# Curating Semantic Linked Open Datasets for Software Engineering

Kavi Mahesh, Aparna Nagarajan,
Apoorva Rao Balevalachilu, and Karthik Prasad

Centre for Knowledge Analytics and Ontological Engineering - KAnOE,
PES Institute of Technology, Bangalore 560085 India
{drkavimahesh,aparna.nagarajan26,
apoorva.rao.b,karthikprasad008}@gmail.com
http://kanoe.org/

**Abstract.** A typical software engineer spends a significant amount of time and effort reading technical manuals to find answers to questions especially those related to features, versions, compatibilities and dependencies of software and hardware components, languages, standards, modules, libraries and products. It is currently not possible to provide a semantic solution to their problem primarily due to the non-availability of comprehensive semantic datasets in the domains of information technology. In this work, we have extracted, integrated and curated a linked open dataset (LOD) called LOaD-IT exclusively on this domain from a variety of sources including other LODs such as Freebase and DBPedia, technical documentation such as JavaDocs and others. Further, we have built a technical helpdesk system using a semantic query engine that derives answers from LOaD-IT. Our system demonstrates how productivity of the software engineer can be improved by eliminating the need to read through lengthy technical manuals. We expect LOaD-IT to become more comprehensive in the future and to find other related practical applications.

**Keywords:** Linked Open Data, software engineering, technical helpdesk, information retrieval, semantic search.

## 1 Introduction

Software developers, test engineers, researchers and students face technical queries in the course of their daily work. The solution to their problems, more often than not, is to go through some technical documentation, search for information, look for a similar question on on-line forums, or to ask somebody who is considered an expert in that domain. Unfortunately, information from these sources is rather unstructured, distributed and difficult to obtain. The process of going through the technical documentation is tedious and time consuming. What the person ideally wants is a quick and precise answer to the question perhaps augmented by a suitable visualization to aid comprehension. The objective of our work is to enhance workplace productivity of information technology workers by reducing the effort and time spent in searching for technical information.

There is currently no readily available LOD that covers the factual knowledge sought by a typical information technology (IT) worker. Thus, our solution begins by semanticizing the relevant documentation by creating a linked open data store, which we call LOaD-IT (available as an RDF dump in N-Quads format at http://datahub.io/dataset/load-it). The potential of this highly specialized linked data is tapped by building Kappa, a technical assistant application prototype (Kappa is hosted at http://kanoe.org/lod/loadit-kappa.html). Kappa takes keyword queries as input and provides specific factual answer(s) instead of large number of HTML pages that may or may not contain the correct and complete answer. It also saves the user from scrolling through long manuals and documents to manually extract the answer. In order to provide results in the form of facts, the application first performs query interpretation as described in [1]. The user of the application need not have any knowledge of the structure of LOaD-IT or knowledge of RDF formats. Simple keywords entered in a search bar suffice. Search results are presented as a list of facts along with hyperlinks to relevant pages. Further, a graphical representation of the relevant RDF sub-graph is shown to help the user visualize the facts. Use of anontology to implement semantic search together with Web search has been demonstrated before (see, e.g., [2]). However, such methods typically require semantic meta-data or annotations to be manually added to existing documents or web pages.

## 2 Creating the Semantic LOD on Information Technology: LOaD-IT

Existing semantic datasets [3] like DBPedia, Freebase and YAGO are quite large and cover multiple domains while focusing heavily on common areas of interest rather than technical domains such as IT. It is unlikely that a retrieval engine on such a general-interest LOD will provide the necessary recall or precision for IT users. Our aim is to curate an LOD specific to the IT and software engineering domains which will also be of smaller size than the above LODs.

Another concurrent objective is to generate RDF quads from technical manuals, user guides and other documentation that are widely available but demand considerable amount of time and effort to sift through. Semanticization of this kind of documentation will contribute to raising the quality of answers (i.e., recall and precision) that the dataset is capable of providing at a fine grain of detail. Extraction of RDF facts from free flowing paragraphs of explanatory texts amounts to solving core problems of natural language processing and semantic information retrieval. Therefore, we take the practical approach of selecting 'low-hanging fruit? by focusing on relatively well-structured technical documentation. Java Platform Standard Edition 7 Documentation API was selected for this purpose because of its definite structure in HTML tables and consistent layout, thereby facilitating extraction of clean and accurate facts. The widespread use of Java in industry and academia is another factor why this documentation API is a good choice for our purposes.

The methodology adopted for constructing the LOD has two parts:

### 2.1 Filtering Existing Datasets

A set of carefully chosen seed words from the domain of IT and software engineering is used to select matching resources from Freebase and DBPedia. For each of these resources, all triples matching the resource in their subject fields are selected. All other triples are filtered out as irrelevant to our domain of interest. These operations were carried out using Unix Shell scripts. The numbers of triples obtained by this method from Freebase and DBPedia are shown in Table 1 below.

### 2.2 Extracting triples from technical documentation

1. A crawler is built to cover the entire documentation collection.
2. A screen scraping package is designed and built to parse the pages and extract facts after a thorough study of the layout of each type of page.
3. Rules are written for logical mapping into quads:
   (a) Check if any reuse of predicates is possible from existing vocabularies like DC, FOAF, SKOS, and SIOC. Reuse wherever possible.
   (b) If compatible predicates are not found, create new predicates that can appropriately describe the relationships. For instance, to link a Java method to its return type or modifier, a new predicate is required.

   (c) Create and publish the required explanatory pages for predicates in RDF and HTML formats. URLs are used to generate quads: (Subject, Predicate, Object, Context) according to the conventions of Linked Open Data.
4. A suitable software library is used to load the quads into a database or data store.

The documentation of Java 7 SE is obtained from the official web site and parsed using a Python Library called BeautifulSoup4 [4]. The LOaD-IT dataset thus obtained has 1.4 million quads. Statistics of the dataset are given in Table 1.

**Table 1.** Numbers and Data Sizes in LOaD-IT

| *Source* | *OriginalSize* | *FilteredSize* | *RetainedQuads* |
|---|---|---|---|
| Freebase | 49 GB | 128 MB | 740149 |
| DBPedia | 44 GB | 184 MB | 433834 |
| Javadocs | < 1 GB | 65 MB | 268275 |
| TOTAL LOaD-IT | | 377 MB | 1442258 |

## 3 Kappa: The LOaD-IT Retrieval Engine

Kappa first constructs the top k query candidates for the given keywords using the methodology outlined in [1]. Once the SPARQL query candidates are generated and the best query is selected, the results are obtained by sending an HTTP request to a REST API provided by our wrapper for LOaD-IT. This API access to LOaD-IT can be used also by other applications in the future. Results obtained as JSON objects are rendered both as triples and graphically to help the user in visualizing the relevant RDF sub-graph of LOaD-IT. An easy-to-use browser-based interface is provided for this purpose. In addition, the query is also sent to external Web search engines (e.g., Bing) as well as on-line technical discussion forums (e.g., StackOverflow). Results from external engines are also displayed to the user to help them look for external content related to the query. Fig. 1(a) shows the overall architecture of LOaD-IT and Kappa.

## 4 Example Scenarios

Two example scenarios and results are presented below to illustrate the usage of Kappa and LOaD-IT.

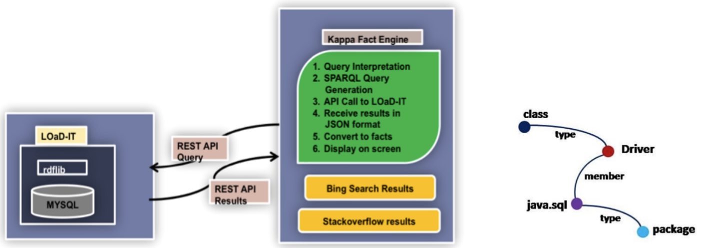
***Scenario 1*** The user wants to know what to write in the import statement in a Java program if he has a need to use a method from the Driver Class in Java. In fact, the user may not even know whether Driver is the name of a Class or a Package.
**Keywords entered: Package Driver**
**Facts retrieved:**

| | | |
|---|---|---|
| java.sql | type | package |
| java.sql | member | Driver |
| Driver | type | class |

A graphical rendering of these three triples in shown in Fig. 1(b)

***Scenario 2*** The user wants to know the details of the Integer Class in Java.
**Keywords entered: Integer**
**Facts retrieved:**

| | | |
|---|---|---|
| Integer | comment | Integer class wraps a value of the primitive type int in an object |
| Integer | member | highestOneBit,hashCode,valueOf,getInteger.. |
| Integer | type | class |



(a) Schematic diagram of the architecture of Kappa

(b) Visualization of Facts Returned by Kappa

**Fig. 1.**

## 5 Conclusion and Future Work

Our primary aim was to create a linked open dataset for the IT domain. After filtering existing datasets, we created a crawler and used screen scraping to extract RDF quads from semi-structured technical documentation. We integrated and normalized the dataset to ensure its compatibility with other LODs and our application Kappa. A reasonably large technical dataset was obtained and its potential was illustrated by a simple yet meaningful and useful application, a fact-finding engine that serves as a technical helpdesk for practicing software engineers.

The dataset can be further enlarged and enriched to include RDF quads extracted from other technical documentation and on-line discussion forums. Further, it can be refined and grown over time by adopting a crowd-sourcing approach. Additionally, a vocabulary can be created specifically for predicates that are useful in a software engineering context and can be standardized to bring in global acceptance. The facts that are rendered can be enriched by natural language processing techniques to bring them closer to proper English.

We expect LOaD-IT to find several other useful applications in the near future. An immediate possibility is to use it as a controlled vocabulary plus ontology for corporate knowledge management. It can also be used to further automate the generation and management of program comments and documentation in software engineering by integrating it with IDE for various programming languages and platforms. None of these is readily possible without LOaD-IT wherein a federated retrieval engine would have to query various general-purpose LODs as well as document repositories and the Web. LOaD-IT together with Kappa promises to deliver the levels of precision and recall to make them useful in real-world software engineering situations.

A demonstration of our application Kappa is available at `http://kanoe.org/lod/loadit-kappa.html`.

## References

1. Tran, T., Wang, H., Rudolph, S. et al.: Top-k Exploration of Query Candidates for Efficient Keyword Search on Graph-Shaped (Rdf) Data. (2009) 405-416
2. Karanth, P., Mahesh, K.: Integrating Knowledge Base Retrieval with Web Search using Semantic Roles In: Lecture Notes in Engineering and Computer Science: Proceedings of the International MultiConference of Engineers and Computer Scientists 2012 (IMECS 2012, ICCS 2012), Vol. 1, ISBN 978-988-19251-1-4 pp. 344-349 IAENG Hong Kong (2012)
3. Bizer, C., Jentzsch, A., Cyganiak, R.: State of the LOD Cloud. Version 0.3 (September 2011), (2011)
4. Richardson, L.: Beautiful Soup-HTML. XML parser for Python, (2008)