim.
possiblyRelatesTo	array of integer (FK - Fact)	[9879, 15378]	Facts that are possibly related to the claim.

ENTITY: FDG-CLAIM-REVIEW			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	integer (PK)	762	Numeric identifier of a claim review.
claimReviewed	text	"Claim made by X about the increase in Y and Z"	A short summary of the specific claims reviewed in a ClaimReview.
reviewAspect	text	"Segment X of the claim"	This review is relevant to this part or facet of the claim reviewed.

reviewBody	text	"This claim seems accurate because facts on the publication XPTO corroborates the statement X."	The actual body of the review.
dateCreated	date	2017-12-15	The date on which the claim was created, or the item was added to a DataFeed.
dateModified	date	2017-12-29	The date on which the claim was most recently modified, or when the item's entry was modified within a DataFeed.
datePublished	date	2017-12-28	Date of first broadcast/publication.
aggregateRating	number	0.567	The overall rating of this review, based on a collection of reviews or ratings, of it.
itemReviewed	integer (PK - Claim)	7646	The claim that is being reviewed/rated.
references	array of integer (FK - Fact)	[5476, 976, 8754]	Facts referenced in this review.

ENTITY: FDG-FACT			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	integer (PK)	782	Numeric identifier of a fact.
name	text	"Climate change report 2018"	The name of the fact.
text	text	"Reports show increase in temperatures within the regions …"	The textual content of this fact.
url	Text	"https://opendata.org/a"	URL of the fact.
about	array of text (FK - Topic)	["climate", "usa", "ecosystem"]	Topics that the fact talks about.

mentions	array of integer (FK - Entity)	[283, 18289, 84933]	Entities mentioned by the fact.
dateCreated	date	2017-12-15	The date on which the fact was created, or the item was added to a DataFeed.
dateModified	date	2017-12-29	The date on which the fact was most recently modified, or when the item's entry was modified within a DataFeed.
datePublished	date	2017-12-28	Date of first broadcast/publication.
temporalCoverageStart	date	2017-01-01	The start date and time of the fact.
temporalCoverageEnd	date	2017-12-31	The end date and time of the
spatialCoverage	text	“Brooklyn, NY, USA”	The spatialCoverage of a fact indicates the place(s) which are the focus of the content.

ENTITY: FDG-ENTITY			
FIELD	**TYPE**	**EXAMPLE**	**DESCRIPTION**
identifier	integer (PK)	7748	Numeric identifier of an entity.
name	text	“Mary’s Status”	The name of the entity.
type	text	"landmark”	Type of the entity. Possible values are “place”, “person” or “landmark”, but are not limited to it.

ENTITY: FDG-TOPIC

FIELD	TYPE	EXAMPLE	DESCRIPTION
name	text	“elections”	The name of the topic.

CERTH is utilizing Elasticsearch to store the raw data that are coming from the data shippers and crawlers of WP3. Different crawling scripts are being developed for different data sources and thus a variation in the available data and attributes appears. Elasticsearch acts as a buffering module (intermediate storage) before the data pre-processing step that homogenizes the imported data into the defined data model.

In the data collection step, three different entities are foreseen, based on the type of the identified data source. These are, a) the **news article records** b) **claim reviews** for a news article c) **Facts** coming from **open data** portals. However, no strict data model is applied in this stage, making use of the NoSQL concept of ES to enable collection of any kind of information from the identified sources.

ENTITY	OWNER	DESCRIPTION
News	CERTH	An article coming from a news source e.g. a newspaper, a news portal etc. This entity contains all the content of the article e.g. text, images, videos, metadata etc.
ClaimReviews	CERTH	Claim reviews are collected from factchecking sites that investigate the validity of a claim and provide evidence of the final outcome.
Facts	CERTH	Facts is information coming from official open data portals (either national or international / European) and are used to support the claims and/or provide insights on a news story.

Table 4 - CERTH Elasticsearch Entities Description

4.2. HDFS

HDFS is a Java-based file system that provides scalable and reliable data storage, and it was designed to span large clusters of commodity servers. Due to FANDANGO’s project nature, designed to handle large volumes of data, HDFS was chosen to support its storage capabilities, handling the processing of raw data and data files like images, videos and other media necessary for the development of the solution. It also acts as the backend storage for other solutions in FANDANGO’s architecture, ensuring long-term scalability of the solution.

Due its storage characteristic, HDFS does not have a defined data model in itself but serves as an overall storage solution for the platform.

4.3. SPARK

Spark is a large-scale in-memory distributed processing framework, which can take advantage of the data locally features on the Hadoop cluster. It offers a set of libraries regarding real-time processing, SQL modules graph analytics functionalities and a powerful machine learning package named as MLlib. In addition, it provides a flexible way of coding since distinct programming languages including Python, Scala or Java are supported.

Moreover, Spark can simultaneously access data online natively by loading in memory the data partitions hosted in HDFS, on the same on which the task is executed. On the other hand, it will also be able to access data stored in external repositories where different functionalities for FANDANGO'S will be allocated.

Livetech will use Spark's data model for machine learning classification of news articles. In the training dataset, the article entity needs to have an attribute called label that specifies fake or real nature of the new, based on urls annotated by users.

The label, in case of a news to be evaluated, will be added after the prediction answer from the machine learning model. It should be clear that Spark won't storage any long-term data and in the next section we will identify in the data model which entities and attributes are need to achieve the aim of the Spark use.

DATA MODEL OVERVIEW

LVT will use the data as a consumer to build FANDANGO's classification model, not maintaining any of the structures.

UPM utilizes Spark as the main tool to perform the preprocessing procedure in order to clean the data extracted from the distinct Crawlers and organized in a proper format to be stored and used in a posterior stage during the ML analysis. Moreover, Spark is crucial to face with large-scale data as it is one of the main objectives of FANDANGO's.

DATA MODEL OVERVIEW

Spark is the principal tool to process the data once it has been collected by the crawlers as an early stage to clean and organise it properly. In a similar way as it is described in previous frameworks. Spark has its own data model in order to connect and map all the entities involved in the ontology.

Moreover, in Figure X, a representation of the logical data model that will be implemented in Spark is presented with the aim of improving the understanding of the data work flow at this stage. In such figure, the raw article represents a single sample of the raw data which is collected by the set of Crawlers and pre-processed using Spark. The data model indicates that from this raw article entity, different components are extracted with the purpose of storing in an adequate format before the ML implementations.

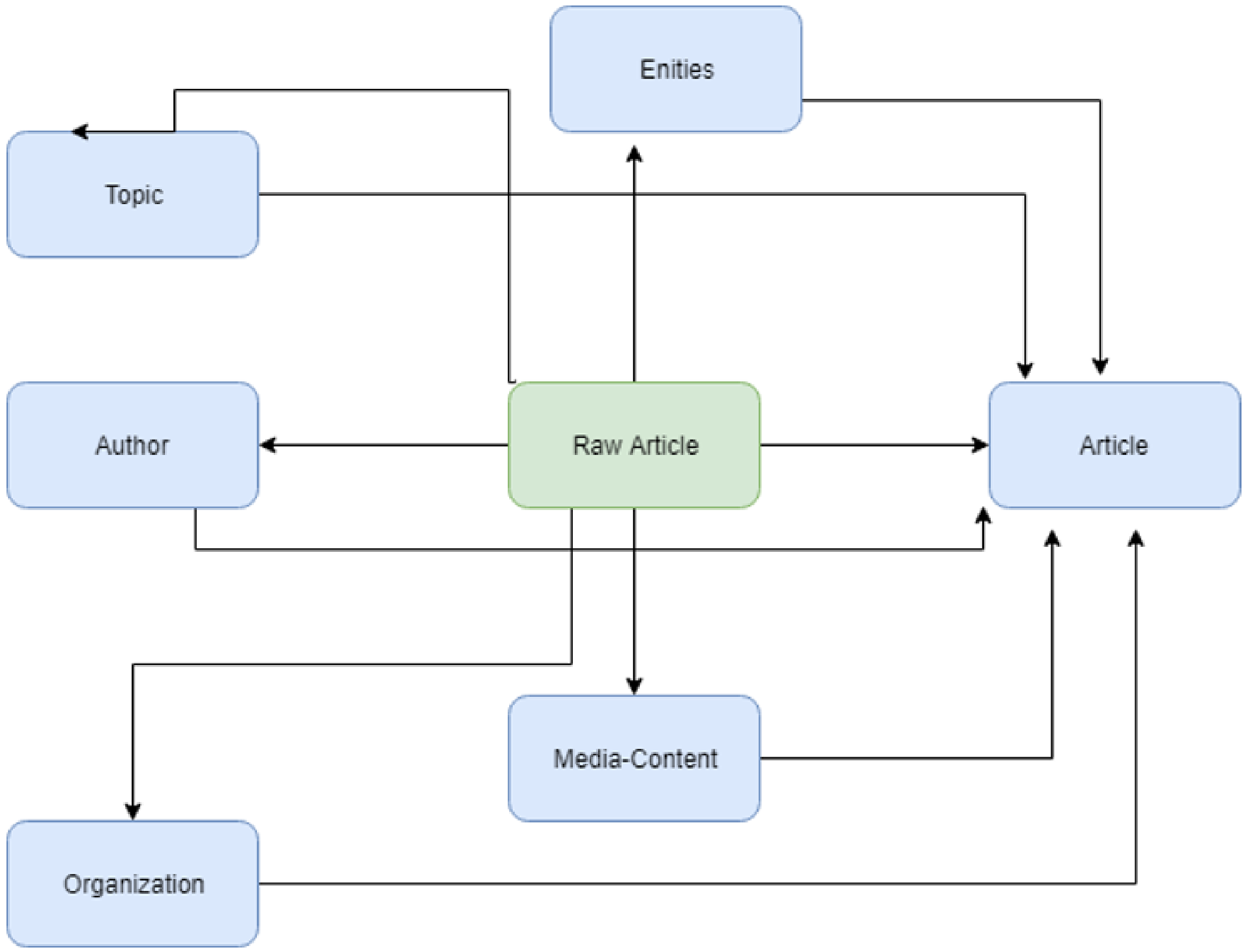

Figure 5 - Spark Logical Data Model

Figure 5 describes each of the entities that will be available in Spark within the CERTH data model.

ENTITY	OWNER	DESCRIPTION
Raw Article	UPM	A raw article that has not been pre-processed yet and has a JSON structure with several fields.
Organization	UPM	An organization obtained after the pre-processing such as a publisher, newspaper.
Author	UPM	The author of the article whether is available after the pre-processing stage.
Article	UPM	An article, such as a news article or piece of investigative report. Newspapers and magazines have articles of many different types and this is intended to cover them all.

Media-Content	UPM	The set of URL's containing images and/or videos that are included in the article.
Entities	UPM	A set of names from places, people or even organizations that are mentioned in the article.
Topics	UPM	A set of keywords included in the article

Table 5 - UPM Spark Entities Description

ENTITY SPECIFICATION

A detailed view of each of the entities is provided below, identifying fields, types, primary keys, foreign keys. Additionally, a description of the purpose of the field and an example of the data contained is added facilitate comprehension.

Data types and structures are specific to Spark framework and, while the compatibility with schema.org standard is maintained whenever possible, some of the types were modified according to the needs of FANDANGO and the capabilities of the software in question.

ENTITY: RAW ARTICLE			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	String	'j1snumMBhU3h5oXprdJl'	String identifier of the raw article.
index	String	'fact_opensources_com'	Elasticsearch index associated to a set of articles.
articleType	String	'article'	The type of content associated to the article such as article, blogs, social media etc.
articleBody	Text	'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et'	The actual body of the article.
articleTitle	Text	'Brussels battens down hatches for no deal.'	The title of the article.
articleSummary	Text	'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et'	A summary of the article.

articleURL	String	'http://feeds.reuters.com/eu-2018'	A URL associated to the article.
articleDomain	String	'http://feeds.reuters.com/'	A URL associated to the domain that published the article.
authors	Array of strings	['J.D.Coldman', 'L.M.Maloney']	A list with the authors involved in the article.
dateDownload	date	2018-05-01	The date when the crawler collected the article.
datePublish	date	2016-06-12	The published date of the article.
images	Array of strings	['http://s2.reuter/100.jpg', 'http://warfar.com/123.jpg']	A list of URL's belonging to the images of the article.
videos	Array of strings	['http://s2.reuter/100.mp4', 'http://warfar.com/123.jmp4']	A list of URL's belonging to the videos contained in the article.
keywords	Array of strings	['dugin', 'culture']	A list of keywords associated to the article.
language	String	'en'	A string code of the language of the article.

ENTITY: ORGANIZATION			
FIELD	TYPE	EXAMPLE	DESCRIPTION
orgIdentifier	Integer	202	Numeric identifier of an Organization.
identifier	string	'j1snumMBhU3h5oXprdJl'	String identifier of the raw article.
Name	string	'The Times'	String representing the name of the Organization

ENTITY: AUTHOR			
FIELD	TYPE	EXAMPLE	DESCRIPTION
authIdentifier	Integer	202	Numeric identifier of an author.

identifier	string	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
name	string	‘J.L. Coldman’	String representing the name of an author associated to the article.
email	string	‘jlcold@gmail.com’	String associated to the email of the author whether it is available.
twitterAccount	string	‘@jlcoldm’	String associated to the twitter account of the author whether is available.

ENTITY: ARTICLE			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	string	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Title	Text	‘Brussels battens down hatches for no deal.’	The title of the article.
Body	Text	‘Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et’	The body of the article.
Summary	Text	‘Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et’	The Summary of the article.
Published Date	date	2018-06-05	The published date of the article.

ENTITY: MEDIA-CONTENT			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	String	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
type	String	‘video’	String type of the media-content including image or video.

URL	String	‘http://s2.reuter/100.jpg’	A URL associated to an image or video contained in the article. Each image or video will be a Media-Content object.

ENTITY: ENTITY			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	String	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Type	String	‘Organization’	A string containing the name of a mentioned entity in the article. It includes Organizations, Names, Places et.
Value	String	‘ONU’	A String value of the mentioned entity.

ENTITY: TOPIC			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	String	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Value	String	‘Cold War	A String value of a topic mentioned in the article.

4.4. NEO4J

Neo4j is an open source graph database software developed entirely in Java. It is a totally transactional database, developed by Neo Technology, a startup of Malmö, Sweden and the San Francisco Bay Area.

The graph structure of Neo4j is extremely convenient and efficient in dealing with structures such as trees extracted for example from XML files, filesystems and networks, which are obviously represented by a graph. Exploring these structures is usually faster than a table database because the search for nodes in relation to a certain node is a primitive operation and does not require multiple steps, usually three implicit in a SQL join, on different tables. Each node contains the index of incoming and outgoing relationships from it.

In FANDANGO’s architecture, NEO4J plays a central role as it will be the basis of the graph analytics procedure which will provide end-users with relevant information regarding the credibility of authors, organizations and other involved entities in the fake-news cycle.

As it was described above, NEO4J will be the core in the graph analytics module, where two principal operations will be performed including create and update the Graph-knowledge Database.

Moreover, NEO4J will access to the data collected and preprocessed at FANDANGO's platform and will build the knowledge graph. Subsequently, a set of graph algorithms will be performed to assess the source credibility task leaded by UPM.

DATA MODEL OVERVIEW

Since NEOJ is the main graph database framework, its data model represents the semantic relationships among the different entities and components based on the Schema ontology presented in previous sections. Therefore, a considerable similarity between this data model and the one presented in Elasticsearch will be observed. Finally, Figure X represents the logical data model that will be implemented in Neo4j.

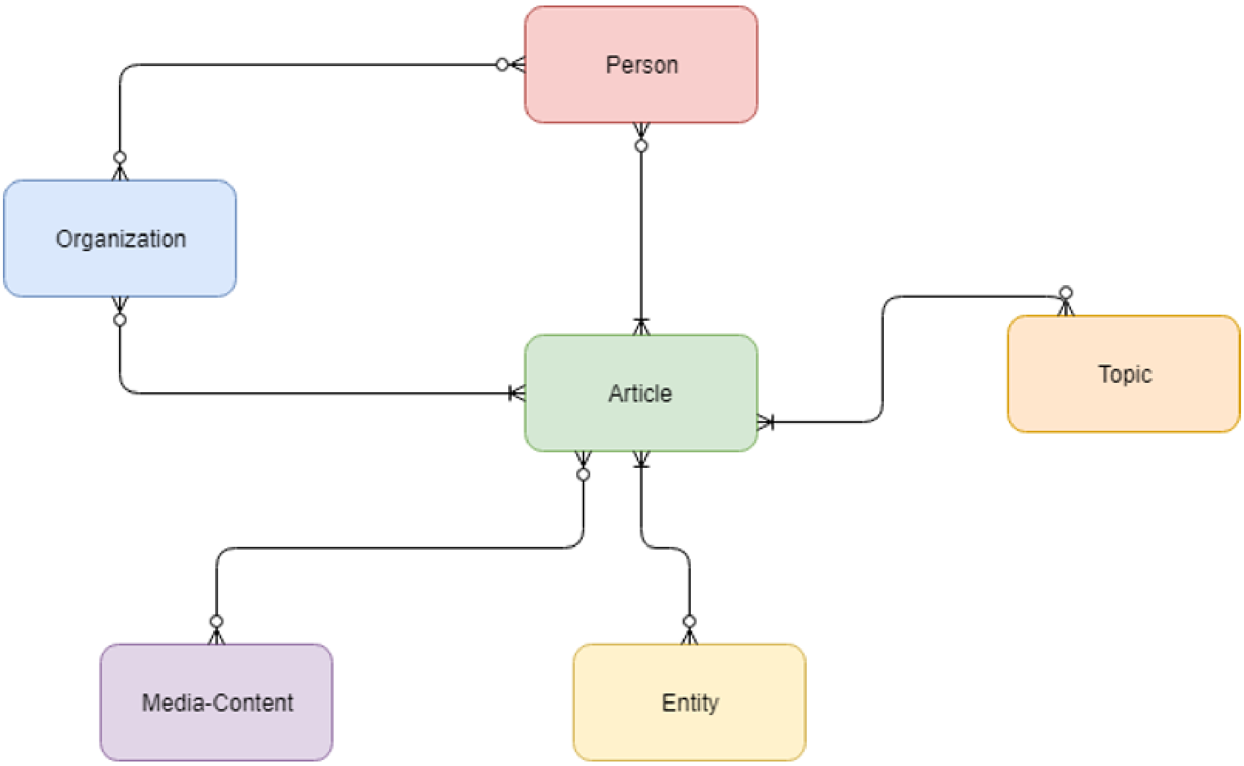

Figure 6 - Neo4J UPM Logical Data Model

ENTITY	OWNER	DESCRIPTION
Organization	UPM	An organization obtained after the pre-processing such as a publisher, newspaper.

Person	UPM	A person, such as a real individual or a fictional persona, like a pseudonym.
Article	UPM	An article, such as a news article or piece of investigative report. Newspapers and magazines have articles of many different types and this is intended to cover them all.
Entities	UPM	An entity mentioned in the article including places, names, organizations etc.
Topics	UPM	A keyword associated to the article.
Media-content	UPM	Images and videos contained in the requested article.

ENTITY SPECIFICATION

In this section, a detailed description of the specifications that each entity will have in NEO4J is presented. As it is observed, the data model is very similar to the one described in the Spark section due to the pre-processing stage that is applied in Spark, will generate a proper format for the data in order to feed afterwards the graph knowledge that will be built in NEO4J. Therefore, the structure of this data model is essentially the same as the one presented in Elasticsearch and Spark with just minor differences regarding the claims that will not be analysed in the graph analysis performance to obtain the desired credibility score.

ENTITY: ORGANIZATION			
FIELD	TYPE	EXAMPLE	DESCRIPTION
orgIdentifier	Integer	202	Numeric identifier of an Organization.
identifier	string	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Name	string	‘The Times’	String representing the name of the Organization
score	Float	0.90	Float value indicating the credibility score of the Organization based on the graph analysis.

ENTITY: PERSON			
FIELD	TYPE	EXAMPLE	DESCRIPTION
authIdentifier	Integer	202	Numeric identifier of an author.
identifier	string	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Name	string	‘J.L. Coldman’	String representing the name of an author associated to the article.
score	Float	0.85	Float value indicating the credibility score of the Person responsible for the article based on the graph analysis.

ENTITY: ARTICLE			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	string	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Title	Text	‘Brussels battens down hatches for no deal.’	The title of the article.
Body	Text	‘Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et’	The body of the article.
Summary	Text	‘Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et’	The Summary of the article.

Published Date	date	2018-06-05	The published date of the article
score	Float	0.75	Float value indicating the credibility score for an article based on the NLP analysis.

Entity: Media-content			
Field	**Type**	**Example**	**Description**
identifier	String	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
type	String	‘video’	String type of the media-content including image or video.
URL	String	‘http://s2.reuter/100.jpg’	A URL associated to an image or video contained in the article. Each image or video will be a Media-Content object.
score	Float	0.90	Float value indicating the credibility score of the media based on the image processing analysis.

Entity: Entity			
Field	**Type**	**Example**	**Description**
identifier	String	‘j1snumMBhU3h5oXprdJl’	String identifier of the raw article.
Type	String	‘Organization’	A string containing the name of a mentioned entity in the article. It includes

			Organizations, Names, Places et.
Value	String	'ONU'	A String value of the mentioned entity.

ENTITY: TOPIC			
FIELD	TYPE	EXAMPLE	DESCRIPTION
identifier	String	'j1snumMBhU3h5oXprdJl'	String identifier of the raw article.
Value	String	'Cold War	A String value of a topic mentioned in the article.

5. CONCLUSION

In this deliverable we present the first iteration of the full implementation of FANDANGO's data model, combining the conceptual vision with the logical implementations. This will pave the way for the technical of the multiple systems required to enhance the data and provide high quality interactions to the users.

A data model allows data to be adequately normalized and defined in terms of what it contains and respective attributes. Without this structure, information systems with large volumes of data will often find challenges in handling data efficiently or delivering meaningful information.

Another important aspect of having this structured data model is integration between the multiple systems, as by modelling the data in each of these systems, you can see relationships and redundancies, resolve discrepancies, and integrate disparate systems so they can work together. As discussed in the deliverables D2.1 and D5.1, and briefly introduced in this document, this integration aspect is of crucial importance in FANDANGO's platform, since there is a vast ecosystem of solutions working together to achieve the expected results.

Moreover, the data model construction also represents an important step of collaboration between end-users and technical partners to make sure that business needs are properly understood and covered by the implementation tasks.

Lastly, the continuity of the project is assured by a well-defined and clearly documented data model, that will serve as basis to other steps of the project, like the deliverable D3.1.