

ws-prov-sign

**Authors:**

Victor Tan, U. Southampton  
Simon Miles, U. Southampton  
Steve Munroe, U. Southampton  
Paul Groth, U. Southampton  
Sheng Jiang, U. Southampton  
John Ibbotson, IBM  
Luc Moreau, U. Southampton

August 30, 2006

# A Profile for Non-Repudiable Process Documentation

## Status of this Memo

This document provides information to the community regarding a profile for creating process documentation that is non-repudiable through the use of signatures, 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 data model for process documentation [MGJ<sup>+</sup>06] describes p-assertions as individual units for documenting process. These p-assertions can be cryptographically signed by asserting actors in order to establish accountability for their creation. This document extends on the data model for the basic p-assertions (relationship, actor-state and interaction) to include support for signatures.

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Goals and Requirements . . . . .	3
1.1.1	Requirements . . . . .	3
1.1.2	Non-Requirements . . . . .	3
<b>2</b>	<b>Terminology and Notation</b>	<b>3</b>
2.1	XML Namespaces . . . . .	4
2.2	Notational Conventions . . . . .	4
2.3	XML Schema Diagrams . . . . .	4
2.4	XPath notation . . . . .	5
<b>3</b>	<b>Signatures in P-Assertions</b>	<b>6</b>
3.1	Signed Interaction P-Assertion . . . . .	6
3.2	Signed Actor State P-Assertion . . . . .	7
3.3	Signed Relationship P-Assertion . . . . .	7
<b>4</b>	<b>Conclusion</b>	<b>8</b>
<b>A</b>	<b>Schema for Signed Process Documentation</b>	<b>10</b>

# 1 Introduction

The data model for process documentation [MGJ<sup>+</sup>06] provides descriptions of different ways in which processes may be documented. Individual items of process documentation, p-assertions, are created by asserting actors with unique identities. Signing these p-assertions links their identities to these p-assertions and establishes accountability, and subsequent liability, for their creation. This document shows how the data model and schema for p-assertions is modified to include a Signature element, and how this element is used in establishing non-repudiation.

## 1.1 Goals and Requirements

The goal of this document is to provide an extended data model for p-assertions which permits support for the use of signatures.

### 1.1.1 Requirements

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

- Explain how signatures in the extended data model can be used for non-repudiation

### 1.1.2 Non-Requirements

This document does not intend to meet the following requirements:

- Explain alternative mechanisms (other than signatures) to achieve non-repudiation.
- Prescribe measures to be undertaken if p-assertions are not signed when they are expected to be, or if the signature on a p-assertion is determined to be invalid.
- Propose a technology specific approach to implementing signatures within the data model.

# 2 Terminology and Notation

All definitions for the concepts and structures found within this document can be found in [TGJ<sup>+</sup>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/PStruct.xsd`

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

Prefix	XML Namespace	Specification(s)
ps	<code>http://www.pasoa.org/schemas/version023s1/PStruct.xsd</code>	[P-Structure]
wsa	<code>http://schemas.xmlsoap.org/ws/2004/08/addressing</code>	[WS-Addressing]
xs	<code>http://www.w3.org/2001/XMLSchema</code>	[XMLSchema]

Table 1: Prefixes and XML Namespaces used in this specification

## 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 document 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.

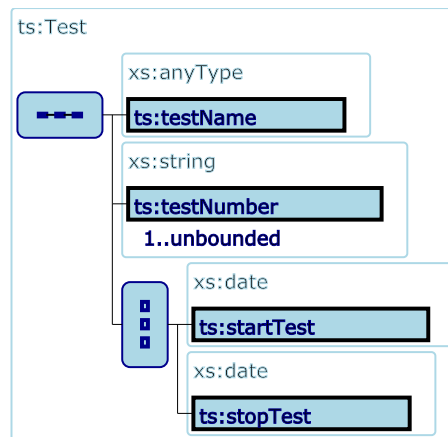


Figure 1: An example XML Schema diagram

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 Signatures in P-Assertions

The provenance of a data item can only be determined through the analysis of its related process documentation. However, process documentation in the form of p-assertions created by an actor only represents that actor's subjective view of events in the process it is documenting. It is therefore important that actors be held accountable for p-assertions they create, since p-assertions with erroneous information will affect other information processing functions. If such accountability is desirable in a specific provenance aware application, it can be made compulsory for actors to sign p-assertions they create. This establishes a direct link between the signed p-assertions and an actor's unique identity, and makes it impossible for it to deny having created a specific p-assertion (the property of non-repudiation).

The use of signatures assumes that actors operate within an environment that supports the use of a public key infrastructure through which certificates are distributed. The verification of a signature on a p-assertion can be achieved by the repository to which p-assertions are submitted for storage, or by querying actors that retrieve p-assertions from this repository. The use of signatures does not guarantee the truth (or otherwise) of the contents of a p-assertion, it only establishes the identity of the actor involved in its creation. In an application environment where accountability is vital, a provenance system that stores or processes p-assertions may be configured to reject p-assertions that are unsigned or whose signatures are invalid.

The determination of how accurately a p-assertion describes an actual event in a past process is unrelated to the use of signatures and has to be achieved through alternative means. This may, for example, involve comparing p-assertions relating to the same message or actor state made by asserting actors with different levels of trustworthiness.

#### 3.1 Signed Interaction P-Assertion

The model for a signed interaction p-assertion is shown in Figure 2.

The basic addition to the data model for a normal interaction p-assertion described in Section 3.3 in [MGJ<sup>+</sup>06] is the `ps:Signature` element. This contains the signature of the asserting actor in a manner appropriate to the representational format of the p-assertion. If the interaction p-assertion is structured as XML, signing is performed using the XML Signature Recommendation [BBF<sup>+</sup>02], where the `ds:Signature` element from the XML Signature schema (<http://www.w3.org/TR/xmlsig-core/> `xmlsig-core-schema.xsd`) is used directly in place of the `ps:Signature` element.

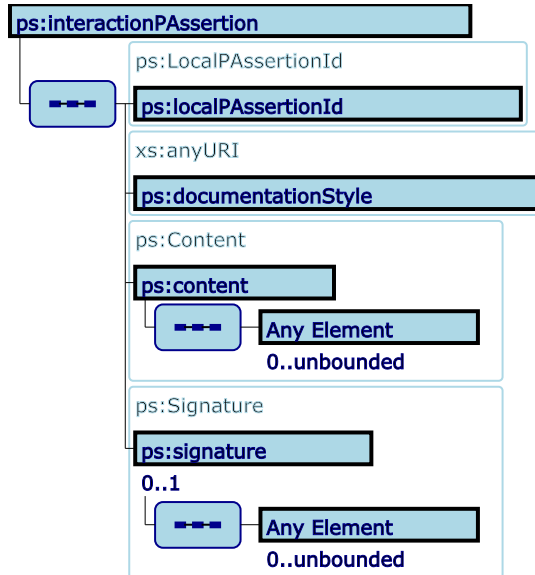


Figure 2: Signed Interaction P-Assertion

### 3.2 Signed Actor State P-Assertion

The model for a signed actor state p-assertion is shown in Figure 3.

The basic addition to the data model for a normal actor state p-assertion described in Section 3.5 in [MGJ<sup>+</sup>06] is the `ps:Signature` element. This contains the signature of the asserting actor in a manner appropriate to the representational format of the p-assertion. If the actor state p-assertion is structured as XML, signing is performed using the XML Signature Recommendation [BBF<sup>+</sup>02], where the `ds:Signature` element from the XML Signature schema (<http://www.w3.org/TR/xmlsig-core/xmlsig-core-schema.xsd>) is used directly in place of the `ps:Signature` element.

### 3.3 Signed Relationship P-Assertion

The model for a signed relationship p-assertion is shown in Figure 4.

The basic addition to the data model for a normal relationship p-assertion described in Section 3.6 in [MGJ<sup>+</sup>06] is the `ps:Signature` element. This contains the signature of the asserting actor in a manner appropriate to the representational format of the p-assertion. If the relationship p-assertion is structured as XML, signing is performed using the XML Signature Recommendation [BBF<sup>+</sup>02], where the `ds:Signature` element from the XML Signature schema (<http://www.w3.org/TR/xmlsig-core/xmlsig-core-schema.xsd>) is used directly in place of the `ps:Signature` element.

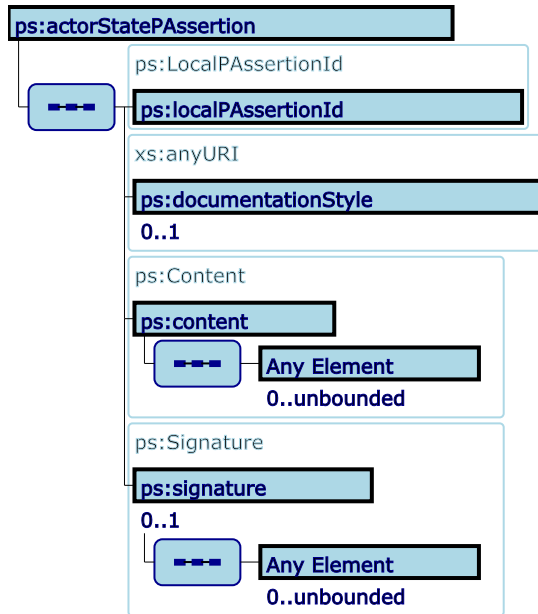


Figure 3: Signed Actor State P-Assertion

## 4 Conclusion

In this document, we have described how the data model and schema for process documentation [MGJ<sup>+</sup>06] can be extended in order to accommodate the inclusion of a Signature element. The inclusion of this element allows p-assertions to be signed, which links the identities of the asserting actors to these p-assertions and ensures that these actors cannot subsequently deny accountability for their contents (the property of non-repudiation). This is important as process documentation in the form of p-assertions created by an actor merely represents that actor’s subjective view of events in the process it is documenting.

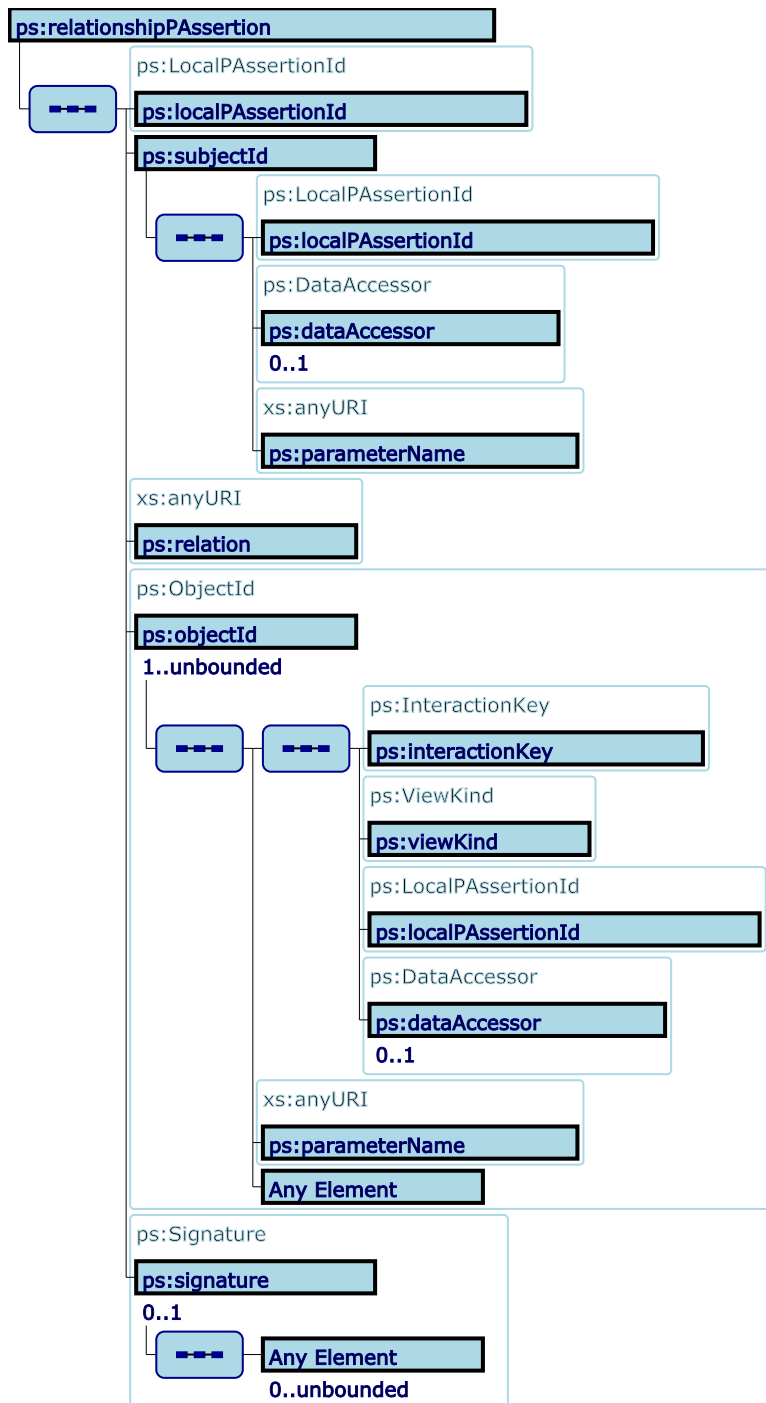


Figure 4: Signed Relationship P-Assertion

# A Schema for Signed Process Documentation

Below we give the full schema for the data model of process documentation that includes signatures for interaction, relationship and actor state p-assertions.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
xmlns:ps="http://www.pasoa.org/schemas/version023s1/PStruct.xsd"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
  attributeFormDefault="unqualified" elementFormDefault="qualified"
  targetNamespace="http://www.pasoa.org/schemas/version023s1/PStruct.xsd">

  <xs:import
    namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
    schemaLocation="./wsaddressing.xsd"/>

  <xs:annotation>
    <xs:documentation>
      The P-Structure. This is a logical view of the contents of a provenance store.
      The P-Structure contains a set of interaction records that document interactions
      between actors.

      Author: Paul Groth

      Copyright (c) 2006 University of Southampton
      See pasoalicense.txt for license information.
      http://www.opensource.org/licenses/mit-license.php
    </xs:documentation>
  </xs:annotation>

  <!-- We define the global elements of the p-structure here so that
    They can be referenced by external schemas. Below we define
    the types of the p-structure. The prefix ps: refers to this
    document. -->
  <xs:element name="pstruct" type="ps:PStructure">
    <xs:annotation>
      <xs:documentation>
        (Root Element Start Here) An instance of the
        p-structure. Each instance of the p-structure contains a
        set of interaction records.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:element name="interactionRecord" type="ps:InteractionRecord">
    <xs:annotation>
      <xs:documentation>
        An interaction record describes the client and service
        view of a particular interaction. An interaction record
        is identified by an interaction key.
      </xs:documentation>
    </xs:annotation>
  </xs:element>

  <xs:element name="view" type="ps:View">
    <xs:annotation>
      <xs:documentation>
        A view of an interaction by an actor.
      </xs:documentation>
    </xs:annotation>
  </xs:element>
```

```

</xs:element>

<xs:element name="interactionKey" type="ps:InteractionKey">
  <xs:annotation>
    <xs:documentation>
      An interaction key contains a message source,
      a message sink, and an interaction id.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="messageSource" type="wsa:EndpointReferenceType">
  <xs:annotation>
    <xs:documentation>
      The source of the message within the sender.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="messageSink" type="wsa:EndpointReferenceType">
  <xs:annotation>
    <xs:documentation>
      The sink of the message within the receiver.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="interactionId" type="xs:anyURI">
  <xs:annotation>
    <xs:documentation>
      A URI that uniquely identifies this interaction .
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="asserter" type="ps:Asserter">
  <xs:annotation>
    <xs:documentation>
      Each view (either client or service) comes from a
      particular actor. The actor that asserts p-assertion
      in a particular view is termed the asserter. The identity
      of the asserter is documented in the corresponding view inside
      the interaction record.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="numberOfExpectedAssertions" type="ps:NumberOfExpectedAssertions">
  <xs:annotation>
    <xs:documentation>
      The number of expected p-assertions to be contained
      within a view as documented by the asserting actor.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<!-- The following elements define the three types of p-assertions. -->
<xs:element name="interactionPAssertion" type="ps:InteractionPAssertion">
  <xs:annotation>
    <xs:documentation>
      Assertion as to the content of an interaction.
    </xs:documentation>
  </xs:annotation>
</xs:element>

```

```

<xs:element name="actorStatePAssertion" type="ps:ActorStatePAssertion">
  <xs:annotation>
    <xs:documentation>
      Information supplied by an actor about its state in the
      context of this interaction . Examples include the
      script that was used in running a service or the time
      when an invocation was sent/received.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="relationshipPAssertion" type="ps:RelationshipPAssertion">
  <xs:annotation>
    <xs:documentation>
      Describes a relationship between a p-assertion recorded
      in this view and another p-assertion made by the
      asserting actor. This can be seen as a triple: subject
      identifier, relation, object identifier.
    </xs:documentation>
  </xs:annotation>
</xs:element>
<!-- End P-assertion defintions -->

<xs:element name="viewKind" type="ps:ViewKind">
  <xs:annotation>
    <xs:documentation>
      Whether a view is from the sender or receiver.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="localPAssertionId" type="ps:LocalPAssertionId">
  <xs:annotation>
    <xs:documentation>
      Uniquely identifies a p-assertion within a view.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="dataAccessor" type="ps:DataAccessor">
  <xs:annotation>
    <xs:documentation>
      An application dependent mechanism for referencing a
      piece of data within a p-assertion.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="parameterName" type="xs:anyURI">
  <xs:annotation>
    <xs:documentation>
      The parameter name of a data item referenced
      in a relationship p-assertion.
    </xs:documentation>
  </xs:annotation>
</xs:element>

<xs:element name="documentationStyle" type="xs:anyURI">
  <xs:annotation>
    <xs:documentation>
      The style of documentation used when recording
      an interaction p-assertion.
    </xs:documentation>
  </xs:annotation>
</xs:element>

```

```

<xs:element name="pAssertionDataKey" type="ps:PAssertionDataKey"/>
<xs:element name="objectId" type="ps:ObjectId"/>
<xs:element name="relation" type="xs:anyURI"/>
<xs:element name="globalPAssertionKey" type="ps:GlobalPAssertionKey"/>
<xs:element name="interactionMetaData" type="ps:InteractionMetaData"/>
<xs:element name="interactionContext" type="ps:InteractionContext"/>
<xs:element name="senderViewKind" type="ps:SenderViewKind"/>
<xs:element name="exposedInteractionMetaData" type="ps:ExposedInteractionMetaData"/>

<!-- Type Definitions -->
<xs:complexType name="InteractionKey">
  <xs:sequence>
    <xs:element ref="ps:messageSource"/>
    <xs:element ref="ps:messageSink"/>
    <xs:element ref="ps:interactionId"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="PStructure">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" minOccurs="0" ref="ps:interactionRecord"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="InteractionRecord">
  <xs:sequence>
    <xs:element ref="ps:interactionKey" />
    <xs:element minOccurs="0" name="sender" type="ps:View">
      <xs:annotation>
        <xs:documentation>
          The senders's view of the interaction .
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element minOccurs="0" name="receiver" type="ps:View">
      <xs:annotation>
        <xs:documentation>
          The receiver's view of the interaction.
          WARNING!!! In future revisions the receiver view
          may not be allowed to include relationship
          p-assertions. If you have an example of the
          usage of relationship p-assertions in this view,
          please contact the authors of the schema.
          Thanks!
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:any namespace="##other" processContents="lax"
      minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Asserter">
  <xs:sequence>
    <xs:any namespace="##other" maxOccurs="unbounded"
      minOccurs="0" />
  </xs:sequence>
</xs:complexType>

```

```

<xs:complexType name="Asserter">
  <xs:sequence>
    <xs:any namespace="##other" maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="NumberOfExpectedAssertions">
  <xs:restriction base="xs:positiveInteger"/>
</xs:simpleType>

<!-- Following the WS-Security spec, we allow any type of identiification -->

<xs:complexType name="View">
  <xs:sequence>
    <xs:element ref="ps:asserter" />
    <xs:choice maxOccurs="unbounded" minOccurs="0">
      <xs:element maxOccurs="unbounded" minOccurs="0"
        ref="ps:interactionPAssertion" />
      <xs:element maxOccurs="unbounded" minOccurs="0"
        ref="ps:relationshipPAssertion" />
      <xs:element maxOccurs="unbounded" minOccurs="0"
        ref="ps:actorStatePAssertion" />
      <xs:element maxOccurs="unbounded" minOccurs="0"
        ref="ps:exposedInteractionMetaData"/>
    </xs:choice>

    <xs:any namespace="##other" processContents="lax"
      minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="RelationshipPAssertion">
  <xs:sequence>
    <xs:element ref="ps:localPAssertionId"/>
    <xs:element name="subjectId">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="ps:localPAssertionId"/>
          <xs:element ref="ps:dataAccessor" minOccurs="0" />
          <xs:element ref="ps:parameterName"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element ref="ps:relation"/>
    <xs:element maxOccurs="unbounded" ref="ps:objectId"/>
    <xs:element maxOccurs="1" minOccurs="0" name="signature" type="ps:Signature" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="GlobalPAssertionKey">
  <xs:sequence>
    <xs:element ref="ps:interactionKey"/>
    <xs:element ref="ps:viewKind"/>
    <xs:element ref="ps:localPAssertionId"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="PAssertionDataKey">
  <xs:complexContent>
    <xs:extension base="ps:GlobalPAssertionKey">
      <xs:sequence>

```

```

        <xs:element minOccurs="0" ref="ps:dataAccessor"/>
    </xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>

<xs:complexType name="InteractionPAssertion">
    <xs:sequence>
        <xs:element maxOccurs="1" minOccurs="1" ref="ps:localPAssertionId"/>
        <xs:element maxOccurs="1" minOccurs="1" ref="ps:documentationStyle"/>
        <xs:element maxOccurs="1" minOccurs="1" name="content" type="ps:Content"/>
        <xs:element maxOccurs="1" minOccurs="0" name="signature" type="ps:Signature"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="ActorStatePAssertion">
    <xs:sequence>
        <xs:element maxOccurs="1" minOccurs="1" ref="ps:localPAssertionId"/>
        <xs:element maxOccurs="1" minOccurs="0" ref="ps:documentationStyle"/>
        <xs:element maxOccurs="1" minOccurs="1" name="content" type="ps:Content"/>
        <xs:element maxOccurs="1" minOccurs="0" name="signature" type="ps:Signature"/>
    </xs:sequence>
</xs:complexType>

<xs:simpleType name="LocalPAssertionId">
    <xs:union memberTypes="xs:long xs:string xs:anyURI"/>
</xs:simpleType>

<xs:complexType name="ViewKind" abstract="true">
    <xs:annotation>
        <xs:documentation>
            Instance documents must select something that is derived
        </xs:documentation>
    </xs:annotation>
</xs:complexType>

<xs:complexType name="SenderViewKind">
    <xs:complexContent>
        <xs:restriction base="ps:ViewKind"/></xs:restriction>
    </xs:complexContent>
</xs:complexType>

<xs:complexType name="ReceiverViewKind">
    <xs:complexContent>
        <xs:restriction base="ps:ViewKind"/></xs:restriction>
    </xs:complexContent>
</xs:complexType>

<xs:complexType name="ObjectId">
    <xs:complexContent>
        <xs:extension base="ps:PAssertionDataKey">
            <xs:sequence>
                <xs:element ref="ps:parameterName"/>
                <xs:any namespace="##other" processContents="lax"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<xs:complexType name="DataAccessor">
    <xs:sequence>
        <xs:any maxOccurs="unbounded" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

```

```

<xs:complexType name="Content">
  <xs:sequence>
    <xs:any namespace="##any" maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="InteractionMetaData">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element name="tracer" type="xs:anyURI"/>
      <xs:any namespace="##other" maxOccurs="unbounded" minOccurs="0"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="InteractionContext">
  <xs:sequence>
    <xs:element ref="ps:interactionKey" />
    <xs:element ref="ps:viewKind"/> <!-- View Kind of the actor who created the metadata -->
    <xs:element ref="ps:interactionMetaData"
      maxOccurs="unbounded" minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="ExposedInteractionMetaData">
  <xs:sequence>
    <xs:element ref="ps:globalPAssertionKey"/>
    <xs:element ref="ps:interactionMetaData"/>
    <xs:element maxOccurs="1" minOccurs="0" name="signature" type="ps:Signature" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Signature">
  <xs:sequence>
    <xs:any namespace="##any" maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

</xs:schema>

```

## References

- [BBF<sup>+</sup>02] Mark Bartel, John Boyer, Barb Fox, Brian LaMacchia, and Ed Simon. XML Signature Syntax and Processing. <http://www.w3.org/TR/xmlsig-core/>, 2002.
- [Bra97] Scott Bradner. Key words for use in RFCs to indicate requirement levels. <http://www.ietf.org/rfc/rfc2119.txt>, 1997.
- [MGJ<sup>+</sup>06] Steve Munroe, Paul Groth, Sheng Jiang, Simon Miles, Victor Tan, and Luc Moreau. Data model for Process Documentation. Technical report, University of Southampton, June 2006.

- [TGJ<sup>+</sup>06] Victor Tan, Paul Groth, Sheng Jiang, Simon Miles, Steve Munroe, and Luc Moreau. WS Provenance Glossary. Technical report, Electronics and Computer Science, University of Southampton, 2006.
- [W3C99] W3C. XML Path Language (XPath) Version 1.0. W3C Recommendation 16 November 1999. <http://www.w3.org/TR/xpath>, 1999.