XPath Profile for the Provenance Query Protocol

Status of this Memo

This document provides information to the community regarding the specification of a profile for using XPath in querying the provenance of data items from process documentation and has the status of a working draft. It does not define any standards or technical recommendations. Distribution is unlimited.

Copyright Notice

Copyright 2006.

Abstract

The provenance query protocol [MMG+06] has been defined, and includes the request for, algorithm to execute and result from a provenance query, as executed by a provenance query engine. Many parts of the request document are unspecified, being dependent on the provenance query engine implementation. This document defines an XPath-based profile by which provenance queries can be fully specified against process documentation that is in, or can be mapped to, XML format.
## Contents

1 Introduction
   1.1 Goals and Requirements ........................................... 3
      1.1.1 Requirements .................................................. 3

2 Terminology and Notation ............................................. 3
   2.1 XML Namespaces .................................................... 3
   2.2 Notational Conventions ............................................ 4
   2.3 XML Schema Diagrams .............................................. 4
   2.4 XPath notation ..................................................... 5

3 XPath Query Data Handle ............................................. 5

4 Single Node XPath Data Accessor ................................... 7

5 XPath Relationship Target Filter .................................. 8
1 Introduction

The provenance query protocol [MMG+06] has been defined, and includes the request for, algorithm to execute and result from a provenance query, as executed by a provenance query engine. Many parts of the request document are unspecified, being dependent on the provenance query engine implementation. This document defines an XPath-based profile by which provenance queries can be fully specified against process documentation that is in, or can be mapped to, XML format.

This document defines a method by which query data handles can be specified as XPath expressions evaluated against the p-structure, data accessors can be specified as XPaths that are usable by the provenance query engine, and relationship target filters can be expressed using XPaths over relationship targets.

1.1 Goals and Requirements

The goal of this document is to define extensions to the provenance query protocol to give a complete XPath-based approach to expressing and evaluating such queries.

1.1.1 Requirements

In meeting this goal, this document must address the following requirements:

- Define the schema of an XPath query data handle.
- Define the schema of an unambiguous (so usable by the provenance query) XPath-based data accessor.
- Define the schema of an XPath relationship target filter.

2 Terminology and Notation

All definitions for the concepts and structures found within this document can be found in [TGJ+06].

2.1 XML Namespaces

The XML Namespace URI that must be used by implementations of this specification is: http://www.pasoa.org/schemas/version023s1/xquery/XQuery.xsd

Table 1 lists XML namespaces that are used in this specification. The choice of any namespace prefix is arbitrary and not semantically significant.
Table 1: Prefixes and XML Namespaces used in this specification

<table>
<thead>
<tr>
<th>Prefix</th>
<th>XML Namespace</th>
<th>Specification(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>xp</td>
<td><a href="http://www.pasoa.org/schemas/version023s1/pquery/XPathPQuery.xsd">http://www.pasoa.org/schemas/version023s1/pquery/XPathPQuery.xsd</a></td>
<td>[XPath]</td>
</tr>
<tr>
<td>pq</td>
<td><a href="http://www.pasoa.org/schemas/version023s1/pquery/ProvenanceQuery.xsd">http://www.pasoa.org/schemas/version023s1/pquery/ProvenanceQuery.xsd</a></td>
<td>[PQuery]</td>
</tr>
<tr>
<td>ps</td>
<td><a href="http://www.pasoa.org/schemas/version023s1/PStruct.xsd">http://www.pasoa.org/schemas/version023s1/PStruct.xsd</a></td>
<td>[PStruct]</td>
</tr>
<tr>
<td>xs</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
<td>[XMLSchema]</td>
</tr>
</tbody>
</table>

2.2 Notational Conventions

The keywords “MUST ”, “MUSTNOT ”, “REQUIRED ”, “SHALL ”, “SHALLNOT ”, “SHOULD ”, “SHOULDNOT ”, “RECOMMENDED ”, “MAY ”, and “OPTIONAL ” in this document are to be interpreted as described in [Bra97].

2.3 XML Schema Diagrams

This documents adopts a graphical notation to describe XML Schema. Figure 1 gives an example of a small XML Schema displayed as a diagram, which is now explained with reference to the figure.

Figure 1 defines the structure of type ts:Test. The type Test contains a sequence of elements, which we now detail. One element in the sequence is ts: testName, which can be any type and must occur once and only once in an instance of ts: Test. ts: Name is followed by element ts: testNumber, which must contain a string. The ts: testNumber element must occur at least once and can occur as many times as needed. This is denoted by the “1..unbounded” under the element. Finally, the sequence contains a choice between two elements, ts: startTest and ts: stopTest, either of which must contain a date.

Below is a simple of description of each of the parts of the XML Schema diagram format.
An element (instance) is represented by the qualified name of the element in the box. By default an element must occur once and only once. Where this restriction does not hold, the text “1..unbounded”, “0..unbounded”, “0..N”, “1..N” (where N is an integer) appears under the element box. The left hand number is the minimum occurrences of the element at the position in the XML document, the right hand number is the maximum (with “unbounded” for no maximum).

A complex type is denoted by a lightly marked box with the qualified name of the type at the top left. The structure of the type is given by the elements, types and control structures within the box.

A horizontal sequence of dots represents a sequence of elements or control structures, that must appear in an element conforming to the type in the surrounding type box.

A vertical sequence of dots represents a choice between elements or control structures, that must appear in an element conforming to the type in the surrounding type box.

### 2.4 XPath notation

In addition to the XML Schema diagrams, an XPath notation [W3C99] is used to identify each individual element in the specification along with its context, in order to describe precisely its meaning and use.

### 3 XPath Query Data Handle

A query data handle can be expressed as an XPath over the p-structure. On evaluating this XPath on a set of process documentation following the p-structure, it will return a set of nodes, which should be interaction or actor state p-assertions or nodes within interaction or actor state p-assertion contents. These are the start data items for the provenance query algorithm. Note that while a provenance query is primarily intended to find the provenance of a single item, there is nothing preventing a query data handle from referring to multiple items, and the
provenance of all of these will be determined and returned by the provenance query engine.

An XPath search is a document instantiating the schema shown in Figure 2.

```
/xp:xpath
    Contains an XPath definition.

/xp:xpath/xp:path
    Contains the actual XPath string itself.

/xp:xpath/xp:namespaceMapping
    Contains a mapping from a prefix used in the XPath string to a namespace.

/xp:xpath/xp:namespaceMapping/xp:prefix
    Contains the prefix in a prefix-to-namespace mapping.

/xp:xpath/xp:namespaceMapping/xp:namespace
    Contains the namespace in a prefix-to-namespace mapping.
```

The following is an example of an XPath query data handle. It finds an element named "ex:data" in the sender view of interactions with a given message sink.

```xml
<pq:queryDataHandle>
    <pq:search>
        <xp:path/>
        <xp:namespaceMapping/>
        <xp:namespaceMapping/>
        <xp:namespaceMapping/>
    </pq:search>
</pq:queryDataHandle>
```
4 Single Node XPath Data Accessor

A data accessor for which the operations required by a provenance query engine can be supported is a single node XPath. This is an XPath, using a subset of the XPath notation, which explicitly refers to one node in a p-assertion’s content and has a normalised form which can be directly compared with other single node XPaths.

A single node XPath part is made up of one of the following concatenated sequences of text strings:

- A slash (“/”), a namespace prefix followed by a colon, e.g. “ns:”, an element name, e.g. “element”, an integer index in brackets, e.g. “[1]”.

- A slash (“/”), an at symbol (“@”), a namespace prefix followed by a colon, e.g. “ns:”, an attribute name, e.g. “attribute”.

- A slash (“/”), the string “text()”, an integer index in brackets, e.g. “[1]”.

A single node XPath is made up of a concatenated sequence of single node XPath parts (where only the first form may be followed by other forms). The single node XPath is a document instantiating the schema shown in Figure 3.

/xp:singleNodeXPath

Contains an XPath definition where the path is a single node XPath.

In order to enable the required data accessor operations to be performed, we define a normalised form of a single node XPath. This form is obtained by
replacing each namespace prefix, e.g. "ns:" with the namespace it denotes in braces, e.g. "\{http://www.example.org\}”.

The operations required for a provenance query engine to use this data accessor are performed as follows.

**Get Accessor For Item** The data accessor for a node in a p-assertion’s XML content is obtained by constructing a single node XPath part for each element from the root of the content down to the node, and concatenating them.

**Test Accessor Equality** Two single node XPath data accessors are equal if their normalised forms are exact matches.

The following is an example of a single node XPath data accessor. It refers to a particular element named “ex:data” inside a p-assertion’s content.

```xml
<ps:dataAccessor>
  <xp:singleNodeXPath>
    <xp:path>/ex:envelope[0]/ex:store[0]/ex:data[0]</xp:path>
    <xp:namespaceMapping>
      <xp:prefix>ex</xp:prefix>
    </xp:namespaceMapping>
  </xp:singleNodeXPath>
</ps:dataAccessor>
```

### 5 XPath Relationship Target Filter

A relationship target filter can be expressed as an XPath over documents instantiating the relationship target schema. If the XPath evaluates to 1 or more nodes, then the result of the filter is true, i.e. the relationship target is in scope. If the XPath evaluates to 0 nodes, then the result of the filter is false, i.e. the relationship target is out of scope.

When expressed as an XPath, the relationship target filter follows exactly the same format as given for the XPath Query Data Handle in Section 3, as shown in Figure 2.

The following is an example of an XPath relationship target filter. It returns true for a given relationship target only if the parameter name of the relationship target is not “http://www.example.com#divisor”, i.e. it excludes all divisor data items from the scope of the provenance query.

```xml
<pq:relationshipTargetFilter>
  <pq:search>
    <xp:xpath>
      <xp:path>/pq:relationshipTarget[ps:parameterName!="http://www.example.com#divisor"]</xp:path>
      <xp:namespaceMapping>
        <xp:prefix>ps</xp:prefix>
      </xp:namespaceMapping>
    </xp:xpath>
  </pq:search>
</pq:relationshipTargetFilter>
```
References


