The Design and Development of a Semantic File System Ontology

Syed Rahman Mashwani
Department of Computer Science
University of Peshawar
Peshawar, Pakistan
syed.rahman@uop.edu.pk

Shah Khusro
Department of Computer Science
University of Peshawar
Peshawar, Pakistan
khusro@uop.edu.pk

***Abstract*—Semantic File System (SFS) is the vision for the future of file systems where information is given with explicit meaning to be processed by machines automatically and consumed by the users easily. SFSs extend traditional file systems to organize and retrieve information according to their semantics, intent and relationships with other resources rather than their physical locations. Ontology-based file system is a step to dissolve the borders between semantic web and semantic desktop to make the desktop part of a single giant web. Unfortunately, to the best of the authors' current knowledge, so far, no effort has been exercised to develop an ontology for SFSs. This work contributes a SFS ontology, which extends NEPOMUK information element ontology framework into the domain of SFSs. The proposed SFS ontology is freely available with full technical definitions of terms and complete class hierarchy to be used for any purpose in information technology (IT). In addition, as a proof-of-concept implementation, we deploy the proposed ontology in the 360° SFS. Finally, to get most of the benefits of the ontology, this paper also presents a semantics-aware file manager.**

Keywords-file systems; information management; information retrieval, semantic desktop

I. INTRODUCTION

Traditional file systems organize files in hierarchy of folders which have certain limitations. In addition, information contained in files varies in structure. Semantics of the information is restricted to the boundaries of their applications, and therefore, are difficult to mine and retrieve. For efficient and precise retrieval of information, machines must be able to understand it, which needs semantics. Semantic web (SW) [1] deals with information overload on the web. Efforts have been put in binging this semantics-awareness to the desktop [2], which has resulted in improvements especially in file retrieval [3]. However, for file systems to be effective and accurate, semantic desktop needs semantic file system (SFS) immensely [4]. Ontology-based SFS has the potential to dissolve the borders between SW and semantic desktop (SD) to make the desktop part of a single giant web. However, to the best of our knowledge, we found no file system ontology published according to linked data principles [5]. Therefore, the main contribution of this paper is the development of an SFS ontology and its publishing on the web to be freely accessible and reusable by other researchers. We deploy the SFS ontology on our developed SFS called 360°-SFS. In addition, this paper presents a semantic-aware file manager (FM) called 360°-FM that is based on the same ontology.

II. MOTIVATION AND NEED

This section elaborates the potential role of SFS ontology with the help of motivating scenarios. These scenarios have been used in the evaluation of SFS ontology in section V.

A. Scenario 1: Single Centralized Machine Friendly Metadata Repository For All Agents

Traditional file systems are limited to the basic metadata about files. The information is preserved by the operating system and application software using different structures, which are not machine friendly. In addition, the application must be aware of the structure of the file in order to use it. Moreover, it became difficult for each application to understand the structure of all possible file types. Therefore, it is necessary to make available the file metadata in a single, sharable repository with easy to understand structure so that any application can easily use it.

B. Scenario 2: Advanced Search Queries

Traditional file systems maintain basic file metadata including file name, date created, modified and time accessed etc., which is exploited by search engines. A semantically enhanced file system could maintain an adequate amount of metadata in a single shared repository with rules enough to perform reasoning by SFSs.

C. Scenario 3: Generating File Recommendations

A user often faces difficulties while organizing files. To handle this issue, machines must be aware of the semantics of directories. This knowledge could be exploited in recommending directories to the users while they organize files. In addition, a directory having name Tim Berners-Lee can be linked e.g., using *owl:sameAs* to his URI or any other resource having matching labels or a music album directory in the file system can be linked to the URI of that album in an Linked Open Data (LOD) set. Linking local directories and files to LOD enables the user to explore additional related information. In a similar way, files could also be recommended

to be interlinked with other files or simply deleted if user intends to do so. E.g.,

- If a user deletes some songs of an album in directory A then songs of the same album that are stored in directory B can be recommended for deletion.
- A file that was created together or frequently accessed with the deleted file can also be recommended.
- Similarly, duplicated, near duplicated or earlier versions of deleted files can be recommended for deletion.

D. Scenario 4: Static Contents in Directories

Traditional file systems organize files in native structures (directories), which are taxonomy-oriented. One of the shortcomings of native structure is that a directory always shows the same contents no matter in what context a user is accessing information. A file importance changes according to different factors, such as file access frequency, time, user's geographical location, or a user obligations (a user may have many appointments, meetings, presentations, etc. in a day). But a directory shows fragments of multiple contexts rather than a single context. A semantically enhanced system could present task specific files to a user. Files could be dynamically organized in such a way that a user can access important files easily.

III. SEMANTIC FILE SYSTEM ONTOLOGY DEVELOPMENT

Ontology can either be developed from scratch or by re-using existing ontologies and vocabularies. Building ontology from the scratch may result in multiple disparate vocabularies for similar entities making the data integration task difficult [2]. Ontology reuse exploits the existing ontological knowledge to create new ontologies [6]. Several existing ontologies describing portions of large domain can be integrated while developing a large ontology [7] to address new or emerging problems. In addition, this reuse reduces the cost and improves the quality of resulting ontology. Moreover, the interoperability among applications gets increased as it provides a deeper, machine-processable and commonly agreed upon understanding of the underlying domain of interest. Without reuse, the lack of integration will treat ontology like traditional static knowledge bases and therefore, will not be able to contribute to the realization of the SW [6, 8]. It is always encouraged not to reinvent the wheel but to use or extend terms from existing widely used vocabularies[7, 9-11]. We follow the same practice of reusing terms from existing popular vocabularies. Based on [11], we performed the following steps for reusing existing ontologies in developing an SFS ontology.

A. Step 1. Discovery of Ontologies for Reuse

Finding ontological resources is an important step for reuse. In SW, ontologies as well as their concepts and properties are identified via URIs, which makes ontologies and ontological entities accessible on the web and discoverable by crawlers [6]. In searching relevant ontologies, we used traditional web search engines like Google, SW search engines including Swogle [12], Watson [13], OntoSearch [14], ontology repositories including DAML ontology library, Protégé OWL library, etc., and Linked Open Vocabularies.

B. Step 2. Selecting Relevant Reuse Candidates

Table I depicts vocabularies that we have manually selected for possible reuse after carefully proofreading. NEPOMUK intends to bring solution to annotate and interlink arbitrary information on the local desktop making it machine processable. In addition, it enables interconnection and information exchange among desktops [15]. NEPOMUK ontologies are carefully crafted by a large team of experts over many years. Its aim is to provide a unified vocabulary for describing desktop resources. W3C Basic Geo is a widely used vocabulary for representing latitude, longitude and altitude information about spatially-located things.

C. Step 3. Customization, Extending and Integrating Relevant Ontologies

The identified reusable ontologies were integrated into the final application ontology. We mostly reused terms of NEPOMUK ontologies. NEPOMUK ontologies provide basic terms to describe desktop resources. To capture more semantics and to best fit in the domain of SFSs we extend them by defining new properties. Table II presents the competency questions to be answered by an SFS ontology.

IV. TERMS DESCRIPTION OF SFS ONTOLOGY

We follow Linked Data best practices [5] while designing our SFS ontology. The namespace of the ontology is defined as "https://w3id.org/sfs-ontology#". Each term in the ontology is accessible via persistent HTTP URIs with content negotiation capability. If a URI is accessed by a human, then human-readable information is provided and if it is accessed by a machine then machine-understandable information is provided to client. The web server is configured to do this redirection by exploiting 303 See Other HTTP status code. To maximize interoperability, almost all classes and most of the properties are reused from NIE Ontology Framework (see Table I). Where applicable, we defined new terms to extend them into the domain of SFSs. In the following, we describe the newly defined/extended terms.

A. Classes Description

FileAccess

FileAccess is uniquely identified by time, latitude, longitude and altitude. An object of *FileAccess* is created upon each file access. The *fileAccessLocation* property relates objects of *nfo:FileDataObject* and *FileAccess* classes. This enables a user to track accessed time and location of a file.

Super Class: rdfs:Resource
In Domain of: fileAccessedLocation
In Range of: nfo:FileDataObject
Restrictions:
sfs:fileAccessedTime *Some(Existential)* dateTime,
sfs:fileAccessedTime *Maximum Cardinality* 1,
geo:location *Some (Existential)* geo:Point,
geo:location *Maximum Cardinality* 1,

TABLE I. REUSE CANDIDATES. THE PERCENTAGE REFERS TO THE TOTAL NUMBER OF REUSED (R)/NEWLY DEFINED (N) TERMS IN SFS ONTOLOGY

Vocabulary Name	Category	Concepts	Properties		Total Terms	% of terms
			Object	Data		
SFS Ontology https://w3id.org/sfs-ontology/	SFS	1	15	7	23 (N)	10.8
NEPOMUK Information Element Core Ontology (NIE) http://www.semanticdesktop.org/ontologies/2007/01/19/nie/	General/ Documents	10	14	28	33 (R)	15.6
NEPOMUK File Ontology (NFO) http://www.semanticdesktop.org/ontologies/2007/03/22/nfo	General/ Documents	58	17	57	63 (R)	29.7
NEPOMUK EXIF Ontology (NEXIF) http://www.semanticdesktop.org/ontologies/2007/05/10/nexif/	Multimedia	5	8	148	47 (R)	22.2
NEPOMUK ID3 Ontology (NID3) http://www.semanticdesktop.org/ontologies/2007/05/10/nid3	Multimedia	10	28	29	29 (R)	13.7
Basic Geo Vocabulary (Geo) http://www.w3.org/2003/01/geo/	Location	2	1	4	6 (R)	2.8
NEPOMUK Annotation Ontology (NAO) http://www.semanticdesktop.org/ontologies/2007/08/15/nao/	Annotations	9	17	15	3 (R)	1.4
NEPOMUK Contact Ontology (NCO) http://www.semanticdesktop.org/ontologies/2007/03/22/nco	Contact	41	33	34	2 (R)	0.9
NEPOMUK Multimedia Ontology (NMM) http://www.semanticdesktop.org/ontologies/2009/02/19/nmm	Multimedia	12	16	21	6 (R)	2.8
Total number of terms in SFS Ontology					**212**	**100%**

TABLE II. COMPETENCY QUESTIONS

Which files are temporally related to user's current context?	Which files are related to the user's current geographical context?	Get me those files which are frequently accessed by the user at a specific geographical location	Get me those files which are frequently accessed by the user at specific temporal duration
Show me all those files which are frequently accessed with the File A	Show me all related files which are in the child directories of the Path XYZ	Show me all related files which are in the parent directories of the Path XYZ	Get me those files which are tagged with (W OR X) AND Y but NOT Z
Get me all files created by XYZ application	Get me all files created by XYZ hardware (Digital Camera etc.)	Get me those files accessed at a specific geographic point, city or country etc.	Get me all those pictures taken with camera XYZ at geographical location XYZ and having resolution higher than XYZ.
Show me duplicate and near duplicate files of file XYZ	Show all those files which are recently created or modified by application XYZ	Show me files which are created together with file XYZ	Get me those files which are frequently accessed on Sunday

B. Properties Description:

fileAccessedLocation_Time

It relates objects of *nfo:FileDataObject* and *FileAccess* classes, which enables a user to track the accessed time and location of a file so that frequently accessed timings and locations can be calculated.

Domains: nfo:FileDataObject
Ranges: FileAccess

belongedToContainer

It is the previous container of a file, which models the containment relations between normal or compressed files and folders at their previously stored locations. A user groups semantically related files in a folder. If a user places a file in one folder and then after some time he/she moves it to another folder then it indicates that files in both folders are somehow related to each other. Tracking previous containers/folders of a file enables relating files based on these semantics.

Domains: nie:DataObject
Ranges: nfo:DataContainer
Disjoint With: nfo:belongsToContainer

fileCreatedByApplication

This property relates a file to the application with which it was created. It helps in maintaining provenance of a file.

Domains: nfo:FileDataObject
Ranges: nfo:Application
Characteristics: Functional

fileModifiedByApplication

This property tracks applications that make changes to a file. This property helps in maintaining provenance of a file.

Domains: nfo:FileDataObject
Ranges: nfo:Application

fileCreatedByUser

It is the user who creates the file. This property helps in maintaining provenance of a file.

Domains: nfo:FileDataObject
Ranges: nco:Contact
Characteristics: Functional

fileCreatedFrom

A file created from (the contents of) another file. If file B is created from file A and file C is created from file B then it can be inferred that file C is created from file A.

Domains: nfo:FileDataObject
Ranges: nfo:FileDataObject
Characteristics: Transitive

fileCreatedTogather

It relates two files that are created together within a predefined threshold time.

Domains: nfo:FileDataObject
Ranges: nfo:FileDataObject
Characteristics: Symmetric

fileDuplicateOf

It relates a file to its exact match. Files are duplicates of each other if they are content-wise 100 % identical.

Domains: nfo:FileDataObject
Ranges: nfo:FileDataObject
Characteristics: Symmetric

fileFrequentlyAccessedWith

It represents a file which is frequently accessed with another file.

Domains: nfo:FileDataObject
Ranges: nfo:FileDataObject
Characteristics: Symmetric

fileModifiedByUser

It tracks users who modify the file. This property helps in preserving provenance information of a file.

Domains: nfo:FileDataObject
Ranges: nco:Contact

fileCreatedOnDevice

It represents the computer or device on which a file is created. This property helps in maintaining provenance of a file.

Domains: nfo:FileDataObject
Ranges: -
Characteristics: Functional

fileModifiedOnDevice

It tracks devices/computers on which a file is modified. A file is annotated each time if it is modified on a different device. This property helps in preserving the provenance information of the file.

Domains: nfo:FileDataObject
Ranges: -

fileManualPeakLocation

Manual location based annotation. The difference between filePeakLocation and fileManualPeakLocation is that in the former annotation is done automatically as a user interacts with files and in the later the annotation is done manually by a user.

Domains: nfo:FileDataObject
Ranges: geo:Point

fileModifiedByApplication

This property relates a file to an application that makes changes to a file. It helps in maintaining provenance information of a file.

Domains: nfo:FileDataObject
Ranges: nfo:Application

fileNearDuplicateOf

This property relates two files that are identical to some extent (e.g. older versions of the same file). It uses a similarity threshold to detect near-duplicates. E.g. If similarity between two files is greater than 80%, then both files are "near duplicates".

Domains: nfo:FileDataObject
Ranges: nfo:FileDataObject
Characteristics: Symmetric

filePeakLocation

This property represents the geographical location where a file is frequently accessed.

Domains: nfo:FileDataObject
Ranges: geo:Point

fileAccessedCounter

This property means the number of times a file is accessed.

Domains: nfo:FileDataObject
Ranges: xsd:integer
Maximal cardinality: 1

fileAccessedTime

File accessed time.
Domains: nfo:FileDataObject, sfs:FileAccessedTime
Range: xsd:dateTime

fileAccessedHistory

The date and time is recorded each time a file is accessed to maintain a file’s accessed history.

Domains: nfo:FileDataObject
Ranges: xsd:dateTime

fileModifiedHistory

The date and time is recorded each time a file is modified to maintain a file’s history.

Domains: nfo:FileDataObject
Ranges: xsd:dateTime

filePeakDay

This property represents a file's peak day, the day on which a file is frequently accessed. Its value ranges from 0-6, with 0 being Sunday.

Domains: nfo:FileDataObject
Ranges: xsd:integer

filePeakHour

This property represents hour of the file. The value ranges from 0 to 23.

Domains: nfo:FileDataObject
Ranges: xsd:integer

contentType

This property represents the minimum age restriction of the user to access a file or folder. If its value is set to 18 then the contents will not be accessible to non-adults.

Domains: nfo:FileDataObject, nfo:DataContainer
Ranges: xsd:integer
Maximal cardinality: 1

V. DEPLOYING ONTOLOGY

The ontology was developed using Protégé and is freely available under a Creative Commons Attribution license from https://w3id.org/sfs-ontology/. SFS ontology was deployed on our developed file system called 360º-SFS, which is freely available from https://w3id.org/360-sfs/. The 360º-SFS automatically creates objects of classes and annotates them while the user interacts with file system resources. We also developed a file manager which enables the user to browse files semantically by exploiting the same centralized knowledgebase.

A. 360º Semantic File System

For proof-of-concept implementation, we deployed the proposed ontology on 360º-SFS. The 360º-SFS enables a user to semantically retrieve files in less time and with less human effort. This is done via NOW and TAGs special folders. The contents of these special folders change dynamically according to the user's current context, as discussed in scenario 5. The system creates instances of the classes and annotates them according to user's interaction with file system resources. This is done with the help of file system events. For instance, if a user creates a file or a folder in the file system then an instance of the *nfo:LocalFileDataObject* or *nfo:Folder* is created respectively and annotated with the required information. If a user deletes a file then upon file delete event that particular object of *nfo: LocalFileDataObject* is deleted from knowledge base along with its all references.

B. 360º File Manager

360º-SFS is backward compatible with traditional file managers as well as all other software applications. However, access to full semantics is not possible through traditional file managers. To make the most from the file system ontology, file managers need to be ontology aware to manipulate file system resources accordingly. Therefore, to further facilitate user's interaction with 360º-SFS, we propose 360º-File Manager (FM), (Figure 1). The 360º-FM extends traditional hierarchical file organization by overcoming some of its limitations. In the following, we explain the interface and functionality of 360º-FM:

1. **"Traditional Pane"** provides access to the stored files in a way like that of traditional file managers.
2. **"Semantic Pane"** enables a user to access files semantically. It shows the contents of NOW and TAGs directories, eliminating the need of going into NOW and TAGs directories.
   2.1. **"Semantic Sub-Pane-1"** This pane has two tabs: NOW and TAGs. NOW tab recommends files only on the basis of temporal and user's geographical location while TAGs tab facilitates file management based on tags. Now tab shows only those files that are accessed frequently at current time and geographical location. The items in Sub-Pane-1 (contents of NOW and TAGs tabs) are computed on the basis of currently selected directory (territory) in Traditional Pane. Only those links are shown in Sub-Pane-1, which are stored within a selected directory. All files stored in the parent directories of the selected directory are ignored. Each time the user selects another directory in Traditional Pane, the contents of Sub-Pane-1 are refreshed to show contextually relevant files accordingly.
   2.2. **"Semantic Sub-Pane-2"** shows files which are semantically related to the currently selected file in the details view of Traditional Pane. Each time a user selects a file in details-view of Traditional Pane, contents of Sub-Pane-2 are refreshed to show contextually relevant files accordingly. The Internal tab shows links to the file system internal files while Linked Cloud tab relates the selected files to linked cloud. The linked cloud could be private linked cloud or Linked Open Data cloud. The Show Inferences check box enables or disables inferences in Sub-Pane-2.

The Semantic Pane is the only difference between 360º-FM and a traditional file manager. In addition, plugins can be developed for traditional file managers to show the semantic contents in a way similar to Semantic Pane. A file manager may allow a user to sort files based on context relevancy. Furthermore, file icon size can increase or decrease as per its relevance to the current context and color gradient of the file's icon can also be modified to make the important files prominent in a directory. As stated in scenario 1, 360º-SFS maintains a single centralized machine friendly metadata repository. The repository is accessible to all agents and advanced queries can be performed, as discussed in scenario 2. The 360º-FM exploits the repository and enables file retrieval semantically. In addition, different applications can exploit the repository for recommendations within their respective interfaces and also semantically related files can be recommended along with Recently Opened Files.

VI. RELATED WORK

The term Semantic File System was first in [16]. File type specific transducers mine and index attributes from files. They used virtual directories for file navigation based on the extracted attributes. Authors in [17] further extended the idea and proposed the SFS that also considers users to define arbitrary metadata for files in the form of key-value pairs. The work of [18-22] are based on tags semantics that enable users to retrieve files based on tags assigned to them. GFS [23] is also a tag-based file system but its advantage over other file systems is that it provides associative access to tags based navigation and does not replace traditional hierarchical folder navigation. Authors in [24] proposed the changing of folder and file icons according to their popularity to assist the user to

locate important files easily. Files are uploaded to a webserver. The popularity of files is calculated based on the users' annotations. In LiFS [25], authors propose linking files internally in a file system and offer support for arbitrary annotations. Authors in [26, 27] propose linking of desktop and web resources. The proposed system enables a user to browse web resources as if they are stored on desktop. In [28], authors describe different benefits of ontology based file systems. Although, many SFSs have been proposed, to the best of our knowledge, no effort has been exercised till date to develop ontology for SFSs and make it available for public use by publishing it according to linked data principles.

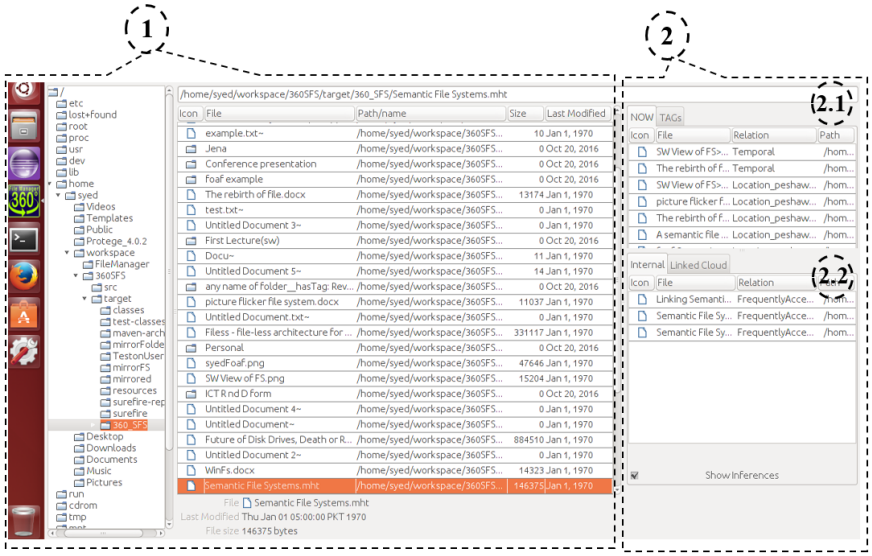

Fig. 1. 360º File Manager – An interface for interacting with file system resources

VII. CONCLUSION

The main contribution of this paper lies in the shape of SFS ontology. SFS ontology intends to provide vocabulary upon which SFSs could be built. We designed and published SFS ontology by following linked data best practices. For interoperability, we reused or extended terms from existing popular vocabularies. The terms of the proposed SFS ontology are provided with persistent URIs with content negotiation capability. For the proof-of-concept implementation, we have deployed our developed 360º-SFS and 360º File Manager using the proposed ontology, which enables the retrieval of files on the basis of semantics, intent and relationships with other files rather than their physical locations as with traditional file systems. The proposed SFS ontology is by no means complete, more terms may appear in future. The SFS ontology is freely available with full technical term definitions and complete class hierarchy for any purpose in information technology. SFS ontology remains open for additions and corrections upon feedback.

REFERENCES

[1] T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web", Scientific American, Vol. 284, No. 5, pp. 28-37, 2001

[2] A. Mylka, L. Sauermann, M. Sintek, L. van Elst, NEPOMUK information element ontology, available at: http://oscaf.sourceforge.net/nie.html, 2013.

[3] B. Schandl, B. Haslhofer, "The sile model: A semantic file system infrastructure for the desktop", The Semantic Web: Research and Applications. ESWC 2009. Lecture Notes in Computer Science, Vol. 5554, pp. 51-65, Springer, Berlin, Heidelberg, 2009

[4] L. Sauermann, A. Bernardi, A. Dengel, "Overview and outlook on the semantic desktop", Proceedings of the 2005 International Conference on Semantic Desktop Workshop: Next Generation Information Management D Collaboration Infrastructure, Vol. 175 pp. 74-91, 2005

[5] T. Berners-Lee, "Linked Data", available at: http://www.w3.org/DesignIssues/LinkedData.html, 2006

[6] E. Simperl, "Reusing ontologies on the Semantic Web: A feasibility study", Data & Knowledge Engineering, Vol. 68, No. 10, pp. 905-925, 2009

[7] N. F. Noy, D. L. McGuinness, Ontology development 101: a guide to creating your first ontology, available at: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf, 2001

[8] R. Stecher, C. Niederée, W. Nejdl, P. Bouquet, "Adaptive ontology re-use: Finding and re-using sub-ontologies", International Journal of Web Information Systems, Vol. 4, No. 2, pp. 198-214, 2008

[9] B. Schandl, B. Haslhofer, "Files are siles: Extending file systems with semantic annotations", International Journal on Semantic Web and Information Systems, Vol. 6, No. 3, pp. 1-21, 2010

[10] M. Poveda Villalón, M. C. Suárez-Figueroa, A. Gómez-Pérez, "The landscape of ontology reuse in linked data", in 1st Ontology Engineering in a Data-driven World (OEDW 2012) Workshop at the 18th International Conference on Knowledge Engineering and Knowledge Management (EKAW2012), Galway, Irlanda, October 8-12, 2012

[11] M. C. Pattuelli, A. Provo, H. Thorsen, "Ontology building for linked open data: A pragmatic perspective", Journal of Library Metadata, Vol. 15, No. 3-4, pp. 265-294, 2015

[12] T. Finin, Y. Peng, R. S. Cost, J. Sachs, A. Joshi, P. Reddivari, R. Pan, V. Doshi, L. Ding et al., "Swoogle: A search and metadata engine for the Semantic Web", 13th ACM International Conference on Information and Knowledge Management, pp. 652-659, ACM, 2004

[13] M. d' Aquin, M. Sabou, M. Dzbor, C. Baldassarre, S. Angeletou, E. Motta, "Watson: A gateway for the Semantic Web", in Poster session of the European Semantic Web Conference, ESWC, Innsbruck, Austria, June 3-7, 2007

[14] Y. Zhang, W. Vasconcelos, D. Sleeman, "Ontosearch: An ontology search engine", Research and Development in Intelligent Systems XXI, pp. 58-69, Springer-Verlag London, 2005

[15] T. Groza, Z. Handschuh, K. Möller, G. Grimnes, L. Sauermann, E. Minack, C. Mesnage, M. Jazayeri, G. Reif, R. Gudjónsdottir, "The NEPOMUK project-on the way to the social semantic desktop", Proceedings of I-Semantics 07, Graz, Austria, pp. 201-211, 2007

[16] D. K. Gifford, P. Jouvelot, M. A. Sheldon, J. W. O'Toole Jr, "Semantic file systems", Proceedings of the thirteenth ACM symposium on Operating systems principles, Pacific Grove, California, United States, pp. 16-25, October 13-16, 1991

[17] D. Garg, V. Mehta, S. Pandit, M. De Rosa, "A writable semantic file system", in: Selected Project Reports, Fall 2005 Advanced OS & Distributed Systems (15-712), pp. 56-65, Carnegie Mellon University, Pittsburgh, 2005

[18] Y. Padioleau, B. Sigonneau, O. Ridoux, "Lisfs: A logical information system as a file system", Proceedings of the 28th International Conference on Software Engineering, pp. 803-806, 2006

[19] Tx0, "Tagsistant: Semantic file system", available at: http://www.tagsistant.net, 2007

[20] S. Schenk, O. Görlitz, S. Staab, "TagFS: Bringing semantic metadata to the filesystem", in: Poster at the 3rd European Semantic Web Conference (ESWC), Budva, Montenegro, June 11-14, 2006

[21] S. Bloehdorn, O. Görlitz, S. Schenk, M. Völkel, "Tagfs - tag semantics for hierarchical file systems", Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06), Graz, Austria. Vol. 8, pp. 6-8, 2006

[22] S. Bloehdorn, "SemFS - semantic file system", available at: http://semanticweb.org/wiki/SemFS, 2009

[23] D. Di Sarli, F. Geraci, "GFS: a Graph-based File System Enhanced with Semantic Features", Proceedings of the 2017 International Conference on Information System and Data Mining, pp. 51-55, April 1-3, 2017

[24] B. Mizrachi, L. S. Deluca, File folder display, US Patent 20170109010, 2017

[25] S. Ames, N. Bobb, K. M. Greenan, O. S. Hofmann, M. W. Storer, C. Maltzahn, E. L. Miller, S. A. Brandt, "LiFS: An attribute-rich file system for storage class memories", 23rd IEEE/14th NASA Goddard Conference on Mass Storage Systems and Technologies, May 15-18, 2006

[26] B. Schandl, "TripFS: Exposing file systems as linked data", Linked Open Data Triplification Challenge, Graz, Austria, September 2-4, 2009

[27] B. Schandl, "Representing linked data as virtual file systems", 2nd International Workshop on Linked Data on the Web Madrid, Spain, April 20, 2009

[28] H. B. Ngo, C. Bac, F. Silber-Chaussumier, T. Q. Le, "Towards ontology-based semantic file systems", 2007 IEEE International Conference on Research, Innovation and Vision for the Future, Hanoi, Vietnam, pp. 8-13, March 5-9, 2007