Web Service APIs for Scribe Registrars, Nexus Diristries, PORTAL Registries and DOORS Directories in the NPD System

Adam G. Craig, Seung-Ho Bae, Teja S. Veeramacheneni,
S. Koby Taswell, and Carl Taswell

`www.BrainHealthAlliance.org`
`ctaswell@BrainHealthAlliance.org`
8 Gilly Flower Street, Ladera Ranch, CA 92694 USA

Abstract. The Nexus-PORTAL-DOORS System (NPDS) has been designed with the Hierarchically Distributed Mobile Metadata (HDMM) architectural style to provide an infrastructure system for managing both lexical and semantic metadata about both virtual and physical entities. We describe version 0.8 of NPDS, including the separation of concerns between the original Problem-Oriented Registry of Tags And Labels (PORTAL) registries and the Domain Ontology Oriented Resource System (DOORS) directories, the combined registry and directory functionality of Nexus diristries, and the RESTful read-only web service API through which resource representation metadata records can be retrieved from these NPDS servers. We also introduce Scribe registrars with a corresponding RESTful read-write web service API for management of metadata records by both software agents accessing the web services directly and human users accessing them indirectly via web applications.

Keywords: Nexus-PORTAL-DOORS · diristry · registry · directory · registrar · HDMM · REST · API · web service · semantic · lexical

1 The Nexus-PORTAL-DOORS System

The Nexus-PORTAL-DOORS System (NPDS) offers a distributed and decentralized infrastructure system for metadata management by which individuals and organizations can maintain their own independent repositories of lexical and semantic metadata about resource entities in a problem domain of interest [12, 13]. These NPDS metadata repositories may also interact with other data and metadata repositories [11, 10]. NPDS comprises a comprehensive approach to building an infrastructure system with principles, strategies, an explicit data schema and messaging specification, and a distributed architectural style for network servers that exchange metadata records about online and offline resources. Since its original design [12] in 2006 and subsequent revision [13] in 2009, NPDS has been built with a 'meta-meta' foundational principle enabling 'metadata about metadata' and interoperability that references other repositories, vocabularies and ontologies.

NPDS comprises 3 types of servers: 1) PORTAL registries for registering resource entities with unique labels (URI or IRI identifiers) and lexical metadata that may include optional tags, controlled vocabulary term labels, and cross-references; 2) DOORS directories for publishing online and offline locations of the identified resource entities and semantic metadata that may include RDF descriptions referencing OWL ontologies; 3) Nexus diristries for combining the functionality of a PORTAL registry and DOORS directory in a single server called a diristry with the term coined by abridging DIRectory and regISTRY.

Analogous to the Internet Registry Information Service (IRIS) and Domain Name System (DNS) protocols, the NPDS approach implements the Hierarchically Distributed Mobile Metadata (HDMM) architectural style [13] to provide and redistribute metadata throughout the web. NPDS servers form a hierarchy consisting of authoritative servers that maintain master copies of records and distribute them to other non-authoritative servers including secondary and caching servers. So that the NPDS servers can communicate with each other and with client applications that retrieve records from them, any NPDS server should adhere to the common NPDS messaging specification and also expose a consistent RESTful API for read-only web service access to metadata records.

To permit diverse implementations of NPDS Scribe registrars that would support customized access, privacy and security, the NPDS specification does not mandate use of any particular required API for the Scribe registrars through which human users or software agents may register, revise and curate resource metadata records for the PORTAL registries, DOORS directories and Nexus diristries. However, we have created implementations of Scribe registrar services as examples that expose a RESTful API for a read-write web service. NPDS provides the original hybrid lexical and semantic approach for metadata and data integration across repositories as well as searching within problem-oriented, concept-constrained and domain-specific repositories. Other projects including Wikidata, Memex, *etc.* [9, 14, 4] have since adopted some of the NPDS data integration and cross-linking ideas that were first published in 2007 online [12].

2 NPDS Read-Only Web Service API

In order to make lexical and semantic metadata easily accessible to automated agents, the NPDS web service API should observe RESTful design principles [1]. The NPDS web service API operates read-only for registries, directories and diristries, responding to GET requests with the requested resource representation or set of representations and replying to all other HTTP methods with status 501 Not Implemented [5]. This new API simplifies implementation by decreasing both the number of path patterns and the chances of collisions with paths for other services or applications on the same host. Separating the read-write service for Scribe registrars from the read-only services for PORTAL registries, DOORS directories and Nexus diristries also provides additional measures of security and independence for write versus read in order to minimize the probability of causing errors in an infrastructure system intended by design to be distributed and decentralized, thus better allowing for independent implementations on diverse platforms with various operating systems, database servers and web servers.

In the service routes described below, we adopt “serverType” to indicate the type of server, either “nexus”, “portal”, or “doors”. At creation, the implementer must assign each server a unique identifier consisting of up to 128 alphanumeric characters, indicated by “serviceTag” below. In the current NPDS implementation, every resource representation metadata record must have at least one unique identifier consisting of up to 128 alphanumeric characters, indicated by “entityTag” below. Version 0.8 of the NPDS read-only web service consists of the following API endpoints:

`{serverType}/{serviceTag}/{entityType}/{infosetStatus}` retrieves descriptions of all entities of the specified type and status (“valid”, “invalid”, “pending” or “any”) from the specified service. Example: `http://npds.telegenetics.net/Nexus/GeneScene/person/any`

`{serverType}/{serviceTag}/{entityTag}` retrieves the description of the specified entity from the specified service. Example: `http://npds.brainhealthalliance.net/Nexus/BrainWatch/ABA`

`{serverType}/{serviceTag}?{queryString}` retrieves resource representations filtered according to the optional query string. Example: `http://npds.portaldoors.net/Nexus/DaVinci?nam=semantic`

The response body contains a document object, serialized in XML in the current implementation, which includes the server response and also the client request if the query string includes the echo flag switch turned on with “ef=1”. Other switches are described on the default help page available at the service root. The server response includes the status code and, on a successful request, an answer node which contains a node for each NPDS server type invoked, which in turn contains a list of resource representation nodes. This response structure keeps the NPDS record data separate from information about the process by which the response was derived, which will be useful for future iterations of NPDS that will respond to more complex queries by drawing on other distributed servers for additional resource representation records from other sites.

3 Scribe Read-Write Web Service API

The Scribe registrar read-write web service should also follow RESTful API design principles in addition to standard practices for HTTP requests and responses [1], [5]. In order to facilitate separation of concerns and decrease the likelihood of unintended write actions by requests intended for read-only interaction, calls to the Scribe read-write service must specify a server type distinct from those available via the NPDS read-only API. The Scribe read-write API is compatible with a wide variety of authentication strategies and allows an authorized client to create and add a new record to a specified service, and to update or delete an existing record. Requests follow the convention for HTTP methods: a GET request reads one or more records; a POST request creates a new record with the data in the request body; a PUT request updates a record by replacing it with the new version in the request body, and a DELETE request deletes a record. The current implementation supports approaches either for deriving the read-write routes from the read-only routes by prepending "scribe", or else simply using "scribe" as first segment with the entityTag as second segment for which an opaque randomized-character alias can also be used:

```
http://npds.brainhealthalliance.net/Scribe/Nexus/BrainWatch/
http://npds.brainhealthalliance.net/Scribe/Nexus/BrainWatch/ABA
http://npds.brainhealthalliance.net/Scribe/ABA
http://npds.brainhealthalliance.net/Scribe/C59DB9FFD13
```

It is important to maintain the Scribe read-write service API independent from the NPDS read-only service API. Doing so enables the service implementer to use any preferred means of creating, editing, and deleting records as determined by organizational considerations such as privacy, security, and customization.

4 Use Case Scenario: Automated Meta-Analysis

Meta-analysis is a valuable but difficult aspect of medical and scientific research wherein investigators analyze multiple reports of primary research results to assess the extent to which they collectively support or refute a given hypothesis [3]. Brain Health Alliance is currently working to build an application for automated meta-analysis on the foundation of the NPDS infrastructure. This application will consist of the following components: Curating web applications provide a human-friendly user interface for the Scribe registrars. Focused web crawlers retrieve information about resources relevant to a problem domain from databases, search engines, and other online resources [2] to populate NPDS repositories with relevant metadata records. Natural language processors translate natural language questions into SPARQL queries and output from statistical analysis packages into natural language answers [7]. Hypothesis-exploring ontologies facilitate more direct translation of questions from domain experts into SPARQL queries by providing a compact set of the most relevant concepts and relationships referencing more comprehensive foundational and domain ontologies to enable query expansion [10]. Inference engines expand queries and extract the relevant information from semantic descriptions of resources in PORTAL, DOORS or Nexus records [8]. Statistical analysis packages compute aggregate effect sizes and confidence intervals from the retrieved data [6].

5 Conclusion

Continuing the strategic approaches of previous versions of the Nexus-PORTAL-DOORS System (NPDS) [12, 13], version 0.8 of NPDS provides a way to bootstrap and bridge the developing semantic web with the existing lexical web. We have introduced here the separation of concerns between the read-only Nexus diristries and the new read-write Scribe registrars, as well as adoption of the serverType parameter as the first segment of the path in all routes for the NPDS web services. Maintaining the registrar behavior separate from the NPDS messaging exchange specification leaves implementing organizations greater freedom to develop customized read-write service APIs for the Scribe registrars. Using the server type as the first path segment helps to avoid collisions with URLs for other services and applications on the same host. As the NPDS specification matures and additional implementations become available, we believe that it will continue to play an important contributing role in developing a hybridized and bridged lexical and semantic web infrastructure.

References

1. Bojinov, V. (2015). *RESTful Web API Design with Node.js.* Packt Publishing Ltd.
2. Chakrabarti, S., Van den Berg, M., and Dom, B. (1999). Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(11):1623–1640.
3. Cooper, H., Hedges, L. V., and Valentine, J. C. (2009). *The handbook of research synthesis and meta-analysis.* Russell Sage Foundation.
4. De Sa, C., Ratner, A., Ré, C., Shin, J., Wang, F., Wu, S., and Zhang, C. (2016). Deepdive: Declarative knowledge base construction. *ACM SIGMOD Record*, 45(1):60–67.
5. Fielding, R. and Reschke, J. (2014). Hypertext transfer protocol (http/1.1): Semantics and content.
6. Grissom, R. J. and Kim, J. J. (2012). *Effect sizes for research: Univariate and multivariate applications.* Routledge.
7. Kaufmann, E. and Bernstein, A. (2007). How useful are natural language interfaces to the semantic web for casual end-users? In *The Semantic Web*, pages 281–294. Springer.
8. Mayfield, J. and Finin, T. (2003). Information retrieval on the semantic web: Integrating inference and retrieval. In *Proceedings of the SIGIR Workshop on the Semantic Web.*
9. Mitraka, E., Waagmeester, A., Burgstaller-Muehlbacher, S., Schriml, L. M., Su, A. I., and Good, B. M. (2015). Wikidata: A platform for data integration and dissemination for the life sciences and beyond. *bioRxiv*, page 031971.
10. Skarzynski, M., Craig, A., and Taswell, C. (2015). SOLOMON: An ontology for sensory-onset, language-onset and motor-onset dementias. In *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*, pages 969–972. IEEE.
11. Taswell, C., Franc, B., and Hawkins, R. (2006). The ManRay Project: Initial development of a web-enabled ontology for nuclear medicine. In *Proceedings of the 53rd Annual Meeting of the Society of Nuclear Medicine, San Diego, CA*, page 1431.
12. Taswell, C. (2008). DOORS to the semantic web and grid with a PORTAL for biomedical computing. *Information Technology in Biomedicine, IEEE Transactions on*, 12(2):191–204.
13. Taswell, C. (2010). A distributed infrastructure for metadata about metadata: The HDMM architectural style and PORTAL-DOORS system. *Future Internet*, 2(2):156.
14. Wilson, B., McGibbney, L., Mattmann, C., Ramirez, P., Joyce, M., and Whitehall, K. (2015). Mememxgate: Unearthing latent content features for improved search and relevancy ranking across scientific literature. In *AGU Fall Meeting Abstracts.*