HR-XML Consortium, Inc.

Extension Best Practices

HR-XML CONSORTIUM

Table of Contents

Extending Schemas 3

Design Requirements 3

Extension Schema Design 3

Use of Business Object Documents (BODs) 4

Wrapper Extension Method 5

Wrapper Constraints 5

Wrapper Example Schema 5

Wrapper Example Instance 6

ANY Extension Method 7

ANY Constraints 7

ANY Element Extension Schema Defined and Declared 7

ANY Example Schema 8

Extension Example Definition 9

ANY Example Instance 9

Appendix A - Namespace Extension Method 11

Position Statement 11

Namespace Extension Method Explained 11

Sample Invalid Instance 11

Extending Schemas

HR-XML Schemas cannot be all things to all people. In the real world of data interchange, there arises the need to include the content models that support real business transactions. In addition, how can implementers experiment with new data structures that lie outside the existing approved specifications? The recognition of the first statement and the need to answer the question give rise to the background for extending schemas.

Given that extension is a reality, how can we accommodate extensions without undermining the principle of open standards? This document is meant to provide guidance on the practice for extending schemas.

Its goal is to show:

1) Official endorsement of different methods for implementation Timely domain releases to public.
2) Conventions for creating extensions to encourage consistency.

Design Requirements

All extensions to approved HR-XML standards SHOULD be done consistently across the Consortium. They MUST retain the basic values of XML, such as validation, especially in the context of any Certification process. Extension methods MUST not lead to an undermining of the open standards mission.

Extensions

As discussed here, “extending” is meant to “add additional elements and attributes to an existing schema”. This is not to be confused with how it is used in the **XML Schemas: Best Practices** discussion, where “extending” means adding functionality to schema that does not currently exist. The latter is for such functionality as being able to verify that the value of element <A> is greater than the value of element <B>. This document refers to “extending” in the former usage only.

Extension Schema Design

The Technical Steering Committee has approved two methods that enable extension of HR-XML schemas without undermining an open standards mission. The “wrapper” and “ANY” techniques are explained herein. Additionally, a “Namespace” extension method was examined and not endorsed.

Use of Business Object Documents (BODs)

The Consortium recommends the use of Business Object Documents (BODs) when exchanging data. However, it is recognized that some organizations may use the Nouns directly rather than use BODs. Refer to the specific extension methods for BOD and Noun extension best practices.

Wrapper Extension Method

This method consists of a schema that uses an HR-XML schema as an <import>. This enables the intact use of HR-XML schemas and can be implemented without any accommodation required by the work group during the design phase. A drawback is the potential for many different root elements in the real world of multiple trading partners.

Wrapper Constraints

1) This extension method should be limited to Nouns. In other words, content extensions must not be created in a wrapper of the BOD. To maintain consistency with the BOD architecture, the BOD architecture requires that:

    a. The BOD contain the ApplicationArea and Data Area
    b. The DataArea contain the Verb and Noun

2) Extension nodes defined separately MAY conform to a file naming convention of:

    a. UserArea1.xsd – first extension file name (i.e. with trading partner A)
    b. UserArea2.xsd – second extension file name (i.e. trading partner B)
    c. And so on…

3) Extensions MUST NOT be used in any certification process.

Wrapper Example Schema

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:hr="http://www.hr-xml.org/3"
targetNamespace="BrilliantHRSoftware.com"
xmlns:hr="BrilliantHRSoftware.com"
elementFormDefault="qualified">
<xsd:import schemaLocation="Candidate.xsd" xmlns="http://www.hr-xml.org/3"/>
  <xsd:element name="MyInternalDoc">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="hr:Candidate" />
        <xsd:element ref="SomeSpecialInfo"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="SomeSpecialInfo">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="WhyPersonWantsJob" type="xsd:string"/>/>
```

```
        <xsd:element name="WhyPersonFeelsTheyShouldBeHired" type="xsd:string"/>/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Wrapper Example Instance

```
<MyInternalDoc
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:hr="http://ns.hr-xml.org/3/Candidate"
xmlns="BrilliantHRSoftware.com"
xsi:schemaLocation="BrilliantHRSoftware.com BrilliantHRSoftware.xsd>
  <hr:Candidate>
    <hr:DocumentID>1067482948</hr:DocumentID>
    <hr:CandidatePerson>
      <hr:PersonName>
        <hr:FormattedName>Andrea Johnson</hr:FormattedName>
      </hr:PersonName>
    </hr:CandidatePerson>
    <hr:CandidateProfile>
      <hr:CandidateObjective>A position where I can do humanitarian works. </hr:CandidateObjective>
    </hr:CandidateProfile>
  </hr:Candidate>
  <SomeSpecialInfo>
    <WhyPersonWantsJob>I'd like to be part of your philanthropic culture. </WhyPersonWantsJob>
    <WhyPersonFeelsTheyShouldBeHired>I am a generous person, which fits well with your
culture.</WhyPersonFeelsTheyShouldBeHired>
  </SomeSpecialInfo>
</MyInternalDoc>
```

ANY Extension Method

This method consists of an HR-XML schema that contains a UserArea of data type "ANY". This structure is found in all nouns and components. It is used to house all extensions to the schema. Definitions of those extensions occur in a separate schema. The UserArea extension overcomes a drawback of the wrapper technique in that it maintains a consistent root element, which is a standard HR-XML defined element. It does, however, require work groups to accommodate by including an extension element in their schema design.

ANY Constraints

1) It is recommended where possible to limit the scope of the extension to the UserArea to which it applies. By limiting the scope of the document that knows about the extended schemas to a given UserArea only that UserArea can make use of the extended content.
2) Referencing the extension schema at the root element should be avoided. This inappropriately permits all UserAreas within the document to use the extended elements.
3) It must be named “UserArea”.
4) It MUST be the last element of the component.
5) It MUST have a minimum occurrence of zero and a maximum occurrence of one.
    a. <xsd:element ref="UserArea" minOccurs="0" maxOccurs="1"/>
6) Extension nodes defined separately MAY conform to a file naming convention of:
    a. UserArea1.xsd – first extension file name (i.e. with trading partner A)
    b. UserArea2.xsd – second extension file name (i.e. trading partner B)
    c. And so on…
7) Extensions MUST NOT be used in any certification process

ANY Element Extension Schema Defined and Declared

The XSD declaration for this element MUST conform to accepted schema design guidelines and scoped globally as such:

```
<xsd:element name="UserArea" type="UserAreaType"/>
<xsd:complexType name="UserAreaType">
  <xsd:sequence>
    <xsd:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

ANY Example Schema

```
<xsd:schema
xmlns="http://www.hr-xml.org/3"
xmlns:bhrs="BrilliantHRSoftware.com"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.hr-xml.org/3"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
<xsd:include schemaLocation="BrilliantHRSoftware.com/BrilliantHRSoftware.xsd "/>
  <xsd:element name="Candidate">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="DocumentID" type="IdentifierType" minOccurs="0"/>
        <xsd:element ref="CandidatePerson" minOccurs="0"/>
        <xsd:element ref="UserArea" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="CandidatePerson">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="PersonID" minOccurs="0"/>
        <xsd:element ref="PersonName" minOccurs="0"/>
        <xsd:element ref="UserArea" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="PersonName">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="FamilyName" type="string" minOccurs="0"/>
        <xsd:element name="GivenName" type="string" minOccurs="0"/>
        <xsd:element ref="UserArea" minOccurs="0"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="UserArea"/>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Extension Example Definition

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:bhrs="BrilliantHRSoftware.com"
targetNamespace="BrilliantHRSoftware.com"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xsd:element name="CandidateExtensions">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="HighestDegree" type="xsd:string"/>
        <xsd:element name="TotalYearsExperience" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="PersonNameExtensions">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="WhatIWishMyParentsHadNamedMe" type="xsd:string"/>
        <xsd:element name="WhatMyPetBirdCallsMe" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

ANY Example Instance

```
<Candidate
xmlns="http://ns.hr-xml.org/3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ns.hr-xml.org/3
Candidate.xsd
BrilliantHRSoftware.com
UserArea1.xsd">
  <DocumentID/>
  <CandidatePerson>
    <PersonID>78907</PersonID>
    <PersonName>
      <FamilyName>Johnson</FamilyName>
      <GivenName/>Andrea</GivenName>
      <UserArea>
        <bhrs:WhatIWishMyParentsHadNamedMe>Bubba</bhrs:WhatIWishMyParentsHadNamedMe>
        <bhrs:WhatMyPetBirdCallsMe>Tweet</bhrs:WhatMyPetBirdMe>
      </UserArea>
    </PersonName/>
```

```
    <UserArea>
      <bhrs:HighestDegree>Associate of Bird Watching</bhrs:HighestDegree>
      <bhrs:TotalYearsExperience>8</bhrs:TotalYearsExperience>
    </UserArea>
  </CandidatePerson>
</Candidate>
```

Appendix A - Namespace Extension Method

Position Statement

This method was examined by the Technical Steering Committee and was not endorsed. The central reason was due to the fact that "in context" extensions prevent a document from being properly valid. In essence, an XML parser cannot validate the data because extensions violate the content models of elements as defined in the schema. Since redefining elements was not an alternative, this method cannot be endorsed for use.

Namespace Extension Method Explained

This method uses **XML Namespaces** to differentiate elements of the HR-XML schema from extension elements. The elements are inserted in context within the HR-XML schema.

Sample Invalid Instance

```
<Candidate
xmlns="http://ns.hr-xml.org/3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ns.hr-xml.org/3
Candidate.xsd
BrilliantHRSoftware.com
BrilliantHRSoftware.xsd">
  <DocumentID>1067482948</DocumentID>
    <CandidatePerson>
      <PersonName>
        <FormattedName>Andrea Johnson</FormattedName>
        <JunkElement xmlns="BrilliantHRSoftware.com">Andrea is an Alias</JunkElement>
      </PersonName>
    </CandidatePerson>
</Candidate>
```

<JunkElement> is not in the content model of <Candidate> even though it is part of a different namespace.