SISO-REF-056-DRAFT

SISO Reference for XML Schema Naming and Design Best Practices

15 January 2016

Prepared by:
Simulation Interoperability Standards Organization Standards Activity Committee (SAC)
## Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Section</th>
<th>Date (MM/DD/YYYY)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revised Draft</td>
<td>All</td>
<td>01/15/2016</td>
<td>Revised Draft for community review</td>
</tr>
<tr>
<td>Initial Draft</td>
<td>All</td>
<td>09/10/2014</td>
<td>Initial Draft for community review</td>
</tr>
</tbody>
</table>
Table of Contents

1 INTRODUCTION................................................................................................................................. 7
  1.1 PURPOSE........................................................................................................................................... 7
  1.2 SCOPE ............................................................................................................................................... 7
  1.3 OBJECTIVES .................................................................................................................................... 8
  1.4 INTENDED AUDIENCE ..................................................................................................................... 8

2 REFERENCES......................................................................................................................................... 8
  2.1 SISO DOCUMENTS ........................................................................................................................... 8
  2.2 OTHER DOCUMENTS ...................................................................................................................... 8

3 DEFINITIONS..................................................................................................................................... 9

4 ACRONYMS AND ABBREVIATIONS ................................................................................................. 11

5 DOCUMENT CONVENTIONS (INFORMATIVE)... ................................................................................. 12
  5.1 DOCUMENT REFERENCES ............................................................................................................. 12
  5.2 NORMATIVE AND INFORMATIVE CONTENT ................................................................................ 12
  5.3 FORMATTING .............................................................................................................................. 13
  5.4 TERMINOLOGY ............................................................................................................................ 13
  5.4.1 Word Usage; Shall, Should, May and Can .................................................................................. 13
  5.4.2 XML Information Set Terminology .......................................................................................... 13
  5.4.3 XML Schema Terminology ...................................................................................................... 14
  5.4.4 XML Namespace Terminology ................................................................................................. 14
  5.5 DOCUMENT ORGANIZATION ....................................................................................................... 14

6 SISO XML CONFORMANCE TARGETS (NORMATIVE)................................................................... 15
  6.1 CONFORMANCE TARGETS OVERVIEW ...................................................................................... 15
  6.2 SISO REFERENCE XML SCHEMAS ............................................................................................ 15
  6.3 SISO SUBSET XML SCHEMAS ................................................................................................. 15
  6.4 SISO EXTENSION XML SCHEMAS AND EXCHANGE XML SCHEMAS ....................................... 16

7 GUIDING PRINCIPLES (INFORMATIVE)......................................................................................... 16
  7.1 SPECIFICATION GUIDELINES ...................................................................................................... 16
  7.1.1 Keep Specification to a Minimum ............................................................................................. 16
  7.1.2 Focus on Best practices for XML Schemas ........................................................................... 16
  7.1.3 Use Specific, Concise Best practices ....................................................................................... 17
  7.2 XML SCHEMA DESIGN GUIDELINES ......................................................................................... 17
  7.2.1 Disallow Content Modification with XML Processors ............................................................. 17
  7.2.2 Use XML Validating Parsers for Content Validation ................................................................. 17
  7.2.3 Validate for Conformance to Reference XML Schemas ......................................................... 17
  7.2.4 Allow Multiple XML Schemas for XML Constraints ............................................................ 18
  7.2.5 Define One Reference XML Schema per Namespace .............................................................. 18
  7.2.6 Disallow Mixed Content ........................................................................................................ 18
  7.2.7 Specify Types for All Constructs ............................................................................................ 18
  7.2.8 Avoid Wildcards in Reference XML Schemas .................................................................... 18
  7.2.9 Provide Default Reference XML Schema Locations ............................................................. 19
  7.2.10 Use Open Standards ............................................................................................................. 19
  7.3 MODELING DESIGN GUIDELINES .............................................................................................. 19
  7.3.1 Namespaces Enhance Reuse .................................................................................................. 19
  7.3.2 Design for Extensibility .......................................................................................................... 20
  7.4 IMPLEMENTATION GUIDELINES ............................................................................................... 20
  7.4.1 Avoid the Need to Display Raw XML Data ............................................................................. 20

Copyright © 2016 SISO. All rights reserved
This is an unapproved SISO Reference Product Draft, subject to change.
1 INTRODUCTION

The Extensible Markup Language (XML) has become a ubiquitous format for the structure and content of data in software applications and across the World Wide Web. Several standards products developed by the Simulation Interoperability Standards Organization (SISO) have used XML to specify the structure and content of data for storing and exchanging information (e.g., Base Object Model, Military Scenario Definition Language, Coalition Battle Management Language, Federation Engineering Agreements Template). These products were developed independently by SISO groups having diverse experience with XML, resulting in different approaches to the use of the language features. In parallel, major organizations, such as the U.S. Government’s National Information Exchange Model (NIEM) program, have developed guidelines, called Naming and Design Best practices (NDBP), for the use of XML intended to facilitate wide-spread understanding and employment while applying a common style. To apply a similar approach to create commonality across XML-based SISO products, this SISO XML Schema Naming and Design Best practices document specifies guidance and best practices for naming and designing components of XML schemas used in SISO products.

This SISO NDBP specifies a set of reusable information components for defining standard information exchange messages, transactions, and documents on a large scale: across multiple communities of interest. These reusable components are rendered in XML Schema documents as type, element, and attribute definitions that comply with the W3C XML Schema [3] specification. The resulting reference XML schemas are available to Modeling and Simulation (M&S) practitioners and developers at:

http://www.sisostds.org/schemas/

The W3C XML Schema [3] standard enables information interoperability and sharing by providing a common language for describing data precisely. The constructs it defines are basic metadata building blocks — baseline data types and structural components. Users employ these building blocks to describe their own domain-oriented data semantics and structures, as well as structures for specific information exchanges and components for reuse across multiple information exchanges. Best practices that profile allowable XML Schema constructs and describe how to use them help ensure that those components are consistent and reusable.

1.1 Purpose

The purpose of this document is to inform SISO product developers about best practices for the style of XML documents when XML is used to specify the structure and content of data supporting a SISO standard. This document is a reference that defines categories of principles and best practices to follow based on the documented selection of a "conformance target" by the SISO group developing the XML product. The group then uses this reference to tailor the principles and best practices to be followed in specific context to their XML products. The tailoring specifically includes changes to requirement levels (shall, should, will, etc.) of the principles and best practices.

The purpose includes the decision to establish SISO’s XML Schema Naming and Design Best Practices as a Reference Product to constrain its use to the scope determined by SISO groups developing XML products. The purpose does not include any scope of requirement related to the review or approval of an XML Product by any SISO committee.

1.2 Scope

This document was developed to specify XML Schema NDBP as they apply to SISO Standards, Guidance, and Reference Products. The document covers the following issues in depth:

- Guiding principles behind the design of SISO XML Schemas
- Best Practices for using XML Schema constructs in SISO products
- Best Practices for modeling and structuring SISO-conformant XML schemas
- Best Practices for creating SISO-conformant instances
- Best Practices for naming SISO XML components
1. Best Practices for extending SISO-conformant XML components

The best practices, principles and definitions contained in this document are intended to communicate the technical details of the XML schema as an information exchange format.

1.3 Objectives

The objectives of this document are to:

- Promote consistency in how SISO XML Schema products are designed.
- Facilitate XML product development and reduce authoring effort for the SISO product.
- Enable automated publication of SISO XML products in natural human readable formats that do not require XML expertise to understand.
- Provide a decision tree for SISO groups to deliberately use to identify and tailor the best practices they intend to follow in developing their XML products.

1.4 Intended Audience

This document targets diverse M&S communities, including developers, procurers, and users worldwide who employ SISO XML products for information exchange and interoperability. Such information exchanges may be between or within organizations. The SISO XML schema products provide system implementers context on which to build specific exchanges. The NDBP provides a methodology for groups to decide and designate the conformance targets and tailor best practices for their XML products.

This document specifies principles and best practices for SISO XML data components and XML schemas. This scope of this document does not include mandatory requirements to be followed by SISO groups.

2 REFERENCES

2.1 SISO Documents

<table>
<thead>
<tr>
<th>Document Number</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SISO-ADM-001-2014 Policy for Numbering of SISO Products</td>
</tr>
<tr>
<td>2</td>
<td>SISO-ADM-005-2011 Policy for the Style and Format of SISO Documents</td>
</tr>
</tbody>
</table>

2.2 Other Documents

<table>
<thead>
<tr>
<th>Document Number</th>
<th>Title</th>
</tr>
</thead>
</table>
### 3 DEFINITIONS

#### Table 3. Terms and Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapter type</td>
<td>An adapter type is a SISO-conformant XML type that adapts external components for use within SISO XML Guidance. An adapter type creates a new class of object that embodies a single concept composed of external components. A SISO-conformant XML schema defines an adapter type.</td>
</tr>
<tr>
<td>Conformance Target</td>
<td>An artifact such as a protocol, document, platform, process or service, which is the subject of Conformance Clauses and Normative Statements. There may be several Conformance Targets defined within a specification, and these targets may be diverse so as to reflect different aspects of a specification. For example, a protocol message and a protocol engine may be different targets.</td>
</tr>
<tr>
<td>Data definition</td>
<td>The data definition of a documented component is the content of the first occurrence of the element xsd:documentation, which is an immediate child of an occurrence of the element xsd:annotation, which is an immediate child of the element that defines the component.</td>
</tr>
</tbody>
</table>
In a SISO-conformant XML schema, a documented component is an XML Schema component that has an associated data definition. These XML schema components have a textual definition, so that the component may be well-understood. XML Schemas that do not document their components accordingly are not SISO-conformant.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchange XML schema</td>
<td>An exchange XML schema is an XML Schema document that meets all of the following criteria:</td>
</tr>
<tr>
<td></td>
<td>• It is explicitly designated as an exchange XML schema by the developing group.</td>
</tr>
<tr>
<td></td>
<td>• It provides the broadest, most fundamental definitions of components in its namespace.</td>
</tr>
<tr>
<td></td>
<td>• It provides the authoritative definition of business semantics for components in its namespace.</td>
</tr>
<tr>
<td></td>
<td>• It contains components that use or are derived from the components in reference XML schemas or exchange XML schemas.</td>
</tr>
<tr>
<td></td>
<td>• It is intended to identify and define the document element information item for a particular information exchange that is described by a SISO XML Schema product. It satisfies all best practices specified in the Naming and Design Best Practices for exchange XML schemas.</td>
</tr>
<tr>
<td>Extension XML schema</td>
<td>A SISO extension XML schema is an XML Schema document that meets all of the following criteria:</td>
</tr>
<tr>
<td></td>
<td>• It is explicitly designated as an extension XML schema by the developing group. It provides the broadest, most fundamental definitions of components in its namespace.</td>
</tr>
<tr>
<td></td>
<td>• It provides the authoritative definition of business semantics for components in its namespace.</td>
</tr>
<tr>
<td></td>
<td>• It contains components that use or are derived from the components in reference XML schemas or exchange XML schemas. It is intended to express the additional vocabulary required for an SISO XML Schema, above and beyond the vocabulary available from reference XML schemas, and to support XML Schema validation. It satisfies all best practices specified in the Naming and Design Best Practices for extension XML schemas.</td>
</tr>
<tr>
<td>External XML schema</td>
<td>An external XML schema is any XML schema that is not a supporting XML schema and that is not a SISO-conformant Product.</td>
</tr>
<tr>
<td>Reference element</td>
<td>A reference element is an element that refers to its value by a reference attribute instead of carrying it as content.</td>
</tr>
<tr>
<td>Reference XML schema</td>
<td>A reference XML schema is an XML Schema document that meets all of the following criteria:</td>
</tr>
<tr>
<td></td>
<td>• It is explicitly designated as a reference XML schema by the developing group. It provides the broadest, most fundamental definitions of components in its namespace.</td>
</tr>
<tr>
<td></td>
<td>• It provides the authoritative definition of business semantics for components in its namespace.</td>
</tr>
<tr>
<td></td>
<td>• It is intended to serve as the basis for components in SISO XML schemas, including subset XML schemas, constraint XML schemas, extension XML schemas, and exchange XML schemas.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic interoperability</td>
<td>The term semantic interoperability is here defined as &quot;the ability of two or more computer systems to exchange information and have the meaning of that information automatically interpreted by the receiving system accurately enough to produce useful results.&quot;</td>
</tr>
<tr>
<td>SISO-conformant XML component</td>
<td>A SISO-conformant XML component is an XML Schema component [3] that is defined by a reference XML schema, an extension XML schema, or an exchange XML schema.</td>
</tr>
<tr>
<td>SISO-conformant XML Schema</td>
<td>An XML Schema document is a SISO-conformant XML schema if and only if it is a reference XML schema, a subset XML schema, an extension XML schema, or an exchange XML schema. Neither constraint XML schemas nor subset XML schemas serve as the primary (cardinal) definitions for components they define. The primary definitions come from reference XML schemas, exchange XML schemas, and extension XML schemas. The XML Schema components defined by these XML schemas are SISO-conformant components.</td>
</tr>
<tr>
<td>Subset XML schema</td>
<td>A subset XML schema is an XML Schema document that meets all of the following criteria:</td>
</tr>
<tr>
<td></td>
<td>- It is explicitly designated as a subset XML schema by the developing group. This may be declared by an SISO XML schema catalog (reference Section 13 Normative - Deploying SISO XML Documents) or by a tool-specific mechanism outside the XML schema.</td>
</tr>
<tr>
<td></td>
<td>- It has a target namespace previously defined by a reference XML schema. That is, it does not provide original definitions for XML schema components, but instead provides an alternate XML schema representation of components that are defined by a reference XML schema.</td>
</tr>
<tr>
<td></td>
<td>- It does not alter the business semantics of components in its namespace. The reference XML schema defines these business semantics.</td>
</tr>
<tr>
<td></td>
<td>- It is intended to express the limited vocabulary necessary for a SISO XML schema product and to support XML Schema validation.</td>
</tr>
</tbody>
</table>

4 ACRONYMS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCC</td>
<td>Complex types with complex content</td>
</tr>
<tr>
<td>CON</td>
<td>Constraint XML schemas</td>
</tr>
<tr>
<td>CSC</td>
<td>Complex types with simple content</td>
</tr>
<tr>
<td>EBNF</td>
<td>Extended Backus-Naur Form</td>
</tr>
<tr>
<td>EXT</td>
<td>Extension and exchange XML schemas</td>
</tr>
<tr>
<td>IEPD</td>
<td>Information Exchange Package Documentation</td>
</tr>
<tr>
<td>INS</td>
<td>XML instance data</td>
</tr>
<tr>
<td>NDBP</td>
<td>Naming and Design Best Practices</td>
</tr>
<tr>
<td>REF</td>
<td>Reference XML schemas</td>
</tr>
</tbody>
</table>
5 DOCUMENT CONVENTIONS (INFORMATIVE)

This document uses formatting and syntactic conventions to clarify meaning and avoid ambiguity. These conventions build upon those specified in SISO-ADM-005 as described in the following sections.

5.1 Document References

This document relies on references to many outside documents. Such references are noted by bold, bracketed inline terms. For example, a reference to ISO/IEC 11179-4:2004 is shown as [1].

5.2 Normative and Informative Content

This document includes a variety of content. Some content is normative, while other content is informative. In general, the informative material appears as supporting text and specific rationales for the normative material.

Conventions used within this document include:

[Principle <number>]

A guiding principle for SISO NDBP.

The principles represent the requirements, concepts, and goals that have helped shape the SISO NDBP. Principles are informative, not normative, but act as the basis on which the best practices are defined.

Accompanying each principle is a short discussion section that justifies the application of the principle to SISO NDBP design.

Principles are numbered in the order in which they appear in the document.

[Best Practice <section>-<number> (<applicability>)]

A best practice for SISO NDBP (e.g., using techniques such as Schematron, “a language for making assertions about patterns found in XML documents”; see http://www.schematron.com/).

Best practices state specific requirements on artifacts, such as XML schemas and instances. Most best practices apply to conformant XML schemas, while others apply to instances. The best practices are normative.

Best practices are stated using both XML InfoSet terminology (elements and attributes) and XML Schema terminology (schema components). The choice of terminology is driven by which standard best expresses the best practices. Certain concepts are more clearly expressed using XML InfoSet information items, others using the XML Schema data model; still others are best expressed using a combination of terminology drawn from both standards.

Best practices have rationales that justify the need for the best practice. For clarity, there may be multiple best practices that have the same rationale.

Best practices and supporting text may use Extended Backus-Naur Form (EBNF) notation as defined by [10].

Best practices are numbered according to the section in which they appear and the order in which they appear within that section. For example, [Best practice 8-1] is the first best practice in Section 10 Normative - XML schema Modeling Best practices.
Each best practice is accompanied by a description of its applicability. This identifies the type of XML schema to which the best practice applies or indicates whether the best practice is applicable to XML documents or element information items. Each entry in the list is a code from Table 5: Codes Representing Conformance Targets. If a code appears in the applicability list for a best practice, then the best practice applies to the corresponding conformance target. The conformance targets are defined in Section 6, SISO XML Conformance.

5.3 Formatting

In addition to special formatting for definitions, principles, and best practices, this document uses consistent formatting to identify SISO XML components. Courier: All words appearing in Courier font are values, objects, keywords, or literal XML text. Italics: All words appearing in italics, when not titles or used for emphasis, are special terms with definitions appearing in this document. Keywords: Keywords reflect concepts or constructs expressed in the language of their source standard. Keywords have been given an identifying prefix to reflect their source. The following prefixes are used:

Throughout the document, fragments of XML Schema or XML instances are used to clarify a principle or best practice. These fragments are specially formatted in Courier font and appear in text boxes. An example of such a fragment follows:

```xml
<xsd:complexType name="ObjectType">
...
</xsd:complexType>
```

Figure 1: Example of an XML Fragment

5.4 Terminology

This document uses standard terminology to explain the principles and best practices that describe XML Schemas.

5.4.1 Word Usage; Shall, Should, May and Can

Within normative content (best practices and definitions), the key words Shall, Should, May and Can in this document are interpreted as defined in SISO-ADM-005.

5.4.2 XML Information Set Terminology

This document uses the concepts of element information items ("element"), attribute information items ("attribute"), and their associated properties as defined by [11] with clarifications as discussed below. Note that in the clarification that follows, the abstract property names appear in square brackets adjacent to the information items to which they belong. For example, "Element[parent+]" discusses the abstract property "parent" of the element information item.

- parent of an element (Element[parent])
- child of an element (Element[children])

Note that the InfoSet properties "Element[parent]" and "Element[children]" correspond to a direct, immediate relationship with an element. Children of an element and their children, and so on, are collectively referred to as descendants of that element. Parents of an element and their parents, and so on, are collectively referred to as ancestors of that element.
• element owning an attribute (Attribute[owner element])

The owner of an attribute is the element that possesses or contains the attribute.

The use of the term document element from [11] to describe the root of all elements in an XML document is preferred over the informal and nonstandard term root element.

5.4.3 XML Schema Terminology

The terms W3C XML Schema [3], XML Schema (upper case XML Schema”), and XSD all refer to the XML Schema definition language, as specified in the two-part XML Schema specification:

• XML Schema Part 2: Datatypes [14]

The term XML schema (lower case “schema”) refers to specific XML schema documents that conform to the XML Schema specifications listed above.

The terms XML instance and XML document refer to an XML instance document, which is defined by and validates to a particular XML schema.

The term XML schema component is defined in [3] as a building block for XML Schema. This document refers to, rather than restates, the definitions of the different XML schema components associated with the XML Schema Abstract Data Model, which are defined in the XML Schema specification. In this document, the name of the referenced XML schema component may appear without the suffix “schema component” (e.g., the term “complex type definition” may be used instead of “complex type definition XML schema component”) to enhance readability of the text.

The term NCName is defined in [14] and refers to XML noncolonized names, which are XML name strings that do not contain the “:” character.

5.4.4 XML Namespace Terminology

This document uses the concept of an XML Namespace as defined by [12] and [13].

5.5 Document Organization

This remainder of this document is organized into sections as follows:

• Section 6 SISO XML Conformance Targets (Normative) - describes terminology, requirements, and artifacts related to SISO conformance.
• Section 7 Guiding Principles - discusses the principles that serve as the foundation of and guidelines for the best practices.
• Section 8 Relation to Standards - discusses the use of the key standards used in the development of SISO.
• Section 9 Normative - XML Schema Design - discusses the best practices for using XML Schema constructs in SISO-conformant XML schemas.
• Section 10 Normative - XML schema Modeling - discusses the best practices for the additional structures and constraints needed to build SISO-conformant XML schemas.
• Section 11 Normative - XML Schema Naming - discuss the best practices used in naming SISO-conformant data components.

NOTE: The ordering of the sections is intended to minimize the number of forward references in the document. For this reason, the naming best practices appear as the last section of the document, so that the concepts being named have already been discussed.

• Annexes - Principles, best practices, and definitions.
6 SISO XML CONFORMANCE TARGETS (NORMATIVE)

This NDBP defines SISO XML conformance. This definition is performed through terminology definitions
and best practices. Together, these define several classes of XML schemas, as well as defining
conformance for XML instances of SISO-conformant XML schemas. Conformance in this context,
determines which class or classes of SISO XML schema an XML schema document conforms with.
These classes of XML schemas are defined, along with the definition of SISO conformance for XML
documents, in Section 6.1, Conformance Targets Overview, below. The XML schemas defined therein
are SISO-conformant XML schemas.

6.1 Conformance Targets Overview

The sections below define the conformance targets for this document. Each best practice in this
document is applicable to one or more of the conformance targets. SISO development groups should
designate their XML schemas as one or more of these conformance targets. The declaration of target
class provides other stakeholders with a context to understand, review and communicate the intended
application of a SISO XML Schema.

Throughout the document, each best practice definition contains a list of applicable conformance targets
(as described in Section 5.2, Normative and Informative Content, above). The best practice is binding for
the targets on this list. This list is normative. This list uses the following codes:

<table>
<thead>
<tr>
<th>Code</th>
<th>Conformance Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>REF</td>
<td>Reference XML schemas</td>
</tr>
<tr>
<td>SUB</td>
<td>Subset XML schemas</td>
</tr>
<tr>
<td>EXT</td>
<td>Extension and exchange XML schemas</td>
</tr>
<tr>
<td>CON</td>
<td>Constraint XML schemas</td>
</tr>
<tr>
<td>INS</td>
<td>XML instance data</td>
</tr>
</tbody>
</table>

6.2 SISO Reference XML Schemas

A SISO XML reference XML schema is a XML schema that shall be the authoritative definition XML
schema for a SISO XML namespace.

The best practices for reference XML schemas are more stringent than are the best practices for other
classes of SISO-conformant XML schemas. reference XML schemas are intended to support the
broadest reuse. They are uniform in their structure. As they are the primary definitions for data
components, they do not need to restrict other data definitions, and they are not allowed to use XML
Schema's restriction mechanisms. reference XML schemas are intended to be as regular and simple as
possible.

6.3 SISO Subset XML Schemas

A definition subset XML schema shall be based on another SISO-conformant XML
schema; a reference XML schema. A subset XML schema is defined such that any valid instance of the
subset XML schema is also a valid instance of the base (reference) XML schema. This means that a
subset XML schema is not allowed to introduce new content, nor is it allowed to extend the data content
defined by a component of the reference XML schema.

For example, a subset XML schema would not be allowed to introduce a new enumeration into a list of
enumerations (simple type restriction) defined by the reference XML schema. Any XML instance that
included the new enumeration would validate against the supposed subset XML schema but would not
validate against the reference XML schema. This would violate the basic premise underlying the use of
subsets: subsets shall be as restrictive as or more restrictive than the reference XML schema.
A **subset XML schema** may omit any construct of the base XML schema that has no effect on XML schema validation, including `xsd:documentation` and `xsd:appinfo` annotations. The reference XML schema on which a **subset XML schema** is based is considered the authoritative source of such annotations.

### 6.4 SISO Extension XML Schemas and Exchange XML Schemas

An **extension XML schema** serves several functions. First, it defines new content within a new namespace, which may be a SISO specific namespace or a namespace shared by external standards organization. This content is SISO-conformant but has fewer restrictions on it than do SISO reference XML schemas. Second, the **extension XML schema** bases its content on content from SISO reference XML schemas, where appropriate. Methods of deriving content include using (by reference) existing components, as well as creating extensions and restrictions of existing components.

For example, an SISO XML Schema may create a type for an SISO-specific phone number and base that type on a type defined by a SISO reference XML schema. This specific phone number type may restrict the SISO reference XML schema type to limit those possibilities that are permitted of the base type.

SISO extensions and restrictions shall include annotations and documentation to be conformant, but they are allowed to use some other constructs that are not allowed in SISO reference XML schemas.

### 7 GUIDING PRINCIPLES (INFORMATIVE)

Principles in this specification provide a foundation for the best practices. These principles are generally applicable in most cases. They should not be used as a replacement for common sense or appropriate special cases.

The principles are not operationally enforceable; they do not specify constraints on XML Schema documents and instances. The best practices are the normative and enforceable manifestation of the principles.

The principles discussed in this section are categorized as follows:

- Specification Guidelines
- XML Schema Design Guidelines
- Modeling Design Guidelines
- Implementation Guidelines

### 7.1 Specification Guidelines

The principles in this section address what material should be included in this NDBP and how it should be represented.

#### 7.1.1 Keep Specification to a Minimum

This specification states what is required for interoperability, not all that could be specified. Certain decisions (such as normative XML comments) could create roadblocks for interoperability, making heavy demands on systems for very little gain. The goal is not standardization for standardization’s sake. The goal is to maximize interoperability and reuse.

[Principle 1]

This reference specifies what is necessary for semantic interoperability and no more.

#### 7.1.2 Focus on Best practices for XML Schemas

This specification should try, as much as is possible, to specify XML schema-level content. This is a specification for XML schemas, and so it should specify XML schemas.
[Principle 2]

This specification SHOULD focus on providing best practices for specifying XML schemas.

7.1.3 Use Specific, Concise Best practices

A best practice should be as precise and specific as possible to avoid broad, hard-to-modify best practices. Putting multiple clauses in a best practice makes it harder to enforce. Using separate best practices allows specific conditions to be clearly stated.

[Principle 3]

This specification SHOULD feature best practices that are as specific, precise, and concise as possible.

7.2 XML Schema Design Guidelines

The principles in this section address how XML Schema technology should be used in designing SISO-conformant XML schemas and instances.

7.2.1 Disallow Content Modification with XML Processors

XML Schema has constructs that can make the data provided by XML processors different before and after XML schema processing. An example of this is the use of XML Schema attribute declarations with default values. Before XML schema validation, there may be no attribute value, but after processing, the attribute value exists.

Within SISO, the purpose of processing instances against XML schemas is solely validation: testing that data instances match desired constraints and guidelines. It should not be used to change the content of data instances.

[Principle 4]

The content of a SISO-conformant data instance SHOULD NOT be modified by processing against XML Schema documents.

7.2.2 Use XML Validating Parsers for Content Validation

SISO XML Schema Guidance is designed for XML Schema validation. A primary goal is to maximize the amount of validation that may be performed by XML Schema-validating parsers.

XML Schema validates content using content models: descriptions of what elements and attributes may be contained within an element, and what values are allowable. It is the XML element hierarchy (elements with attributes and unstructured content, contained by other elements) that the XML Schema definition language specifies and that XML Schema validating parsers can validate.

Mechanisms involving linking using attribute and element values are useful, but they should only be relied on when absolutely necessary, as XML Schema-validating parsers cannot readily validate them. For example, if a link is established via attribute values, an XML Schema-validating parser cannot determine that participants have appropriate type definitions. Whenever possible, SISO XML content should rely on XML syntax that can be validated with XML Schema.

[Principle 5]


7.2.3 Validate for Conformance to Reference XML Schemas

Systems that operate on XML data have the opportunity to perform multiple layers of processing. Middleware, XML libraries, XML schemas, and application software may process data. The primary purpose of XML Schema validation is to restrict processed data to that data that conforms to agreed-upon best practices. This restriction is achieved by marking as invalid that data that does not conform to the best practices defined by the XML schema.
[Principle 6]
Systems that use SISO-conformant data SHOULD mark as invalid data that does not conform to the best practices defined by applicable XML Schema documents.

7.2.4 Allow Multiple XML Schemas for XML Constraints
The SISO XML Schema Guidance does not attempt to create a one-size-fits-all XML schema to perform all validation. Instead, it creates a set of reference XML schemas, on which additional constraints may be placed. It also does not focus on language-binding XML Schema implementations, which convert XML Schema definitions into working programs. It is, instead, focused on normalizing language and preserving the meaning of data.

[Principle 7]
Constraints on XML instances MAY be validated by multiple XML schema validation passes, using multiple XML schemas for a single namespace.

7.2.5 Define One Reference XML Schema per Namespace
SISO uses the concept of a reference XML schema, which defines the structure and content of a namespace. For each SISO-conformant namespace, there is exactly one SISO reference XML schema. A user may use a subset XML schema or constraint XML schema in place of a reference XML schema, but all SISO-conformant XML documents should validate against a single reference XML schema for each namespace.

[Principle 8]
Each SISO-conformant namespace SHOULD be defined by exactly one reference XML schema.

7.2.6 Disallow Mixed Content
XML data that use mixed content are difficult to specify and complicate the task of data processing. Much of the payload carried by mixed content is unchecked and does not facilitate data standardization or validation.

[Principle 9]
SISO-conformant XML schemas SHOULD NOT specify data that uses mixed content.

7.2.7 Specify Types for All Constructs
XML Schema components within SISO all have names. This means that there are no anonymous types, elements, or other components defined by SISO. Once an application has determined the name (i.e., namespace and local name) of an attribute or element used in SISO-conformant instances, it will also know the type of that attribute or element.
There are no local attributes or elements defined by SISO, only global attributes and elements. This maximizes the ability of application developers to extend, restrict, or otherwise derive definitions of local components from SISO-conformant components. Using named global components in XML schemas maximizes the capacity for reuse.

[Principle 10]
SISO-conformant XML schemas SHOULD NOT use or define local or anonymous components, as they adversely affect reuse.

7.2.8 Avoid Wildcards in Reference XML Schemas
Wildcards in SISO-conformant XML schemas work in opposition to standardization. The goal of creating harmonized, standard XML schemas is to standardize definitions of data. The use of wildcard mechanisms (such as xsd:any, which allows insertion of an arbitrary number of elements from any namespace) allows nonstandard data to be passed via otherwise standardized exchanges.
Avoidance of wildcards in the standard XML schemas encourages the separation of standardized and non-standardized data. It encourages users to incorporate their data into SISO in a standardized way. It also encourages users to extend in a way that may be readily incorporated into SISO.

**[Principle 11]**

SISO-conformant components SHOULD NOT incorporate wildcards unless absolutely necessary, as they hinder standardization by encouraging use of non-standardized data rather than standardized data.

### 7.2.9 Provide Default Reference XML Schema Locations

SISO XML Schema provides three ways to specify the physical location of an XML Schema document: XML schemaLocation, an attribute of the element `xsd:import`, along with `xsi:schemaLocation` and `xsi:noNamespace XML SchemaLocation`, attributes of an XML Schema document element. In all of these uses, the specification explicitly maintains that the XML schema location specified is a hint, which may be overridden by applications.

**[Principle 12]**

XML Schema locations specified within SISO-conformant reference XML schemas SHOULD be interpreted as hints and as default values by processing applications.

### 7.2.10 Use Open Standards

The cooperative efforts of many knowledgeable individuals have resulted in many important published information standards. Where appropriate and applicable, SISO ought to leverage these standards.

**[Principle 13]**

SISO standards and XML schemas SHOULD leverage and enable use of other open standards.

### 7.3 Modeling Design Guidelines

The principles in this section address the design philosophy used in designing the SISO conceptual model.

#### 7.3.1 Namespaces Enhance Reuse

SISO is designed to maximize reuse of namespaces and the XML schemas that define them. When referring to a concept defined by SISO, a user should ensure that instances and XML schemas refer to the namespace defined by SISO. User-defined namespaces should be used for specializations and extension of SISO constructs but should not be used when the SISO structures are sufficient.

**[Principle 14]**

SISO-conformant instances and XML schemas SHOULD reuse components from SISO distribution XML schemas when possible.

SISO relies heavily on XML namespaces to prevent naming conflicts and clashes. Reuse of any component is always by reference to both its namespace and its local name. All SISO XML component names have global scope. Therefore, validation always occurs against the reference XML schemas or subsets thereof.

**Example:**

```xml
<xsd:element ref="nc:BinaryCaptureDate" minOccurs="0" maxOccurs="unbounded"/>
```

**Figure 2: Example of the use of a namespace**

In this example, `nc:BinaryCaptureDate` is reused by referencing its element declaration through both its namespace (which is bound to the prefix `nc:`) and its local name (`BinaryCaptureDate`). If an...
element named BinaryCaptureDate is declared in another namespace, it is an entirely different element than nc:BinaryCaptureDate. There is no implicit relationship to nc:BinaryCaptureDate. From a business perspective, the two elements are likely to be related in the sense that they may have very similar semantic meanings. They may have essentially the same meaning, but slightly different properties. Such a relationship may commonly exist. However, any relationship between the two elements should be made explicit using methods outlined in this document.

[Principle 15]

A component SHOULD be identified by its local name together with its namespace. A namespace SHOULD be a required part of the name of a component. A component's local name SHOULD NOT imply a relationship to components with similar names from other namespaces.

7.3.2 Design for Extensibility

SISO XML NDBP is designed to be extended. Numerous methods are considered acceptable in creating extended and specialized components.

[Principle 16]

SISO-conformant XML schemas and standards SHOULD be designed to encourage and ease extension and augmentation by users and developers outside the standardization process.

7.4 Implementation Guidelines

The principles in this section address issues pertaining to the implementation of software applications that use the SISO XML NDBP.

7.4.1 Avoid the Need to Display Raw XML Data

XML documents should be developed to enable the consistent use of human-friendly graphical user interfaces (GUIs) to view and publish XML data. Browsers and similar GUI technologies provide human-readable presentation of information stored in XML documents that do not require the human user to have an understanding of the specific syntax of XML.

[Principle 17]

XML documents SHOULD be designed to enable and facilitate automated and consistent processing of information.

7.4.2 Leave Implementation Decisions to Implementers

SISO XML NDBP is intended to be an open specification supported by many diverse implementations. It was designed from data requirements and not from or for any particular system or implementation. Use of SISO XML products should not depend on specific software, other than XML Schema-validating parsers.

[Principle 18]

SISO XML schemas SHOULD NOT depend on specific software packages, software frameworks, or software systems for interpretation of XML instances.

[Principle 19]

SISO XML schemas and standards SHOULD be designed such that software systems that use SISO XML products may be built with a variety of off-the-shelf and free software products.

7.5 Modeling Guidelines

The SISO XML NDBP specifies SISO-conformant components, XML schemas, and instances. These guidelines influence and shape the more-specific principles and best practices in this document. They are derived from best practices and from discussions within the SISO XML Drafting Group. This list may grow and evolve as SISO XML style guidance matures.
The principles in this section address decisions that data modelers may face when creating SISO-conformant XML schema representations of domain data. These guidelines are not absolute (the key word is SHOULD). It may not be possible to apply all guidelines in every case. However, they should always be considered.

### 7.5.1 Documentation

As will be described in later sections of this document, all SISO XML components are documented through their definitions and names. Although it is often very difficult to apply, a data component definition should be drafted before the data component name is finalized.

Drafting the definition for a data component first ensures that the author understands the exact nature of the entity or concept that the data component represents. The component name should subsequently be composed to summarize the definition. Reversing this sequence often results in data definitions that very precisely describe the component name but do not adequately describe the entity or concept that the component is designed to represent. This can lead to the ambiguous use of such components.

[Principle 20]

A data component definition SHOULD be drafted before the associated data element name is composed.

### 7.5.2 Consistent Naming

Components in SISO should be given names that are consistent with names of other SISO XML components. Having consistent names for components has several advantages:

1. It is easier to determine the nature of a component when it has a name that conveys the meaning and use of the component.
2. It is easier to find a component when it is named predictably.
3. It is easier to create a name for a component when clear guidelines exist.

[Principle 21]

Components in SISO SHOULD be given names that are consistent with names of other SISO XML components. Such names SHOULD be based on simple best practices.

### 7.5.3 Reflect the Real World

SISO develops, manages, maintains, and promulgates user-driven Modeling and Simulation (M&S) standards that improve the technical quality and cost efficiency of M&S implementations across the worldwide M&S community. To help facilitate unambiguous understanding of SISO reusable XML components, the names and structures should represent and model the informational aspects of objects and concepts that users are most familiar with. Types should not simply model collections of data.

[Principle 22]

Component definitions in SISO-conformant XML schemas SHOULD reflect real-world concepts.

### 7.5.4 Be Consistent

There should be no conflicts of meaning among types. This holds for types within a namespace, as well as types in different namespaces. A type should be used consistently in similar situations for similar purposes. Types should be defined for clear understanding and ease of intended use.

[Principle 23]

Component definitions in SISO-conformant XML schemas SHOULD have semantic consistency.

### 7.5.5 Reserve Inheritance for Specialization

Specialization should not be applied simply for the sake of achieving property inheritance. Specialization should be applied only where it is meaningful and appropriate to model permanent sibling subclasses of a base class that are mutually exclusive of one another.
[Principle 24]
Complex type definitions in SISO-conformant XML schemas SHOULD use type inheritance only for specialization.

Note that application of type augmentations is a well-defined exception to this guideline.

7.5.6 Do Not Duplicate Definitions
A real-world entity should be modeled in only one way. The definition of a type or element should appear once and only once. Multiple components of identical or closely similar semantics hinder interoperability because too many valid methods exist for representing the same data. For each data concept represented, there should be only one component (and associated type) to represent it.

Components with very similar semantics may exist in different contexts. For example, a complex type created for a particular exchange may appear to have identical or closely similar semantics to a complex type defined in another SISO XML schema, or XML schema subset of the SISO product being authored. However, the type defined at the exchange level will have much more precise business requirements and syntax, compared with the broad definitions that are heavily reused. Specific contextual definitions should be considered semantic changes. This includes the application of augmentations to create a specialized type for a specific use.

Two components may have the same definition while having different representations. For example, a string may hold the complete name of a person, or the name may be represented by a structure that separates the components of the name into first, last, etc. The definition of alternative representations should not be considered duplication.

[Principle 25]
Multiple components with identical or undifferentiated semantics SHOULD NOT be defined.
Component definitions SHOULD have clear, explicit distinctions.

7.5.7 Keep It Simple
All SISO XML content and structure is fundamentally based on business requirements for information exchange. To encourage adoption and use in practice, SISO should implement business requirements in simple, consistent, practical ways.

[Principle 26]
SISO-conformant XML schemas SHOULD have the simplest possible structure, content, and architecture consistent with real business requirements.

7.5.8 Be Aware of Scope
The scope of components defined in SISO-conformant XML schemas should be carefully considered. Some components represent simple data values, while others represent complex objects with many parts and relationships. Components should exist in layers. Components should exist as small, narrowly scoped, atomic entities that are used to consistently construct more broadly scoped, complex components (and so on).

[Principle 27]
Components defined by SISO-conformant XML schemas SHOULD be defined appropriate for their scope.

7.5.9 Be Mindful of Namespace Cohesion
Namespaces should maximize cohesion. The namespace methodology helps prevent name clashes among communities or domains that have different business perspectives and may choose identical data names to represent different data concepts. A namespace should be designed so that its components are consistent, may be used together, and may be updated at the same time.
[Principle 28]

XML namespaces defined by SISO-conformant XML schemas SHOULD encapsulate data components that are coherent, consistent, and internally related as a set. A namespace SHOULD encapsulate components that tend to change together.

8 RELATION TO STANDARDS

This section specifies the conformance standards and specifications for this SISO XML Style Guide. Where SISO XML Style Guidance differs from public standards, the rationale for those differences is discussed in this section. The complete list of standards and specifications referenced in this section appears in Appendix D: References.

8.1 XML 1.0

[Best Practice 8-1] (REF, SUB, EXT, CON)

SISO XML schema SHALL conform to XML as specified by [11].

Rationale

XML is a well-known, commonly used W3C Recommendation. It is supported by a large number of commercial and open-source software tools. It is a simple, well-defined, semi-structured data format that is flexible enough to allow for easy extension. XML works with many other powerful associated technologies such as XML Schema, XSLT, and XPath. XML artifacts of SISO conform to the most recent W3C recommendation for XML.

8.2 XML Namespaces

[Best Practice 8-2] (REF, SUB, EXT, CON)

The XML schema SHALL conform to the specification for namespaces in XML, as defined by [12] and [13].

Rationale

SISO is the organization dedicated to the promotion of modeling and simulation interoperability and reuse for the benefit of diverse M&S communities, including developers, procurers, and users worldwide. The ultimate scope of SISO XML standards is anticipated to be quite large. The primary purpose of namespaces is to avoid naming conflicts, which for SISO is quite common, since SISO stakeholders and IEPD developers define and name many of their own data components independently. Therefore, in SISO, XML namespaces are employed both to avoid name clashes and to provide a level of independence to participating domains.

8.3 XML Schema

[Best Practice 8-3] (REF, SUB, EXT, CON)


Rationale

XML Schema has become the generally accepted XML schema language and is experiencing the most widespread adoption. Although other XML schema languages exist that offer their own advantages and disadvantages, the current approach is to base SISO XML Schemas on the W3C XML Schema [3] specification.
8.4 Documented Components

Good data definitions are fundamental to data interoperability. You cannot effectively exchange what you cannot understand. SISO employs the guidance of [3] as a baseline for its data component definitions. All SISO XML Components are documented.

[Best Practice 8-4] (REF, EXT)

Within a SISO-conformant XML schema, the data definition provided for each documented component SHALL follow the requirements and recommendations for data definitions given by [3].

Rationale

To advance the goal of creating semantically rich SISO XML-conformant XML schemas, it is necessary that data definitions be descriptive, meaningful, and precise. [3] provides standard structure and best practices for defining data definitions. SISO uses this standard for component definitions.

Note that the metadata maintained for each SISO XML component contains additional details, including domain-specific usage examples and keywords. Such metadata is used to enhance search and discovery of components in a registry, and therefore, is not included in XML schemas.

For convenience and reference, the summary requirements and recommendations in [3] are reproduced here:

ISO 11179 Requirements

A data definition SHALL:

• Be stated in the singular.
• State what the concept is, not only what it is not.
• Be stated as a descriptive phrase or sentence(s).
• Contain only commonly understood abbreviations.
• Be expressed without embedding definitions of other data or underlying concepts.

ISO 11179 Recommendations

A data definition SHOULD:

• State the essential meaning of the concept.
• Be precise and unambiguous.
• Be concise.
• Be able to stand alone.
• Be expressed without embedding rationale, functional usage, or procedural information.
• Avoid circular reasoning.
• Use the same terminology and consistent logical structure for related definitions.
• Be appropriate for the type of metadata item being defined.

In addition to the requirements and recommendations of [3], SISO XML Guidance applies additional best practices to data definitions. These best practices are detailed in Section 10.2.1, Human-Readable (Natural Language) Documentation.

9 NORMATIVE - XML SCHEMA DESIGN BEST PRACTICES

The W3C XML Schema [3] Language provides many features that allow a developer to represent a logical data model many different ways. This section establishes best practices for the use of XML Schema constructs within SISO-conformant XML schemas. Because the XML Schema specifications are
flexible, comprehensive best practices are needed to achieve a balance between establishing uniform XML schema design and providing developers flexibility to solve novel data modeling problems.

Note that external XML schemas (non-SISO-conformant XML schemas) do not need to obey the best practices set forth in this section. So long as XML schema components from external XML schemas are adapted for use with SISO, according to the modeling best practices in Section 6.4, they may be used as they appear in the external standard, even if the XML schema components violate the best practices for SISO-conformant XML schemas.

The XML Schema design best practices in this section fall into the following categories:

- Restrictions on XML Schema Constructs
- xsd:schema Document Element
- Namespace Imports
- Annotations
- Type Definitions
- Additional Definitions and Declarations

9.1 Restrictions on XML Schema Constructs

A number of XML Schema constructs are not used within SISO-conformant XML schemas. Some of these constructs create problems for interoperability, with tool support, or with clarity or precision of data model definition.

9.1.1 No Mixed Content

[Best Practice 9-1] (REF, SUB, EXT)
Within the XML schema, an element xsd:complexType SHALL NOT own the attribute mixed with the value true.

[Best Practice 9-2] (REF, SUB, EXT)
Within the XML schema, an element declaration that is of complex content SHALL NOT own the attribute mixed with the value true.

Rationale
Mixed content allows the mixing of data tags with text. Languages such as XHTML use this syntax for markup of text. SISO-conformant XML schemas define XML that is for data exchange, not text markup. Mixed content creates complexity in processing, defining, and constraining content.

Well-defined markup languages exist outside SISO and may be used with SISO data. external XML schemas may include mixed content and may be used with SISO products. However, mixed content shall not be defined by SISO-conformant XML schemas in keeping with [Principle 9].

9.1.2 No Notations

[Best Practice 9-3] (REF, SUB, EXT)
The XML schema SHALL NOT contain a reference to the type definition xsd:N O T A T I O N or to a type derived from that type.

[Best Practice 9-4] (REF, SUB, EXT)
The XML schema SHALL NOT contain the element xsd:notation.

Rationale
XML Schema notations allow the attachment of system and public identifiers on fields of data. The notation mechanism does not play a part in validation of instances and is not supported by SISO.
9.1.3 No XML Schema Inclusion

[Best Practice 9-5] (REF, SUB, EXT)

The XML schema SHALL NOT contain the element `xsd:include`.

Rationale

Element `xsd:include` brings XML schemas defined in separate files into the current namespace. It breaks a namespace up into arbitrary partial XML schemas, which needlessly complicates the XML schema structure, making it harder to reuse and process, and also increases the likelihood of conflicting definitions. Inclusion of XML schemas that do not have namespaces also complicates XML schema understanding. This inclusion makes it difficult to find the realization of a specific XML schema artifact and create aliases for XML schema components that should be reused. Inclusion of XML schemas also violates [Principle 8], as it uses multiple XML schemas to construct a namespace.

9.1.4 No XML Schema Redefinition

[Best Practice 9-6] (REF, SUB, EXT)

The XML schema SHALL NOT contain the element `xsd:redefine`.

Rationale

The `xsd:redefine` element allows an XML Schema document to restrict and extend components from a namespace, in that very namespace. Such redefinition introduces duplication of definitions, allowing multiple definitions to exist for components from a single namespace. This violates [Principle 8] that a single reference XML schema defines a SISO-conformant namespace.

9.1.5 Wildcard Restrictions

There are many constructs within XML Schema that act as wildcards. That is, they introduce buckets that may carry arbitrary or otherwise non-validated content. Such constructs violate [Principle 11], and as such provide implicit workarounds for the difficult task of agreeing on the content of data models. Such workarounds should be made explicitly, outside the core data model.

9.1.5.1 No Unconstrained Type Substitution

[Best Practice 9-7] (REF, SUB, EXT)

The XML schema SHALL NOT reference the type `xsd:anyType`.

Rationale

XML Schema has the concept of the "ur-type," a type that is the root of all other types. This type is realized in XML schemas as `xsd:anyType`. SISO-conformant XML schemas shall not use `xsd:anyType`, because this feature permits the introduction of arbitrary content (i.e., untyped and unconstrained data) into an XML instance. SISO intends that the XML schemas describing that instance describe all constructs within the instance.

9.1.5.2 No Unconstrained Text Substitution

[Best Practice 9-8] (REF, SUB, EXT)

The XML schema SHALL NOT reference the type `xsd:anySimpleType`.

Rationale

XML Schema provides a restriction of the "ur-type," which contains only simple content. This provides a wildcard for arbitrary text. It is realized in XML Schema as `xsd:anySimpleType`. SISO-conformant XML schemas shall not use `xsd:anySimpleType` because this feature is insufficiently constrained to provide a meaningful starting point for content definitions. Instead, content should be based on one of the more specifically defined simple types defined by XML Schema.
9.1.5.3 Untyped Elements Shall Be Abstract

[Best Practice 9-9] (REF, SUB, EXT)

Within the XML schema, an element declaration with the attribute name and without the attribute type SHALL carry the attribute abstract with the value true.

Rationale

Untyped element declarations act as wildcards that may carry arbitrary data. By declaring such types abstract, SISO XML Guidance allows the creation of type independent semantics without allowing arbitrary content to appear in XML instances.

9.1.5.4 No Untyped Attributes

[Best Practice 9-10] (REF, SUB, EXT)

Within the XML schema, an attribute declaration with attribute name SHALL carry the attribute type.

Rationale

Untyped XML Schema attributes allow arbitrary content, with no semantics. Attributes shall have a type so that specific syntax and semantics will be provided.

9.1.5.5 No Unconstrained Element Substitution

[Best Practice 9-11] (REF, SUB)

The XML schema SHALL NOT contain the element xsd:any.

Rationale

The xsd:any particle (see Model Group Restrictions for an informative definition of particle) provides a wildcard that may carry arbitrary content. The particle xsd:any may appear within constraint XML schemas, extension XML schemas, and exchange XML schemas.

9.1.5.6 No Unconstrained Attribute Substitution

[Best Practice 9-12] (REF, SUB, EXT)

The XML schema SHALL NOT contain the element xsd:anyAttribute.

Rationale

The xsd:anyAttribute element provides a wildcard, where arbitrary attributes may appear. The element xsd:anyAttribute may appear within constraint XML schemas or within other XML schemas that are not SISO-conformant, but it is prohibited in SISO-conformant XML schemas.

9.1.6 Component Naming Restrictions

All SSO components shall be named. That is, type definitions, and element and attribute declarations shall be given explicit names — local and anonymous component definition is not allowed. Note that XML Schema enforces the placement of attribute group and model group definitions as top-level components, which forces the components to be named.

9.1.6.1 No Anonymous Type Definitions

[Best Practice 9-13] (REF, SUB, EXT)

Within the XML schema, any occurrence of the element xsd:complexType or xsd:simpleType SHALL appear as an immediate child of the element xsd:schema.

Rationale

SISO does not support anonymous types in SISO-conformant XML schemas. All XML Schema "top-level" types (children of the document element) are required by XML Schema to be named. By
requiring SISO type definitions to be top level, they are forced to be named and are therefore globally reusable.

9.1.6.2 No Local Element Declarations

[Best Practice 9-14] (REF, SUB, EXT)

Within the XML schema, any element declaration carrying the attribute name SHALL appear as an immediate child of the document element xsd:schema.

Rationale

All XML schema components defined by SISO-conformant XML schemas shall be named, accessible from outside the defining XML schema, and reusable across XML schemas. Local element definitions provide named elements that are not reusable outside the element’s defined context. Requiring named SISO XML elements to be top level ensures that they are globally reusable.

9.1.6.3 No Local Attribute Definitions

[Best Practice 9-15] (REF, SUB, EXT)

Within the XML schema, any attribute declaration owning the attribute name SHALL appear as an immediate child of the document element xsd:schema.

Rationale

All XML schema components defined by SISO-conformant XML schemas are named, accessible from outside the defining XML schema, and reusable across XML schemas. Local attribute definitions provide named attributes that are not reusable outside the attribute’s defined context. Requiring named SISO XML attributes to be top level ensures that they are globally reusable.

9.1.7 No Uniqueness Constraints

[Best Practice 9-16] (REF, EXT)

The XML schema SHALL NOT contain any of the elements xsd:unique, xsd:key, xsd:keyref, xsd:selector, or xsd:field.

Rationale

XML Schema provides SISO with the ability to apply uniqueness constraints to XML schema-validated content. These mechanisms, however, establish relationships in a way that is very difficult to understand, extend, and keep consistent through XML schema reuse. These elements may be used in subset XML schemas and constraint XML schemas.

9.1.8 Model Group Restrictions

Complex content definitions in XML Schema use model group XML schema components. These XML schema components, xsd:all, xsd:choice and xsd:sequence, also called compositors, provide for ordering and selection of particles within a model group.

XML Schema defines a particle as an occurrence of xsd:element, xsd:sequence, xsd:choice, xsd:any (wildcard) and xsd:group (model group) within a content model. For example, an xsd:sequence within an XML Schema complex type is a particle. An xsd:element occurring within an xsd:sequence is also a particle.

9.1.8.1 No Recursively Defined Model Groups

[Best Practice 9-17] (REF, SUB)

Within the XML schema, any immediate child of a model group xsd:sequence element SHALL be one of xsd:annotation, xsd:element, xsd:choice, xsd:sequence.
Within the XML schema, any immediate child of a model group `xsd:sequence` element SHALL be one of `xsd:annotation`, `xsd:element`, `xsd:choice`, `xsd:sequence` or `xsd:any`.

Note: This best practice precludes the use of the particle `xsd:group` as an immediate child of a model group for extension schemas.

Within the XML schema, any immediate child of a model group `xsd:choice` element SHALL be one of `xsd:annotation`, `xsd:sequence` or `xsd:element`.

Note: This best practice precludes the use of the particle `xsd:group`, `xsd:any` as an immediate children of a model group.

**Rationale**

XML Schema provides the capability for model groups to be recursively defined. This means that a sequence may contain a sequence, and a choice may contain a choice. These best practices are designed to keep content models simple, comprehensive, and reusable: The content of an element should boil down to a simple list of elements, defined in as straightforward a manner as is possible to meet requirements.

**9.1.8.2 XML Schema Particle Cardinality Restrictions**

Within the XML schema, if the element `xsd:sequence` carries the attribute `minOccurs`, it SHALL set the value for the attribute to 1.

Within the XML schema, if the element `xsd:sequence` carries the attribute `maxOccurs`, it SHALL set the value of the attribute to 1.

**Rationale**

XML Schema allows each particle to specify cardinality (how many times the particle may appear in an instance). SISO XML Guidance restricts the cardinality of `xsd:sequence` particles to exactly one, to ensure that content model definitions are defined in as straightforward a manner as possible.

**Discussion**

Note that the particle `xsd:any` is not allowed in reference XML schemas or subset XML schemas by [Best practice 9-11].

Note also that element declarations acting as a particle (particles formed by `xsd:element`) may have any cardinality; they are not restricted by this best practice. Should a user desire the behavior that would be obtained from the use of special cardinalities on these particles, he or she should define them within explicitly named elements.

**9.1.8.3 Default Value Restrictions**

XML Schema provides the capability for element and attribute declarations to provide default values when XML instances using those components do not provide values.

Within the schema, any element `xsd:element` SHALL NOT carry the attribute default.

Within the schema, any element `xsd:attribute` SHALL NOT carry the attribute default.
Rationale

The use of default values means that the act of validating a schema will insert a value into an XML instance where none existed prior to schema validation. Schema validation is for rejection of invalid instances, not for modifying instance content, as specified in [Principle 3].

9.1.9 Block Substitution Restrictions

XML Schema provides a mechanism that will prevent substitution for an element declaration or type definition. That is, an element declaration may declare one or more of the following:

1. An instance of this element declaration may not substitute an extended type.
2. An instance of this element declaration may not substitute a restricted type.
3. An instance of this element declaration may not substitute another element.

These restriction mechanisms are very useful in instances; they allow restriction of content models down to exact types and elements. However, in shared data models, they limit reuse and customization options, in opposition to [Principle 14].

[Best Practice 9-24] (REF, EXT)

Within the XML schema, an element declaration SHALL NOT carry the attribute block.

[Best Practice 9-25] (REF, EXT)

Within the XML schema, a complex type definition SHALL NOT carry the attribute block.

[Best Practice 9-26] (REF, SUB, EXT)

Within the XML schema, the document element xsd:schema SHALL NOT carry the attribute blockDefault.

Rationale

Restriction of substitution options reduces capacity for reuse; thus, it is forbidden within SISO-conformant XML schemas. In particular, setting the block value at the XML schema level complicates understanding of component definitions.

9.1.10 Final Value Restrictions

XML Schema provides the capability for type definitions and elements to declare a final value. This value prevents the creation of derived components. In shared data models, this capability limits reuse and customization options, in opposition to [Principle 14].

[Best Practice 9-27] (REF, SUB)

Within the XML schema, a simple type definition SHALL NOT carry the attribute final.

[Best Practice 9-28] (REF, SUB)

Within the XML schema, a complex type definition SHALL NOT carry the attribute final.

[Best Practice 9-29] (REF, SUB)

Within the XML schema, an element declaration SHALL NOT carry the attribute final.

[Best Practice 9-30] (REF, SUB, EXT)

Within the XML schema, the document element xsd:schema SHALL NOT carry the attribute finalDefault.

Rationale

Restriction of derivation options reduces capacity for reuse and so is forbidden within reference and subset XML schemas. As well, the use of finalDefault complicates understanding of XML schemas, so it is only allowed in constraint XML schemas.
9.2 XML Schema xsd:schema Document Element

The features of XML Schema allow for flexibility of use for many different and varied types of implementation. SISO XML Guidance requires consistent use of these features.

[Best Practice 9-31] (REF, SUB, EXT, CON)
Within the XML schema, the document element xsd:schema SHALL carry the attribute targetNamespace.

[Best Practice 9-32] (REF, SUB, EXT, CON)
Within the XML schema, the value of the required attribute targetNamespace on the document element xsd:schema SHALL match the production <absolute-URI> as defined by [8].

Rationale
XML Schemas without defined namespaces provide definitions that are ambiguous, in that they are not universally identifiable.

Absolute URIs are the only universally meaningful URIs. URIs include both URLs and URNs. Finding the target namespace using standard XML Base technology is complicated and not specified by XML Schema. Relative URIs are not universally identifiable, as they are context-specific.

Discussion
The document element xsd:schema may contain optional attributes attributeFormDefault and elementFormDefault. The values of these attributes are immaterial to a SISO-conformant XML schema, as each attribute defined by a SISO-conformant XML schema shall be defined at the top level and so shall be qualified with the target namespace of its declaration.

[Best Practice 9-33] (REF, SUB, EXT, CON)
Within the XML schema, the document element xsd:schema SHALL carry the attribute version of the SISO Product the XML schema applies to.

[Best Practice 9-34] (REF, SUB, EXT, CON)
Within the XML schema, the value of the required attribute version on the document element xsd:schema SHALL follow the SISO product numbering schema as specified in SISO-ADM-001[1].

Rationale
It is very useful to be able to tell one version of a XML schema from another. Apart from the use of namespaces for versioning, it is sometimes necessary to release multiple versions of XML schema documents. Such use might include:

• subset XML schemas and constraint XML schemas
• Error corrections or bug fixes
• Documentation changes
• Contact information updates

In such cases, a different value for the version attribute implies a different version of the XML schema. No specific meaning is assigned to specific version identifiers.

9.3 Namespace Imports

XML Schema requires that namespaces used in external references be imported using the xsd:import element. The xsd:import element appears as an immediate child of the xsd:schema element. A XML schema shall import any namespace which

1. Is not the local namespace, and
2. Is referenced from the XML schema.  

The behavior of import statements is not necessarily intuitive. In short, the import introduces a namespace into the XML schema containing the import; it has no transitive effect. If the namespaces of an import statement are not referenced from the XML schema, then the import statement has no effect. The import statement cannot be used to direct XML schema locations for XML schemas not referenced from the XML schema performing the import. The XML schema location directed by the import element may be overridden by user directive at the parser, or by being overridden by import elements from other XML schemas.

Imports of namespaces should be made as uniform as possible; all XML schemas in a XML schema set should agree on what XML schema location goes with a particular namespace. Otherwise, behavior may be dependent on the behavior of the parser and the order of components in instance documents.

9.3.1 xsd:import Element Restrictions

[Best Practice 9-35] (REF, SUB, EXT)  
Within the XML schema, the element xsd:import SHALL carry the attribute namespace.

[Best Practice 9-36] (REF, SUB, EXT)  
Within the XML schema, the value of the required attribute namespace owned by the element xsd:import SHALL match the production <absolute-URI> as defined by [8].

Rationale  
An import that does not specify a namespace is enabling reference to non-namespaced components. SISO requires that all components have a defined namespace. It is important that the namespace declared by a XML schema be universally defined and unambiguous. Use of the standard XML Base for processing is not specified by XML Schema; thus it is not supported here.

[Best Practice 9-37] (REF, SUB, EXT)  
Within the XML schema, the element xsd:import SHALL carry the attribute XML schemaLocation.

Rationale  
An import that does not specify a XML schema location gives no clue to processing applications as to where to find an implementation of the namespace. Even though such a provided XML schema location may be overridden, it is important that an initial default be provided for processing.

[Best Practice 9-38] (REF, SUB, EXT)  
Within the XML schema, the value of the required attribute XML schemaLocation carried by the element xsd:import SHALL match either the production <absolute-URI> or the definition of "relative-path reference," as defined by [8].

Rationale  
The default value may be specified either as absolute or relative URIs. Since URNs are not resolvable, they are inappropriate for use in XML schemaLocation. The requirement for conformance to "relative-path reference" is required to avoid the more obscure syntax of "network-path reference" and the system-specific "absolute-path reference."

[Best Practice 9-39] (REF, SUB, EXT)  
Within the XML schema, the value of the required attribute XML schemaLocation carried by the element xsd:import SHALL be resolvable to a XML schema document file that is valid according to [15] and [14].
Rationale
The XML Schema specification requires that the object imported via `xsd:import` shall be a XML schema document. This best practice reinforces that requirement.

Discussion
Note that relative URI references are dereferenced from the location of the XML schema document performing the import, not from the location of an instance or other XML schema.

9.3.2 Including XML Content from Other Namespaces
Within an XML Schema document, there are several mechanisms to include XML content that is not from the XML or XML Schema namespaces. Those mechanisms are:

1. Carrying attributes from other than the XML or XML Schema namespaces on an element in the XML Schema namespace.
   By the best practices of XML Schema, any element may have attributes that are from other namespaces. These attributes do not participate in validation but may carry information useful to tools that process XML schemas.

2. Adding content to the elements `xsd:appinfo` and `xsd:documentation`.
   XML Schema allows arbitrary XML content to be included within annotations. Such XML does not participate in validation but may communicate useful information to XML schema readers or processors.

SISO XML Guidance requires all such XML content to be "schema-valid." That is, it shall have a XML schema, and it shall validate against that XML schema. The XML schemas shall be introduced via `xsd:import` elements within the XML schema that uses the imported content. This is for two reasons:

1. Some tools require imports of namespaces used within XML schemas and validate against those XML schemas.
2. The definition and the validity of content within XML schemas should be clear.

[Best Practice 9-40] (REF, SUB, EXT)
Within the XML schema, when a namespace other than the XML namespace or the XML Schema namespace is used, it SHALL be imported into the XML schema using the `xsd:import` element.

Rationale
This best practice ensures that used namespaces have recognizable defining sources and that they will cooperate with existing tools.

[Best Practice 9-41] (REF, SUB, EXT)
Within the XML schema, when a namespace other than the XML namespace or the XML Schema namespace is used, its content SHALL be valid with respect to the XML schema imported for that namespace.

Rationale
XML Schema does not address the XML schema-validity of content used for annotations or attributes on XML schema components. This best practice ensures that content used in such a manner is XML schema-valid. This encourages interoperable data definitions and XML schema documents.

9.4 Annotations
Annotations document XML components for humans, in natural language, and machines in structured
data. Annotations are not a substitute for natural language specifications that define how a SISO XML schema is used to further interoperability and reuse.

[Best Practice 9-42] (REF, EXT)

Within the XML schema, an element SHALL have at most one instance of an element `xsd:annotation` as an immediate child.

Rationale

XML Schema allows annotations to be added to components in a fairly loose manner: there may be multiple annotations, that each have multiple `documentation` or `appinfo` elements. This flexibility in the syntax provides no additional expressivity but does complicate processing, so it is forbidden in SISO XML Schemas.

9.4.1 Human-Readable (Natural Language) Documentation

XML Schema describes the content of `xsd:documentation` elements as "user information." This information is targeted for reading by humans in natural language (English). The XML Schema specification does not say what form human-targeted information should take. Within SISO, user information is plain text with no formatting or XML structure.

[Best Practice 9-43] (REF, EXT)

Within the XML schema, the content of the `xsd:documentation` element that constitutes the data definition of a component SHALL be character information items as specified by [11].

Rationale

According to the XML Schema specification, the content of `xsd:documentation` elements is intended for human consumption, whereas other structured XML content is intended for machine consumption. Therefore, the `xsd:documentation` element SHALL NOT contain structured XML data. As such, any XML content appearing within a documentation element is in the context of human-targeted examples and should be escaped using `&lt;` and `&gt;`.

XML comments are not XML Schema constructs and are not specifically associated with any XML schema-based components. As such, comments are not considered semantically meaningful by SISO and may not be retained through processing of SISO XML schemas.

[Best Practice 9-44] (REF, SUB, EXT)

XML comments SHALL not be used for persistent information about constructs within the XML schema.

Rationale

Since XML comments are not associated with any specific XML Schema construct, there is no standard way to interpret comments. As such, comments should be reserved for internal use, and XML Schema annotations should be preferred for meaningful information about components. SISO specifically defines how information should be encapsulated in SISO-conformant XML schemas via `xsd:annotation` elements.

9.4.2 Machine-Readable Annotations

XML Schema provides special annotations for support of automatic processing. The XML Schema specification provides the element `xsd:appinfo` to carry such content and does not specify what style of content they should carry. In SISO, `xsd:appinfo` elements carry structured XML content.

[Best Practice 9-45] (REF, EXT)

Within the XML schema, any immediate child of an `xsd:appinfo` element SHALL be an element information item or a comment information item.
Rationale

Application information elements are intended for automatic processing; thus they should contain machine-oriented XML data.

[Best Practice 9-46] (REF, EXT)

Within the XML schema, any element that is an immediate child of an xsd:appinfo element SHALL be in a namespace.

Rationale

Use of default namespace is allowed, but content has to have a real namespace, and namespaces shall be declared. The XML namespaces specification includes the concept of content not in a namespace. Non-namespaced data runs counter to the principle of distinctly identifiable data definitions.

[Best Practice 9-47] (REF, EXT)

Within the XML schema, an element in the XML Schema namespace SHALL NOT occur as a descendant of any element xsd:appinfo.

Rationale

SISO-conformant XML schemas are designed to be very easily processed. Although uses of XML Schema elements as content of xsd:appinfo elements could be contrived, it is not current practice and could seriously complicate the authoring of XML schema validators and processors, such as XSLT [16], which may evaluate XML elements by their namespaces and names. Forbidding the use of XML Schema elements outside valid uses of XML schema will simplify such processing.

9.5 Comments

SISO reserves comments for use in documenting Meta information about the XML schema as a SISO product. Comments are used for including information such as the appropriate SISO copyright that applies to the product (reference SISO-ADM-005)[2]. Other uses include identifying the change request that resulted in a change to the XML schema product.

[Best Practice 9-48] (REF, EXT)

Comments SHALL NOT be used to document the normative or descriptive aspects for any component of SISO XML Schemas.

9.6 Type Definitions

XML Schema provides a variety of ways to define new types. This section covers the SISO XML Guidance restrictions on defining complex types, with both simple and complex content.

9.6.1 Complex Type Definitions

XML Schema provides a large amount of flexibility in the creation of complex types. SISO XML guidance narrows the XML schema capability to a smaller set of constructs.

Note that best practices on prohibited constructs (Section 9.1.6.1: No Anonymous Type Definitions, above) forbid defining complex types as local types. All complex type definitions shall be top-level, named components.

XML Schema makes a distinction between complex types with simple content versus complex types with complex content. Complex types with simple content (CSCs) have content that is not allowed to contain XML elements. Complex types with complex content (CCCs) have content that does contain XML elements. Since mixed content is prohibited in SISO XML Schemas by [Best practice 8-1], all SISO-conformant complex types are either CSCs or CCCs.
9.6.2 Simple Content (CSC) Restrictions

Within a SISO-conformant XML schema, a complex type with simple content (CSC) can be created in one of two ways:

1. By extension of an existing CSC.
2. By extension of an existing simple type.

Both of these methods use the element xsd:extension.

[Best Practice 9-49] (REF)

Within the XML schema, the element xsd:simpleContent SHALL have as an immediate child the element xsd:extension.

Rationale

This best practice ensures that the definition of a CSC will use the XML Schema extension facility. This allows for the above cases while disallowing much more complicated syntactic options available in XML Schema.

Note that the applicability of the above best practice allows for use of xsd:restriction within xsd:simpleContent in subset XML schemas, extension XML schemas, and exchange XML schemas.

Although the two above methods have similar syntax, there are subtle differences. SISO’s conformance best practices ensure that any complex type has the necessary attributes for representing IDs, metadata, and link metadata. So case 1 does not require adding these attributes, as they are guaranteed to occur in the base type.

However, in case 2, where a new complex type is created from a simple type, the attributes for complex types shall be added.

[Best Practice 9-50] (REF, SUB, EXT)

Within the schema, given an element xsd:simpleContent with a child xsd:extension owning an attribute base, if the attribute base has a value that resolves to the name of a simple type, then the element xsd:extension SHALL have an immediate child element xsd:attributeGroup.

Rationale

This best practice ensures that a CSC that is created as an immediate extension of a simple type adds the attributes required for a complex type.

9.6.3 XML schema Complex Content (CCC) Restrictions

Within a reference XML schema, a complex type with complex content (CCC) can be created in one of two ways:

1. By extension of an existing complex type (CCC or CSC).
2. By extension of the type structure:ComplexObjectType.

Both of these methods use the element xsd:extension. Within extension XML schemas, exchange XML schemas, and subset XML schemas, the use of xsd:restriction to create complex types with complex content is also allowed.

[Best Practice 9-51] (REF)

Within the XML schema, the element xsd:complexContent SHALL have as an immediate child the element xsd:extension.

Rationale

SISO does not support, as conformant, the use of complex type restrictions in Reference schemas.

SISO defines a language, in which specific content is allowed. It does not specify messages that
forbid content. Such restrictions may be performed in non-conformant XML schemas or within
constraint XML schemas or other artifacts of constraint.

Note that XML Schema requires use of the attribute base on xsd:extension.

Note also that the applicability allows for the use of restriction in subset XML schemas, extension XML
schemas, exchange XML schemas, and constraint XML schemas.

The xsd:extension element says that the type under definition is an extension of another type. That
type shall be limited to those used with SISO XML Schemas.

[Best Practice 9-52] (EXT)

Within the XML schema, given an element xsd:complexContent with a child xsd:restriction
owning an attribute base, the attribute base SHALL have a value that resolves to the name of a
complex type that is a SISO-conformant component.

[Rationale]

This ensures that a CCC defined through restriction has well-defined semantics.

9.7 Additional Definitions and Declarations

XML Schema provides a variety of ways to declare and define elements and attributes.

9.7.1 Element Declarations

Within SISO-conformant XML schemas, elements may be declared as abstract. Element declarations
shall be at the top level, as best practices in other sections prohibit the use of local elements. Elements
may be defined without a type, but any element declaration that has no type shall be declared abstract by
[Best practice 9-13], which forbids anonymous type definitions.

Within an element declaration, the attributes fixed, nillable, and substitutionGroup may be
used as per the XML Schema specification. The attribute form is irrelevant to SISO, as SISO-
conformant XML schemas may not contain local element definitions by [Best practice 9-14].

Element uses (element declarations acting as particles) shall reference top-level named elements. In an
element use, SISO allows any values for the XML Schema properties “max occurs” and “min occurs.”

Based on a variety of user requirements, all elements in the SISO XML schemas are defined to allow a nil
value. For example, the following XML instances are permitted in SISO-conformant instances:

<nc:ActivityDate></nc:ActivityDate>

OR

<nc:ActivityDate/>

Nil value allowance or restriction is only significant to elements of nontextual types (e.g., dates and
numeric values) and elements of text types that have restricted value space (e.g., code). This is because
an unrestricted text typed element always contains the empty string (“”) in its value space. However, for
numeric values and restricted text type elements, SISO allows users to tighten constraints as required in
IEPDs by resetting nillable="false".

9.7.2 Attribute Declarations

Attribute declarations shall be declared with a type by [Best practice 9-10], which forbids anonymous type
definitions for attributes.

Within an attribute declaration, the attribute fixed may be used as per the XML Schema specification.
Within an attribute declaration, the attribute form is irrelevant to SISO, as SISO-conformant XML
schemas may not contain local attribute declarations.

Attribute uses (attribute declarations acting as particles) shall be uses of top-level named attributes.

SISO-conformant XML schemas may not define local named attributes within type definitions. Within an
attribute use, the attributes fixed and use may be used as per the SISO XML Schema NDBP specification.

10 NORMATIVE - XML SCHEMA MODELING BEST PRACTICES

SISO XML Guidance provides a framework for modeling concepts and relationships as XML artifacts. The data model is implemented via XML Schema. However, XML Schema does not provide sufficient structure and constraint to enable translating from a conceptual model to a XML schema and then to instances of the concepts. SISO provides additional guidance for modeling concepts.

10.1 xsd:schema Document Element Restrictions

[Best Practice 10-1] (REF, SUB, EXT, CON)
Two XML Schema documents SHALL have the same value for attribute targetNamespace carried by the element xsd:schema, if and only if they represent the same set of components.

[Best Practice 10-2] (REF, SUB, EXT, CON)
Two XML Schema documents SHALL have the same value for attribute targetNamespace carried by the element xsd:schema, and different values for attribute version carried by the element xsd:schema if and only if they are different views of the same set of components.

Rationale
These best practices embody the basic philosophy behind SISO’s use of namespaced components: A component is uniquely identified by its class (e.g. element, attribute, type), its namespace (a URI), and its local name (an unqualified string). Any two matching component identifiers refer to the same component, even if the versions of the XML schemas containing each are different.

10.2 Annotations

SISO-conformant XML schemas define data models for the purpose of information exchange. A major part of defining data models is the proper definition of the contents of the model. What does a component mean, and what might it contain? How should it be used? SISO-conformant XML schemas contain the invariant part of the definitions for the data model. The set of definitions includes:

1. A text definition of each component. This describes what the component means. The term used in this specification for such a text definition is data definition.
2. The structural definition of each component. This is made up of XML Schema component definitions, along with certain application information (appinfo). When possible, meaning is expressed via XML Schema mechanisms: type derivation, element substitution, specific types and structures, as well as names that are trivially parseable. Beyond that, SISO XML-specific syntax shall be used, as discussed in this section.

10.2.1 Human-Readable (Natural Language) Documentation

By other best practices, a XML schema component shall contain at most one element xsd:annotation. An element xsd:annotation, in turn, contains at most elements xsd:documentation and xsd:appinfo. The content of the first element xsd:documentation on a component is the data definition for the component.

[Best Practice 10-3] (REF, EXT)
Within the XML schema, any element xsd:complexType SHALL be a documented component.

[Best Practice 10-4] (REF, EXT)
Within the XML schema, any element xsd:simpleType SHALL be a documented component.
[Best Practice 10-5] (REF, EXT)
Within the XML schema, any element `xsd:element` that is an immediate child of an element `xsd:schema` SHALL be a documented component.

[Best Practice 10-6] (REF, EXT)
Within the XML schema, any element `xsd:attribute` that is an immediate child of an element `xsd:schema` SHALL be a documented component.

[Best Practice 10-7] (REF, EXT)
Within the XML schema, any element `xsd:enumeration` SHALL be a documented component.

[Best Practice 10-8] (REF, EXT)
Within the XML schema, the document element `xsd:schema` SHALL be a documented component.


Rationale
Data definitions should be concise, precise, and unambiguous without embedding additional definitions of data elements that have already been defined once elsewhere (such as object classes).

W3C XML Schema [3] states that definitions should not be nested inside other definitions.
Furthermore, a data dictionary is not a language dictionary. It is acceptable to reuse terms (object class, property term, and qualifier terms) from a component name within its corresponding definition to enhance clarity, as long as the requirements and recommendations of [3] are not violated. This further enhances brevity and precision.

10.3 Simple Type Definitions
SISO places very few restrictions on the definition of simple types in conformant XML schemas. The use of lists should be reserved for cases where the data is fairly uniform.

[Best Practice 10-10] (REF, SUB, EXT)
Within the XML schema, a simple type definition that uses `xsd:list` SHOULD NOT be defined if any member of the list requires a property or metadata that is different than other members of the list.
All members of the list SHOULD have the same metadata, and should be related via the same properties.

Rationale
The members of a list are not individually addressable by SISO metadata techniques. The members are also not individually addressable by properties; a property has a value of all the members of the list. SISO XML Guidance provides no method for individually addressing a member of a list. If an individual member of a list needs to be marked up in a manner different than other members of the list, the use of individual elements may be preferred to the definition of a list simple type.

10.4 Component Usage

[Best Practice 10-11] (REF, SUB, EXT)
Any type definition referenced by a component within the XML schema SHALL be from one of the following:
1. The XML schema being defined.
2. A namespace imported as SISO-conformant.
3. The XML Schema namespace.
### Rationale

SISO-conformant XML schemas are based on other SISO-conformant XML schemas and the supporting namespaces. This simplifies processing and understanding of data.

#### [Best Practice 10-12] (REF, SUB, EXT)

Any element declaration referenced by a component within the XML schema SHALL be from one of the following:

1. The XML schema being defined.
2. A namespace imported as SISO-conformant.
3. An external namespace, in accordance with the best practices for external XML schemas as specified by this specification.

#### [Best Practice 10-13] (REF, SUB, EXT)

Any attribute declaration referenced by a component within the XML schema SHALL be from one of the following:

1. The XML schema being defined.
2. A namespace imported as SISO-conformant.
3. The XML namespace.
4. An external namespace, in accordance with the best practices for external XML schemas as specified by this specification.

### Rationale

SISO-conformant XML schemas are based on other SISO-conformant XML schemas. All attributes and elements shall be from SISO-conformant XML schemas, the XML namespace, or an external namespace. This applies to elements referenced for substitution groups as well. It does not apply to content of the XML schema (e.g., within annotations) or to the XML Schema declarations themselves. It applies only to attributes and elements referenced by the XML Schema components.

#### 10.4.1 Reference Elements

In XML instances, relationships between data objects are expressed as XML elements:

1. Data objects are expressed as XML elements.
2. XML elements contain attributes and other elements.

In this way, there is generally some implicit relationship between the outer element (the "containing" element, also known as the parent element) and the inner elements (the contained elements, also known as the child elements). Such expression of relationships is said to be by containment.

Expression of all relationships via element containment is not always possible. Situations that cause problems include:

- Circular relationships. For example, suppose that object 1 has a relationship to object 2 and object 2 has a relationship to object 1. Expressed via containment, this relationship would result in infinite recursive descent.
- Repeated relationships. For example, suppose object 1 has a relationship to object 2 and object 3 has a relationship to object 2. Expressed via containment, this would result in a duplicate of object 2.

A method that solves this problem is the use of references. In a C or assembler, a pointer would be used. In C++, a reference might be used. In Java, a reference value might be used. The method defined by the XML standard is the use of ID and IDREF. An IDREF refers to an ID. SISO uses this method and assigns it specific semantics.
1. **Best Practice 10-14** (REF, SUB, EXT)

Within the XML schema, a reference element and only a reference element SHALL be defined to be of type IDREF.

**Rationale**

reference elements shall be of the IDREF type, and elements of the IDREF type shall be reference elements. This best practice ensures that users always create reference elements using IDREF and cannot use IDREF for any other purpose.

2. **Best Practice 10-15** (REF, SUB, EXT)

Within the XML schema, any two elements of the form NCName and NCNameReference where the string value of NCName is the same in both forms, SHALL be defined to have identical semantics.

SISO XML Guidance recognizes no difference in meaning between a reference element and an element that is not a reference element.

**Rationale**

SISO-conformant data instances may use concrete data elements and reference elements as needed, to represent the meaning of the fundamental data. There is no difference in meaning between reference and concrete data representations. The two different methods are available for ease of representation. No difference in meaning should be implied by the use of one method or the other.

Assertions that indicate "included" data is intrinsic, while referenced data is extrinsic, are not valid and are not applicable to SISO-conformant data instances and data definitions.

SISO XML schema requires the use of an attribute of type IDREF XML Schema to define reference elements. According to the best practices of XML, such an attribute shall contain a value that is represented by an attribute of type ID. In SISO-conformant instance, the targets of IDREFs are expected to be values of the attribute ID.

3. **10.5 XML schema Using External XML Schemas**

There are a variety of commonly used standards that are represented in XML Schema. Such XML schemas are generally not SISO-conformant. SISO-conformant XML schemas may reference components defined by these external XML schemas. SISO-conformant components may be constructed from XML schema components that are not SISO-conformant.

XML schema SISO-conformant XML schemas may work with external XML schemas by creating external adapter types.

A single method is used to integrate external components into SISO-conformant XML schemas: SISO-conformant XML types are constructed from the external components.
Components defined by external XML schemas are called **external components**. A SISO-conformant XML type may use external components in a specific way: to construct a SISO-conformant XML type from external components. The goal in this method is to preserve as a single unit a set of data that embodies a single **concept** from an external standard.

For example, a SISO-conformant XML type may be created to represent a bibliographic reference from an external standard. Such an object may be composed of multiple elements and types from the external standard. These pieces are put together to form a single NIEM-conformant XML type. For example, an element representing an author, a book, and a publisher may be included in a single bibliographic entry. A SISO-conformant XML type built from these components may be used as any other SISO-conformant XML type. That is, elements may be constructed from such a type, and those elements are fully SISO-conformant.

To construct such a component, a SISO-conformant XML schema shall first import an external XML schema.

**Rationale**

A SISO-conformant XML schema has well-known documentation points. Therefore, a XML schema that imports a SISO-conformant namespace need not provide additional documentation. However, when an external XML schema is imported, appropriate documentation shall be provided at the point of import because documentation associated with external XML schemas is undefined and variable. In this particular case, documentation of external XML schemas is required at their point of use in SISO XML schemas.

**Best Practice 10-16** (REF, EXT)

Within the XML schema, an element `xsd:import` that imports a namespace defined by an external XML schema SHALL be a documented component.

**Rationale**

An adapter type should contain the information from an external standard to express a complete concept. This expression should be composed of content entirely from an external XML schema. Most likely, the external XML schema will be based on an external standard with its own legacy support.
In the case of an external expression that is in the form of model groups, attribute groups, or types, additional elements and type components may be created in an external XML schema, and the adapter type may use those components.

[Best Practice 10-18] (REF, EXT)
Within the XML schema, an element reference used in an adapter type definition SHALL be a documented component.

[Best Practice 10-19] (REF, EXT)
Within the XML schema, an attribute reference used in an adapter type definition SHALL be a documented component.

Rationale
In normal (conformant) type definition, a reference to an attribute or element is a reference to a documented component. Within an adapter type, the references to the attributes and elements being adapted are references to undocumented components. These components shall be documented to provide comprehensibility and interoperability. Since documentation made available by nonconformant XML schemas is undefined and variable, documentation of these components is required at their point of use, within the conformant XML schema.

[Best Practice 10-20] (REF, SUB, EXT)
Within the XML schema, an adapter type SHALL NOT be extended or restricted.

Rationale
Adapter types are meant to stand alone; each type expresses a single concept from an external XML schema, and adapter types are maintained in separate XML schemas that only contain adapter types. In this way, processors may easily switch modes, processing SISO-conformant content in one way, and external content in another. SISO Subset XML schemas

Subset XML schemas are XML schemas that are based on other SISO-conformant XML schemas but have been modified for any of several reasons. A subset XML schema may be created that limits what is considered valid data to a subset of what is valid against the base XML schema. The subset XML schema may also remove constructs from the XML schema that do not affect XML Schema validation of instances against the XML schema, which may include removing documentation, appinfo annotations, and comments.

[Best Practice 10-21] (SUB)
The value of the targetNamespace attribute owned by the xsd:schema document element of the subset XML schema shall be the same as the value of the targetNamespace attribute owned by the xsd:schema document element of the reference XML schema.

[Best Practice 10-22] (SUB)
The XML schema shall be constructed such that any instance that is XML Schema valid against the XML schema shall also be XML Schema valid against the base XML schema.

Rationale
A subset XML schema is a briefer, abridged form of its base XML schema. The subset XML schema is intended to stand in the place of the base XML schema for the purpose of XML Schema validation in many situations. As such, it is imperative that the subset XML schema sustain the constraints expressed by the base XML schema.

10.6 XML Schema Container Elements
Often an XML document may contain multiple items, such as Dates and POCs. SISO recommends the uses an XML Element as a container class to support such potential multiples. These might include for

Copyright © 2016 SISO. All rights reserved
This is an unapproved SISO Reference Product Draft, subject to change.
example Dates, POCs, Associations, and Keywords. Within the resulting XML document based on the XML schema, a container element would be represented with repeating individual sub elements.

```xml
<Keywords>
  <Keyword ms:taxonomy="Functionality" ddms:value="LVC Environment" />
  <Keyword ms:taxonomy="Mission Space" ddms:value="Air Warfare" />
  <Keyword ms:taxonomy="Products" ddms:value="Force Allocation" />
  <Keyword ms:taxonomy="Battlespace" ddms:value="Mission" />
  <Keyword ms:taxonomy="Maturity" ddms:value="Mature - cyclical improvement" />
  <Keyword ms:taxonomy="Service" ddms:value="JCS" />
  <Keyword ms:taxonomy="Aggregation" ddms:value="Squadron" />
</Keywords>
```

In this example Keywords provides the container, which contains one or more Keyword sub elements.

[Best Practice 10-23] (REF, SUB, EXT)

Within the XML schema, repeating elements of the same type SHALL occur under a “containing” complex type.

Rationale

It is easier for to support container classes in software if the containers follow a consistent pattern across XML schema implementations.

11 NORMATIVE - XML SCHEMA NAMING BEST PRACTICES

This section outlines the best practices used to create names for NIEM data components previously discussed in this document. Data component names shall be understood easily both by humans and by machine processes. These best practices improve name consistency by restricting characters, terms, and syntax that could otherwise allow too much variety and potential ambiguity. These best practices also improve readability of names for humans, facilitate parsing of individual terms that compose names, and support various automated tasks associated with dictionary and controlled vocabulary maintenance.

11.1 Extension of XSD Namespace Simple Types

[Best Practice 11-1] (REF, SUB, EXT)

Within the XML schema, a complex type that is a direct extension of a simple type from the XML Schema namespace simple type MAY use the same local name as the simple type if and only if the extension adds no content other than the attribute group structures:SimpleObjectAttributeGroup.

Rationale

It is useful to build complex type bases for further extension.. Requiring name changes for those NIEM-provided complex type bases would work against user understanding, for those already familiar with the names of the XML Schema namespace simple types being extended.

11.2 Usage of English

[Best Practice 11-2] (REF, SUB, EXT)

The name of any XML Schema component defined by the XML schema SHALL be composed of words from the English language, using the Oxford English Dictionary [6] (current edition), as specified by the SISO Policy for the Style and Format of SISO Documents (SISO-ADM-005)[2].

Rationale

The English language has many spelling variations for the same word. For example, American English “program” has a corresponding British spelling “programme.” This variation has the potential to cause interoperability problems when XML components are exchanged because of the different
names used by the same elements. Providing users with a dictionary standard for spelling will mitigate this potential interoperability issue.

11.3 Characters in Names

[Best Practice 11-3] (REF, SUB, EXT)

The name of any XML Schema component defined by the XML schema SHALL contain only the following characters:

- Upper-case letters ("A"-"Z").
- Lower-case letters ("a"-"z").
- Digits ("0"-'9").
- Hyphen ("-").

Other characters, such as the underscore ("_") character and the period ("." ) character SHALL NOT appear in component names in SISO-conformant XML schemas.

[Best Practice 11-4] (REF, SUB, EXT)

The hyphen character ("-" ) MAY appear in component names only when used as a separator between parts of a single word, phrase, or value, which would otherwise be incomprehensible without the use of a separator.

Rationale

Names of standards and specifications, in particular, tend to consist of series of discrete numbers. Such names require some explicit separator to keep the values from running together. The separator used within SISO is the hyphen. Names of SISO XML components follow the best practices of XML Schema, by [Best practice 8-3]. SISO XML components also shall follow the best practices specified for each type of XML Schema component.

11.4 Character Case

[Best Practice 11-5] (REF, SUB, EXT)

Within the XML schema, any attribute declaration SHALL have a name that begins with a lower-case letter ("a"-"z").

[Best Practice 11-6] (REF, SUB, EXT)

Within the XML schema, any XML Schema component other than an attribute declaration SHALL have a name that begins with an upper-case letter ("A"-'Z").

Camel case is the practice of writing compound words or phrases where the words are joined without spaces and are capitalized within the compound words [9].

[Best Practice 11-7] (REF, SUB, EXT)

The name of any XML Schema component defined by the XML schema SHALL use the camel case formatting convention.

Rationale

The foregoing best practices establish lowerCamelCase for all SISO components that are XML attributes and UpperCamelCase for all SISO components that are types, elements, or groups.
11.5 Character Sets

[Best Practice 11-8] (REF, SUB, EXT INS)
Each instance XML document to include schemas SHALL utilize UTF-8 encodings.

Rationale
All XML processors must be able to read entities in both the UTF-8 and UTF-16 encodings as required by [11]. The requirement to use UTF-8 encodings for instance documents only applies to SISO, and not to outside stakeholders who wish to use SISO standards in simulation architectures that employ other character sets and languages.

11.6 Use of Acronyms and Abbreviations

Acronyms and abbreviations have the ability to improve readability and comprehensibility of large, complex, or frequently used terms. They also obscure meaning and impair understanding when their definitions are not clear or when they are used injudiciously. Acronyms and abbreviations should be used with great care. Acronyms and abbreviations that are used shall be documented in the natural language specification of the SISO XML schema product, and used consistently throughout the XML schema.

[Best Practice 11-9] (REF, SUB, EXT)
The XML schema SHALL consistently use acronyms, abbreviations, and word truncations within defined names.

[Best Practice 11-10] (REF, SUB, EXT)
Acronyms, abbreviations shall be defined in the natural language specification of the SISO XML schema product, and used consistently throughout the XML schema.

12 NORMATIVE – INSTANCE (SAMPLE) XML DOCUMENTS

Sample XML documents have the ability to improve the comprehensibility of applying XML schemas, as intended, to construct XML documents. They also obscure meaning and impair understanding when their implementations do not utilize all documented components of a SISO XML schema.

[Best Practice 12-1] (REF, SUB, EXT)
Every SISO XML schema product shall include a sample (instance) XML document.

[Best Practice 12-2] (REF, SUB, EXT)
Every sample XML document accompanying a SISO XML schema product shall include at least one occurrence of every documented component defined in the SISO XML schema.

Rationale
Sample XML documents improve comprehension of a XML schema by depicting how XML documents are intended to be constructed in XML.

13 NORMATIVE - DEPLOYING SISO XML DOCUMENTS

SISO encourages the distribution and sharing of both trial-use standards (and supporting XML schemas) as well as the finalized versions of such XML schemas.

13.1 SISO Schema Location

These XML schemas are hosted at http://www.sisostds.org/schemas/

XML Schemas are to be stored on http://www.sisostds.org/schemas/ within the root folder with no directory hierarchy. Alternatively the XML schemas can be stored on other sites to include open source organizations.
13.2 SISO Schema Catalog

An index.htm (catalog) file is also maintained to show the available XML schemas and supporting examples, independent of where the XML schema products are physically stored. The index.htm file specifies the following information for each SISO XML schema product:

- **Target Namespace** – The target namespace of the SISO XML Schema product.
- **Version** – The product version of the SISO XML Schema product.
- **XML Schema Location** – The web accessible location URL of the SISO XML schema product.
- **SISO Product Number** – The assigned SISO product number of the SISO XML schema product.

**APPENDIX A – INFORMATIVE PRINCIPLES**

The following principles were applied in this document to the development of best practices (requirements) of compliance for SISO schemas.

<table>
<thead>
<tr>
<th>Principle</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Principle 1]</td>
<td>This specification SHOULD specify what is necessary for semantic interoperability and no more.</td>
</tr>
<tr>
<td>[Principle 2]</td>
<td>This specification SHOULD focus on providing best practices for specifying XML schemas.</td>
</tr>
<tr>
<td>[Principle 3]</td>
<td>This specification SHOULD feature best practices that are as specific, precise, and concise as possible.</td>
</tr>
<tr>
<td>[Principle 6]</td>
<td>Systems that use SISO-conformant data SHOULD mark as invalid data that does not conform to the best practices defined by applicable XML Schema documents.</td>
</tr>
<tr>
<td>[Principle 8]</td>
<td>Each SISO-conformant namespace SHOULD be defined by exactly one reference XML schema.</td>
</tr>
<tr>
<td>[Principle 10]</td>
<td>SISO-conformant XML schemas SHOULD NOT use or define local or anonymous components, as they adversely affects reuse.</td>
</tr>
<tr>
<td>[Principle 11]</td>
<td>SISO-conformant components SHOULD NOT incorporate wildcards unless absolutely necessary, as they hinder standardization by encouraging use of non-standardized data rather than standardized data.</td>
</tr>
<tr>
<td>[Principle 12]</td>
<td>XML Schema locations specified within SISO-conformant reference XML schemas SHOULD be interpreted as hints and as default values by processing applications.</td>
</tr>
<tr>
<td>[Principle 14]</td>
<td>SISO-conformant instances and XML schemas SHOULD reuse components from SISO distribution XML schemas when possible.</td>
</tr>
<tr>
<td>Principle</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>[Principle 15]</td>
<td>A component SHOULD be identified by its local name together with its namespace. A namespace SHOULD be a required part of the name of a component. A component’s local name SHOULD NOT imply a relationship to components with similar names from other namespaces.</td>
</tr>
<tr>
<td>[Principle 16]</td>
<td><strong>SISO-conformant XML schemas</strong> and standards SHOULD be designed to encourage and ease extension and augmentation by users and developers outside the standardization process.</td>
</tr>
<tr>
<td>[Principle 17]</td>
<td>XML data SHOULD be designed for automatic processing. XML data SHOULD NOT be designed for literal presentation to people. SISO standards and XML schemas SHOULD NOT use literal presentation to people as a design criterion.</td>
</tr>
<tr>
<td>[Principle 18]</td>
<td>SISO SHOULD NOT depend on specific software packages, software frameworks, or software systems for interpretation of XML instances.</td>
</tr>
<tr>
<td>[Principle 19]</td>
<td>SISO XML schemas and standards SHOULD be designed such that software systems that use SISO XML products may be built with a variety of off-the-shelf and free software products.</td>
</tr>
<tr>
<td>[Principle 20]</td>
<td>A data component definition SHOULD be drafted before the associated data element name is composed.</td>
</tr>
<tr>
<td>[Principle 21]</td>
<td>Components in SISO SHOULD be given names that are consistent with names of other SISO XML components. Such names SHOULD be based on simple best practices.</td>
</tr>
<tr>
<td>[Principle 23]</td>
<td>Component definitions in <strong>SISO-conformant XML schemas</strong> SHOULD have semantic consistency.</td>
</tr>
<tr>
<td>[Principle 24]</td>
<td>Complex type definitions in <strong>SISO-conformant XML schemas</strong> SHOULD use type inheritance only for specialization.</td>
</tr>
<tr>
<td>[Principle 25]</td>
<td>Multiple components with identical or undifferentiated semantics SHOULD NOT be defined. Component definitions SHOULD have clear, explicit distinctions.</td>
</tr>
<tr>
<td>[Principle 26]</td>
<td><strong>SISO-conformant XML schemas</strong> SHOULD have the simplest possible structure, content, and architecture consistent with real business requirements.</td>
</tr>
<tr>
<td>[Principle 27]</td>
<td>Components defined by <strong>SISO-conformant XML schemas</strong> SHOULD be defined appropriate for their scope.</td>
</tr>
<tr>
<td>[Principle 28]</td>
<td>XML namespaces defined by <strong>SISO-conformant XML schemas</strong> SHOULD encapsulate data components that are coherent, consistent, and internally related as a set. A namespace SHOULD encapsulate components that tend to change together.</td>
</tr>
</tbody>
</table>
## APPENDIX B – NORMATIVE BEST PRACTICES

All Best Practices identified in this document comply with the Principles as outlined in Table 6. Informative Principles of Naming and Design.

### Table 7. Normative Best practices for Naming and Design Compliance

<table>
<thead>
<tr>
<th>Best Practice</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Best Practice 8-1] (REF, SUB, EXT, CON)</td>
<td>SISO XML schema SHALL conform to XML as specified by [11]. XML is a well-known, commonly used W3C Recommendation. It is supported by a large number of commercial and open-source software tools. It is a simple, well-defined, semi-structured data format that is flexible enough to allow for easy extension. XML works with many other powerful associated technologies such as XML Schema, XSLT [16], and XPath. XML artifacts of SISO conform to the most recent W3C recommendation for XML.</td>
</tr>
<tr>
<td>[Best Practice 8-2] (REF, SUB, EXT, CON)</td>
<td>The XML schema SHALL conform to the specification for namespaces in XML, as defined by [12] and [13].</td>
</tr>
<tr>
<td>[Best Practice 8-4] (REF, EXT)</td>
<td>Within a SISO-conformant XML schema, the data definition provided for each documented component SHALL follow the requirements and recommendations for data definitions given by [4].</td>
</tr>
<tr>
<td>[Best Practice 9-1] (REF, SUB, EXT)</td>
<td>Within the XML schema, an element xsd:complexType SHALL NOT own the attribute mixed with the value true.</td>
</tr>
<tr>
<td>[Best Practice 9-2] (REF, SUB, EXT)</td>
<td>Within the XML schema, an element declaration that is of complex content SHALL NOT own the attribute mixed with the value true.</td>
</tr>
<tr>
<td>[Best Practice 9-3] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT contain a reference to the type definition xsd:notation or to a type derived from that type.</td>
</tr>
<tr>
<td>[Best Practice 9-4] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT contain the element xsd:notation. XML Schema notations allow the attachment of system and public identifiers on fields of data. The notation mechanism does not play a part in validation of instances and is not supported by SISO.</td>
</tr>
<tr>
<td>[Best Practice 9-5] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT contain the element xsd:include.</td>
</tr>
<tr>
<td>[Best Practice 9-6] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT contain the element xsd:redefine.</td>
</tr>
<tr>
<td>[Best Practice 9-7] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT reference the type xsd:anyType.</td>
</tr>
<tr>
<td>Best Practice</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>[Best Practice 9-8] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT reference the type <code>xsd:anySimpleType</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-9] (REF, SUB, EXT)</td>
<td>Within the XML schema, an element declaration with the attribute name and without the attribute type SHALL carry the attribute abstract with the value true.</td>
</tr>
<tr>
<td>[Best Practice 9-10] (REF, SUB, EXT)</td>
<td>Within the XML schema, an attribute declaration with attribute name SHALL carry the attribute type.</td>
</tr>
<tr>
<td>[Best Practice 9-11] (REF, SUB)</td>
<td>The XML schema SHALL NOT contain the element <code>xsd:any</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-12] (REF, SUB, EXT)</td>
<td>The XML schema SHALL NOT contain the element <code>xsd:anyAttribute</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-13] (REF, SUB, EXT)</td>
<td>Within the XML schema, any occurrence of the element <code>xsd:complexType</code> or <code>xsd:simpleType</code> SHALL appear as an immediate child of the element <code>xsd:schema</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-14] (REF, SUB, EXT)</td>
<td>Within the XML schema, any element declaration carrying the attribute name SHALL appear as an immediate child of the document element <code>xsd:schema</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-17] (REF, SUB)</td>
<td>Within the XML schema, any immediate child of a model group <code>xsd:sequence</code> element SHALL be one of <code>xsd:annotation</code>, <code>xsd:element</code>, <code>xsd:choice</code>, <code>xsd:sequence</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-18] (EXT)</td>
<td>Within the XML schema, any immediate child of a model group <code>xsd:sequence</code> element SHALL be one of <code>xsd:annotation</code>, <code>xsd:element</code>, <code>xsd:choice</code>, <code>xsd:sequence</code> or <code>xsd:any</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-19] (EXT)</td>
<td>Within the XML schema, any immediate child of a model group <code>xsd:choice</code> element SHALL be one of <code>xsd:annotation</code>, <code>xsd:sequence</code> or <code>xsd:element</code>.</td>
</tr>
<tr>
<td>[Best Practice 9-20] (REF, SUB, EXT)</td>
<td>Within the XML schema, if the element <code>xsd:sequence</code> carries the attribute minOccurs, it SHALL set the value for the attribute to 1.</td>
</tr>
<tr>
<td>[Best Practice 9-21] (REF, SUB, EXT)</td>
<td>Within the XML schema, if the element <code>xsd:sequence</code> carries the attribute maxOccurs, it SHALL set the value of the attribute to 1.</td>
</tr>
<tr>
<td>[Best Practice 9-22] (REF, SUB, EXT, CON)</td>
<td>Within the schema, any element <code>xsd:element</code> SHALL NOT carry the attribute default.</td>
</tr>
<tr>
<td>[Best Practice 9-23] (REF, SUB, EXT, CON)</td>
<td>Within the schema, any element <code>xsd:attribute</code> SHALL NOT carry the attribute default.</td>
</tr>
<tr>
<td>[Best Practice 9-24] (REF, EXT)</td>
<td>Within the XML schema, if an element declaration carries the attribute block, it SHALL set the value for the attribute to the empty string.</td>
</tr>
<tr>
<td>Best Practice</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>[Best Practice 9-25] (REF, EXT)</td>
<td>Within the XML schema, if a complex type definition carries the attribute block, it SHALL set the value for the attribute to the empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-26] (REF, SUB, EXT)</td>
<td>Within the XML schema, if the document element xsd:schema carries the attribute blockDefault, it SHALL set the value for the attribute to the empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-27] (REF, SUB)</td>
<td>Within the XML schema, if a simple type definition carries the attribute final, it SHALL set the value for the attribute to the empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-28] (REF, SUB)</td>
<td>Within the XML schema, if a complex type definition carries the attribute final, it SHALL set the value for the attribute to the empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-29] (REF, SUB)</td>
<td>Within the XML schema, if an element declaration carries the attribute final, it SHALL set the value for the attribute to the empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-30] (REF, SUB, EXT)</td>
<td>Within the XML schema, if the document element xsd:schema carries the attribute finalDefault, it SHALL set the value for that attribute to the empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-31] (REF, SUB, EXT, CON)</td>
<td>Within the XML schema, the document element xsd:schema SHALL carry the attribute targetNamespace.</td>
</tr>
<tr>
<td>[Best Practice 9-32] (REF, SUB, EXT, CON)</td>
<td>Within the XML schema, the value of the required attribute targetNamespace on the document element xsd:schema SHALL match the production &lt;absolute-URI&gt; as defined by []. XML Schemas without defined namespaces provide definitions that are ambiguous, in that they are not universally identifiable.</td>
</tr>
<tr>
<td>[Best Practice 9-33] (REF, SUB, EXT, CON)</td>
<td>Within the XML schema, the document element xsd:schema SHALL carry the attribute version of the SISO Product the XML schema applies to.</td>
</tr>
<tr>
<td>[Best Practice 9-34] (REF, SUB, EXT, CON)</td>
<td>Within the XML schema, the value of the required attribute version on the document element xsd:schema SHALL NOT be an empty string.</td>
</tr>
<tr>
<td>[Best Practice 9-35] (REF, SUB, EXT)</td>
<td>Within the XML schema, the element xsd:import SHALL carry the attribute namespace.</td>
</tr>
<tr>
<td>[Best Practice 9-36] (REF, SUB, EXT)</td>
<td>Within the XML schema, the value of the required attribute namespace owned by the element xsd:import SHALL match the production &lt;absolute-URI&gt; as defined by [8].</td>
</tr>
<tr>
<td>[Best Practice 9-37] (REF, SUB, EXT)</td>
<td>Within the XML schema, the element xsd:import SHALL carry the attribute XML schemaLocation.</td>
</tr>
<tr>
<td>[Best Practice 9-38] (REF, SUB, EXT)</td>
<td>Within the XML schema, the value of the required attribute XML schemaLocation carried by the element xsd:import SHALL match either the production &lt;absolute-URI&gt; or the definition of &quot;relative-path reference,&quot; as defined by [8].</td>
</tr>
<tr>
<td>Best Practice</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>[Best Practice 9-39] (REF, SUB, EXT)</td>
<td>Within the XML schema, the value of the required attribute XML schemaLocation carried by the element xsd:import SHALL be resolvable to a XML schema document file that is valid according to [XML SchemaStructures] and [XML SchemaDatatypes].</td>
</tr>
<tr>
<td>[Best Practice 9-40] (REF, SUB, EXT)</td>
<td>Within the XML schema, when a namespace other than the XML namespace or the XML Schema namespace is used, it SHALL be imported into the XML schema using the xsd:import element.</td>
</tr>
<tr>
<td>[Best Practice 9-41] (REF, SUB, EXT)</td>
<td>Within the XML schema, when a namespace other than the XML namespace or the XML Schema namespace is used, its content SHALL be valid with respect to the XML schema imported for that namespace.</td>
</tr>
<tr>
<td>[Best Practice 9-42] (REF, EXT)</td>
<td>Within the XML schema, an element SHALL have at most one instance of an element xsd:annotation as an immediate child.</td>
</tr>
<tr>
<td>[Best Practice 9-43] (REF, EXT)</td>
<td>Within the XML schema, the content of the xsd:documentation element that constitutes the data definition of a component SHALL be character information items as specified by [11].</td>
</tr>
<tr>
<td>[Best Practice 9-44] (REF, SUB, EXT)</td>
<td>XML comments SHALL not be used for persistent information about constructs within the XML schema.</td>
</tr>
<tr>
<td>[Best Practice 9-45] (REF, EXT)</td>
<td>Within the XML schema, any immediate child of an xsd:appinfo element SHALL be an element information item or a comment information item.</td>
</tr>
<tr>
<td>[Best Practice 9-46] (REF, EXT)</td>
<td>Within the XML schema, any element that is an immediate child of an xsd:appinfo element SHALL be in a namespace.</td>
</tr>
<tr>
<td>[Best Practice 9-47] (REF, EXT)</td>
<td>Within the XML schema, an element in the XML Schema namespace SHALL NOT occur as a descendant of any element xsd:appinfo.</td>
</tr>
<tr>
<td>[Best Practice 9-48] (REF, EXT)</td>
<td>Comments SHALL NOT be used to document the normative or descriptive aspects for any component of SISO XML Schemas.</td>
</tr>
<tr>
<td>[Best Practice 9-49] (REF, SUB, EXT)</td>
<td>Within the XML schema, the element xsd:complexType SHALL have as an immediate child either the element xsd:complexContent or the element xsd:simpleContent.</td>
</tr>
<tr>
<td>[Best Practice 9-50] (REF)</td>
<td>Within the XML schema, the element xsd:simpleContent SHALL have as an immediate child the element xsd:extension.</td>
</tr>
<tr>
<td>[Best Practice 9-51] (REF)</td>
<td>Within the XML schema, the element xsd:complexType SHALL have as an immediate child the element xsd:extension.</td>
</tr>
<tr>
<td>Best Practice</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>[Best Practice 9-52] (EXT)</td>
<td>Within the XML schema, given an element xsd:complexContent with a child xsd:restriction owning an attribute base, the attribute base SHALL have a value that resolves to the name of a complex type that is a SISO-conformant component.</td>
</tr>
<tr>
<td>[Best Practice 10-1] (REF, SUB, EXT, CON)</td>
<td>Two XML Schema documents SHALL have the same value for attribute targetNamespace carried by the element xsd:schema, if and only if they represent the same set of components.</td>
</tr>
<tr>
<td>[Best Practice 10-2] (REF, SUB, EXT, CON)</td>
<td>Two XML Schema documents SHALL have the same value for attribute targetNamespace carried by the element xsd:schema, and different values for attribute version carried by the element xsd:schema if and only if they are different views of the same set of components.</td>
</tr>
<tr>
<td>[Best Practice 10-3] (REF, EXT)</td>
<td>Within the XML schema, any element xsd:complexType SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-4] (REF, EXT)</td>
<td>Within the XML schema, any element xsd:simpleType SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-5] (REF, EXT)</td>
<td>Within the XML schema, any element xsd:element that is an immediate child of an element xsd:schema SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-6] (REF, EXT)</td>
<td>Within the XML schema, any element xsd:attribute that is an immediate child of an element xsd:schema SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-7] (REF, EXT)</td>
<td>Within the XML schema, any element xsd:enumeration SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-8] (REF, EXT)</td>
<td>Within the XML schema, the document element xsd:schema SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-9] (REF, EXT)</td>
<td>A data definition SHALL NOT contain explicit representational or data typing information such as number characters, type of characters, etc., unless the very nature of the component can be described only by such information.</td>
</tr>
<tr>
<td>[Best Practice 10-10] (REF, SUB, EXT)</td>
<td>Within the XML schema, a simple type definition that uses xsd:list SHOULD NOT be defined if any member of the list requires a property or metadata that is different than other members of the list. All members of the list SHOULD have the same metadata, and should be related via the same properties.</td>
</tr>
<tr>
<td>[Best Practice 10-11] (REF, SUB, EXT)</td>
<td>Any type definition referenced by a component within the XML schema SHALL be from one of the following: SISO-conformant XML schemas are based on other SISO-conformant XML schemas and the supporting namespaces. This simplifies processing and understanding of data.</td>
</tr>
<tr>
<td>[Best Practice 10-12] (REF, SUB, EXT)</td>
<td>Any element declaration referenced by a component within the XML schema SHALL be from one of the following:</td>
</tr>
<tr>
<td>Best Practice</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>[Best Practice 10-13] (REF, SUB, EXT)</td>
<td>Any attribute declaration referenced by a component within the XML schema SHALL be from one of the following: SISO-conformant XML schemas are based on other SISO-conformant XML schemas. All attributes and elements shall be from SISO-conformant XML schemas, the XML namespace, or an external namespace. This applies to elements referenced for substitution groups as well. It does not apply to content of the XML schema (e.g., within annotations) or to the XML Schema declarations themselves. It applies only to attributes and elements referenced by the XML Schema components.</td>
</tr>
<tr>
<td>[Best Practice 10-14] (REF, SUB, EXT)</td>
<td>Within the XML schema, a reference element and only a reference element SHALL be defined to be of type IDREF.</td>
</tr>
<tr>
<td>[Best Practice 10-15] (REF, SUB, EXT)</td>
<td>Within the XML schema, any two elements of the form NCName and NCNameReference where the string value of NCName is the same in both forms, SHALL be defined to have identical semantics. SISO XML Guidance recognizes no difference in meaning between a reference element and an element that is not a reference element.</td>
</tr>
<tr>
<td>[Best Practice 10-16] (REF, EXT)</td>
<td>Within the XML schema, an element xsd:import that imports a namespace defined by an external XML schema SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-17] (REF, SUB, EXT)</td>
<td>Within the XML schema, an adapter type SHALL be composed of only elements and attributes from an external standard.</td>
</tr>
<tr>
<td>[Best Practice 10-18] (REF, EXT)</td>
<td>Within the XML schema, an element reference used in an adapter type definition SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-19] (REF, EXT)</td>
<td>Within the XML schema, an attribute reference used in an adapter type definition SHALL be a documented component.</td>
</tr>
<tr>
<td>[Best Practice 10-20] (REF, SUB, EXT)</td>
<td>Within the XML schema, an adapter type SHALL NOT be extended or restricted.</td>
</tr>
<tr>
<td>[Best Practice 10-21] (SUB)</td>
<td>The value of the targetNamespace attribute owned by the xsd:schema document element of the subset XML schema shall be the same as the value of the targetNamespace attribute owned by the xsd:schema document element of the reference XML schema.</td>
</tr>
<tr>
<td>[Best Practice 10-22] (SUB)</td>
<td>The XML schema shall be constructed such that any instance that is XML Schema valid against the XML schema shall also be XML Schema valid against the base XML schema.</td>
</tr>
<tr>
<td>[Best Practice 10-23] (REF, SUB, EXT)</td>
<td>Within the XML schema, repeating elements of the same type SHALL occur under a “containing” complex type.</td>
</tr>
<tr>
<td>[Best Practice 11-1] (REF, SUB, EXT)</td>
<td>Within the XML schema, a complex type that is a direct extension of a simple type from the XML Schema namespace simple type MAY use the same local name as the simple type if and only if the extension adds no content other than the attribute group structures:SimpleObjectAttributeGroup.</td>
</tr>
<tr>
<td>Best Practice</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>[Best Practice 11-3] (REF, SUB, EXT)</td>
<td>The name of any XML Schema component defined by the XML schema SHALL contain only the following characters: Other characters, such as the underscore ('_') character and the period (',') character SHALL NOT appear in component names in SISO-conformant XML schemas.</td>
</tr>
<tr>
<td>[Best Practice 11-4] (REF, SUB, EXT)</td>
<td>The hyphen character ('-') MAY appear in component names only when used as a separator between parts of a single word, phrase, or value, which would otherwise be incomprehensible without the use of a separator.</td>
</tr>
<tr>
<td>[Best Practice 11-5] (REF, SUB, EXT)</td>
<td>Within the XML schema, any attribute declaration SHALL have a name that begins with a lower-case letter ('a'-'z').</td>
</tr>
<tr>
<td>[Best Practice 11-6] (REF, SUB, EXT)</td>
<td>Within the XML schema, any XML Schema component other than an attribute declaration SHALL have a name that begins with an upper-case letter ('A'-'Z').</td>
</tr>
<tr>
<td>[Best Practice 11-7] (REF, SUB, EXT)</td>
<td>The name of any XML Schema component defined by the XML schema SHALL use the camel case formatting convention.</td>
</tr>
<tr>
<td>[Best Practice 11-8] (REF, SUB, EXT)</td>
<td>Each instance XML document to include schemas SHALL utilize UTF-8 encodings. Rationale.</td>
</tr>
<tr>
<td>[Best Practice 11-9] (REF, SUB, EXT)</td>
<td>The XML schema SHALL consistently use acronyms, abbreviations, and word truncations within defined names.</td>
</tr>
<tr>
<td>[Best Practice 11-10] (REF, SUB, EXT)</td>
<td>Acronyms, abbreviations shall be defined in the natural language specification of the SISO XML schema product, and used consistently throughout the XML schema.</td>
</tr>
<tr>
<td>[Best Practice 12-1] (REF, SUB, EXT)</td>
<td>Every SISO XML schema product shall include a sample XML document.</td>
</tr>
<tr>
<td>[Best Practice 12-2] (REF, SUB, EXT)</td>
<td>Every sample XML document accompanying a SISO XML schema product shall include at least one occurrence of every documented component defined in the SISO XML schema.</td>
</tr>
</tbody>
</table>