Legal Information Systems and the Semantic Web

Staffan Malmgren

1 Introduction

Legal information systems offers unique possibilities and challenges compared to other classes of knowledge management and information retrieval systems. One such difference is the sliding scale between authoritative and non-authoritative information.[1] Another is that the non-authoritative information often is used to structure and augment the authoritative information.[2] This means that even if authoritative content is unique (there is only one Code of contracts, not several from which you can pick and choose), non-authoritative content is non-unique and alternate (you can pick and choose from a number of *commentaries* on the code of contracts). A similar division exists between public legal information and private legal information.[3]

1 Seipel, Peter, "En elektronisk rättskällelära", 2005, http://www.rattsinfo.se/2005/051122g.pdf .

2 Schweighofer, Erich, "Computing Law: From Legal Information Systems to Dynamic Legal Electronic commentaries", in "Festskrift till Peter Seipel", 2006.

3 Magnusson Sjöberg, Cecilia, "Critical Factors in Legal Management", p 22 (1998).

Information modelling is the process of structuring information within a particular domain so that concepts, properties and relationships can be expressed in an unambiguous and machine-readable way.

To model a legal information system, it's useful to think in terms of *resources*, where a resource can be both big (such as the body of all Supreme Court decisions) and small (such as a specific sentence in a section of law). There should be a declarative and self-describing model for *metadata* about these resources. *Links* between, and *annotations* of, the resources must be possible within the same metadata model. And the metadata should have *provenance* to make it possible to find out when and by whom a certain linking and annotation statement was made.

A system which provides this is the technology stack known as Semantic Web.

1.1 The Semantic Web Stack

The semantic web (or *semweb*) technology is often referenced to as "the semantic web stack", since it is a *stack* of technologies, where the fundamental ones serve as the building blocks for the more advanced ones.

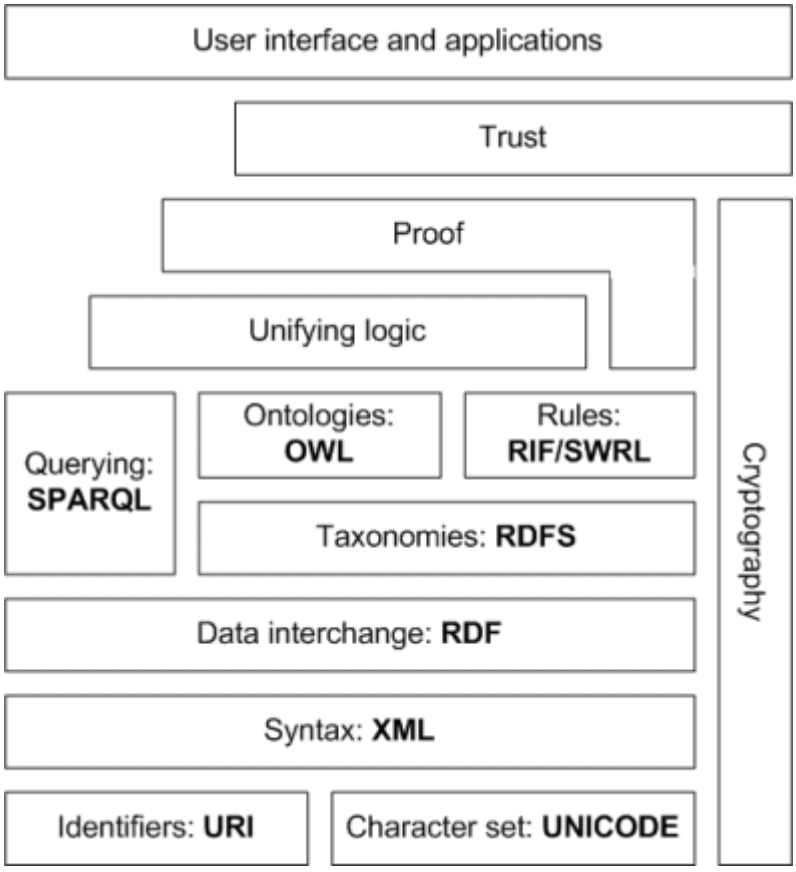

Figure 1: The semantic web stack

At the bottom of this stack are a number of technologies that are well established in traditional web technology. All of these will not be described in this article, but the ones with particular importance for building legal information systems will be. At the bottom, we have URIs (*Uniform Resource Identifier*). As the abbreviation implies, an URI is a standardized way of *identifying* a *resource* in a unambiguous and *uniform* way in the form of a short sequence of characters.

A bit further up is RDF (Resource description framework), a model for describing, or expressing statements about, these resources. Above that, there are standards for describing the terms in our descriptions in such a way that we can automate inference and reasoning about the data (RDFS and OWL), and for searching the data (SPARQL). These building blocks will be described in more detail in this article.

1.2 What the Semantic Web Is Not

1.2.1 It's Not About Search

It is sometimes suggested that information systems, when built on the semantic web, will be able to give better search results. While this can be true, semantic web is not about information retrieval (IR) in any fundamental sense, but rather about describing information and modelling knowledge. When human knowledge about things is specified in such a way that it is machine readable it opens possibilities for - amongst other things - search. Most of this article will be about the modelling of legal information, but at the end some possible information retrieval techniques will be suggested.

1.2.2 It's Not About Content

The semweb stack is a framework for making *statements about* resources, not conveying the *contents of* those resources. Questions about how to accurately mark up and modelling the content of legal information documents are best answered with other technologies. Put shortly, the semantic web is about metadata, not data.[4]

4 This is a very general assertion that ignores the fact that information, which in one context is considered data, can in another context be considered metadata. See also the W3C working draft "Representing Content in RDF 1.0", http://www.w3.org/TR/Content-in-RDF10/ .

That said, semantic web can and should be used alongside technologies for marking up and modelling content, particularly web based technologies such as HTML and XML. The notion of a "resource" builds on earlier work on how to address content on the web. In this article, we will not address the issues of document modelling particularly, but we will show how the RDFa syntax enables us to mix document data, e.g. expressed in XHTML, with RDF metadata statements about that same data.

A document may be decomposed into at least three parts: layout/style, structure and contents.[5] The semantic web can help with modelling the structure. If used pervasively, it may remove the need to create custom document formats for the legal domain, e.g. with XML schema, so that more standard document formats with better tool support can be used (e.g. XHTML[6] or OpenDocument[7]). Instead of creating a custom document format, one can create a custom RDF vocabulary (see below).

1.3 Building Blocks: Resources, Predicates and Vocabularies

The first step in modelling information is to identify resources, that is, the things that we have knowledge about. These resources are uniquely identified by URIs. As an example, the URI for the RDF semantics specification document is `<http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>`.[8] This URI happens also to be a URL (*Uniform Resource Locator*) that identifies a particular document and also contains enough information to actually let us retrieve the document.

This is possible because the URL tells us which communication protocol to use (`http`), which host computer to contact (`www.w3c.org`) using that protocol, and which resource to request from that host computer (`/TR/2004/REC-rdf-mt-20040210/`).

Another example, the URI for the 2005 Nordic yearbook of legal informatics is `<urn:isbn:9788245005547>`. This URI is not a URL, but rather a URN (*Unique Resource Name*). The string of characters does not tell us how to retrieve the book (since it is not published electronically and available over the

5 Magnusson Sjöberg, ibid., p 33.

6 "XHTML™ 1.1 - Module-based XHTML", http://www.w3.org/TR/xhtml11/ .

7 "OASIS Open Document Format for Office Applications (OpenDocument) TC", http://www.oasis-open.org/committees/office .

8 In this text, all URIs are surrounded by angle brackets <like this>, to be consistent with the Turtle syntax for RDF. In a similar manner, all *string literals* are surrounded by double quotes "like this".

web, this would be difficult), but it tells us enough that we can use it as a unique and machine readable identifier for the book. It tells us that this is, in fact, a URN, that it belongs to a particular *name space* (`isbn`), and some information that is specific to that name space (`9788245005547`). Using this information, a *URN resolver*[9] could tell you more about this resource, such as where to buy it online or where the nearest library carrying it is located.

An important aspect of a URI is that you must ensure that it is indeed unique. At the same time, there is no global registry of all possible URIs, so in theory two or more parties can claim the same URI for two totally different uses.

If you assume that `<urn:isbn:9788245005547>` is the URI for the Nordic yearbook, and your URN resolver assumes that it is the URI for the latest Dan Brown book, you will not be happy with the information that it returns. How to achieve this uniqueness differs between different types of URIs. For URLs, domain names are guaranteed to be unique (at least at a particular point in time) by virtue of the domain name system. Only one party can be in control of the domain name `www.w3c.org` at any given time. That party can, presumably, be relied on to not create any conflicting URIs in it's resource space (W3C will not create two different resources, then name them both `/TR/2004/REC-rdf-mt-20040210/`). For ISBN URNs, the ISBN system makes sure that no numbers collide.

Another important aspect of a URI is that it identifies *resources*, not just documents. A URI can refer to a geographical location,[10] a living person,[11] or even to an abstract concept, e.g. the concept of "natural rights".

A third important aspect of a URI is that, as an identifier, it is *opaque*. You may not assume anything about the URI just by looking at it. For example, if you have two URIs, `<http://example.org/doc>` and `<http://example.org/doc#section>`, you may not assume that the latter is part of, or in any way connected, with the former.

Once we have URIs to identify our resources, we can start describing things. As a first example, we might want to model our knowledge that Dag Wiese Schartum was the editor for the yearbook:

---

9 I.e. a piece of software that can perform the necessary database queries, lookups etc to transform the URN to a URL (if the resource can be retrieved over the Internet) or otherwise more information about the resource.

10 See, e.g., "A Uniform Resource Identifier for Geographic Locations ('geo' URI)" (draft), http://tools.ietf.org/html/draft-ietf-geopriv-geo-uri-04 .

11 A requirement for such a URI is that it *cannot* be a URL, since any resource retrievable over a network is, by definition, a document, and people are not documents (nor, as of December 2009, retrievable over a network). See W3C, "Cool URIs for the semantic web", http://www.w3.org/TR/cooluris/ .

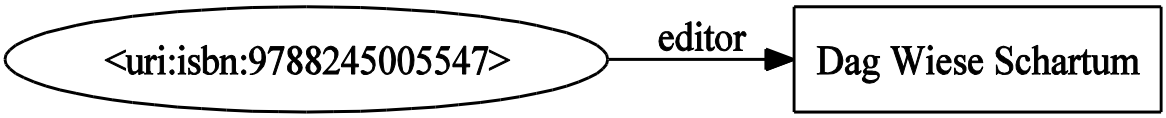

Figure 2: A simple statement

This is a *triple* – a collection of three atomic components called the subject, the predicate, and the object. The subject is the yearbook resource (i.e. the thing we have some information about), the predicate is the type of information we have ("editor"), and the object is the actual information (in this case, "Dag Wiese Schartum"). In a similar way, we can describe other properties of this resource using other predicates (what year it was published in, which publisher, how many pages, its library classification in a library, etc).

Another thing is that we specified that Dag Wiese Schartum was the "editor" for the book, but we did not specify what we meant by "editor". For a publishing house and a computer programmer, "editor" has very different meanings (for a programmer it is the name of a software program that enables the editing of text files). We would like to *unambiguously* specify the type of information that we are describing.

Similarly, there might be more than one person named "Dag Wiese Schartum". As the information stands, we have no idea *which person* was the editor, only that the editor was *someone named* Dag Wiese Schartum. This name can also be written in different ways. If we have some similar information about another book, say:

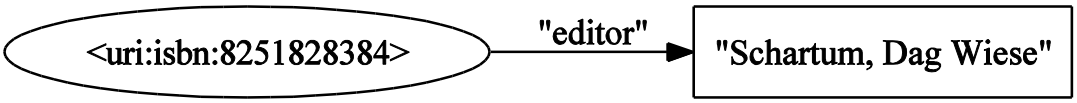

Figure 3: A statement with a different, yet similar subject

Then we won't know that both these books were edited by the same person.

These are problems of non-uniqueness and ambiguity. We can solve these as we solved the problem on how to uniquely and unambiguously identify the book itself, that is, by using URIs. A modified example is

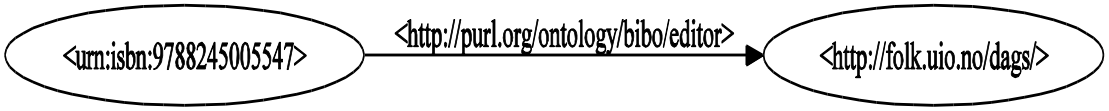

Figure 4: An unambiguous example

It is not as readable at first, but it tells us much more. `<http://purl.org/ontology/bibo/editor>` is a *term* from the *vocabulary* that the Bibliographic Ontology Specification specifies.[12] A vocabulary is a set

12 http://bibliontology.com/.

of terms we can use as predicates when making statements. In this case, we use the vocabulary that the Bibliographic Ontology working group has specified for bibliographic information. This vocabulary has about 140 terms which are used to describe the bibliographic information about books and other documents.

Instead of the string "Dag Wiese Schartum" we now use a web URI `<http://folk.uio.no/dags/>`. This URI refers to an existing web page, but that is not a requirement.[13] As a URI, this is a resource which we can then use as a subject in another triple:

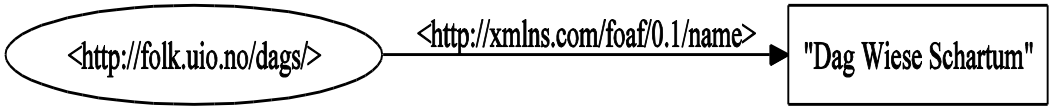

Figure 5: The object from the previous statement used as the subject of another statement

Putting it all together, we now can start to model some more information:

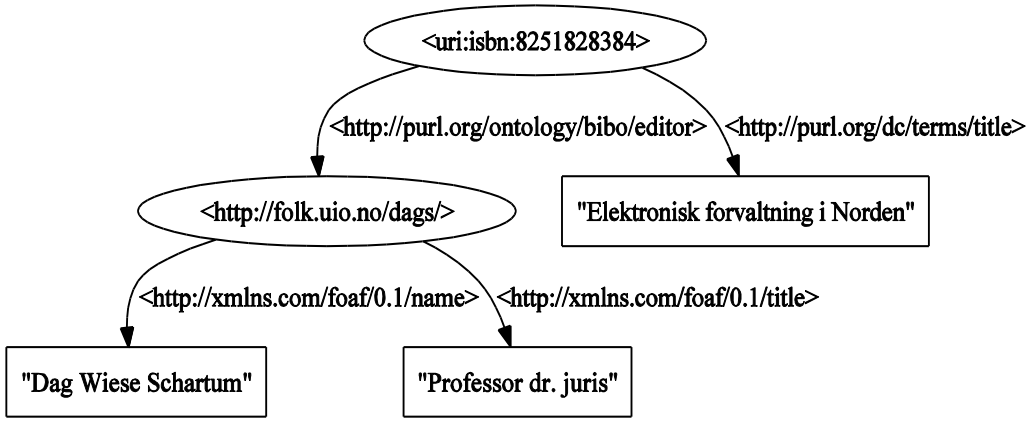

Figure 6: A complete graph

The same information can be expressed in text form. There are many different formats for expressing RDF statements in a textual form, but it is important not to confuse the syntax itself with the RDF model. Some syntaxes are optimized for easier parsing and serialization (e.g the N-triples format[14]), others for integration with XML-based tools (e.g. the RDF/XML format[15]), but they can all represent the same data. In this article, a format called Turtle (Terse RDF Triple

13 In fact, it is preferable that it doesn't, as a resource can't be *both* a web page and a living person, as mentioned earlier. In this article we blur this distinction, in order to keep things simple.

14 Part of the W3C Recommendation "RDF Test Cases", http://www.w3.org/TR/rdf-testcases/#ntriples.

15 "RDF/XML Syntax Specification (Revised)", http://www.w3.org/TR/REC-rdf-syntax/.

Language)[16] is used, as it is the easiest syntax for humans to read. The above graph contains exactly the same information as this text serialization below:

```
@prefix bibo: <http://purl.org/ontology/bibo/>
@prefix foaf: <http://xmlns.com/foaf/0.1/>
@prefix dct: <http://purl.org/dc/terms/>

<urn:isbn:9788245005547> bibo:editor
<http://folk.uio.no/dags/> ;
    dct:title "Elektronisk forvaltning i Norden" .

<http://folk.uio.no/dags/> foaf:name "Dag Wiese Schartum" ;
    foaf:title "Professor" .
```

As you can see, the Turtle data defines a set of prefixes (`bibo, foaf and dct`) and maps these to different URIs. Once a prefix has been defined, it can be used as a shorthand for the same URI, so that instead of using a long identifier like `<http://purl.org/ontology/bibo/editor>`, one can use the shorter and more readable `bibo:editor` instead.

1.4 Simple Reasoning with Predicates

The book we have been using as an example is an anthology, i.e. it consists of separate articles, with each article having a separate author and title. We can model one such article and connect it to the book using the following statements:

```
<http://www.juridicum.su.se/iri/cems> foaf:name "Cecilia Mag-
nusson Sjöberg"
<urn:uuid:ebcc5b50-e044-11de-8a39-0800200c9a66> dct:author
<http://www.juridicum.su.se/iri/cems> ;
    dct:title "Rätt rättsinformation i e-förvaltningen" ;
    dct:isPartOf <urn:isbn:9788245005547> .
```

Or, displayed in graph form:

16 See “Turtle – Terse RDF Triple Language”, http://www.w3.org/TeamSubmission/turtle/

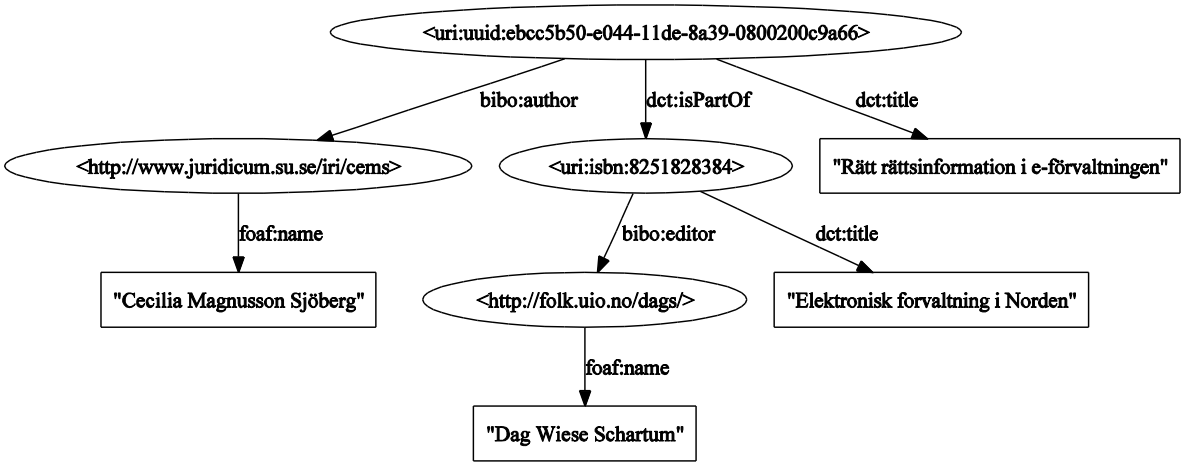

Figure 7: A complex graph

Note that since the article itself does not have an ISBN number, there is no immediately obvious way of creating a URI for it. In the example, we created a UUID URN so that we would have an identifier for the resource. We could have used a number of other URI design strategies, or avoided using a URI at all (see part 2.1 for more on this).

The interesting part is the `dct:isPartOf` predicate statement in the last statement. The semantics for this predicate is intuitive; it tells us that one resource is included (physically or logically) in another resource. Using this, we can create an information retrieval system where we look for information about one resource (the article "Rätt rättsinformation i e-förvaltningen", such as where it is located on the shelves) and receive search results about another resource (the book "Elektronisk forvaltning i norden").

There is nothing special about such an IR *system* per se – any decent bibliographic management system will be able to handle resources that are part of other resources. The interesting thing is that we specified two independent resources, and their connectedness, using only *system-independent* information using a open standard.

2 A Semweb Architecture for Legal Information

2.1 Modelling the Resources

In a legal information system, a resource can be a law, a particular section of a law, a court case, a dissenting opinion or just about anything else that we might

want to describe or refer to. This means that we must have some idea of how to create URIs for these resources.[17]

As an example, the case NJA 2005 s 361 refers to article 13 of the Personal Data Act (1998:204). When stating this in RDF, the legal case will be the subject, and article 13 of the Act will be the object. These are connected by a predicate, the semantics for which should be that the verdict in the case refers to upon that particular article .

The subject might be referred to by the URI `<http://rinfo.lagrummet.se/publ/rattsfall/nja/2005s361>` and the object by the URI `<http://rinfo.lagrummet.se/publ/sfs/1998:204#P13>`. The outside *user* of this data may not draw any conclusion from the way the URIs appear to be structured, as URIs are opaque. However, for the *designer* of the data, it makes sense to use already-existing identifiers as part of the URI.

The first URI (`<http://rinfo.lagrummet.se/publ/rattsfall/nja/2005s361>`) contains:

- a fixed base URI consisting of the domain name that the URI designer controls (`rinfo.lagrummet.se`)
- a segment indicating that the URI identifies a published content resource (`/publ`)
- a segment indicating that the resource is a legal case (`/rattsfall`)
- a segment indicating that the case is published in the reporter of the Swedish supreme court, Nytt Juridiskt Arkiv (`/nja`)
- a final segment indicating that this case can be found on page 361 in the 2005 edition (`/2005s361`).

There are a number of different schemes of URI design suggested for legal resources. The one used in the above examples (and in the rest of this article) come from the Swedish Legal information project.[18] One of particular interest is the newly proposed URN:LEX scheme which strives to be a global, jurisdiction-neutral way of citing any legal document, including parts thereof.[19]

---

17 It is not critical to have a URI for a resource in order to make statements about it. In a collection of RDF statements there can be "blank nodes" or BNodes, resources which are the subjects of some statements and objects of others, without having a URI.

18 http://rinfoprojektet.wordpress.com/projektdokumentation/

19 Spinosa, Francesconi & Lupo, "A Uniform Resource Name (URN) Namespace for Sources of Law (LEX)" (draft) http://tools.ietf.org/html/draft-spinosa-urn-lex-00

2.2 Modelling the Predicates

Having modelled our information resources, the next step is to say something about them. The middle part of a triple statement is the *predicate*, and we use this to specify what kind of information we are stating, e.g. "this thing (the object) has a *title*, which is […]". But using regular everyday words for our predicates is insufficient. For example, what do we mean when we say that something has a certain title?

- If we're talking about a book, then the title is the name of the book, as assigned by the author or editor.
- If we're talking about a person, it could be their job title, i.e. a designation of their job description, or it could be any sort of prefix or suffix added to the persons name to signify an official position (e.g. "President") or academic qualification (e.g. "Professor").
- If we're talking about property such as a plot of land, title signifies a certain bundle of rights to that property including possession, use and enclosure.

In other contexts, "title" can mean yet other things. In order to be useful, the predicates that we use must be unambiguous, and must have defined *semantics*, i.e. we must know what we mean by using them.

The ambiguity problem is solved in the same way that we solved the ambiguity for resources – we use URIs for our predicates. This ensures that there will not be confusion between information using "title" in the book sense and information using "title" in the person sense.

The semantics problem is harder, and a lot of effort goes into crafting sets of terms that can be used as predicates for describing information of different kinds. We call such a set of defined terms a vocabulary. Commonly used vocabularies include DCMI Metadata Terms (sometimes called Dublin Core, after the standards organization that defined them),[20] which is used for information about documents, and FOAF,[21] which is used for information about people.

The Dublin Core definition of "title" is "A name given to the resource". The FOAF definition of "title" is "Title (Mr, Mrs, Ms, Dr. etc)". While these descriptions are terse, they enable the user to better understand the meaning of the term.

20 http://dublincore.org/documents/dcmi-terms/ .

21 http://www.foaf-project.org/ .

2.3 Modelling the Subjects

The third part of a RDF statement is the object, i.e. the actual data. There are two kinds of possible objects: resources and literals. We are already familiar with resources and know that they are in the form of URIs. A resource can be the object of one statement and the subject of another – the resource `<http://folk.uio.no/dags/>` used in the first part is an example of this.

Literals, on the other hand, are not URIs. The simplest form is the untyped string literal, e.g. a short segment of text. In our earlier example, the book title "Elektronisk forvaltning i Norden" was an example of a string literal.

Literals can be typed, i.e. be given explicit information about what kind of data they contain. "2009-12-10" can be a document reference number, or it can be a calendar date – by specifying that it is a date we remove any possible ambiguity about this (note that the predicate itself may also remove this ambiguity by virtue of being defined e.g. only for dates).[22]

Literals can also be language tagged, which is useful when dealing with legal information available in more than one language. For example, the EC Directive 2003/98/EC (the "PSI Directive") have different titles for each official EC language. While the Directive itself might be thought of as a single resource (it is equally binding regardless of language version, and references to it will usually not be language-dependent), we should be able to specify it's different titles in different languages:

```
<urn:lex:eu:council:directive:2003-11-17;98> dct:title
    "re-use of public sector information"@en ,
    "vidareutnyttjande av information från den offentliga sek-
    torn"@sv .
```

Or, in graph form:

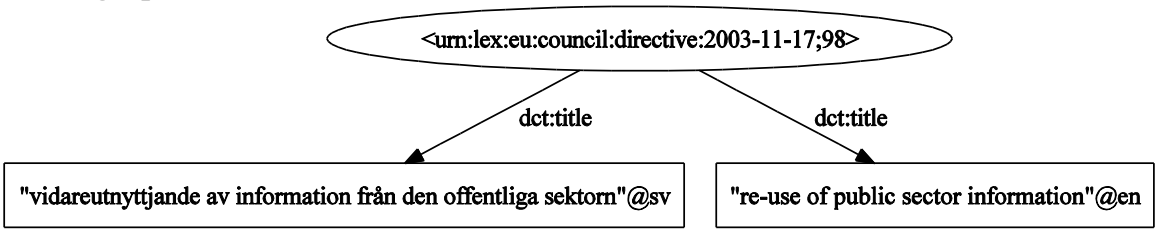

Figure 8: Multiple language labels

22 Datatypes, like predicates, are uniquely defined by URIs. In particular, the datatypes defined by the XML Schema specification are often used.

2.4 Formalizing the models

A vocabulary can be defined more or less stringently. In its simplest form, it can be just a plain text list of terms. But in essence, a vocabulary is a set of statements about resources, the resources in this case being the predicates themselves. Therefore, we can represent a RDF vocabulary in RDF.

For example, this is the Dublin Core definition of "title", as expressed in RDF:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2008/05/skos#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
dcterms:title rdfs:label "Title"@en-US ;
    dcterms:description "A name given to the resource."@en-US ;
    rdfs:isDefinedBy <http://purl.org/dc/terms/> ;
    dcterms:issued "2008-01-14" ;
    dcterms:modified "2008-01-14" ;
    rdf:type rdf:Property ;
    dcterms:hasVersion
    <http://dublincore.org/usage/terms/history/#titleT-001> ;
    skos:note "In current practice, this term is used primarily
    with literal values; however, there are important uses with
    non-literal values as well.  As of December 2007, the DCMI
    Usage Board is leaving this range unspecified pending an
    investigation of options."@en-US ;
    rdfs:subPropertyOf dc:title .
```

In this example, predicates from two additional vocabularies are used, in effect, to define the Dublin core vocabulary – the RDFS[23] and SKOS[24] vocabularies. These, together with the OWL family of vocabularies,[25] provide a rich set of building blocks for defining vocabularies where (parts of) the semantic definitions are themselves machine readable.

Of particular interest in the above is the `rdf:type` predicate. This is used to state that a resource is an instance of a class.[26] A class may be described as what the resource *is*, as opposed to properties, which is what the resource *has*. For example, the resource `<urn:isbn:9788245005547>` *is* a `bibo:Book` that *has* a

23 "RDF Vocabulary Description Language 1.0: RDF Schema", http://www.w3.org/TR/rdf-schema/ .

24 http://www.w3.org/2004/02/skos/ .

25 http://www.w3.org/TR/owl2-overview/ .

26 "RDF Vocabulary Description Language 1.0: RDF Schema", section 3.3, http://www.w3.org/TR/rdf-schema/#ch_type .

`bibo:editor`. The subject of a `rdf:type` statement should always be a URI for the same reasons that a predicate should.

2.5 Connecting Information From Different Sources

A collected set of statements from a single source is normally referred to as a *graph* – since statements usually connect resources to each other, the statements can be ordered in a graph (and even visualized, which we saw in part 1 of this article). When processing information, e.g. in order to build a complete legal information system, information from multiple graphs can be combined, as long as they contain statements about the same resources.

In any legal information system, there will usually be authoritative information, i.e. stuff that have been promulgated or decided by a public body with the authority to do so. There may also be non-authoritative information such as annotations of a statute, or connections between legal cases and scholarly analyses.

Traditionally, a uniform way of citing statutes and cases has been the way that non-authoritative information connects with authoritative information. References to statutes will be done using the same way that statutes themselves refer, e.g. "49 § personuppgiftslagen (1998:204)" to refer to article 49 of the Personal Data Act,[27] rather than referring to a specific page in a printed volume of the statutes. Legal case analyses on the other hand generally refer to a Swedish Supreme Court verdict using the syntax "NJA 2005 s 361" – that is, a reference to a specific page in a printed volume of the cases, rather than referring to the case number or date for the verdict.

In a legal information system based on semantic web technologies, such references will be done using resource URIs. Traditional citation styles may still be used for display purposes (most users of a legal system will understand a reference to "NJA 2005 s 361" in the user interface, but few will understand a reference like `<http://rinfo.lagrummet.se/publ/rattsfall/nja/2005s361>`.

3. Implementing the System

In order to build a system based on RDF and semantic web you need three things: A source from where you can get RDF triples, a place to put them, and a way to create a browseable user interface from them.

27 That is, the article number, followed by the § symbol, followed by the name of the Act and it's unique number within parenthesis

3.1 Representing the Metadata

As stated earlier, the semantic web is not primarily about data representation, but metadata models. It is, however, convenient to handle the data and the metadata within the same document, as metadata will often be mixed in with the data. A documents' title is metadata, but will normally appear in the document itself as well. For example, if the actual text of the Data Protection Act is stored in this HTML document:

```
<html>
    <head>
        <title>Personal data act (SFS 1998:204)</title>
    </head>
    <body>
        <h1>Personal data act</h1>
        <h2>SFS 1998:204</h2>
        <hr/>
        <h2>General provision</h2>
        <h3>Purpose of this act</h3>
        <p>
            <span class="articlenum">Article 1:</span>
            The purpose of this Act is to protect people
            against the violation of their personal integrity
            by processing of personal data.
        </p>
    </body>
</html>
```

Then we can add markup to the document to indicate existing metadata (and also add more):

```
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:dct="http://purl.org/dc/terms/"
    xmlns:rpubl="http://rinfo.lagrummet.se/ns/2008/11/rinfo/pub
    l#">
    <head>
        <title>Personal data act (SFS 1998:204)</title>
    </head>
    <body about="http://rinfo.lagrummet.se/publ/sfs/1998:204"
    typeof="rpubl:Act">
        <h1 property="dct:title">Personal data act</h1>
        <h2 property="rpubl:actNumber">SFS 1998:204</h2>
        <hr/>
        <h2>General provision</h2>
        <h3>Purpose of this act</h3>
        <p typeof="rpubl:Article"
        about="http://rinfo.lagrummet.se/publ/sfs/1998:204#P1">
```

```
            <span rel="dct:isPartOf"
            href="http://rinfo.lagrummet.se/publ/sfs/1998:204/"
            />
            <span property="rpubl:articleNumber" con-
            tent="1">Article 1:</span>
            The purpose of this Act is to protect people
            against the violation of their personal integrity
            by processing of personal data.
        </p>
    </body>
</html>
```

This is an example of encoding RDF metadata using the RDFa standard. An RDFa parser can process the XHTML document and produce the following six RDF triples:

```
<http://rinfo.lagrummet.se/publ/sfs/1998:204> a rpubl:Act;
    dct:title "Personal data act";
    rpubl:actNumber "SFS 1998:204".

<http://rinfo.lagrummet.se/publ/sfs/1998:204#P1> a
rpubl:Article;
    dct:isPartOf <http://rinfo.lagrummet.se/publ/sfs/1998:204>;
    rpubl:articleNumber "1".
```

Or, as expressed as a graph:

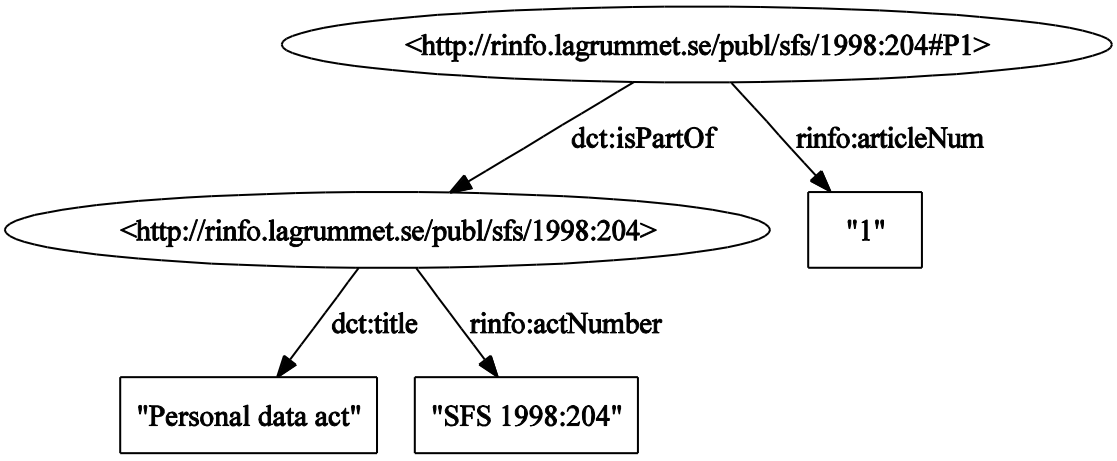

Figure 9: Extracted RDF graph

3.2 Storing and Querying the Metadata

Systems for storing large amounts of RDF statements are called *triple stores*. A triple store is one kind of database system, although that term is nowadays mostly associated with RDBMSs, databases using the relational data paradigm.

RDF and the semantic web uses a different paradigm and so uses different applications for loading and storing the data as well as searching and aggregating it.

Retrieving information programmatically from a database is typically done through a query language, i.e. a special purpose formal language that enables the user to specify a number of constraints. When querying a triple store, it returns all information in the store that satisfies the constraints in the query. The standard query language for RDF data is SPARQL.[28] In the SPARQL model, a query is constructed as a graph pattern, i.e. a set of statements forming a graph, but with variables in place of one or more subjects, predicates, or objects. A very simple query to retrieve the title of the resource with URI `<urn:isbn:9788245005547>` would look like the following:

```
PREFIX dct: <http://purl.org/dc/elements/1.1/>
SELECT ?title
WHERE
{
  <urn:isbn:9788245005547> dct:title ?title .
}
```

In this case, `?title` is a variable that acts as a placeholder in the query. The triple store will check if there is any triple that can fit into this pattern. If we run this query against the data set that we used in the first part of this article, we will receive a *solution sequence*, i.e. a list of possible solutions to the query. In this case, there will only be one solution as the resource `<urn:isbn:9788245005547>` only has a single `dct:title`.

?title
"Elektronisk forvaltning i Norden"

A less specific query would be to see all resources in the store that have `dct:titles`. This query uses two variables:

```
PREFIX dct: <http://purl.org/dc/elements/1.1/>
SELECT ?resource, ?title
WHERE
{
  ?resource dct:title ?title .
}
```

28 http://www.w3.org/TR/rdf-sparql-query/. The name is an abbreviation for "SPARQL Query Language for RDF" – in other words, it's a *recursive acronym*.

Run against the example data from part 1 of this article, this will produce a solution sequence with two rows:

?resource	?title
<urn:isbn:9788245005547>	"Elektronisk forvaltning i Norden"
<urn:uuid:ebcc5b50-e044-11de-8a39-0800200c9a66>	"Rätt rättsinformation i e-förvaltningen"

3.3 An example application: Inbound links

Assuming that we have a repository of documents stored in a structured XML format such as XHTML, and a triple store with RDF data extracted from those documents, it becomes possible to transform the documents so that the text of the document is presented alongside references to the document text. We assume that the document consists of sections that have been given individual resource URIs (such as articles in a statute), and that references are made to these individual resources.

In order to create this view, we need a document, a list of references to resources in that document, and a set of rules for performing the transformation. In order to create the list of references, we use a SPARQL query, and in order to express our transformation rules we use a XSLT stylesheet.

The SPARQL query should find all resources that references any resource in the document. Legal sources refer to each other for a variety of reasons, creating a variety of relations between resources. If our metadata captures this variety by using different predicates for different reasons, we might want to adjust our query to just select one or a few of these predicates. However, when modelling our vocabulary it makes sense to make our custom relationship predicate be a subproperty of a generic relationship property, such as dct:references.[29] Declaring a particular predicate as a subproperty of another (using `rdfs:subPropertyOf` in the formal vocabulary[30]) means that all resources related by the other predicate are also related by the particular predicate. The query will in it's simplest case look something like this

```
PREFIX dct: <http://purl.org/dc/elements/1.1/>
SELECT ?source, ?label, ?target
WHERE
{
```

29 According to the DCMI Metadata Terms, "A related resource that is referenced, cited, or otherwise pointed to by the described resource."

30 http://www.w3.org/TR/rdf-schema/#ch_subclassof .

```
  ?resource dct:isPartOf
<http://rinfo.lagrummet.se/publ/sfs/1998:208> .
  ?source dct:references ?resource .
  ?source dct:title ?label
}
```

This query will return a list like this (resources abbreviated for readability):

?source	?label	?target
<.../sfs/2001:454 #P4>	4 § Lag (2001:454) om behandling av personuppgifter inom social-tjänsten	<.../sfs/1998:204# P2>
<.../sfs/2006:378 #P2>	2 § 2 st Lag (2006:378) om lägenhetsregis-ter	<.../sfs/1998:204# P2>
<.../sfs/1998:527 #P2>	2 § Lag (1998:527) om det statliga personadressre-gistret	<.../sfs/1998:204# P3>

In order to create the final document, we need to combine this information with the starting document. This is done using XSLT,[31] creating a transformation template for handling each addressable section of the source document. As long as the triple store can return the query results using a XML format,[32] the template can then fetch information from the query result using the `document()` XSLT function.[33]

4. Critical Factors in Vocabulary Management

Having described the technology foundation and sketched a simple system for presenting information using a corpus properly marked up with an appropriate

31 "XSL Transformations", http://www.w3.org/TR/xslt

32 Such as SPARQL Query Results XML Format, http://www.w3.org/TR/rdf-sparql-XMLres/ .

33 For more information about this approach, see Kirchberger, Christine & Malmgren, Staffan, "Inbound links – picking the low hanging fruit from the semantic web", presented at Workshop on legislative XML 2008.

vocabulary, it might be time to take a step back and ask what it means to properly mark up something, and what can be considered an appropriate vocabulary.

A model for evaluating information standards for representing legal information was developed in the Corpus Legis project.[34] The model lists eight critical factors to consider when designing and implementing legal document management.

4.1 Establish a Consensus on the Purpose of the System and its Applications Area(s)

The system described can be classified as *docware*, and at its core it is defined by our metadata model, including our vocabulary. Ideally, we wish to make it useable for any handling of legal information. A law librarian, a case manager at a law firm, and a legislator may use three (or more) separate systems supporting wildly different use cases. Our metadata model should be flexible enough to support these use cases, yet simple enough so that it is actually manageable to implement it in the systems.

4.2 Document Analysis

The semantic data model has gained wide support in the general document management community, e.g. amongst librarians and publishers. There exists a rich set of vocabularies describing general properties of documents. Ideally a legal vocabulary should build upon these well understood general vocabularies.

But legal information has a number of particular traits which are not captured by more general vocabularies. For example, a reference from one legal resource to another can have a number of very different meanings:[35]

- Definition: One resource refers to another resource that defines a term used in the first resource.
- Requirements: One resource refers to another resource stipulating the requirements for the first resource to be applicable.

34 Magnusson Sjöberg, Cecilia, "Critical Factors in Legal Document Management", 1998, p 16 .

35 Kirchberger & Malmgren, ibid. Note that the list is not exhaustive, and only covers references between statutory-type resources.

- Additional requirements: One resource contains requirements, and also refers to another resource containing additional requirements, for applicability.
- Multiple requirements: One resource refers to multiple other resources, all containing requirements for applicability (ranging from general to specific requirements)
- Non-application: One resource contains requirements which, when satisfied, makes another resource unapplicable.

Different jurisdictions and legal traditions will result in different requirements for a legal information vocabulary. For example, the strong focus on preparatory works in Swedish law will result in a well-defined way of handling these in a Swedish legal information vocabulary, while a EC Law vocabulary will put more focus on the relation between primary, secondary law and supplementary law.

4.3 Deciding the Markup Levels: Layout, Structure or Content

The definition of a vocabulary is only concerned with metadata, particularly defining the semantics of structural elements. It should be combined with the usage of an appropriate document type, preferably using some XML and some schema mechanism.[36] One advantage of this separation of concerns is that the XML document type does not need to be tailored to legal information at all, enabling the use of document types with strong tool support such as XHTML or OpenDocument.

4.4 Strategy for Schema and DTD Design

The vocabulary will be used by both producers and consumers of legal information. When designing it, it is important to get the input from both these stakeholder groups. While the consumers will mostly be a homogenous group in terms of their requirements for the vocabulary, the producers' requirements will differ substantially depending on the kind of information they provide. A court will have different requirements than a legislator, as their information is different.

36 Languages for expressing document schemata includes DTDs, XML Schema, and RelaxNG.

It is therefore important to develop the vocabulary in an iterative fashion, soliciting feedback from the different stakeholders and refining it based on that feedback. Since vocabulary design is a highly technical process, it is particularly important to try and bridge the gaps between technologists, information specialists and jurists.

4.4 Preparation of Future IR Functions

Any system for storing and managing legal information will have some kind of information retrieval functions, although they need not always be the sophisticated full-text, probabilistic relevance-ranking systems we most often associate with the term.

The consumer stakeholder group will mostly be concerned with IR functions, as they can be important when differentiating different end user-applications. Their input must therefore be solicited in the design phase.

4.6 Compatibility Issues

The problems that RDF and the semantic web solves are not new, and they have been solved before. What is different about the semantic web initiative is that it is fundamentally built on the distributed, hyperlinked foundation of the World Wide Web, and that it is a stack of vendor-independent open standards. There exist multiple implementations of every component in the stack (RDF parsers, triple stores, SPARQL query engines, OWL reasoners…). This strong focus on standards ensures that the software tools will have less compatibility issues.

However, when designing custom vocabularies, there can be compatibility issues. If we wish to use a system designed for the American market to store our Swedish legal information, it will probably not know how to deal with our fine-grained hierarchy of different reference types between statues, court decisions and preparatory works. Proper vocabulary design might minimize these problems – if we e.g. define a predicate for reference between a court case and a page in a preparatory work, used as a support for a teleological interpretation of a statute (e.g. `rinfoex:TeleologicalInterpretationBasis`), we should in our vocabulary state that this predicate is more specific version (`rdfs:subPropertyOf`) the more general `dct:references`.[37]

37 http://dublincore.org/documents/dcmi-terms/#terms-references

4.7 Complexity of the Chosen Tool(s)

RDF and the rest of the semantic web stack is complex. This follows from the fact that the real-world knowledge it attempts to model is complex.

However, the complexity can be distributed among the actors. The vocabulary designers need to have a strong grip of both knowledge representations in general, the semantic web standards in particular, as well as the specific domain of legal information.

The implementors, i.e. the people designing information systems that produce or consume information, need to have an understanding of basic RDF concepts and be able to read the formally specified vocabulary. But they don't need to have deep knowledge of taxonomy and ontology construction.

The end users on the producing side need to understand the vocabulary as explained in plain text, and have access to a content authoring system that makes it easy to enter the metadata without e.g. having to write RDFa markup. But they do not need knowledge of the RDF model or the semweb stack.

The end users on the consumer side need to understand basic aspects about legal information such as how statutes and court cases refer to one another, but the complexities of the structured information need not be exposed.

4.8 Development Trends in the Standard Community

The semantic web vision has been in place for ten years now.[38] Compare to the explosion of the web itself, which was transformed from a project proposal to 360 million users in the same period,[39] the growth of the semantic web may seem slow. However, the interest is steadily increasing and new products supporting semantic web standards appear regularly.

5 Further Developments

The purpose of this article is to give a hands-on overview of what the semantic web technology stack is, and how it can be put to use for structuring legal information. It can only scratch the surface of what is possible. To conclude, a few samples of challenges and possibilities that the semantic web can mean for legal information.

---

38 Berners-Lee, Tim; Fischetti, Mark, “Weaving the Web: The Past, Present and Future of the World Wide Web by its Inventor” (1999).

39 http://www.internetworldstats.com/stats.htm .

5.1 Challenges

5.1.1 Semantic Fidelity and Complexity

When designing semantics for data and metadata, one question is what level of *semantic fidelity* to aim for – should we attempt to capture and describe all possible metadata e.g. by creating a resource URI for each individual letter of the text, having predicates for paper size and paper quality of the original print, and similar aspects? Or should we focus on higher level concepts such as the URI of a document as a whole, and just capture a few key properties of this, such as its title?

If we make our vocabularies and metadata models more complex, we can achieve better semantic fidelity and enable more use cases for our information. But we run the risk of creating an unmanageable system where the actual metadata quality is lower.

Correspondingly, if we make the vocabulary simple, we achieve low semantic fidelity when we cannot capture potentially useful metadata, which limits the use cases we can have for the information. But a simpler system has a better potential to be used correctly with high metadata quality.

5.1.2 Automated Legal Reasoning

Whenever terms like "inferencing" and "reasoners" are mentioned in the context of legal information, it raises the question of whether it is possible to express legal norms in such a way that a computer can apply them. The kind of legal information that is produced today is not adequate for this kind of automated legal reasoning, even when using very advanced semantic web concepts (such as modelling semantics using the full OWL ontology).

For a computer to be able to do reasoning, the information needs to be encoded in an unambiguous way, while legal information such as statutes and legal cases have ambiguity built in. Creating norms for automated reasoning systems is a very different activity compared to creating norms for humans to follow (and specialists to interpret). While such a system might be more predictable by virtue of being deterministic, it would also be less flexible and above all non-transparent to all but those versed in both legal theory and computer science.

Furthermore, systems for automated reasoning, particularly those used in the semantic web, are mostly concerned with first-order predicate logic[40] (by way of

40 Russell & Norvig, "Artificial Intelligence – A modern approach", p 186 (1995).

description logics),[41] meaning logical systems expressing propositions about what *is*, quantified for a particular domain. Legal norms are more closely related with deontic logic, which is concerned with propositions about what is permissible, forbidden, obligatory, etc.[42]

5.2 Future Work

5.2.1 Semantic Information Retrieval

Legal information retrieval is different from general information retrieval in that it is normally done by well educated users, who are familiar with the structure of the information being retrieved. When retrieving information, it is crucial that the IR system does not miss potentially relevant information. At the same time, the amount of legal information available is huge, making the task of sorting out the less relevant parts by hand prohibitively expensive. Put in traditional IR terms, the user of a legal information has high requirements on the *recall* of the system, i.e. it's ability to return all relevant information in a query. But for practical reasons, the system must also have good *precision*, i.e. the ability to filter out non-relevant information.[43]

Increasing an IR systems recall often decreases it's precision, and vice versa. Therefore the *F-measure* is a useful metric when evaluating IR system performance.[44] The F-measure is a score based upon a weighting between the precision and the recall of the system. The two can be differently weighed depending on how important the precision or recall is for the usage of the system. The standard weighting, which places equal importance on the precision and the recall, is called the F_1 measure, but for legal information retrieval, where recall is more important, the F_2 measure might be a better evaluation metric as it weights recall twice as much as precision. It is defined as:

$$F_2 = \frac{5 * (\text{precision} * \text{recall})}{4 * \text{precision} + \text{recall}}$$

41 E.g. the definition of OWL DL, http://www.w3.org/TR/2004/REC-owl-features-20040210/#s1.3. See also the different profiles in OWL 2 and their relationship to different families of description logics, http://www.w3.org/TR/owl2-profiles/.

42 McNamara, Paul, "Deontic logic", in the Stanford Encyclopedia of Philosophy, http://plato.stanford.edu/entries/logic-deontic/ .

43 Manning et al, "Introduction to Information Retrieval", p 142 (2008).

44 Manning et al, ibid., p 144.

In order to achieve a good F_2 score, the quality of the query matters. Having information marked up using semantic web models makes it possible to design query user interfaces that let the user craft very precise questions. To list all legal cases from the last five years that reference a particular Swedish Act and also refer to the European Convention on Human Rights would require the user to select the type(s) of resources being searched for (legal cases), the properties of those resources (date within the last five years), and how those resources should refer to other resources.

If the information is structured using vocabularies with machine-readable semantics such as SKOS or OWL, even more advanced queries become possible.

5.2.2 "Mix and Match" Legal Information Services

Most legal information services can be evaluated according to three different aspects:

1. The breadth and completeness of public legal information such as statutes and legal cases
2. The amount and quality of private legal information such as statute commentary.
3. The features and user interface of the service

One service may have a superior user interface, while another may have a wealth of useful commentaries and a third may have a complete archive of legal cases. The user may prefer to work in the user interface of one service, but needs the information contained in another service.

When using standardized resource URIs and metadata, it becomes possible for the user interface of one system to handle structured data from another, creating a federated or combined information system. This might not always be possible for business reasons, but if the standards are generally adopted, the technological possibility can create a *tiered legal information market*, where some actors focus on providing well structured publically available information, others add value to this by annotating and commenting on it, and still others package the two kinds of information in a variety of usable interfaces. The end user would then be able to pick and choose components to create a custom legal information service.