Noname manuscript No.
(will be inserted by the editor)

Linked Data for the International Aid Transparency Initiative

Karel S. Brandt · Victor de Boer

Received: datezz / Accepted: date

Abstract It is widely recognised that the effectiveness of aid can be improved by providing transparent insight into aid activities. The International Aid Transparency Initiative (IATI), a multi-stakeholder initiative that seeks to improve the transparency of aid, has developed an open standard for the publication of aid information. As of 2013, over 150 donors, NGOs and governments have registered to the IATI registry by publishing their aid activities in this XML standard. Taking the IATI model as an input, we have created a Linked Data model based on requirements elicitated from qualitative interviews using an iterative requirements engineering methodology. We have converted the IATI open data from a central registry to Linked Data and linked it to various other datasets such as World Bank indicators and DBpedia information. This dataset is made available for re-use at http://semanticweb.cs.vu.nl/iati/. To demonstrate the added value of this Linked Data approach, we have created several applications which combine the information from the IATI dataset and the datasets it was linked to. As a result, we have shown that creating Linked Data for the IATI dataset and linking it to other datasets give new valuable insights in aid transparency. Based on actual information needs of IATI users, we were able to show that linking IATI data adds significant value to the data and enables fulfilling the needs of IATI users.

Keywords Linked Data · Data Enrichment · Linked Data Applications · ICT 4 Development · Open Development · Aid Transparency Data

K. S. Brandt
VU University Amsterdam
Faculty of Sciences
Amsterdam, The Netherlands
E-mail: k.s.brandt@few.vu.nl

V. de Boer
VU University Amsterdam
the Network Institute
Faculty of Sciences
Amsterdam, The Netherlands
Tel.: +31(0)205988723
Fax: +31(0)205981234
E-mail: v.de.boer@vu.nl

1 Introduction

During the 3rd high level forum on Aid Effectiveness, held in September 2008, the Accra Agenda for Action called for organisations, varying from governments to NGOs, to disclose their information on aid activities in a timely and transparent manner[1]. It was during this same event that the International Aid Transparency Initiative (IATI) was founded. This initiative has been working on creating an open standard for publishing aid activities to make "information about aid spending easier to access, use and understand"[2].

The first organisation to publish aid information in the IATI standard was the UK Department for International Development (DFID) in January 2011 and as of 2013, over 150 organisations have published their aid information. The IATI standard is defined as a set of XML schema files[3]. According to the Organisation for Economic Co-operation and Development (OECD) approximately $500 billion is spent every year on aid activities[4]. For this aid money to be spent as effectively

[1] http://siteresources.worldbank.org/ACCRAEXT/Resources/4700790-1217425866038/AAA-4-SEPTEMBER-FINAL-16h00.pdf
[2] http://www.aidtransparency.net/
[3] http://iatistandard.org/schema/
[4] http://stats.oecd.org/Index.aspx?DatasetCode=TABLE1

as possible, it is crucial to be as transparent as possible [12]. Increased aid transparency could lead to more effective planning by partnering donors, a better way for monitoring performance, a reduced scope for corruption and meaningful insights for domestic tax payers as they can see where their tax money is going.

We believe that creating a Linked Data model based on the IATI model, linking this model to external datasets and creating applications based on this model can add value to the existing IATI dataset, thereby increasing the transparency and efficiency of aid spending. Linked Data is about publishing structured data on the Web and connecting data between different data sources, effectively allowing data in one source to be linked to data in another source [2]. It is the standard for sharing heterogeneous distributed datasets on the Web and is built upon the Resource Description Framework (RDF) in order to create meaningful data nodes and relationship between these nodes. Uniform Resource Identifiers (URIs) identify these nodes or relationships, making it possible to reuse these URIs in other datasets. By reusing existing Linked Data vocabularies, it is possible to compare IATI data to other datasets which use the same vocabularies or to extract information automatically from IATI data by making use of applications which are able to interpret these vocabularies.

Additionally, linking IATI data to other datasets can be of added value, since this allows for a comparison between the datasets or the addition of extra metadata to the IATI data. Examples of datasets that the IATI Linked Data could be linked to are governmental information, financial reports, activity or project data, administrative and geographical data. This paper presents three contributions: 1) a Linked Data model based on the IATI model; 2) the data conversion process and linked datasets for the IATI Linked Data and 3) Linked Data applications we built that exploit this linked data.

For creating a Linked Data model, we focus on setting up the model in a structured manner in order to create a reproducible process for converting that IATI data to Linked Data. The model itself is based on requirements as described in Section 3. Furthermore, since provenance is becoming increasingly more important on the Web of Data [7], we record the provenance of the conversion within the IATI Linked Data model. Finally, we research which external schemas and vocabularies can be used for mapping concepts and relations in the IATI Linked Data model, improving interoperability. This is described in Section 4.

The second topic of this paper concerns the external datasets that will be linked to the IATI Linked Data model. Since the Web of Data contains many data sources, we look for relevant datasets that can be linked to the IATI dataset. Based on user needs as elicitated from qualitative interviews with experts on the IATI domain, we are interested in finding relevant datasets and algorithms which are able to link these datasets to the IATI Linked Data model. This is described in Section 5.

Finally, in Section 6 we research how applications that use the produced IATI Linked Data can show its added value as opposed to Web applications built without the use of Linked Data. The specified user needs will serve as an input for these applications as we aim to fulfill these user needs with the applications. Also, we look at the design choices made in the IATI Linked Data model and find out what their impact on the final applications is. We finally reflect on the generalisation and re-usability of the methodology for modelling, conversion and linking used in Section 7.

2 Background and Related Work

2.1 IATI model

In order to understand the conversion process of the IATI model to a Linked Data model as described in Section 4, we must first explain the contents of the IATI model itself (we have used version 1.03 of this model). The IATI model is divided up into three parts: *activities*, *organisations* and *codelists*, which we detail below.

2.1.1 Activities

Activity files contain data on individual aid activities. However, one XML file can contain multiple activities, each activity indicated by the ⟨iati-activity⟩ element[5].

The top node for each IATI activity, ⟨iati-activity⟩, not only indicates the start of an activity, but also defines some metadata on the file, as well as several default values. As can be seen in the XML snippet below, metadata is specified through the @last-updated-datetime, @hierarchy, @version and @linked-data-uri attributes.

```
<iati-activity hierarchy="1" xml:lang="en" default-
          currency="GBP" last-updated-datetime=
          "2013-01-23T00:00:00" version="1.02"
          linked-data-uri="http://ld.cafod.[..]">
```

[5] a full example of an IATI activity element can be found in an online appendix to this paper at https://github.com/KasperBrandt/IATI2LOD/raw/master/PaperAppendix/online_appendix.pdf

Another important element is the ⟨iati-identifier⟩ element, which uniquely identifies this activity. We use the value of this element in the activity URIs of our Linked Data model to uniquely identify an activity, as can be seen in Section 4.2. Furthermore, most elements have a reference to either the codelist or organisation models. As can be seen from the XML snippet below, the ⟨reporting-org⟩ element, which indicates the organisation that reports on the activity, has a reference to the organisation model through the @ref attribute. Also, the @type attribute indicates the organisation type, which refers to one of the codelists[6]

```
<reporting-org ref="GB-CHC-285776" type="21"> CAFOD
</reporting-org>
```

Most of the activity model consists of elements that either refer to codelists or organisations, or denote a literal value such as a title or description. However, some elements, such as the ⟨conditions⟩ and ⟨result⟩ elements, do not refer to any codelist but represent information that is hard to compare across IATI files. It is interesting to see that these elements are not often used, whereas the elements containing references to codelists or organisations are. A count of various elements and the @linked-data-uri attribute, as gathered from all IATI activity files on July 1st, 2013, sorted by their number of occurrences, can be found in Table 1.

Element	Count
⟨transaction⟩	685,271
⟨policy-marker⟩	384,636
⟨sector⟩	291,365
⟨iati-activity⟩	233,193
⟨recipient-country⟩	187,142
⟨budget⟩	99,031
⟨planned-disbursement⟩	49,575
⟨recipient-region⟩	44,824
⟨condition⟩	15,133
⟨result⟩	11,383
⟨location⟩	6,854
⟨coordinates⟩	2,552
⟨gazetteer-entry⟩	1,200
⟨indicator⟩	0
@linked-data-uri	0

Table 1 Count of the main elements and the @linked-data-uri attribute, as gathered from all IATI activity files on July 1st, 2013, sorted by their number of occurrences

The table shows that a total of 233,193 activities have been specified as of July 1st, 2013, and the most commonly used elements are ⟨transaction⟩, ⟨policy-marker⟩ and ⟨sector⟩. Even more so, the table shows that these elements are often specified multiple times within an ⟨iati-activity⟩ element.

[6] In Section 4.2, we show how free text elements such as these are processed in the IATI Linked Data model.

2.1.2 Organisations

The XML files describing organisations can contain multiple organisations in one file. Each individual organisation is specified according to an ⟨iati-organisation⟩ element. The number of elements contained by the organisation model is less than the activity model and the model contains several elements for specifying the budgets that an organisation has allocated[7].

The organisation model also contains metadata on the XML file within such an ⟨iati-organisation⟩ element through the @last-updated-datetime attribute. However, in comparison to the activity model, no @version or @linked-data-uri attributes can be specified. Furthermore, most elements are similar to the activity model, such as the ⟨iati-identifier⟩ and ⟨name⟩ elements. The organisation model has three elements for the specification of budgets. The total budget for a certain period can be specified, as well as a breakdown per country or organisation in the ⟨recipient-country-budget⟩, ⟨recipient-org-budget⟩ and ⟨total-budget⟩ elements.

Element	Count
⟨document-link⟩	3,957
⟨reporting-org⟩	3,597
⟨iati-organisation⟩	3,525
⟨recipient-country-budget⟩	714
⟨recipient-org-budget⟩	112
⟨total-budget⟩	19

Table 2 Count of the main elements sorted by their number of occurrences

The number of entries in the organisation model is significantly less than the number of entries in the activity model. In Table 2 the exact number of elements is shown. A total of 3,525 organisations have been specified according to the IATI organisation model and the elements that are most frequently used are ⟨document-link⟩ and ⟨reporting-org⟩.

2.1.3 Codelists

The codelists can be seen as an IATI thesaurus, allowing the IATI users to specify standardised elements within their activity or organisation files and making it possible to compare these elements throughout the

[7] An example XML file for the "Department for International Development" organisation can be found in the online Appendix https://github.com/KasperBrandt/IATI2LOD/raw/master/PaperAppendix/online_appendix.pdf

complete IATI datamodel. In total, 35 codelists are specified in the IATI model, as is shown on the IATI codelists website[8]. In general, the following elements are attributed to a codelist: ⟨code⟩, ⟨name⟩, ⟨description⟩, ⟨language⟩, ⟨category⟩, ⟨category-name⟩, ⟨abbreviation⟩ and ⟨category-description⟩ . However, most codelists are not categorised and therefore only consist of a code, a name and sometimes a description.

When looking at the activity and organisation models, we see that the codelists are always referenced by their codes, usually through a @code or @ref attribute within an element. However, as was stated in Section 2.1.1, most elements allow a name to be specified as a text value in addition to the codelist code.

A complete overview of the number of codelists and their categories is found in Table 3, showing that the IATI codelist model contains a total of 1,490 codes. Most of these codes originate from the organisation codelist, which specifies most organisations within the IATI model by their own code.

Element	Count
⟨code⟩	1,490
⟨abbreviation⟩	415
⟨description⟩	272
⟨category⟩	269
⟨category-description⟩	127
⟨codelist⟩	35

Table 3 Count of the main elements of the IATI codelists

This standardised setup of defining codes within the IATI model allows us to reuse these codes in our Linked Data model and represent it as a thesaurus, as is described in Section 4. In the next section, we describe the metadata of the IATI files.

2.1.4 Metadata

Metadata on the IATI activity and organisation files is not mentioned on the IATI website and therefore we were surprised to find additional information when looking up the XML files. Besides the sparsely available metadata from the IATI files themselves, such as the @version and @last-updated-datetime attributes within the ⟨iati-activity⟩ and ⟨iati-organisation⟩ elements, a number of elements are provided through the IATI API[9]. Through this API, a JSON file can be looked up, of which an example is shown in the online appendix.

All of the data as shown in the example JSON file are standard and provided for each XML file that is looked up through the use of this API. Since no description is given for the fields within this file on the IATI website, we interpreted the results of this file ourselves. Some interesting fields present in these JSON files are: *maintainer*, *author*, *download_url*, and *data_uploaded*. These fields are used to construct the provenance model, as described in Section 4.

2.2 Linked Data Model

Governmental or organisational open data are usually not published as "5 star linked data"[10], but rather as CSV or another non-proprietary format. The same can be said about IATI data, which is currently published in XML and can be retrieved in both XML and JSON formats from the IATI registry[11]. This allows anyone to reuse the available open data, but it does not allow for linking the IATI data to external datasets in the Linked Open Data (LOD) cloud. Therefore, we will show the potential of non-5 star datasets by converting the IATI data to 5 star Linked Data.

Related work on this topic is mostly focused on transforming open governmental data to Linked Data. Particularly the http://data.gov and http://data.gov.uk websites, respectively originating from the United States and the United Kingdom, have been working on creating a Linked Data portal for the governmental data of their country. The efforts and pitfalls that have been encountered while working on http://data.gov.uk have been documented by Shadbolt[15]. The main worries as posed in this article are the confidentiality of the data, the interpretation of data, the quality of information and lost value of information through the conversion process. Even though these worries are mainly focused on the governmental domain, they are relevant for our domain as well. Confidentiality of the data are partly disregarded since the IATI data are distributed under an open license and therefore we assume all data are available for reuse. Interpretation of the data, however, is a main concern that remains intact for our dataset. Anyone can create an application based on the IATI Linked Data model, but the way these applications are interpreted is completely up to the users of the application. Furthermore, the quality of the information provided in the IATI activity or organisation files is another point of worry. When looking at the organisation files, some organisations have been specified according to an ⟨organisation⟩ element, instead of

[8] http://iatistandard.org/103/codelists/. An example of the XML file for the "Aid Type" codelist is shown in the online appendix.

[9] http://iatiregistry.org/registry-api

[10] http://5stardata.info/

[11] http://iatiregistry.org/

an ⟨iati-organisation⟩ element. Diverting from the standard makes the conversion process significantly harder. Also, the IATI standard allows for specifying contradictory information. One could, for example, state that the ⟨recipient-country⟩ of an activity is Afghanistan, but on the other hand specify a location in a different country within the ⟨location⟩ element of the activity. Finally, losing information through the conversion process of the XML files to Linked Data is a main concern of ours as well. Since the information as specified IATI Linked Data model should not have a gap when compared to the corresponding XML files, information loss should be prevented. Creating a gap between the two models might cause IATI users to distrust the IATI Linked Data model, which could result in the dismissal of the model itself.

Other related projects include GovTrack55[12], a civic project that collects data about the U.S. Congress and republishes the data in XML and as Linked Data. Also, a project by Goodwin et al. [6] used linked geographical data to enhance spatial queries on the administrative geographic entities in Great Britain. Another project that publishes Linked Data and provides open source tools for participating in collaborative Linked Data production is the TWC LOGD Portal [5].

Moreover, related work on converting IATI data to Linked Data can be viewed at the Linked Data wiki page of the IATI standard[13]. The site provides an initial workflow for converting IATI data into Linked Data and upload the data to a triple store as a prototype. However, the XSLT script for converting the IATI data considers only a subset of the elements within the IATI files, whereas it would be preferable to convert all available elements, and does not provide a robust solution for converting the IATI data to Linked Data. From this background and related work we take into account the worries of converting an open dataset to Linked Data in the process of converting the IATI data to Linked Data, as shown in the descriptions of the requirements in Section 3.

2.3 Linking Datasets

Even though the number of links between the RDF datasets is constantly growing, it still only comprises a small fraction of the number of links that could in theory be set [13]. For finding the links that could possibly be of relevance to a dataset, (semi-)automated link generation approaches are often used. Depending on the domain, techniques for generating links can vary. Some domains have commonly used standards, such as ISBN or ISSN numbers in the publication domain, making linking the concepts in various datasets straightforward by using the same identification schemas. Other datasets, without a standardised identification schema, generally use the similarity of entities within both datasets to generate their links [1].

For generating relevant links, several link generation frameworks are available. One of these frameworks is Silk [9,16] which has its own flexible configuration language, the Silk Link Specification Language (Silk-LSL), to specify the conditions concepts or relations must fulfill in order to be interlinked. Links are generated by making use of similarity values for each URI pair and a threshold which determines whether the pair should be linked.

The problem we address regarding the IATI data is to research which external datasets are relevant to the IATI dataset and which algorithms are best suited for linking these datasets to the IATI dataset. As described in Section 2.1.3, the IATI model consists of a number of codelists which contain standardised items such as ISO codes for countries and standard codes for currencies. Some of these codelists can be modelled directly to existing datasets, whereas we might be able to match others by making use of similarity functions.

Related work on generating links again mostly focusses on Government Linked Data (GLD). Especially the efforts of linking the datasets of the http://data.gov website are documented well [4]. Besides using standard vocabularies such as FOAF and Dublin Core (DC)[14] in their datasets, they have been linked to other datasets in the LOD cloud. A semi-automated approach and Semantic MediaWiki[15] were used in order to create links to DBpedia and GeoNames for example.

As for related work on IATI Linked Data, not much work had been done, since only a prototype of the Linked Data model was available so far. However, in this prototype links were explored between R4D publication records and the DFID projects that fund them [3]. In our paper we did not use any of the advanced linking techniques such as the Silk framework. Instead, we created our own algorithms using simple requests to webservices and SPARQL queries to external triple stores for matching the codelists of the IATI Linked Data model directly to the external datasets. The datasets that are matched to the IATI Linked Data model and the algorithms that are used in the matching process are described in Section 5. In the next section we describe the background of Linked Data applications and the related work relevant to the IATI model.

[12] http://www.govtrack.us
[13] http://wiki.iatistandard.org/tools/linked_data/start
[14] http://dublincore.org
[15] http://semantic-mediawiki.org

2.4 Linked Data Applications

The data contained by the LOD cloud keeps growing with publishing Linked Data becoming increasingly popular. However, in order for the data to become relevant for the general public, applications will have to be developed to give a clear insight into the Linked Data. At present the efforts of creating Linked Data applications can be broadly classified into three categories[1]: a) Linked Data browsers, b) Linked Data search engines and c) Domain-specific Linked Data applications which use integrated datasets for a specific user-task. An example of such an application is the Supreme Court Justices Decision Making application[16]. We here describe examples of Domain-specific applications.

Related work concerning IATI applications is found in the form of general applications that provide visualisations or overviews of the IATI data itself, without linking to external datasets. We have compiled a list of currently available applications based on IATI data:

Aid Transparency Tracker which allows all aid donor schedules to be put together into a single application. This enables an assessment of their overall ambition; to compare fields across schedules; to show the publication of fields over time from different donors, and provides CSV, JSON/JSONP and iCal feeds for each donor. Available at http://tracker.publishwhatyoufund.org/.

AidView AidView is a prototype demonstrating how detailed and timely data on aid activities shared through IATI can be made accessible to a wide range of people through tablets and modern Web browsers. Available at http://www.aidview.net/.

Akvo Openaid Similar to AidView, Akvo Openaid also provides the possibility to explore the IATI data and filter on results or a range of indicators, such as publishers, budgets, countries, regions and sectors. Available at http://openaidsearch.org/.

Development Tracker This site provides a window on British aid spending, presenting users with detailed information on international development projects funded by the UK government. It draws on data published by DFID, other UK government departments, and some of DFID's delivery partners (NGOs that receive and spend UK aid funds). Available at http://devtracker.dfid.gov.uk/.

IATI Explorer The IATI Explorer allows the user to select aid activities published to the IATI Registry and then further filter and browse the data through an interactive interface. It is possible to access a summary and full-view of each activity, and get the underlying data as XML, JSON or CSV. Available at http://iatiexplorer.org/.

International Aid Transparency Initiative Data This application, by Aidan Berentsen, makes it possible to filter available data by country, sector or donor, and to explore what is contained in the data over time. It won the Data Visualisation Competition by Google and The Guardian[17]. Available at http://iatid.com/.

The Section 6, we show that a Linked Data application for IATI data provides additional value to the existing applications through its linked external content.

3 Requirements

Creating a Linked Data model for the IATI model, selecting relevant datasets to link to the IATI Linked Data model and developing Linked Data applications all require extensive domain knowledge of the IATI standard and the needs of the IATI users. We have therefore decided to adopt a requirements engineering approach in which we take qualitative interviews for extracting domain knowledge from a user or domain expert. The iterative requirements engineering approach we have employed is based on the one described by Loucopoulos and Karakostas [11].

Starting out with the problem domain, the IATI standard[18], we first selected to interview the IATI user experts of the OIPA framework[19], which provides an API for the IATI datasets, to gain an initial overview of the IATI model. The goal of this meeting was to retrieve user needs and technical requirements from these domain experts. Furthermore, we opted to have another meeting with a member of the IATI working group in order to collect more qualitative information about IATI and refine the previously gathered user needs and technical requirements. In the following sections, we explain the requirements and user needs that were elicitated from these interviews. These serve as the guidelines for setting up the Linked Data model, as described in Section 4.

3.1 Model Requirements

The requirements of the model are a combination of requirements elicitated from the interviews with the IATI

[16] http://data-gov.tw.rpi.edu/demo/stable/supremeCourt/demo-10016-portal.html

[17] http://www.guardian.co.uk/data/series/google-data-visualisation-competition

[18] http://iatistandard.org/

[19] http://oipa.openaidsearch.org/

experts and general design principles related to the creation of Linked Data models. A general design principle that we adhere to is that all data from the IATI XML files should be preserved and converted to RDF, even when non-IATI elements are used. As mentioned in Section 2.2, information loss from a conversion to the RDF format causing a gap between the information that was specified by the publisher and the information in the Linked Data model could make the publishers mistrust the new model. Therefore we decided that all elements, even when an element is not specified as a standard IATI element, should be taken into account when converting an IATI XML to RDF. Unfortunately it is not always possible to convert an element, since required fields are missing, in which case the element is ignored.

Another key point as elicitated from the interviews is that the structure of the IATI URIs, as used in the Linked Data model, should be similar to the IATI model itself. By doing so, the IATI Linked Data model is instantly recognisable for the IATI users and consistent with the IATI model. The same holds for the names of the elements and attributes used in the IATI model and IATI Linked Data model. These should be kept similar in order to keep both models consistent. Besides these design principles, we have created six requirements in order to create a structured process for converting all information of the IATI files, as is shown in the list below:

1. Underlying elements of an XML element are modelled as a new named node and a relation is created from the upper element to the underlying element. Attributes and text values of an element are modelled as literals with a direct relation to the element itself.
   (a) **Exception:** When an underlying element only contains a text value or just one attribute, this value or attribute of the underlying element is modelled as a literal and directly related to the upper element.
2. The relationship between an element and either an underlying element or an attribute takes the form of 'upper-element' - 'underlying-element' or 'element' - 'attribute'. Text values of an element are related to their element through the *rdfs:label* property.
3. All nodes have a *rdf:type* property to denote the type of the node.
4. Blank nodes are not allowed.
5. In case a URI cannot be given a unique identifier or reference, a hash is incorporated in the URI.
6. The number of nodes is kept to a minimum.

We start out by stating that a new named node is created for every underlying element. For attributes and text values of an element a relation is created to the element itself, with the values of the attribute or text modelled as literals. This will keep the structure of the IATI files similar to the Linked Data model. However, there is an exception to this requirement: if the underlying element only consists of a text value or just one attribute, no extra node is needed and therefore a direct relationship between the text value or attribute of the underlying element and the upper element is created, modelling these values as literals. This results in the following specification for the first requirement.

This requirement has two other exceptions, which are specific to the IATI model. First of all, identifiers are often specified in as a @ref or @code attribute of an element. In order to create unique URIs for each element, identifiers are appended to the URI of its element. Also, when a IATI codelist is specified, the URI specific to that codelist is used instead of modelling the value as a literal. This allows for a general approach to modelling codelist URIs.

Furthermore, in order to stick to the same structure as the IATI model, the relationship between an underlying element and its upper element is in the form of 'upper-element'-'underlying-element'. The relationship between an element and its attributes is in this form: 'element'-'attribute' and the text values of an element are related to their element through the *rdfs:label* property. The snippet of IATI XML below illustrates these first two requirements.

```
<iati-activity>
 <iati-identifier>GB-CHC-285776-CHA024</iati-identifier>
 <transaction ref="42737">
   <tied-status code="5"/>
   <value value-date="2010-10-01">700000</value>
 </transaction>
</iati-activity>
```

In this example an activity identifier is specified in the ⟨iati-identifier⟩ element, which is used to identify all URIs concerning this activity. Since this element only contains a text value, the element will be directly linked to its upper node (requirement 1a). However, since it is also the identifier which uniquely identifies the activity, it is appended to the URI of the activity as well. This creates the activity URI iati:activity/GB-CHC-285776-CHA024[20].

Also, a transaction of the activity – containing a reference number for the transaction – is specified as an underlying element of the activity. This creates a relationship between the activity element and its underlying transaction element through an iati:activity-

[20] The namespace 'iati:' will be used in this document. In practice, we have used the http://purl.org/collections/iati/ as a namespace for the Linked Data model. Also, in this case, 'iati-' is stripped from the element.

transaction property (requirement 2). Thus a first RDF triple[21] that can be extracted from the XML snippet is:

```
iati:activity/GB-CHC-285776-CHA024
iati:activity-transaction
iati:activity/GB-CHC-285776-CHA024/transaction/42737.
```

Furthermore, when looking at the underlying elements of the ⟨transaction⟩ elements, the ⟨tied-status⟩ element is specified with a @code attribute. Since this element only has one attribute, we can directly link the element to the ⟨transaction⟩ element (requirement 1a). The @code attribute is a reference to the "Tied Status" codelist, which has the general form of *iati:codelist/ TiedStatus/ [code]*. This creates the following triple:

```
iati:activity/GB-CHC-285776-CHA024/transaction/42737
iati:transaction-tied-status
iati:codelist/TiedStatus/5 .
```

Requirement 3 states that each node has a *rdf:type* property to denote the type of the node. This requirement does not originate from the interviews, but we found it useful during the creation of SPARQL queries. Another requirement is to disallow blank nodes in the model, since it is impossible to set external RDF links to a blank node, and merging data from different sources becomes much more difficult when blank nodes are used [8]. Furthermore, updating triples in a triple store becomes harder when not all nodes are named and the experts in the IATI technical group advised us not to use blank nodes. Instead, when a URI cannot be given a unique identifier or reference, a hash is used in the URI. This hash will be based on underlying elements or attributes of the element. It varies per element which underlying elements or attributes the hash is based on. However, the required parts of an element are always required for the hash as well and if these are missing, the element is disregarded.

At the same time, we have tried to keep the number of nodes (e.g. a seperate URI for a transaction is considered a node) to a minimum in order to keep the number of triples as low as possible. The rationale behind reducing the number of nodes is to keep the model simple and prevent the creation of unnecessary nodes in the model. Reducing the number of nodes will also reduce the number of clauses within SPARQL queries performed on the IATI Linked Data model. In combination with the first two requirements, this has led to a number of design choices during the process.

If we take a look at the above sample XML snippet again, according to requirement 1, a new node should have been created for the ⟨value⟩ element, related to the ⟨transaction⟩ node. However, in order to reduce the number of nodes (requirement 6), we have opted to create an *iati:transaction-value* and *iati:transaction-value-date* property directly linked to the transaction node:

```
iati:activity/GB-CHC-285776-CHA024/transaction/42737
iati:transaction-value "700000" ;
iati:transaction-value-date "2010-10-01" .
```

Furthermore, since all nodes should have a *rdf:type* property to denote the type of the node (requirement 3), we can identify two such cases: an activity and a transaction. Therefore two extra triples are created, both having a *rdf:type* property to respectively an *iati: activity* and *iati:transaction* object:

```
iati:activity/GB-CHC-285776-CHA024 rdf:type
iati:activity .

iati:activity/GB-CHC-285776-CHA024/transaction/42737
rdf:type iati:transaction .
```

3.2 User needs

The IATI users can be roughly categorised in four different categories based on the open personas as put together by AidInfo[22]:

Funders Funders want to know where the money of their organisation is spent, but also where the money of other funders, preferably in their own sector, is going. This allows for better judgement as to where their investment are needed most.

Governments Governments of developing countries mostly want to know how much money is spent in their country and what budgets and planned disbursements for their country are, possibly comparing themselves to other countries.

Locals Locals residing in developing countries are interested in the organisations and projects going on in their immediate local area. It is also of interest to them who is funding these projects.

Public The public is interested in a wide variety of topics. One likely topic is to see the spendings of their own government, so that they can see where their tax money is going.

Three IATI experts have been interviewed, both from the technical side of IATI as well as the implementing side. One of the experts had been working on a list of user needs together with other representatives from the field. This list can be viewed in the online appendix. From this list, we have extracted the following needs to focus on in this paper, since they allow us to show that

[21] We here use Turtle notation listing subject, predicate and object: http://www.w3.org/TeamSubmission/turtle/

[22] http://www.aidinfolabs.org/category/tools/people

linking the IATI Linked Data model to external datasets can fulfill actual information needs which cannot be fulfilled with just data from the IATI dataset:

1. In total, how much does a given country receive in aid?
2. A comparative index of aid versus the Human Development Index.
3. What is the geographic location of a project? How much aid went to a given province, constituency or village?
   (a) Is the aid spent in places where the need is highest? Is it well distributed across the country?
   (b) Can we attribute sub-national breakdowns for aid so we can see how much goes to different parts of recipient countries?
4. How does violent conflict in recipient countries affect aid activities?
5. How does aid spending as registered in the IATI standard compare to World Bank indicators?

The first item does not need external data and can be fulfilled with just IATI data. However, we did select this need because it can be easily realised as the start of an application. For the other items connections to external data sources are required. The last user need was not extracted from the list of user needs. Instead, it was expressed as one of the items OIPA is working on in the interview with the IATI user experts of the OIPA framework. In Sections 5 and 6 we show how the linking and application are shaped by these user needs.

4 Linked Data model

4.1 Approach and Methodology

As is described in Section 3, the Linked Data model is based on the elicitation of the requirements and the IATI standard from qualitative interviews with IATI user experts. During the creation of the model, feedback was requested to experts of the technical IATI group in order to validate the model and the setup requirements. Based on this feedback, the model and the requirements have been adjusted or updated in an iterative fashion. Additional feedback was requested from different channels as the model was published on relevant groups on LinkedIn, as well as Linked Data experts at the VU University Amsterdam because of their experience with Linked Data and provenance models. The model is evaluated based on this feedback, as is described in Section 4.4. Next to creating the Linked Data model itself, the properties and concepts of the IATI Linked Data model are linked to existing vocabularies. In order to so, we have evaluated every concept and property within the IATI model and looked for suiting standard vocabularies which can be mapped to these relations or concepts.

4.2 Results

Complete textual descriptions and visualisations of both the IATI model and the Linked Data model can be found at http://iati2lod.appspot.com/model.

4.2.1 IATI activities

The start of converting an IATI activity is the activity identifier, as specified in the ⟨iati-identifier⟩ element[23]. If this element is missing the activity is not converted, since it is a required element according to the IATI standard. If the element is specified, the conversion of the activity is started. Besides the *iati:activity/[identifier]* URI (requirement 1a), it is also stated that the activity is of *rdf:type iati:activity* (requirement 3). An exception from the requirements is the ⟨title⟩ element, for which the *rdfs:label* property is used instead of a *iati:activity-title* property as would be expected from requirement 2. The reason for doing so is because the ⟨title⟩ element fits the *rdfs:label* property according to Semantic Web standards. For example, the first triple in the example below is derived from the ⟨title⟩ element:

Two attributes of the activity element have a direct relationship to the activity URI: @hierarchy and @linked-data-uri. The @hierarchy attribute is modelled according to requirement 2 using the *iati:activity- hierarchy* property. An exception is made for the @linked-data-uri attribute, which has the *owl:sameAs* property, since this element contains a link to additional information on the same activity. According to requirement 1a, only the value of one other underlying element is linked directly to the activity URI: ⟨website⟩. Containing just the URL of the activity's website, it is related to the activity node through the *iati:activity-website* property.

Because of requirement 6, a different approach is chosen for the ⟨activity-date⟩ element. In order to keep the number of nodes to a minimum, each @type attribute of the ⟨activity-date⟩ element has its own property: *iati:start-actual-date*, *iati:end-actual-date*, *iati:start-planned-date* and *iati:end-planned-date*. The corresponding text values of the elements are modelled by adding '-text' after each corresponding property. This results in the following three triples when converting the example XML from the online appendix[24].

[23] 'iati-' is stripped from this element.

[24] We did not further model datetimes as xsd literals or model them as resources. Although they could be used for

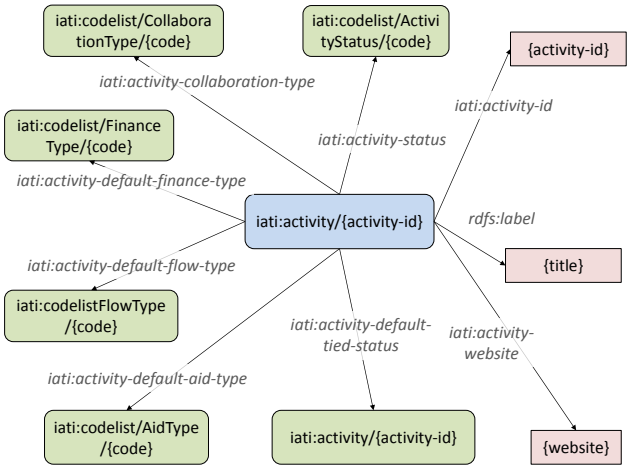

Fig. 1 Visualisation of the relations directly mapped to the activity URI

```
iati:activity/GB-CHC-285776-CHA024
 rdfs:label "Emergency support to individuals and \
     households affected by food insecurity in Chad" ;
 iati:start-actual-date "2012-07-12" ;
 iati:end-planned-date "2013-05-14" .
```

Furthermore, several codelist elements have a direct relationship to the activity URI using a codelist URI. Starting with the ⟨activity-status⟩, it is linked using the *iati:activity-status* property and *iati:codelist/Activity Status/[code]* as an object. The other underlying elements containing a codelist can be seen in Figure 1.

However, most underlying elements do not have a direct relationship to the activity URI and require an extra node. The ⟨reporting-org⟩ and ⟨participating-org⟩ elements, for example, need an extra node for specifying their underlying elements and attributes. According to requirement 3, these nodes should have a *rdf:type* property to denote the type of node. Since both elements indicate an organisation, they are specified as an *iati:organisation*. The text values of the elements are modelled using a *rdfs:label* property (requirement 2). The @ref, @type and @role attributes all have a relation to the organisational node in the form of 'element'-'attribute' in accordance with requirement 2. But since the @ref attribute is 'strongly recommended' by the IATI model, it sometimes is missing. In these cases we use the text value of the element as an input for creating a hash (requirement 5). When no reference or name (text value) is specified, the element is ignored.

⟨other-identifier⟩ is an element which has a text serving as an identifier, so the text value of the element is embedded in the URI in order to uniquely identify this node. In addition, the text value also has a relationship to this node using the *rdfs:label* property. The @owner-ref attribute is allocated to a codelist and modelled according to requirement 2. The @owner-name attribute is modelled according to requirement 2 as well, resulting in the *iati:other-identifier-owner-name* property. In addition, the *rdf:type* of the node is *iati:identifier* (requirement 3).

An element which occurs underneath other elements as well is the ⟨description⟩ element. In all cases the description is modelled in the same way: since it contains both a description in the text value of the element and has a @type attribute, an extra node is created for each description (requirement 2). The node contains a hash (requirement 5) based on the description and the description itself is modelled using a *iati:description-text* property (requirement 2). In this case we did not employ the *rdfs:label* property since the description field could contain a long description which we did not find suitable for this property.

The ⟨contact-info⟩ element is completely modelled according to requirement 2. Since multiple elements of this kind can be specified, a hash is placed in the URI of the ⟨contact-info⟩ element (requirement 5), based on all the text values of all underlying elements. Furthermore, it is of *rdf:type iati:contact-info* (requirement 3).

Two very similar elements are the ⟨recipient-country⟩ and ⟨recipient-region⟩ elements. Both are modelled completely according to requirement 2. The text values of these elements have a relation to their element through a *rdfs:label* property, the @percentage attribute is modelled as a literal and the @code attribute corresponds to respectively a country or region codelist. The @code attribute is used as a reference within the URI of the ⟨recipient-country⟩ or ⟨recipient-region⟩ elements as well, since it is a required attribute by the IATI model. If not specified, the element will be ignored.

The first element which is more extensive is ⟨location⟩. Based on requirement 5, an extra node is created with a hash which is based on the ⟨description⟩, ⟨administrative⟩, ⟨name⟩, ⟨coordinates⟩ and ⟨gazetteer-entry⟩ elements. If not one of these elements is specified, the node is ignored. All elements are modelled as could be expected from requirement 2, except for ⟨name⟩ and ⟨coordinates⟩. The ⟨coordinates⟩ element has three attributes: @latitude, @longitude and @precision. All have a direct relationship to the location URI in order to prevent another node for the ⟨coordinates⟩ element, thus reducing the number of nodes according to requirement 6. The ⟨name⟩ element is modelled through a *rdfs:label* property. This results in a total of 14 triples for the ⟨location⟩ element in the example XML of the online appendix.

⟨sector⟩ and ⟨policy-marker⟩ are two special elements, since they are the only elements using special codelists.

further linking, the user needs did not call for these types of date-time resources, nor did we find them in the linked datasets. At a later stage these could be further enriched.

The @vocabulary attribute is used in the codelist URI, as well as the @code attribute. The URI for a sector is built up as follows: *iati:activity/[activity-identifier]/ sector/[vocabulary]/[code]*. Other than that, these elements are modelled in accordance with requirement 2.

In the ⟨budget⟩ and ⟨planned-disbursement⟩ elements, requirement 6 is used to prevent the forming of extra nodes. This leads to direct relations from these elements to their @type attribute and the values of the ⟨value⟩, ⟨period-start⟩ and ⟨period-end⟩ elements. The @iso-date attribute and text values of the ⟨period-start⟩ and ⟨period-end⟩ have a direct relation to the ⟨planned-disbursement⟩ or ⟨budget⟩ elements. Since these elements also appear in the organisation model (see 4.2.2), their properties have taken the more general form of *iati:start-date* and *iati:end-date* for the @iso-date attribute, making mapping to external vocabularies (see 4.2.5) less complex. The corresponding text values of these elements are related to the ⟨budget⟩ and ⟨planned-disbursement⟩ nodes by adding *-text* to the properties of the @iso-date attribute.

The ⟨related-activity⟩ element has a slightly adjusted property, since the relationship of the activity URI to this element is specified using the *iati:related-activity* property. It relates to another activity, therefore the identifier specified as text is used in the following way within the related-activity URI: *iati:activity/[identifier]*.

The remaining elements ⟨transaction⟩, ⟨document-link⟩, ⟨conditions⟩ and ⟨result⟩ are specified as can be expected from requirements 1, 2, 3 and 5. For example, the hash of the ⟨conditions⟩ is based on the text values of the element, whereas the hash of the ⟨results⟩ element is based on both the ⟨title⟩ and ⟨description⟩ elements. Finally, the contents of the ⟨legacy-data⟩ element are unknown and can differ per activity. Therefore this element will always be modelled as can be expected from the requirements.

The IATI activity model is converted to Linked Data by making use of structured requirements as specified in Section 3. However, in specific cases exceptions to these requirements have been made in order to create a fitting solution for bringing the model into practice.

4.2.2 IATI organisations

In comparison to the activity model, the IATI model for organisations is less extensive. Even more so, most of the elements in the organisation model overlap with the activity model. The ⟨iati-identifier⟩, ⟨reporting-org⟩ and ⟨document-link⟩ elements, for instance, are almost exactly the same and will be treated the same as they are in the activity model, the only difference being the default URI for organisations: *iati: organisation/[organisation-id]*. An exception to the requirements is made concerning the ⟨name⟩ element. Instead of using the *iati: organisation-name* property, as would be expected from requirement 2, the *rdfs:label* property is used, similar to the ⟨title⟩ element in the activity model.

The remaining three elements, ⟨recipient-org-budget⟩, ⟨total-budget⟩ and ⟨recipient-country-budget⟩, are similar to the ⟨budget⟩ element of the activity model. Their use of hashes (requirement 5) are all based on their ⟨start-date⟩, ⟨end-date⟩ and ⟨value⟩ elements.

The ⟨recipient-org-budget⟩ element has an extra element for specifying the recipient organisation, which has direct relationship to the ⟨recipient-org-budget⟩ element through the *iati:recipient-org* and *iati:recipient-org-ref* properties in order to reduce the number of nodes in accordance with requirement 6. The first property relates to a codelist for organisations and the latter shows the actual @ref attribute through a standard literal. The ⟨recipient-country-budget⟩ works in the same way for the specified country or countries in the ⟨recipient-country⟩ element. As a result, the three classes defined in the ontology of the organisation RDFS model are a subset of the entities used in the activity model.

After creating the activity model, we were able to reuse most of this work for the organisation model since the elements in these two models are mostly overlapping. Since this model is a lot smaller, we did not have to divert from the requirements as often as for the activity model. Only the ⟨name⟩ element has been modelled using a *rdfs:label* property as an exception. In the next section we describe the codelist model, which does differ from the activity and organisation models since it contains only standard values and acts as a thesaurus for the complete IATI model.

4.2.3 IATI codelists

Creating a model for the IATI codelists works in a generic way. The difference between the codelists is that some codelists are categorised, whereas others do not have a category, and the fact that some codelists have a description or an abbreviation where others do not.

The general method for converting IATI codelists to a Linked Data model starts with the name of the codelist. We take the 'Aid Type' codelist as an example[25]. The base URI of this codelist is *iati:codelist/ AidType* having the *rdfs:label* "AidType". For example, the URI for codelist code "A01" becomes *iati:codelist/ AidType/A01*. Since the Aid Type codelist has categories, descriptions and also descriptions for its categories, we can go all the way down the chart. The code itself is

[25] see online appendix

linked to the codelist by means of a *iati:code* property. The name of the codelist code has a relationship to the codelist code URI through a *rdfs:label* property and the description of the codelist is modelled through a *rdfs:comment* property. The codelist code has a relationship to its category using the *iati:has-member* property and the category is part of the codelist category URI being *iati:codelist/AidType/category/A* in this case. And just like the codelist itself, the code, name and description respectively have a *iati:code*, *rdfs:label* and *rdfs:comment* property to relate their values to the codelist category. Finally, each node has its own type, being *iati:codelist* for the general codelist node, *iati:codelist-category* for categories of the codelist and *iati:codelist-code* for the instances within the codelist.

By converting the IATI codelist model according to the structured method as described above, we created a general thesaurus for all codelists. In general, we did not adhere to requirement 1 by creating new nodes for each category. If we look at the example XML in the online appendix, we see that the ⟨category⟩ and ⟨category-name⟩ elements are underlying elements directly under each ⟨AidType⟩ element for example. We created a separate entity for the categories in these cases in order to distinguish between the codes and their categories. The requirements as specified in Section 3 are mostly useful for converting the activity and organisation model. Since the codelist model can be seen as a thesaurus with a standardised setup, a different approach suited this model better.

4.2.4 Provenance model

In our approach to modelling the provenance, we focused on the metadata of the XML files. Provenance on this level is useful for knowing when a file was last updated or by whom it was updated. An extensive list of metadata for each file is collected when retrieving the data through the IATI JSON endpoint[26]. An example of this JSON data can be found in the online appendix.

Because this metadata relates to an IATI XML file which can contain multiple IATI activities, and usually does, we had to find a way of specifying the metadata once for all these activities. Therefore we introduced named graphs for each XML file, having the *rdf:type* property specifying an *iati:graph*, and so the provenance is formulated on graph level. Each named graph thus contains all activities or organisations that are specified in an IATI XML file and through this named graph the provenance for each of the activities or organisations can be queried. The implementation details of these named graphs are described in Section 4.3.

[26] http://www.iatiregistry.org/api/rest/dataset/[dataset]

In general we have created a relationship for every entry of the JSON data directly to the named graph, with two exceptions: an extra node is created for the maintainer and author entries. Even though it is only possible to specify one maintainer and one author in the JSON file, an extra node is created for these elements in order to specifically map these to the FOAF vocabulary (see 4.2.5). A *rdf:type* property related to both these nodes is created, being *iati:maintainer* and *iati:author* respectively. Furthermore, their name and email both have their own relationship to the extra node using the *iati:maintainer-name*, *maintainer-email*, *author-name* and *author-maintainer* properties.

As we show in the next section, we have provided additional metadata showing information on the conversion process of the XML files to RDF. Useful for the technical background on the conversion, this allows the user to see which scripts were used to convert IATI XML files to Linked Data and when this conversion took place. Examples of the JSON and provenance files can be found on http://iati2lod.appspot.com/model/examples.

The resulting dataset structure is also described with the VoID vocabulary[27], where generic metadata about the dataset as a whole is provided (including provenance data). However, currently VoID does not allow for fine-grained definition of subsets of data using named graphs[28]. An alternative to the current model of using named graphs and PROV would be to define VoID descriptions for each of the named graphs, however, we believe that most of the VoID metadata is only interesting at the IATI-level and not at the level of the individual datasets.

4.2.5 Vocabularies

In addition to using the standard IATI vocabulary, we use the PROV vocabulary[29], a W3C standard, for mapping provenance. This created an extra triple for each named graph having the *rdf:type* of *prov:Entity*[30]. A *prov:Entity* can be derived from a different *prov:Entity* and can be generated by a *prov:Activity*. Therefore, *prov:wasDerivedFrom* and *prov:wasGeneratedBy* properties are added to the provenance model. The former property points to the source of the IATI XML file, whereas the latter creates an extra URI node with the *rdf:type* of *prov:Activity*. This node has a relation to the IATI activities that are distilled from the source file using a *prov:generated* property, the start date of the file

[27] the VoID file is found at http://tinyurl.com/l8hvd97
[28] http://www.w3.org/TR/void/
[29] http://www.w3.org/TR/prov-primer/
[30] The PROV namespace is http://www.w3.org/ns/prov#

generation through a *prov:startedAtTime* property and the script that was used by means of a *prov:used* property.

Besides using the PROV model for provenance, several other vocabularies are mapped to the Linked Data model using *rdfs:subClassOf* and *rdfs:subPropertyOf*. A complete overview of all mappings is given in the model descriptions on http://iati2lod.appspot.com/model. The list of vocabularies that are used can be found below.

SKOS The Simple Knowledge Organisation System[31] is a vocabulary for organising standard thesauri. Therefore the IATI Linked Data codelist model has extra triples linking its classes and properties to this vocabulary in order to create a standardised hierarchy.

DCT The Dublin Core Terms vocabulary[32] contains classes and properties for dates, descriptions, languages, identifiers and document formats. Since these classes and properties are abundantly present in the IATI model as well, many relationships are created from the IATI model to the properties and classes in the DCT vocabulary.

FOAF The Friend Of A Friend[33] vocabulary is linked to the *iati:contact-info* class in the activity model, as well as the *iati:author* and *iati:maintainer* classes in the provenance model.

ORG The Organisation vocabulary[34], a W3C standard, contains the *org:Organisation* class. Extra triples are added to the IATI Linked Data model creating a relationship between this class and *iati:organisation* classes in the IATI model.

GEO The classes in the Geo vocabulary[35] have a relationship to all country, region and location classes, as well as the latitude and longitude properties of a location.

CC The Creative Commons vocabulary[36] is only used in one triple within the IATI model, which creates a relationship between *iati:source-document-license* in the IATI provenance model to the *cc:license* property.

PROV As mentioned above, the PROV vocabulary has relationships to the provenance model. Different than the other vocabularies though, since the PROV vocabulary does not always have a relationship to a IATI class or property, but this vocabulary has some stand-alone classes and properties within the IATI Linked Data model.

Next to the vocabularies mentioned above, we have considered using other vocabularies such as Description Of A Project[37], the Finance ontology[38] or Data Cube[39]. Either based on the user needs, redundancy or model incompatibility, these vocabularies are not used.

In total, 21.5% of the properties and 26.9% of the classes in the activity model have a relation to an external vocabulary. In addition, a third of the properties and classes in organisation model also has a relation to an external vocabulary. In the codelist model, 42.9% of the properties are related to an external vocabulary, but also all classes are. Finally, in the provenance model 31.9% of the properties and, again, all classes have a relation to an external vocabulary.

4.3 Implementation

As for most of the implementation work in this paper, the conversion process is implemented using Python scripts. All scripts for converting IATI XML files to RDF are available at the GitHub repository of this paper on https://github.com/KasperBrandt/IATI2LOD/tree/master/IATI2LOD/src/conversion%20scripts. Starting with the process for gathering all IATI XML files, we use a crawler on the IATI API[40] (*IatiCrawler.py*) to look for all activity, organisation and codelist files. These files, as well as the JSON files, are saved locally so they can be processed later.

The httplib2[41] Python library is used to connect to the IATI API that returns a list of datasets, which can be used to retrieve the JSON file for each activity or organisation file. When looking for codelist, the API returns the names of the codelists which can be directly looked up since these XML files are located on the IATI website. However, for activity and organisation files, the XML usually resides on the website of the organisation that reported the XML file and the address has to be retrieved within the JSON file. Once an XML file is found, both the XML and the JSON file are stored to disk. Retrieving all activity, organisation and codelist files takes approximately an hour using this script. Since calculation time was no constraint for us, we did not try to optimise these scripts.

[31] http://www.w3.org/2004/02/skos
[32] http://dublincore.org/documents/dcmi-terms/
[33] http://www.foaf-project.org/
[34] http://www.w3.org/TR/vocab-regorg/
[35] www.w3.org/2003/01/geo/
[36] http://creativecommons.org/ns
[37] http://semanticweb.org/wiki/DOAP
[38] http://fadyart.com/en/index.php?option=com_content&view=article&id=121&Itemid=68
[39] http://www.w3.org/TR/2013/CR-vocab-data-cube-20130625/
[40] http://www.iatiregistry.org/api/
[41] code.google.com/p/httplib2/

The next step in the conversion process is to convert these XML and JSON files to RDF. The standard Python library ElementTree[42] is used to process the XML files, which are then converted to RDF using the RDFLib[43] library. For each type of file – activity, organisation and codelist – a different script is used to initiate this process.

These scripts look at all the files within the specified folder containing XML files and – in case of the activity script – extract activities from the files. The activities themselves, however, are processed by the *IatiConverter.py* and *IatiElements.py* scripts. The former makes sure every element of the activity, organisation or codelist is processed and passes these elements on to the latter script. In this script, a function for every element is specified, turning the XML into a RDFLib Graph. Elements in the XML files that are not in the IATI model are processed by a different function in a generic manner. Finally, the original script makes sure that the RDFLib Graphs are exported into RDF in the Turtle syntax and stored to a local folder. The script for converting all activity files runs for approximately 4 hours, whereas the organisation script is finished within 15 minutes and the codelist script within 5 minutes.

A ClioPatria[44] triple store is setup and available at http://semanticweb.cs.vu.nl/iati/. The data can be explored here and a SPARQL endpoint is provided at http://semanticweb.cs.vu.nl/iati/sparql/. Since ClioPatria runs on Prolog, a Prolog script is used to load each RDF/Turtle file into the triple store. The last update of the conversion and triple store was performed on the 1st of June, 2013. By then, 233,193 activities, 3,525 organisations and 1,490 codelist codes were imported. The total number of triples in the triple store is approximately 35 million and it takes about half an hour to upload all data into the triple store.

This paper describes the conversion and publication of the data. The conversion scripts are publicly available and can be run on new and updated data. Since the IATI registry provides dates showing when the files have been last updated, it is possible to create additional scripts for updating the data in the triple store. However, for a truly sustainable dataset, this should be owned and maintained not by a university but by a transparency organization. We are working with IATI representatives on the uptake and transferral of the model and the data. As such, we are currently not concerned with the maintenance.

[42] http://docs.python.org/2/library/xml.etree.elementtree.html

[43] https://github.com/RDFLib

[44] http://cliopatria.swi-prolog.org

4.4 Evaluation

When looking at our approach of creating the Linked Data model, we focused on setting up the requirements and creating the model first and linking the model to external vocabularies afterward. This approach has its advantages, the IATI Linked Data model being consistent with the original IATI model for instance, creating a recognisable model for the IATI users. However, it also has its disadvantages when looking at the possibilities of mapping the classes and properties of the Linked Data model to external vocabularies. The Data Cube model, for instance, required a completely different build up of the model and the data, therefore it is not possible to map this vocabulary onto the current Linked Data model.

A different example is the PROV model, for which we have asked feedback to a working member on the W3C PROV Primer group. In general, the provenance model in itself looked as it should and was technically viable. However, it was noted that the PROV model is currently used to indicate metadata on the IATI files and the conversion process of the IATI XML data to the Linked Data model, whereas it could also be applied to the IATI model itself, e.g. by modelling the ⟨reporting-org⟩ element as a *prov:Agent* to which IATI activities are attributed to, being of type *prov:Entity*. This would allow to see provenance on the activity level instead of the XML file level. However, linking the PROV model to the activities without changing the current IATI Linked Data model might still be possible, but further research is needed to find out to what extent this is possible.

Since we had to adhere to the structure of the original IATI model and since we did not want any information to be lost, sometimes we had to create additional nodes causing more triples and SPARQL queries with more clauses. Especially for description elements, it turned out that we had to create an extra node for every instance of this element because each description has a description type. For our model, it would have been more convenient to disregard this description type, but in order to prevent information loss we did decide to create the extra nodes.

Furthermore, the requirements that have been setup allow for a generic way of converting IATI XML files to a Linked Data model. However, in our experience, creating a model which can be used in practice always creates certain exceptions and therefore design choices have to be made, even though that might mean diverting from the requirements that have been setup. In this light, the requirements really should be seen as strict guidelines and we deviated from these requirements a

number of times. Nevertheless, setting up these requirements help in creating a structured overview of the conversion process.

The evaluation of the Linked Data model itself is done by gathering expert feedback on both the requirements and the model. In order for the model to be used in practice, it is needed that setting up a Linked Data model remains an iterative process in which the model and its requirements are under a constant feedback loop, just like the IATI model itself. The resulting Linked Data model as introduced in this paper should therefore be seen as the first iteration of this process and further research should be done for extracting additional feedback and updating the requirements and model accordingly.

Concerning the vocabularies that are linked to the IATI Linked Data model, the links have been evaluated by experts in the Linked Data field and in the IATI technical group. These links are found to be technically correctly linked and useful in practice. Even though a large number of properties and classes remains unlinked to any external vocabulary, they can still be linked in a different iteration without changing the original structure of the IATI model.

We note that an important aspect of the model is to bring it in practice. In that sense, live queries[45] on the triple store containing data from the Linked Data model have been succesfully run by a member of the Department for International Development[46] in collaboration with an IATI expert in order to show all the transactions which include 'GB-1' as the providing organisation. Since no other information source is yet available to show this information easily using such a small query, it shows that this model and the data in the triple store can be used in practice and provides technical users with the capability of performing advanced queries on the IATI dataset.

As an additional validation of the usability of the produced linked data, we refer to the work of Lemmens and Keßler[10], who used the Linked Data directly from the SPARQL endpoint for creating geo-information visualisations. Similarly, the data were used by Capadisly et al. for the Semantic Web Challenge submission[14]. In both cases this was done without cooperation or assistance from the authors of this paper.

[45] https://gist.github.com/practicalparticipation/5684302

[46] https://www.gov.uk/government/organisations/department-for-international-development

5 Linked Datasets

In this section, we describe the external datasets linked to the Linked Data model. In Section 5.1, a description of the approach and methodology is given. Section 5.2 shows the results of linking the datasets as well as the implementation details. Finally, these results are evaluated in Section 5.3.

5.1 Approach and Methodology

Our main approach for linking external datasets starts by picking relevant external datasets. Since the LOD cloud contains a lot of information, it is possible to choose from nearly any linked dataset for creating links to the IATI dataset. In order to get relevant information, we focus on the specific IATI user needs as specified in Section 3.2. The second step is to match the data from the IATI dataset to the external datasets through the use of matching algorithms. Even though sophisticated matching algorithms exist, as can be read in Section 2.3, we have employed simple matching algorithms through the use of Python scripts. This allowed us to fully customise the linking process ourselves, including any domain specific exceptions in the linking rules and custom API calls in the matching process, as opposed to using existing linking software such as Silk.

5.2 Results

In this section, we show the results concerning the linking of the datasets, starting with the relevant datasets in 5.2.1. In the following sections we elaborate on the links with these datasets and the algorithms that are employed to match the datasets, as well as the implementation details. We have picked relevant datasets based on the IATI user needs as described in Section 3.2.

5.2.1 DBpedia

Linking the IATI dataset to DBpedia is necessary to retrieve the Human Development Index and violent conflicts for every country (user need 2 and 4). Therefore we had to find the corresponding country page for every country in the IATI country codelist, consisting of the country code and name. The country codes are specified in the ISO 3166-1-alpha-2 standard[47].

Unfortunately DBpedia has no direct relation to an ISO code, so another way of linking the IATI codes to

[47] http://www.iso.org/iso/country_codes/iso_3166_code_lists/country_names_and_code_elements.htm

DBpedia had to be found. We have employed a workaround using the linked dataset of CIA Factbook. This dataset also did not contain any ISO codes, but it does contain a *factbook:internetcountrycode* property for every country. These codes matched the country ISO codes by making the codes lowercase and adding a period in front of the code. This way we found the resource link to a country in the CIA factbook.

The DBpedia dataset contains an *owl:sameAs* relation to the CIA Factbook resources, using this relation we could match the country with the IATI dataset. However, if a country could not be found using this method, another query is performed on the DBpedia dataset itself, directly querying for the internet codes using the *dbpprop:cctld* property. This combination of these two queries provided better results than only querying for the internet code on DBpedia in the first place. For the above algorithm, we used the *FactbookCountries.py* script, which first uses a SPARQL query on the CIA Factbook endpoint[48] to retrieve all countries and their internet country codes[49].

Another SPARQL query is then performed on the DBpedia endpoint[50] which retrieves all DBpedia countries and the URIs specified by the *owl:sameAs* property[51]. Then all countries in the IATI country codelist are retrieved from the local codelist XML and converted to the internet country code format (e.g. 'AF' to '.af'). These codes are matched with the results from the CIA Factbook query, thereby retrieving the CIA Factbook URI of these countries and matching the CIA Factbook to the IATI dataset. These CIA Factbook URIs are then matched to the results of the DBpedia query, which contains both the DBpedia URI for countries as well as the CIA Factbook URIs through a relation containing the *owl:sameAs* property. If a match is found, the DBpedia URI is matched to the original IATI country code.

The final step includes another SPARQL query to the DBpedia dataset, retrieving all countries and their internet codes as specified on DBpedia through the *dbpprop:cctld* property. All countries from the IATI country codelist that were not matched in the first part of this algorithm are then matched to this dataset using their internet code again. If the internet code is found this way, the DBpedia URI of the country is also linked to the IATI country codelist. An *owl:sameAs* property is used in order to denote the relationship between these URIs. In order to store these matches in the triple store, the RDFLib Python library is used again to create local RDF files in the Turtle format for the matching URIs, which can then be uploaded to the triple store.

[48] http://wifo5-03.informatik.uni-mannheim.de/factbook/sparql

[49] The SPARQL query itself is specified at https://gist.github.com/KasperBrandt/6106898

[50] http://dbpedia.org/sparql

[51] This SPARQL query is available at https://gist.github.com/KasperBrandt/6106928

5.2.2 GeoNames

User need 3 concerns the specific geographical location of an aid activity. Mapping to GeoNames has been done on two levels: country and location. Starting with the mapping on country level, we looked up the country code from the codelist and used the GeoNames search webservice[52] to find the related GeoNames code of the country. This is done by making use of the following GeoNames 'feature codes'[53]: PCLI, PCLD, TERR, PCL, PCLF, PCLIX, PCLS and PCLH – in that order. These all represent some kind of populated place or territorium.

The script that is used for finding GeoNames countries starts by looking up all IATI countries in a local codelist XML file. The GeoNames search webservice is then used for each country to retrieve all matches from all the feature codes. If only one match is found in all feature codes, the GeoNames code of this entity is retrieved and used to form the URI of this GeoNames entity in the form of http://sws.geonames.org/[code]. If no matches could be found, the country is not matched and in case multiple results are returned from the GeoNames webservice, the order of the feature codes as specified above is taken and the first result is matched. Again, the final step of the script is to use the RDFLib library for creating RDF triples in the Turtle format and store this in a local file.

For matching the IATI dataset on the locations level we needed a more complex algorithm. Locations can be specified according to a number of fields in the IATI standard: Location Type, Name, Description, Administrative Country, Coordinates and Gazetteer entry. Even though this allows IATI users to specify a location accurately, most files either did not specify the ⟨location⟩ element, or specify it very scarcely, containing just a reference on country level for instance. The algorithm that is used for matching locations works in the following manner: first a classification from 1 to 5 is made based on the ⟨location⟩ element, separating the locations which have coordinates specified from the elements containing sparser information. Location elements which cannot be classified are ignored. This simple classification has the following values:

1. a longitude, latitude and precision specified.

[52] http://api.geonames.org/search?

[53] http://www.geonames.org/export/codes.html

2. a longitude and latitude specified.
3. a label and a country label specified.
4. just a label specified.
5. just a country label specified.

Each classification value has its own algorithm. For locations classified by number 1 we make use of the precision values, a lower precision value indicating a more precise location, whereas a precision value above 5 indicates country or even continent level. For precision values lower than 3 the GeoNames service is queried for a city or place. Precision value 3 indicates a region or district and therefore the GeoNames feature code of 'ADM2' is used. Precision values 4 and 5 indicate a state or province corresponding to the 'ADM1' feature code. For locations in the second classification category, only the latitude and longitude values are used for querying GeoNames. First, a search is conducted for a place or city based on the specified coordinates. If this returns no results, a broad search on nearby GeoNames objects is performed. Finally, algorithms 3 to 5 all make use of the general search function within the GeoNames API. Locations in category 3 will first be searched on their label and if no matches were found, their country label is used for searching. Since the GeoNames API returns the most relevant items on top of their search result, the top result will be matched to the location. Locations classified in the fourth and fifth category only make use of a search on respectively their label or country label.

The implemented script first collects all locations from the IATI activity XML files and categorises these into one of the five categories as specified above. Then a query is performed on one of the GeoNames webservices, such as *findNearbyPlaceName*, *findNearby* or *countryCode*, depending on the category of the location. Again, no SPARQL queries are used for retrieving this information, since these webservices return a GeoNames code which can be used to create the corresponding URI of the GeoNames entity. Finally, as for all the linked datasets, the triples are stored in Turtle format in a local file.

5.2.3 World Bank indicators

User need 5 requires a connection with World Bank indicators. Using a simple SPARQL query on the World Bank SPARQL endpoint[54], the ISO codes of the IATI country codelist can be directly matched to the World Bank dataset. These are specified according to a *skos: notation* relation.

The implementation of this algorithm is, just like the algorithm itself, very straightforward. The country codes are first collected from the local country codelist XML files. Then a SPARQL query is performed on the World Bank endpoint to find the matching URI in the World Bank dataset. The query that is used can be viewed at https://gist.github.com/KasperBrandt/6107069. This query also returns all URIs of the *owl:sameAs* property linked to the World Bank URIs, because they contain URIs of the Eurostat dataset as well. Finally, the matches that are found are converted to triples using the RDFLib library and stored to a local file.

5.2.4 Other Datasets

Besides links to DBpedia, GeoNames and World Bank indicators, we have created links to other datasets as well. Even though these datasets are not specifically required by the user needs and not used in the Linked Data applications as of yet, only a few extra lines of code were needed to gather these links because of their *owl:sameAs* relations to either the DBpedia or World Bank dataset:

CIA Factbook As was noted in 5.2.2, DBpedia links are created using a CIA Factbook connection. These contain population, government, military, and economic information for nations that are recognised by the United States. For implementation details, see 5.2.2.

Eurostat The Eurostat dataset is closely related to the World Bank data, containing general statistics and indicators of European countries, and is matched through an *owl:sameAs* property in the World Bank dataset. For implementation details, see 5.2.4.

OECD The Organisation for Economic Co-operation and Development dataset[55] contains links to DBpedia, BFS, ECB and FAO and therefore acts as a central hub for connecting various datasets. The dataset itself contains data on economic, social and governance variables of mainly developing countries. The *OecdCountries.py* script is used to match the countries, containing the SPARQL query as formulated on https://gist.github.com/KasperBrandt/6107021.

BFS The Bundesamt für Statistik dataset[56] of the Swiss Federal Statistical Office contains general statistics of mainly European countries. An *owl:sameAs* property in the OECD dataset is specified to both BFS and DBpedia, making it possible to link these datasets and the implementation is similar to the OECD dataset.

ECB The European Central Bank has a linked dataset[57] on monetary operations, government finances

[54] http://worldbank.270a.info/sparql

[55] http://oecd.270a.info/

[56] http://datahub.io/dataset/bfs-linked-data

[57] http://ecb.publicdata.eu/

and more of European countries. However, they also have an entry for most non-European countries in their dataset. The matches are created through the OECD dataset and the implementation is similar to this dataset as well.

FAO The dataset of the Food and Agriculture Organisation[58] of the United Nations contains agricultural research information. Like the BFS and ECB datasets, it can be linked, using the OECD dataset.

Transparency International The data that are collected from Transparency International are composed of Corruption Perceptions Index data, directly matched on their SPARQL endpoint[59] to the World Bank dataset. To create links, we used the *Transparency-Countries.py* script and the SPARQL query found at https://gist.github.com/KasperBrandt/6107044.

5.3 Evaluation

We evaluated the above described algorithms by means of precision and recall values. This evaluation is performed manually by the authors. For some datasets it is possible to evaluate the full dataset by hand, whereas for other, larger, datasets we have picked a subset and extrapolated those evaluations to the complete dataset in order to retrieve the exact number of concepts within the IATI dataset that can be correctly matched. In Table 8, an overview of all datasets and their respective precision and recall values is given. In the online appendix, an extensive overview of all mappings is shown. Below, we discuss the evaluation in more detail.

Dataset	Precision	Recall
GeoNames countries	1.0	0.996
DBpedia	0.98	0.976
CIA Factbook	1.0	0.96
World Bank indicators	1.0	0.848
Eurostat	1.0	0.848
OECD	1.0	0.832
ECB	1.0	0.820
BFS	1.0	0.804
FAO	1.0	0.764
Transparency International	1.0	0.728
GeoNames locations	0.655	0.363

Table 4 All precision and recall values of the linked datasets, sorted by the recall values

After a manual evaluation of DBpedia links, we did find that five countries were incorrectly matched, mainly due to the fact that some countries have the same internet code. For example, Guadeloupe and Martinique have the same internet code as France and are therefore linked to the DBpedia country page of France. This gives us a recall of 0.976 and a precision of 0.98.

For the country mappings to the GeoNames dataset, 249 out of the 250 countries in the IATI country codelist were correctly found. The only country that could not be matched is the Netherlands Antilles, since the 'AN' code could not be found by the GeoNames webservice[60].

The results for the GeoNames locations algorithm are less satisfying. In total, a number of 6,854 locations have been specified in the IATI XML files and 2,237 locations have been matched using the GeoNames service. Since evaluating all 2,237 links manually is impractical, we decided to take two random sets of 100 links and evaluate these. Of the first set of 100 links, we found 47 to be relevant. The second sample set gave us 84 relevant results. The combination of these two evaluation sets gives us a precision of 0.655 and a recall of 0.363. The number of relevant results when looking at all locations would approximately be 2,488 out of 6,854. In comparison, the number of activities in our dataset is 233,193. Since we would only have a very small subset of the activities linked to a correct GeoNames location, 1.07%, and since the precision and recall values are very low, we decided not to use these in links in our Linked Data application. When looking at these matches, we found both the contents of the IATI files as the actual results for matching the *iati:location* nodes disappointing. As only 6,854 locations have been specified from a total of 233,193 activities, meaning that a mere 2.94% of the activities has a ⟨location⟩ element, the starting point for creating matches was not optimal. Furthermore, we found the quality of the ⟨location⟩ element contents in the IATI files disappointing. In most cases, only the country had been specified, which makes this element a duplicate version of the ⟨recipient-country⟩ or ⟨recipient-region⟩ elements. Also, multiple locations are sometimes specified within one element – e.g. "Kenya, South Africa and Namibia" – and some location element contain a just dash or "Local" in the description field. In the first case we could have extended our scripts to incorporate multiple locations. However, we decided not to do so because of time constraints. Furthermore, more sophisticated methods for matching unclear locations will not help much, as it is still impossible to match a location which has only "Local" in its description field.

The remaining datasets are all matched on country level through the use of the ISO codes. This results in a precision value of 1 for all these datasets, making these standardised codes a safe manner of matching countries. When looking at the World Bank indicators, 212

[58] http://data.fao.org/

[59] http://transparency.270a.info/sparql

[60] The 'AN' code has since been added.

out of 250 countries are correctly matched to the IATI dataset, resulting in a recall value of 0.848. The reason that some countries could not be found is because they mostly do not exist in the World Bank dataset. Especially small countries, such as Gibraltar or Nauru, and old colonies or overseas islands, such as Réunion or Mayotte, are missing from their dataset.

All other datasets are linked through a relation containing the *owl:sameAs* property within either the DBpedia dataset or World Bank indicators. It is therefore not surprising to see that these datasets score lower or equal to these datasets on their recall values. The Eurostat dataset has exactly the same scores as the World Bank indicators, showing that all countries in the World Bank dataset also have a link to the Eurostat dataset.

Furthermore, since the OECD dataset acts as a central point for the ECB, BFS and FAO datasets, the evaluations show that this dataset ranks best of these four datasets. The same holds for the Transparency International dataset, which derives its links through the World Bank countries. The evaluations show that not all countries in the World Bank dataset have a link to the Transparency International dataset, causing a lower recall value of 0.728 for the latter. All of the links that have been created have been uploaded to the triple store, even though not all are used in the Linked Data applications. These applications only make use of the DBpedia dataset and World Bank indicators.

6 Linked Data Applications

In this section, we describe the Linked Data applications, which are available at the applications page of the IATI2LOD website[61].

6.1 Approach and Methodology

The methodology for creating the Linked Data application is largely based on the user needs as described in Section 3.2. The main goal is to show the added value of Linked Data as opposed to using just the IATI data. We did not take into account any usability or user experience methodologies, but focused on the added value through the use of Linked Data instead. For each of the applications we have evaluated whether it meets a specific user need. Since these needs could not have been met when using just the IATI dataset, the resulting applications show that having the IATI model as Linked Data will provide added value for users.

[61] http://iati2lod.appspot.com/applications

6.2 Results

All applications have been built in the Google App Engine (GAE)[62], which allows us to use Python scripts in combination with the Jinja2[63] template engine for creating dynamic websites. All maps and graphs in the applications make use of Google Charts[64], effectively allowing us to focus solely on the data without having to pay attention to the layout of these graphs. Furthermore, data are not queried live on the triple stores, as this drastically decreased the performance of the applications. Instead, the data of the external datasets are gathered through SPARQL queries and stored in the GAE datastore. For each of the applications describe below, we give an explanation of the SPARQL query that is used to gather the data and the performance of this query. All of these SPARQL queries can be found on https://gist.github.com/KasperBrandt.

6.2.1 General applications

The general applications take user need 1 as an input. In order to answer this need, we have built a number of different applications which can be viewed on http://iati2lod.appspot.com/applications/general. No externally linked datasets have been used for creating these applications.

First of all, an application is created showing an overview of the total number of activities on a World map. For creating this overview, the sum of all activities having the *iati:activity-recipient-country* property is taken per country. The countries with a larger amount of aid activities are colored darker green in the World map, whereas countries with fewer activities are colored light green. As a result the map shows that Afghanistan and Pakistan are the countries having the largest number of activities, respectively 4,846 and 4,313 activities.

A SPARQL query[65] collects all activities with an *iati:activity-recipient-country* property as well as the country code belonging to the activity. The results are grouped per country and a count is performed on the number of activities belonging to each of the countries. This SPARQL query performs fairly good, returning results in approximately 4 seconds. However, from this overview the first user need can not be completely satisfied, since it only shows the number of aid activities and not the amount of aid a country receives. Therefore

[62] https://developers.google.com/appengine/
[63] http://jinja.pocoo.org/
[64] https://developers.google.com/chart/
[65] available at https://gist.github.com/KasperBrandt/6107184

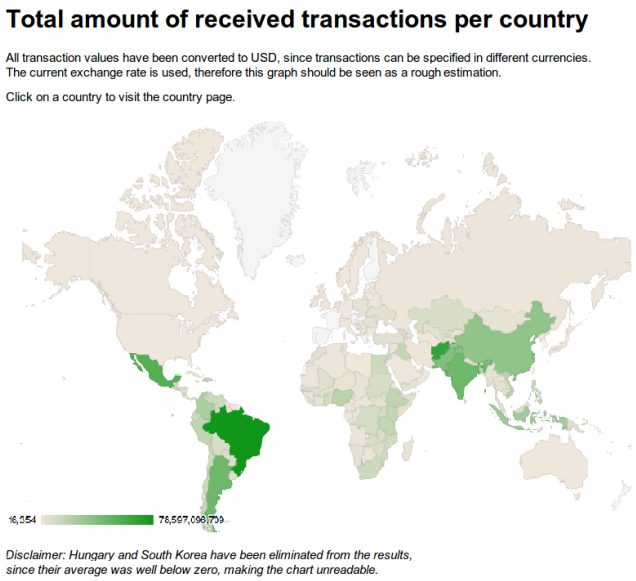

Fig. 2 Screenshot of one developed application showing the world map of the total amount of transactions

we have created several other World maps, showing the total amount of transactions, budgets and planned disbursements which can be linked to the recipient countries.

These three applications all work in the same way: the financial data are gathered from the IATI dataset and summed up per country and per currency. In order to be able to compare all values, it is necessary to convert all currency values to the same currency. Since converting currencies for each separate transaction or budget is a very costly calculation, we decided to take the current exchange rate and convert all values to US Dollars. As can be seen from Figure 2, Brazil receives the biggest amount in aid transactions. The three SPARQL queries used for gathering the amount of transactions, budgets and planned disbursements are similar to each other. The only difference being the *iati:activity-transaction* property which is used for transactions, as compared to *iati:activity-budget* for budgets and *iati:activity-planned-disbursement* for planned disbursements. As an example, we take the transaction SPARQL query, located at https://gist.github.com/KasperBrandt/6107203. This query is more complicated than the previous one, mainly because it needs to replace all commas in the transaction values and cast these values to floats in order to be able to sum up all values. Other than that, each transaction, as specified by a *iati:activity-transaction* property, is queried and the country code, currency code and value of the transaction is retrieved. The results are then grouped by country first and currencies second, and the sum of all transaction values is calculated per country-currency combination.

Because of the double grouping and the calculation of the sum for each group, these queries take a lot longer to calculate results. In general it takes about 50 seconds for the triple store to give a result on each of the three queries, very long for a live application. This is the main reason we have opted for querying and calculating these results beforehand and storing it in the GAE datastore, as the loading time for the current live application does not take longer than 2 seconds.

6.2.2 Human Development Index

The second application[66] shows a comparison between the IATI data and the Human Development Index (HDI). For information about the HDI we have used the linked DBpedia dataset, which contains HDI information for most countries. This application combines a SPARQL query for summing up all transaction values – the same query which is used for the general application of transactions – and a query to the DBpedia database, gathering the HDI rank and year in which the rank was set.

Unfortunately, not all countries on DBpedia have a HDI rank or year, a total of 61 countries did not. Countries for which an HDI value was found on DBpedia are sorted in reverse, to show the countries with a lower HDI on top. It would be expected that these countries, since their development status is lowest, receive more aid than the countries higher ranked in the HDI. Of course other indicators, such as the population size, should be taken into account as well in order to get a clear comparison. However, when comparing just the IATI data versus the HDI, some interesting observations can already be made. For example, Peru – number 63 on the HDI list – receives much more aid money ($ 16,856,825,629) than Eritrea ($ 961,504,336) or Gambia ($ 863,481,886), respectively 165th and 168th on the HDI list.

A simple SPARQL query combines the retrieval of the country's thumbnail, abstract and HDI, see https://gist.github.com/KasperBrandt/6107271. The thumbnails and abstracts are used on the country pages, as described in 6.2.4, but since the query is very simple and only used to store the data once, we opted to retrieve all country data from DBpedia at the same time. The performance of this query is very good, giving results back nearly instantly. However, since it would be needed to perform this query for every country, it would still take

[66] See http://iati2lod.appspot.com/applications/general?graph=country-hdi

more than 10 seconds to retrieve all the data. In combination with the query for retrieving the total transaction values per country, this would mean that retrieving information for this application could take up to a minute.

6.2.3 World Bank indicators

The World Bank has a number of indicators for various topics, for example on agriculture and rural development. In the Linked Data application, a subset of the indicators is chosen to be compared with the IATI data. The selected indicators all have US Dollars as their units, since we can not compare indicators having a different type of unit, such as '% of GDP', to the IATI transactions dataset. Furthermore, only indicators containing data for all countries have been selected. Most of the World Bank indicators have missing data, especially on Third World countries, and therefore these indicators are disregarded. The indicators that have been selected are: current account balance, net official development assistance and official aid received, net official development assistance received, total reserves, and GDP.

In the application, the user has to select a country and one of the indicators. Based on the selected country and indicator a comparison graph of the IATI transactions and the indicator is shown per year. In order to show this graph, a SPARQL query is performed on the IATI dataset which is similar to the query that is used for showing the total amount of transactions of the general applications. However, since we have to show the data for each year, the results are grouped per year as well. Another SPARQL query is performed on the World Bank data in order to gather the data for the selected indicator and country. Only the indicator data from 1990 onwards are shown, the IATI data usually do not go further back than 2000.

Unlike the other datasets, we did not query the World Bank SPARQL endpoint[67] for retrieving the data, since this endpoint was very unstable at the time and often did not return results. Instead we used their webservice[68]. However, to show that it is possible to use a triple store, we rebuilt their data according to the Data Cube vocabulary and uploaded it to the IATI triple store. Since the World Bank dataset is modelled using the Data Cube vocabulary, we use the measure and dimension of the cubes to retrieve the value, year and country belonging to the indicator. The query that was used is shown below.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX iati: <http://purl.org/collections/iati/>
PREFIX cube: <http://purl.org/linked-data/cube#>
PREFIX meas: <http://purl.org/linked-data/sdmx/2009/measure#>
PREFIX dim: <http://purl.org/linked-data/sdmx/2009/dimension#>
PREFIX wb: <http://worldbank.270a.info/dataset/>
SELECT ?country ?yearint ?value
WHERE {
        ?obs a cube:Observation.
        ?obs meas:obsValue ?value .
        ?obs dim:refPeriod ?year .
        ?obs dim:refArea ?country .
        FILTER regex(str(?country), "iati", "i") .
        FILTER regex(?year, "[0-9]") .
        BIND (xsd:int(?year) as ?yearint) .
        FILTER (?yearint >= 1990)
}
```

In addition, a SPARQL query for retrieving the total transactions per country per year is used in order to be able to compare the transactions to World Bank indicators. This query is similar to the total transactions per country, only the year is now retrieved as well and the results are also grouped by year. The query is shown below and performs worse than the total transactions per country query. Therefore this query is not suited at all for live applications. Even more so, the Google App Engine does not allow for connections taking longer than 60 seconds. Since this query usually took longer than a minute, it was not possible to create this application at all without pre-calculating the data and storing it to the GAE datastore.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX iati: <http://purl.org/collections/iati/>

SELECT ?countrycode ?year ?curcode (sum(?float) as ?sum)
WHERE {
  ?act iati:activity-transaction ?trans .
  ?trans iati:value ?value .
  ?trans iati:value-currency ?currency .
  ?trans iati:transaction-date ?date .
  BIND (SUBSTR(?date, 1, 4) as ?year) .
  ?currency iati:code ?curcode .
  BIND (REPLACE(?value, ",", "", "i") AS ?correctedval) .
  BIND (xsd:float(?correctedval) as ?float) .
  ?act iati:activity-recipient-country ?country .
  ?country iati:country-code ?code .
  ?code iati:code ?countrycode
} GROUP BY ?countrycode ?year ?curcode
```

6.2.4 Country pages

The main application bringing all of the above together is the country pages. The user can navigate to these country pages from the general applications by clicking on a country in the World map or by clicking on the name of a country in the HDI overview.

As is shown in Figure 3, the country pages consists of various elements. Starting from the top of the page, information from DBpedia is shown in the form of the flag of the country, a short abstract of the country and the HDI year and rank of the country which is shown

[67] http://worldbank.270a.info/sparql

[68] http://api.worldbank.org/countries/[country_code]/indicators/[indicator]

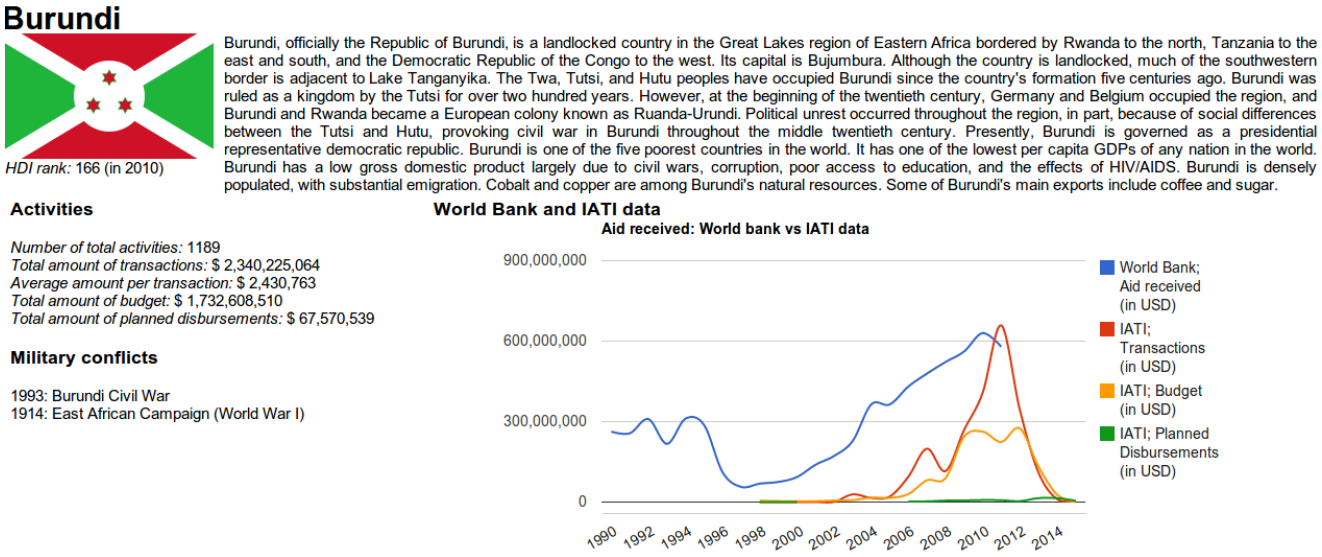

Fig. 3 Country page of Burundi (*http://iati2lod.appspot.com/applications/countries?country=BI*)

underneath the flag. This information is gathered using one SPARQL query on the DBpedia endpoint as described in 6.2.2.

Furthermore, a small summary on the IATI activities is shown on the left side of the page. This corresponds to the information of the general applications, showing the total number of activities, the total amount of transaction, the average amount per transaction, the total amount of budgets and the the total amount of planned disbursements, as specified in the IATI files. Underneath the IATI information a list of military conflicts is shown. This, again, is gathered from DBpedia by querying for all military conflicts linked to a country since 1900. The year of the conflict is shown, as well as the name of the conflict.

Finally, the central graph on the page shows the World Bank indicator *net official development assistance and official aid received* per year since 1990, as well as the total amount of transactions, budgets and planned disbursements per year. The queries needed to create this graph are similar to the ones used on the World Bank indicator page. However, additional queries are needed to calculate the budgets and planned disbursements per year, since these are not shown on the World Bank page. Here again, for speed pre-calculated data are cached in the GAE datastore.

The SPARQL queries that are used in this application consist of general IATI activity information as described in 6.2.1, the thumbnail, abstract and HDI information retrieval from DBpedia, as described in 6.2.2, and World Bank data and transaction values per year as described in 6.2.3. In addition, data on violent conflicts are needed from DBpedia as well, for which we created a SPARQL query that retrieves all conflicts at once as shown on https://gist.github.com/KasperBrandt/6107152. In the application itself, a filter is then placed on the selected country. Furthermore, budget values per year[69] and planned disbursement values per year[70] are needed. These SPARQL queries are similar to the transaction per year query.

6.2.5 Evaluation

The evaluation of the Linked Data applications is done based on the user needs (see Section 3.2). From the general applications, the user is able to see how much aid each country receives in transactions, budgets and planned disbursements. Therefore, the first user need is satisfied by making use of these applications. However, the data as shown in these applications should be taken as a very rough estimate, since not all aid donors are currently publishing their aid activities in IATI format and therefore we can not be sure that these applications give a correct overview of all received aid of a country in total.

The second user need is partly fulfilled by the application showing IATI transactions next to the Human Development Index. Not all data on the HDI are available, which makes the overview incomplete. However, countries having an HDI rank specified can be compared to one another and to the amount of aid transactions. It should be taken into account that this is a very flat comparison with just IATI data and HDI as variables, whereas the overview could give a better picture

[69] https://gist.github.com/KasperBrandt/6107120
[70] https://gist.github.com/KasperBrandt/6107282

of the aid division between countries if other variables would be taken into consideration as well.

User needs 3, 3a and 3b are not sufficed at all. Because data on locations are too scarcely available and because some of this data could not be correctly matched, we have decided not to incorporate any specific location based applications.

When looking at user need 4, it is satisfied, albeit partly, by the country pages. After a manual evaluation on the military conflicts that are available on DBpedia we noticed that not all conflicts are available. This overview only covers military conflicts, meaning that some civil wars or genocides are not taken into account. However, from the list of military conflicts in combination with the abstract of the country, a general sense of the state and history of each country is accounted for.

Finally, the World Bank indicator application is created in order to fulfill user need 5. Even though not all World Bank indicators are available within the application, the indicators that are eligible for a comparison to IATI transactions, budgets or planned disbursements are. Also, we have noticed that a lot of data missing from the World Bank indicators, making it impossible to create a comparison to most of these indicators. Therefore user need 5 is essentially fulfilled.

A number of comments also have to be made regarding the specification of the data in the IATI files. We mostly had issues with the content of the IATI files as provided by the aid donors, since many of the IATI files actually do not have a ⟨recipient-country⟩ element or the code is not specified according to the IATI country codelist. For example, some files have "Afghistan" or "AFG" specified as the country code, whereas the correct code for Afghanistan would be "AF".

Other than working with invalid country codes, we also found the specifications for money values such as transactions, budgets or planned disbursements to vary. First of all, several currencies were used for which we had to use exchange rates to calculate the money value in US Dollars. Even though this was to be expected, it does complicate comparisons. Since the historical exchange rates are not available as open data and since it would be computationally very costly to calculate each value according to the actual exchange rate of the historical transaction date, we decided to use a general exchange rate for each currency based on the current exchange rates. Furthermore, some transactions are specified with commas, but most are specified without. Since the IATI model guidelines do not state the exact format of the money values, this was also to be expected. As a result, we checked whether commas were used to denote decimals. This is not the case, so we removed all commas in order to get valid integers. In addition, we found many negative transactions. The exact meaning of these transactions remains to be guessed, but we have assumed that these transactions are to be extracted from an earlier transaction and therefore these values are also taken into consideration. As a result, the total transaction values for Hungary and South Korea ended up being well below zero and we decided to disregard these values in the World map of the general applications, since these maps become unreadable containing these values.

Another point of interest is that transactions have been used more often than the budget or planned disbursements elements. In general, the transaction element should indicate the committed or actual money flowing in or out of an activity, whereas the budget element should be used for the total yearly or – the preferred – quarterly budget. As for planned disbursements, these are originally meant to specify future releases of funds. However, even within the IATI userbase there has been some confusion in the use of these terms and elements[71]. So, in the end, we believe that the IATI transactions give the best representation of the money flows in aid activities.

Finally, we express our concern regarding the calculation time of SPARQL queries, especially when creating applications that use data from multiple datasources. In our opinion, this is a weak spot in the Web of Data. A solution for this problem is to retrieve all data needed for an application, pre-calculate it if needed and store it in one location. However, this creates multiple stores containing the same dataset or a subset thereof, which is not an ideal situation for updating data.

To conclude, we have created several Linked Data applications which show that specific IATI user needs are fulfilled through the use of Linked Data, which could not have been realised with solely the data of the IATI dataset. However, we see that there is room for improvement on multiple levels. First of all, the way data are specified in the IATI files by aid organisations is not optimal, which causes missing data in the Linked Data applications.

7 Generalisation

In general, we believe that making a generic approach for converting an open dataset to Linked Data is not feasible, since domain knowledge and user input are required in order to create a fitting conversion. However, following an approach similar to ours in this paper makes sure that the model can be used in practice. The

[71] http://support.iatistandard.org/entries/21431418-Budgets-and-Planned-Disbursements

steps that should be taken should follow these rough guidelines:

1. Specify the requirements of the model and the domain user needs, elicitated from interviews with users or experts. Take into account the structure of the model in combination with external vocabularies that fit the domain and the purpose of the model.
   (a) Get user or expert feedback on these requirements and the model itself in an iterative fashion, while working on the model itself. This is an ongoing process throughout the creation of the model.
2. Based on the user needs and the domain of the model, select relevant datasets in order to link these to the created Linked Data model.
3. Define algorithms for linking the selected datasets to the Linked Data model and evaluate these links. Decide for each of the datasets if the evaluation is sufficient for using the linked dataset in practice.
4. Create applications that serve the user needs and evaluate these with the users themselves.

Furthermore, the requirements as defined in Section 3.1 allow for a generic way of converting an open dataset to Linked Data. First of all, the two design principles of completeness of the data and a consistent use of structure and use of terms between the open data model and the Linked Data model makes sure that the Linked Dataset does not have information loss and can immediately be used by users of the open dataset, since the structure is familiar. When these design principles are relevant, the 6 requirements as specified in Section 3.2 should be adhered to, since they make sure that the data are complete and consistent in practice. Even though some requirements have been formulated in order to specifically convert an XML file to Linked Data, it is possible to use these requirements as well for other datatypes, such as JSON.

However, even when completeness and consistency are not relevant, requirements 3 to 6 are still useful for setting up a Linked Data model. In any case, giving a *rdf:type* property to every node in the model (requirement 3) allows for easy querying of those type of nodes and automatically creates an ontology for the model. Also, avoiding blank nodes (requirement 4) has become a general design principle since it is not possible to set external RDF links to this URI and updating triples in a triple store containing blank nodes is much harder than a triple stored containing just named nodes. Requirement 5 makes sure that no blank nodes are needed, since a hash is used when no identifier is present to uniquely identify a node. Finally, keeping the number of nodes to a minimum (requirement 6) is a design choice which makes sure that the model is not needlessly complex and also allows for simpler SPARQL queries.

The implementation as used in this paper has not been setup for a generic implementation of a Linked Data conversion. The Python scripts as used for crawling the IATI registry and converting the IATI data into RDF are specifically created for the IATI dataset. In contrast to the requirements, which are setup in a generic manner, these scripts contain a function for every specific element in the IATI dataset. These scripts are only reusable when the specific elements of a dataset are modelled. The same holds for the implementations of the algorithms for linking datasets and the Linked Data applications. These have been tailored to suit the purpose of the IATI dataset and its links to external vocabularies.

Concluding this section, we have shown that a generic approach for creating a Linked Data model and its applications makes sure that a practical model is created with applications that fulfill specific user needs. Furthermore, the requirements as described in this paper are useful for generating a Linked Data model in general, even when the design principles are not the same as ours. Finally, the implementations in this paper do not allow for a generic way of implementing a Linked Data conversion, but have been created specifically for the IATI dataset.

8 Conclusions

In this paper, a Linked Data model is introduced based on the IATI model. Requirements for creating the model are defined and all elements of the IATI model – activities, organisations and codelists – are converted to Linked Data and are mapped to often-used Linked Data vocabularies. As we have shown in Sections 3.1 and 4.2, setting up design principles and requirements for the model allows for a structured method of creating a Linked Data model. Even more so, the requirements are setup in a generic manner, allowing other datasets to create a Linked Data model in a structured manner as well. As provenance has become a central topic in Linked Data, a provenance model for the IATI dataset is described in 4.2.4. However, this provenance model describes metadata on the level of IATI files and the conversion process, whereas it is also a possibility to use it to describe the provenance on the aid activities themselves. Further research into this possibility is needed.

The data converted in this way are accessible as five-star Linked Open Data and part of the LOD cloud[72].

Besides creating a model, external datasets are linked to the IATI Linked Data model. A list of external datasets relevant to the IATI Linked Data model, such as the DBpedia, GeoNames and World Bank indicators, is shown in 5.2.1. Based on the user needs as elicitated from interviews with domain experts these datasets have been selected. Also, additional datasets have been linked to the Linked Data model that go beyond the scope of the user needs. The algorithms for linking these datasets are evaluated in 5.3. Generally, the algorithms only employed a strategy for exact matches and we are satisfied with the precision and recall values of most datasets. An exception is the algorithm for linking the IATI dataset to the GeoNames locations dataset, for which we were not satisfied with the results. However, creating a better algorithm for linking this dataset is hard, since the bad results are mainly caused by the specification of the data in the original IATI files.

Furthermore, Linked Data applications have been created based on the user needs. The applications as described in 6.2 show the added value of Linked Data since most of these applications, could not have been realised with just IATI data and without linking the external datasets to the IATI Linked Data model.

We have also found that the impact of the design choices on the Linked Data model are of not much influence when looking at the Linked Data applications. In general, the design principles and requirements that we adhere to in this paper make the IATI Linked Data model the way it is currently structured. But if the model had been structured otherwise, the applications themselves could probably be reproduced showing the same information as they currently do.

Finally, Section 7 shows how the results of this paper are related to converting open datasets to Linked Data in general. We show that it is possible to employ a generic approach for creating a Linked Data model from an open dataset. In addition, the setup requirements are useful for generating a Linked Data model in general, even for models with different design principles. However, all scripts and files used to create and convert the IATI model to Linked Data have been customised for the IATI dataset, therefore it is not possible to reuse any of the implementation directly.

[72] A landing page is available at http://datahub.io/dataset/activity/iati-as-linked-data

9 Acknowledgements

We would like to thank Tim Davies for his help on the Linked Data model and general explanations of the IATI model. We thank Christophe Guéret and Paul Groth for their feedback on the model. Finally, we thank Siem Vaessen and Vincent van 't Westende for their thoughts on and explanations of the IATI model.

References

1. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. International Journal on Semantic Web and Information Systems (IJSWIS) **5**(3), 1–22 (2009)
2. Bizer, C., Heath, T., Idehen, K., Berners-Lee, T.: Linked data on the web. In: Workshop at the 17th International World Wide Web Conference. Beijing (2008)
3. Davies, T., Edwards, D.: Emerging implications of open and linked data for knowledge sharing in development. IDS Bulletin **43**(5), 117–127 (2012)
4. Ding, L., DiFranzo, D., Graves, A., Michaelis, J., Li, X., McGuinness, D.L., Hendler, J.: Data-gov Wiki: Towards Linking Government Data. In: AAAI Spring Symposium: Linked Data Meets Artificial Intelligence, vol. 10, pp. 1–1 (2010)
5. Ding, L., Lebo, T., Erickson, J.S., DiFranzo, D., Williams, G.T., Li, X., Michaelis, J., Graves, A., Zheng, J.G., Shangguan, Z., et al.: Twc logd: A portal for linked open government data ecosystems. Web Semantics: Science, Services and Agents on the World Wide Web **9**(3), 325–333 (2011)
6. Goodwin, J., Dolbear, C., Hart, G.: Geographical linked data: The administrative geography of Great Britain on the semantic web. Transactions in GIS **12**(s1), 19–30 (2008)
7. Hartig, O., Zhao, J.: Publishing and consuming provenance metadata on the web of linked data. In: Provenance and Annotation of Data and Processes, pp. 78–90. Springer (2010)
8. Heath, T., Bizer, C.: Linked data: Evolving the web into a global data space. Synthesis lectures on the semantic web: theory and technology **1**(1), 1–136 (2011)
9. Jentzsch, A., Isele, R., Bizer, C.: Silk-Generating RDF Links while Publishing or Consuming Linked Data. In: ISWC Posters&Demos. Citeseer (2010)
10. Lemmens, R., Kessler, C.: Geo-information visualizations of linked data. In: Proceedings of the 17th AGILE Conference on Geographic Information Science,3-6 June 2014, Castelln, Spain (2014)
11. Loucopoulos, P., Karakostas, V.: System requirements engineering. McGraw-Hill, Inc. (1995)
12. Moon, S., Williamson, T.: Greater aid transparency: crucial for aid effectiveness. Overseas Development Institute project briefing (2010)
13. Rodriguez, M.A.: A graph analysis of the linked data cloud. arXiv preprint arXiv:0903.0194 (2009)
14. Capadisli, S.A., Riedl, R.: Linked statistical data analysis http://stats.270a.info/. Submission to the 2013 Semantic Web Challenge http://csarven.ca/linked-statistical-data-analysis (2013)
15. Shadbolt, N.: Why Open Government Data? lessons from data. gov. uk (2010)
16. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: The Semantic Web-ISWC 2009, pp. 650–665. Springer (2009)