A Model to Support Interpretation of Embedded Metadata without Formal Schema by Linking a Metadata Instance to DCMI Description Set Profiles

Tsunagu Honma
Graduate School of Library, Information and Media Studies, University of Tsukuba, Japan
tsuna@slis.tsukuba.ac.jp

Mitsuharu Nagamori
Faculty of Library, Information and Media Science, University of Tsukuba, Japan
nagamori@slis.tsukuba.ac.jp

Shigeo Sugimoto
Faculty of Library, Information and Media Science, University of Tsukuba, Japan
sugimoto@slis.tsukuba.ac.jp

Abstract

There are a number of HTML documents that include metadata on the Web and a number of information services that provide metadata embedded using metadata standards across domains. Those metadata are, however, encoded in various different schemas and in different serialization formats, which makes it hard to automatically extract and interpret. The primary reason of the difficulty is the lack of interpretation rules for metadata, e.g., lack of definition of metadata vocabularies, lack of definition of encoding syntax and so forth. This paper proposes a model to support interpretation of embedded metadata without formal schema by linking a metadata instance to DCMI Description Set Profiles (DSP). An XPath Expression addresses a metadata instance encoded in HTML, and DSP define metadata schema. We propose extending DSP to include XPath for linking a metadata instance to a metadata schema. This paper also shows an experimental system that extracts metadata using extended DSP.

Keywords: Metadata Extraction; Metadata Interpretation; Metadata Schema; DCMI Application Profile

1. Introduction

A huge amount of metadata in various domains and purposes is produced and delivered over the Internet and Web. Network users may use metadata as it is and, more importantly, they can use the metadata in combination with other metadata in order to acquire new information, to create a new service. For example, metadata is described and published as Linked Data (Tim Berners-Lee, 2009) by a wide variety of communities (e.g. DBpedia). Linking between resources connects metadata mutually, and there are already new information services realized by combining those metadata embedded in HTML documents. Each community has various metadata that are designed and encoded for their own purpose, which means that metadata may differ from community to community. In order to use some metadata in combination with others, computers need to know differences in metadata description rules or encoding rules. For that purpose, it is necessary to acquire the metadata schema used for creation of a metadata instance.

There are standards for encoding and embedding metadata in HTML Documents, for example RDFa,[1] Microdata,[2] etc. Those standards define rules to encode metadata instances, and define formats for embedding metadata in an HTML document. Computers can interpret metadata embedded according to these standards. However, on the Web, metadata is often encoded in an individual format, and embedded into HTML documents by web-content creators without explicit

[1] http://www.w3.org/TR/rdfa-syntax/
[2] http://www.w3.org/TR/microdata/

https://doi.org/10.23106/dcmi.952135962

reference to the schema being used. In many cases where such individual formats are used, it becomes difficult to automatically find the interpretation rules of the metadata.

In this paper, we propose a model for the interpretation of embedded metadata without formal schema. By connecting metadata instances and metadata schema, embedded metadata, which is addressed by an XPath Expression,[3] is interpreted using DCMI Description Set Profiles (Nilsson, 2008), including constraints of description resources and properties.

The rest of this paper is organized as follows. Section 2 discusses metadata interpretation using metadata schema. Section 3 describes a workflow of metadata creation and indicates a gap between embedded metadata and the relevant metadata schema. Section 4 proposes a model for interpretation of embedded metadata without formal schema by linking an embedded metadata instance to DCMI Description Set Profiles. Section 5 shows a system that extracts metadata from HTML documents using our model. Finally, we present discussion and a conclusion in Section 6.

2. Implicit Links between Embedded Metadata and Metadata Schema

Many HTML documents available on the Web include embedded metadata. For example, newspaper companies put metadata in their articles on the Web, such as the date of issue, keywords and a name of the author(s). Users who access or browse those web pages use embedded metadata to find, summarize, classify, and understand those web pages.

In order for computers to use the metadata in such HTML documents, it is necessary to extract and interpret the metadata. For extracting and interpreting metadata, it is necessary to have definitions (1) of which information is embedded, and (2) where it is located in HTML documents.

Those definitions are often used implicitly within metadata creation applications, and it is difficult to guess those definitions from embedded metadata. A metadata schema defines that an article has a title, and a web page has the value of title included in the HTML element "h1". However, constraints on the title in the metadata schema are not linked to the embedded metadata explicitly. If we can connect the embedded metadata to the metadata schema, which defines what information is embedded, computers will be able to use the embedded metadata.

Metadata schema registries—(Nagamori, 2011), The Dublin Core Metadata Registry,[4] Open Metadata Registry[5]—exist as services that support sharing of metadata schemas on the Internet and promote reuse of metadata schemas and metadata interoperability. It is possible to share definitions of what information is embedded, so we are hoping to increase the shareability of metadata schema.

There are standard formats for encoding and embedding metadata in HTML documents. In these standards (e.g. RDFa, Microdata (Schema.org), Microformats), there are rules for how to extract metadata, embedded into HTML documents using standard formats, suitable for use by computers. On the other hand, Peter Mika reports (2010) that less than 10% web pages that have embedded metadata follow standard formats. Metadata are embedded in web pages but encoded without following any metadata standards. Such metadata is hard for computers to use because it lacks interpretation rules or extraction procedures.

[3] http://www.w3.org/TR/xpath/
[4] http://dcmi.kc.tsukuba.ac.jp/dcregistry/
[5] http://metadataregistry.org/

https://doi.org/10.23106/dcmi.952135962

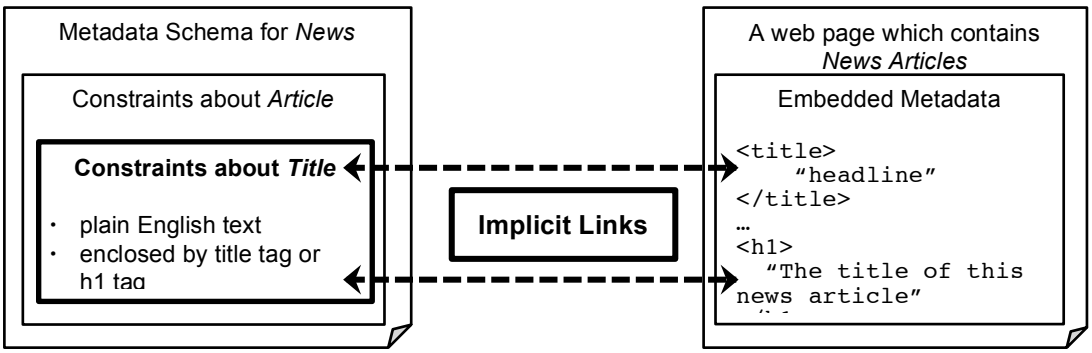

FIG. 1. Implicit Links between Embedded Metadata and Metadata Schema

For example, Figure 1 shows implicit links between a news article on the Web with embedded metadata and its metadata schema. The left part of Figure 1 shows that "Article title should be enclosed by h1 tag, and title text should be in plain English", which is a scheme designed for this document (or the class of this document) by its creator in order to encode metadata in the article shown on the right. If the schema is missing or not properly linked to the metadata embedded in the article, it is hard for computers to identify the title text in the article. In some cases, the embedded metadata is encoded in a standard format using standard metadata vocabulary but in many cases the metadata is encoded in a private format using a non-standard vocabulary. Thus, in many cases there are implicit links between the metadata schema and metadata embedded in Web resources that make it hard for third party to use the metadata.

In this research, we aim to increase computers' potential for extracting, interpreting and using the metadata in HTML documents by linking a metadata schema and embedded metadata explicitly.

3. A Model of Metadata Creation Workflow and the Gap to Metadata Practice

This section shows a model of workflow for creating metadata for embedding Web pages with Dublin Core Application Profile (DCAP), and discusses problems found in metadata practices based on the model. The problems are analyzed in order to design a support tool for interpreting embedded metadata.

3.1 A Model of Workflow for Metadata Creation with Dublin Core Application Profile

As we described in section 2, for computers to use metadata embedded in HTML documents, it is necessary to describe the relationship between the metadata schema and the embedded metadata. Figure 2 shows a model of workflow for creating and embedding metadata with DCAP. A Guideline for DCAP proposes that a metadata schema is defined with following steps,

1) Defining Functional Requirements: Defining clear goals for the application,
2) Selecting or Developing a Domain Model: Defining what things your metadata will describe, and the relationships between those things,
3) Selecting or Defining Metadata Terms: Choose properties for describing the things in a domain model,
4) Designing the Metadata Record with a Description Set Profile (DSP): Describing the metadata record's design in detail,
5) Usage Guidelines: How the metadata creator describes metadata based on Description Set Profile, and
6) Syntax Guidelines: Defining how encode metadata for machine-readability.

https://doi.org/10.23106/dcmi.952135962

Metadata instances and HTML documents are built on DCAP. Figure 2 also shows relationships between DCAP, metadata instances and HTML documents contained metadata instances. A metadata creator describes metadata instances in accordance with DSP that defines property and/or value constraints. For example, when DSP defines "A title of book is described as dc:title", a metadata creator who will describe metadata about a book *"Gone With the Wind"* , this can be described by the following RDF triple,

<urn:isbn:9781451635621> <dc:title> "Gone With the Wind".

A Metadata creator embeds metadata instances to HTML documents using Syntax Guidelines, which are shown in figure 2, defines how people serialize and embed metadata in HTML, XML etc. Typical examples of Syntax Guidelines are DC-HTML and DC-XML.

On the other side, when we extract and interpret metadata that is described based on a workflow of metadata creation with DCAP, we need the following types of definitions,

- Type-1: Definitions of how metadata instances are serialized and embedded in HTML documents.
- Type-2: Definitions of constraints for metadata instances. For example, "values of dc:creator are interpreted as persons who are the authors of book".

Type-1 definitions are required for detecting what kind of metadata instances is embedded where. Computers cannot separate metadata instances from the other strings in HTML documents with no rules on how metadata creators are embedded. Even if computers separate metadata instances from each other, they must be able to detect which string is a property and which string is a value. The rules of metadata serializing and embedding are useful for satisfying these requirements for extracting metadata from HTML documents. Type-2 definitions are needed for interpreting metadata instances. For example, "dc:title" is used for describing the titles of resources, but the meaning of "title" is different from in other applications. When using computers to interpret metadata, it is necessary to have a clear definition for each property and value in the application. These two types of definitions are also required when computers extract and interpret metadata. However, they are not explicit in HTML documents for computers. Syntax Guidelines are an especially important component for extracting metadata from HTML documents.

The workflow described above shows an ideal case for metadata creation and embedding into an HTML text. To extract and interpret those metadata instances, it is necessary to provide DSP and Syntax Guidelines for computers. Unfortunately, creators of HTML documents rarely follow the ideal model. Some metadata creators use implicit and informal rules for metadata creation and embedding because learning rules and guidelines to create metadata can be costly. Figure 3 shows an example of metadata instances embedded in an HTML document. In this case, there are some implicit rules for embedding metadata instances, (1) keywords are embedded as a comma separated text in a 'content' attribute of 'meta' tag which has a value 'keywords' in 'name' attribute, (2) the document title is tagged using the 'title' and 'h1' tags, (3) publishing date is embedded in the 'div' tag which has a class attribute 'published'. All of these metadata are hard to automatically and appropriately extract from the HTML text.

https://doi.org/10.23106/dcmi.952135962

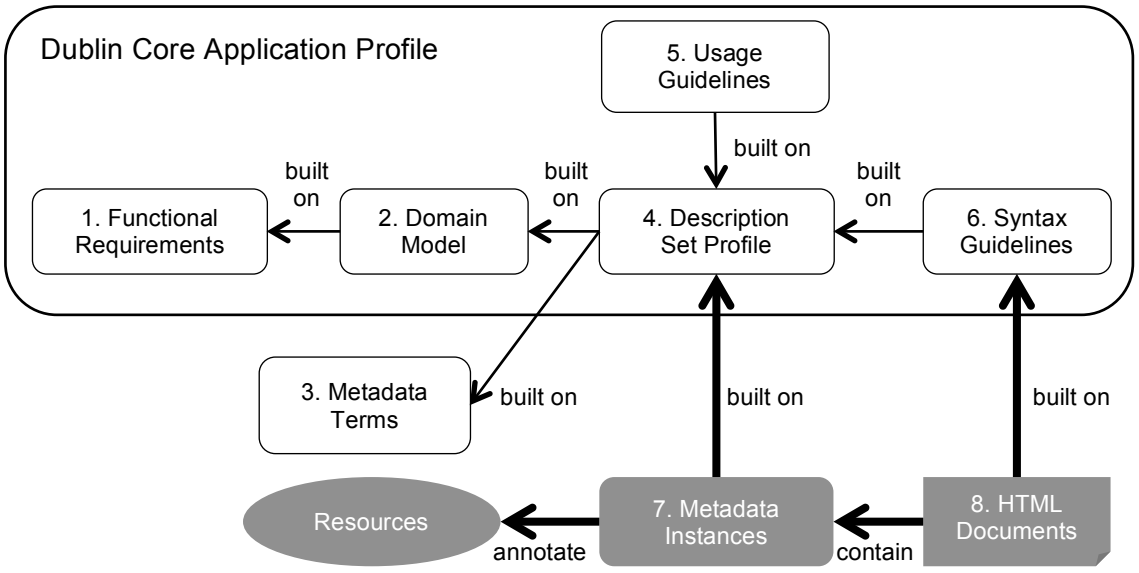

FIG. 2. A Workflow of Metadata Creation with DCAP: Relationship between DCAP and Metadata Instances

The following issues have to be solved in order to cope with the embedded metadata,

(1) identifying the location of the embedded metadata,
(2) identifying metadata units in the embedded metadata and identifying attribute-value pairs,
(3) identifying handling rules for embedded metadata – proper value extraction, and
(4) identifying semantic features of attributes and the attribute value classes to process the metadata appropriately.

3.2. Bridge the Model and the Practice to Improve the Usability of Embedded Metadata

In the discussion in the previous section, we identified the following cases of embedded metadata.

Case 1: Embedded metadata using a standard format that is well formed,
Case 2: Embedded metadata using a standard format that is not well formed,
Case 3: Embedded metadata using a metadata schema designed on an application by application basis that is well formed, and
Case 4: Embedded metadata not using a metadata schema designed for a particular application.

Where metadata is "well formed" it is created using well-defined vocabularies and following structural constraints specified in its metadata schema. In Cases 1 and 2, metadata schema need not be given designed on an application-by-application basis.

In Case 1, it is easy to extract the metadata from an HTML document. In Case 2, human intervention is required to find the appropriate metadata vocabularies to correctly extract metadata. In Case 3, human intervention to link the metadata to its metadata schema is required. In Case 4, human interpretation of the metadata is required to find or create metadata schema for the metadata, which is a challenging intellectual process.

In this study, we propose to help metadata extraction from HTML documents for Case 3. It is possible for a human to use the HTML documents to create a minimum metadata schema that may be incomplete but usable for metadata extraction. Thus, in this study, we assume that a metadata schema is always given to the HTML documents. The next step is to define systematic rules to use the schema for metadata extraction. In this study, we propose to use XPath

https://doi.org/10.23106/dcmi.952135962

Expressions to find and extract metadata from HTML documents. We propose to include the XPath Expressions in the metadata schema defined as a 'description set profile'.

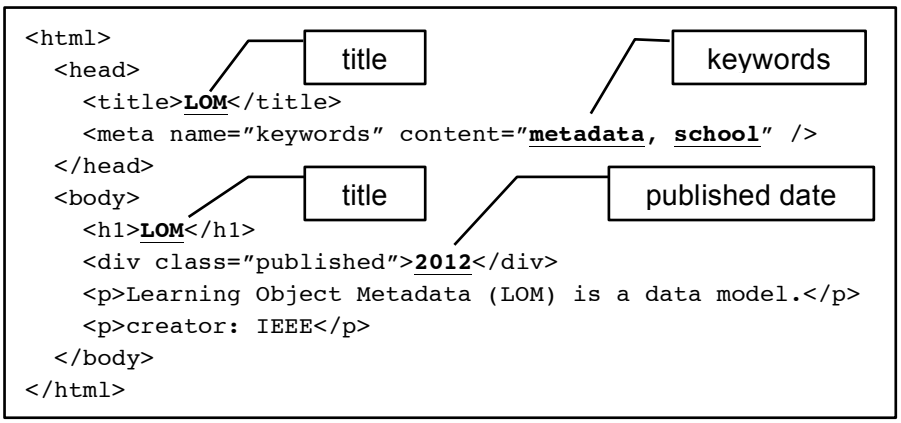

FIG. 3. An example for embedding metadata instances in a HTML document

4. A Model for Linking Embedded Metadata to a Metadata Schema

In the preceding section, we discussed the gap between an ideal model for embedded metadata creation and the practices of embedded metadata creation. In this section, we propose to add an XPath Expression to DCMI Description Set Profiles (DSPs). The XPath Expression specifies a template to find embedded metadata in a document. In other words, the XPath Expression that is an extension of DCMI DSPs, works as a link which connects a DSP to embedded metadata. The sections below show the key tasks for extending a DSP—identification of embedded metadata, creation of an XPath Expression for the metadata, and the inclusion of the XPath Expression in a DSP.

4.1. Identification and Parsing of Embedded Metadata

In this study, we use DSP to define a metadata schema as mentioned in section 3.3. In general, a metadata is a set of attribute-value pairs. However, in some cases, a metadata instance embedded in an HTML document does not contain any attributes but a value(s) where the attribute is determined by context. In any cases of metadata description styles, we need to create an XPath Expression to appropriately point to embedded metadata and include the expression in a DSP. We need first to identify the embedded metadata and parse the metadata text in preparation for the creation of an XPath Expression. The identification/parsing task can be classified into the following cases according to the type of embedded metadata.

- Extract the character strings enclosed by HTML tags such as '*title*', '*h1*', etc. that are specified as tags to express a metadata attribute. The extracted text is the value of the metadata attribute.
- Extract the character strings associated with '*name*' and '*content*' attributes in a '*meta*' tag. The values of these attributes comprise an attribute-value metadata pair.
- HTML elements of '*div*' tag, '*span*' tag, etc. are often used to express metadata instances using standard metadata formats. The attribute-value pairs of these elements should be extracted as metadata instances embedded in an HTML document.
- As a table is often used to express a list of metadata, it is useful to extract table attributes and their corresponding values as a list of attribute-value pairs of metadata.

An XPath Expression is created to identify the extracted attribute-value pair. The XPath Expression created in this task should be used in the next tasks to form an extended DSP.

https://doi.org/10.23106/dcmi.952135962

4.2. Include XPath Expression in DSP

Our model uses DCMI Description Set Profiles (DSPs) to describe definitions as metadata schema. A DSP which defines the structure and restriction of metadata descriptions, is part of a DCMI Application Profile.

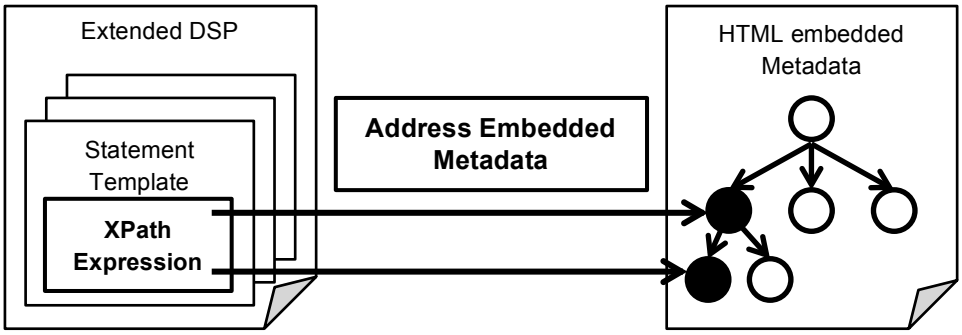

FIG. 4. The Model for Linking DSP to Embedded Metadata

In the Guidelines for DCMI Application Profiles (Coyle & Baker, 2009), it is explained *"A DCAP defines metadata records which meet specific application needs while providing semantic interoperability with other applications on the basis of globally defined vocabularies and models"*. In DCMI Application Profiles, the DSP defines the structure and constraints of the metadata in a machine-readable format. In this research, we use a Statement Template, which defines constraints about a property, as an object linking embedded metadata, By using a Statement Template, computers are able to interpret properties and values clearly. In addition, the Statement Template can specify the domain and range of the property.

Figure 4 shows an overview of how to link embedded metadata to a DSP using XPath. Each Statement Template and XPath point to the value of the property. They are used to enable the interpretation of the metadata.

Figure 5 shows an example of an extended DSP including an XPath for news articles. The DescriptionTemplate, which defines the constraints of an article, has a StatementTemplate. A StatementTemplate defines constraints of the property "*title*". In our model, a StatementTemplate has an XPath using <dxl:location> and <dxl:value>. In this example, values of "*title*" are described by the place pointed to by an XPath Expression like "*/html/head/title*" and "*//h1*" in HTML documents.

5. Implementation

We have developed an experimental system that extracts metadata from HTML documents and accumulates the metadata in a database in the RDF format. Figure 6 shows the process of extracting and accumulating the embedded metadata. That process starts when a user browses a web page. A Web proxy hands over a URI and the HTML document to a Metadata Integrator. The Metadata Integrator cleans the HTML document by Tidy that is a tool for generating a valid HTML/XHTML document. The Metadata Integrator passes a URI and HTML document to the Metadata Extractor which extracts metadata from the HTML document using extended DSP or parsers for standard formats (e.g. RDFa, Microdata). If an HTML document has embedded metadata in a standard format, the Metadata Extractor uses parser libraries for the format. If an HTML document has embedded metadata in a format designed for a particular service, the Metadata Extractor extracts the metadata using the XPath Expression in the extended DSP defined for the service. and the Extractor interprets the extracted metadata with the constraints of the properties and returns the extracted metadata instance to the Metadata Integrator. The Metadata Integrator converts the metadata instance into an RDF graph and accumulates it in the

https://doi.org/10.23106/dcmi.952135962

RDF repository. Figure 7 shows an example of extracted metadata from the HTML document described in figure 3. In this way, we provide a huge RDF graph constructed of embedded metadata in the HTML documents that a user browsed.

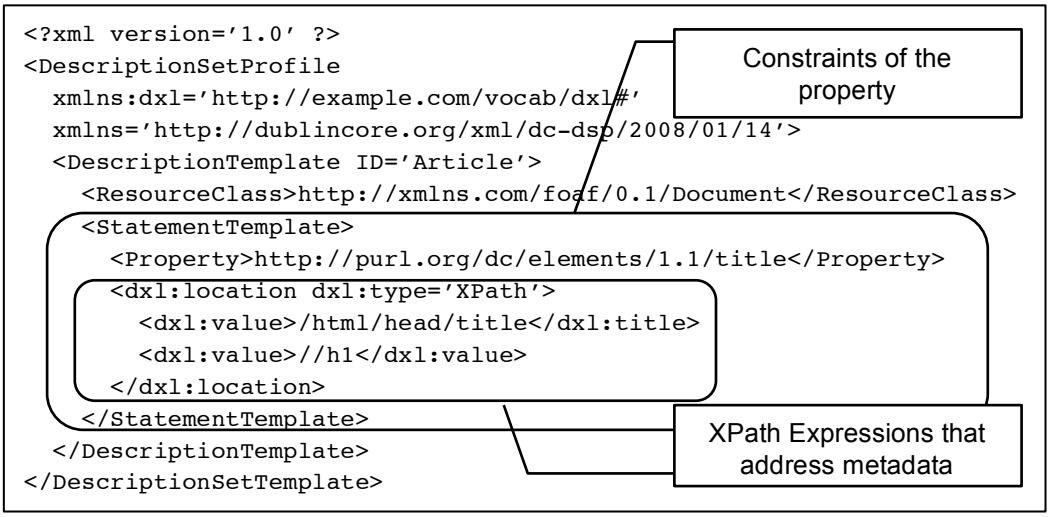

FIG. 5. An example of Extended DSP for describing metadata about "Article"

6. Discussions and Conclusion

In this paper, we propose a model to support the interpretation of embedded metadata without formal schema by linking a metadata instance to DCMI Description Set Profiles. Metadata is currently embedded in HTML documents based on a workflow that we described in section 3. In our model, that workflow will need to include the following steps.

1. Create web content,
2. Embed values of metadata descriptions in some formats for each purpose,
3. Link embedded metadata with XPath,
4. Make metadata schema about embeded metadata explicit using DSP, and
5. Extend DSP to include XPath into Statement Template.

These steps are suitable for existing HTML documents for creating metadata. Therefore, we can use embedded metadata in the huge amount of HTML documents on the Web. This research has the notable feature of having taken notice of the relation between encoded metadata instances and metadata schema instead of the relationship between a metadata instance and a metadata schema. The metadata embedded in HTML documents are encoded. This research enables the use of embedded metadata for the interpretation of such encoded metadata.

There are two major problems to be solved in our model. One is the problem of the cost of defining DSP with XPath Expressions, and the other is the problem of the URIs of resources. We explain the details and solutions of each problem.

https://doi.org/10.23106/dcmi.952135962

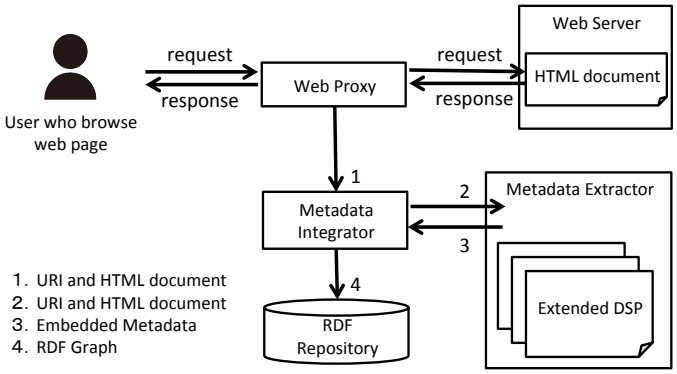

FIG. 6. An Overview of the process for extracting and accumulating metadata using extended DSP

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix dc: <http://purl.org/dc/elements/1.1/>
@prefix foaf: <http://xmlns.com/foaf/0.1/>

<>
  dc:title "LOM" ;
  dc:date "2012" ;
  dc:subject "metadata" ;
  dc:subject "school" ;
  dc:creator [
    foaf:name "IEEE"
  ] .
```

FIG. 7. An Example of Extracted Metadata using our Implementation

The first problem is the high cost of defining DSP with XPath Expression for the extraction and interpretation of encoded metadata instances. DSP are composed of a Description Template, which defines the constraints of the description resources, and a Statement Template, which defines the constraints of the property. Metadata creators choose or create metadata vocabularies for defining their Statement Template, and construct a Description Template of Statement Templates. There are so many metadata vocabularies that it is difficult for metadata creators to select which vocabulary to use. The time and effort for making the lexical selection and the time and effort of DSP description can be lessened by the reuse of an existing DSP. In the metadata schema registry called Meta-Bridge (Metadata Information Infrastructure Initiative), a function diverts the registered DSP and creates a new DSP. We can reduce the effort by just adding the necessary changes to the existing DSP. For example, web pages published by newspaper companies have some metadata. The newspaper companies embed similar metadata descriptions in their HTML documents. So common articles are defined by a DSP, and then the newspaper companies extend the common articles' DSP for their own purposes. Searching of the existing DSPs will be similar to the creation of DSP that can be used in common, and the purpose of a particular community will serve as a subject.

The second problem is being unable to give a suitable URI to the resource described in the encoded metadata. It is important that resources have a URI for sharing metadata but there is also a lot of metadata without URIs. In many cases, metadata is provided for character strings in HTML documents but the resource that is described is not clearly identified. Figure 8, for example, shows a few lines described with Microformats. That is embedded metadata about a person and an event without a URI for the resource. If other communities reuse this metadata on the Web, they cannot refer to the resources. Computers cannot understand what is "Takuma Sato"

https://doi.org/10.23106/dcmi.952135962

is or what "IndyCar race" is without a URI. Since a resource without a URI cannot be referred to from other resources on the Web, it is difficult to share and reuse the metadata.

There is an approach that proposes finding a candidate resource from authorities, such as the name of a person, making it possible to give URIs to the resources. In this approach, metadata creators have to specify which authorities are used for character strings to be treated as resources. This can be specified in the Statement Template. DSPs cannot give restrictions of vocabularies to the subject for metadata description. So we have to provide a system to convert character strings into resources with URI for the subjects of metadata descriptions.

There are lots of metadata embedded in HTML documents but they are often poorly encoded and hard to use for third party users who do not know their metadata schemas. The system shown in this paper helps those third party users automate the metadata extraction process. More importantly, the RDF instances accumulated in the repository form a useful resource to link the HTML documents and data on the Web.

```
<div class="hnews">
   <div   class="vcard"><span   class="fn">Takuma   Sato</span></div>
   topped a wet opening practice session for <div class="vevent">this
   weekend's IndyCar race on <span class="location">Detroit's Belle
   Isle street circuit</span></div>.
</div>
```

FIG. 8. Example for Microformats describing metadata about a person

Acknowledgements

The authors are grateful to Liddy Nevile and Jan Askhoj for their help and contributions. This work was supported by KAKENHI (#23500295), in part by a Grant-in-Aid for Scientific Research (C).

References

Berners-Lee , Tim. (2009). Linked Data - Design Issues. Retrieved, March 26, 2012, from http://www.w3.org/DesignIssues/LinkedData.html

Coyle, Karen & Thomas Baker. (2009). Guidelines for Dublin Core Application Profiles. Retrieved, March 26, 2012, from http://dublincore.org/documents/profile-guidelines/.

Dublin Core Metadata Registry. Retrieved, March 26, 2012, from http://dcmi.kc.tsukuba.ac.jp/dcregistry/

Metadata Information Infrastructure Initiative. Meta Bridge. Retrieved, March 26, 2012, from http://www.metabridge.jp/.

Mika, Peter. (2010). The role of Linked Data in Search and Online Media. FIA Ghent, 2010. Retrieved from http;//fi-ghent.fi-week.en/files/2010/12/1430-Mika.pdf

Nagamori, Mitsuharu, Masahide Kanzaki, Naohisa Torigoshi and Shigeo Sugimoto. (2011). Meta-Bridge:A Development of Metadata Information Infrastructure in Japan. Proceedings of International Conference of Dublin Core and Metadata Applications, 2011, 63-68.

Nilsson, Mikael. (2008). Description Set Profiles: A constraint language for Dublin Core Application Profiles.Retrieved, March 26, 2012, from http://dublincore.org/documents/2008/03/31/dc-dsp/.

Open metadata registry. Retrieved, March 26, 2012, from http://metadataregistry.org/

https://doi.org/10.23106/dcmi.952135962