

eCH-0018: XML Best Practices

Name	XML Best Practices
Standard-Nummer	eCH-0018
Kategorie	Best Practice / Musterlösung
Reifegrad	Definiert
Status	Vorschlag für Vernehmlassung
Ausgabedatum	2004-10-31
Gültig seit	2004-12-16
Gültigkeitsdauer	-
Änderungen	-
Ersetzt Standard	-
Basiert auf	-
Sprachen	Deutsch und Französisch
Autoren	Fachgruppe XML-Standards: Hanspeter Salvisberg hanspeter.salvisberg@unisys.com (bis Version 0.07) Alexander Pina alexander.pina@unisys.com (bis Version 0.07) Erik Wilde net.dret@dret.net
Herausgeber / Vertrieb	Verein eCH, Laupenstrasse 18a, 3008 Bern 031 560 00 20 / info@ech.ch / www.ech.ch

Zusammenfassung

Das vorliegende Dokument beschreibt Regeln, welche bei der Benutzung von XML und von XML Schemas in eCH Standards zu berücksichtigen sind. Dabei wird das Schwergewicht auf Basismechanismen und Grundsatzüberlegungen gestellt, welche sich die Benutzern von XML Schemas in der Regel stellen.

Inhaltsverzeichnis

1	Anwendungsbereich	5
1.1	XML Terminologie.....	5
1.2	Terminologie der Empfehlungen	5
1.3	Weiterführende eCH-Dokumente	6
	Teil I: XML Dokumente.....	7
2	XML Versionen.....	7
2.1.1	Problemstellung	7
2.1.2	Empfehlungen.....	7
3	Generelle Form von XML Dokumenten	7
3.1	Character Encodings	8
3.1.1	Problemstellung	8
3.1.2	Empfehlungen.....	8
3.2	Zeilenumbrüche	9
3.2.1	Problemstellung	9
3.2.2	Empfehlungen.....	9
4	Entities.....	9
4.1	Internal Entities	10
4.1.1	Empfehlungen.....	10
4.2	External Entities	10
4.2.1	Empfehlungen.....	11
4.3	Character References.....	11
4.3.1	Empfehlungen.....	11
5	Namen von Elementen und Attributen	12
5.1	Die Sprache der Namen	12
5.1.1	Problemstellung	12
5.1.2	Empfehlungen.....	12
5.2	Namenskonventionen	12
5.2.1	Problemstellung	12
5.2.2	Empfehlungen.....	13
6	XML Namespaces	14
6.1	Namespace Namen	14
6.1.1	Problemstellung	14
6.1.2	Empfehlungen.....	14
6.2	Namespace Deklarationen	15
6.2.1	Problemstellung	16
6.2.2	Empfehlungen.....	16
6.3	Namespace Präfixe	16
6.3.1	Problemstellung	16
6.3.2	Empfehlungen.....	16

7	Die Sprache von Inhalten.....	17
7.1	Textuelle Inhalte.....	17
7.1.1	Problemstellung	17
7.1.2	Empfehlungen.....	17
7.2	Aufzählungswerte	17
7.2.1	Problemstellung	17
7.2.2	Empfehlungen.....	18
Teil II: XML Schemas.....		19
8	Kennzeichnungen im XML Schema.....	19
8.1	Namespaces	19
8.1.1	Problemstellung	19
8.1.2	Empfehlungen.....	19
8.2	Versionierung.....	19
8.2.1	Problemstellung	19
8.2.2	Empfehlungen.....	20
9	Namen von Schema-Komponenten.....	20
9.1	Namen von Typen	20
9.1.1	Problemstellung	20
9.1.2	Empfehlungen.....	20
9.2	Namen von Named Groups	21
9.2.1	Problemstellung	21
9.2.2	Empfehlungen.....	21
9.2.3	Empfehlungen Attribut Gruppen	21
9.2.4	Empfehlungen Named Model Gruppen	21
10	Dokumentation.....	22
10.1	Problemstellung	22
10.2	Empfehlungen.....	22
Teil III: XML Schema Instanzen.....		24
11	Schema-Information in Instanzen	24
11.1	Zugehörigkeit zu einem XML Schema.....	24
11.1.1	Problemstellung	24
11.1.2	Empfehlungen	24
11.2	Versionierung.....	24
11.2.1	Problemstellung	24
11.2.2	Empfehlungen	25
12	Verwendung von Schema-Komponenten	25
12.1	Namen von Elementen und Attributen.....	25
12.1.1	Problemstellung	26
12.1.2	Empfehlungen	26
12.2	Defaultwerte für Elemente und Attribute.....	26

12.2.1	Problemstellung	26
12.2.2	Empfehlungen	26
13	Type Substitution.....	27
13.1	Das xsi : type Attribut.....	27
13.1.1	Problemstellung	27
13.1.2	Empfehlungen	27
13.2	Substitution Groups	27
13.2.1	Problemstellung	27
13.2.2	Empfehlungen	28
Anhang A – Referenzen.....		29
Anhang B – Mitarbeit & Überprüfung.....		30
Anhang C – Zuständigkeit und Mutationswesen.....		30
Anhang D – Urheberrechte.....		31
Anhang E – Glossar		31

Status des Dokuments

Das vorliegende Dokument enthält den Text, der dem Expertenausschuss am 16.12.2004 zur Veröffentlichung der Vernehmlassung **vorgeschlagen** wird. Es wurde noch nicht offiziell genehmigt und hat daher noch keine normative Kraft.

1 Anwendungsbereich

Der vorliegende Standard definiert Richtlinien für die Verwendung und Erstellung von XML Dokumenten und für die Verwendung von XML Schemas in eCH Standards. Da auch XML Schemas XML Dokumente sind, werden in einem ersten Teil (Teil I) allgemeine Richtlinien zu XML verfasst, während in den darauffolgenden Teilen spezielle Richtlinien zu XML Schemas (Teil II) und zu XML Dokumenten als Instanzen von XML Schemas (Teil III) erläutert werden.

1.1 XML Terminologie

In der Literatur werden die Begriffe "XML Dokument" und "XML Instanz" oftmals synonym verwendet. Der XML Standard spricht konsequent von "XML Dokumenten", zudem legt der Begriff der "Instanz" die Existenz einer Klassenbeschreibung (im Fall von XML kann das z.B. eine DTD oder ein XML Schema sein) nahe, aber das Konzept des well-formed XML erlaubt auch Dokumente ohne Klassenbeschreibung. Aus diesem Grund wird hier konsequent der Begriff "XML Dokument" verwendet.

Das "Document Element" eines XML Dokumentes ist das Element, das alle anderen Elemente enthält. Häufig wird hierfür der Begriff "Root Element" verwendet, dieser ist jedoch nicht im Standard definiert und wird hier daher vermieden.

Im Zusammenhang mit XML Schema ist zu beachten, dass ein "XML Schema" eine logische Menge an Komponenten ist, die in einem oder mehreren "Schema Dokumenten" definiert sein können. Ein Schema Dokument ist ein XML Dokument, das die XML Elemente des XML Schema Namespaces verwendet, um Schema Komponenten zu definieren. Ein Schema Dokument kann andere Schema Dokumente mittels `xs:include` und `xs:redefine` einbinden, und das gesamte XML Schema besteht dann aus der Auflösung aller solcher Referenzen zwischen "Schema Dokumenten" zu einem logischen "XML Schema". Mit `xs:import` werden Referenzen auf (globale) Komponenten in anderen Schemas ermöglicht (die referenzierten Komponenten werden in diesem Fall aber nicht Teil des referenzierenden Schemas.)

1.2 Terminologie der Empfehlungen

Richtlinien in diesem Dokument werden gemäss der Terminologie aus [RFC2119] angegeben, dabei kommen die folgenden Ausdrücke zur Anwendung, die durch GROSSSCHREIBUNG als Wörter mit den folgenden Bedeutungen kenntlich gemacht werden (Zitat aus RFC 2119):

- **MUST:** This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.

- **MUST NOT:** This phrase, or the phrase "SHALL NOT", mean that that definition is an absolute prohibition of the specification.
- **SHOULD:** This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- **SHOULD NOT:** This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- **MAY:** This word, or the adjective "OPTIONAL", mean that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option **MUST** be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option **MUST** be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

1.3 Weiterführende eCH-Dokumente

Neben dem vorliegenden Dokument beschäftigen weiterführende Dokumente mit verschiedenen Aspekten von XML-Technologien, die vom vorliegenden Dokumente unbehandelte Fragen diskutieren:

- Beschreibung von Namespaces [eCH-0033]
- Design von XML Schemas [eCH-0035]
- Modellierung für den XML-orientierten Datenaustausch [eCH-0036]

Teil I: XML Dokumente

2 XML Versionen

XML existiert in zwei Versionen, XML 1.0 [XML10Third] als der ersten festgelegten Version, und XML 1.1 [XML11] als einer überarbeiteten Version. Die Unterschiede zwischen den beiden Versionen sind recht gering (<http://www.w3.org/TR/xml11/#sec-xml11>), es ist dabei zu beachten, dass bei der Verwendung von XML 1.1 automatisch auch XML Namespace 1.1 [XMLNS11] verwendet werden, da der XML Namespace Mechanismus (siehe Abschnitt 6) keine Methode hat, eine Version (abgesehen von der XML-Version) anzugeben.

2.1.1 Problemstellung

XML 1.1 nimmt nur minimale Änderungen an XML 1.0 vor, kann jedoch nur von Software verarbeitet werden, die XML 1.1 unterstützt. Viele Software (Parser und andere XML-verarbeitende Programme) unterstützen kein XML 1.1, und abgesehen von den Features, die nun auch in XML die Zeilenende-Konventionen von Grossrechnern (<http://www.w3.org/TR/xml11/#sec-line-ends>) zulassen, ist XML 1.1 in den seltensten Fällen notwendig.

Viele XML-Software prüft nicht die Version des verarbeiteten XML und akzeptiert daher (oftmals fälschlicherweise) auch XML 1.1 als Eingabe. Ist die Software jedoch nicht explizit mit XML 1.1 kompatibel, so kann die Verwendung von XML 1.1 spezifischen Features in XML Dokumenten zu unstabilem Verhalten führen.

2.1.2 Empfehlungen

- **SHOULD NOT:** XML Dokumente sollten kein XML 1.1 verwenden, einzig wenn XML 1.1 Features unabdingbar notwendig und die Verwendung von XML 1.1 mit allen Beteiligten abgestimmt worden ist, kann XML 1.1 verwendet werden.
- **MUST:** Wird XML 1.1 verwendet, so muss die deutlich gekennzeichnet sein und es muss darauf hingewiesen werden, dass die betreffenden Dokumente nur mit XML 1.1 fähiger Software verarbeitet werden dürfen.

3 Generelle Form von XML Dokumenten

XML ist ein zeichenorientiertes Format, d.h. es ist auf der Grundlage von Zeichen definiert und nicht als binäres Format. Aus diesem Grunde sind im Zusammenhang mit der Verwendung von XML die klassischen Fragen zeichenorientierter Formate zu klären, insbesondere der verwendete Zeichenvorrat (Abschnitt 3.1), und der Umgang mit Zeilenenden (Abschnitt 3.2). Generelle Begriffsdefinition und weitergehende Betrachtungen zur Verwendung von Zeichen und Zeichensätzen in einer offenen Umgebung finden sich in [WebChar10].

3.1 Character Encodings

Es wird angenommen, dass der verwendete Zeichensatz (das Character Set) im Fall von XML immer Unicode oder eine Untermenge davon (z.B. ASCII oder ISO-8859-?) ist. Das spezifische Character Encoding hingegen (also die Frage, wie ein Zeichen des Zeichensatzes codiert wird), kann unterschiedlich sein.

3.1.1 Problemstellung

Durch Verwendung von unterschiedlichen Character Encodings kann es zu Inkompatibilitäten bezüglich Character Encodings kommen. Die entwickelten Schemas und XML Dokumente sollten mit einer zukunftssicheren Codierung weltweit einsetzbar sein.

In der XML-Spezifikation <http://www.w3.org/TR/REC-xml/#charencoding> ist festgehalten, dass die beiden Codierungen UTF-8 und UTF-16 als 'MUST' definiert sind, wobei Dokumente mit UTF-16 als 'MUST' und mit UTF-8 als 'MAY' mit einer „Byte Order Mark (BOM)“ zu kennzeichnen sind.

Dies hat zur Folge, dass ASCII Dokumente automatisch unproblematisch sind, weil ASCII Dokumente nichts anderes als UTF-8 ohne BOM sind (was erlaubt ist, s.o.), die nur die Zeichen U+0000 bis U+007F benutzen.

Legacy Systeme, die keine UTF-8 Codierung unterstützen, müssen eine andere Form der Codierung wählen. Die Unterstützung der Codierung ist in diesem Fall abhängig von der verwendeten XML-Software.

3.1.2 Empfehlungen

- **SHOULD:** Als Codierung sollte UTF-8 verwendet werden.
- **SHOULD:** Die „encoding“ Deklaration in der XML-Deklaration sollte immer angegeben werden.
- **MUST:** Ist die verwendete Codierung nicht US-ASCII oder UTF-8, so muss die „encoding“ Deklaration in der XML-Deklaration angegeben werden.

Die Art der Codierung der XML Dokumente sollte in der XML-Deklaration in jedem XML Dokument angegeben werden.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Im Falle von Legacy Systemen, welche UTF-8 nicht unterstützen, sind nach Absprache aller beteiligten Partner die von allen Systemen unterstützten Codierungen zu benutzen, diese müssen in der XML-Deklaration angegeben werden.

Beispiele:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<?xml version="1.0" encoding="ISO-2022-JP"?>
```

Der folgende Abschnitt ist zur Information (Auszug aus dem XML-Standard, <http://www.w3.org/TR/REC-xml/#NT-EncodingDecl>). Er hat keine normative Aussagekraft.

In an encoding declaration, the values "UTF-8", "UTF-16", "ISO-10646-UCS-2", and "ISO-10646-UCS-4" SHOULD be used for the various encodings and transformations of Unicode / ISO/IEC 10646, the values "ISO-8859-1", "ISO-8859-2", ... "ISO-8859-*n*" (where *n* is the part number) SHOULD be used for the parts of ISO 8859, and the values "ISO-2022-JP", "Shift_JIS", and "EUC-JP" SHOULD be used for the various encoded forms of JIS X-0208-1997

3.2 Zeilenumbrüche

XML basiert auf einer zeichenbasierten Syntax (Markup), in der die Struktur von Daten durch Elemente und Attribute bestimmt wird. Viele XML-Tools (z.B. XML-Editoren) erzeugen von sich aus viele Zeilenumbrüche zwischen Elementen, die kein relevanter Teil der Daten sind, sondern lediglich die Menschenlesbarkeit des XML vergrössern (<http://www.w3.org/TR/REC-xml/#sec-white-space>).

3.2.1 Problemstellung

Aus Applikationssicht sind sogenannte Whitespace Text Nodes (Text-Knoten, die nur aus Whitespace bestehen, also Space, TAB, NL oder CR) fast nie relevant, und Applikationen sollten so programmiert werden, dass diese Unterschiede keine Auswirkungen auf die Interpretation haben.

ACHTUNG: Whitespace Text Nodes in XML Dokumenten sind signifikant, falls im Dokument durch das xml : space="preserve" Attribut gekennzeichnet, oder falls das Schema für das Element Mixed Content erlaubt (in DTDs mit (#PCDATA | . . .) * markiert, in XML Schema mit minOccurs="true"). Auf diese signifikanten Whitespace Text Nodes bezieht sich die folgende Empfehlung nicht!

3.2.2 Empfehlungen

- **SHOULD**: XML sollte so formatiert werden, dass es gut menschenlesbar ist (Zeilenumbrüche und Einrückungen), soweit dies aus Platzgründen vertretbar ist.
- **MUST**: Applikationen dürfen nicht davon ausgehen, dass der Whitespace in XML-Dokumenten in einer bestimmten Art formatiert ist.

4 Entities

In XML gibt es das Konzept von Entities, unter denen man sich generell ein Stück referenzierbaren Inhalts vorstellen kann. Entities werden deklariert (<http://www.w3.org/TR/REC-xml/#sec-entity-decl>, durch DTD Konstrukte) und referenziert (<http://www.w3.org/TR/REC-xml/#sec-references>, mittels ihres Namens und dem Markup &Name;). Je nachdem, ob das referenzierte Stück Inhalt intern oder extern ist, gibt es internal (Abschnitt 4.1) oder external (Abschnitt 4.2) Entities. Character References (Abschnitt 4.3) sind technisch gesehen keine Entity Referenzen, sind aber syntaktisch und von der Funktion her sehr ähnlich und werden daher ebenso im vorliegenden Abschnitt behandelt.

Das generelle Problem mit Entities ist, dass sie von XML Schema nicht unterstützt werden, d.h. XML Schema hat keine Sprachmittel zur Deklaration von Entities.

4.1 Internal Entities

Das aus HTML bekannte Konstrukt `ü` für ein "ü" ist nichts anderes als eine Referenz auf ein in der HTML-DTD definiertes Entity. Da die Entity-Deklaration direkt den sogenannten Replacement-Text enthält, handelt es sich um ein Internal Entity (<http://www.w3.org/TR/REC-xml/#sec-internal-ent>). Im Fall von XML Schema Instanzen können keine Internal Entities verwendet werden, es sei denn, sie werden im "Internal Subset" (<http://www.w3.org/TR/REC-xml/#NT-intSubset>) eines XML Dokumentes deklariert.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE text [
<!ENTITY uuml "&#xFC;" >
]>
<text>Internal Entity u Uml aut: &uuml;
Character Reference u Uml aut: &#xFC;
Unicode Character u Uml aut: ü
</text>
```

4.1.1 Empfehlungen

- **SHOULD NOT:** XML Dokumente sollten keine Entity-Deklarationen im Internal Subset enthalten. Sind XML Dokumente Instanzen einer DTD, so sind Referenzen auf in der DTD definierte Internal Entities unproblematisch, im Fall von XML Schema sollte auf die Verwendung von im Internal Subset definierte Internal Entities verzichtet werden.
- **SHOULD:** Falls Entities verwendet werden, sollten soweit irgend möglich die in XHTML 1.0 [XHTML10Sec] definierten Entities (<http://www.w3.org/TR/xhtml1/#h-A2>) verwendet werden, weil dadurch auf etablierte Standardbezeichnungen für Sonderzeichen zurückgegriffen wird.

4.2 External Entities

External Entities sind der Mechanismus, mit dem man in XML (mit Hilfe von DTD-Mechanismen) Include-ähnliche Funktionen implementiert. Im Gegensatz zu einem herkömmlichen Include muss man bei External Entities zunächst das Entity deklarieren (<http://www.w3.org/TR/REC-xml/#sec-external-ent>, mit der URI der externen Resource) und anschliessend referenzieren. Die Referenzierung des External Entities ist dann das "Include-Statement", das das Einfügen der externen Resource bewirkt.

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE text [
<!ENTITY xx SYSTEM "i ncl udes/xx. xml " >
]>
<text>Ei nfügen des I nhal ts von Dokument xx. xml : &xx; </text>
```

4.2.1 Empfehlungen

- **SHOULD NOT:** XML Dokumente sollten keine Entity-Deklarationen im Internal Subset enthalten. Sind XML Dokumente Instanzen einer DTD, so sind Referenzen auf in der DTD definierte External Entities unproblematisch, im Fall von XML Schema sollte auf die Verwendung von im Internal Subset definierte External Entities verzichtet werden.
- **SHOULD:** Sind Verweise auf externe Ressourcen notwendig, so sollte dafür XInclude [XInclude] verwendet werden.

4.3 Character References

Character References (<http://www.w3.org/TR/REC-xml/#NT-CharRef>) sehen Entity Referenzen ähnlich, referenzieren aber kein deklariertes Entity, sondern ein durch seinen Code Point identifiziertes Zeichen aus dem Unicode Zeichensatz. Im Prinzip kann man jedes Zeichen (in Element-Inhalten und Attributwerten) in einem XML Dokument durch eine entsprechende Character Reference ersetzen, dies verringert jedoch die Lesbarkeit des Dokuments und vergrössert das Dokument erheblich. Aus diesem Grunde werden oftmals nur Zeichen als Character Reference angegeben, die sich nicht über die Tastatur eingeben lassen, oder die sich mit dem für das XML Dokument verwendeten Character Encoding nicht codieren lassen (z.B. kann ein Ü nicht mit US-ASCII codiert werden, aber über die Character Reference `ü` dennoch in einem US-ASCII codierten XML Dokument verwendet werden).

Beispiel:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE text [
<!ENTITY uuml "&#xFC;" >
]>
<text>Internal Entity u Uml aut: &uuml;
Character Reference u Uml aut: &#xFC;
Unicode Character u Uml aut: ü
</text>
```

4.3.1 Empfehlungen

- **SHOULD:** Character References sollten soweit wie möglich vermieden werden. Dies ist am einfachsten und in den meisten Fällen möglich durch eine geeignete Wahl des Character Encodings (siehe Abschnitt 3.1) des XML Dokuments.
- **MAY:** Sollte das Character Encoding des XML Dokuments gewünschte Unicode Characters nicht codieren können, oder sollten diese Zeichen nicht direkt eingegeben werden können, so können die entsprechenden Zeichen als Character References im Dokument aufgeführt werden.
- **SHOULD:** Bei der Verwendung von Character References ist die hexadezimale Schreibweise (`☺`) der dezimalen Schreibweise (`☺`) vorzuziehen.

5 Namen von Elementen und Attributen

Namen kommen in XML Dokumenten für Elemente und Attribute (und Processing Instruction Targets, auf die nicht näher eingegangen wird) vor. Die Festlegungen im vorliegenden Abschnitt beziehen sich auf die in XML Dokumenten sichtbaren Namen (Elemente und Attribute). Abschnitt 8 behandelt als Ergänzung dazu die Namen in XML Schemas (alle benannten Komponenten neben Elementen und Attributen, also Notationen, Typen, Identity Constraints und Named Groups).

5.1 Die Sprache der Namen

5.1.1 Problemstellung

Wenn die Schemas mit lokalisierten Namen, also auf der Grundlage lokaler Sprachen, entwickelt werden, besteht die Gefahr, dass die Schemas nur in dieser spezifischen Sprachregion akzeptiert werden. Um der Mehrsprachigkeit der Schweiz, aber auch internationalen Anforderungen, gerecht zu werden, muss eine weltweit gültige Lösung angestrebt werden. Zudem wird dadurch die Konsistenz bei der Kombination verschiedener Schemas erhöht, die durch das Auftreten verschiedensprachiger Namen im gleichen XML Dokument stark leiden würde.

5.1.2 Empfehlungen

- **SHOULD:** Die Sprache der Namen ist Englisch
- **MAY:** Falls im Schema Namen verwendet werden, die eine sprachspezifische Bedeutung haben, die sich nicht übersetzen lässt (z.B. juristische Fachwörter), so können diese in der Sprache geschrieben werden, für die die Bedeutung erhalten bleiben muss.
- **SHOULD NOT:** Die obige Ausnahmeregelung sollte nicht dafür verwendet werden, um Schemas dreisprachig parallel zu führen.

Es wird angenommen, dass die Personen, welche mit den Schemas arbeiten, der englischen Sprache mächtig sind. Englische Namen in XML Dokumenten haben sich mit wenigen Ausnahmen in praktisch allen Anwendungsbereichen etabliert.

Die Empfehlungen in diesem Abschnitt sind im Zusammenhang mit Abschnitt **Fehler! Verweisquelle konnte nicht gefunden werden.** zu lesen, der sich mit der Sprache von Aufzählungswerten beschäftigt und in Abschnitt 7.2.2 die gleichen Empfehlungen formuliert wie die hier aufgeführten Empfehlungen für die Sprache der Namen von Elementen und Attributen.

5.2 Namenskonventionen

5.2.1 Problemstellung

Um die Lesbarkeit von XML Dokumenten zu erhöhen und eine allgemein gültige Grundlage für alle XML-Entwickler zu schaffen, ist eine Standardisierung der Struktur der verwendeten Namen (und nicht nur ihrer Sprache) extrem wichtig.

Für die Anwender eines Schemas ist es um vieles einfacher, wenn das Vokabular selbsterklärende und intuitive Namen verwendet, Abkürzungen vermieden werden, und Konsistenz hinsichtlich der Verwendung der Namenskonventionen besteht.

XML ist case-sensitive, d.h. jede Auswahl an Klein- und Grossbuchstaben ist relevant und muss in genau dieser Form überall verwendet werden, und aus diesem Grund werden die folgenden Empfehlungen gegeben.

5.2.2 Empfehlungen

5.2.2.1 Notation der Namen

- **SHOULD:** Alle Namen (mit Ausnahme von Akronymen, siehe Abschnitt 5.2.2.2) werden in Kleinbuchstaben geschrieben.

Beispiel:

```
<gi venname>KI aus</gi venname>  
<surname>Huber</surname>
```

- **MUST:** Doppelpunkte (:) dürfen nicht in Namen verwendet werden, dies würde zu Konflikten mit XML Namespaces (Abschnitt 6) führen und die Dokumente mit praktisch allen XML-Technologien unverarbeitbar machen.
- **SHOULD NOT:** Unterstriche (_), Bindestriche (-) und Punkte (.) sollten nicht in Namen verwendet werden.

Beispiel:

```
Person.Name      → falsch  
Binary_Field     → falsch
```

- **SHOULD:** Namen aus zusammengesetzten Begriffen sollten soweit irgend möglich vermieden werden.
- **SHOULD:** Sind Namen aus zusammengesetzten Begriffen unumgänglich, so werden sie mit CamelCase Notation (mit kleinem Anfangsbuchstaben) verwendet.

Beispiel:

```
<recordNumber>12345678</recordNumber>
```

5.2.2.2 Abkürzungen (Akronyme)

Akronyme kommen in verschiedener Form vor, meistens komplett aus Grossbuchstaben (XML), aber teilweise auch als Mischung (WebDAV), manchmal auch mit kleinem Buchstaben beginnend (xNAL). Um die Abkürzungen wiedererkennbar zu machen, sollte nichts an der etablierten Schreibweise verändert werden, falls sie verwendet werden sollen.

- **SHOULD:** Grundsätzlich sollten keine Abkürzungen verwendet werden. Auch wenn Abkürzungen in einer Wissensdomäne für allgemein bekannt gehalten werden, heisst das nicht, dass sie von allen möglichen Anwendern verstanden werden.
- **MAY:** Gängige Abkürzungen können verwendet werden, müssen aber allgemein bekannt sein.
- **MUST:** Die Bedeutung der Abkürzungen muss im Schema explizit beschrieben und dokumentiert werden.

Beispiel:

- EAN (European Article Numbering) EAN-Artikelnummer
- ISBN (International Standard Book Number) ISBN-Nummer eines Buches

6 XML Namespaces

Die Verwendung und vor allem die Definition und Verwaltung von XML Namespaces im eCH Umfeld wird in einem anderen eCH Dokument [eCH-0033] beschrieben. XML Namespaces [XMLNS] werden verwendet, um Namen eines Vokabulars (z.B. durch eine DTD oder ein XML Schema definiert) eindeutig zu benennen, der Namespace Name (eine URI) und der lokale Name ergeben zusammen einen sogenannten "Qualified Name" (QName), der die Namen weltweit eindeutig macht. Namespaces werden durch die Verbindung des Namespace Namens (Abschnitt 6.1) mit einem Präfix deklariert (Abschnitt 6.2), und in QNames durch die Verwendung eines deklarierten Präfixes (Abschnitt 6.3) referenziert.

6.1 Namespace Namen

Namespace Namen sind URIs, müssen jedoch nicht auf eine existierende Ressource verweisen. Um den Umgang mit Namespaces zu vereinfachen, verlangt [eCH-0033], dass der Namespace Name auf eine Ressource verweisen muss, und regelt zudem, welches Format diese Ressource haben muss. Der Name eines Namespaces ist eine URI, kann also im Prinzip ein beliebiges URI Scheme (z.B. `http` oder `ftp`) verwenden.

6.1.1 Problemstellung

Die grosse Freiheit bei der Auswahl von Namespace Namen macht es schwierig, eine Auswahl zu treffen, in welcher Form Namespace Namen definiert werden sollte. Damit Namespace Namen im Kontext von eCH einem leicht nachvollziehbaren Schema folgen, wird mit den folgenden Richtlinien das Aussehen von Namespace Namen näher geregelt.

6.1.2 Empfehlungen

- **MUST:** eCH Namespace Namen verwenden das `http` URI Scheme

- **MUST:** Der Domain Name ist der allgemein verwendete Web Server Name der definierenden Instanz (z.B. www. ej pd. admi n. ch), mit oder ohne führendes www, je nachdem wie dies verwendet wird.
- **SHOULD:** Der Namespace-Pfad sollte kurz und prägnant sein, idealerweise unter einem gemeinsamen Präfix für alle Namespace Namen, die Empfehlung lautet xml ns. Der Grund dafür ist, dass dieser Name über lange Zeit (die Benutzungsdauer der mit dem Namespace verbundenen Schemas) gültig bleiben sollte, und für einfache Namen lässt dies einfacher erreichen als für komplizierte Pfade.
- **MUST:** Es muss einen Pfad (idealerweise einen Namen) geben für den Anwendungsfall, , z.B. <http://www.ech.ch/xml ns/appbsp>, der auf eine den allgemeinen Anwendungsfall beschreibende Resource verweisen
- **MUST:** Es muss auf den allgemeinen Pfad aufbauende Pfade für konkrete Versionen des Schemas geben, z.B. ht tp: //www. ech. ch/xml ns/appbsp/v1 auf die Beschreibung der ersten Version des Schemas.
- **SHOULD:** Die Namespace Beschreibung unter ht tp: //www. ech. ch/xml ns/appbsp sollte auf alle existierenden Versionen verweisen.
- **MUST:** Der gleiche Namespace Name darf nur für kompatible Schemas verwendet werden, d.h. für Schemas, bei denen die Validierung von Instanzen mit allen Versionen des Schemas erfolgreich möglich ist. Inkompatible Änderungen müssen eine Versionierung des Namespace Namens zur Folge haben.
- **MUST:** Rückwärtskompatible Applikationen müssen die Richtlinien der Namespace Name Versionierung respektieren und Dokumente mit älteren Namespace Namen akzeptieren.

Das Resultat der oben aufgeführten Festlegungen ist ein Namespace Name, der schematisch betrachtet aus den folgenden Komponenten besteht:

http	://	www.ech.c h	/	xml ns	/	appbsp	/	v1
URI Scheme	Syntax	Domain Name	Syntax	Name für Namespacebeschreibungen	Syntax	Anwendungsfal l	Syntax	Versio n

6.2 Namespace Deklarationen

Namespace Deklarationen treten in einem XML Dokument als Attribute auf, die mit der Buchstabenfolge xml ns beginnen. Namespaces können in einem beliebigen Element deklariert werden und sind dann deklariert für dieses Element und alle Nachfolger (direkte oder indirekte Kinder des Elementes).

ACHTUNG: Der Default Namespace (deklariert mit xml ns="...") gilt nicht für Attribute, d.h. Attribute ohne Präfix sind nie einem Namespace zugeordnet. Sollen Attribute aus einem Namespace referenziert werden, so muss dies explizit mittels eines Präfix erfolgen.

6.2.1 Problemstellung

Namespaces können irgendwo in einem XML Dokument deklariert werden, was es schwierig machen kann, ausgehend von einem Element einfach alle dort gültigen Namespaces zu erkennen. Es sollte daher von dieser grossen Flexibilität von Namespace-Deklarationen kein Gebrauch gemacht werden, um die Übersichtlichkeit von XML Dokumenten mit Namespaces zu vergrössern.

6.2.2 Empfehlungen

- **SHOULD:** Namespaces sollten nur im Document Element deklariert werden.
- **SHOULD:** Namespaces sollten nicht umdeklariert werden, d.h. der gleiche Präfix mit einem anderen Namespace Name belegt werden an verschiedenen Stellen im Dokument.
- **SHOULD:** Namespaces sollten nicht undeklariert werden, d.h. Namespace-Deklarationen aufgehoben werden, so dass sie an manchen Stellen im Dokument gültig sind, an manchen daruntergelegenen Stellen aber nicht (in XML Namespace 1.0 [XMLNS] ist das Undeklarieren nur erlaubt für den Default Namespace, in XML Namespaces 1.1 [XMLNS11] auch erlaubt für mit Präfix deklarierte Namespaces).

6.3 Namespace Präfixe

An sich haben Namespace Präfixe nur eine lokale Bedeutung, sie werden verwendet, um in einem XML Dokument den Bezug zwischen der Namespace-Deklaration (xml ns: html ="http://www.w3.org/1999/xhtml ") und einem qualifizierten Namen (<html : ti tl e>) herzustellen.

6.3.1 Problemstellung

Die Wahl des Präfixes ist lokal und kann vom Erzeuger eines XML Dokuments vorgenommen werden. "Sprechende" Namespace Präfixes sind die Regel und machen ein Dokument einfacher lesbar. Eine gleichartige Wahl des Präfixes für alle Dokumente, die Namen eines bestimmten Namespaces verwenden, macht die Dokumente einfacher wiedererkennbar und einfacher vergleichbar.

6.3.2 Empfehlungen

- **SHOULD:** Namespaces sollten mit Ihren empfohlenen oder üblichen Namespace Präfixen (für eCH Schemas durch [eCH-0033] definiert, für andere Schemas soweit definiert oder etabliert) verwendet werden. Abweichungen sollten nur in Ausnahmen (z.B. Konflikt zweier Präfixe) vorgenommen werden.
- **SHOULD:** Bestehen keine durch [eCH-0033] definierten oder anderweitigen Präfixe, so sollte ein "sprechender" Name gewählt werden.

7 Die Sprache von Inhalten

XML Dokumente verwenden Markup, d.h. sie kombinieren Struktur-Informationen (Elemente und Attribute) mit Inhalt (Element-Inhalte und Attributwerte). Zur Sprache des Markup sind die Empfehlungen in Abschnitt 5 gegeben worden, der vorliegende Abschnitt beschäftigt sich mit der Sprache von Inhalten.

7.1 Textuelle Inhalte

XML-Dokumente enthalten an vielen Stellen Text als Inhalt, im Prinzip an all denen Stellen, an denen der Inhalt durch das Schema nicht auf stärker festgelegt Typen (wie Zahlen, Datumsdarstellungen oder die in Abschnitt **Fehler! Verweisquelle konnte nicht gefunden werden.** behandelten Wertaufzählungen) eingeschränkt ist.

7.1.1 Problemstellung

Inhalte in XML Dokumenten können in unterschiedlichen Sprachen auftreten, und mit dem `xml:lang` Attribut (<http://www.w3.org/TR/REC-xml/#sec-lang-tag>) definiert XML einen Mechanismus, wie die Sprache von Inhalten angegeben werden kann. Das `xml:lang` Attribut muss in einem Schema (z.B. einer DTD oder einem XML Schema) definiert werden wie jedes andere Attribut auch, aber durch seine Herkunft aus dem XML Namespace selber ist es über alle Schemas hinweg identifizierbar als das Attribut zur Sprachidentifizierung.

7.1.2 Empfehlungen

- **MUST:** Sprach-Codierungen sind im `xml:lang` Attribut anzugeben.
- **MUST:** Sprach-Codierungen sind gemäss RFC 3066 [RFC3066] darzustellen. Applikationen müssen die zweistelligen ISO 639 [ISO639] Codierungen (de, fr, en) oder die kombinierten ISO 639/3166 [ISO3166] Codierungen (de-CH, en-US) verwenden.
- **SHOULD:** Die detaillierten ISO 639/3166 Codierungen (de-CH, en-US) sind, soweit angebracht, den weniger detaillierten ISO 639 Codierungen (de, fr, en) vorzuziehen.
- **MUST:** Applikationen müssen alle von RFC 3066 definierten Sprachcodierungen erkennen können.

7.2 Aufzählungswerte

Im Gegensatz zu den in Abschnitt 7.1 beschriebenen textuellen Inhalten kann es auch durch das Schema vorgegebene Inhalte geben, insbesondere neben sprachunabhängigen Werten wie Zahlen oder Daten auch Aufzählungen, deren Werte in einer bestimmten Sprache definiert sind.

7.2.1 Problemstellung

Aufzählungen legen ähnlich wie die in Abschnitt 5 beschriebenen Namen von Elementen und Attributen Namen fest, die in Instanzen verwendet werden. Aus diesem Grund sollten sich diese Werte sollten sich an der Sprache des Schemas orientieren, und die folgenden Empfehlungen sind entsprechen denen aus Abschnitt 5.1.2.

7.2.2 Empfehlungen

- **SHOULD:** Die Sprache der Aufzählungswerte ist Englisch
- **MAY:** Falls im Schema Aufzählungswerte verwendet werden, die eine sprachspezifische Bedeutung haben, die sich nicht übersetzen lässt (z.B. juristische Fachwörter), so können diese in der Sprache geschrieben werden, für die die Bedeutung erhalten bleiben muss.
- **SHOULD NOT:** Die obige Ausnahmeregelung sollte nicht dafür verwendet werden, um Schemas dreisprachig parallel zu führen.

Teil II: XML Schemas

8 Kennzeichnungen im XML Schema

XML Schemas können in sehr einfacher Form benutzt werden, insbesondere ist es nicht notwendig, einen Namespace für die definierten Namen anzugeben. Des Weiteren muss auch keinerlei Versionierungsinformation angegeben werden, XML Schema bietet hier keine Unterstützung für Versionierung. Die folgenden Richtlinien behandeln die damit zusammenhängenden Aspekte

8.1 Namespaces

XML Namespaces (wie Abschnitt 6 beschrieben) können durch beliebige Mechanismen beschrieben werden, im besonderen durch ein XML Schema.

8.1.1 Problemstellung

Jedes Schema, das in einem offenen Umfeld benutzt wird, sollte einen Namespace definieren, und dadurch die im Schema definierten Namen eindeutig referenzierbar machen. Die Struktur von Namespace Namen ist in Abschnitt 6.1 beschrieben.

8.1.2 Empfehlungen

- **MUST:** Jedes XML Schema muss einen XML Namespace definieren
- **MUST:** Auf den von einem Schema definierten Namespace muss mit dem `targetNamespace` Attribut des `xs: schema` Elementes verwiesen werden.

8.2 Versionierung

Schemas verändern sich im Laufe der Zeit, sie werden korrigiert, erweitert, neuen Bedürfnissen angepasst, und sollten trotzdem zuverlässig anwendbar sein. In einem offenen Umfeld muss man mit der Koexistenz verschiedener Versionen, Instanzen verschiedener Versionen, und Software-Komponenten mit Support für verschiedene Versionen rechnen. Die folgenden Richtlinien definieren dafür Minimalanforderungen.

8.2.1 Problemstellung

Gemäss Abschnitt 6.1 wird die Versionierung von Major Versions (also von Versionen, die nicht rückwärtskompatibel sind) über den Namespace Name geführt. Damit bleibt als offenes Problem die Versionierung von Minor Versions. XML Schema kennt zwar ein `version` Attribut auf dem `xs: schema` Element, dieses hat jedoch keine definierte Bedeutung und ist vor allem auch nicht in der Instanz sichtbar. Aus diesem Grund definiert eCH ein eigenes Schema (mit einem eigenen Namespace) mit einem `minorVersion` Attribut, dass in allen eCH Schemas importiert werden sollte.

Dieses Schema dient dazu, die Minor Version in eCH Instanzen in einer standardisierten Weise erkennen zu können.

8.2.2 Empfehlungen

- **MUST:** Schemas müssen so geschrieben sein, dass die Minor Version auf dem Document Element mittels eines Attributes angegeben werden muss, und dieses Attribut muss auf use="required" gesetzt sein.
- **SHOULD:** Für das oben verlangte Attribut sollte das minorVersion Attribut aus dem <http://www.ech.ch/xmlns/minorversion/v1> Namespace verwendet werden. Dieser Namespace ist unter dem genannten Namespace Namen dokumentiert.

9 Namen von Schema-Komponenten

Abschnitt 4 beschreibt Richtlinien für die Form von Namen in XML Dokumenten. In XML Schema gibt es Namen für alle benannten Komponenten, also Elemente, Attribute, Notationen, Typen, Identity Constraints und Named Groups (<http://www.w3.org/TR/xmlschema-1/#concepts-data-model>). Generell betrachtet sind Namen von Schema-Komponenten auch Namen innerhalb von XML Dokumenten, und aus diesem Grund gelten die in Abschnitt 4 aufgeführten Richtlinien für die Sprache von Namen (Abschnitt 5.1) und die Form von Namen (Abschnitt 5.2) auch für Namen von Schema-Komponenten.

9.1 Namen von Typen

9.1.1 Problemstellung

Typen in XML Schema sind entweder Simple oder Complex Types, die als Grundlage für andere Typen (Typableitung) oder für Elemente oder Attribute verwendet werden. Typen erhalten einen Namen (falls sie nicht anonym verwendet werden, also direkt in einer anderen Definition enthalten sind) und werden über diesen Typnamen referenziert.

9.1.2 Empfehlungen

- **MUST:** Typen in XML Schema werden mit den gleichen Namensregeln wie Element- und Attributnamen definiert.
- **MUST:** Am Ende des Namens ist "Type" anzubringen.
- **SHOULD:** Der Name des Typs sollte auf seine Verwendung hinweisen. Wird ein Typ ausschliesslich oder hauptsächlich für ein Element/Attribut verwendet, so sollte er dessen Namen (mit dem Zusatz "Type") tragen.

Beispiel:

- personType → richtig
- personTyp → falsch ("Typ" statt "Type")
- CustomerType → falsch (muss mit Kleinbuchstaben beginnen)

9.2 Namen von Named Groups

9.2.1 Problemstellung

Named Groups in XML Schema sind wiederverwendbare Komponenten, die Teile eines Complex Types enthalten können, entweder Attributmengen ("Attribute Groups"), oder Teile von Inhaltsmodellen ("Named Model Groups").

9.2.2 Empfehlungen

- **MUST:** Named Groups in XML Schema werden mit den gleichen Namensregeln wie Element- und Attributnamen definiert.

9.2.3 Empfehlungen Attribut Gruppen

- **MUST:** Bei Attribut-Gruppen wird am Ende des Namens "AttributeGroup" angebracht.

Beispiel:

- genericAttributeGroup → richtig
- transactionAttributGruppe → falsch ("AttributGruppe" statt "AttributeGroup")
- TransactionAttributeGroup → falsch (muss mit Kleinbuchstaben beginnen)

9.2.4 Empfehlungen Named Model Gruppen

- **MUST:** Bei Named Model Groups wird am Ende des Namens "Group" angebracht.

Beispiel:

- keyValueGroup → richtig
- keyValueGruppe → falsch ("Gruppe" statt "Group")
- TransactionFragmentGroup → falsch (muss mit Kleinbuchstaben beginnen)

10 Dokumentation

10.1 Problemstellung

Ein XML Schema definiert nur die syntaktische Struktur von XML Dokumenten, es sagt nichts über deren Bedeutung aus. Die Bedeutung der Syntax sollte zum Teil durch Dokumentation im Schema selber (sofern es die Bedeutung der kleinsten Einheiten und der daraus zusammengesetzten Komponenten ist), und darüber in begleitender Dokumentation (wenn es um Zusammenhänge und den grösseren Anwendungskontext geht) erfolgen. Für die Dokumentation im Schema können Kommentare im XML Schema in XML Kommentare oder in spezielle XML Schema Elemente verwendet werden, beide Arten von Kommentaren haben keine formale Bedeutung für das XML Schema.

Die Sprache der Dokumentation ist dem Kontext entsprechend zu wählen. Je nach Verwendungszweck des Schemas ist es nicht immer nötig und/oder finanziell vorteilhaft, alle Sprachgruppen zu unterstützen. Es ist das für das Anwendungsfeld angemessenes Mass für die Sprache/n der Dokumentation zu finden. Folgende Anwendungskontexte wurden identifiziert:

- Projektübergreifend (beteiligt sind mehrere Stellen der öffentlichen Verwaltung)
- Projektspezifisch (lokalisiert)
- Technisch

10.2 Empfehlungen

- **MUST:** Der `targetNamespace` des Schemas muss auf eine Namespace Beschreibung gemäss [eCH-0033] zeigen, über die das Schema selber, begleitende Dokumentation, und weitere relevante Ressourcen zugänglich gemacht sind.
- **MUST:** Die Dokumentation in XML Schemas ist mittels `xs: annotation` Elementen und darin enthaltenen `xs: documentation` Elementen zu führen.
- **MUST NOT:** Dokumentation in XML Schemas darf nicht in XML Kommentaren (`<!-- ... -->`) geführt werden.
- **MUST:** Alle globalen Komponenten (d.h. die auf dem Top-Level des Schemas befindlichen, d.h. die direkten Kinder des `xs: schema` Elements) sind zu dokumentieren.
- **SHOULD:** Alle vorhandenen Komponenten sollten dokumentiert werden, soweit sie nicht trivial oder selbsterklärend sind.
- **MUST:** Die `xs: documentation` Elemente müssen durch `xml:lang` Attribute hinsichtlich der Dokumentationssprache beschrieben werden (siehe Abschnitt 7 zu allgemeinen Regeln zur Sprachmarkierung in XML Dokumenten). Ist die Dokumentation einsprachig, so genügt die Codierung der Dokumentationssprache im `xml:lang` Attribut des `xs: schema` Elements.

- **MUST:** Schemas im Schweiz-weiten Kontext sind mehrsprachig zu dokumentieren. Die Sprache jeder Dokumentation ist im `xml:lang` Attribut des `xs:documentation` Elementes anzugeben.
 - Deutsch (zwingend)
 - Französisch (zwingend)
 - Italienisch (optional)
- **MAY:** Projektspezifische (lokalisierte) Schemas können in der Projektsprache dokumentiert werden.
- **MAY:** Technische Schemas können zusätzlich oder ausschliesslich in Englisch dokumentiert werden.
- **MUST:** Alle Schemas müssen eine Top-Level Beschreibung besitzen.
 - Die Beschreibung muss eine kurze prägnante Einleitung unter Angabe des Verwendungszwecks und des Kontextes des Schemas bieten.

Beispiel Dokumentation:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="DocumentationSample">
    <xs:annotation>
      <xs:documentation xml:lang="de">Dies ist ein Dokumentationsbeispiel</xs:documentation>
      <xs:documentation xml:lang="fr">C'est une demonstracion Documentation</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="demo" type="demoType">
          <xs:annotation>
            <xs:documentation xml:lang="de">Dokumentation</xs:documentation>
            <xs:documentation xml:lang="fr">Documentation</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Teil III: XML Schema Instanzen

11 Schema-Information in Instanzen

XML Dokumente, die Instanzen eines XML Schemas sind, sollten diese Information in einer Art und Weise enthalten, die den Umgang mit den Instanzen möglichst einfach gestaltet. Dies betrifft zum einen die Frage, wie die Zugehörigkeit zu einem Schema überhaupt signalisiert wird, und des weiteren die Frage, wie Versionierungsinformation behandelt wird.

11.1 Zugehörigkeit zu einem XML Schema

Die Zugehörigkeit eines XML Dokumenten zu einem Schema kann im Prinzip über zwei Mechanismen sichtbar werden: zum einen über den verwendeten Namespace (falls, wie in Abschnitt 8.1 empfohlen, das Schema einen Namespace Namen definiert), und zum anderen über die Möglichkeit des Dokumentes, mittels des `xsi : schemaLocat i on` Attributes direkt auf das Schema zu verweisen.

11.1.1 Problemstellung

Mit dem Namespace Namen und dem `xsi : schemaLocat i on` Attribut gibt es praktisch zwei Alternative Methoden, die Schema-Zugehörigkeit eines XML Dokumentes zu beschreiben, wobei die Verwendung des Namespaces zwingend notwendig ist, damit das XML Dokument korrekt ist.

11.1.2 Empfehlungen

- **MUST:** XML Dokumente müssen den verwendeten Namespace deklarieren (dies sollte gemäss der in Abschnitt 6 aufgeführten Richtlinien erfolgen).
- **SHOULD NOT:** XML Dokumente sollten nicht mittels des `xsi : schemaLocat i on` Attributes auf das Schema verweisen. Der Namespace Namen sollte die relevante Identifikation des Schemas sein.
- **SHOULD NOT:** Enthält eine Instanz ein `xsi : schemaLocat i on` Attribut, so sollten sich eine Applikation nicht darauf verlassen, unter ader angegebenen URI tatsächlich das XML Schema aufzufinden, sondern der Interpretation den Namespace Namens Priorität geben.

11.2 Versionierung

Ein weiterhin offenes Problem ist die Frage, wie die Minor Version in der Instanz vermerkt ist. Mit dem Namespace Namen ist die Major Version abgedeckt, weil gemäss Abschnitt 6.1 die Major Version Teil des Namespace Namen sein sollte.

11.2.1 Problemstellung

Die Minor Version sollte in der Instanz vermerkt sein, damit Applikationen auch abhängig von der Minor Version (und nicht nnur abhängig von der im Namespace Namen vermerkten Major Version) die

richtige Verarbeitung einleiten können. Die folgenden Empfehlungen sind insbesondere im Zusammenhang mit den Empfehlungen aus Abschnitt 8.2 zu lesen.

11.2.2 Empfehlungen

- **MUST:** XML Dokumente, die Instanzen von XML Schemas sind, müssen die komplette Versionsinformation enthalten. Da im Fall von eCH XML Schemas nur die Major Version im Namespace Namen enthalten ist, muss die Minor Version über ein Attribut des Document Element angegeben werden.
- **SHOULD:** Für das oben verlangte Attribut für die Minor Version Number sollte das `minorVersion` Attribut aus dem <http://www.ech.ch/xmlns/minorversion/v1> Namespace verwendet werden. Dieser Namespace ist unter dem genannten Namespace Namen dokumentiert.

12 Verwendung von Schema-Komponenten

12.1 Namen von Elementen und Attributen

Elemente und Attribute in XML Schema sind entweder global definiert (d.h. auf dem Top Level des Schemas, also direkt unterhalb des `xs:schema` Elements) oder lokal (d.h. als Teil anderer Komponenten und demnach in diesen definiert). Globale Namen eines Schema müssen immer mit qualifizierten Namen (mit dem `targetNamespace` des Schemas qualifiziert) verwendet werden, während lokale Namen per Default unqualifiziert (also aus keinem Namespace) verwendet werden.

ACHTUNG: Der Default Namespace bezieht sich nur auf Elemente, nicht aber auf Attribute. Dies heisst, dass Attribute, die mit einem Namespace Namen referenziert werden sollen (weil sie global definiert wurden oder im XML Schema `attributeFormDefault="qualified"` wurde), immer mit einem Präfix versehen werden müssen. Im Allgemeinen sind Attribute jedoch lokal definiert und es ist `attributeFormDefault="unqualified"` (der Default) gültig, so dass die Attribute unqualifiziert verwendet werden können.

Beispiele:

```
<el1 attr1="x" xmlns="http://example.com/" />  
<ex:el2 attr2="x" xmlns:ex="http://example.com/" />  
<ex:el3 ex:attr3="x" xmlns:ex="http://example.com/" />
```

In diesem Beispiel ist `el1` aus dem Namespace `http://example.com/` und `attr1` aus keinem Namespace, auch `el2` ist aus dem Namespace `http://example.com/` und `attr2` aus keinem Namespace, während auch `el3` aus dem Namespace `http://example.com/` ist, aber in diesem Fall `attr3` ebenso. Der dritte Fall ist selten und kommt fast nur in Fällen vor, in denen Attribute verwendet werden, die auf vielen Elementen definiert sind, daher eine gewisse Eigenständigkeit haben, und aus einem eigenen Namespace stammen, wie z.B. das in Abschnitt 7 beschriebene `xml:lang` Attribut.

12.1.1 Problemstellung

Durch die `elementFormDefault` und `attributeFormDefault` Attribute des Schemas kann angegeben werden, in welcher Form lokal definierte Namen des Schemas verwendet werden müssen (global definierte Namen müssen immer qualifiziert referenziert werden). Diese Attribute haben also einen grossen Einfluss darauf, wie Instanzen eines Schemas hinsichtlich der Verwendung von Namespaces auszusehen haben.

12.1.2 Empfehlungen

- **SHOULD:** Ist für `elementFormDefault` nichts oder `elementFormDefault="unqualified"` (der Default-Wert) angegeben, so sollte der `targetNamespace` des Schemas mit Präfix deklariert werden und die global definierten Elemente mit diesem Präfix versehen werden.
- **SHOULD:** Ist `elementFormDefault="qualified"` angegeben, so kann (falls dieses Schema das für diese Instanz hauptsächlich verwendete ist) der `targetNamespace` des Schemas als Default Namespace deklariert werden und alle Elemente ohne Präfix verwendet werden.
- **SHOULD:** Ist `attributeFormDefault="qualified"` angegeben, so sollte der `targetNamespace` des Schemas mit Präfix deklariert werden und die lokal definierten Attribute mit diesem Präfix versehen werden.

12.2 Defaultwerte für Elemente und Attribute

12.2.1 Problemstellung

XML Schema erlaubt es, Defaultwerte für Elemente und Attribute anzugeben (in DTDs ist dieser Mechanismus nur für Attribute vorhanden). Auf diese Weise können im Schema Werte angegeben werden, die im Dokument verwendet werden, falls die entsprechende Komponente im Dokument nicht vorhanden ist (bzw. vorhanden aber leer, dies gilt nur für Elemente).

Defaultwerte machen eine Validierung gemäss dem Schema zwingend notwendig, weil sich die Interpretation des Dokuments durch die Validierung ändert (die Defaultwerte kommen als Resultat der Validierung zu den im Dokument selbst vorgefundenen Informationen hinzu). Aus diesem Grund sind Defaultwerte ein recht problematischer Mechanismus, der grosse Auswirkungen auf das Verarbeitungsmodell hat.

12.2.2 Empfehlungen

- **SHOULD:** XML Dokumente sollten sich nicht auf in einem Schema definierte Defaultwerte verlassen, sondern Werte immer explizit angeben, auch wenn die Werte identisch mit im Schema definierten Defaultwert sind.
- **MUST:** Verlässt sich der Erzeuger eines XML Dokumentes auf Defaultwerte, indem er diese Werte in Dokumenten nicht angibt, so darf er das nur tun, falls hinsichtlich der weiteren Verarbeitung des Dokumentes sichergestellt ist, dass diese Schema-basiert erfolgt und daher die Defaultwerte in Betracht ziehen wird.

13 Type Substitution

XML Schema ist typorientiert, d.h. alle Elemente und Attribute haben einen Typ, der aus ihrer Definition ersichtlich ist, auch wenn man diesen Typen einer Instanz alleine (also dem Dokument ohne das Schema) nicht ansieht. XML Schema definiert Typersetzungsverfahren, die in zwei unterschiedlichen Arten verwendet werden können. In beiden Fällen können die Verfahren nur für Elemente abgewendet werden.

In beiden Fällen wird in der Instanz ein Typ anstelle des im Schema angegebenen Typen verwendet, wobei der Typ in der Instanz vom im Schema angegebenen Typen abgeleitet sein muss. Im Fall des `xsi : type` Attributes (Abschnitt 13.1) wird der ersetzte Typ explizit in der Instanz angegeben, im Fall der Ersetzungsgruppen (Abschnitt 13.2) ergibt sich der ersetzte Typ durch das verwendete Element.

13.1 Das `xsi : type` Attribut

In einer Instanz kann ein Element seinen Typ explizit mit dem `xsi : type` Attribut angeben, damit hat das Element in dieser Instanz einen anderen Typ als Schema angeben.

13.1.1 Problemstellung

Ist ein Element im Schema nicht explizit mit `block=" . . . "` definiert, oder das `blockDefault` Attribut des `xs : schema` Elements gesetzt, so ist Type Substitution mittels `xsi : type` für dieses Element erlaubt (vorausgesetzt, es gibt einen Typen, der vom Typen des Elements abgeleitet ist und nicht in seiner Ableitungsart durch `block` verboten wurde).

13.1.2 Empfehlungen

- **SHOULD:** Ist im Schema explizit dokumentiert oder sogar durch das Schema-Design erzwungen (Typen als `abstract="true"` definiert), dass `xsi : type` verwendet werden kann oder soll, so ist dies zulässig. In der Verarbeitung dieser Instanzen ist dann aber darauf zu achten, dass diese typbasiert erfolgt, so dass die Typzuweisung nicht über den Typ eines Elements im Schema, sondern über den `xsi : type` aus der Instanz erfolgt.

13.2 Substitution Groups

In einer Instanz kann ein Element explizit mit dem `substitutionGroup` Attribut angeben, dass es das referenzierte Element in einer Instanz ersetzen kann. Es darf dann in einer Instanz überall dort vorkommen, wo das im `substitutionGroup` Attribut referenzierte Element erlaubt ist. Der Mechanismus kann auch über mehrere Ebenen definiert werden.

13.2.1 Problemstellung

Ist ein Element im Schema nicht explizit mit `final=" . . . "` definiert, oder das `finalDefault` Attribut des `xs : schema` Elements gesetzt, so kann dieses Element als Head einer Substitution Group

dienen. Die Members der Substitution Group referenzieren den Kopf im substitutionGroup Attribut der Elementdefinition.

13.2.2 Empfehlungen

- **SHOULD:** Ist im Schema explizit dokumentiert oder sogar durch das Schema-Design erzwungen (Elemente, die Substitution Group Heads sind, als abstract="true" definiert), dass Substitution Groups verwendet werden können oder sollen, so ist dies zulässig. In der Verarbeitung dieser Instanzen ist dann aber darauf zu achten, dass diese konsequent auf die Substitution Groups Rücksicht nimmt.

Anhang A – Referenzen

[eCH-0033]	Erik Wilde, <i>XML Namespace Beschreibungen für eCH Schemas</i> , Technical Report eCH-0033, eCH , 2005.
[eCH-0035]	Erik Wilde, <i>Design von XML Schemas</i> , Technical Report eCH-0033, eCH , 2005.
[eCH-0036]	Erik Wilde, <i>Modellierung für den XML-orientierten Datenaustausch</i> , Technical Report eCH-0033, eCH , 2005.
[ISO639]	International Organization for Standardization, <i>Codes for the Representation of Names of Languages</i> , ISO 639, July 2002.
[ISO3166]	International Organization for Standardization, <i>Codes for the Representation of Names of Countries and their Subdivisions</i> , ISO 3166, November 2001.
[RFC2119]	Scott O. Bradner, <i>Key words for use in RFCs to Indicate Requirement Levels</i> , Internet Best Current Practice RFC 2119, March 1997. ftp://ftp.isi.edu/in-notes/rfc2119.txt
[RFC3066]	Harald Tveit Alvestrand, <i>Tags for the Identification of Languages</i> , Internet best current practice RFC 3066, January 2001. ftp://ftp.isi.edu/in-notes/rfc3066.txt
[WebChar10]	Martin J. Dürst, François Yergeau, Richard Ishida, Misha Wolf, Tex Texin, <i>Character Model for the World Wide Web 1.0: Fundamentals</i> , World Wide Web Consortium, Working Draft PR-charmod-20041122, November 2004. http://www.w3.org/TR/2004/PR-charmod-20041122
[XInclude]	Jonathan Marsh, David Orchard, <i>XML Inclusions (XInclude) Version 1.0</i> , World Wide Web Consortium, Candidate Recommendation CR-xinclude-20040413, April 2004. http://www.w3.org/TR/2004/CR-xinclude-20040413
[XHTML10Sec]	Steven Pemberton, <i>XHTML 1.0: The Extensible HyperText Markup Language (Second Edition)</i> , World Wide Web Consortium, Recommendation REC-xhtml1-20020801, August 2002. http://www.w3.org/TR/2002/REC-xhtml1-20020801
[XML10Third]	Tim Bray, Jean Paoli, C. Michael Sperberg-McQueen, Eve Maler, François Yergeau, <i>Extensible Markup Language (XML) 1.0 (Third Edition)</i> , World Wide Web Consortium, Recommendation REC-xml-20040204, February 2004. http://www.w3.org/TR/2004/REC-xml-20040204
[XML11]	Tim Bray, Jean Paoli, C. Michael Sperberg-McQueen, Eve Maler, François Yergeau, John Cowan, <i>XML 1.1</i> , World Wide Web Consortium, Recommendation REC-xml11-20040204, February 2004. http://www.w3.org/TR/2004/REC-xml11-20040204
[XMLNS]	Tim Bray, Dave Hollander, Andrew Layman, <i>Namespaces in XML</i> , World Wide Web Consortium, Recommendation REC-xml-names-19990114, January 1999. http://www.w3.org/TR/1999/REC-xml-names-19990114
[XMLNS11]	Tim Bray, Dave Hollander, Andrew Layman, Richard Tobin, <i>Namespaces in XML 1.1</i> , World Wide Web Consortium, Recommendation REC-xml-names11-20040204, February 2004. http://www.w3.org/TR/2004/REC-xml-names11-20040204

Anhang B – Mitarbeit & Überprüfung

Hiermit danken die Autoren allen Mitgliedern der eCH Arbeitsgruppe XML für ihre Mitarbeit und ihre wertvollen Anregungen, Korrekturen und Erweiterungsvorschläge.

Böhm	Markus	Microsoft Schweiz GmbH
Bucher	Hans-Ulrich	Avataris AG
Eisenhut	Claude	Eisenhut Informatik
Gähler	Urs	VRSG
Hotz	Jürg	Kanton Thurgau (SIK)
Keller	Adrian K.	SAG Software Systems AG
Müller	Willy	ISB
Ostertag	Patrick	Etat de Fribourg (SIK)
Pina	Alexander	Unisys (Schweiz) AG
Probst	Fabian	Fachhochschule Solothurn Nordwestschweiz
Salvisberg	Hanspeter	Unisys (Schweiz) AG
Schmid	Harald	
Wiedmer	Hans Ulrich	KOGIS - LT
Eugster	Patrick	Sun
Vock	Hansruedi	BIT
Cédric	Perrenoud	ISC EJPD
Keller	Adrian K.	SAG Software Systems AG T-Systems
Sulzer	Christof	Novell (Schweiz) AG

Anhang C – Zuständigkeit und Mutationswesen

Für die Überarbeitung dieses Standards ist die eCH Fachgruppe XML-Standards zuständig.

Anhang D – Urheberrechte

Wer **eCH**-Standards erarbeitet, behält das geistige Eigentum an diesen. Allerdings verpflichtet sich der Erarbeitende mittels spezieller, schriftlicher Vereinbarung, sein betreffendes geistiges Eigentum oder seine Rechte an geistigem Eigentum anderer, sofern möglich, den jeweiligen Fachgruppen und dem Verein **eCH** kostenlos zur uneingeschränkten Nutzung und Weiterentwicklung im Rahmen des Vereinszweckes zur Verfügung zu stellen.

Die von den Fachgruppen erarbeiteten Standards können unter Nennung der jeweiligen Urheber von **eCH** unentgeltlich und uneingeschränkt genutzt, weiterverbreitet und weiterentwickelt werden.

eCH-Standards sind vollständig dokumentiert und frei von lizenz- und/oder patentrechtlichen Einschränkungen. Die dazugehörige Dokumentation kann unentgeltlich bezogen werden.

eCH-Standards sind vollständig dokumentiert und frei von lizenz- und/oder patentrechtlichen Einschränkungen. Die dazugehörige Dokumentation kann unentgeltlich bezogen werden.

Diese Bestimmungen gelten ausschliesslich für die von **eCH** erarbeiteten Standards, nicht jedoch für Standards oder Produkte Dritter, auf welche in den **eCH**-Standards Bezug genommen wird. Die Standards enthalten die entsprechenden Hinweise auf die Rechte Dritter.

Anhang E – Glossar

Ein ausführliches Online-Glossar ist unter <http://dret.net/glossary/> zu finden.

ASCII	<i>American Standard Code for Information Interchange</i>	der kleinste gemeinsame Nenner (fast) aller Zeichencodierungen (7-Bit Codierung), darauf aufbauende Codierungen erweitern ASCII z.B. um Umlaute (ISO 8859-1) oder ganz allgemein internationale Zeichen (Unicode)
Attribut		Knoten in einem XML Dokument, können Elementen zugeordnet werden (enthalten zusätzliche Informationen zu einem Element)
Content Model		bestimmt in einem Schema (DTD oder XML Schema) den erlaubten Inhalt eines Element-Typs (definiert damit die erlaubte Verwendung des Element-Typs)
DTD	<i>Document Type Definition</i>	die im XML Standard selber definierte Schema Sprache für XML Dokumente, hat insbesondere Schwächen im Bereich der Datentypen
Element		grundlegendstes Strukturierungsmerkmal eines XML Dokuments
Entity		ein Stück XML Text, kann definiert und referenziert werden (es gibt verschiedene Typen, die wichtigsten sind General und Parameter Entities)
HTML	<i>Hypertext Markup Language</i>	Dokumentenformat des WWW, basiert auf SGML und ist inkompatibel mit XML (aus diesem Grund gibt es mit XHTML eine XML-basierte Version von HTML)
Information Set		das Datenmodell von XML, modelliert XML als Information Items mit Properties
ISO 8859		8-Bit Codierung von Zeichen, weit verbreitet, aber inkompatibel mit UTF-8
Markup		die physische Form eines XML Dokuments (Serialisierung des Information Set)

Namespaces		Methode zur Benennung und Verwendung von Namensräumen in XML
Processing Instruction		Verarbeitungsanweisungen in XML Dokumenten, nicht Teil des eigentlichen Inhalts, sondern eher Zusatzinformationen (Syntax: "<?name content?>")
Schema		Beschreibung einer Klasse von XML Dokumenten, die bekanntesten Dialekte sind DTD (Teil von XML selber) und XML Schema (separater Standard)
Unicode		Standard für die Referenzierung und Codierung sehr vieler Zeichen
URI/URL	<i>Universal Resource Identifier/Locator</i>	Standard für die Adressierung von Ressourcen auf dem Web. besteht aus einem <i>Scheme Identifier</i> (z.B. "http:") und weiterer Information zur Adressierung
UTF-8	<i>Unicode Transformation Format 8</i>	die XML Standard-Codierung von Unicode Zeichen, jedes ASCII Dokument ist auch ein UTF-8 Dokument (UTF-8 codiert Zeichen als 1-6 Bytes)
valid		well-formed und den Einschränkungen einer DTD gehorchend
well-formed		ein XML Dokument ist well-formed, wenn es den XML Syntax-Regeln gehorcht
XInclude	<i>XML Include</i>	regelt die Einbindung von externen XML-Ressourcen in XML Dokumente
XML	<i>Extensible Markup Language</i>	Sprache für die Repräsentation von baumstrukturierten Dokumenten und ihr Schema
XML Schema		Schema Sprache für XML, die deutlich leistungsfähiger ist als die in XML DTDs
XSLT	<i>XSL Transformations</i>	Teil von XSL, auf XML-Transformation spezialisierte Programmiersprache die es ermöglicht, ein XML Dokument in eine andere Repräsentation (XML, HTML, ...) zu transformieren

Änderungskontrolle, Prüfung, Genehmigung

Version	Datum	Name oder Rolle	Bemerkungen (geändert, geprüft, genehmigt)	*(geplant)
0.01	2003-10-21	Hp. Salvisberg	Erstellung	
0.03	2004-03-04	Alexander Pina	Ausformulierung des Gerüstes	
0.04	2004-03-08	Alexander Pina	Revision/Review Korrekturen für Gremiums Review	
0.05	2004-03-10	Alexander Pina	Korrekturen nach Review XML Gremium eCH	
0.06	2004-05-06	Alexander Pina	Korrekturen eCH Sitzung, Review	
0.07	2004-05-17	Bianca Bredow	Review	
0.08	2004-09-13	Erik Wilde	1. Durchsicht Erik Wilde	
0.09	2004-11-15	Erik Wilde	Einarbeitung der Kommentare vom 22.9.04 (eCH-Sitzung)	
0.10	2004-12-01	Erik Wilde	Einarbeitung der Kommentare vom 25.11.04 (eCH-Sitzung)	
1.0	2004-12-06	Erik Wilde	Abschliessende Änderungen für Version 1.0	