# From XML to RDF step by step: Approaches for Leveraging XML Workflows with Linked Data

Marta Borriello, Vistatec <marta.borriello@vistatec.com>

Christian Dirschl, Wolters Kluwer Germany <cdirschl@wolterskluwer.de>

Axel Polleres <axel.polleres@wu.ac.at>

Phil Ritchie <philr@vistatec.ie>

Frank Salliau <frank.salliau@ugent.be>

Felix Sasaki <felix.sasaki@dfki.de>

Giannis Stoitsis <stoitsis@agroknow.com>

## Table of Contents

Introduction ..... 1
Motivation ..... 1
The Relation to RDF Chimera ..... 2
Background: The FREME Project ..... 3
Business Case Motivation Examples ..... 3
The Case of Agro-Know and Wolters Kluwer - Linked Data in XML Publishing Workflows ..... 3
The Case of Vistatec - Linked Data in XML Localization Workflows ..... 5
The Case of iMinds - Linked Data in Book Metadata ..... 6
Approaches for Linked Data Integration in XML Workflows ..... 7
Approach 1: Convert XML to Linked Data ..... 8
Approach 2: Embedd Linked Data into XML via Structured Markup ..... 9
Approach 3: Anchor Linked Data in XML Attributes ..... 9
Approach 4: Embed Linked Data in Metadata Sections of XML Files ..... 10
Approach 5: Anchor Linked Data via Annotations in XML Content ..... 10
Relating Business Cases and Integration Approaches ..... 11
Routes to bridge between RDF and XML ..... 11
Conclusion ..... 13
Acknowledgments ..... 13
Bibliography ..... 13

# Introduction

## Motivation

There have been many discussions about benefits and drawbacks of XML vs. RDF. In practice more and more XML and linked data technologies are being used together. This leads to opportunities and uncertainties: for years companies have invested heavily in XML workflows. They are not willing to throw them away for the benefits of linked data.

In XML workflows XML content is

- Generated, from scratch or based on existing content;
- processed, e.g.: validated, queried, transformed; and
- stored in various forms, e.g.: as XML, in a different format (e.g. PDF / HTML output); and
- potentially input to other XML or non-XML workflows.

Each part of the workflow may include huge amounts of XML data. This can be XML files themselves, but also additional related items like: XSLT or XSL-FO stylesheets for transformation or printing, XQuery based queries, or XML Schema / DTD / Relax NG schemata for validation etc.

For many potential users of linked data, giving up these workflows is not an option. Also, changing even a small part of the workflow may lead to high costs. Imagine one element `linkedDataStorage` added to an imaginary XML document:

### Example 1. XML document with linked data embedded in an element

```
<myData>
 <head>...</head>
 <body>
 <linkedDataStorage>...</linkedDataStorage> ...
 </body>
</myData>
```

Adding this element to an XML element may break various aspects of the workflow, like:

- Validation: the underlying schema does not understand the new element.
- Transformation: a transformation may expect a certain element as the first child element of body. If `linkedDataStorage` is the first child, the transformation would fail.

One may argue that good XML schema will leave space for expansion using lax validated parts or by accepting attributes from other namespaces, for example. Nevertheless, in practice we have to work with a lot of existing XML Schemas, related tooling and workflows. So creating extension points and deploying lax validation may not be an option in real life.

Whereas the strict validation against schemas on the one hand is in the XML world often seen as a feature of the toolchain, on the other hand, such adding of elements and schemaless integration of different (parts of) datasets is actually one of the main “selling points of RDF”. However, note that on the contrary, even in the RDF world, users are starting to demand tools for stricter schema validation, which has recently lead to the foundation of a respective working group around RDF Data Shapes in W3C.[1] So, overall there seems to be lots to learn for both sides from each other.

This paper wants to help with XML and RDF integration to foster incremental adoption of linked data, without the need to get rid of existing XML workflows. We are discussing various integration approaches. They all have benefits and drawbacks. The reader needs to be careful in deciding which approach to choose.

# The Relation to RDF Chimera

In her keynote at XML Prague 2012 and a subsequent blog post, Jeni Tennison discussed RDF chimera [http://www.jenitennison.com/2012/06/30/rdf-chimera.html]. She is arguing that for representing RDF, syntaxes like RDF/XML or JSON or JSON-LD should be seen as a means to achieve something - a road, but not a destination. An example is a query to an RDF data store, and the outcome is represented in an HTML page.

[1] See http://www.w3.org/2014/data-shapes/.

The goal of our paper is different. We assume that there is existing content that benefits from *data integration* with linked data - without turning the content into a linked data model. Let's look at an example: imagine we have the sentence `Berlin is the capital of Germany!`. There are many linked data sources like DBpedia [http://dbpedia.org/about] that contain information about `Berlin`; it would add an enormous value to existing content (or content creation workflows) if such information could be taken into account. This does not mean - like in the case of RDF chimera - to give up the XML based workflows, but to provide means for the data integration. In this view we can see the linked data process as a type of enrichment, hence we call the process *enriching XML content* with linked data based information.

# Background: The FREME Project

FREME[2] is an European project funded under the H2020 Framework Programme. FREME is providing a set of interfaces (APIs and GUIs) for multilingual and semantic enrichment of digital content. The project started in February 2015, will last for two years and encompasses eight partners. The partners provide technology from the realm of language and data processing, business cases from various domains, and expertise in business modeling. This expertise is of specific importance since both language and linked data technologies are not yet widely adopted. The challenge of XML re-engineering for the sake of linked data processing is one hindrance that needs to be overcome to achieve more adoption.

FREME provides six e-Services for processing of digital content:

- e-Internationalisation based on the Internationalisation Tag Set (ITS) Version 2.0.
- e-Link based on the Natural Language Processing Interchange Format (NIF) and linked (open) data sources.
- e-Entity based on entity recognition software and existing linked entity datasets.
- e-Terminology based on cloud terminology services for terminology management and terminology annotation web service to annotate terminology in ITS 2.0 enriched content.
- e-Translation based on cloud machine translation services for building custom machine translation systems.
- e-Publishing based on cloud content authoring environment (for example e-books, technical documentation, marketing materials etc.) and its export for publishing in Electronic Publication (EPUB3) format.

This paper will not provide details about the services - examples and more information on FREME can be found at http://api.freme-project.eu/doc/current/

All e-services have in common that XML content is a potential input and output format: via FREME, XML content can be enriched with additional information, to add value to the content. But FREME is only one example: many other linked data projects involve companies working with linked data content.

# Business Case Motivation Examples

## The Case of Agro-Know and Wolters Kluwer - Linked Data in XML Publishing Workflows

Agro-Know is data oriented company that helps organisations to manage, organise and open their agricultural and food information. One of the main activities of Agro-Know is the aggregation of bibliographic references from diverse sources to support online search services like AGRIS [http://agris.fao.org/] of the Food and Agricultural Organisation of the United Nations. Agro-Know is doing

[2]See the FREME project homepage at http://freme-project.eu/ for more information.

so by aggregating metadata records from data providers such as journals, small publishers, universities, research centers, libraries and national aggregators. The metadata aggregation workflow of Agro-Know includes parts for metadata analysis, harvesting, filtering, transformation, enrichment, indexing and publishing. The main goal of applying the different steps of the workflow is to end up with a well formated and complete metadata record that is compatible to the metadata standard for agricultural sciences, namely AGRIS AP [http://www.fao.org/docrep/008/ae909e/ae909e00.HTM]. The majority of the metadata records that are collected are in XML following several metadata formats such as DC, AGRIS AP, DOAJ, MODS, MARC 21 etc. The processed metadata records are published in AGRIS AP, JSON and RDF.

Within such metadata aggregation workflow, Agro-Know is facing several challenges related to the enrichment of the metadata records. One such example is non-structured information about authors that in many cases is not following an authority file and includes additional information in the same XML tag like affiliation, email and location. This information cannot be automatically transformed to structured information and remaining in the tag, it reduces the quality of provided filtering options in the search functionality of the online service. In addition to that, since an authority file is not used on the data provider side, this results in an ambiguity problem as the same author may appear with many different variations of the name. Another problem is the absence of subject terms in the metadata records from a multilingual vocabulary such as AGROVOC [http://aims.fao.org/vest-registry/vocabularies/agrovoc-multilingual-agricultural-thesaurus], that consists of more than 32.000 terms available in 23 languages. Including AGROVOC terms in the metadata records can semantically enhance the information and can enable better discovering services at the front end application.

To solve such problems, Agro-Know is using the FREME e-services in order to improve a) the online services that are offered to the end users and b) the semantics of the metadata records that is provided to other stakeholders of this data value chain, such as publishers. The main goal will be to add the structured information in the XML records by keeping the level of intervention at a minimum level in order to eliminate the revisions required in the existing workflows. Examples of how an XML part referring to authors can be enriched using FREME e-services is presented in the table below. In this case, including the ORCID [http://orcid.org/] identifier may help in disambiguation but also in the enrichment of the information as we can show to the end user of the online service additional valuable information directly retrieved from ORCID.

| **Before FREME** | **Result of deploying FREME** |
|---|---|
| `<dc:creator>`<br>`<ags:creatorPersonal>`<br>`Stoitsis, Giannis,`<br>`Agroknow`<br>`</ags:creatorPersonal>`<br>`</dc:creator>` | `<dc:creator>`<br>`<ags:creatorPersonal>Stoitsis,`<br>`Giannis</ags:creatorPersonal>`<br>**`<nameIdentifier schemeURI=`**<br>**`"http://orcid.org/"`**<br>**`nameIdentifierScheme=`**<br>**`"ORCID">0000-0003-3347-8265`**<br>**`</nameIdentifier>`**<br>**`<affiliation>Agroknow</affiliation>`**<br>`</dc:creator>` |
| `<dc:subject>`<br>`<ags:subjectClassificati`<br>`scheme="ags:ASC">`<br>`<![CDATA[J10]]>`<br>`</ags:subjectClassificatio`<br>`</dc:subject>` | `<dc:subject`<br>**`freme-enrichment=`**<br>**`"http://aims.fao.org/aos/agrovoc/c_426`**<br>**`http://aims.fao.org/aos/agrovoc/c_24135`**<br>**`http://aims.fao.org/aos/agrovoc/c_4644`**<br>**`http://aims.fao.org/aos/agrovoc/c_7178">`**<br>`<ags:subjectClassification scheme=`<br>`"ags:ASC"><![CDATA[J10]]>`<br>`</ags:subjectClassification>`<br>`</dc:subject>` |

Wolters Kluwer is a global information service provider with businesses mainly in the legal, tax, health and financial market. The fundamental transformation process from folio to digital and service offerings that is currently disrupting the whole industry requires also more efficient and streamlined

production processes. In the last ten years, the industry has very much focused on XML based publishing workflows, where all relevant information resides as metadata within the documents, which are structured according to proprietary DTDs or XML schemas. The industry is slowly starting to adapt linked data principles, because they offer the required flexibility, scalability and information interoperability that XML or also relational database models do not offer. As a first step, metadata is extracted from the documents and stored in parallel in graph databases. Already this step requires a major shift in technology as well as business culture, because the focus and added-value moves away from pure content to data and information.

Wolters Kluwer Health is customer of Agro-know and has integrated its XML delivery channel for enriched scientific references mainly in the area of agriculture. Agro-know is offering more and more added value services using linked data principles and in this course reduces the traditional XML-based delivery pipeline step by step in order to stimulate usage of the superior channels. This change causes major challenges at the customer's side. Semantic Web technology and standards are not yet common solutions in publishing. Therefore technical infrastructure as well as skills have to be developed in order to get things even started. This requires a certain transition period, where the old delivery channel remains stable and the customer can prepare the changes.

In such a scenario, Wolters Kluwer recommends that the source provider enables the customer to locally keep his old production workflow in place as long as it is needed. This could be achieved e.g. by making the conversion script available as open source. In addition, a proper documentation about the differences from old to new is also vital for success. Ideally, a direct communication flow between vendor and customer would help to lower concerns and accelerate uptake of the new process.

# The Case of Vistatec - Linked Data in XML Localization Workflows

Vistatec is a leading provider of digital content translation and locale adaptation services for international businesses. These businesses use a variety of Vistatec multilingual services to publish a range of content types including: corporate communications; marketing collateral; e-commerce catalogues; and product, travel destination, and leisure experience descriptions.

Vistatec's production processes are highly automated and based on XML standards for systems integration, process component interoperability and the capture and use of metadata.

The localization process, content types and end consumers of the content all benefit greatly from FREME semantic enrichment and entity discovery and linking e-services.

The successful adoption of technology hinges upon the ease of use. Vistatec has adapted its open source XLIFF [https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xliff] editor, Ocelot [http://open.vistatec.com/ocelot/index.php?title=Main_Page], to consume FREME e-services in a transparent and optimal way using configurable pipelines.

The table below summarizes the steps of a typical localization workflow and the benefits that can be realized through the use of FREME e-services:

| **Process Step** | **FREME e-service** | **Benefit** |
|---|---|---|
| Conversion of native document to Extensible Localization Interchange File Format | e-Internationalization | Define translatable portions of the document. |
| Translation | e-Terminology and e-Entity | These services help linguists to identify and use appropriate translations suitable for the subject domain. |
| Semantic enrichment | e-Link | Suggest information resources which relate to the subject matter of the content. |

| | | |
|---|---|---|
| Content publication | e-Pub | Incorporation of markup for entity definitions and hyperlinks to information resources which relate closely to the subject matter of the content. |

A tutorial from the FREME documentation[3] shows how XLIFF can be processed via FREME e-Services. We process the following XLIFF element, using the example sentence from a previous section:

```
<source>Berlin is the capital of Germany!</source>
```

The e-Entity service identifies `Berlin` as a named entity with a certain type and a unique URI. The result of the process is not stored in XML, but as a linked data representation including **offsets** that point to the string content. The URI

```
<http://freme-project.eu/#char=25,32>
```

identifies the entity, offsets and entity related information. A complete linked data representation, using the turtle syntax, looks as follows.

### Example 2. Linked data representation using offsets for pointing to existing XML content

```
<http://freme-project.eu/#char=25,32> ...
(1) nif:anchorOf          "Germany"^^xsd:string ;
(2) nif:beginIndex        "25"^^xsd:int ;
(3) nif:endIndex          "32"^^xsd:int ; ...
(4) itsrdf:taClassRef     <http://nerd.eurecom.fr/ontology#Location>;
(5) itsrdf:taIdentRef     <http://dbpedia.org/resource/Germany>.
```

The linked data statements expressed in above representation are:

- The annotation is (1) anchored in the string Germany.
- The annotation (2) starts at character 25 and (3) ends at character 32.
- The annotation is related to (4) the class URI http://nerd.eurecom.fr/ontology#Location.
- The entity is (5) uniquely identified with the URI http://dbpedia.org/resource/Germany.

The example should make clear why we are not looking at a data conversion task (like in the discussion on RDF chimera), but at a data integration task. We don't aim at changing the XLIFF source element, but at relating it to the information provided via the linked data representations. The data integration approaches discussed in the section called "Approaches for Linked Data Integration in XML Workflows" are ways to achieve this goal.

# The Case of iMinds - Linked Data in Book Metadata

iMinds, Flanders' digital research hub, conducts research on book publishing in close collaboration with the Flemish publishing industry. An important aspect in book publishing is book metadata. As the volume in published books increases, and as the book industry becomes more and more digital, book metadata also becomes increasingly important: book publishers want their books to be found on retail sites. Correct and rich metadata is a prerequisite for online discoverability.

Within the publishing value chain, stakeholders have since long agreed to use a standard format for book metadata: ONIX for Books[4]. This XML-based standard has been adopted worldwide and is used by publishers to communicate book product information in a consistent way in electronic form.

[3] See http://api.freme-project.eu/doc/0.4/tutorials/spot-entities-in-xliff.html
[4] See http://www.editeur.org/83/Overview/

ONIX for Books is developed and maintained by EditEUR[5], the international group coordinating development of the standards infrastructure for electronic commerce in the book, e-book and serials sectors. The ONIX for Books standard and corresponding workflow are solidly embedded in the publishing retail chain, from publisher to distributor to retailer. Migrating to other technologies, e.g. using linked data, requires a substantial investment which stakeholders in this industry are very reluctant to make.

We do not recommend to substitute this standard with a fully fledged linked data approach. However, we find there are cases where linked data can be beneficial as enrichment of existing ONIX metadata. The example below shows a possible usage of linked data in ONIX.

**Author information as linked data**: An ONIX file typically contains information about the author(s) of a book. These are called contributors (other types of contributors are illustrators, translators etc.).

Below you can find a block of metadata with information about the author of a book, Jonathan Franzen. A possible enrichment with linked data might be to insert the link to the authority record about Jonathan Franzen on viaf.org, via the Entity tag. Please note that these tags are not valid within the current ONIX schema and are used here merely as an example of a possible enrichment.

This enrichment may prove useful in several ways:

- disambiguation of the author (using the VIAF identifier); and
- providing a link to more information on the author.

### Example 3. A potential approach for embedding linked data identifiers into ONIX

```
<Contributor>
 <NameIdentifier>
  <NameIDType>
   <IDTypeName>Meta4Books ContributorID</IDTypeName>
   <IDValue>65097</IDValue>
  </NameIDType>
 </NameIdentifier>
 <ContributorRole>A01</ContributorRole>
 <SequenceNumber>1</SequenceNumber>
 <NamesBeforeKey>Jonathan</NamesBeforeKey>
 <KeyNames>Franzen</KeyNames>
 <Entity>
  <URI>http://viaf.org/viaf/84489381/</URI>
 </Entity>
</Contributor>
```

# Approaches for Linked Data Integration in XML Workflows

The following approaches are a non exhaustive list. The aim is to provide examples that are currently in use and show their benefits and drawbacks. The examples are anchored in the business cases described in the section called “Business Case Motivation Examples”. The structure of the approach description is always as follows:

- Name the example;
- Explain what actually happens with some XML code snippets; and

[5] See http://www.editeur.org/.

- Explain drawbacks and benefits, both from a linked data and an XML processing point of view.

# Approach 1: Convert XML to Linked Data

**What actually happens:** XML is converted into linked data. The XML content itself is not touched, but an additional set of data, i.e. a linked data representation is created.

```
<xs:element name="lingualityType">
 <xs:simpleType>
  <xs:restriction base="xs:string">
   <xs:enumeration value="monolingual"/>
   <xs:enumeration value="bilingual"/>
   <xs:enumeration value="multilingual"/>
 </xs:restriction>
</xs:simpleType>
</xs:element>
```

During the mapping of XML to linked data several decisions have to be taken. Sometimes information is being lost. A real example is the mapping of META-SHARE to linked data[6]. META-SHARE provides metadata for describing linguistic resources.

Using this element could look like this, expressing that a resource is multilingual, e.g. a multilingual dictionary:

```
<lingualityType>multilingual</lingualityType>
```

A linked data model that represents this data type could look as follows:

### Example 4. Linked data model that represents parts of the META-SHARE schema

```
### Class
ms:linguality
    rdf:type owl:ObjectProperty ;
    rdfs:domain ms:Resource ;
    rdfs:range ms:Linguality .
### Property
ms:Linguality
    rdf:type owl:Class .
#### Instances
ms:monolingual a ms:Linguality.
ms:bilingual a ms:Linguality.
ms:multilingual a ms:Linguality.
```

This statement expresses the same information like in the XML representation: a given resource, e.g. a dictionary, is multilingual.

**What are the benefits:** The benefit of this approach is that existing XML workflows don't need to be changed at all. The linked data representation is just an additional publication channel for the information. In this sense, the approach is similar to the RDF chimera discussion. It is however still different, since the conversion from XML to RDF involves RDF focused data modeling and aims at integration with further linked data sources.

The additional RDF representation can be integrated with other linked resources without influencing the XML workflow. This is what actually is being done with the META-SHARE case: via the LingHub

[6]See http://www.lrec-conf.org/proceedings/lrec2014/pdf/664_Paper.pdf for more details on the META-SHARE case and on how the conversion from XML to linked data was achieved.

portal[7], META-SHARE and other types of metadata for linguistic resources is converted to RDF and being made available. A user then can process integrated, multiple linked data sources without even knowing that there is an XML representation available.

**What are the drawbacks:** The approach requires a completely new tool chain, aiming at linked data integration based on XML (or other format based) data sources. The existing setup, e.g. used to analyze the XML structures with XQuery or to transform it via XSLT, cannot be used. A query like "Give me all linguistic resources that are multilingual" can be executed via XPath easily in the XML representation. In standard XQuery there is no bridge to SPARQL. However, work has been done to create this bridge, see the section called "Routes to bridge between RDF and XML".

Another drawback of this approach is that it is not always possible to convert XML completely into RDF - in particular, RDF has no facility for representing mixed content, an essential part of processing textual, human language content. A takeaway of the section called "The Relation to RDF Chimera" is that data should always be in the format that best fits its representation, and URIs can serve as a bridge between formats. The usage of URIs for bridging between RDF and XML is discussed in the section called "Approach 5: Anchor Linked Data via Annotations in XML Content".

# Approach 2: Embedd Linked Data into XML via Structured Markup

**What actually happens:** linked data is embedded into HTML. Various syntaxes can be used, e.g. JSON-LD, RDFa, or microdata. This approach is deployed e.g. in schema.org [http://schema.org/]; see the schema.org homepage for various markup examples.

We take again the sentence `Berlin is the capital of Germany` from a previous section. The integration of information from Wikipedia with the string `Berlin` can be achieved as follows:

```
<a itemscope itemtype="http://schema.org/Place" itemprop="url"
 href="https://en.wikipedia.org/wiki/Berlin">Berlin</a>
```

With this approach search engines will interpret the link `https://en.wikipedia.org/wiki/Berlin` as a machine readable link to integrate additional information about the place Berlin, taken from Wikipedia as a data source. The item being the source of data integration is identified as being of type place via the URI `http://schema.org/Place`.

**What are the benefits:** the approach works well in situation in which the hooks for data integration can be embedded into XML or HTML content. The search engine optimization scenario via schema.org is the prototypical case for doing this. The embedding may also be done in a dedicated markup part for metadata using the JSON-LD or other linked data syntaxes, without changing the text content; see for details the section called "Approach 4: Embed Linked Data in Metadata Sections of XML Files".

**What are the drawbacks:** RDFa and Microdata changes the content and includes new markup. That may not be an option for XML tool chains that don't "understand" the new markup, e.g. lead to validation errors. JSON-LD or turtle may have the same issues: where should a tool store this data in the XML structure if no general metadata location is available?

# Approach 3: Anchor Linked Data in XML Attributes

**What actually happens:** an identifier is embedded in the XML structure. This identifier serves as a bridge between the XML and RDF structures. The below example uses the `its:taClassRef` attribute to store the identifier.

```
<source ...>
 <mrk ...its:taIdentRef="http://dbpedia.org/resource/Berlin">
  Berlin</mrk> is the capital of Germany!</source>
```

[7]See http://linghub.lider-project.eu/.

The data integration task, i.e. fetching additional information from linked data sources about `Berlin`, can be executed relying on this information. The outcome may then be stored directly in the XML source. Below we assume that the population was fetched using the DBpedia information.

```
<source ...>
 <mrk ...its:taIdentRef="http://dbpedia.org/resource/Berlin">
  Berlin</mrk> (population: 3517424)...</source>
```

For different purposes, separate linked data queries could be set up. They rely on the same identifier `http://dbpedia.org/resource/Berlin`.

**What are the benefits:** using an XML attribute that is already available in the format in question means that no new types of markup is needed. That is, existing XML toolchains can stay as is, including validation or transformation processes.

**What are the drawbacks:** the data integration is postponed. The completed integration, if needed, needs to choose one of the other approaches discussed in this paper. Also, the data integration does not leave a trace. Further processing steps in the (XML) toolchain cannot identify that the string `(population: 3517424)` is a result of a data integration process.

# Approach 4: Embed Linked Data in Metadata Sections of XML Files

**What actually happens:** many XML vocabularies have metadata sections that may contain arbitrary content. This is also true for XLIFF discussed in the section called “The Case of Vistatec - Linked Data in XML Localization Workflows”. The outcome of the linked data processing could be stored in such a section.

**What are the benefits:** Compared to the section called “Approach 2: Embedd Linked Data into XML via Structured Markup”, the size of the content itself is not growing with additional, linked data related markup.

**What are the drawbacks:** There is no per se relation to the content. Like in Example 2, “Linked data representation using offsets for pointing to existing XML content”, one may create pointers to the XML content, here using character offsets. But the pointers may be fragile, if one thinks e.g. of reformatting, insertion or deletion of content or markup. In addition, some linked data syntaxes may interfere with XML validation or well formedness constraints.

# Approach 5: Anchor Linked Data via Annotations in XML Content

**What actually happens:** A generalized approach of the section called “Approach 3: Anchor Linked Data in XML Attributes” means that linked data is stored separately from XML structures and that there is a reference from linked data to the XML content in question. In the section called “The Case of Vistatec - Linked Data in XML Localization Workflows”, we were using character offsets. The W3C Web Annotation Data Model [http://www.w3.org/TR/2015/WD-annotation-model-20151015/] allows to realize such anchoring. Character offsets are just one way of anchoring the annotation. One can also use XPath expressions, see the following example.

### Example 5. Anchoring annotations in XML via the Web annotation data model

```
{  "id": "http://example.com/myannotations/a1",
 "type": "Annotation",
 "target": { "type": "SpecificResource",
  "source": "http://example.com/myfile.xml",
  "selector": { "type": "FragmentSelector",
   "conformsTo": "http://www.w3.org/TR/xpath/",
```

```
 "value": "/xlf:unit[1]/xlf:segment[1]/xlf:source/xlf:mrk[1]" },
"itsrdf:taIdentRef": "http://dbpedia.org/resource/Berlin",
"itsrdf:taClassRef": "http://schema.org/Place",
"http://dbpedia.org/property/population" : "3517424"  } }
```

The XPath expression in above linked data representation (which uses the JSON-LD syntax) selects the XLIFF mrk element from the example in the section called “Approach 3: Anchor Linked Data in XML Attributes”.

**What are the benefits:** In addition to the approach 3, here we are able to add the linked data information in the separate annotation, e.g. the population of Berlin; there is no need to change the content itself. If needed for certain applications, we can use this annotation approach to generate others. URIs pointing to the content are an important aspect of such format conversions. THe forehand mentioned ITS 2.0 specification shows an example of 1) generating linked data annotations anchored in XML content [http://www.w3.org/TR/its20/#conversion-to-nif], and 2) integrating the separate annotations into the markup content [https://www.w3.org/TR/its20/#nif-backconversion]. The forehand described FREME framework deploys this approach in its ITS enabled e-Internationalisation [http://api.freme-project.eu/doc/0.4/knowledge-base/eInternationalization.html].

**What are the drawbacks:** the resolution of linked data information potentially can be computationally expensive, see e.g. lot’s of XPath expressions to compute for annotations. Also, if the source content changes, the anchoring mechanism may not work anymore. Some mechanisms are more robust (e.g. XPath expressions), some may be more precise (e.g. the character offset based anchoring).

# Relating Business Cases and Integration Approaches

The following table relates the three business cases and the various integration approaches.

| **Business case** | **Integration approaches being considered** | **Actual current or experimental praxis** |
|---|---|---|
| Linked data in XML publishing workflows | Approach 1: convert XML into linked data | XML workflow kept, linked data conversion scripts to be made available |
| Linked data in XML localization workflows | Approach 3: anchor linked data in XML attributes; Approach 4: embed linked data in metadata sections of XML files | No established practice in localisation industry |
| Linked data in book metadata | Approach 4: embed linked data in metadata sections of XML files | No established practice in localisation industry |

It becomes obvious that industries take different approaches towards linked data integration. This can be explained with availability of native linked data tooling, knowledge about its usage, and complexity and potential costs of adapting existing XML workflows.

# Routes to bridge between RDF and XML

As for **Approach 1** (converting XML into linked data), in fact existing XML transformation tools like XSLT and XQuery could be used out of the box with the caveat that the result is mostly tied to the RDF/XML representation, that has various disadvantages, foremost verbosity. More "modern" RDF serializations like Turtle or JSON-LD cannot be created out of the box by XML tools straightforwardly, plus additional filtering of querying on the resulting RDF triples needs to be encoded directly into the XML toolchain, which might be easier solvable in the RDF world itself, e.g. using SPARQL. Likewise, as for **Approach 2**, we have already identified, that the XML toolchain is not tailored to process and

consume RDF or similar meta-data formats natively. We face the same problem in **Approaches 3-5**, where RDF-like sources are just linked out of XML content, without the necessary toolchain tightly coupled to XML tools that could process the RDF content natively.

So, there is certainly a gap to bridge here in terms of tooling, but recently, that partially seems to change: there are academic approaches to encode SPARQL into XML processors, such as encoding SPARQL to XSLT or XQuery, cf. e.g. Fischer et al. (2011) and Groppe et al. (2008). Plus there are actually some XML processors like SAXON start supporting SPARQL natively, cf. https://developer.marklogic.com/learn/semantics-exercises/sparql-and-xquery.

Alternatively, given that SPARQL actually can produce XML or JSON (among others) as output format[8], it is possible to directly consume the results of SPARQL queries in XML tools, however more complex use cases need some scripting around this, plus intermediate results for an overall transformation need to be stored and processed separately, ending up in a heterogeneous toolchain, comprising XML tools, SPARQL processors and potentially even another scripting language on top.

Additionally, there are new "hybrid" but integrated toolchains arising that try to combine the two worlds of XML and RDF in a "best-of-both-worlds" approach: most prominently, we'd like to mention as an example here the XSPARQL project[9], that aims at integrating SPARQL into XQuery in a compilation approach: that is, queries on RDF data or to a remote SPARQL endpoint serving Linked Data can be embedded into an XQuery. For instance, the following query transforms geographic RDF data queried from the file http://nunolopes.org/foaf.rdf into a KML file:

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>

<kml xmlns="http://www.opengis.net/kml/2.2">{
 for $name $long $lat from <http://nunolopes.org/foaf.rdf>
 where { $person a foaf:Person; foaf:name $name;
         foaf:based_near [ a geo:Point;
         geo:long $long;
         geo:lat $lat ] }
 return <Placemark>
  <name>{fn:concat("Location of ", $name)}</name>
   <Point>
    <coordinates>{fn:concat($long, ",", $lat, ",0")}</coordinates>
   </Point>
 </Placemark> }
</kml>
```

The boldface part of the above query is actually SPARQL syntax embedded into XQuery. An XSPARQL compiler translates this query to a native XQuery that delegates these query parts to a native SPARQL query processor. More details on this approach can be found in Bischof et al. (2012).

Note that also the other way, i.e. transforming XML data into RDF is supported in this approach, by allowing SPARQL's CONSTRUCT clauses in the return claus of an XQuery, as shown in the following short example, which transforms XML data from Open Streetmap to RDF:

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
prefix kml: <http://earth.google.com/kml/2.0>
let $loc := "WU Wien" for $place in doc
  (fn:concat("http://nominatim.openstreetmap.org/search?q=",
    fn:encode-for-uri($loc),
    "&amp;format=xml"))
let $geo := fn:tokenize($place//place[1]/@boundingbox, ",")
construct { <http://www.polleres.net/foaf.rdf#me>
```

[8] See http://www.w3.org/TR/sparql11-overview/#sparql11-results for details.
[9] See http://xsparql.sourceforge.net for details.

```
foaf:based_near [ geo:lat {$geo[1]}; geo:long {$geo[3]} ] }
```

More recently, XSPARQL has been extended to cover the recent SPARQL1.1 specification, plus support for JSON as input and output format (by internally representing JSON in a canonical XML representation have been added, cf. Dell'Aglio et al. (2014).

While XSPARQL, which has actually started as a W3C member submission in 2009[10], is just one possible route, the authors believe that joint efforts in standardization bodies to bridge the gaps between RDF and XML in order to enable such transformations and integrated tooling in a standard way should be further pursued.

# Conclusion

This paper discussed the motivation for integrating RDF and XML. We looked at various business case scenarios that can benefit from this integration. We then discussed several approaches to realize the integration. Finally, we looked into technical solutions that integrate the actual XML and RDF technology stacks.

A reviewer of this paper suggested consider XProc for integrating RDF and XML workflows. XProc 2.0 [https://www.w3.org/TR/xproc20/] will have the ability to pass information other than XML data between steps; it would be possible to pass RDF data between XProc steps and have filtering and processing steps for that RDF data. This would allow processing of XML data with XML tools (XSLT, XQuery), while tracking and also processing RDF data with e.g. SPARQL or XSPARQL. This approach sounds promising but has not been explored in this paper, so we leave it to future work.

A next steps could be to discuss the integration approaches in a broader community, e.g. in a dedicated forum like a W3C community group. This could also help to move the forehand described XML - RDF standardization work forward. Such standardization has been discussed in the past. It is the hope of the authors of this paper that it brings new insights to this discussion, with the real-life needs from actual applications who more and more are in need of the integration.

# Acknowledgments

The creation of this paper was supported by the FREME project under the Horizon 2020 Framework Programme of the European Commission, Grant Agreement Number 644771.

# Bibliography

Bischof, S., S. Decker, T. Krennwallner, N. Lopes and A. Polleres. Mapping between RDF and XML with XSPARQL. Journal on Data Semantics (JoDS), 1(3):147-185, 2012.

Dell'Aglio, D., A. Polleres, N. Lopes and S. Bischof. Querying the web of data with XSPARQL 1.1. In ISWC2014 Developers Workshop, volume 1268 of CEUR Workshop Proceedings. CEUR-WS.org, October 2014.

Fischer, P., D. Florescu, M. Kaufmann and D. Kossmann D (2011). Translating SPARQL and SQL to XQuery. In: Proceedings of XML Prague'11, pp 81–98.

Groppe S., J. Groppe, V. Linnemann, D. Kukulenz, N. Hoeller, C. Reinke (2008). Embedding SPARQL into XQuery/XSLT. In: SAC'08. ACM, New York, pp 2271–2278.

[10]See http://www.w3.org/Submission/2009/01/.