Producing and Using Linked Open Government Data in the TWC LOGD Portal

Timothy Lebo, John S. Erickson, Li Ding, Alvaro Graves, Gregory Todd Williams, Dominic DiFranzo, Xian Li, James Michaelis, Jin Guang Zheng, Johanna Flores, Zhenning Shangguan, Deborah L. McGuinness and Jim Hendler

Abstract As open government initiatives around the world publish an increasing number of raw datasets, citizens and communities face daunting challenges when organizing, understanding, and associating disparate data related to their interests. Immediate and incremental solutions are needed to integrate, collaboratively manipulate, and transparently consume large-scale distributed data. The Tetherless World Constellation (TWC) at Rensselaer Polytechnic Institute (RPI) has developed the TWC LOGD Portal based on Semantic Web principles to support the deployment of Linked Open Government Data. The portal is not only an open source infrastructure supporting Linked Open Government Data production and consumption, but also serves to educate the developers, data curators, managers, and end users that form the growing international open government community. This chapter introduces the informatic challenges faced while developing the portal over the past two years, describes the current design solutions employed by the portal's LOGD production infrastructure, and concludes with lessons learned and future work.

1 Introduction

In recent years the release of Open Government Data (OGD) has become more common and has emerged as a vital communications channel between governments and their citizens. Since 2009, governments around the world[1] including the United States, United Kingdom, Australia, Norway, and Greece have built Web portals to provide datasets to their citizens and worldwide consumers alike. These datasets provide a wide range of information significant to the daily lives of citizens such as locations of toxic waste dumps, regional health-care costs, and local government spending. Citizens have become consumers of OGD: a study conducted by

Tetherless World Constellation, Rensselaer Polytechnic Institute, 110 8th St., Troy, NY 12180, USA e-mail: erickj4@rpi.edu

[1] http://www.data.gov/community/

the Pew Internet and American Life Project reported that 40% of adults went online in 2009 to access some kind of government data[18]. One direct benefit of OGD is richer governmental transparency: citizens may now access the data sources behind previously-opaque government applications, enabling them to perform their own analyses and investigations not supported by existing tools. Moreover, instead of merely being *read-only* end users, citizens may now participate in collaborative government information access, including *mashing up* distributed government data from different agencies, discovering novel facts and rules, developing customized applications, and providing government agencies specific feedback about how to more effectively perform their governmental role.

Several technological challenges requiring significant time and effort must be overcome to fully realize this potential. Although many open government datasets are available for public access, most have been published using formats that do not permit distributed linking and do not help consumers understand their content. As stewards of a vast and diverse collection of official government data, the cost of individual agencies deploying Linked Open Government Data by themselves is prohibitive. Due to interoperability, scalability and usability constraints, "raw" OGD datasets are typically released *as is*; many datasets are encoded in formats not conducive to automated machine processing, and datasets from different sources are encoded using heterogeneous structures with ambiguous or differing meanings.

Since substantial human effort is needed to make raw datasets comprehensible, only a small proportion of the government data available has been published in an easily-reusable form using open principles. To accelerate the progress of opening more government data, new approaches are required to produce Linked Open Government Data as quickly as possible while allowing for incremental improvements developed by a broad community with diverse knowledge, skills, and objectives.

Instead of employing Linked Data principles, OGD release efforts such as Sunlight Foundation's National Data Catalog, Socrata, and Microsoft's OData use Web-friendly RESTful APIs to address these infrastructural challenges. However, data APIs provide only a partial solution. By their nature, APIs abstract away details and thus restrict access to the underlining data; there is typically no way for consumers to inspect, reuse, or extend the data model behind an API. This poses problems for application developers and end users because the data itself cannot be easily shared and reused, causing each API to act as an isolated silo of data that requires effort from each application developer to connect.

A global community of developers is applying Semantic Web technologies and Linked Data principles to overcome data integration challenges and take full advantage of OGD [1, 2]. The emerging Linked Open Data methodology[2] enables full data access using Web standards. Publishers can release raw data dumps instead of devoting time to design special-purpose data access APIs that make assumptions about consumer needs. Instead, consumers can integrate distributed government data in Linked Data form without advance coordination with publishers, allowing others to benefit without waiting for each agency to adopt linked data design principles.

[2] http://linkeddata.org/

Since only a few government agencies have released their data in RDF formats, they need tools, infrastructure, and guidance to impart a wide variety of data with appropriate structure for human and machine consumption and to make data elements linkable. The TWC LOGD Portal has been designed and deployed to serve as a resource for the global LOGD community and has helped make the LOGD vision real. It stands as a reference model for the practical application of using Linked Data techniques to integrate disparate and heterogeneous government data.

This chapter describes our approach to fulfilling these needs and is organized as follows. Section 2 provides an overview of the TWC LOGD Portal, which provided motivation, context, and design constraints for the production workflow described in Section 3. Section 4 discusses the challenges faced when republishing third party data and the approaches taken to increase transparency of the production workflow. Section 5 discusses aspects of deploying LOGD on the portal and its use to create mashups. Section 6 reviews work related to producing and consuming government data with and without Linked Data principles. Section 7 concludes with a summary of our research, deployment contributions, and an outline of future directions.

2 The TWC LOGD Portal

The LOGD production workflow described in this chapter was developed to support the TWC LOGD Portal[3], described more fully in [4]. To serve the growing international open government community, the portal was created to meet three challenges:

- *LOGD Production*: Because many OGD datasets are released by different agencies using various formats and vocabulary, developers spend a lot of effort cleaning, restructuring, and linking related OGD datasets before they can develop applications. To reduce these initial costs, we created a persistent and incremental LOGD production infrastructure to incorporate and reuse individual efforts.
- *LOGD Consumption*: Using LOGD as a basis, developers can quickly develop and replicate government data mashup applications on the Web. To illustrate the benefits of LOGD in government applications, our team has developed more than fifty demonstrations using a wide range of readily-available Web technologies.
- *LOGD Community*: LOGD stakeholders need community support to collaborate and share best practices. To this end, the TWC LOGD Portal implements social semantic Web and provenance technologies to inter-link demos and tutorials that demonstrate best LOGD practices. Supporting open source principles is essential in developing the LOGD community, so the portal uses third-party open source code including Virtuoso and Drupal6; hosts `csv2rdf4lod`[4] on GitHub; hosts all converted data, conversion configurations, and metadata on the Web; and hosts demo code and SPARQL queries on a Google Code project[5].

[3] http://logd.tw.rpi.edu

[4] http://purl.org/twc/id/software/csv2rdf4lod

[5] http://code.google.com/p/data-gov-wiki/

3 Producing Linked Open Government Data

The LOGD production workflow is the centerpiece of the TWC LOGD Portal. In this section, we introduce six stages of dataset integration. Five of these stages are designed to minimize human effort for incorporating a new dataset as Linked Data, while the remaining stage enables data modelers to add well-structured and well-connected descriptions to the initial representation. We describe enhancement types that a data modeler is most likely to use[6], along with a selection of more advanced enhancement types that elucidate the diversity of structural schemes employed by tabular government datasets. Throughout this section, we use portions of the White House Visitor Access Records[7] as a running example.

We describe an extension of the VoID[8] Dataset class to establish a three level dataset hierarchy that accounts for the RDF data resulting from incremental activities when accumulating new datasets, enhancing existing datasets, and handling new releases of those datasets already accumulated. Further, we highlight the correspondence between a dataset's URI and its role within the three-level VoID hierarchy. We then describe how this same correspondence is reused in our design to populate a SPARQL endpoint's named graphs.

After applying the five stages to create initial Linked Data from an OGD dataset and taking advantage of a sixth stage to enhance its representation, we describe how to handle an inevitable situation: a source organization releases a new version of a dataset we have already incorporated, published – and are using in applications. We use this situation to highlight several data organization challenges and how we solve them using a three-level namespace decomposition that simultaneously supports naming entities within and across datasets, establishing vocabularies that apply at different breadths, and performing bottom-up incremental integration of diverse datasets within and across source organizations – and among the Web of Data.

3.1 Producing Initial Conversions with Minimal Human Effort

As illustrated in Figure 1, data integration is achieved by iteratively following a few stages for each dataset of interest. These stages are designed to minimize initial human effort so that all data is available as Linked Data as quickly as possible, yet focused human efforts to understand, use, and enrich particular portions of the data are accumulated for sharing among the rest of the community. The six major stages are *Name*, *Retrieve*, *Adjust*, *Convert*, *Enhance*, and *Publish*. Two of these stages are optional and can be omitted or postponed in situations where the source data is already in an amenable format (*Adjust*) and/or consumers do not yet have a compelling use case to warrant enhancement (*Enhance*).

---

[6] Based on our experience with curating hundreds of datasets during the past two years.

[7] http://purl.org/twc/pages/whitehouse-visitor-access-records

[8] Vocabulary of Interlinked Datasets is described at http://www.w3.org/TR/void/

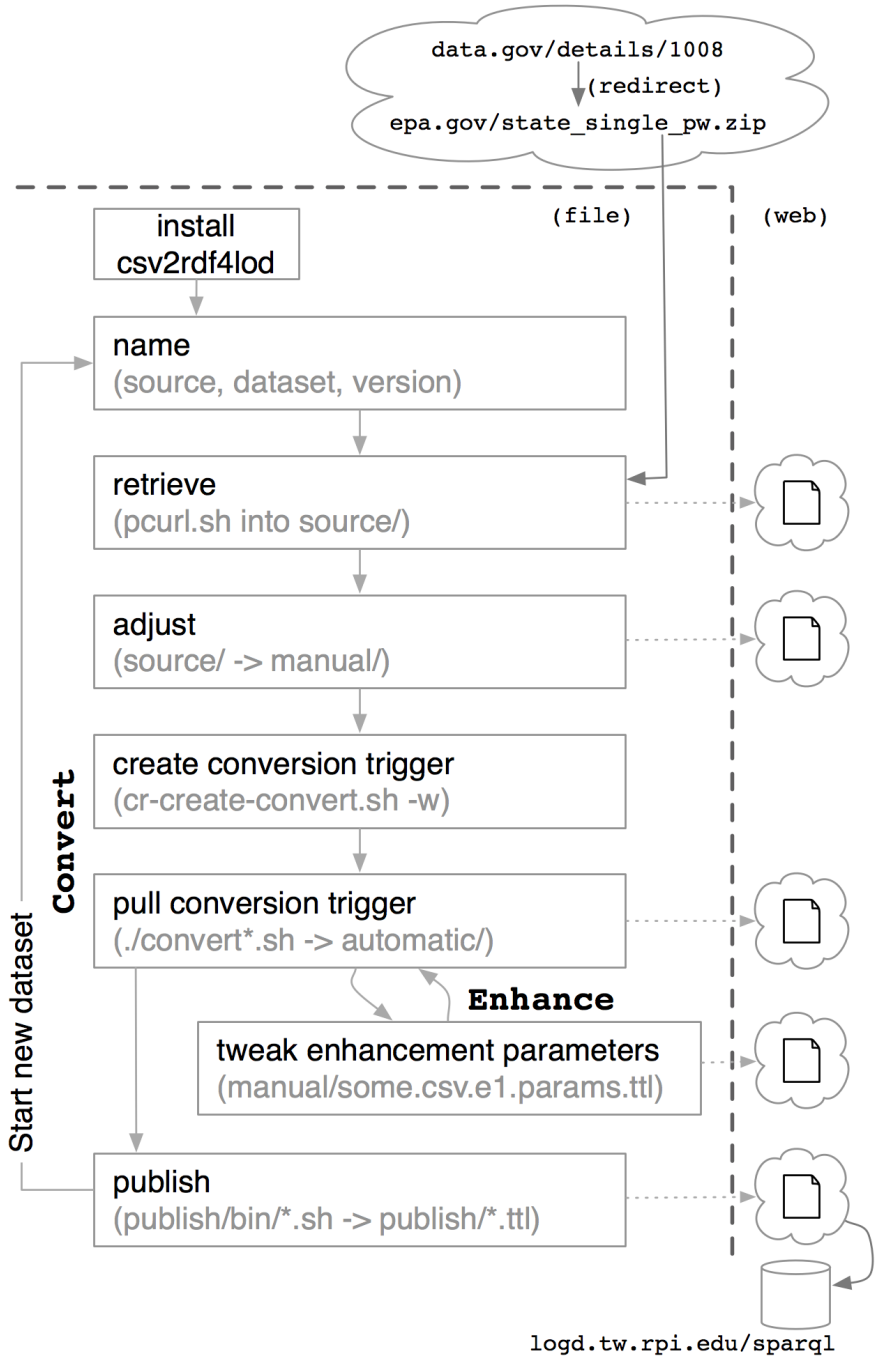

Fig. 1 The major stages of the LOGD production workflow are performed on the server, while the final and intermediate results are made available on the Web as dump files and through a SPARQL endpoint. Entities described in the SPARQL endpoint are available as resolvable Linked Data. Associations among the retrieved files and conversion results are encoded in RDF by provenance-aware utilities. Five of the six production stages require minimal human effort; the sixth enhancement stage can be perform as needed without disrupting applications built against earlier results.

Name: To name a dataset, three identifiers are assigned; the first identifies the **source** organization providing the data, the second identifies the particular **dataset** that the organization is providing, and the third identifies the **version** (or release) of the dataset. For example, *whitehouse-gov*, *visitor-records*, and *0510* identify the data that was available from whitehouse.gov[9] on July 8th, 2010. These identifiers are used to construct the URI for the dataset itself[10], which in turn serves as a namespace for the entities that the dataset mentions. These three identifiers should be assigned thoughtfully and consistently, as they provide the basis for the entire naming convention and may be used by third party consumers to orient with and navigate among the resulting data. Decisions for these three identifiers should be guided by reusing the source organization's terminology. To identify the source organization, we recommend reusing a form of their Web domain name, such as *london-gov-uk* or *ncdc-noaa-gov*. To identify the dataset, we recommend reusing a title[11] that the source organization provides or would recognize and associate to their collection.

Retrieve: After naming a dataset, its associated data files and documentation are retrieved. Such a retrieval creates a snapshot of the dataset available, while subsequent retrievals of the same dataset will create potentially different snapshots (since the source may remove, replace, or augment previous offerings). The assigned version identifier distinguishes the data from each snapshot. When possible, we recommend reusing version identifiers provided by the source organization (such as *release-23* for USDA's nutrition dataset[12]), but we have found that these are rarely provided by data publishers[13]. In the absence of a more suitable identifier, we recommend assigning a version identifier according to the publish, last-modified, or retrieval dates[14] in the form *2011-Mar-17*[15]. A provenance-enabled URL fetch utility is used to retrieve URLs, which stores a PML file[12] in the same directory as the retrieved file and records provenance including the user account initiating retrieval, the government URL requested, time requested, and checksum of the file received.

Adjust: Although manually modifying files retrieved from authoritative sources should be avoided, unrepeatable human intervention may be necessary to accommodate the format required by the conversion process. Manual adjustments are minimized by specifying appropriate conversion parameters (discussed in Section 3.2), but process transparency is maintained by storing results separately from their originals and recording the provenance associating the adjusted files to their predecessors, indicating the type of process applied, and citing the user account reporting the modifications.

---

[9] http://www.whitehouse.gov/files/disclosures/visitors/WhiteHouse-WAVES-Released-0510.csv

[10] http://logd.tw.rpi.edu/source/whitehouse-gov/dataset/visitor-records/version/0510

[11] If acronyms are expanded, titles are more informative to a broader audience.

[12] http://www.ars.usda.gov/services/docs.htm?docid=8964

[13] A version identifier is gleaned from the White House by inspecting part of its data file URLs.

[14] These three types of dates are listed in order of preference because, for example, the publish date more closely identifies the source organization's dataset than the date one happened to retrieve it.

[15] This date format was chosen to facilitate human readability and to follow the hierarchical nature of the URI; date information that one would want to query should be – and is – encoded in RDF.

Convert: `csv2rdf4lod`[16] converts tabular data files to RDF according to interpretation parameters encoded using a conversion vocabulary[17]. The use of parameters instead of custom code to perform conversions enables repeatable, easily inspectable, and queryable transformations; provides more consistent results; and is an excellent source of metadata. XML-based data files can be converted to RDF using parameter-driven utilities such as Krextor[9]. Output from `csv2rdf4lod` includes provenance for the RDF dump files it produces by citing its invocation time, the converter version and hash, the input file, the transformation parameters used, the parameter authors, and the user account invoking the conversion. Each conversion that uses different transformation parameters is named with a **layer identifier**[18] to distinguish it from other interpretations of the same input files retrieved. An initial *raw layer* is produced with minimal effort by providing only the three identifiers already assigned (source, dataset, and version). Although easy to create, the *raw layer* is the result of a *naive* interpretation; rows become subjects, column headers become predicates, and cells assert a single triple with an untyped string literal. Enhancing *raw* to make more meaningful RDF is highly encouraged.

Enhance: Because the enhancement stage is the only one of six that requires significant human effort, it is described more completely in Section 3.2. Enhancement can be performed after (or instead of) the initial conversion and it can be published well after (or instead of) publishing the initial conversion. In either case, the well-structured, well-connected addition will augment what has already been published and will not interfere with applications built against the initial conversion. Further, applications are able to discover the new enhancements of previous layers, may automatically adjust to use it, and can fall back if the enhancements are "too" different.

Publish: Publication begins with the conversion results and can include making dump files available for download, loading results into a triple store, and hosting resolvable URIs so they are available to Linked Data utilities. Different portions of the full RDF dataset are aggregated into separate dump files to allow consumers to retrieve specific portions that fulfill their needs. A complete dump file is hosted on the Web and includes the raw layer, any enhancement layers, *owl:sameAs* triples, retrieval provenance, conversion provenance, and metadata describing its VoID hierarchy. The SPARQL endpoint is loaded by retrieving the RDF dump files that the production workflow previously hosted on the Web. This transparency allows anyone else to reproduce the state of the triple store for their own purposes. Provenance of loading a named graph with an RDF file from the Web is stored in the named graph itself, enabling users to trace data from what they are querying, through the conversion process and retrieval, and to the original government data. The data files originally retrieved from the government along with any other intermediate files are also hosted on the Web and associated by RDF assertions, enabling consumers to inspect and repeat the integration processes responsible for the data offered.

---

[16] `csv2rdf4lod`'s URI is http://purl.org/twc/id/software/csv2rdf4lod

[17] The conversion vocabulary namespace is http://purl.org/twc/vocab/conversion/

[18] Because their meanings are difficult to name concisely and uniformly, enhancement layers are distinguished using incrementing counting numbers to provide a simple temporal ordering.

3.2 Enhancing an Initial Conversion by Creating a Second Layer

Although the rapid production of large quantities of Linked Open Government Data may be useful, such data will be much more valuable if attention has been given to correctly model dataset content by reusing existing vocabularies, referencing entities commonly recognized from other data sources, and structuring in more natural (and less record-like) representations. The five stages described in the previous section quickly produce an initial conversion that permits exploration using standard Linked Data utilities. These stages provide a concrete basis for learning the content, discussing with experts, determining more "appropriate" RDF representations of the domain, and developing prototypes.

As described earlier, the initial conversion is *naive*; rows become subjects, columns become predicates, and cells assert a single triple with an untyped string literal. The initial interpretation of the tabular structure creates the first *layer* of descriptions for entities in a versioned dataset. For example, two triples in an initial layer (*:thing_2 raw:namefirst "CHRISTINE"; raw:access_type "VA"*) are illustrated in Figure 2. Enhancing the same versioned dataset creates a second *layer* that adds more descriptions for the same entities[19] (*:visitor_2 foaf:firstName "CHRISTINE"; e1:access_type a:Visitor_Access*). A *layer* can be defined as the set of triples whose predicates fall within a given namespace. For example, one could specify a FOAF layer of the Billion Triple Challenge datasets or a Dublin Core layer of the latest DBPedia dataset. Since each conversion provides a new interpretation of the original data, predicates from each conversion need to be in distinct layers. For example, the enhanced layer in Figure 2 changed *access_type*'s range from a Literal to a Resource; using the same predicate across layers would cause logical issues when applying OWL reasoning, or would break applications expecting literal values when the enhancement is added. To avoid these issues, predicates are named within a namespace specific to the layer.

`csv2rdf4lod` converts tabular data files to RDF according to interpretation parameters encoded using a conversion vocabulary. The following sections highlight the enhancements[20] most frequently used, where the prefix `c:` abbreviates the conversion vocabulary namespace[21]. Similarities between the enhancement names and the axioms in RDFS and OWL were a design goal, since the enhancements behave in analogous ways when considering the triple or property created from each cell situated within a particular column. The distinction between `csv2rdf4lod` and traditional RDFS/OWL reasoners is that the former produces RDF inferences from tabular literals while the latter produce RDF inferences from existing RDF triples. This allows an LOGD publisher to choose between materializing the inferences with `csv2rdf4lod` to avoid using an inferencing engine at query time, or delaying the inferences until query execution and installing an additional inference engine.

---

[19] When renaming subjects, older names point to newer names using http://prefix.cc/con

[20] The full list of enhancements is at http://purl.org/twc/vocab/conversion/Enhancement.html

[21] The conversion vocabulary namespace is http://purl.org/twc/vocab/conversion/

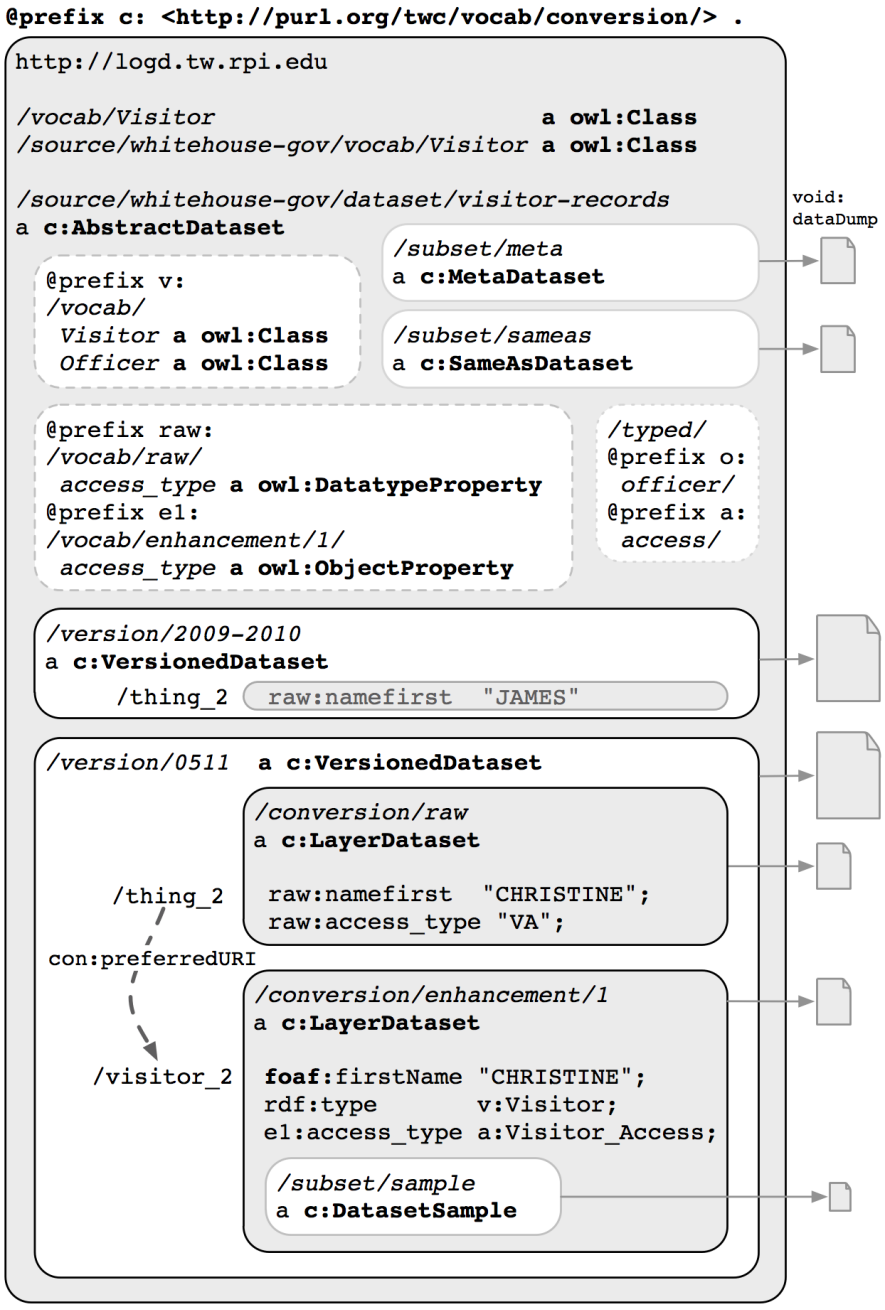

Fig. 2 Namespaces decompose according to *source*, *dataset*, and *version* identifiers assigned when retrieving data from other organizations; and *layer* identifiers assigned when interpreting it in different ways. Each step in the namespace decomposition corresponds to a void:Dataset URI that is a VoID superset of the datasets named within its namespace. URIs for entities, properties, and classes created from a dataset are named in namespaces corresponding to their breadth of applicability. Data integration is achieved incrementally by reinterpreting source data to use entities, properties, and classes from broader namespaces within this namespace decomposition or from existing vocabulary that is already used in the Semantic Web. For grammar defining URIs, see [4].

3.2.1 Most Popular Row-Based Enhancements

Enhancements can control the triple's subject[22]. **c:domain_template** is used to rename the subject of the triple. In the example shown in Figure 2, it was used to rename the subject from *:thing_2* to *:visitor_2*. Variables specified in the template are used to name the subject according to one or more values from the row. For example, one could name the visitor *:CHRISTINE_ADAMS* by using the template *"[#2]_[#1]"*. A string label provided by **c:domain_name** is used to type the subject[23]. A full class URI is created within the dataset's vocabulary namespace. For example, specifying *"Visitor"* will create the class *v:Visitor* in Figure 2 and type *:visitor_2* to *v:Visitor*. **c:subclass_of** is used to associate class URIs in the dataset's namespace to any class URI. This is done by associating the local class label to its superclass (e.g., *"Visitor"* to foaf:Person). Templates may also be used to specify the superclass (e.g., *"Visitor"* to *"[/]/vocab/Visitor"*).

Enhancements can control the triple's predicate. **c:label** is used to rename the property created from a column. For example, we could have renamed the *raw:access_type* property to *e1:has_access*. Renaming properties also enables one to merge properties from multiple columns[24]. **c:equivalent_property** is used to omit a local predicate in favor of an external one. For example, in Figure 2, we omit *e1:namefirst* and use *foaf:firstName* instead. **c:comment** will create an *rdfs:comment* on the predicate created. When we find the documentation[25] *"Type of access to the complex (VA = Visitor Access)"* for *access_type*, we can use this enhancement so that all conversions will further describe the predicate created.

Enhancements can describe how to interpret the value of a cell. **c:interpret** is used to replace entire cell values with alternative values. For example, we can specify that *"VA"* should be interpreted as *"Visitor Access"*. This can serve as a codebook for archaic abbreviations that the source organization uses. A special case of this can also specify that triples with empty object strings or values like *"!NULL!"* should be omitted. The **c:*pattern** enhancements specify how to interpret the cells as dates and date times. For example *"M/d/yy HH:mm"* will cast the value into an *xsd:dateTime*. **c:delimits_object** can indicate a regular expression to use to delimit a single cell's value into multiple tokens. For example, the cell value *"AAPL,T"* would become two triples; one for *nyse:AAPL* and one for *nyse:T*.

Enhancements can control the triple's object. **c:range_template** is used to rename the object in the same way that **c:domain_template** renames the subject. **c:range** can cast the cell value to a typed literal[26], an *rdfs:Resource*, or keep it as an *rdfs:Literal*. **c:range_name** will type the object in the same way that **c:domain_name** types the subject (**c:subclass_of** is used in the same way, too).

---

[22] Due to space considerations, we are omitting an entire class of enhancements that specify structural characteristics of the input file. These are critical because they significantly reduce the need for manual edits to "prepare" an input file for conversion.

[23] Row subjects are left untyped until an enhancement can provide a meaningful one.

[24] The initial conversions never create the same predicate for two columns in the same table.

[25] http://www.whitehouse.gov/files/disclosures/visitors/WhiteHouse-WAVES-Key-1209.txt

[26] xsd:integer, xsd:decimal, xsd:boolean, xsd:date, xsd:dateTime

c:links_via specifies an RDF data source from which to assert *owl:sameAs* triples. For example, if a cell value is *"POTUS"*, we can point to an RDF dataset containing the triple *http://dbpedia.org/resource/President_of_the_United_States dct:identifier "POTUS"*. When promoting the cell value to a local resource (with **c:range**), it will also reference DBPedia's URI for the President of the United States[27]. This enhancement behaves like a locally-scoped *owl:InverseFunctionalProperty*.

3.2.2 Advanced Row-Based Enhancements

More advanced enhancements are also available. Table entries describing multiple concepts are "normalized" using **c:bundled_by**, which changes the subject of a cell's triple from the row URI to a URI created from another cell or a URI minted for an implicit entity. For example, the *raw:namelast*, *raw:namefirst*, and *raw:namemid* predicates in the White House Visitor Access Records dataset could be bundled into an implicit *foaf:Person*. The same dataset would actually be best represented by bundling values to about a half dozen different entities involved in a visit (e.g., the appointment caller, officer making the appointment, the meeting location, and a few time intervals). **c:object_search** is used to match arbitrary regular expressions against the cell value to assert descriptions of the subject by populating predicate and object templates with the expression's captured groups. For example, a regular expression finding stock ticker symbols in tweet texts can result in annotating the tweet's URI with *sioc:subject* triples citing the stock's URI. This eliminates the need for applications to use the SPARQL *regex* filter. **c:predicate/c:object** pairs can also be used to add arbitrary descriptions to the subjects and objects created from a converted cell. **c:ExampleResource** annotates particular rows as exemplary, which become void:exampleResources in the resulting conversion. **c:multiplier** will scale values when casting to numeric datatypes.

3.2.3 Beyond Binary Relations: Enhancing with Cell-Based Subjects

The initial conversion interprets columns as *binary* relations; where rows become subjects, columns become predicates, and cells assert a single triple with an untyped string literal. However, many tabular data represent *n-ary* relations. For example, life expectancy in Wales by region, age, and time[28]; estimated and actual budgets for U.S. federal agencies by fiscal year and program type[29]; and states that have instated anti-smoking laws in bars, restaurants, and workplaces over several years[30] are all poorly represented with an interpretation that assumes a binary relation. Although many tabular data that require n-ary interpretations are statistical, they need not be.

---

[27] *:POTUS owl:sameAs http://dbpedia.org/resource/President_of_the_United_States .*

[28] http://purl.org/twc/pages/qb-example

[29] http://purl.org/twc/tiny-url/nitrd-fy11

[30] http://purl.org/twc/tiny-url/nci-nih-smoking-law-coverage

When a table expresses an n-ary relationship, the columns are not *relations*, but *entities involved in* a relation. This situation is better represented by using the cell as the subject of the triple and asserting many triples from the cell to other values in the row, one or more entities in the column header(s), and the cell value[31]. The following example illustrates the results of interpreting the U.S. budget statistics as row-based binary relationships versus a cell-based n-ary relationship. The latter permits the addition of new entities without requiring new predicates and modified queries to account for them, which is an objective of the RDF Data Cube effort[32].

```
:thing_6
 raw:agency                                          "NIH 2";
 raw:fiscal_year                                     "FY 2010";
 raw:estimate_request                                "Estimate";
 raw:high_end_computing_infrastructure_and_application "468.3".

:thing_6_4
 base_vocab:agency         typed_agency:NIH;
 base_vocab:fiscal_year    :FY_2010;
 e1:estimate_request       :Estimate;
 e1:program_component_area :HECIA;
 rdf:value                 468300000;
 muo:measuredIn            dbpedia:United_States_dollar .
```

3.3 Extending VoID to Organize Incremental Developments

In Sections 3.1 and 3.2, we noted that three identifiers (*source*, *dataset*, *version*) are assigned when creating an initial LOGD conversion and a fourth identifier (*layer*) is assigned when enhancing it. Now that we have introduced the six *stages* of the LOGD production workflow, we can revisit three of them to consider the *sets of data* resulting from their completion. Three specializations of void:Dataset are used to group triples resulting from different incremental stages that are performed. Figure 2 illustrates the void:subset hierarchy of one abstract dataset, two versioned datasets, and three[33] layer datasets. Merely **naming** a dataset with *source* and *dataset* identifiers does not result in any data triples. However, this abstract dataset is often the first level at which a data consumer will discover it[34] or the only level at which a data publisher will maintain its offerings[35]. For example, Figure 2 illustrates the abstract dataset *visitor-records* that is named with a URI created by appending its source and dataset identifiers to a base URI[36]. Merely **retrieving** data files also does not result

[31] https://github.com/timrdf/csv2rdf4lod-automation/wiki/Converting-with-cell-based-subjects

[32] http://publishing-statistical-data.googlecode.com

[33] To abbreviate, the *raw* layer of versioned dataset *2009-2010* is neither named nor typed.

[34] For example, we mentioned the abstract dataset "White House Visitor Records" when introducing our running example in the beginning of Section 3.

[35] For example, data.gov does not distinguish among dataset versions, just abstract datasets.

[36] For a grammar that defines most of the URI design, see [4].

in any data triples, but will lead to RDF when they are converted. Figure 2 shows the URIs for two versioned datasets (ending in *2009-2010* and *0511*), which are VoID subsets of the abstract dataset *visitor-records*. Versioned datasets are named with URIs formed by appending their version identifier to the abstract dataset's URI. Finally, **converting** the data files[37] from *0511* creates a layer dataset named *raw* that is a VoID subset of the versioned dataset. A second layer dataset (*enhancement/1*) of *0511* is created when a second interpretation is applied to the same input data. Although other specializations of void:Dataset are shown in Figure 2, they do not delineate sets of triples resulting from one of the six LOGD production stages.

3.4 Reusing Dataset URIs for Named Graph Names

Organizing RDF datasets (*abstract*, *versioned*, and *layer*) according to results of three incremental integration stages (*name*, *retrieve*, and *enhance*) that reflect three levels of granularity and are consistently named according three provenance-based identifiers (*source*, *dataset*, and *version*) allows a data consumers to navigate, evaluate, and adopt the portions of LOGD appropriate for their use. To complete this consistency from retrieval to publishing, datasets are loaded into a triple store's named graphs whose names correspond to the URIs of the datasets being loaded. This allows a data consumer to anticipate a dataset's location within a triple store when knowing only the URI of the dataset, which is consistently constructing knowing only the source, dataset, and version of interest. Practically, loading the RDF of a particular dataset is achieved by resolving its URI, selecting the URL of its void:dataDump, and loading the dump file into the triple store's named graph. To facilitate data cataloging and exploration, all c:MetaDataset void:subsets are found and loaded into a single named graph.

3.5 Handling Updated Datasets by Creating a Second Version

After naming, retrieving, adjusting, converting, enhancing, and publishing a dataset from another organization, data curators are likely to face the situation where the source organization updated the dataset's original data files. Although the change could happen for a variety of reasons (it could contain corrections, augment the previous, or simply replace it), they all present challenges that can be addressed using a three-level (*source*, *dataset*, *version*) URI naming scheme.

As mentioned earlier, the source, dataset, and version identifiers *whitehouse-gov*, *visitor-records*, and *0510* identify the data that was available from the White House on July 8th, 2010. Although the requirement to assign a version identifier for a dataset before retrieving any data may seem superfluous, its importance becomes

[37] http://www.whitehouse.gov/files/disclosures/visitors/WhiteHouse-WAVES-Released-0511.zip

evident when we revisit the same dataset page on August 30th, 2010 to find that the previous file has been replaced with a new one[38]. While the structure of the table – and its intended interpretation – did not change, the content completely changed. Assigning a new version identifier (*whitehouse-gov*, *visitor-records*, ***0810***) distinguishes the RDF produced from this newly retrieved data file. The same is true for ***0910*** (last modified September 24, 2010 when it was retrieved on October 1, 2010), ***0511*** (last modified May 27, 2011 when it was retrieved on June 15, 2011), and ***2009-2011*** (last modified December 29, 2010 when retrieved on June 15, 2011).

As mentioned in Section 3.2, `csv2rdf4lod` uses enhancement parameters encoded in RDF to create an enhancement layer. Since these were already defined for the first version retrieved, they are reapplied to the data files of new versioned datasets without requiring additional human effort to re-specify the parameters.

3.6 Using Dataset URIs as Namespaces for Entities and Vocabularies

Dataset URIs are used as namespaces for the entities that they describe and the vocabularies they use to describe them. In general, we cannot assume that the entities described by rows in different versions of a table are identical. For example, comparing the first data row of versions *2009-1011* and *0511* in Figure 2, we see drastically different first names *"JAMES"* and *"CHRISTINE"*. So, different URIs are created for subjects (*:thing_2* in ***2009-1011*** versus *:thing_2* in ***0511***). The URIs for predicates and objects, however, are shared across versions. For example, *raw:namefirst* and *a:Visitor_Access* are used in both versions. Although these characteristics are a fundamental aspect of being a data*set*, the enhancements **c:domain_template**, **c:equivalent_property**, and **c:range_template** are available to change this default behavior to suit different kinds of updates performed by a source organization.

Although decomposing the namespace according to provenance-based identifiers (*source*, *dataset*, and *version*) provides an effective way to distinguish among *who* "said" *what* (*when*), enhancements must be specified to integrate what is being said. Fortunately, the same namespace decomposition provides a natural scheme to incrementally integrate datasets of interest using a bottom-up technique. The URIs for entities, predicates, and classes are controlled by providing URI templates that are evaluated during conversion, giving curators control to create vocabularies and entity names at any level within the namespace or use external vocabularies directly – all within a single enhancement configuration. For example, in Figure 2, *v:Visitor* is scoped by the abstract dataset, but *Visitor* classes that apply across all White House datasets (*/source/whitehouse-gov/vocab/Visitor*) or across all source organizations (*/vocab/Visitor*) are also defined. The inverse operation becomes very powerful; we can now query for all datasets within our entire collection that mention visitors.

[38] http://www.whitehouse.gov/files/disclosures/visitors/WhiteHouse-WAVES-Released-0827.csv

4 Transparent LOGD Production Using Provenance Metadata

Although Linked Data provides many benefits, the retrieval, conversion, enhancement, and republication of another organization's data raises important questions about the integrity of the resulting data products and any applications that rely upon them. An inherent consequence of integrating data from disparate sources is that *distinctions diminish*. Once integrated, important distinctions such as *who*, *where*, and *when* information came from are at risk. The preservation of these distinctions becomes increasingly important when the sources of integration vary significantly in degrees of authority, reputability, policies, and documentation. Ironically, an integrated content view obscures important answers about *how* it came to be.

Although the *results* from the LOGD production workflow are important and useful for the open government and linked data communities, we do not consider the workflow a success unless it also accounts for *how* those results came to be. To achieve this transparency, the workflow captures a wealth of context when performing each of the six stages of integration. When **naming** the dataset, identifiers for the *source*, *dataset*, and *version* are used to frame the integration process around *who* is providing *what* data, and *when* they provided it. By using these identifiers to construct the URI for the dataset, and by using the dataset URI as a namespace for the entities it mentions and the vocabulary it uses to describe them, we implicitly encode three aspects of provenance for datasets, their entities, and their vocabulary. When **retrieving**, the user account initiating retrieval, the government URL requested, time requested, a variety of HTTP interactions, and the received file's checksum are captured and encoded in RDF. This information is critical because it establishes the connection between the original source and all subsequent results. When **adjusting**, results are stored separately from their originals and are associated to their predecessors by indicating the type of process applied and citing the user account that reported the modifications. When **converting** and **enhancing**, the invocation time, converter version and hash, input file, enhancement parameters, enhancement authors, and the user account invoking the conversion are captured. When **publishing**, the intermediate and final results are hosted on the Web and associated with the provenance descriptions captured throughout the production process. Finally, provenance of loading a named graph with an RDF file from the Web is stored in the named graph itself, enabling users to trace data from what they are querying, through the conversion process and retrieval, and to the original source.

Provenance metadata describes the context of critical steps throughout the LOGD production workflow. Consumers that are merely looking for additional ways to understand the data or may even question the validity of the final product may use this additional information to determine for themselves whether the concerns they have are caused by the aggregation process or rest with the original data source. The ability to accurately and comprehensively acknowledge organizations and individuals contributing to a result is an additional benefit of having provenance information, which not only facilitates conformance with data usage policies, but also provides the basis of incentive structures to reward previous contributions while motivating additional contributions [16].

The application of provenance within the TWC LOGD Portal is an ongoing research area, but we have already established processes for data consumers to debug mashups collaboratively [14], access explanations for the workflows that lead to mashups [13], and have greater trust in mashup results [11]. The provenance at the triple level[39] that `csv2rdf4lod` provides allows inquiry and inspection at the assertion level, such as *How do you know that the UK gave Ethiopia $107,958,576 USD for Education in 2007/8?*, which is answered by clicking an *Oh yeah?*[40] link that provides URL of the original government spreadsheet, the cell that caused the triple, the interpretation parameters applied, and the author of the transformation parameters. This information is obtained by invoking a SPARQL DESCRIBE query on the triple's subject and predicate, causing provenance fragments of the original CSVs rows and columns to be combined by a custom Jena DESCRIBE handler.

The provenance of the LOGD workflow has been encoded primarily using the Proof Markup Language [12], but other popular provenance vocabularies such as Provenir [17], Hartig's Provenance Vocabulary [8], and OPM [15] have been incorporated to describe certain aspects when it is more natural to do so. Other more traditional vocabularies (e.g., FOAF, SIOC, DC Terms, NFO[41]) have also been used where appropriate.

5 LOGD Deployment and Usage

5.1 Cost-effective Deployment of Linked Open Government Data

Our work on LOGD production suggests architectural guidelines and design patterns for the development of LOGD ecosystems. The TWC LOGD Portal implementation demonstrates how LOGD can be generated quickly, at fairly low cost and can be incrementally improved through systematic enhancement. We have also found opportunities to further reduce human intervention in LOGD production through automation. For example, rather than relying on users to contribute links across LOGD datasets, several semi-automated methods have been developed [3, 10], including the automatic detection of special entities such as U.S. state identifiers that have been instantiated across different LOGD datasets, and using OWL inference to connect semantically-related properties.

Linked Open Government Data, together with relevant Semantic Web technologies, was officially deployed by the U.S. government as part of its open government data initiative in May 2010. As of May 2011 the TWC LOGD Portal hosts more than 9.9 billion RDF triples from 1,838 OGD datasets published by 82 different data sources from over twenty countries, including special political regions and

[39] The triple-level provenance that `csv2rdf4lod` provides is reification-based, so the size of the provenance encoding is a function of the sum, *not* the product, of the table's rows and columns.

[40] The "Oh yeah?" button is described at http://www.w3.org/DesignIssues/UI.html

[41] http://www.semanticdesktop.org/ontologies/nfo/

international organizations; most of these datasets are from Data.gov. The Portal infrastructure has enhanced 1,505 datasets and has accumulated 8,335 *owl:sameAs* statements for 37 datasets (including 25 Data.gov datasets) linking to LOD datasets such as DBpedia, GeoNames and GovTrack.

TWC has made its `csv2rdf4lod` conversion tool, demo source code, SPARQL queries and configurations for dataset conversions available as open source. We are also currently working with several community-based organizations to mentor them in the creation and exploitation of LOGD directly from local-government data sources in their localities. We have also recently extended the TWC LOGD Portal with two key additions: an *Instance Hub* that will serve as a catalog of canonical URIs to be used when producing Linked Data based on U.S. government datasets, and a *International LOGD Dataset Catalog*[5][42] that provides a comprehensive, searchable, RDF-based inventory of over 300K OGD datasets by aggregating over 50 OGD dataset catalogs released by over 20 countries . We believe both the Instance Hub and International LOGD Dataset Catalog will be valuable resources that can be used and maintained by the LOGD community.

5.2 Scalability and Quality in LOGD Production

It is computationally prohibitive to turn all OGD datasets into high quality enhanced data. Therefore, our LOGD production offers both *basic* LOGD production (using "raw" conversion configuration) requiring little if any human intervention, and *advanced* LOGD production (using "enhancement" conversion configuration) that enables users to use their domain knowledge to generate higher-quality LOGD data. The former approach is highly scalable since most of it is automated, while the latter approach supports the need for high quality LOGD production. Additional scalability is achieved through the social aspects of LOGD production. By decomposing the OGD data processing workflow into smaller stages it is possible to assign tasks to contributors with appropriate skill sets and domain knowledge.

5.3 Rapid LOGD Mashup Development using LOGD datasets

LOGD consumption complements its production. Producing well-structured and well-connected data facilitates the conception and creation of mashups to combine multiple government datasets or leverage datasets outside government domain. While we briefly discuss how LOGD datasets can be used to construct mashups here, a more detailed description of LOGD mashups, see *The Web is My Back-end* also in this issue.

[42] http://purl.org/twc/application/international-logd-dataset-catalog

LOGD datasets have been made available on the TWC LOGD Portal as downloadable RDF dump files. Application developers can load the LOGD datasets (typically LOGD produced from the latest version of a dataset) into a SPARQL endpoint to enable Web-based SPARQL queries. Applications then submit SPARQL queries to the endpoints to integrate multiple datasets and retrieve data integration results. LOGD datasets can be further linked by common entity URIs generated during the enhancement process.

Application developers may also query multiple SPARQL endpoints to achieve larger-scale data integration. A typical example might be to query the TWC LOGD Portal's SPARQL endpoint to retrieve government data (containing entities that map to DBpedia using owl:sameAs), and then query DBpedia for additional descriptions about the entities in government data. For example, the "Linking Wildland Fire and Government Budget" mashup[43] mashes up U.S. government budget information (released by OMB), statistics of wildland fire (released by Department of the Interior) with famous fires reported on Wikipedia.

The declarative, open source nature of LOGD datasets and the provenance metadata associated with LOGD data makes LOGD consumption more transparent. Developers can locate LOGD datasets used in demonstrations and learn to integrate multiple datasets by reading the corresponding SPARQL queries. For example, in recent Web Science courses at RPI senior undergraduate students in the Information Technology program successfully completed course projects that required them to learn from the published LOGD datasets and corresponding demos on the TWC LOGD Portal.

6 Related Work

Dataset Catalog Services: Open government data initiatives typically begin with the publication of online catalogs of raw datasets; these catalogs usually feature keyword search and faceted browsing interfaces to help users find relevant datasets and retrieve corresponding metadata including dataset descriptions and download URLs. For example, Data.gov maintains three dataset catalogs including the *Raw Data Catalog*, *Tool Catalog* and *Geodata Catalog*: the first two share one faceted search interface, while the *Geodata Catalog* has a separate interface. Data.gov also uses a Microsoft BING-based search. The OpenPSI Project (http://www.openpsi.org) collects RDF-based catalog information about the UK's government datasets to support government-based information publishers, research communities, and Web developers. CKAN (Comprehensive Knowledge Archive Network) (http://ckan.net/) is an online registry for finding, sharing and reusing datasets. As of January 2011 about 1600 datasets had been registered with CKAN, and CKAN has been used to generate the LOD cloud diagram and to support dataset listings in Data.gov.uk. CKAN publishes its native dataset metadata in JSON format but is also experi-

[43] `http://logd.tw.rpi.edu/demo/linking_wildland_fire_and_government_budget`

menting with RDF encoding (http://semantic.ckan.net/). The TWC LOGD Portal publishes LOGD dataset metadata in RDF and provides a combined search over the three catalogs of Data.gov. As noted earlier, TWC is currently extending its metadata-based search technique to include federated government data from around world.

API-based OGD Data Access: As an alternative to making raw data directly available for download, several projects offer Web-based data APIs that enable developers to access government data within their applications. For example, the Sunlight Foundation (http://sunlightfoundation.com/) has created the National Data Catalog (http://nationaldatacatalog.com/) which makes federal, state and local government datasets available and provides data APIs via a RESTful Web service. Socrata (http://opendata.socrata.com) is a Web platform for publishing datasets that provides a full catalog of all their open government datasets, along with tools to browse and visualize data, and a RESTful Web API for developers. Microsoft has also entered this space with their OData (http://www.odata.org) data access protocol and their Open Government Data Initiative (OGDI) (http://ogdi.codeplex.com); recently a small number of OGD datasets have been published on Microsoft's Azure Marketplace DataMarket (https://datamarket.azure.com/). Currently, none of these platforms enable data to be linked specifically at the data level, and none of the APIs provide a way for developers to see or reuse the underlying data model, making it hard to extend the model or use it for further mashups.

Linked Open Government Data: There is an increasing number of Linked Data projects involving government data in the U.S. and around the world. GovTrack (http://www.govtrack.us) is a civic project that collects data about the U.S. Congress and republishes the data in XML and as Linked Data. Goodwin et al. [7] used Linked Geographical Data to enhance spatial queries on the administrative geographic entities in Great Britain. Data.gov.uk has released LOGD datasets together with OGD raw datasets since its launch in January 2010. The LOD2 project [6] proposes a definition for *knowledge extraction* that can provide guidance in "better" or "worse" information modeling choices. According to their definition, knowledge extraction requires "the extraction result to go beyond the creation of structured information" by "reusing existing formal knowledge (identifiers or ontologies)" and "facilitating inferencing". The group surveyed tools and techniques for knowledge extraction, described them using an OWL Tool Survey Schema, and provide the results as linked data[44]. The diversity of tools in their survey and others like it[45] suggest a variety of requirements from different shareholders within the Linked Data community, and by extension the Linked Open Government Data community.

[44] http://data.lod2.eu/2011/tools/ket/

[45] http://www.mkbergman.com/sweet-tools/

7 Conclusion and Future Work

The TWC LOGD Portal has been recognized as playing an important role in U.S. open government data activities including helping with the deployment of Semantic Web technologies within the Data.gov website, the official access point for open government data from U.S. federal government agencies. This success is due in part to the large volume of LOGD data (billions of RDF triples) produced by the TWC LOGD portal and the agile development of demonstrations of LOGD data published through the Portal. In particular, the LOGD production infrastructure demonstrates a scalable solution for converting raw OGD datasets into RDF, and an extensible solution for the incremental enhancement of LOGD to improve its quality.

TWC's LOGD production infrastructure makes both research and development contributions, especially: we designed a data organization model for LOGD datasets to support persistent and extensible LOGD production; we have developed a collection of open source tools, especially `csv2rdf4lod`, to provide infrastructural support to LOGD production automation; we have designed and captured provenance metadata, covering data structural relations and data processing workflow, to support deeper understanding of LOGD data and accountability evaluation over LOGD datasets from multiple sources.

Future work will always involve integrating additional OGD datasets, which is motivated by our recent international dataset catalog that aggregates metadata and download URLs for over three hundred thousand OGD datasets worldwide. While we have presented an infrastructure to systematically and repeatedly improve the quality of LOGD by linking entities and reusing existing ontologies, further work is needed to expose this functionality in a form approachable by users and their collaborators. One approach is our recent "instance hub" that will allow government agencies to define canonical URIs for entities frequently used in OGD datasets but not defined in Linked Data sources. Finally, as datasets become integrated and connect, it will be very interesting to begin to analyze this connectivity in general as well as for particular user interests.

8 Acknowledgements

The work in this paper was supported by grants from the National Science Foundation, DARPA, National Institutes of Health, Microsoft Research Laboratories, Lockheed Martin Advanced Technology Laboratories, Fujitsu Laboratories of America and LGS Bell Labs Innovations. Support was also provided by the Air Force Research Laboratory. Sponsorship details may be found on the TWC LOGD Portal.

References

1. Harith Alani, David Dupplaw, John Sheridan, Kieron O'Hara, John Darlington, Nigel Shadbolt, and Carol Tullo. Unlocking the potential of public sector information with semantic web technology. In *ISWC/ASWC*, pages 708–721, 2007.
2. Tim Berners-Lee. Putting government data online. `http://www.w3.org/DesignIssues/GovData.html`, accessed Sep 25, 2010, 2009.
3. Li Ding, Dominic Difranzo, Alvaro Graves, James Michaelis, Xian Li, Deborah L. McGuinness, and Jim Hendler. Data-gov wiki: Towards linking government data. In *Proceedings of the AAAI 2010 Spring Symposium on Linked Data Meets Artificial Intelligence*, 2010.
4. Li Ding, Timothy Lebo, John S. Erickson, Dominic DiFranzo, Gregory Todd Williams, Xian Li, James Michaelis, Alvaro Graves, Jin Guang Zheng, Zhenning Shangguan, Johanna Flores, Deborah L. McGuinness, and Jim Hendler. Twc logd: A portal for linked open government data ecosystems. *Web Semantics: Science, Services and Agents on the World Wide Web*, In Press, Accepted Manuscript:–, 2011.
5. John Erickson, Yongmei Shi, Li Ding, Eric Rozell, Jin Zheng, and Jim Hendler. Twc international open government dataset catalog (triplification challenge submission). In *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS 2011, Graz, Austria.*, ACM International Conference Proceeding Series. ACM, 2011.
6. Sebastian Hellmann et al. Report on knowledge extraction from structured sources. Technical Report Deliverable 3.1.1, LOD2 - Creating Knowledge out of Interlinked Data, March 2011.
7. John Goodwin, Catherine Dolbear, and Glen Hart. Geographical linked data: The administrative geography of great britain on the semantic web. *Transactions in GIS*, 12(s1):19–30, 2009.
8. Olaf Hartig. Provenance information in the web of data. In *Proceedings of the Linked Data on the Web (LDOW) Workshop at WWW*, Madrid, Spain, 2009.
9. Christoph Lange. Krextor – an extensible XML→RDF extraction framework. In Chris Bizer, Sören Auer, and Gunnar AAstrand Grimnes, editors, *Scripting and Development for the Semantic Web (SFSW)*, number 449 in CEUR Workshop Proceedings, Aachen, May 2009.
10. Timothy Lebo and Gregory Todd Williams. Converting governmental datasets into linked data. In *Proceedings of the 6th International Conference on Semantic Systems*, I-SEMANTICS '10, pages 38:1–38:3, 2010.
11. Xian Li, Timothy Lebo, and Deborah L. McGuinness. Provenance-based strategies to develop trust in semantic web applications. In *The Third International Provenance and Annotation Workshop (IPAW 2010)*, pages 182–197, 2010.
12. Deborah L. McGuinness, Li Ding, Paulo Pinheiro da silva, and Cynthia Chang. Pml 2: A modular explanation interlingua. In *Proceedings of the AAAI 2007 Workshop on Explanation-Aware Computing*, July 2007.
13. Deborah L. McGuinness, Vasco Furtado, Paulo Pinheiro da Silva, Li Ding, Alyssa Glass, and Cynthia Chang. Explaining semantic web applications. In *Semantic Web Engineering in the Knowledge Society*, pages 1–24. Information Science Reference, 2008. (chapter 1).
14. James Michaelis and Deborah L. McGuinness. Towards provenance aware comment tracking for web applications. In *The Third International Provenance and Annotation Workshop (IPAW 2010)*, pages 265–273, 2010.
15. Luc Moreau, Ben Clifford, Juliana Freire, Joe Futrelle, Yolanda Gil, Paul Groth, Natalia Kwasnikowska, Simon Miles, Paolo Missier, Jim Myers, Beth Plale, Yogesh Simmhan, Eric Stephan, and Jan Van den Bussche. The open provenance model core specification (v1.1). *Future Generation Computer Systems*, 27(6):743 – 756, 2011.
16. M. A. Parsons, R. Duerr, and J.-B. Minster. Data Citation and Peer Review. *EOS Transactions*, 91:297–298, August 2010.
17. Satya S. Sahoo, Christopher Thomas, and Amit Sheth. Knowledge modeling and its application in life sciences: A tale of two ontologies. In *In Proceedings of WWW*, 2006.
18. Aaron Smith. Government online. URL: `http://www.pewinternet.org/Reports/2010/Government-Online.aspx`, accessed on Jan 25, 2011, 2010.