

On-demand Text Analytics and Metadata Management with S4

Marin Dimitrov¹, Alex Simov¹ and Yavor Petkov¹

¹*Ontotext AD, 47A Tsarigradsko Shose blvd., Sofia, Bulgaria
{marin.dimitrov, alex.simov, yavor.petkov}@ontotext.com*

Keywords: text analytics, linked data, knowledge graphs, semantic web, software-as-a-service, database-as-a-service

Abstract: Semantic technologies provide a new, promising approach for smart data management and analytics. At the same time, the adoption of an emerging technology is usually limited by factors such as its perceived complexity, cost and performance. Startups and mid-size businesses often have very limited resources to evaluate and prototype with emerging technologies, even if their potential for more efficient data management and analytics is significant. The Self-Service Semantic Suite (S4) provides an integrated platform for cloud-based text analytics and Linked Data management as-a-service, so that companies in the early stage of evaluating and adopting semantic technologies can easily access a full suite of semantic data management and text analytics capabilities for smart data analytics in various domains.

1 INTRODUCTION

Some of the typical challenges related to efficient data analytics within enterprises include: valuable information locked in unstructured data sources and available only through inaccurate keyword-based search; a large number of heterogeneous data sources and data silos leading to data quality and reuse problems; slow and rigid data integration processes hindering the access to all the relevant and up-to-date information required for analytics.

Semantic technologies provide a novel approach for data integration, discovery and analytics with significant advantages for a variety of analytical use cases. Among the main advantages of semantic technologies are: the flexible, graph-based RDF data model (W3C, 2014b) which facilitates agile schema-less data integration of heterogeneous data sources; the ability to interlink entities of interest found in text – people, organisations, locations, events and topics – into knowledge graphs; the ability to use formal ontologies as a common metadata layer on top of different data sources and silos; a very expressive RDF query language (SPARQL); the ability to infer implicit facts from data, based on formal reasoning rules; the ability to map and interlink between different schemata and data sources and perform semantic search based on meaning, rather than keywords or schemata; Linked Open Data (Heath, 2011) as a novel data publishing

and interlinking paradigm, that can facilitate access to vast amounts of open data on the web.

A number of large companies in various industry sectors – media and publishing, healthcare and life sciences, oil & gas, cultural heritage and digital libraries, retail and finance – have recently adopted semantic technologies in order to achieve higher agility and efficiency when dealing with the modern data analytics challenges. At the same time the wider adoption of semantic technolgis is currently limited by factors such as the perceived complexity and cost for deploying semantic data management solutions. Startups and mid-size businesses often have limited resources for evaluating and prototyping with novel data management approaches, while large enterprises often have complex and inefficient procurement processes which hamper the speed of technology adoption and innovation.

2 THE SELF-SERVICE SEMANTIC SUITE

The Self-Service Semantic Suite¹ (S4) aims at reducing the cost and complexity of semantic technology adoption by providing an integrated platform for cloud-based text analytics and Linked Data management as-a-service. With S4 the companies in the early stages of evaluating and adopting semantic technologies have the ability to easily and quickly apply a full suite of semantic data management and text analytics capabilities for solutions in various domains, without the need for complex planning, budgeting, provisioning and operations.

The main use cases for text analytics and Linked Data management as-a-service with S4 fall into three broad categories:

- *Reducing time-to-market* – companies who are still experimenting with semantic technologies (“technology enthusiasts” and “visionaries”, based on their openness towards emerging technology innovations (Moore, 2014)), need capabilities for semantic data management and text analytics that are available from the “get-go” and do not require complex on-boarding, integration and customization, so that the organisations can deliver new products and prototypes at a rapid rate.
- *Reducing risk* – companies who fall in the group of “pragmatists” when it comes to technology innovation adoption, benefit from a low-cost and low-risk option for experimenting and adopting semantic technologies, without the need to commit to license purchases and hardware provisioning, or deal with inefficient internal procurement processes. By using a platform for semantic data management and text analytics as-a-service, such companies can thoroughly evaluate semantic technologies maturity,

reliability, performance and ROI potential before committing to it.

- *Optimising costs* – even the companies who have already successfully evaluated the potential ROI of semantic technologies and are committed to their long term adoption can often achieve cost reductions by switching to a cloud deployment with pay-per-use cost model.

S4 provides a self-service and on-demand set of components for semantic data management and text analytics, covering key aspects of the data management lifecycle:

- On-demand and reliable access to central *Linked Open Datasets* such as DBpedia, Freebase, GeoNames, MusicBrainz and WordNet
- A self-managed and a fully-managed scalable *RDF database as-a-service* in the Cloud, for private RDF knowledge graphs
- Various *text analytics services* for news, biomedical documents and social media, which extract valuable insight from unstructured content

2.1 Linked Data

Linked Data provides a novel data publishing and interchange paradigm that facilitates publishing, interlinking and reuse of large amounts of data within the organisation, or between the organisations within a supply chain, based on four simple principles (Heath, 2011):

- Use URIs as names for things
- Use HTTP URIs, so that people can look up those names
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
- Include links to other URIs, so that they can discover more things.

The amount of openly available Linked Data has been growing at a rapid rate in the recent years

¹ <http://s4.ontotext.com/>

(Schmachtenberg, 2014), though various performance and availability problems associated with many public LOD endpoints (Buil-Aranda, 2013) still hamper the wider LOD adoption.

To alleviate the various performance and availability problems associated with open Linked Data, S4 provides a reliable and metered access to key datasets from the LOD cloud via the FactForge² large-scale semantic data warehouse (Damova, 2012). More than 5 billion LOD triples, describing 500 million entities, are available to S4 developers via integrated and aligned datasets such as DBpedia, Freebase, GeoNames, and MusicBrainz as well as ontologies and vocabularies like Dublin Core, SKOS and PROTON.

2.2 RDF Databases

RDF databases represent a special class of graph databases where data is modelled based on the semantics of the RDF (W3C, 2014b), and OWL (W3C, 2012) formal model specifications, and data is queried via SPARQL (W3C, 2013). The main advantages of using RDF databases for data management include: the schema agility, ease of integration of heterogeneous data sources, expressive query language, and strong compliance to standards, improved data portability and tool interoperability, as well as ability to infer new, implicit facts from the data.

S4 provides an RDF database-as-a-service capability based on one of the leading enterprise RDF databases: GraphDB (Bishop, 2011). The cloud database infrastructure of S4 is available in two flavours: a *self-managed* cloud database where the user is in full control of operational aspects – such as availability, performance tuning, backups and restores – and a *fully managed* cloud database where the S4 platform takes care of all aspects related to database administration, provisioning and operations.

The self-managed database in the cloud provides an on-demand and private database server (single tenant model) suitable for organizations that need only the occasional, yet high-performance and reliable access to private RDF datasets, in cases where an on-premise software and hardware deployment would not be cost optimal.

The fully managed RDF database in the cloud provides pay-per-use 24/7 access to private RDF databases and SPARQL endpoints within a multi-tenant model. Operational aspects such as security, availability, monitoring and backups are fully handled by S4 on behalf of the users. The security isolation and resource utilisation control of the different database instances hosted within the same virtual machine in the Cloud is achieved by employing a container-based architecture with the Docker technology.

2.3 Text Analytics Services

Extracting value from text is among the main challenges of Big Data analytics at present, with precious value being locked within vast amounts of unstructured data which is difficult to analyse. Semantic technologies are a good fit for dealing with the “variety” aspect of Big Data, and structured, semi-structured and unstructured data sources can be interlinked into semantic (RDF) graphs.

S4 provides various services for real-time text analytics:

- *News Analytics* – the service performs information extraction and entity linking to large open knowledge graphs such as DBpedia, Freebase and GeoNames (Damova, 2012). The text analysis process is a combination of rule-based and machine learning techniques (Georgiev, 2013).
- *News Classifier* – the service performs categorisation of news articles according to the 17 top-level categories of the IPTC Subject Reference System (IPTC, 2003).

² <http://factforge.net/>

- *Biomedical Analytics* – the service can recognize more than 130 biomedical entity types (Georgiev, 2011) and semantically link them to a large-scale biomedical LOD knowledge base (LinkedLifeData³).
- *Twitter Analytics* – the service is based on the TwitIE open source microblog analysis pipeline (Bontcheva, 2013) and it performs named entity recognition of various classes of entities as well as normalisation of most common abbreviations frequently found in tweets.

3 ARCHITECTURE

The architecture of S4 is based on best practices and design patterns for scalable AWS cloud architectures (AWS, 2014).

3.1 Public Cloud Platform

S4 is currently deployed on a public AWS⁴ cloud platform and it utilizes various cloud infrastructure services such as:

- *distributed storage* via Simple Storage Service (S3), Elastic Block Storage (EBS), and DynamoDB
- *elastic computing* via Elastic Compute Cloud, Auto Scaling and Elastic Load Balancer
- *application integration* via the Simple Queue Service (SQS) and Simple Notification Service (SNS)

3.2 S4 Architecture

S4 follows the principles of micro-service architectures and it is comprised of the following main components and layers (Figure 1):

- *Load balancer* – the entry point to all S4 services is the AWS load balancer which redirects incoming requests to one of the available routing nodes.
- *Routing nodes* – these instances host various micro-service frontends to the text analytics nodes, the LOD as well as the database nodes. The job of these nodes is just to perform pre-processing and post-processing (if necessary) and to forward the client request (a document for text processing or a database query/update) to the proper backend node – a text analytics node, a database node, or the LOD server itself. All instances host the same set of stateless front-end services and this layer is automatically scaled up or down (new instances added or removed) based on the current system load and performance. The communication between the routing nodes and the LOD server and database nodes is synchronous, while the communication with the text analytics nodes is asynchronous (via a distributed queue).
- *Text analytics nodes* – these instances are responsible for processing the text documents sent for analysis to S4. They host the different text analytics services for news, biomedical documents and social media. This layer is also automatically scaled up or down based on the current system load and performance.
- *Database nodes* – a database node is a virtual machine that hosts a number of independent GraphDB instances packaged as Docker containers. Each database container stores its data on a dedicated network-attached storage volume (EBS), and EBS volumes are not shared between different database containers for improved performance and isolation. New database nodes are dynamically added by the *Coordinator* node when there are no free database container slots left on the current database nodes. If a database node has free container slots then it periodically contacts the coordinator for the IDs of databases which are still waiting to be deployed, so that the database node can fill up its full hosting capacity as soon as possible – by attaching the dedicated EBS volume for the database to one of its available containers.
- *Coordinator* – there is a single coordinator which is responsible for distributing the database initialisation tasks among the active database nodes. The coordinator keeps a “routing table” with information on the

³ <http://linkedlifedata.com/>

⁴ <http://aws.amazon.com/>

databases hosted by each database node. If a database node crashes, then the coordinator will mark its databases as non-operational and will re-distribute them (their IDs) to other database nodes with free database slots, or to the new database node which will be automatically instantiated by the AWS Auto Scaler to replace the crashed node.

- *Linked Data server* – currently the LOD data available through S4 is hosted on the FactForge semantic data warehouse.
- *Integration services* – a distributed queue and a distributed push messaging service are used for the loose coupling and asynchronous communication between the components of the platform. For example, the requests for text processing are first handled by a routing node, which puts a processing request in the distributed queue (SQS) so that one of the available text analytics nodes will pull the request, process it, and send the result back to the routing node. This way the routing and text analytics nodes are not aware of their number and topology and they can be scaled up/down independently. In a similar manner, the distributed push messaging service (SNS) is used for loose coupling between the database nodes, routing nodes and the coordinator. Each database node sends “heartbeats” several times per minute via the notification service, so that routing nodes and the coordinator get a confirmation that the databases hosted by that node are still operational. Each database node also sends periodic updates regarding the databases it is hosting, so that the routing nodes and the coordinator can update their routing tables if necessary.
- *Distributed storage* – AWS Simple Storage Service (S3) is used for transient storage of documents, and for database backups. The data for the database nodes is stored on high-performance network-attached storage volumes (EBS).
- *Metadata store* – a distributed key-value store (DynamoDB) stores simple metadata regarding the databases hosted on the platform (user ID, dedicated EBS volume ID, configuration parameters, text analytics components metadata, user accounts, etc.), as well as the logging data from all platform operations.
- *Management and monitoring services* – various management microservices cover

operational aspects such as logging, reporting, account management, quota management and operations monitoring.

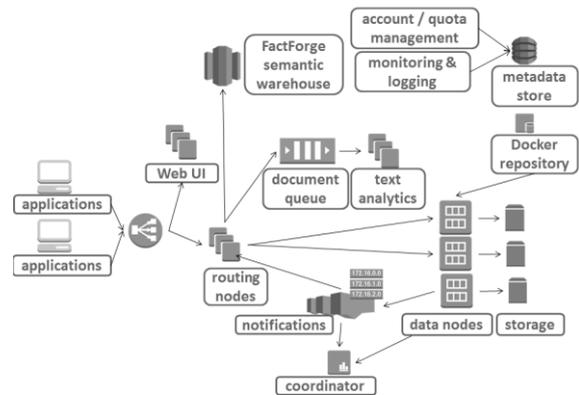


Figure 1 S4 architecture.

3.3 Operations

The behaviour of each of the S4 sub-systems (text analytics, Linked Data server, database as-a-service) is described next.

3.3.1 LOD Access

Since S4 just provides metered access to the FactForge semantic warehouse (Damova, 2012), the job of the routing node that receives a request for the LOD server (a SPARQL query for some LOD dataset) is to just forward the query to FactForge and then return the result back to the client application in a synchronous manner. There is only one instance of the FactForge warehouse.

3.3.2 Text Analytics

When a routing node receives a document to be processed by one of the text analytics services on S4, it packages the document with additional metadata and puts the request in the distributed queue (SQS). One or more text analytics nodes are constantly polling the queue for pending requests for document processing. When a text analytics node retrieves a request from the queue, it will process it and return the result directly to the routing node that originated the request (this information is available in the metadata of the request), so that the routing node can in turn return the result to the client application that sent the document for processing. Neither the routing nodes, nor the text analytics nodes are aware of the exact number of topology of the nodes and new nodes can be added or removed dynamically as needed.

3.3.3 Database As-a-Service

Unlike the text analytics sub-system, the routing nodes need to be aware of the exact topology of the database nodes, since a client request has to be directed to the proper database node hosting the client database at that particular moment. For this reason, the database as-a-service sub-system is more complex and it involves a Coordinator node and a distributed push messaging service for asynchronous communication between the routing nodes, the database nodes and the Coordinator.

Each routing node maintains a routing table so that it knows which database node is currently hosting a particular client database: the routing data for each database includes the IP address (of the database node) and the port (of the Docker container running on the database node) where the database is currently hosted. If there is a change in the database layer topology (e.g. a database node crashes and another one is instantiated to replace it and host its databases) then the routing tables are updated.

When a routing node starts, it immediately subscribes to the distributed notifications service (SNS), so that it can start receiving heartbeats and routing updates from the database nodes. If a client request is forwarded by the load balancer to the new routing node before it has its routing table fully initialised, then the node will just queue the client request locally. After a short period of time the routing node will receive the heartbeat and routing notifications from all database nodes, so that it can start forwarding client requests to the proper database node. After a routing node forwards the client request to the proper database node, it waits for the database response and then forwards the response back to the originating client application (the communication between the client application the routing node and the database node is synchronous).

When the Coordinator starts, it first reads the metadata about all databases on the platform from the metadata store, and then subscribes to the notifications service (SNS) in order to receive heartbeats and routing updates from the database nodes. If after short period of time there is still a database (listed in the metadata store) that no database node is currently hosting, the coordinator assumes that this database is down and will send its ID to the next database node which contacts the coordinator requesting new databases for initialisation.

When a database node starts, it initialises its database containers from the local Docker repository

and immediately contacts the coordinator to request a list of databases (IDs) which to host on its local containers. When the coordinator provides the information, the database node just attaches the dedicated network-attached storage (EBS) volume for the database to itself and performs the final OS level configuration so that the database container can be fully initialised. At this point, a Docker container with a running GraphDB instance and an attached data volume is fully operational on the database node. The next step for the database mode is to subscribe to the notifications service and start sending regular heartbeats and routing updates to the routing nodes and the coordinator. At this point the database node is ready to serve client requests forwarded to its active databases by the routing nodes.

3.4 Dealing with Failure

3.4.1 LOD Access

The FactForge semantic warehouse is single-point-of-failure component related to LOD access, since due to its large scale and hardware requirements it is not cost efficient at present to provide multiple replicas of the warehouse for improved availability. Nonetheless, the warehouse availability and performance is constantly being monitored, and FactForge is listed as the LOD endpoint with the highest availability and reliability in a recent analysis by (Buil-Aranda, 2013).

3.4.2 Text Analytics

In the case of a routing node failure the load balancer will automatically stop redirecting requests to the problematic node and the Auto Scaler will instantiate a new replacement node. Only the currently open connections from client applications to the problematic routing node will be terminated abnormally, while the rest of the system will be fully operational. If a text analytics node fails while processing documents, then after a short period of time the documents that it was processing will become visible as messages in the distributed message queue again, so that a different healthy backend node can pull them for processing (messages are deleted from the queue only upon returning the result to the client application, and marked as “invisible” while being processed by some text analytics node).

3.4.3 Database As-a-Service

In the case of a routing node failure the load balancer will start redirecting requests to other (healthy) routing nodes. Meanwhile the AWS Auto Scaler will instantiate a new routing node to replace the failed one. The new node follows the initialisation steps described in the previous section and it will be soon fully operational. Database nodes and the Coordinator are not affected by a routing node failure.

If the Coordinator crashes, the Auto Scaler will automatically instantiate a replacement coordinator node. The new coordinator will follow the standard initialisation steps, build its routing table and start distributing non-operational database IDs when requested by the database nodes with free containers.

If a database node crashes it will stop sending heartbeats to the notification service and the routing nodes will be aware that the databases hosted on that database node are not operational, so that they will start queueing the client requests for these databases locally. The Coordinator will mark the databases hosted by the failed data node as non-operational and will be ready to send their IDs to the next database node which requests pending databases for initialisation. Meanwhile, the Auto Scaler will detect the node failure and instantiate a replacement node, which will follow the initialisation steps from the previous section: first request pending databases from the coordinator, then start sending heartbeats and routing updates to the routing nodes and the coordinator. Soon the routing nodes will be aware of the new location of the failed databases and will start forwarding the client requests that were being queued.

A combination of nodes may crash at any time (including *all* of the nodes on the platform) but the recovery will always follow the steps above, with the following dependencies:

- The coordinator does not depend on any active routing / database nodes to be operational.
- Database nodes depend on the coordinator to be operational, so that they can receive database initialisation tasks for their hosted containers.
- Routing nodes depend on the database nodes to be operational, so that they can initialise their routing tables and forward client requests to the proper database

3.5 Scalability

Routing nodes are dynamically added based on the current system load. When a new routing node is

added, the load balancer will automatically start redirecting some of the incoming requests to it.

Text analytics nodes are also dynamically added when the number of documents waiting for processing in the queue exceeds a pre-defined threshold.

The integration services (distributed message queue and distributed push messaging) are designed by AWS so that they can scale up to thousands of messages processed per second.

The database nodes are currently not replicated, so if a particular database experiences a usage spike and becomes overloaded, its performance will temporarily decrease. At the same time, due to the simple container and network-attached storage based architecture that S4 employs, it is possible to quickly scale up the database container by re-deploying it on a bigger virtual machine with more memory and CPU cores. EBS volume performance can also be increased (at a higher cost).

4 CONCLUSIONS AND FUTURE WORK

This paper presents the Self-Service Semantic Suite (S4), a cloud platform providing on-demand access to key capabilities for semantic data management: access to large open knowledge graphs (Linked Open Data), RDF databases as-a-service, and various text analytics services.

Several existing platforms offer text analytics as-a-service capabilities: *Alchemy*, *Bitext*, *DatumBox*, *MeaningCloud*, *OpenCalais*, *OpenAmplify*, *Saplo*, *Semantria*, etc. *Dydra* provides capabilities for RDF databases as-a-service. Some of the platforms for text analytics as-a-service already cover multiple languages and provide support for sentiment analytics as well. The main differentiation of the S4 platform is that it provides an integrated suite for semantic analytics which allows for content from unstructured data sources to be semantically

enriched and interlinked into an RDF knowledge graph, so that semantic search and discovery can be utilised for data analytics.

S4 has already been deployed in production⁵, but a variety of improvements are planned or already in development:

- Availability of multilingual text analytics services, including services for sentiment analytics;
- Asynchronous, batch processing of large volumes of text, in addition to the current synchronous mode of operation;
- Adoption of JSON-LD (W3C, 2014a) for the text analytics services output;
- Integration of 3rd party tools for visual exploration and navigation of large scale Linked and RDF data;
- Integration of Linked Data Fragment based containers (Verborgh, 2014), as an alternative approach for scalable querying of RDF data.

ACKNOWLEDGEMENTS

Some of the work related to S4 is partially funded by the European Commission under the 7th Framework Programme, project DaPaaS⁶ (No. 610988).

REFERENCES

- Amazon Web Services, 2014. AWS Reference Architectures. Available at <http://aws.amazon.com/architecture>
- Bishop, B., Kiryakov, A., Ognyanoff, D., Peikov, I., Tashev, Z., Velkov, R., 2011. OWLIM: A family of scalable semantic repositories. In *Semantic Web Journal*, vol 2, number 1.
- Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M.A., Maynard, D., Aswani, N., 2013. TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. In *RANLP'2013, International Conference on Recent Advances in Natural Language Processing*.
- Buil-Aranda, C., Hogan, A., Umbrich, J., Vandenbussche, P., 2013. SPARQL Web-Querying Infrastructure: Ready for Action? In *ISWC'2013, 12th International Semantic Web Conference*.
- Damova, M., Simov, K., Tashev, Z., Kiryakov, A., 2012. FactForge: Data Service or Diversity through Inferred Knowledge over LOD. In *AIMSA'2012, 15th International Conference on Artificial Intelligence: Methodology, Systems and Applications*.
- Georgiev, G., Pentchev, K., Avramov, A., Primov, T., Momtchev, V., 2011. Scalable Interlinking of Bio-Medical Entities and Scientific Literature in Linked Life Data. In *CALBC'2011, Workshop on Collaborative Annotation of a Large Biomedical Corpus*.
- Georgiev, G., Popov, B., Osenova, P., Dimitrov, M., 2013. Adaptive Semantic Publishing. In *WaSABi'2013, Workshop on Semantic Web Enterprise Adoption and Best Practices*.
- Heath, T., Bizer, C., 2011. *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool.
- IPTC, 2003. Subject Reference System Guidelines. Available at http://www.iptc.org/std/NewsCodes/0.0/documentation/SRS-doc-Guidelines_3.pdf
- Schmachtenberg, M., Bizer, C., Paulheim, H., 2014. Adoption of Linked Data Best Practices in Different Topical Domains. In *ISWC'2014, 13th International Semantic Web Conference*.
- Moore, G., 2014. *Crossing the Chasm: Marketing and Selling Disruptive Products to Mainstream Customers*, Harper Business, 3rd edition.
- Verborgh, R., Hartig, O., De Meester, B., Haesendonck, G., De Vocht, L., Vander Sande, M., Cyganiak, R., Colpaert, P., Mannens, E., Van de Walle, R., 2014. Querying Datasets on the Web with High Availability. In *ISWC'2014, 13th International Semantic Web Conference*.
- W3C, 2012. OWL 2 Web Ontology Language Document Overview (2nd Edition). W3C Recommendation (December 2012). Querying Datasets on the Web with High Availability. In *ISWC'2014, 13th International Semantic Web Conference*.
- W3C, 2013. SPARQL 1.1 Overview. W3C Recommendation (March 2013).
- W3C, 2014a. JSON-LD 1.0 – a JSON-based Serialisation for Linked Data. W3C Recommendation (January 2014).
- W3C, 2014b. RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation (February 2014).

⁵ <http://s4.ontotext.com/>

⁶ <http://project.dapaas.eu/>