FS2KG: From File Systems to Knowledge Graphs (Demo)

Yannis Tzitzikas[1,2,*]

[1]Institute of Computer Science, FORTH-ICS, Greece
[2]Computer Science Department, University of Crete, Greece

Abstract

The tree-structured and semantics-neutral approach of file systems is the dominant method for organizing information, decades now. In this paper we elaborate on the following two questions: (a) can a file system structure be benefited by a Knowledge Graph (KG), (b) can the construction of a KG be facilitated by the file system? To this end we propose an automatic method for producing KGs from folder structures, which can be configured through small, and easy to write, configuration files that can be placed in the desired folders to guide the KG construction. We present `FS2KG`, an implementation of the approach. The approach can facilitate the rapid creation of KGs, as well as various file system related tasks.

Keywords

Knowledge Graph creation, File systems, Semantic Access over File Systems

1. Motivation, Challenges, Methodology and Approach

File systems offer a tree structure consisting of folders and files, and the same structuring is offered by cloud-based file systems too. This simple tree-structured and semantics-neutral approach of file systems is the dominant method with which we organize information for decades. The idea of using the term (and metaphor) folder for designing hierarchical file systems dates back to 1958 [1], while the first file system to support arbitrary hierarchies of directories was used in the Multics operating system in 1965, half a century ago! We could say that the main benefits from the typical hierarchical organization of file systems is that: (a) it allows *grouping* resources (through folders with names and unlimited nesting level), (b) it allows *naming* resources *relatively* to their parent folder, and (c) it allows *moving/copying/deleting* these resources in *one shot*, i.e. all contained resources are moved/copied/deleted. However, a weakness of this structuring method is that each resource (file or folder) should be placed (and appears) in one place. The "shortcuts" that file systems typically offer is a remedy, but it is quite weak (one way links; not bidirectional). Consequently, file systems do not support a multi-faceted approach for locating resources. Two questions that arise are: (a) since Knowledge Graphs (KG) are labeled graphs, and not trees, could this extra expressiveness be leveraged for the contents of our file system? and (b) since there is a need for practical and effective methods for producing KGs, as automatically as possible, could the ubiquitous use (and knowledge

International Semantic Web Conference 2022
tzitzik@ics.forth.gr (Y. Tzitzikas)
https://www.ics.forth.gr/~tzitzik (Y. Tzitzikas)
0000-0001-8847-2130 (Y. Tzitzikas)

of using) file systems, be leveraged for speeding up, or just facilitating, the creation of KGs? Both directions could have significant impact. The first would enable leveraging the Semantic Web technologies in every day tasks. The second would assist the creation of KGs, something desirable, since there is a need for practical and mature tools to foster knowledge engineering (there are some critiques about the practicality and availability of tools for the Semantic Web, e.g. see [2], and an elaborated discussion of these critiques at [3]).

Challenges: Enriching a file system with a KG is a challenging task, since a file system contains very heterogeneous material since it is used for various purposes and tasks. For instance, one part of the file system may contain training material (books, papers, slides, assignments, student exercises), another part various personal material (family documents, photos and videos, travel information), datasets, software code and systems and others. Moreover, applications also use the file system and create and modify parts of it.

Methodology: We started by inspecting existing file system structures, and reflecting on what we would like to achieve, what file system weakness we would like to tackle. We came up with various ideas, that we implemented and tested, and only those that seemed effective were included in the proposed tool FS2KG.

Approach: In brief, we propose supporting two fundamental interrelated aspects: *folder structure* and *semantic network* with connections between these two. The core schema is illustrated in Figure 1 (upper part), where with "*" we denote multiplicity (as in UML Class Diagrams). The approach is equipped with methods that create entities based on the files and folders of the file system, as well as by extracting them from csv files. The big picture is sketched in Figure 1 (lower part).

Related Work: In comparison to the line of research under the term "semantic desktop" that was developed 15 years ago (e.g. [4, 5, 6, 7, 8]), we could say that the current work has a more modest, but realistic, vision: not to integrate data, applications, and tasks, but to focus on the data part (folders and files). As pointed out in [9], existing Semantic Desktops are either too complicated, or not scale well, and a real "killer app" is still missing. The approach proposed in this paper is more tightly related with the classical file system usage. It adopts a modular configuration approach, there is no dependency to a central repository, or central configuration, or any other service.

2. The Functionality of FS2KG

FS2KG supports a default operation that requires no configuration. It starts by traversing the file system from the desired folder(s). Each folder is represented by a class, subfolder relationships by `rdfs:subClassOf`, while each file is represented as a named individual classified under the class of its folder. However the user can place a ".kg" file in some folders to configure the creation of the KG in the corresponding part of the file system. In particular, a ".kg" file contains configuration parameters, in the form of key-value pairs. It supports commands: (I) for **scope restriction**, i.e. with `traverse=off` the traversal stops, and we can ignore files based on their extension, e.g. `ignoreExt=tmp;aux`. (II) for the **automatic creation and classification of entities corresponding to subfolders**, e.g. with `subFoldersClass=example:Student` for each subfolder of the hosting folder an entity is created (belonging to the *Semantic Network* view), with

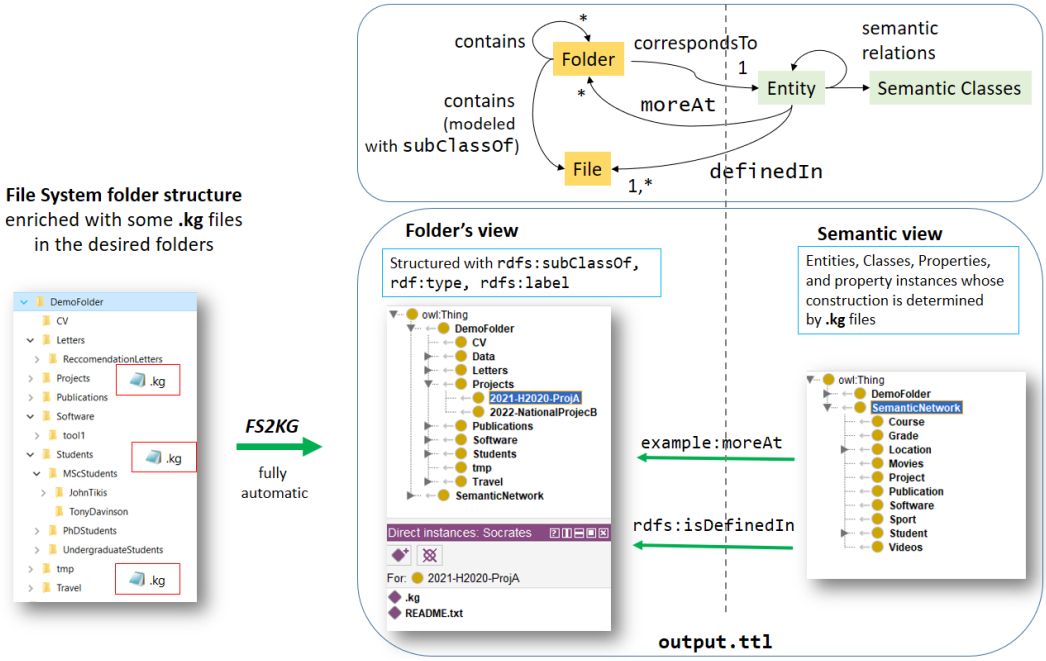

Figure 1: The core connections and the big picture

a link `example:moreAt` pointing to the class of that folder. An example is shown in Figure 2 where with `subFoldersClass=example:Student` in the ".kg" file of the folders `RecomendationLetters` and `MScStudents` we managed to get one entity for each student which is connected with these folders. (III) for **leveraging 'readme' files**, i.e. with `readme=on` if we encounter readme files they are connected with the entities of the corresponding folders. (IV) for **adding arbitrary metadata**, i.e. extra, explicitly specified, triples can be associated to a file, say `f1.pdf`, by placing them in a file `f1.kg` in the same folder. (V) for **extracting and transforming data from the desired csv files**. Specifically `FS2KG` adopts the following convention: If we want to perform extraction from a file, say "`fn.y`", we can create a file "`fn.kg`" where we place rules to extract data from the corresponding file. We support an easy to use language (much simpler than existing approaches, like [10]), with which we can construct RDF triples (data, taxonomies, ontologies), from csv files. For example, suppose a file with name `Connections.txt` that contains lines of the form: `Leonardo;Rome;Football`.

We can place in the same folder a file `Connections.kg` with:

```
C1=example:Student
C2=example:Location
C3=example:Sport
R=C1,example:livesAt,C2;C1,example:likes,C3
```

Property `C1` refers to the first column, and its value means that the values that occur in the first column of the data file should become instances of the class `example:Student`. Consequently, with the first three lines (properties C1-C3), we manage to classify all values that appear in the csv file to the classes `Student`, `Location` and `Sport`. The last row contains two rules, separated

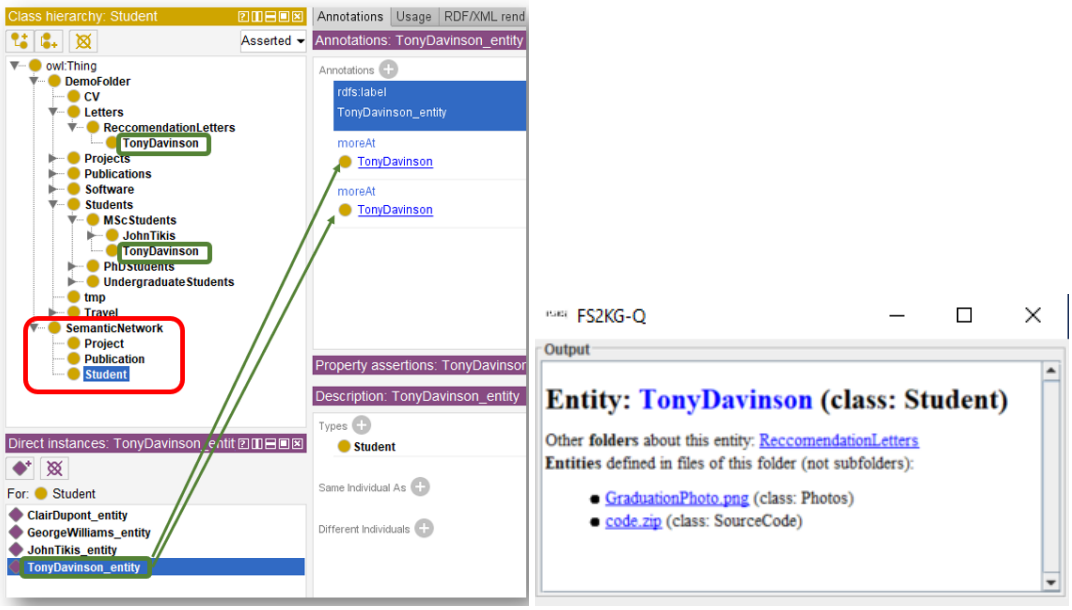

Figure 2: Left:The two class hierarchies (folder's view and Semantic Network), and their connection through entities, Right: the GUI of the query client)

by semicolon, for creating relationships. The first is "`C1,example:livesAt,C2`" that states that the values in C1 should be connected via `example:livesAt` with the values of C2. Analogously, the second rule relates the values of the first column with the values of the third column. If we also have `provenance=on` then these extracted entities will be connected through `rdfs:isDefinedIn` with the corresponding file. FS2KG also offers a light weight query client, shown in Fig. 2(right).

Efficiency. The application of FS2KG over a file system of 140 GB that contains 60 K folders and 382 K files takes only 90 seconds and produces a ttl file of size 140 MB.

Use Cases. We can identify two main scenarios: (S_1) Over existing file systems *to enable querying, identification and grouping of entities scattered in different subfolders.* To this end, one immediate next step is the implementation of an explorer that combines the functionality of the classical file explorer with the query client (as shown in Figure 2).

(S_2) Over folder structures and files created *for facilitating KG construction.* For example, the user can use the file system to define a taxonomy (e.g. of papers organized in categories), instead of having to use a taxonomy/ontology editor. Moreover, arbitrary KGs can be constructed from csv files through FS2KG and the supported extraction language.

3. Conclusion

Finding an effective method to conciliate freedom of file system usage, and Knowledge Graph integrity and usability, is a challenging task. We will demonstrate FS2KG a tool for the automatic creation of KGs from file systems that supports a modular (and easy to use) configuration approach relying on small configuration files in the folders, and KG reconstruction at any moment. The tool is open source and available at https://github.com/YannisTzitzikas/FS2KG,

subject to a plethora of extensions. We have decided to include in FS2KG a sort of core functionality. On top of this, several straightforward extensions are applicable (since they have already been studied in isolation) including: (a) representation of the filesystem's file metadata in RDF (as in [4]), (b) extraction of the embedded in the files metadata and representation in RDF (as in [11]), (c) instance matching over the KG to establish connections between entities whose name is slightly different in different folders, (d) regex-based specification of the desired files/folders (as in web crawlers), (e) information extraction capabilities from files according to their type (text, images, etc) based on the application context and requirements at hand (including scripts in the '.kg' files), (f) materialization of the extracted triples from big csv files, to avoid re-extracting them in the next KG reconstruction, if the files have not been changed in the meantime, and (g) keyword search based on both the contents of the files and produced KG (as in [12]).

References

[1] G. Barnard III, L. Fein, Organization and retrieval of records generated in a large-scale engineering project, in: Papers and discussions presented at the December 3-5, 1958, eastern joint computer conference: Modern computers: objectives, designs, applications, 1958, pp. 59–63.

[2] R. Verborgh, M. Vander Sande, The semantic web identity crisis: in search of the trivialities that never were, Semantic Web 11 (2020) 19–27.

[3] A. Hogan, The semantic web: Two decades on, Semantic Web 11 (2020) 169–185.

[4] C. Jenkins, M. Jackson, P. Burden, J. Wallis, Automatic RDF metadata generation for resource discovery, Computer Networks 31 (1999) 1305–1320.

[5] L. Sauermann, A. Bernardi, A. Dengel, Overview and outlook on the semantic desktop., in: Semantic Desktop Workshop, volume 175, Citeseer, 2005.

[6] B. Schandl, SemDAV: a file exchange protocol for the semantic desktop., in: SemDesk'06: Proceedings of the 5th International Conference on Semantic Desktop and Social Semantic Collaboration, 2006.

[7] L. Sauermann, L. Van Elst, A. Dengel, PIMO - a framework for representing personal information models, Proceedings of I-Semantics 7 (2007) 270–277.

[8] L. Drăgan, S. Decker, Knowledge management on the desktop, in: International Conference on Knowledge Engineering and Knowledge Management, Springer, 2012, pp. 373–382.

[9] C. Jilek, M. Schröder, S. Schwarz, H. Maus, A. Dengel, Context spaces as the cornerstone of a near-transparent and self-reorganizing semantic desktop, in: European Semantic Web Conference, Springer, 2018, pp. 89–94.

[10] Y. Marketakis, N. Minadakis, H. Kondylakis, K. Konsolaki, G. Samaritakis, M. Theodoridou, G. Flouris, M. Doerr, X3ML mapping framework for information integration in cultural heritage and beyond, International Journal on Digital Libraries 18 (2017) 301–319.

[11] Y. Marketakis, M. Tzanakis, Y. Tzitzikas, Prescan: towards automating the preservation of digital objects, in: Proceedings of the International Conference on Management of Emergent Digital EcoSystems, 2009, pp. 404–411.

[12] C. Nikas, G. Kadilierakis, P. Fafalios, Y. Tzitzikas, Keyword search over RDF: Is a single perspective enough?, Big Data and Cognitive Computing 4 (2020) 22.