2

Alternative Bootstrapping Design for the PORTAL-DOORS Cyberinfrastructure with Self-Referencing and Self-Describing Features

Carl Taswell
Global TeleGenetics, Inc.
USA

1. Introduction

A distributed registry-directory system as a cyberinfrastructure in support of navigation, search, and queries of the semantic web has been designed using a paradigm built in analogy with the corresponding systems established for the original web. The Problem Oriented Registry of Tags And Labels (PORTAL) and the Domain Ontology Oriented Resource System (DOORS) for the semantic web (Taswell, 2008a) are intended to function as interacting systems of registries and directories for the semantic web in a manner analogous to those used for the original web, namely, the Internet Registry Information Service (IRIS) and Domain Name System (DNS).

These interacting network systems of PORTAL registries and DOORS directories, collectively called the PORTAL-DOORS System (PDS), have been designed as resource metadata registering and publishing systems to address three major problems: cybersilos, transition barriers, and search engine consolidation. In a comprehensive literature review, Taswell (2008a) discussed the current cybersilo problem and barriers to the transition from original web to semantic web. More recently, Mowshowitz and Kumar (2009) provided commentary with growing concerns about search engine consolidation.

Guided by design principles intended to address all three major problems, the PDS is architected to operate as a hybrid between and bridge from original to semantic web by bootstrapping itself in a manner in which both the infrastructure system and its data are distributed physically and virtually in terms of both content and control of content. As a consequence, the distributed design itself prevents the possibility of search engine consolidation in a manner entirely analogous to the success of IRIS-DNS in preventing the consolidation of internet domain name registries or directories.

Beyond the original published design (Taswell, 2008a) that serves as the abstract architectural blueprint for PDS, some concrete interface schemas with basic ontologies have been drafted for prototype registries in fields relevant to biomedical computing and

radiological informatics. These draft prototypes include a formal semantic definition of pharmacogenomic molecular imaging which provides a use case that demonstrates search across multiple specialty domains (Taswell, 2008b).

However, such XML-based models represent only a piece of the puzzle. A full implementation requires many other software components especially back-end database servers and front-end clients for the PORTAL registries and DOORS directories. In order to gain practical experience testing development of alternative implementations, it is necessary to begin working with real data stored in actual database servers. Therefore, the roadmap for PDS development shifted from revision of XML schemas and OWL ontologies to construction of a prototype relational database model.

This database model has been purposefully chosen to be initially a traditional relational model rather than an RDF-based triple store. This decision was made not only because of the guiding principle that PDS must be capable of operating as a hybrid and a bridge but also because of the pervasive availability of relational databases in comparison with newer kinds of databases. Recent research (Zhuge et al., 2008) on the use of relational database models for semantic systems also suggests that relational databases may continue to play an important role rather than being completely displaced by RDF-based triple stores for semantic systems.

This report describes the relational database models now implemented for revisions of the PDS design including both the original design with PORTAL and DOORS servers and a new bootstrapping design with NEXUS servers. This new design more explicitly realizes the guiding principle for PDS that it should operate in a bootstrapping manner.

2. Methods

Altova XMLSpy (www.altova.com) and Microsoft Visual Studio 2008 with SQL Server 2008 (www.microsoft.com) were used as the integrated development environments for software development experiments with various partial implementations of the PDS designs. An iterative process of software development, debugging, testing, and analysis for re-design beginning from both the XML perspective and the SQL perspective resulted in SQL, XML, and ASP.net code for both the original separate PORTAL-DOORS design as well as a new alternative combined NEXUS design with distinct advantages. Analysis for re-design of the entire system also addressed the following concerns: (i) eliminating redundancies, (ii) improving the separation of functionalities between the various servers and services, and (iii) resolving any other issues that may arise.

3. Analysis

All essential design concepts initially proposed (Taswell, 2008a) have been successfully retained in the software implementations. However, on analysis for re-design, certain redundancies were noted that precluded an improved separation of functionality between PORTAL registries and DOORS directories. For example, in the architectural blueprint for PDS (Taswell, 2008a), *resource tags* were declared as permitted metadata maintained for a

resource at both PORTAL registries and DOORS directories. Redundancies of this kind complicate maintenance in general as well as the intended use of the PORTAL and DOORS networks primarily as lexical and semantic systems, respectively.

For the purposes of use in PDS, lexical and semantic systems are defined here as follows: A lexical system (aka "dumb" system) is an information system in which words are processed as character strings that have no inherent meaning to the processing agent, and more specifically, character strings are processed without use of RDF, OWL, and related technologies. A semantic system (aka "smart" system) is one in which words have defined meaning to the agent processing them with logic-based reasoners, and more specifically, character strings are processed with use of RDF, OWL, and related technologies. Thus, PORTAL registries, as primarily a lexical system should register the *resource labels* and *resource tags*, while DOORS directories, as primarily a semantic system should publish the *resource locations* and *resource descriptions*. This re-design eliminates the unnecessary redundancies and complications of maintaining *resource tags* at both PORTAL registries and DOORS directories.

Moreover, on analysis for re-design, a circular reference was noted that required resolution for implementation. According to the original blueprint (Taswell, 2008a), PORTAL registries were designed to restrict registration of resource metadata at each domain-specific registry to those resources meeting the criteria required for the problem-oriented domain declared for that particular registry. For example, a person or organization interested in building and maintaining a problem-oriented registry for zoology (say "ZooPORT") most likely would not permit registration of resources related to stars unless the star is an animal such as a starfish. And vice versa, managers of a problem-oriented registry for astronomy (say "AstroPORT") most likely would not permit registration of resources related to animals unless the animal is the name of a star or constellation of which there are many such as Leo (Lion), Lepus (Hare) or Lupus (Wolf).

At the same time, DOORS directories were designed to publish the resource descriptions providing the RDF triples and thus the semantic information necessary to determine eligibility of the resource for registration in the particular PORTAL registry. However, directories in the DOORS server network were intended for use not only in a manner that would serve any PORTAL registry (whether AstroPORT, ZooPORT, or any of the four existing prototype registries BioPORT, GeneScene, ManRay, and BrainWatch) but also in a manner that would publish the resource descriptions with the RDF triples about the resource. This important semantic information necessarily determines eligibility of the resource for registration in the relevant governing PORTAL registry if that registry has elected to impose restrictions for definition of the problem-oriented domain. This situation constitutes a contradictory circular reference, because according to the original blueprint (Taswell, 2008a), a resource must be registered first at a PORTAL registry before it can be described at a DOORS directory, whereas in this scenario, it must be described before it can be registered. Both temporal sequences are not simultaneously possible.

Various solutions for implementations that resolve this circular reference problem include the following: (A) Splitting the resource description into a PORTAL required portion and a

DOORS permitted portion; (B) Using record status codes “Invalid”, “Pending”, and “Valid” exchanged between PORTAL and DOORS; (C) Using PORTAL resource tags instead of DOORS resource descriptions to determine eligibility; and (D) Building an alternative design that combines both PORTAL and DOORS services into a single NEXUS service. Although solution (A) would resolve the circular reference problem, it would also preclude implementation of PORTAL and DOORS systems as primarily lexical and semantic, respectively, because it would require semantic processing on PORTAL in addition to that on DOORS. However, solution (D) would not only resolve the circular reference problem, but would also enable a multiplicity of configurations according to varying composition of service operations in different components that can co-exist with each other on the same or different servers.

Finally, throughout the iterative development, analysis, and re-design process, it became apparent that the terminology used by the original blueprint (Taswell, 2008a) for PDS could be improved for better clarity. In particular, the original terminology did not adequately distinguish between the metadata about the *resource entity*, and the metadata about the *resource record*. As a consequence, PDS terminology has been revised and used throughout code development to represent this distinction between the primary metadata about the *resource entity* and the secondary metadata about the *resource record*.

Here the primary metadata is metadata about the thing of interest (the entity) whereas the secondary metadata is the metadata about the metadata (the record). Moreover, the term *resource representation* refers to all of the metadata, both primary and secondary, when considered together collectively. In general, however, any use of the term *resource* by itself should be construed (when not otherwise apparent from context) as referring to the *entity* rather than the *record* or *representation* so that the original definition remains valid: *a resource may be any entity whether abstract or concrete, whether online in the virtual world or offline in the physical world*.

4. Results

Software has been developed for the PORTAL-DOORS System that eliminates the redundancies, clarifies the terminology, and resolves the circular reference problem of the original blueprint (Taswell, 2008a). To implement the necessary revision of the original design, both solutions (B) and (C) were chosen. In addition, solution (D) has also been implemented for the alternative new design. This new scheme called the *combined design* can coexist together with the original scheme called the *separate design*. Thus, any node in the PDS network can be built as a separate PORTAL node, separate DOORS node, or a combined PORTAL-DOORS node also called a NEXUS node (see Fig. 1). The new combined design offers significant advantages in enabling an efficient self-referencing, self-describing, and bootstrapping process amongst the core system constituents (agents, registrants) and components (registrars, registries, and directories).

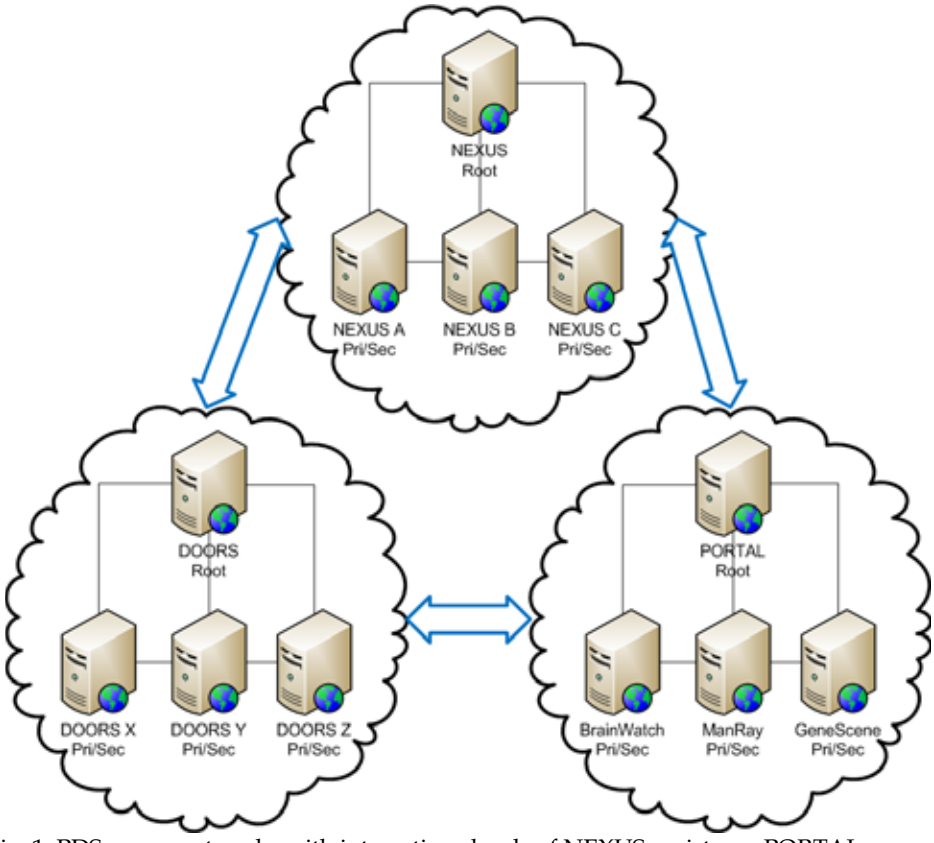

Fig. 1. PDS server networks with interacting clouds of NEXUS registrars, PORTAL registries, and DOORS directories. NEXUS servers may expose either the NEXUS registrar service for the separate design or the integrated set of NEXUS registrar, PORTAL registry, and DOORS directory services for the combined design.

Figure 2 displays a diagram summarizing the basic structure of data records with both *required* and *permitted* fields at both PORTAL registries and DOORS directories. When providing registrar services for separate PORTAL and DOORS nodes, NEXUS registrars operate in a manner consistent with the original separate design. However, when providing registrar services for a combined PORTAL-DOORS node, NEXUS registrars can also operate in a manner that enables integrated storage of both PORTAL and DOORS record data on the same server. Figure 3 displays a diagram depicting the relational database model for the current 0.5 draft version of the PDS schemas available at www.portaldoors.org. This data structure model shows the primary and foreign keys that provide referential integrity constraints for the relational database tables of a NEXUS server node in the network system.

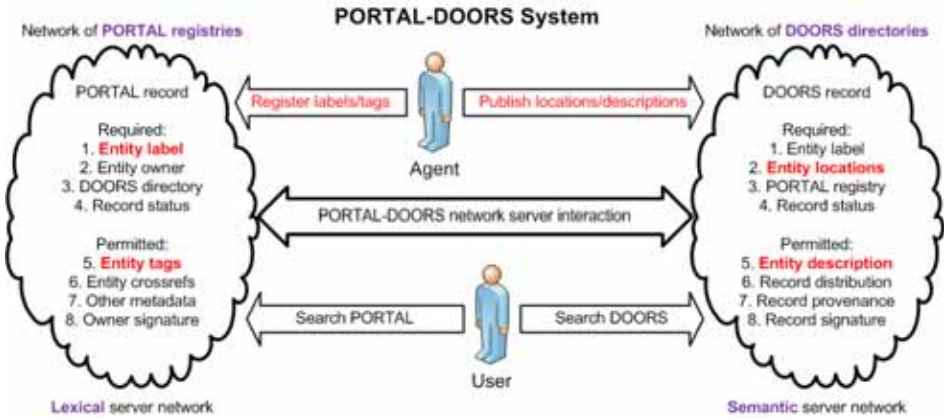

Fig. 2. Resource metadata registered and published by agents for search by users in the PDS networks. Fields within data records are considered *required* or *permitted* with respect to the schemas maintained by the root servers (see Fig. 1).

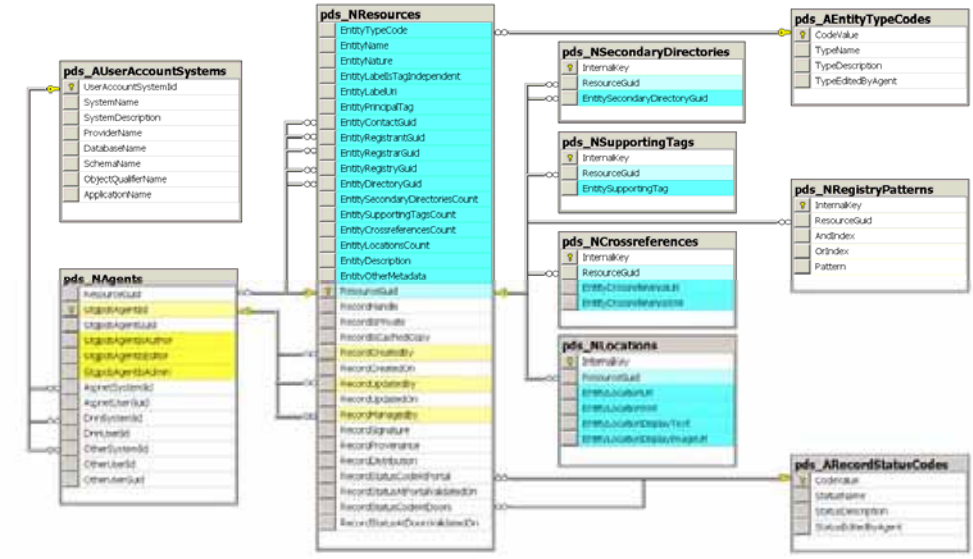

Fig. 3. Relational database model for NEXUS combined design server with integrated storage of both PORTAL and DOORS data record fields.

All PDS tables in the database are named with the prefix `pds_` to distinguish them from the tables of administrative providers such as Microsoft's ASP.Net authentication and authorization services and their database tables named with the prefix `aspnet_`. The table `pds_NResources` serves as the main NEXUS resources table with primary key

ResourceGuid for the related records connected via foreign keys ResourceGuid in each of the dependent tables pds_NSupportingTags, pds_NCrossReferences, pds_NSecondaryDirectories, and pds_NLocations. With the column ordering for the main table pds_NResources as displayed in Figure 3, note that the fields displayed above the primary key are *resource entity* metadata fields whereas those displayed below the primary key are *resource record* metadata fields. Because of the different ways that the metadata can be used, providing distinct keys for the different subsets of metadata offers greater convenience for various programming and interface contexts. Thus, RecordHandle, EntityLabelUri, and ResourceGuid serve as keys for the *resource entity*, *resource record*, and entire *resource representation*, respectively.

For the *resource entity* metadata within the main table pds_NResources, there are five directly self-referencing relations from fields with the suffix _Guid to five other resources for the EntityContact_, EntityRegistrant_, EntityRegistrar_, EntityRegistry_, and EntityDirectory_. There is no requirement that any of the necessary metadata for each of these five resources be stored at the same NEXUS server node. However, if so, then it can be referenced via the _Guid, and if not, then it can be referenced via the analogous _LabelUri fields (not shown in Figure 3). For example, the resource for the EntityContact_ can be referenced internally via EntityContactGuid or externally via EntityContactLabelUri. Check constraints can be used to prevent both the _Guid and the _LabelUri for the EntityContact_ from being simultaneously non-null. Alternatively, appropriate programming logic can be used to maintain precedence of the internal reference via the _Guid over the external reference via the _LabelUri, or vice versa, depending on the non-null values of these fields in the context of the status of the boolean field RecordIsCachedCopy.

For the *resource record* metadata within the main table pds_NResources, there are three indirectly self-referencing relations from fields with the suffix _By to three other potential resources for the RecordCreatedBy, RecordUpdatedBy, and RecordManagedBy agents. The indirect self-referencing via the auxiliary linking table pds_NAgents provides a simple permission management system implemented with the feature of sufficient flexibility to interface with various user account provider systems, and simultaneously, to render optional the publication of any information pertaining to agents as resources distinct from the contacts and registrants.

Thus, the linking table pds_NAgents mediates between the set of tables for PDS and another set of tables for the authentication and authorization system for managing agent access to inserting updating and deleting records in the NEXUS tables. The linking table has a primary key GtgpdsAgentIid and various alternative optional fields available for linking to user membership providers such as the field AspnetUserGuid for linking to Microsoft's ASP.Net membership provider, OtherUserGuid for linking to an alternate generic user membership provider, etc. In addition, the table pds_NAgents provides the foreign key ResourceGuid for linking back to a resource in the main table pds_NResources for use in a scenario where the persons with responsibility for managing resources in the database are themselves identified and described in the main table.

Regardless of whether an agent is published as a resource, and regardless of whether a resource is an agent, contact or registrant of type person or of any other type, any resource may be flagged as non-publishable by the boolean field `RecordIsPrivate` in the table `pds_NResources`. Also, regardless of code implementation with persistence of the value stored in the field `ResourceEntityLabelUri` or otherwise computed dynamically by concatenation of the `ResourceEntityPrincipalTag` with the label of the entity's registry, it should be emphasized that any PDS implementation must maintain the important requirement of uniquely identifying resources by the *resource entity label* which must be an IRI or URI. For PDS draft version 0.5, both SQL code for the relational database model and XML Schemas for data structure interfaces are available for download from www.portaldoors.org with an operational web site implemented at www.telegenetics.net now available for registration of resources relevant to the problem-oriented domains of the GeneScene, ManRay, BioPort, and BrainWatch registries.

5. Discussion

As a cyberinfrastructure, PDS can be considered an *information-seeking support system* (Marchionini and White, 2009). With an appropriately enhanced user interface, PDS can be considered a *facetted search tool* (Schraefel, 2009). Regardless of use as infrastructure system or application tool, PDS interlinks registries, directories, databases, and knowledgebases across domain-specific fields, disciplines, and specialties. It assures globally unique identification of resources while promoting interoperability and enabling cross registry and cross directory searches between different problem-oriented, not technology-restricted, domains because of the fundamental definition of a resource as any entity, abstract or concrete, online or offline.

PDS has been designed as a hybrid bootstrap and bridge to transition from the old lexical web to the new semantic web, and allows for all constructs from free tagging and folksonomies to microformats and ontologies. It supports mass participation and collaboration via its hierarchical and distributed but decentralized and localizable infrastructure, and as a consequence, provides a democratized solution to the problem of search engine consolidation. Mowshowitz and Kumar (2009) discuss both the realities and the risks of search engines that effectively restrict access to information, and argue that this problem represents a serious concern.

With its infrastructure designed in a distributed manner that permits localized control of policies and content and thereby prevents the possibility of search engine consolidation, the PORTAL-DOORS model is most similar in conceptual paradigm to the IRIS-DNS model that inspired it. In contrast, it can be compared to other familiar models for information management systems exemplified by the Google search engine (www.google.com) or the Wikipedia encyclopedia (www.wikipedia.org). In the case of Google, the server infrastructure is distributed (to some degree) but not the control of content (unless "paid placement" is considered). In the case of Wikipedia, the servers and content are centralized but the control of content is shared by all contributing anonymous authors and editors. However, in the case of PORTAL-DOORS, the server infrastructure, the content control, and the content itself are all shared and distributed. Moreover, the design of the PORTAL-

DOORS framework remains analogous to that of the IRIS-DNS framework with mechanisms that enable data records to be distributed and mobile with request forwarding and response caching.

Continuing progress on the development of PDS with its NEXUS registrars, PORTAL registries, and DOORS directories will focus on implementing all features of the design including both data structures and operational methods for both independent and interacting servers. Content for PORTAL-DOORS will be contributed manually by human agents as has been done for IRIS-DNS. Later, when software agents, webbots, and converters become available, content will be generated automatically or semi-automatically. For manually contributed content compared with automatically generated content, there may be a trade-off in the quality of content produced versus the rate of content production. This trade-off would not be applicable to those situations where existing databases only need an appropriate interface for inbound queries and wrappers for outbound responses.

6. Conclusion

A new bootstrapping combined design for PDS, together with the original separate design for PDS, has been implemented for NEXUS registrars, PORTAL registries, and DOORS directories and demonstrated with the problem-oriented domains declared for the GeneScene, ManRay, BioPORT, and BrainWatch prototype registries. The combined design has many important advantages during early stages of PDS adoption and use. However, the separate design will become useful when concerns about performance, efficiency, and scalability become more significant.

7. References

Marchionini, G. & White, R.W. (2009). Information-Seeking Support Systems (Guest Editors' Introduction to Beyond Search). *IEEE Computer*, Vol. 42, No. 3, pp. 30-32 (DOI 10.1109/MC.2009.88).

Mowshowitz, A. & Kumar, N. (2009). And Then There Were Three. *IEEE Computer*, Vol. 42, No. 2, pp. 108-107 (DOI 10.1109/MC.2009.36).

Schraefel, M.C. (2009). Building Knowledge: What's beyond Keyword Search? *IEEE Computer*, Vol. 42, No. 3, pp. 52-59 (DOI 10.1109/MC.2009.69).

Taswell, C. (2008a). DOORS to the semantic web and grid with a PORTAL for biomedical computing. IEEE *Transactions on Information Technology in Biomedicine*, Vol. 12, No. 2, pp. 191-204 in Special Section on Bio-Grid (DOI 10.1109/TITB.2008.905861).

Taswell, C. (2008b). PORTAL-DOORS Infrastructure System for Translational Biomedical Informatics on the Semantic Web and Grid. *Proceedings of American Medical Informatics Association Summit on Translational Bioinformatics*, poster 43, San Francisco, March 2008.

Zhuge, H.; Xing, Y. & Shi, P. (2008). Resource space model, OWL and database: Mapping and integration. *ACM Transactions on Internet Technology*, Vol. 8, No. 4, pp. 1-31 (DOI 10.1145/ 1391949.1391954).