Towards Networked Linked Data-Driven Web3D Applications

René Schubotz
EADS Innovation Works
rene.schubotz@eads.net

Andreas Harth
Institute AIFB
Karlsruhe Institute of Technology (KIT)
harth@kit.edu

ABSTRACT

The Web of Data has grown to a size of several billion triples and provides human interface opportunities and challenges beyond those of the traditional Web. Although Linked Data is now generated at a fast pace and very large scale, we observe that browsing and visualisation of Linked Data is still in its infancy. In parallel to the enormous boost in Linked Data, recent work on integrating 3D graphics capabilities into the W3C technology stack provides fresh momentum for an effort to extend the Web with 3D content and technologies. In light of the timid uptake and consumption of Linked Data by non-technical audiences, we make the case for Web3D-based user interfaces to the Web of Data and aim at promoting synergistic research in the Web3D and Web of Data communities. To that end, we describe a scenario that requires the combination of Linked Data from geospatial and encyclopedic data sources and the transformation of the combined Linked Data into a format amenable to Web3D rendering. Based on the scenario, we derive high-level requirements and propose a Linked Data-driven design pattern based on REST architecture principles that satisfies the requirements. Our prototypical implementation shows that a combination of current Web technologies is sufficient to implement distributed application that ultimately arrive at Web3D renderings of Linked Data. Based on the experiments, we identify and discuss areas which require further research.

1. INTRODUCTION

The World Wide Web has irreversibly changed the way we collect, validate and disseminate data and information. The Web is built on the idea of hyperlinking documents into a single *global document space*, hereby providing the fundamental principles for accessing, navigating and publishing Web documents. Until recently, however, the principles that promoted the Web of Documents have not been applied to the abundance of data related to media, bibliography, encyclopedical knowledge, or geo sciences. Historically, these data sets have been made available as documents in various content formats or from databases accessible through Web APIs. Lacking uniform identification, retrieval and representation mechanisms, Web data sets have been fragmented into disconnected data silos.

To publish, access and integrate data on the Web, Berners-Lee [4] devised the Linked Data Principles, a set of best practices based on a small set of standards, i.e., Uniform Resource Identifiers (URIs), the Hypertext Transfer Protocol (HTTP), the Resource Description Framework (RDF) as well as a number of serialisation formats. By establishing typed data-level links between items from disparate data sources, Linked Data facilitates the creation of a single *global data space*, enables the integration of data from disparate sources and ultimately provides expressive query capabilities.

Automated production and interweaving of Linked Data at a large scale has now grown the Web of Data to a size of 25.2 billion triples[1] and an increasing number of applications demonstrates the potential of exploiting the Web of Data. For example, the GADM-RDF[2] project contains the world's administrative boundaries and interlinks to existing spatial datasets. The DBpedia[3] project [2] constantly derives a Linked Data corpus from the Wikipedia encyclopedia and describes more than 3.64 million interlinked entities, containing factual data on persons, places and organisations.

The growing Web of Data, however, provides human interface opportunities and challenges beyond those of the traditional Web. A significant challenge is found in the formalised RDF encoding of Linked Data. RDF, a format created for machine consumption, limits the use of Linked Data to those capable of reading and interpreting raw RDF. Even for a technical audience, the lack of visual aid impedes the identification of relevant information and the unearthing of previously unseen correlations. Aiming at improved accessibility and usability of Linked Data, a limited number of Linked Data browsers and visual search engines offer both text-based and visual presentation options. The Tabulator browser [5], for example, provides several conventional user interface metaphors such as a timeline and a tabular view. VisiNav[11], a system based on an interaction model designed to search and navigate large amounts of Linked Data, offers amongst other things a 2D graph visualisation

[1] `http://www4.wiwiss.fu-berlin.de/lodcloud/`
[2] `http://gadm.geovocab.org/`
[3] `http://dbpedia.org/`

of Linked Data items. Although Linked Data is now generated at very large scale, we observe that visualisation and browsing of Linked Data is still in its infancy.

In parallel to the enormous boost in Linked Data technologies, recent work [3, 15, 10] on integrating 3D graphics capabilities into the W3C technology stack, together with a general consumer trend in 3D movies and televisions, provides fresh momentum for an effort to extend the Web with 3D content and technologies. Using technologies such as JavaScript, WebGL, Cascading Style Sheets and the Document Object Model increasingly sophisticated 3D content starts now to be seamlessly integrated with HTML.

Clear and coherent Web3D visualisations may help in conveying complex Linked Data, support users in sense making and information exploration and discovery, and provide descriptive, intuitive and effective user interfaces to the Web of Data. For example, Web3D graph visualisations may be used to display the overall structure of the Web of Data and support navigation and exploratory discovery of individual Linked Data sets and items. More importantly, after obtaining an overall understanding of data content and structure, a detailed inspection and analysis of relevant Linked Data items typically follows. For small amounts of Linked Data, a textual presentation is often an adequate choice for the technical audience. However, as the amount and diversity of the result sets grow, suitable Web3D visualisations may be an appropriate tool for improving the users's understanding.

In light of the timid uptake and consumption of Linked Data by non-technical audiences, we aim at promoting synergistic research in the Web3D and Web of Data communities. We therefore describe an experiment into publishing, accessing and combining Linked Data and Web3D graphics functionality to arrive at a 3D user interface for geospatial and statistical Linked Data.

Our contribution is in showing how to combine Linked Data and Web3D functionality accessible via RESTful interfaces into a coherent system which allows for the querying, combination and transformation of Linked Data into a format amenable to Web3D rendering.

The remainder of the paper is organised as follows. In Section 2 we cover related work, and in Section 3 we derive and describe several high-level requirements a *Linked Data-driven Web3D application* should fulfil. We provide an application prototype as well as a description of its essential components in Section 4. Furthermore, we discuss the interplay of data-providing and data-processing RESTful services. In Section 5 we discuss our lessons learned, point out limitations and suggest areas which require future work. We conclude with summary in Section 6.

2. RELATED WORK

There are several approaches in the literature that describe the design and functionality of user interfaces for the Web of Data. These user interfaces can be grouped into systems that solely provide text-based functionality and those that include visual presentation options.

We start by reviewing systems that use textual representations of Linked Data items and their relationships. Dipper[4] provides a Linked Data endpoint for retrieval and browsing of Linked Data from a set of Linked Data repositories. Where available human-readable RDF labels are used to format the results. The Disco Hyperdata Browser[5] allows the user to navigate the Web of Data and renders all information as a HTML table of property-value pairs. Where available, human-readable labels are used to format the results. As for Disco, Marbles[6] allows navigating the Web of Data. Marbles strives to improve the user experience by formatting Linked Data for presentation to end users; different colours are used to distinguish the sources of the information retrieved. Piggy Bank[7], a Firefox extension, allows the extraction of Linked Data from web sites and enables users to perform faceted search and browsing across Linked Data collected by multiple users. Again, results are presented using list and tabular interfaces.

User interfaces offering visual presentations are provided by more advanced RDF browsers or specialised Linked Data applications. For example, the Tabulator browser [5] provides several conventional user interface metaphors such as a timeline and a map view. RDF Gavity[8] is a tool for visualising ontologies on top of the JUNG Graph API and supports user-specified graph filters for specific views. Browsers such as Fenfire[9] or FOAFNaut[10] visualize the relationships among Linked Data items employing a graph visualisations. VisiNav[11], a system designed to search and navigate large amounts of Linked Data based on an interaction model, offers amongst other things a 2D graph visualisation of Linked Data items and their relationships.

Although Linked Data is now generated at very large scale, we observe that visualisation and browsing of Linked Data is still in its infancy. With the exception of DBpedia Mobile[11], a location-centric DBpedia client application for mobile devices, most of the reviewed systems focus on supporting navigation and exploratory discovery in the overall Web of Data. Visualisation techniques for a detailed inspection and analysis of Linked Data query results, i.e., a user's particular region of interest in the Web of Data, are typically restricted to tabular or list presentations. In contrast to these existing systems, we provide a proof-of-concept prototype illustrating how to achieve Web3D visualisations of Linked Data query results in a completely distributed architecture.

3. REQUIREMENTS FOR LINKED DATA-DRIVEN WEB3D APPLICATIONS

Following Hausenblas [12], a *Linked Data-driven Web application* consumes and potentially manipulates Linked Data sets and offers appropriate interfaces and content for both

[4] `http://api.talis.com/stores/iand-dev1/items/dipper.html`
[5] `http://www4.wiwiss.fu-berlin.de/bizer/ng4j/disco/`
[6] `http://www5.wiwiss.fu-berlin.de/marbles`
[7] `http://simile.mit.edu/wiki/Piggy_Bank`
[8] `http://semweb.salzburgresearch.at/apps/rdf-gravity/`
[9] `http://fenfire.org/`
[10] `http://www.foafnaut.org/`
[11] `http://wiki.dbpedia.org/DBpediaMobile`

humans and machines. We extend this definition towards *Linked Data-driven Web3D applications* in a straight-forward way and discuss several high-level requirements for Linked Data-driven Web3D applications in the following.

Unifying Web3D and Linked Data representations: The description and augmentation of 3D models and virtual spaces with semantic metadata and ontologies has been in the focus of the 3D modeling community for a long time, yet, the issue is far from resolved. The Linked Data community, however, successfully facilitates this task for disparate data sources by establishing typed data-level links between data items. We believe that several strategies exists in order to, at least partially, transfer the Linked Data principles to the Web3D. The conventional approach, cf. Pittarello et al. [14], is to describe the semantics of Web3D content in annotation tags provided by the Web3D specification or by means of RDFa[12] annotations at the tag attribute level. In order to benefit from such semantic annotations, the initial extraction and subsequent management of semantic scene annotations is required. A very radical concept following Kalogerakis et al.[13] is to accept RDF as the single data model for publishing 3D content on the Web and to merge graphics content and scene semantics into the Web of Data. However, this leaves open pressing questions on how to efficiently render and manipulate graphics content in its RDF encoding. As a third stance and as a trait of Linked Data-driven Web3D applications, we adopt REST principles throughout and retrieve Web3D content as REST representations of Linked Data resources by content-negotiation. Although an established REST design practice, we feel that this concept requires further thought.

On-demand integration: The diversity and size of Linked Data sets make it difficult – even in our prototype application – to just fetch and integrate Linked Data and relevant geometries from all sources and load the combined Web3D scene graph into a Web browser for rendering. Given that source datasets are often distributed, we require the overall Linked Data-driven Web3D application (dispersed across multiple providers of data, functionality and client-side rendering) to access and integrate data on demand using the pull model. Existing systems that demonstrate initial results for integrating semantics into virtual environments suffer from scalability and real-time performance issues. Our aim is to use caching and asynchronous communication to achieve performance conductive to interactive visualisations; our initial experiments indicate that our system requires more optimisations to achieve the goal.

Integration of content and functionality: Accepting the preceding argument of on-demand integration, a Linked Data-driven Web3D application needs to integrate both content and functionality. Examples for content are the GADM-RDF or DBpedia datasets; examples for functionality are 3D meshing and shading services. The various components have to be connected together in a way that achieve the intended application purpose. We see the Web as a platform and want to combine services that are decentralised.

[12]`http://www.w3.org/TR/xhtml-rdfa-primer/`

Declarative integration: We aim to use declarative means to describe data and links between the data, to arrive at an ecosystem in which data providers interlink their datasets without the need for direct coordination. An ideal system would figure out the data sources to combine with dedicated data processing services to ultimately arrive at the stated goal of a user. However, such a system has proven elusive. We thus opt for a declarative coordination of the interoperation between data sources and functionality.

Interactive visualisations: Typically, Linked Data-driven Web3D applications require that user input is possible, e.g. for selecting a visual encoding. For an interactive application, the overall system must provide high-throughput and high update rates with low latency. While these qualities require further optimisations of the underlying Linked Data and Web3D technologies, the overall user-perceived latency can be improved using techniques such as incremental content loading or user reassurance during wait times.

4. A PROTOTYPICAL APPLICATION

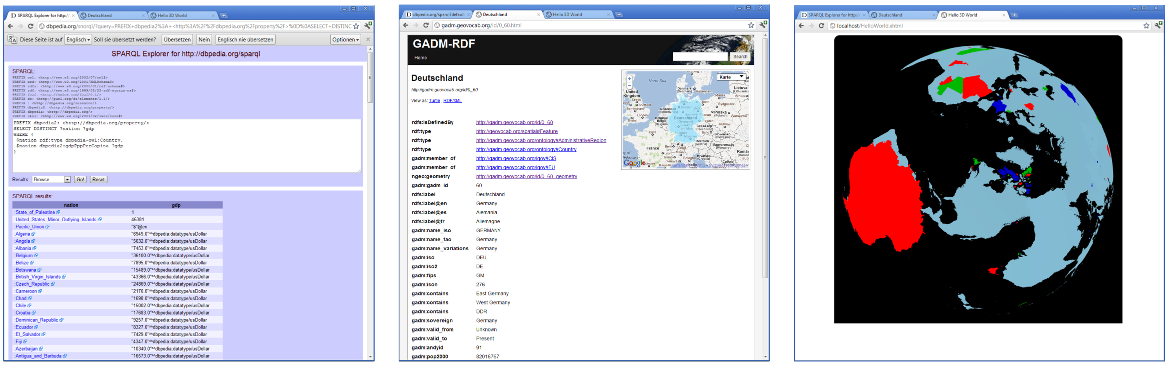
In the following, we describe a straightforward Linked Data-driven Web3D application leveraging the emerging 3D graphics capabilities of the W3C technology stack as well as the increasing amount of Linked Data available on the Web. Given statistical data from DBpedia (see Figure 1(a)) and geospatial data from GADM-RDF (see Figure 1(b)), we aim at generating an interactive 3D choropleth map [9] visualising the world distribution of wealth (see Figure 1(c)). We call our prototype application *"Hello (3D) World!"* and believe that its interactive Web3D interface allows the user to effectively grasp the meaning of the underlying Linked Data.

Realising the application involves several steps: querying across data sources; transforming query results into a Web3D scene graph; and rendering the Web3D scene graph. Based on REST principles, our prototype declaratively combines Linked Data into a format amenable to 3D rendering. To this end, we use RESTful services[13] for basic computer graphics, e.g. shading, meshing and coordinate transformations, SPARQL[14] for queries spanning multiple disparate Linked Data sources containing heterogeneous data, and XQuery[15] for executing the overall application logic.

Antedating a more detailed description of all involved artifacts, Figure 2 demonstrates our Linked Data-driven design and depicts the following interaction patterns:

1. Via *GADM-RDF*'s Linked Data Services[16], resolve Linked Data resources of the world's nations and retrieve RDF including (i) URIs for their KML geometries and (ii) owl:sameAs'[16] to interlinked data sets. This corresponds to executing the SPARQL query (cf. Listing 2) against *GADM-RDF*. Cache the results locally.
2. Resolve each ?*geo* URI from Step 1, parse the retrieved KML data and send each $< kml : Polygon >$ to

[13]`http://jersey.java.net/`
[14]`http://www.w3.org/TR/rdf-sparql-query/`
[15]`http://www.w3.org/TR/xquery/`
[16]owl:sameAs links indicate that two URI references actually refer to the same thing.



(a) DBpedia on the world's GDP. (b) Geo data in GADM-RDF. (c) *Hello (3D) World!*

Figure 1: Interactive Web3D choropleth map of the gross domestic product of countries.

the REST service *Geo2Ecef* transforming WGS84 into ECEF coordinates. Post the transformed polygons to the REST service *Poly2Mesh* which in turn responds with fresh URIs for each generated XML3D mesh. The results are cached locally.

3. Using the owl:sameAs's from Step 1, execute the query shown in Listing 3 against *DBpedia* and retrieve each nation's gross domestic product based on the purchasing power parity. Post the ?*gdp* literals to the REST service *Gdp2Shader* which responds with fresh URIs for XML3D shaders. Cache the results locally.

4. Use the URIs generated in Step 2 and Step 3 and generate an XML3D-compliant scene graph.

5. Return the generated XML3D-compliant scene graph to the browser for rendering and display.

In the following, we describe the general high-level components of Linked Data-driven Web3D applications using the example of *"Hello (3D) World!"*.

4.1 Runtime Environment

Linked Data-driven Web3D applications require some runtime environment for implementing their business logic. The business logic coordinates the invocation of the various data sources and services in the right order to transform and adapt the content into a 3D scene according to the specifications of the user. Such runtime environments should enable access to various distributed data sources using various languages, permit highly granular selection based on both the hierarchy and ordering of structured data, and allow for complex data transformations that may involve hierarchical restructuring.

Developed by the XML Query working group of the W3C, XQuery is defined as a *query language* for XML data. Path expressions can be used to select based on XML structure (both hierarchical and sequential) and XML result structures can be easily generated. The XQuery Update Facility provides the ability to update data or to add annotations and metadata; specialised capabilities beyond the large selection of built-in functions and operators are provided via an extension mechanism through function libraries. XQuery supports library modules, i.e., collections of function definitions and global variable declarations. The XQuery engine imports a module once during compilation, the compiled module is then made available through the static XQuery context. In particular, we make use of the EXPath http-client module[17]. This module allows one to deal with most aspects of the HTTP protocol, and hereby enables the *invocation of RESTful data providing and processing services.*

XQuery appears especially alluring given the fact that XSPARQL[18], a merge of SPARQL and XQuery providing concise solutions for mapping between XML and RDF in either direction, is defined as an extension of XQuery's formal semantics adding a few normalization mapping rules. In essence, XSPARQL queries are reduced to XQuery with interleaved calls to a SPARQL engine via the SPARQL protocol [1]. On this account and considering the fact that XQuery has been successfully brought to the Web browser [8, 6], we opted for XQuery to implement our business logic in a concise and declarative fashion (cf. Listing 1). For future work, we suggest the porting of XSPARQL to the Web browser and a thorough investigation on using JavaScript and rdfQuery[19] for implementing Linked Data-driven applications.

4.2 Linked Data Providers

Linked Data-driven Web3D applications consume and manipulate various data sets and offer appropriate interfaces and content for both humans and machines. While we strive to ultimately arrive at 3D renderings of complex data, some technology for the retrieval, combination and manipulation of data stored in disparate sources is needed. The query language SPARQL can be used to express queries across diverse Linked Data sources. Its syntax resembles SQL, but SPARQL is far more powerful, enabling queries spanning multiple disparate data sources containing heterogeneous data.

Our prototypical implementation employs SPARQL in order to retrieve Linked Data of the world's nations from GADM-

[17] `http://www.expath.org/modules/http-client/`
[18] `http://www.w3.org/Submission/xsparql-language-specification/`
[19] `http://code.google.com/p/rdfquery/`

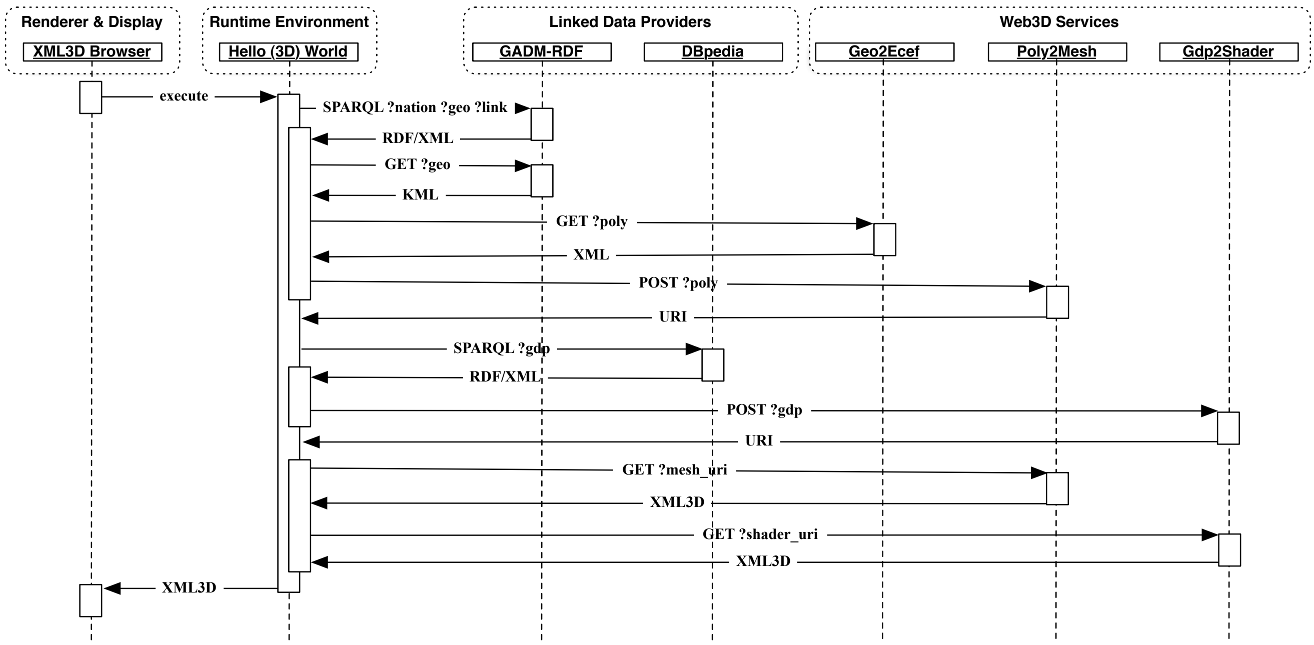


Figure 2: *"Hello (3D) World!"* sequence diagram.

RDF . The corresponding query shown in Listing 2. In addition, each nation's gross domestic product based on the purchasing power parity is retrieved from DBpedia using the SPARQL query shown in Listing 3.

```
PREFIX spatial: <http://geovocab.org/spatial#>
PREFIX ngeo: <http://geovocab.org/geometry#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
SELECT ?nation ?geo ?link
WHERE {
  ?nation rdf:type spatial:Country .
  ?nation ngeo:geometry ?geo .
  ?nation owl:sameAs ?link
}
```

Listing 2: SPARQL query against GADM-RDF

```
PREFIX dbpedia2: <http://dbpedia.org/property/>
SELECT ?gdp
WHERE {
  ?link dbpedia2:gdpPppPerCapita ?gdp .
}
```

Listing 3: SPARQL query against DBpedia

4.3 Web3D Services

Our prototype invokes Web3D REST services performing preprocessing and mapping operations. Invocation of the preprocessing service *geo2ecef* performs coordinate transformations, whereas the mapping services *poly2mesh* and *gdp2shader* produce xml3d elements. All Web3D services are implemented using JAX-RS and Jersey and deployed to Apache Tomcat Servlet containers. Adhering to Fielding's architectural principles [7], we designed these services in the following ways.

geo2ecef. This service is implemented to accept a kml:Polygon, to convert its geodetic coordinates into Earth-centred earth-fixed coordinates, and to respond with results in XML format. In the following, we give an exemplary GET request and the service's response.

```
GET /geo2ecef?input=
  '<kml:Polygon>x1,y1,z1...</kml:Polygon>' HTTP/1.1
Host: http://www.host.org
Accept: application/xml

HTTP/1.1 200 OK
Server: http://www.host.org
Content-Type: application/xml
<ecefPolygon>f(x1,y1,z1)...</ecefPolygon>
```

gdp2shader. This service accepts numerical values for the generation of xml3d:shader elements to be resolved by a Linked Data-driven application. The xml3d:shader element is derived from the service input and a predefined colour scale. Listing 4 shows XML3D code generated by this service. Anticipating more complex shader generation, the service is implemented as POST and responds with Linked Data containing fresh URIs for xml3d:shader elements. Additionally, this service provides a GET method for shader retrieval.

```
<shader script="urn:xml3d:shader:phong">
<float3 name="diffuseColor">0.08 0.0 0.0</float3>
<float name="ambientIntensity">0.4</float>
</shader>
```

Listing 4: Generated XML3D shader element

poly2mesh. When receiving POST requests, this mapping service accepts kml:Polygons, spawns triangulation threads

```
(: Set endpoints and queries :)
let $gadm := http://...gadm.../search?rdf_type=gadm:Country
let $dbpedia := http://dbpedia.org/sparql?query=
let $query :=   PREFIX dbpedia2: <http://dbpedia.org/property/>
                SELECT DISTINCT ?gdp { XX dbpedia2:gdpPppPerCapita ?gdp }
                LIMIT 1000
let $geo2ecef := http://...geo2ecf.../convert?polygon=
let $poly2mesh := http://...geo2mesh.../

(: Get nation URIs from GADM :)
let $nations := data( http:get-node( $gadm )[2]//rdfs:seeAlso/@rdf:resource )

(: Get geom URIs from GADM :)
let $geoms := for $nation in $nations
              return data( http:get-node( concat( $nation, ".rdf" ) )[2]//ngeo:geometry/@rdf:resource )

(: Get link URIs from GADM :)
let $links := for $nation in $nations
                     return data( http:get-node( concat( $nation, ".rdf" ) )[2]//owl:sameAs )

(: Query DBpedia, then get shader URIs :)
$let shaders := for $link in $links
                let $ep := fn:concat( $dbpedia, fn:replace( $query, 'XX' $link ))
                let $gdps := data( http:get-node( $ep )[2]//sparql:binding[@name="gdb"]/sparql:literal )
                        for $gdp in $gdps
                        return http:post( $poly2mesh, $gdp )

(: Get KML, parse <kml:Polygon>, convert to ECEF and get mesh URIs :)
let $meshes := for $geo in $geos
                let $polygons := http:get-node(fn:concat( $geo, '.kml' ))[2])//kml:Polygon
                        for $polygon in $polygons
                        let $ecefs := http:get-node( fn:concat( $geo2ecef, $polygon ))[2]
                                for $ecef in $ecefs
                                return http:post( $geo2mesh, $ecef )

(: Generate XML3D scene :)
<html>
<xml3d ...>
<defs> ... {
for $shaderUri in $shaders
return http:send-request(<http:request href="{data($shaderUri)}" method="get" />)[2]
} ... <defs/>
<view .../> ... {
for $meshUri in $meshes
return http:send-request(<http:request href="{data($meshUri)}" method="get" />)[2]
} </xml3d>
</html>
```

Listing 1: Application logic of *"Hello (3D) World!"* in XQuery

for POSTed kml:Polygons and immediately returns Linked Data containing fresh URIs. These URIs identify xml3d:mesh elements being computed using a Delaunay triangulation backend[20] and can be resolved by Linked Data-driven applications. The service additionally provides a GET method for xml3d:mesh retrieval; an exemplary xml3d:mesh element generated by this service is shown in Listing 5.

```
<mesh type="triangles">
<int name="index">0 1 2 1 3 2</int>
<float3 name="position">0.96 ... -0.13</float3>
<float3 name="normal">0.0 ... 1.0</float3>
<float2 name="texcoord">1.0 ... 0.0</float2>
</mesh>
```

Listing 5: Generated XML3D mesh element

4.4 Renderer and Display

Being focused on the task of conveying complex Linked Data to the user, we aim at rendering and displaying 3D visualisations in the browser. Via the browser, users have the ability to change various parameters that the application designer has provided. The input from the user interface triggers queries to the execution engine which in turn combines content from Linked Data sources and functionality from services into a Web3D scene graph. The Web3D scene graph is handed to the rendering and displaying environment for generating the visual application output. We decided for XML3D as rendering and displaying environment for its clean and comprehensible specification; although we sadly miss support for external references in the current implementation. As future work, we would like to assess alternative Web3D proposals with respect to ease of use, performance and scalability.

[20] http://code.google.com/p/poly2tri/

5. DISCUSSION

We now discuss lessons learned based on the experiences gained during experimentation and put our requirements into perspective.

The success of implementing a Linked Data-driven 3D Web Application crucially depends on the availability of highquality data. In our scenario, we depend on a complete set of links between countries in GADM-RDF and DBpedia. While GADM initially provided some links, both datasets were only sparsely interlinked. Only after completing the linkage we were able to query both datasets in combination.

The size of the geometries in GADM-RDF were another stumbling block. Geometries of countries can reach several MB, and merely transmitting these files takes time. We solved the issue by accessing more coarse-grained versions of the geometries. For the triangulation, which is a relatively time-consuming calculation, we opted for a service which queues the required computation steps and asynchronously processes the input. We use caching throughout (via HTTP's Last-Modified header and a Squid cache[21]) to avoid costly re-computation.

We use SPARQL queries and XQuery programs to access data and coordinate computation, which allows for a concise specification of the various computation steps. We used SPARQL to query the data in combination, and XQuery to invoke functionality, as XQuery engines can be extended with user-defined functions. However, using two separate query engines is suboptimal; an execution environment which natively supports RDF queries and XQuery-like programming constructs would be our more elegant system of choice. Ideally, such an interpreter of data integration programs would run client-side in a web browser, to remove the bottleneck which a centralised query processor represents. Whether XSPARQLis a suitable candidate for such a language is subject to future work.

Finally, the lack of maturity of the deployed Web3D implementation required some workarounds; for example, the XML3D specification allows for providing URIs which the browser would dereference and include in the scene graph. Such functionality would fit our distributed architecture very well, however, current implementation lacks these network lookups, and thus we need to construct a large monolithic file with the scene graph rather than having a scene graph file which can include network-accessible references.

6. CONCLUSION

The overall goal of Linked Data-driven Web3D applications is to enable the end user to query, understand and interact with often complex and incomprehensible Linked Data in more user-friendly and efficient ways. Suitable and coherent Web3D visualisations may help in conveying complex Linked Data, support users in sense making and information exploration and discovery, and eventually provide descriptive, intuitive and effective user interfaces to the Web of Data. To facilitate the specification and implementation of such Linked Data-driven Web3D applications, we provide an architecture blueprint that enables the integration of distributed content and functionality into coherent Linked Data-driven applications with Web3D user interfaces.

7. REFERENCES

[1] W. Akhtar, J. Kopecký, T. Krennwallner, and A. Polleres. XSPARQL: traveling between the XML and RDF worlds - and avoiding the XSLT pilgrimage. In *Proceedings of the 5th European Semantic Web Conference*, ESWC'08, pages 432–447. Springer, 2008.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data The Semantic Web. In *Proceedings of 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC+ASWC 2007)*, pages 722–735. Springer, Nov. 2007.

[3] J. Behr, P. Eschler, Y. Jung, and M. Zöllner. X3dom: a dom-based html5/x3d integration model. In *Proceedings of the 14th International Conference on 3D Web Technology*, volume 1 of *Web3D '09*, pages 127–135. ACM, 2009.

[4] T. Berners-Lee. Linked data. `http://www.w3.org/DesignIssues/LinkedData.html`, 2006.

[5] T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and analyzing linked data on the semantic web. In *Proceedings of the 3rd International Semantic Web User Interaction Workshop*, 2006.

[6] A. Couthures. Compiling xquery code into javascript instructions using xslt. In *XML Prague 2012*, pages 125–139, 2012.

[7] R. T. Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, 2000. AAI9980887.

[8] G. Fourny, M. Pilman, D. Florescu, D. Kossmann, T. Kraska, and D. McBeath. XQuery in the browser. In *Proceedings of the 18th International Conference on World Wide Web*, pages 1011–1020. ACM, 2009.

[9] M. Friendly. Milestones in the history of thematic cartography, statistical graphics, and data visualization. In *Seeing Science: Today*. American Assoc. for the Advancement of Science, Feb. 2008.

[10] Google. O3d. `http://code.google.com/p/o3d/`, 2011.

[11] A. Harth. VisiNav: A system for visual search and navigation on web data. *Journal of Web Semantics*, 8(4):348–354, 2010.

[12] M. Hausenblas. Linked data applications - the genesis and the challenges of using linked data on the web. Technical Report 2009-07-26, Digital Enterprise Research Institute, 2009.

[13] E. Kalogerakis, S. Christodoulakis, and N. Moumoutzis. Coupling ontologies with graphics content for knowledge driven visualization. In *Proceedings of the IEEE Conference on Virtual Reality*, VR '06, pages 43–50. IEEE, 2006.

[14] F. Pittarello and A. De Faveri. Semantic description of 3d environments: a proposal based on web standards. In *Proceedings of the Eleventh International Conference on 3D Web Technology*, Web3D '06, pages 85–95. ACM, 2006.

[15] K. Sons, F. Klein, D. Rubinstein, S. Byelozyorov, and P. Slusallek. Xml3d: interactive 3d graphics for the web. In *Proceedings of the 15th International Conference on Web 3D Technology*, Web3D '10, pages 175–184. ACM, 2010.

[16] S. Speiser and A. Harth. Integrating linked data and services with linked data services. In *Proceedings of the 8th Extended Semantic Web Conference*, pages 170–184. Springer, 2011.

[21] `http://www.squid-cache.org/`