Chapter 6

The Semantic Web Publishing Vocabulary

Graph names provide the hooks for asserting meta-information about distinct graphs. In order to use the graph naming mechanism for representing information together with quality-related meta-information, the Named Graphs data model has to be supplemented with vocabularies for expressing the types of meta-information that are relevant within an application domain. One type of meta-information which is commonly required in the context of Web-based information systems is provenance information about the origin of information, e.g. who said what and when.

This section introduces the *Semantic Web Publishing Vocabulary (SWP)*, an RDF-Schema vocabulary for expressing information provision related meta-information and for assuring the origin of information with digital signatures. The vocabulary has been developed together with Jeremy Carroll (Hewlett Packard Labs, United Kingdom), Patrick Stickler (Nokia, Finland), and Pat Hayes (Institute for Human and Machine Cognition, United States).

The Semantic Web Publishing Vocabulary is designed for information syndication processes in which information is passed through multiple intermediaries. These syndication processes imply three basic roles:

Information Providers publish information in various forms. Information providers have different degrees of commitment towards published information, e.g. they might believe information to be true or might be in doubt about the reliability of published information. In order to prove the origin of information and to ensure that information is not altered in the syndication process, information providers can digitally sign information.

Information Syndicators are intermediaries who collect information from

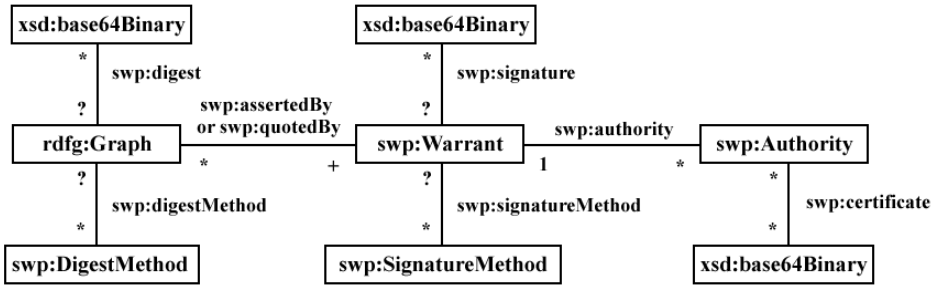

Figure 6.1: The Semantic Web Publishing Vocabulary (SWP).

multiple information providers and distribute collected information to information consumers or other syndicators. Information syndicators might add meta-information about the syndication process to syndicated information. They are not committed to the truth of information, as they are merely quoting other sources.

Information Consumers receive information directly from information providers or through information syndicators. For assessing the quality of received information, information consumers require meta-information about the origin of information and the syndication process. In order to verify the origin of information, information consumers might require information to be digitally signed.

Figure 6.1 gives an overview of the Semantic Web Publishing Vocabulary. The vocabulary consists of two parts: The first part defines terms for authorizing information and for representing information provision related meta-information. The second part defines terms for representing digital signatures. The namespace of the SWP vocabulary is `http://www.w3.org/2004/03/trix/swp-2/`. In the following, the SWP namespace is abbreviated with the prefix `swp:`.

6.1 Authorizing Named Graphs

The basic idea of the SWP vocabulary is to record the authorizing relationship between a named graph and an *authority* in the form of a *warrant*. An authorizing relationship means that the authority in some sense commits itself to the content of the graph. The SWP vocabulary provides terms for representing different propositional attitudes, such as asserting or quoting, towards a graph. Warrants may also record other properties of an authorizing relationship such as the validity- or expiry date.

Term	*Description*
swp:Authority	Class of all authorities. Information providers as well as information syndicators may act as authorities.
swp:Warrant	Class of all warrants. A warrant records an authorizing relationship between a graph and an authority.
swp:assertedBy	The subject graph is asserted by the authority specified for the object warrant. The triples of the graph are taken to be claims made by that authority.
swp:quotedBy	The triples of the graph are quoted by the authority and are thus not taken to be claims made by that authority.
swp:authority	Defines the authority of a warrant.
swp:validFrom	Defines the start of the validity period of a warrant.
swp:validUntil	Defines the end of the validity period of a warrant.
swp:sourceURL	URL for retrieving a representation of the graph. This URL may be used if the graph cannot be retrieved by dereferencing the graph name URI reference.

Table 6.1: SWP terms for authorizing graphs.

Table 6.1 gives an overview of the SWP terms for authorizing graphs. The `swp:authority` property relates warrants to authorities. The `swp:assertedBy` and `swp:quotedBy` properties capture the propositional attitude of the relationship between a graph and a warrant. These take a named graph as a subject and a `swp:Warrant` as object; `swp:authority` takes a warrant as a subject and a `swp:Authority` as an object. Each warrant must have a unique authority. Intuitively, `swp:assertedBy` means that the warrant records an endorsement or assertion that the graph is true, while `swp:quotedBy` means that the graph is being presented without any comment being made on its truth.

Figure 6.2 shows an example graph set which uses the SWP vocabulary for representing authorizing relationships. The first graph `ex:Graph1` contains a triple recommending to sell the stock with the identifier `<urn:x-ISIN:US4581401001>` (line 5). The graph is asserted by a warrant with the authority `<mailto:reynolds@ft.com>` (lines 6-7). Lines 8 and 9 define the validity period of the warrant. `ex:Graph1` is a self-asserting graph, as line 6 contains the triple `ex:Graph1 swp:assertedBy ex:Graph1`. Self-asserting graphs will be called *warrant graphs* in the following. The third graph (lines 16-22) quotes `ex:Graph1` and `ex:Graph2`. The graph is asserted by the information syndicator `<http://information-syndicator.com>`.

```
1.  @prefix swp: <http://www.w3.org/2004/03/trix/swp-2/> .
2.  @prefix ex: <http://www.fu-berlin/suhl/bizer/exampleDataset> .
3.
4.  ex:Graph1 {
5.      <urn:ISIN:US4581401001> ex:rating ex:Sell .
6.      ex:Graph1 swp:assertedBy ex:Graph1 .
7.      ex:Graph1 swp:authority <mailto:reynolds@ft.com> .
8.      ex:Graph1 swp:valid-from "2005-11-20"^^xsd:date
9.      ex:Graph1 swp:valid-until "2005-11-30"^^xsd:date
10. }
11. ex:Graph2 {
12.      <urn:ISIN:US4581401001> ex:rating ex:Buy .
13.      ex:Graph2 swp:assertedBy ex:Graph2 .
14.      ex:Graph2 swp:authority <mailto:richard@miller.com> .
15. }
16. ex:Graph3 {
17.      ex:Graph1 swp:quotedBy ex:Graph3 .
18.      ex:Graph1 swp:sourceURL <http://www.moodys.com/rec45.rdf>
19.      ex:Graph2 swp:quotedBy ex:Graph3 .
20.      ex:Graph2 swp:sourceURL <http://www.finBlog.com/post32.rdf>
21.      ex:Graph3 swp:assertedBy ex:Graph3 .
22.      ex:Graph3 swp:authority <http://information-syndicator.com> .
23. }
```

Figure 6.2: Graph set using the Semantic Web Publishing Vocabulary for representing authorizing relationships

6.2 Signing Named Graphs

In order to prove the origin of information and to ensure that information is not altered in the syndication process, information providers may decide to digitally sign named graphs. A digital signature of a named graph is formed by computing a digest of the named graph and by signing this digest using a standard signature algorithm like DSA [FIP95b] or RSA [KS98]. Graph signatures are verified by recomputing the digest of the signed graph and by decoding the original digest from the signature using the public key of the information provider. If both digests are equal, it is proved that the graph originates from the holder of the public key and that it has not been altered in the syndication process.

Content syndication processes, where information is passed over multiple information syndicators, pose special requirements on digital signatures for named graphs:

1. Information syndicators combine graphs from different sources into

graph sets and may forward these graph sets using a different serialization syntaxes as the original documents containing the graphs. Therefore, a graph signature should still be verifiable if a graph is contained in a different graph set and if this graph set is serialized using a different syntax.

2. The RDF data model does not assign any semantic to blank node identifiers [KC04]. Two graphs that do not share blank node identifiers may still be semantically equivalent within an application context. Therefore, graph signatures should be independent from blank nodes identifiers and should still be verifiable if blank nodes are renamed.

The Semantic Web Publishing Vocabulary provides terms for representing digital signatures, for indicating the signature method that was used to compute a signature, and for representing cryptographic keys and certificates. Table 6.2 gives an overview about the signature-related terms of the SWP vocabulary. Graph signatures are attached to warrants using the `swp:signature` property. The value of the `swp:signature` property is an RDF literal representing the signature of the graph that is asserted or quoted by the warrant. The signature is encoded using the base64 algorithm [FB96]. The `swp:signatureMethod` property identifies the signature method that was used to calculate the signature.

Computing a digital signature for a large amount of data is expensive. Therefore, it is common practice to calculate a digest of the data and to sign this digest instead of the original data [FIP95b]. There have been two proposals for computing syntax- and blank node identifier-independent digests of RDF graphs:

- Jeremy Carroll proposes an algorithm for transforming semantically equivalent graphs into a canonical serialization [Car03]. The algorithm renames blank nodes in a uniform fashion and sorts triples into a lexical order. After canonicalizing a graph, its digest can be computed from the canonical serialization using a standard hash function like SHA1 [FIP95a] or MD5 [Riv92].

- Craig Sayers and Alan Karp propose a second algorithm for computing digests of RDF graphs [SK04]. The algorithm does not rely on an intermediate canonical serialization but computes separate hash values from each triple in the graph. These values are aggregated into a set hash afterwards. Blank node identifier independence is achieved by adding triples that capture the original blank node identifiers to the graph before calculating the set hash. These triples are used in the signature

Property	*Description*
swp:signature	The value of this property is the signature to be used to authenticate the graphs with which the subject warrant is associated.
swp:signatureMethod	The value of this property is the signature method by which the signature specified for the subject warrant was constructed.
swp:digest	The value of this property contains a digest value for the subject graph.
swp:digestMethod	The value is the digest method by which the digest value specified for the graph subject was constructed.
swp:hasKey	The value is some kind of public key which belongs to the authority. The key is represented by an XML literal containing a XML Signature keyInfo element.
swp:certificate	The value is the base64 encoding of a binary (ASN.1 DER) X.509 certificate containing the public key of the authority.

Table 6.2: Signature-related terms of the SWP vocabulary.

verification process to temporary relabel blank nodes with their original identifiers. Sayers and Karp do not specify a single hash and aggregation function for computing the set hash, but discuss the advantages and disadvantages of different options. For instance, SHA1 [FIP95a] or MD5 [Riv92] could be used for hashing triples and XOR, multiplication or addition could be used for aggregating the resulting hash values into a set hash.

Both methods can be used to calculate the digest of an RDF graph that is part of a named graph. But as a named graph consists of an RDF graph and a graph name, the graph name has to be reflected in the digest as well. One option to achieve this is to separately digest the graph name and the RDF graph and to combine both digests afterwards using an aggregation function like XOR.

The SWP vocabulary provides terms for describing which combination of canonicalization-, digest-, aggregation-, and signature-algorithms is used to compute a signature. Each of these *signature methods* is identified by a URI reference. Table 6.3 summarizes the signature methods that are defined by the SWP vocabulary. The signature method `swp:swp:JjcC14N-sha1-xor-dsa`, for instance, indicates that a signature is

URI Reference	*Description*
swp:JjcC14N-sha1-xor-dsa	Signature method combining Carroll's C14N algorithm with the SHA1 digest function, XOR as aggregation function and the DSA signature function.
swp:JjcC14N-sha1-xor-rsa	Signature method combining Carroll's C14N algorithm with the SHA1 digest function, XOR as aggregation function and the RSA signature function.
swp:JjcC14N-md5-xor-dsa	Signature method combining Carroll's C14N algorithm with the MD5 digest function, XOR as aggregation function and the DSA signature function.
swp:SaKaDig-sha1-xor-dsa	Signature method combining Sayer & Karp's digest algorithm, using the SHA1 hash function and the XOR aggregation function, with the DSA signature function.
swp:SaKaDig-sha1-xor-rsa	Signature method combining Sayer & Karp's digest algorithm, using the SHA1 hash and the XOR aggregation function, with the RSA signature function.

Table 6.3: URI references for identifying RDF signature methods.

formed by transforming the RDF graph of a named graph using Carroll's canonicalization algorithm [Car03], digesting the canonical serialization and the graph name using the SHA1 [FIP95a] hash function, combining both digests using XOR as aggregation function and finality signing the digest using the DSA [FIP95b] signature function.

For verifying the signature of a named graph, the information consumer requires the public key or digital certificate [HPFS02] of the information provider. The SWP vocabulary defines terms for adding public keys and certificates to published information. A `SWPAuthority` may have a `swp:hasKey` property. The value of this property is some kind of public key which belongs to the authority. For representing keys, the SWP vocabulary reuses the `keyInfo` data structure from the XML signature recommendation [ERS02]. Within RDF, the `keyInfo` element is represented as an XML literal [KC04]. The `swp:certificate` property is used to represent digital certificates of an `SWPAuthority`. The value of the `swp:certificate` property is the base64 encoding [FB96] of a binary (ASN.1 DER) X.509 certificate [HPFS02] containing

```
1.  @prefix swp: <http://www.w3.org/2004/03/trix/swp-2/> .
2.  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
3.  @prefix ex: <http://www.fu-berlin/suhl/bizer/exampleDataset> .
4.
5.  ex:SignedGraph {
6.      <urn:ISIN:US4581401001> ex:rating ex:Sell .
7.      ex:SignedGraph swp:assertedBy ex:SignedGraph .
8.      ex:SignedGraph swp:authority <mailto:reynolds@ft.com> .
9.      ex:SignedGraph swp:valid-from "2005-11-20"^^xsd:date .
10.     ex:SignedGraph swp:valid-until "2005-11-30"^^xsd:date .
11.     ex:SignedGraph swp:signatureMethod swp:JjcC14N-md5-xor-rsa .
12.     ex:SignedGraph swp:signature
13.             "AZ8QWEJ05HaDsh4iHYmsJfMDVl..."^^xsd:base64Binary .
14. }
```

Figure 6.3: Signing a single graph.

the public key of the authority.

Figure 6.3 shows how the Semantic Web Publishing Vocabulary is used to represent the signature of a named graph. The named graph `ex:SignedGraph` is signed by the authority `<mailto:reynolds@ft.com>`. Line 11 asserts that the signature method `swp:JjcC14N-sha1-xor-rsa` was used to calculate the signature given in line 13. Note that the `swp:signatureMethod` triple is added to the graph before the digest calculation in order to be able to detect subsequently changed signature methods. The `swp:signature` triple is added after the digest calculation. Before verifying the signature, this triple has to be removed from the graph again.

Beside of signing single graphs, the SWP vocabulary also provides for signing warrant graphs which assert or quote multiple graphs. All graphs that are asserted or quoted by a warrant graph will be called *warranted graphs* in the following. For ensuring the integrity of these warranted graphs, SWP uses a similar technique as XML signature [ERS02]: First, the digests of all warranted graphs are calculated. These digests are added to the warrant graph as `<NameOfWarrantedGraph> swp:digest "DigestValue"` triples. The method that is used to compute a digest is indicated by a `<NameOfWarrantedGraph> swp:digestMethod <digestMethod>` triple for each digest. The SWP vocabulary defines URI references for identifying several digest methods. These URI references are explained in Table 6.4.

After adding the `swp:digest` and `swp:digestMethod` triples, the warrant graph is signed as described above. For verifying the integrity of the warranted graphs, an information consumer first verifies the signature of the warrant graph. Afterwards, the information consumer recalculates the digests of

URI Reference	*Description*
swp:JjcC14N-sha1-xor	Digest method combining Carroll's C14N algorithm, the SHA1 digest function and the XOR aggregation function.
swp:JjcC14N-md5-xor	Digest method combining Carroll's C14N algorithm, the MD5 digest function and XOR as aggregation function.
swp:SaKaDig-sha1-xor	Digest method proposed by Craig Sayer and Alan Karp using the SHA1 hash function and XOR as aggregation function.

Table 6.4: URI references for identifying RDF digest methods.

the warranted graphs using the indicated digest methods. The integrity of the warrented graphs is ensured, if the recalculated values equal the values given by the `swp:digest` triples.

Figure 6.4 shows a graph set consisting of two warranted graphs `ex:WarrantedGraph1` and `ex:WarrantedGraph2` and a warrant graph `ex:WarrantGraph`. Line 12 contains the information that `ex:WarrantedGraph1` is asserted by `ex:WarrantGraph`. Line 14 and 15 contain the digest for `ex:WarrantedGraph1`. The method that was used to compute the digest is indicated in line 13.

6.3 Related Work

This section compares the SWP vocabulary with two related standards: The Dublin Core Element Set [ISO03a] and XML Signature Syntax [ERS02].

6.3.1 Dublin Core Element Set

Similar to SWP warrants, the Dublin Core [ISO03a] elements `dc:creator`, `dc:publisher` and `dc:contributor` relate information resources to persons or institutions. The difference between both vocabularies lies in types of representable relations: The Semantic Web Publishing Vocabulary is focused on the commitment of an authority towards the truth of information. Asserting a graph implies a claim by the authority that the content of the graph is true. In contrast, the Dublin Core terms focus on the role of a person or institution in the process of creating an information resource. Thus, Dublin Core elements do not imply anything about the truth of created information.

A second difference lies in the way both vocabularies are used within

```
1.  @prefix swp: <http://www.w3.org/2004/03/trix/swp-2/> .
2.  @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
3.  @prefix ex: <http://www.fu-berlin/suhl/bizer/exampleDataset> .
4.
5.  ex:WarrantedGraph1 {
6.      <urn:ISIN:US4581401001> ex:rating ex:Sell .
7.  }
8.  ex:WarrantedGraph2 {
9.      <urn:ISIN:DE0007236102> ex:rating ex:Buy .
10. }
11. ex:WarrantGraph {
12.      ex:WarrantedGraph1 swp:assertedBy ex:WarrantGraph .
13.      ex:WarrantedGraph1 swp:digestMethod swp:JjcC14N-sha1-xor .
14.      ex:WarrantedGraph1 swp:digest
15.             "qZk+NkcGgWq6PiVxeF..."^^xsd:base64Binary .
16.      ex:WarrantedGraph2 swp:assertedBy ex:WarrantGraph .
17.      ex:WarrantedGraph2 swp:digestMethod swp:JjcC14N-sha1-xor .
18.      ex:WarrantedGraph2 swp:digest
19.             "kpRyejYS4uxwT9I74F..."^^xsd:base64Binary .
20.      ex:WarrantGraph swp:assertedBy ex:WarrantGraph .
21.      ex:WarrantGraph swp:authority <mailto:reynolds@ft.com> .
22.      ex:WarrantGraph swp:signatureMethod swp:JjcC14N-md5-xor-dsa .
23.      ex:WarrantGraph swp:signature
24.             "i6GB+VsWq5fJKzQcBB4..."^^xsd:base64Binary .
25.      <mailto:reynolds@ft.com> swp:certificate
26.             "iVxeFDJ0..."^^xsd:base64Binary .
27. }
```

Figure 6.4: Signing multiple graphs with a single signature.

RDF. The Dublin Core working draft Expressing Dublin Core Metadata using the Resource Description Framework (RDF) [NPJN06] specifies that Dublin Core elements are used as predicates of RDF triples describing a resource, for instance `<resource> dc:creator "Name"`. The SWP vocabulary captures a relationship between an authority and an information resource using warrants as an additional level of indirection. This reification of the relationship allows the relationship to be described using additional properties, such as validity and expiry date.

6.3.2 XML-Signature Syntax

The W3C XML-Signature Syntax and Processing recommendation [ERS02] defines a vocabulary for describing the process of computing a digital signature from arbitrary, URL-addressable data and for representing digital

signatures, public keys, and certificates in the form of XML elements. The design of the Semantic Web Publishing Vocabulary was inspired by XML signature. They main difference between both approaches is that the Semantic Web Publishing vocabulary represents signatures as RDF, which facilitates the processing of SWP signatures within RDF-oriented applications. Second, XML signature provides terms for identifying XML canonicalization methods but does not define terms or identifying RDF-specific canonicalization and digest methods. The SWP vocabulary closes this gap by defining terms for identifying RDF-specific methods.