XLike

Deliverable D1.3.2

Final prototype of data infrastructure

Editor:	Esteban García-Cuesta, iSOCO
Author(s):	Blaz Fortuna, JSI; Mitja Trampus, JSI; Blaz Novak, JSI; Esteban García-Cuesta, iSOCO;
Deliverable Nature:	Prototype (P)
Dissemination Level: (Confidentiality)	Public (PU)
Contractual Delivery Date:	M15
Actual Delivery Date:	3.4.2013
Suggested Readers:	Developers creating software components
Version:	1.0
Keywords:	datasets; linguistic corpora; news stream; news indexing

Disclaimer

Full Project Title:	Cross-lingual Knowledge Extraction
Short Project Title:	XLike
Number and Title of Work package:	WP1 – Definition and Data Provision
Document Title:	D1.3.2 – Final prototype of data infrastructure
Editor (Name, Affiliation)	Esteban García-Cuesta, iSOCO
Work package Leader (Name, affiliation)	Blaz Fortuna, JSI
Estimation of PM spent on the deliverable:	12 PM

Executive Summary

This document provides the final description of the XLike data infrastructure which has been and will be the main source of data to be used by the different components of the project and towards accomplishing with the defined use cases.

The document is mostly based on the implemented work from month three until now, and it also includes the updated parts of the deliverable D1.3.1 “Early prototype of data infrastructure”. Since month three some enhancements and updates have been done including the addition of new data sources and some improvements on the infrastructure performance and accessibility.

This deliverable describes this data infrastructure covering the needs of STA and Bloomberg as end-users which are represented through the case-studies in D1.2.1 “Requirements for early prototype” for year one, and D1.2.2 “Requirements for demonstrator” for year two. Regarding the main outcome of the work related to this deliverable it is the newsfeed prototype which is available at a public URL[1] and provides the data infrastructure for the project. This data infrastructure has been already used for the development of the multilingual language processing prototype and the year one early project prototype.

This document does not duplicate or include any information which was already in the deliverable D1.3.1 “Early prototype of data infrastructure” and therefore it may be needed to consult it for a better understanding.

[1] http://newsfeed.ijs.si/

Table of Contents

Executive Summary 3
Table of Contents 4
List of Figures 5
List of Tables 6
Abbreviations 7
1 Introduction 8
2 System performance 9
3 News Stream Additional Sources 11
3.1 Bloomberg 11
3.2 STA (Slovenian News Agency) 11
3.3 Twitter 13
4 Multilingual linguistic analysis and annotation 15
5 Conclusions 16
References 17

List of Figures

Figure 1 Average time latency decreases due to infrastructure improvements. 10
Figure 2 Average relative to the desired target time latency decreases due to infrastructure improvements (expressed in percentage) 10
Figure 3 Median and average time latency (seconds) after applying the improvements. 10
Figure 4 Median and average time latency (seconds) relative to desired target (expressed in percentage). 10
Figure 5 Number of crawl and processed news articles by the JSI Newsfeed system from February 2012 to March 2013. 10
Figure 6 Number of obtained articles from Bloomberg's site per hour. 11
Figure 7 Rate of domestic STA news articles 13
Figure 8 Rate of domestic STA articles since November. 13
Figure 9 Rate of aggregated foreign news articles received through STA. 13
Figure 10 Rate of aggregated foreign news articles received through STA, since November. 13

List of Tables

Table 1 Number of articles retrieved from STA domestic news stream, since August 26th of 2012..... 12
Table 2 Number of articles retrieved from STA aggregation feed, since August 26th of 2012..... 12

Abbreviations

NLP	Natural Language Processing
API	Application Programming Interface
POS	Part of Speech
I/O	Input/Output
SSD	Solid State Drive
RSS	Really Simple Syndication
XML	eXtended Markup Language

1 Introduction

This deliverable provides an updated list of the available data sources within the project and the enhancements done in the news stream infrastructure towards providing a robust and a real time newsfeed as a common data source for the project.

The system performance is measured by the number of articles that is able to process and by the capability for processing on real time the actual sources of information. Due to initial hardware constraints the performance of the system was not as good as expected and some improvements have been needed. The inclusion of a social media source as Twitter also generates a larger volume of data to be processed comparing it with the volume generated by press agencies or news publishers as STA or Bloomberg.

These two aspects: a) the inclusion of a larger volume of data) and b) the adaptation of the infrastructure to use new hardware/software, are treated in the document.

In the following, the infrastructure's enhancements are explained in section 2. Section 3 provides an update of the sources used towards providing complete news accessibility to the project. Section 4 provides the description of the new data format resulting from the execution of the XLike pipeline which is being stored in the data infrastructure, and finally section 5 shows some conclusions.

2 System performance

This section shows some performance improvements which have been made since month three when it was delivered the D1.3.1 “Early prototype of data infrastructure” [1]. By then, the first version of the Newsfeed service had a median time of about four hours from being published to being discovered by the system and this delay in time has been decreased.

The system has a configuration parameter describing the desired latency for each RSS source feed but it had no effect since the system where the newsfeed was running was too slow to process the feeds at the desired rate, and therefore leading to allocate all the feeds in the ready queue. A side effect of this drawback is that the prioritization capabilities based on latency configuration are lost.

An analysis of the database access patterns has shown that the bottleneck was due to the part of system that matches newly discovered URLs with the ones which are already in the database, including redirect aliases. Revisiting an RSS feed contained, on average, only 4.2% of new URLs whereas the rest of them were either duplicates from previous visits of the feed or were received from other feeds with overlapping coverage areas. For example, this happen whenever a single news site provides multiple feeds posting the same link in more than one, or also sometimes due to news aggregation sites. Also, a significant problem occurred with some news sites that publish multiple years worth of their news articles in a single RSS feed, with some reaching over 50.000 URLs per RSS download.

Instead of increasing the revisit times for most of the feeds, which would improve the situation with regard to the revisit times of high priority feeds, but wouldn't have changed median time discovery process, we extended the database tablespace with a pair of SSD drives and created a new set of hash value based indexes on the URL list, partitioned by published date month value. Then, the first check only tests for matching articles in the last two months, and only if that fails, it checks the entire database. This improvement has decreased the database I/O load to acceptable levels, and decreased the median article discovery time to approximately two hours.

Other bottleneck was in RSS downloading and XML parsing process. The first version of the RSS feed monitoring stage of the pipeline was implemented as a single-thread process using greenlet routine system which provides parallelization and simplifies the database locking and consistency management. It was found out that the interplay between delays caused by CPU-bound parsing of large RSS XMLs and database query submission latencies caused by row locking turned out into large pipeline stalls.

To overcome this problem the database was refactored so that all of the shared data structures could be used by multiple RSS feed monitoring processes at the same time, and split the download and parsing of feeds between multiple processes based on the hash value of the internal ID of the feed. This modification allows the use of database sharding in the future whenever the I/O load exceeds the capabilities of a single database server.

Currently eight processes are being used to parallelize the feed processing, however only about three CPUs worth of processing power are used at any time. After processing the backlog, median latency of feed revisit decreased to within 10% of the set point. Figures 1 and 2 show the obtained results at the time of applying the improvements implemented. Figure 1 shows the decrease of feed revisit latency in seconds and Figure 2 the relative percentage to the desired value at the time of implementing the improvements. Figure 3 and Figure 4 also show the typical values throughout one day at the time of writing, in the same units, for a median latency of 1230 seconds and 114% relative to setpoint -- we expect that we will be needed about 11 minutes to revisit a feed that is set to be monitored every 10 minutes. The blue line shows the median latency, and the green fill shows the average latency, which is much larger due to inclusion of feeds that are only revisited once every day or less.

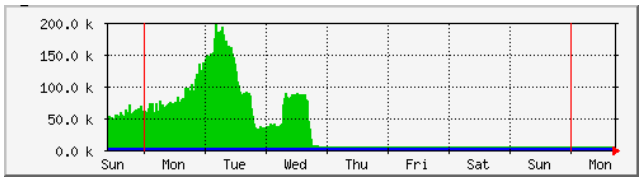

Figure 1 Average time latency decreases due to infrastructure improvements.

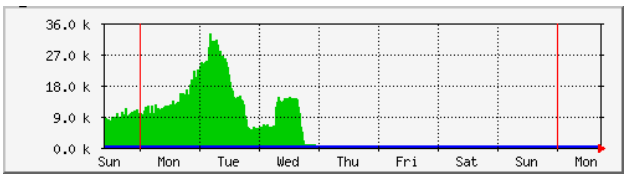

Figure 2 Average relative to the desired target time latency decreases due to infrastructure improvements (expressed in percentage)

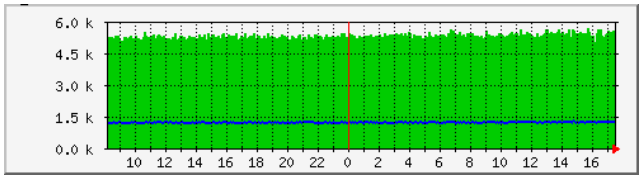

Figure 3 Median and average time latency (seconds) after applying the improvements.

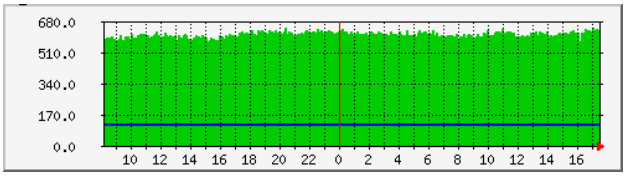

Figure 4 Median and average time latency (seconds) relative to desired target (expressed in percentage).

The rest of the processing pipeline does not have any bottleneck, so the system can easily be scaled to approximately an order of magnitude more of sources by using the same hardware and software. Figure 5 shows the number of articles found and processed during the time span of one year.

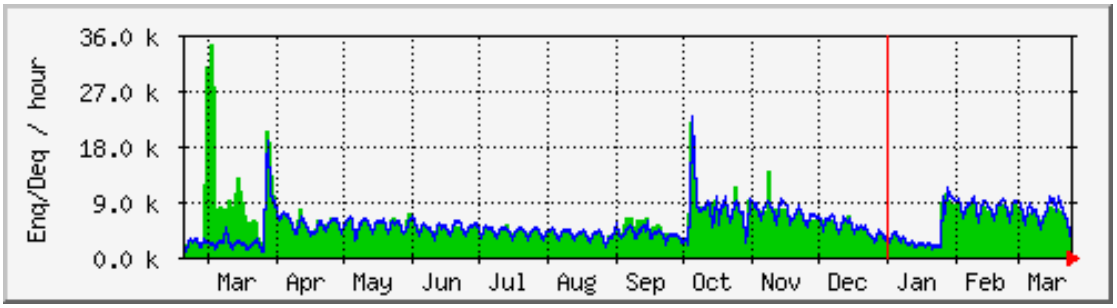

Figure 5 Number of crawl and processed news articles by the JSI Newsfeed system from February 2012 to March 2013.

3 News Stream Additional Sources

This section describes the new added sources to the newsfeed[2] component for the project XLike. Since the initial prototype, the Newsfeed has been extended with the following data sources: two new real-time data streams from Bloomberg and STA which are part of the XLike consortium and prolific creators and syndicators of news content, and a subset of Twitter which is currently the best-known microblogging service.

In the next we describe the characteristics of the mentioned newly incorporated data sources and the mapping between the metadata schemas involved.

3.1 Bloomberg

Bloomberg.com is the world's largest business and financial news site. It provides approximately 20 articles per hour on average, with most of the content published during US business hours at peak rates of over 50 per hour.

Regarding the accessibility whereas the news articles are freely available on the web, Bloomberg's website does not provide a public RSS feed with links directly to the content. For the purposes of the XLike project, Bloomberg has provided a custom built feed which periodically (every 10 minutes) is checked by the newsfeed component for searching new links.

Due to the required use of HTTP authentication, we also use a separate process to monitor this feed. This also allows us to guarantee a fixed low latency for this source. Discovered URLs are queued for download by the existing infrastructure, and marked with access control tags 'xlike' and 'bloomberg', to prevent them from being included in the public output data stream. Due to the existing pipeline is used for every needed step during the data gathering process with the exception of the link discovery the content obtained from Bloomberg is processed exactly in the same way as any other general news obtained by any other source but it is stored in a separate set of output XML files.

In the Figure 6 we show the number of incoming articles from Bloomberg feed since its introduction in the Newsfeed system by half November 2012.

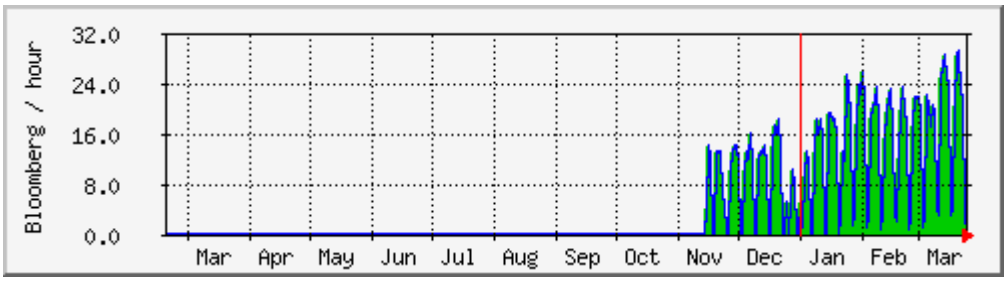

Figure 6 Number of obtained articles from Bloomberg's site per hour.

3.2 STA (Slovenian News Agency)

STA[3] provides to XLike project with two independent news article streams. The first covers articles authored by the agency itself and are composed mostly of content in Slovenian language, whereas the other stream aggregates news articles authored by foreign news agencies.

[2] http://newsfeed.ijs.si
[3] http://www.sta.si/en/

Table 1 shows the number of STA articles received by the first stream since August 26th of 2012 ordered by language, and the Table 2 shows the number of articles received from STA foreign aggregation feed ordered by news agency during the same timeframe.

Table 1 Number of articles retrieved from STA domestic news stream, since August 26th of 2012.

Language	Number of articles
Slovenian	66686
English	6761

Table 2 Number of articles retrieved from STA aggregation feed, since August 26th of 2012.

Agency	Number of articles
DPA	134036
ANSA	117391
AFP	86623
AFP (English version)	77595
AP	54038
APA	44279
TANJUG	41934
HINA	32136
DPA (English version)	22802
ITAR-TASS	17851
MTI	3532
TASR	2920

Regarding the accessibility, the STA articles are not provided in HTML format from the public web site but are received through a dedicated web page which generates a structured XML representation for every news item. Both streams are inserted directly into the database bypassing the discovery, download and HTML cleaning phases, and going directly to the semantic processing step. Every article is also tagged with 'sta' and 'xlike' access control tags in order to prevent them from appearing in the public dataset.

The mapping schema of the articles stored in the newsfeed database follows the description provided at [4] in the section “Stream Contents and Format” which also can be consulted in D6.1.2 “Early Prototype: Annex C” [2]. All the articles retrieved from STA share the same database as well as all other news articles and are compliant with the defined schema. Also, both of the two introduced feeds have a feed description similar to the other feeds including an ID from the same shared namespace. Furthermore, in order to distinguish

[4] http://newsfeed.ijs.si/

the provider of an article we annotated the article with a provider tag which is accessible in the <tags> element of the final XML.

Providers can also be distinguished using the source site ID which is set to the origin agency URL. Any other tags received in the source feed are copied directly to the <tags> set in the final XML structure.

The Figures 7, 8, 9, and 10 shows the daily and yearly rate of articles received from both of the STA feeds.

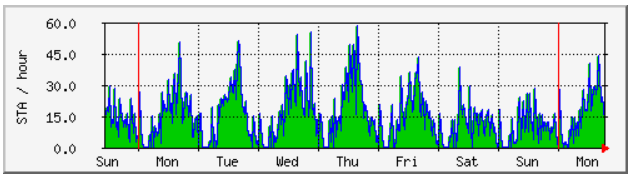

Figure 7 Rate of domestic STA news articles .

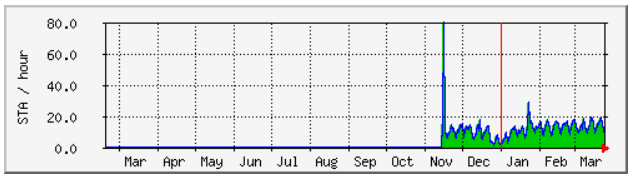

Figure 8 Rate of domestic STA articles since November.

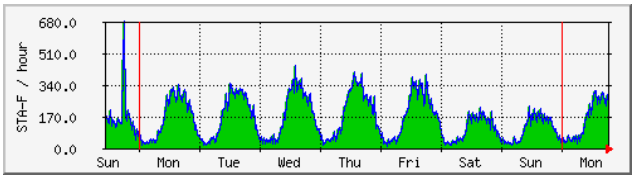

Figure 9 Rate of aggregated foreign news articles received through STA.

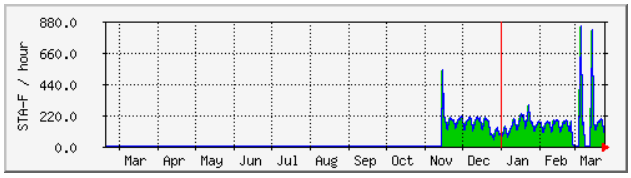

Figure 10 Rate of aggregated foreign news articles received through STA, since November.

3.3 Twitter

Twitter[5] is a microblogging service and its users are from most of the countries of the world publishing content in all major languages. Twitter offers a free uniform 1% sample of the posted messages as a real-time stream amounting roughly to four million messages per day. We have incorporated that stream into our XLike pipeline and present it to the end users together with the other data sources following a unified data schema.

Regarding the accessibility we have used Twitter's REST API (version 1.1)[6] based on OAuth authentication for accessing to the twitter stream.

Twitter's schema consists of a number of different types of events. As most of them relate to data about the evolution of the social network rather than to content itself, we have ignored many of them as: 'friends', 'event', 'for_user', 'control' and 'limit', therefore this type of data does not go into the processing stage. On the other hand, for events of type 'text', e.g. the ones which actually contain new tweets we have performed the following schema mapping:

- ID: we form our internal ID simply by adding "tw-" at the beginning of the Twitter's tweet ID. This will ensure the traceability of data.
- URL: we reconstruct the HTTP URL of the tweet from its metadata. This metadata encodes, in an interpretable way, both the author's username and the tweet ID. This allows us to keep track of the user which creates that tweet without changing the schema towards Twitter specifics.

[5] https://twitter.com/
[6] https://dev.twitter.com/docs/api

- Source coordinates: The API object contains the coordinates (longitude and latitude) of the place where the tweet was published in the property *tweet.coordinates.coordinates* (this is not always known).

- Story coordinates: The API object may also contain annotations denoting places that are being talked about in the tweet. These annotations are stored in the API object's *tweet.place* and each element representing a place is defined by the *tweet.place.bounding_box.coordinates* property. From a practical point of view this is a list of latitude-longitude pairs that delimit the area being discussed in the tweet. As our current schema only models points we converted the bounding box by averaging out all of its border's coordinates.

The full schema for describing a tweet within twitter API is described and can be consulted on [7].

All Twitter content is already included in the final aggregated stream of web content. It is distinguishable, among other above mentioned ways, by its hostname tag ("twitter.com") and is treated differently due to its relatively high data volume. At now, it currently bypasses the deep linguistic analysis and it is plugged directly into the content caching and distribution service (see Figure 1 in D1.3.1 [1]).

[7] https://dev.twitter.com/docs/platform-objects/tweets

4 Multilingual linguistic analysis and annotation

In this section we provided an overview of the additional data generated by the XLike technological functionalities and how have been incorporated to the data infrastructure through the enrichment of the original data.

At month three we reported the use of the English articles enriching them with results of XLike multi-linguistic analysis by using Enrycher[8]. Since then this process of enrichment has been extended to several other XLike languages (es, en, sl, zh, ca, de). We currently perform the following analyses:

- Multi-lingual tokenization: this process performs the tokenization of the text for each language.
- Multi-lingual lemmatization: this process finds the lemma for each word of the text analyzed.
- Multi-lingual POS analysis: this process obtains the part of speech roles of each word of the text under analysis.
- Multi-lingual entity extraction and disambiguation: this process obtains the entities disambiguated from the text.

Individual project partners contribute different parts to making the multilingual analyses work:

- JSI's Enrycher provides shallow and deep analysis for Slovene and English articles.
- iSOCO provide shallow analysis for English, Spanish and Catalan articles.
- THU provides shallow analysis for Chinese articles.
- FF Zagreb provides shallow analysis for German articles.

KIT provides semantic analysis (named entity disambiguation) for all non-minority languages (English, German, Spanish), building on top of the output provided by the other partners. All services (see D6.1.2 for descriptions of the services [3]) use a common API and also share the format for encoding the results of analyses. This ensures interoperability and enables, for example, KIT's engine to build on top of outputs for all the majority languages.

The services are accessed in a non-blocking, multithreaded fashion taking into account the fact that some XLike functionalities (e.g. deep linguistic processing) are complex tasks and often are difficult to perform in the short time frame provided by the rate of incoming messages (<100ms). So far, we currently maintain a maximum of 10 messages simultaneously for avoiding overwhelming the service providers.

This multilingual property of the project, similarly to some others described in this deliverable, significantly increases the geographical spread of this multilingual pipeline and the number of machines on which it depends for its fully functioning. For gaining some control over this drawback, all the processing stages are being monitored and the generated graphs are aggregated at a central server which also provides visualizations of the metrics it collects.

All this generated new data (jointly with the gathered source information) is accessible via streaming at http://newsfeed.ijs.si/stream/[9]. The URL accepts an optional ?after=TIMESTAMP parameter, where TIMESTAMP takes the ISO format yyyy-mm-ddThh:mm:ssZ (Z denotes GMT timezone). The server returns the oldest gzip created later than TIMESTAMP. HTTP headers (Content-disposition: attachment; filename="...") contains the new gzip's filename which can be used to generate the next query, and so on. If the after parameter is too recent (no newer gzips available), HTTP 404 is returned. If no after is provided, the oldest available gzip is then returned.

[8] http://enrycher.ijs.si/

[9] Right now it only can be used internally due to copyright issues.

5 Conclusions

This deliverable describes the final data infrastructure updates which will be the main source of data to be used in the XLike project. This deliverable jointly with the previous version [1] forms the complete documentation regarding the used data sources, how they have been integrated, and what type of accessibility is provided to retrieve the stored and structured data.

For this purpose a description of the new sources of information is presented in this document including the Bloomberg and STA feeds as part of the consortium resources. We have preserved their privacy due to property rights but all the gathered data is used for the purposes of the use cases of the project. Also, the inclusion of a social media source has been added by using the public Twitter API which allows the access to the one percent of the published tweets.

The inclusion of this new data sources and the need of accomplishing with the year one early prototype as part of the use cases accomplishment has led to some improvements on the previous JSI newsfeed which have been explained showing the associated improvements in performance.

At this stage we are using all the explained data sources for the purposes of the project and to accomplish with the defined use cases, and there is not any new source to be added in the near future (year two of the project) although they will be added as needed. We are also studying the use o Apache Cassandra[10] for storing and accessing the data due to the real time constraints that have appeared in the project and the good capabilities that this tool has regarding scalability and accessibility. Although it is still a preliminary work it may be used as part of the infrastructure during this second year of the project.

[10] http://cassandra.apache.org/

References

[1] XLike deliverable D1.3.1 “Early prototype of data infrastructure”.

[2] XLike deliverable D6.2.1 “Early prototype”.

[3] XLike deliverable D6.1.2 “Final Toolkit Architecture Specification”

