

eCH-0018: Meilleures pratiques XML

Nom	Meilleures pratiques XML
Numéro de norme	eCH-0018
Catégorie	Norme
Degré de maturité	Expérimental
Etat	Approuvé
Date d'édition	21.6.2005
Valable dès le	25.8.2005
Durée de validité	-
Version	1.0
Remplace la norme	-
Se fonde sur	-
Modifications	-
Langues	Allemand et français
Editeur / Distributeur	Association eCH, Laupenstrasse 18a, 3008 Berne 031 560 00 20 / info@ech.ch / http://www.ech.ch/
Auteurs	Groupe spécialisé "Normes XML": Hanspeter Salvisberg (hanspeter.salvisberg@unisys.com) (jusqu'à la version 0.07) Alexander Pina (alexander.pina@unisys.com) (jusqu'à la version 0.07) Erik Wilde (net.dret@dret.net)

Résumé

Le présent document décrit les règles à respecter lors de l'utilisation du langage XML et de schémas XML dans les normes eCH. Il met l'accent sur les mécanismes de base et les principes qui intéressent en règle générale les utilisateurs de schémas XML.

Table des matières

1	Etat du document.....	5
2	Secteur d'application.....	5
2.1	Terminologie XML.....	5
2.2	Terminologie des recommandations.....	5
2.3	Documents eCH complémentaires.....	6
3	Documents XML en tant qu'instances de classes.....	7
3.1	Exposé du problème.....	7
3.2	Recommandations.....	7
4	Versions XML.....	8
4.1	Exposé du problème.....	8
4.2	Recommandations.....	8
5	Forme générale des documents XML.....	9
5.1	Principe du tout-en-un.....	9
5.1.1	Exposé du problème.....	9
5.1.2	Recommandations.....	9
5.2	Codage des caractères.....	9
5.2.1	Exposé du problème.....	10
5.2.2	Recommandations.....	10
5.3	Coupages de ligne.....	11
5.3.1	Exposé du problème.....	11
5.3.2	Recommandations.....	11
5.4	XML et données binaires.....	11
5.4.1	Exposé du problème.....	11
5.4.2	Recommandations.....	12
5.5	Gestion des versions.....	12
5.5.1	Exposé du problème.....	12
5.5.2	Recommandations.....	12
6	Entités.....	12
6.1	Internal Entities.....	13
6.1.1	Recommandations.....	13
6.2	External Entities.....	13
6.2.1	Recommandations.....	14
6.3	Character References.....	14
6.3.1	Recommandations.....	14
7	Noms des éléments et des attributs.....	15

7.1	La langue des noms	15
7.1.1	Exposé du problème	15
7.1.2	Recommandations	15
7.2	Conventions de nommage	16
7.2.1	Exposé du problème	16
7.2.2	Recommandations	16
7.3	Utilisation uniforme des noms	17
7.3.1	Exposé du problème	18
7.3.2	Recommandations	18
8	Espaces nominatifs XML.....	18
8.1	Déclarations d'espaces nominatifs	19
8.1.1	Exposé du problème	19
8.1.2	Recommandations	19
8.2	Préfixes d'espace nominatif	19
8.2.1	Exposé du problème	20
8.2.2	Recommandations	20
9	La langue des contenus	20
9.1	Contenus textuels	20
9.1.1	Exposé du problème	20
9.1.2	Recommandations	21
9.2	Valeurs d'énumération	21
9.2.1	Exposé du problème	21
9.2.2	Recommandations	21
10	Définitions dans le schéma XML	22
10.1	Espaces nominatifs	22
10.1.1	Exposé du problème	22
10.1.2	Recommandations	22
10.2	Gestion des versions.....	22
10.2.1	Exposé du problème	22
10.2.2	Recommandations	23
11	Noms des composantes d'un schéma	23
11.1	Noms des types	23
11.1.1	Exposé du problème	23
11.1.2	Recommandations	23
11.2	Noms des Named Groups.....	24
11.2.1	Exposé du problème	24
11.2.2	Recommandations	24
11.2.3	Recommandations concernant les groupes d'attributs.....	24
11.2.4	Recommandations concernant les Named Model Groups.....	24
12	Documentation,	25

12.1	Exposé du problème	25
12.2	Recommandations	25
13	Information sur le schéma dans les instances.....	27
13.1	Appartenance à un schéma XML.....	27
13.1.1	Exposé du problème	27
13.1.2	Recommandations	27
13.2	Gestion des versions.....	28
13.2.1	Exposé du problème	28
13.2.2	Recommandations	28
14	Utilisation de composantes d'un schéma.....	28
14.1	Noms des éléments et des attributs	28
14.1.1	Exposé du problème	29
14.1.2	Recommandations	29
14.2	Valeurs par défaut pour les éléments et les attributs	30
14.2.1	Exposé du problème	30
14.2.2	Recommandations	30
15	Exclusion de responsabilité/Références aux droits de tiers	30
16	Droits d'auteur.....	31
	Annexe A: Exemple de schéma XML et de document XML.....	32
	Annexe B: Références	35
	Annexe C: Collaboration et contrôle	37
	Annexe D: Compétences et procédures de mutation	38
	Annexe F: Glossaire.....	38

1 Etat du document

Le présent document a été **approuvé** par le comité d'experts le 25 août 2005. Il a force de norme pour le champ d'application défini dans le secteur de validité fixé.

2 Secteur d'application

La présente norme définit les directives à appliquer pour l'utilisation et l'élaboration de documents XML et pour l'utilisation de schémas XML dans les normes eCH. Les schémas XML étant eux-mêmes des documents XML, elle formule d'abord (partie I) des directives générales concernant ce langage, avant de présenter des directives spéciales concernant les schémas XML (partie II) et les documents XML en tant qu'instances de schémas XML (partie III).

2.1 Terminologie XML

La littérature spécialisée utilise souvent les termes "document XML" et "instance XML" comme s'ils étaient synonymes. La norme XML parle systématiquement de "documents XML", alors que le terme "instance" suggère l'existence d'une définition de classes (dans le cas du langage XML, cela peut être par exemple une DTD ou un schéma XML), le concept du XML "well-formed" permettant toutefois aussi l'utilisation de documents sans description de classes. C'est pourquoi nous emploierons ici le terme de "document XML".

Le "Document Element" d'un document XML est l'élément qui contient tous les autres éléments. Pour le désigner, on utilise fréquemment le terme "élément racine", qui n'est toutefois pas défini dans la norme et que nous n'utiliserons donc pas dans le présent document.

En relation avec le langage XML Schema, on observera qu'un "schéma XML" constitue une quantité logique de composantes qui peuvent être définies dans un ou plusieurs "documents schéma". Un "document schéma" est un document XML qui utilise les éléments XML de l'espace nominatif XML Schema pour définir des composantes "schéma". Un "document schéma" peut intégrer d'autres documents (`xs:include` et `xs:redefine`) et l'ensemble du schéma XML devient alors un schéma XML logique après résolution de toutes ces références entre "documents schéma". L'expression `xs:import` permet de faire référence à des composantes (globales) définies dans d'autres schémas (ces composantes ne font alors pas partie intégrante du schéma qui y fait référence).

2.2 Terminologie des recommandations

Les directives formulées dans ce document sont classées selon la terminologie de la norme [RFC 2119], les expressions suivantes, écrites en LETTRES CAPITALES, sont utilisées avec les significations indiquées (citation du document 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.)

2.3 Documents eCH complémentaires

Outre la présente norme, d'autres documents, consacrés à différents aspects des technologies XML, présentent des aspects non traités ici. Le présent document ne contient aucune référence normative aux documents en cours d'élaboration au moment de la publication de la norme eCH-0018. Une fois publiés, ces documents pourront toutefois être lus en complément à la norme CH-0018.

- Description d'espaces nominatifs XML [eCH-0033]
- Conception de schémas XML [eCH-0035]
- Modélisation de l'échange de données orienté XML [eCH-0036]

Partie I: Documents XML

3 Documents XML en tant qu'instances de classes

En principe, les documents XML, en tant que documents "well-formed", ne peuvent suivre aucune autre règle que celles de la norme XML, mais seront en outre, en règle générale, des instances d'une classe de document. Les classes de documents sont souvent définies par des langages de représentation de schéma, qui fixent, par des règles ou autres définitions, les conditions que doit remplir une instance de cette classe.

3.1 Exposé du problème

XML définit lui-même son propre langage de représentation de schéma, la *DTD (Document Type Definition)*. Les DTD ont une position spéciale du fait qu'elles font partie intégrante de la norme XML et doivent par conséquent être prises en charge par tout logiciel XML. D'autre part, les DTD comportent des faiblesses marquées au niveau performance, notamment dans le domaine des types de données et des mécanismes de modélisation. C'est pourquoi le W3C a développé un autre langage de représentation de schéma, appelé *XML Schema* [XMLSchema1, XMLSchema2], qui est plus performant que les DTD. Par son utilisation dans des normes telles que WSDL (description d'interfaces pour Web Services) et XQuery (langage de consultation de bases de données XML), XML Schema est devenu la nouvelle norme de fait pour les descriptions de schémas XML. XML Schema comporte toutefois aussi des faiblesses dans des secteurs d'application spéciaux, qui sont mieux couverts par d'autres langages de représentation de schéma, tels que Schematron [ISO19757-3] ou RELAX NG [RELAXNG].

Le développeur d'un vocabulaire XML devra choisir le langage de représentation de schéma qu'il choisira, d'une part, pour pouvoir écrire un schéma aussi bon que possible, mais aussi, d'autre part, pour être compris par les autres utilisateurs du schéma et bénéficier des logiciels auxiliaires adéquats.

3.2 Recommandations

- **SHOULD**: Les classes de documents XML devraient être décrites par un schéma permettant aux utilisateurs de comprendre les règles selon lesquelles les documents de cette classe doivent être conçus.
- **SHOULD**: Les classes de documents XML devraient être décrites par un schéma XML nettement plus performant que les DTD dans les secteurs des types de données et de la modélisation.
- **SHOULD NOT**: Les classes de documents XML ne devraient pas être décrites par des DTD, car ces dernières permettent une définition moins précise que XML Schema.
- **MAY**: Si le cas d'application le justifie, il est possible d'utiliser des schémas complémentaires ou alternatifs écrits dans d'autres langages de représentation (p. ex.

Schematron ou RELAX NG), qui sont moins répandus que XML Schema, mais offrent en revanche d'autres fonctionnalités.

- **SHOULD**: Lors du traitement de documents XML, des validations devraient être effectuées aux endroits critiques du processus de travail, pour garantir le respect de XML.

4 Versions XML

XML existe en deux versions: XML 1.0 [XML10Third], qui est la version initiale et XML 1.1 [XML11], qui est une version remaniée. Ces deux versions présentent assez peu de différences (<http://www.w3.org/TR/xml11/#sec-xml11>); on observera à cet égard que l'utilisation de XML 1.1 implique automatiquement celle des espaces nominatifs XML 1.1 [XMLNS11], car le mécanisme de l'espace nominatif XML (voir chapitre 8) ne prévoit aucune possibilité d'indiquer de version (sauf pour la version XML proprement dite).

4.1 Exposé du problème

La version XML 1.1 ne présente que des modifications minimales par rapport à la version 1.0. Elle ne peut toutefois être traitée que par du logiciel supportant explicitement XML 1.1. De nombreux logiciels (programmes d'analyse et autres programmes traitant le langage XML) ne comprennent pas la version 1.1, qui n'est d'ailleurs que très rarement nécessaire, comme par exemple pour les fonctionnalités permettant l'utilisation, dans XML aussi, des conventions de fin de ligne des grands ordinateurs (<http://www.w3.org/TR/xml11/#sec-line-ends>).

De nombreux logiciels XML ne vérifient pas la version du langage XML traité et acceptent par conséquent (très souvent à tort) la version 1.1 comme entrée. Si le programme concerné n'est pas explicitement compatible avec XML 1.1, l'utilisation de fonctions spécifiques à cette version dans des documents XML peut entraîner des instabilités.

4.2 Recommandations

- **SHOULD NOT**: Les documents XML ne devraient pas utiliser la version XML 1.1, sauf si des fonctions de celle-ci sont absolument nécessaires et que son utilisation a été coordonnée avec toutes les parties prenantes.
- **MUST**: Si elle est utilisée, la version XML 1.1 doit être identifiée clairement dans la documentation et il y a lieu d'indiquer que les documents concernés ne peuvent être traités que par des logiciels compatibles avec XML 1.1. La version XML 1.1 est toujours reconnaissable dans les documents, car la déclaration XML (qui contient l'indication de la version) est obligatoire dans XML 1.1 (<http://www.w3.org/TR/xml11/#NT-XMLDecl>).

5 Forme générale des documents XML

XML est un format orienté caractères, c'est-à-dire qu'il est défini sur la base de caractères et non pas comme format binaire. Préalablement à l'utilisation de XML, les questions classiques se posant pour les formats orientés caractères devront donc être éclaircies; on déterminera notamment le jeu de caractères utilisé (paragraphe 5.1) et la manière de traiter les fins de ligne (paragraphe 5.3). Une définition générale des termes et des considérations plus détaillées sur l'utilisation de caractères et de jeux de caractères dans un environnement ouvert figurent dans [WebChar10].

5.1 Principe du tout-en-un

XML définit non seulement un format de données, mais comprend différentes normes (XML elle-même, XML Schema) qui définissent aussi un modèle de traitement devant englober un logiciel conforme à cette norme. En partant de ce modèle de traitement, on peut par conséquent utiliser le format XML de manière qu'il exige, avant même le traitement spécifique à l'application, un traitement non trivial tel que l'assemblage d'un document sur la base de plusieurs éléments provenant de différentes sources.

5.1.1 Exposé du problème

Les DTD (avec external entities et attributs par défaut) et XML Schema (avec attributs par défaut) permettent de définir des documents et/ou des schémas dont l'interprétation dépend de ressources externes et/ou du schéma concerné. Cela rend le traitement des documents de ce genre de classe complexe et sujet à l'erreur.

5.1.2 Recommandations

- **SHOULD**: Les classes de documents et les documents devraient être définis et utilisés de manière que toute l'information nécessaire pour le traitement d'un document y soit contenue.
- **MUST NOT**: Des valeurs par défaut ne doivent pas être définies pour les attributs dans le schéma, sauf s'il s'agit de constantes définies par le schéma lui-même et dont l'existence peut être considérée comme une condition préalable lors du traitement.

Commentaire: cette règle ne concerne pas les dépendances logiques au niveau des applications, lesquelles peuvent bien entendu être représentées par des mécanismes appropriés. Elle interdit uniquement l'utilisation de mécanismes XML ou spécifiques au schéma qui répartissent l'information sur différentes ressources.

5.2 Codage des caractères

Le jeu de caractères (Character Repertoire) utilisé pour XML est toujours Unicode ou un sous-ensemble de celui-ci (par ex. ASCII ou ISO-8859-X, donc l'un des codes ISO-8859). Le codage

spécifique des caractères (donc la question de savoir comment sont codés les caractères du jeu utilisé) peut en revanche s'en écarter.

5.2.1 Exposé du problème

L'utilisation de différents codages de caractères peut entraîner des incompatibilités. Les schémas et les documents XML réalisés devraient pouvoir être utilisés dans le monde entier avec un codage axé sur l'avenir.

La spécification XML <http://www.w3.org/TR/REC-xml/#charencoding> précise que les deux codes UTF-8 et UTF-16 sont définis comme 'MUST', les documents ayant UTF-16 comme 'MUST' et UTF-8 comme 'MAY' devant être identifiés par un "Byte Order Mark" (BOM).

Il en résulte que les documents ASCII ne posent automatiquement aucun problème parce qu'ils ne sont rien d'autres que des documents UTF-8 sans BOM (ce qui est autorisé de toute façon) qui n'utilisent que les caractères U+0000 à U+007F.

Les anciens systèmes ne supportant pas le codage UTF-8 doivent choisir une autre forme. Le support du codage dépend alors du logiciel XML utilisé.

5.2.2 Recommandations

- **SHOULD**: UTF-8 devrait être utilisé pour le codage.
- **MUST**: La déclaration "encoding" doit toujours être indiquée dans la déclaration XML.
- **MUST**: Si le codage utilisé n'est pas US-ASCII ou UTF-8, la déclaration "encoding" doit être indiquée dans la déclaration XML.

Le genre de codage des documents XML doit être indiqué dans la déclaration XML de chaque document XML.

Exemple:

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

Pour les anciens systèmes ne supportant pas UTF-8, on utilisera, après coordination avec toutes les parties prenantes, des codes supportés par tous les systèmes. Ces codes doivent être indiqués dans la déclaration XML.

Exemples:

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

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

5.3 Coupures de ligne

XML se base sur une syntaxe orientée caractères (Markup), dans laquelle la structure des données est déterminée par des éléments et des attributs. De nombreux utilitaires XML (par ex. les éditeurs) génèrent entre les éléments de nombreuses coupures de ligne qui n'ont aucune signification au niveau des données, mais servent uniquement à améliorer la lisibilité du langage XML (<http://www.w3.org/TR/REC-xml/#sec-white-space>).

5.3.1 Exposé du problème

Du point de vue des applications, ce qu'on appelle les Whitespace Text Nodes (nœuds de texte ne se composant que d'espaces blancs, donc les codes barre d'espacement, tabulateur, passage à la ligne ou retour à la ligne) n'ont pratiquement jamais d'importance, et les applications devraient être programmées de façon que les différences en la matière n'aient aucun effet sur l'interprétation.

ATTENTION: Dans les documents XML, les Whitespace Text Nodes ont une signification si l'attribut `xml:space="preserve"` est activé dans le document ou si le schéma est autorisé pour l'élément Mixed Content (par l'expression `(#PCDATA | . . .)*` dans les DTD, par l'expression `mixed="true"` dans XML Schema). Les recommandations ci-après ne concernent pas ces Whitespace Text Nodes qui ont une signification pour l'application!

5.3.2 Recommandations

- **SHOULD**: XML devrait être formaté de manière à être bien lisible pour l'utilisateur (coupures de ligne et indentations) dans la mesure où la place disponible le permet.
- **MUST**: Les applications ne doivent pas se baser sur le fait que le Whitespace est formaté d'une manière déterminée dans les documents XML.

5.4 XML et données binaires

XML convient principalement pour les contenus textuels et ne permet pas d'intégrer directement des données binaires. Bien que ces dernières puissent être intégrées dans un document XML par transformation en une représentation textuelle, ce type de représentation est peu efficace et convient uniquement pour de petites quantités de données.

5.4.1 Exposé du problème

Si des données XML doivent être combinées avec des données binaires, seules de petites quantités de données binaires devraient être intégrées dans le schéma XML, des quantités plus importantes devant être codées séparément et intégrées par référence. Pour ce genre de données, XML Schema définit les types `xs:hexBinary` (chiffres hexadécimaux en tant que caractères) et `xs:base64Binary` (données binaires représentées en tant que lettres de l'alphabet selon la norme [RFC2045]). La représentation `xs:base64Binary` est plus compacte que la représentation `xs:hexBinary` (`xs:base64Binary` augmente le volume des données d'environ 33%, alors que `xs:hexBinary` l'accroît de 100%).

5.4.2 Recommandations

- **SHOULD:** On ne devrait pas intégrer dans des documents XML des quantités relativement importantes de données binaires, mais les coder séparément et les intégrer par référence. La référence peut être spécifique à l'application ou suivre une norme générale telle que [XLink].
- **SHOULD:** On peut intégrer dans des documents XML de petites quantités de données binaires. Pour le codage, on utilisera de préférence le type de données `xs:base64Binary` de XML Schema. En ce qui concerne l'utilisation et, en particulier, l'observation du traitement des espaces blancs dans XML Schema Base64, nous conseillons de consulter XML Schema: <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/#base64Binary>

5.5 Gestion des versions

Les applications XML ont une longue durée de vie, pendant laquelle différentes versions de schémas peuvent être définies et coexister. Pour permettre leur identification, les schémas doivent être dotés d'une désignation de version.

5.5.1 Exposé du problème

Le problème de la gestion des versions des schémas et de la documentation de différentes versions est traité au chapitre 10 au niveau de la définition du schéma et dans [eCH-0033] à celui de sa description.

Au niveau du document, il faut garantir que celui-ci contient l'information l'identifiant comme document d'une version déterminée.

5.5.2 Recommandations

- **MUST:** Les documents XML doivent porter une identification de version s'ils sont les instances d'un schéma. Cette indication identifie la version du schéma.

6 Entités

Dans XML est définie la notion d'entités (entities), par laquelle on peut se représenter, d'une manière générale, une partie de contenu à laquelle il est possible de faire référence. Les entités sont déclarées (<http://www.w3.org/TR/REC-xml/#sec-entity-decl>, par des constructions DTD) et l'on peut y faire référence (<http://www.w3.org/TR/REC-xml/#sec-references>, au moyen de leur nom et de la balise `&Name;`). Les entités peuvent être "internal" (paragraphe 6.1) ou "external" (paragraphe 6.2) suivant que la partie de contenu référencée est interne ou externe. Les Character References (paragraphe 6.3) ne sont pas, d'un point de vue technique, des références d'entité, mais leur sont très similaires sur le plan de la syntaxe et de la fonction et sont par conséquent aussi traitées dans ce chapitre.

Le problème général rencontré avec les entités est qu'elles ne sont pas supportées par XML Schema, c'est-à-dire que ce langage ne comprend aucune possibilité de déclaration d'entités.

6.1 Internal Entities

L'expression `&uuml` ; utilisée dans le langage HTML pour coder le caractère "ü" n'est rien d'autre qu'une référence à une entité définie dans la DTD HTML. Comme la déclaration d'une entité comprend directement ce qu'on appelle le texte de remplacement, il s'agit ici d'une internal entity (<http://www.w3.org/TR/REC-xml/#sec-internal-ent>). Aucune internal entity ne peut être utilisée dans un document XML Schema, sauf si elles sont déclarées dans le sous-ensemble interne ((<http://www.w3.org/TR/REC-xml/#NT-intSubset> de celui-ci

Exemple:

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

6.1.1 Recommendations

- **SHOULD NOT:** Les documents XML ne devraient contenir aucune déclaration d'entité dans leur sous-ensemble interne. Si des documents XML sont des instances d'une DTD, les références à des entités internes définies dans la DTD ne posent aucun problème. Dans le cas de XML Schema, il faudrait renoncer à l'utilisation d'entités internes définies dans le sous-ensemble interne.
- **SHOULD:** Si l'on utilise des entités, il faudrait, dans la mesure du possible, se servir de celles qui sont définies dans XHTML 1.0 [XHTML10Sec] (<http://www.w3.org/TR/xhtml1/#h-A2>) parce que l'on fait ainsi appel à des désignations standard, bien établies, pour les caractères spéciaux.

Les cinq entités prédéfinies dans XML pour les caractères de balisage sont exclues de ces recommandations, car elles doivent, par définition, être reconnues par tout processeur XML. Ces entités sont les caractères amp (Ampersand: &), lt (Less Than: <), gt (Greater Than: >), apos (Apostrophe: '), und quot (Quote: ").

6.2 External Entities

Les entités externes constituent le mécanisme par lequel on implémente en XML (au moyen de mécanismes DTD) des fonctions similaires à une commande Include. Avec les external entities, il faut toutefois d'abord déclarer l'entité (<http://www.w3.org/TR/REC-xml/#sec-external-ent>, avec l'URI de la ressource externe) avant d'y faire référence, contrairement au cas d'une commande Include

traditionnelle. La référence à l'entité externe est alors la commande Include qui produit l'inclusion de la ressource externe.

Exemple:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE text [
<!ENTITY xx SYSTEM "includes/xx.xml" >
]>
<text>Insertion du contenu du document xx.xml: &xx;</text>
```

6.2.1 Recommandations

- **SHOULD NOT:** Les documents XML ne devraient contenir aucune déclaration d'entité dans leur sous-ensemble interne. Si des documents XML sont des instances d'une DTD, les références à des entités internes définies dans la DTD ne posent aucun problème. Dans le cas de XML Schema, il faudrait renoncer à l'utilisation d'entités internes définies dans le sous-ensemble interne.
- **SHOULD:** Si des références à des ressources externes sont nécessaires, on devrait utiliser à cet effet la commande XInclude [XInclude].

6.3 Character References

Les Character References (<http://www.w3.org/TR/REC-xml/#NT-CharRef>) ressemblent aux Entity References, mais ne font pas référence à une entité déclarée, mais à un élément, identifié par son Code Point, du jeu de caractères Unicode. En principe, tout caractère (contenu et attribut) peut être remplacé dans un document XML par la Character Reference correspondante. Toutefois, cela réduit la lisibilité du document et en augmente considérablement la taille. Souvent, on utilise donc les Character References uniquement pour représenter les caractères que l'on ne peut pas entrer au clavier ou qui ne peuvent pas être codés à l'aide du Character Encoding utilisé pour le document XML (par ex. un ü ne peut pas être codé en US-ASCII, mais peut quand même être utilisé dans un document XML codé US-ASCII s'il est remplacé par la Character Reference ü).

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

6.3.1 Recommandations

- **SHOULD:** Les caractères qui peuvent être utilisés directement dans le code Character Encoding choisi ne devraient pas l'être en tant que Character References.

- **SHOULD:** Dans la mesure du possible, on évitera l'utilisation de Character References. La manière la plus simple, et qui aboutit dans la plupart des cas, d'y parvenir est de choisir, pour le document XML, un Character Encoding (voir paragraphe 5.1) approprié.
- **MAY:** Si le Character Encoding du document XML ne peut pas coder certains caractères Unicode, ou si ces derniers ne peuvent pas être entrés directement au clavier, ils pourront être mentionnés dans le document en tant que Character References.
- **SHOULD:** Si l'on utilise des Character References, on préférera la représentation hexadécimale (☺) à la représentation décimale (☺).

7 Noms des éléments et des attributs

Des noms sont utilisés dans les documents XML pour désigner des éléments et des attributs (ainsi que des Processing Instruction Targets, qui ne sont pas traités dans ce document). Les règles fixées dans ce chapitre concernent les noms visibles dans les documents XML (pour les éléments et les attributs). En complément, le chapitre 10 traite les noms utilisés dans les schémas XML (pour toutes les composantes désignées en plus des éléments et des attributs, donc les notations, les types, les Identity Constraints et les Named Groups).

7.1 La langue des noms

7.1.1 Exposé du problème

S'ils sont développés à l'aide de noms localisés, donc sur la base d'une langue locale, les schémas risquent de n'être acceptés que dans la région linguistique concernée. Une solution applicable dans le monde entier doit être recherchée pour tenir compte du plurilinguisme de la Suisse, mais aussi des exigences internationales. Lorsque plusieurs schémas sont combinés entre eux, une telle solution augmente en outre la cohérence, qui aurait fortement à souffrir de l'utilisation de noms en différentes langues dans le même document XML.

7.1.2 Recommandations

- **SHOULD:** La langue des noms est l'anglais.
- **MAY:** Si le schéma utilise des noms qui ont une signification spécifique à la langue correspondante et ne peuvent pas être traduits (par ex. certains termes juridiques), ceux-ci peuvent être écrits dans la langue pour laquelle leur sens doit être conservé.
- **SHOULD NOT:** L'exception ci-dessus ne devrait pas être mise à profit pour réaliser parallèlement des schémas en trois langues.
- **MUST:** Les seuls caractères autorisés dans les noms sont les lettres minuscules et majuscules ASCII (a-z et A-Z), les chiffres 0 à 9 ainsi que le souligné (_), le point (.) et le trait d'union (-).

Les personnes travaillant avec les schémas sont censées maîtriser la langue anglaise. A peu d'exceptions près, l'utilisation de noms anglais dans les documents XML s'est bien établie dans pratiquement tous les domaines d'application.

Les recommandations faites dans ce chapitre doivent être lues en relation avec le chapitre 9.2 qui traite de la langue des valeurs d'énumération et qui formule, au paragraphe 9.2.2, les mêmes recommandations que celles qui sont mentionnées ici pour la langue des noms des éléments et des attributs.

7.2 Conventions de nommage

7.2.1 Exposé du problème

La normalisation de la structure des noms utilisés (et non seulement de leur langue) est extrêmement importante pour améliorer la lisibilité des documents XML et créer une base de validité générale pour tous les développeurs XML.

La tâche des utilisateurs des schémas est fortement simplifiée si le vocabulaire utilisé pour ceux-ci est bien explicite et utilise des noms intuitifs, si l'on renonce aux abréviations et si l'utilisation des conventions de nommage est cohérente.

Le langage XML est "case sensitive", c'est-à-dire que le choix des majuscules et des minuscules y est important et doit être respecté partout. C'est pourquoi nous formulons les recommandations ci-après.

7.2.2 Recommandations

7.2.2.1 Notation des noms

- **SHOULD:** Tous les noms (à l'exception des acronymes, voir paragraphe 7.2.2.2) s'écrivent en minuscules.

Exemple:

```
<name>Klaus Huber</name>  
<city>Zürich</city>
```

- **MUST NOT:** Les deux points (:) ne peuvent pas être utilisés à l'intérieur de noms, car cela causerait des conflits avec les espaces nominatifs XML (chapitre 8) et rendrait les documents correspondants inutilisables par la quasi-totalité des technologies XML.
- **SHOULD NOT:** Le souligné (_), le trait d'union (-) et le point (.) ne devraient pas être utilisés dans les noms.

Exemple:

```
Person.Name      → faux  
Binary_Field     → faux
```

- **SHOULD**: Dans la mesure du possible, on évitera d'utiliser des noms composés.
- **MAY**: S'il n'est pas possible d'éviter l'utilisation de noms composés, ceux-ci seront écrits en "casse chameau" (en commençant par une minuscule).

Exemple:

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

- **SHOULD**: Quelle que soit la convention choisie pour la structure des noms et les caractères qu'ils peuvent utilisés, elle devrait être respectée systématiquement dans le schéma concerné.

7.2.2.2 Abréviations (acronymes)

Les acronymes se présentent sous différentes formes, le plus souvent entièrement en majuscules (XML), mais parfois aussi avec les deux types de casse (WebDAV), même en commençant par une minuscule (xNAL). Pour qu'elles puissent être reconnues, les abréviations devraient toujours être écrites de la façon habituelle.

- **SHOULD**: On n'utilisera en principe pas d'abréviations. Même les abréviations connues de manière générale dans un domaine de connaissance ne sont pas obligatoirement comprises par tout un chacun.
- **MAY**: Les abréviations usuelles peuvent être utilisées à condition d'être connues de manière générale.
- **MUST**: L'utilisation d'abréviations doit être décrite et documentée explicitement dans le schéma.

Exemple:

- EAN (European Article Numbering) Numéro d'article EAN
- ISBN (International Standard Book Number) Numéro ISBN d'un livre

7.3 Utilisation uniforme des noms

En XML, les noms d'éléments et d'attributs sont en principe toujours dépendants du contexte. Des éléments sont définis globalement dans les DTD, de manière qu'un nom d'élément désigne toujours un élément de la même déclaration. Par contre, les attributs sont déclarés localement dans les DTD (dans la ATTLIST de l'élément), de sorte que le même nom peut être utilisé plusieurs fois. Les DTD ne connaissent aucun mécanisme pour les attributs déclarés globalement, de sorte qu'une utilisation cohérente des mêmes noms d'attributs est laissée aux bons soins du concepteur des DTD.

Dans XML Schema, la situation est plus compliquée, du fait que les éléments et les attributs peuvent être déclarés tant localement que globalement. C'est pourquoi il est possible dans XML Schema de

définir des éléments et/ou des attributs ayant le même nom, mais pouvant apparaître en différents endroits d'un document, avec une définition complètement différente.

7.3.1 Exposé du problème

Pour l'utilisateur d'un schéma, le fait que des éléments et/ou des attributs ayant le même nom peuvent être utilisés différemment (parce qu'il existe différentes définitions locales dans différents contextes) et/ou ont une signification différente (parce que la sémantique change en fonction du contexte) prête à confusion. Un nom d'élément ou d'attribut devrait être univoque en ce qui concerne son type et sa sémantique afin d'éviter tout malentendu et toute difficulté dans l'utilisation d'un schéma.

7.3.2 Recommandations

- **SHOULD:** Les éléments et attributs de même nom doivent être du même type, c'est-à-dire que l'utilisation d'éléments et d'attributs de nom identique et autorisés à différents endroits d'un document doit toujours être la même. Un élément ou un attribut dont le type n'est pas le même dans différents contextes devrait y avoir un nom différent (ce problème ne peut se poser que pour les éléments et attributs définis localement, car s'ils sont définis globalement, ils ont toujours le même type).
- **SHOULD:** Les éléments et attributs de même nom doivent avoir la même sémantique, c'est-à-dire que leur sens ne doit pas changer en fonction du contexte où ils apparaissent. La sémantique d'un élément ou d'un attribut est indépendante de sa syntaxe; par conséquent, il ne suffit pas que la syntaxe reste la même (c'est-à-dire que le type soit identique, ainsi qu'exigé au paragraphe précédent) pour garantir que la sémantique demeure inchangée.

8 Espaces nominatifs XML

L'utilisation et, surtout, la description et la gestion d'espaces nominatifs XML dans l'environnement eCH sont décrites dans un autre document eCH [eCH-0033]. Les espaces nominatifs XML [XMLNS] sont utilisés pour désigner clairement les noms d'un vocabulaire (défini par exemple par une DTD ou par un schéma XML), le nom de l'espace nominatif (une URI [RFC3986]) et le nom local formant ensemble ce qu'on appelle un nom qualifié (QName). Ainsi, les noms sont univoques dans le monde entier. Les espaces nominatifs sont déclarés par la combinaison de leur nom (paragraphe 8.1) et d'une préfixe (paragraphe 8.1), et sont référencés dans les Qnames par l'utilisation d'une préfixe déclaré (paragraphe 8.2).

8.1 Déclarations d'espaces nominatifs

Dans un document XML, les déclarations d'espaces nominatifs apparaissent sous forme d'attributs commençant par la suite de lettres `xmlns`. Des espaces nominatifs peuvent être déclarés dans un élément quelconque et le sont alors pour celui-ci et tous ses descendants (directs ou indirects).

ATTENTION: L'espace nominatif par défaut (déclaré sous la forme `xmlns=""`) ne s'applique pas aux attributs, c'est-à-dire que les attributs sans préfixe ne sont jamais alloués à un espace nominatif. S'il faut faire référence à des attributs d'un espace nominatif, on le fera toujours explicitement à l'aide d'un préfixe.

8.1.1 Exposé du problème

Des espaces nominatifs peuvent être déclarés à n'importe quel endroit d'un document XML, ce qui peut rendre difficile la reconnaissance, à partir d'un simple élément, de tous les espaces nominatifs qui y sont valides. Afin d'améliorer la clarté des documents XML contenant des espaces nominatifs, il vaudrait donc mieux ne pas faire usage de la grande flexibilité octroyée pour la déclaration de ces derniers.

8.1.2 Recommandations

- **SHOULD:** Les espaces nominatifs ne devraient être déclarés que dans le "Document Element".
- **SHOULD:** Les espaces nominatifs ne devraient pas être redéclarés, c'est-à-dire que le même préfixe ne devrait pas correspondre à différents noms d'espaces nominatifs à plusieurs endroits du document.
- **SHOULD:** Les espaces nominatifs devraient rester déclarés une fois pour toutes, c'est-à-dire que les déclarations d'espaces nominatifs ne devraient jamais être annulées, afin d'éviter qu'elles soient valides à certains endroits du document seulement (XML Namespace 1.0 [XMLNS] n'autorise l'annulation de la déclaration que pour l'espace nominatif par défaut, alors que XML Namespace 1.1 [XMLNS11] l'autorise aussi pour les espaces nominatifs déclarés avec préfixe).

8.2 Préfixes d'espace nominatif

En tant que tels, les préfixes d'espace nominatif n'ont qu'une signification locale; ils sont utilisés pour établir, dans un document XML, la relation entre la déclaration d'espace nominatif (`xmlns:html="http://www.w3.org/1999/xhtml"`) et un nom qualifié (`<html:title>`). On peut même renoncer complètement aux préfixes si un seul espace nominatif est utilisé et cela par la déclaration de l'espace nominatif par défaut (`xmlns="http://www.w3.org/1999/xhtml"`), de sorte que les éléments sans préfixe (`<title>`) peuvent être utilisés. Si plusieurs espaces nominatifs sont utilisés dans un document, il devient toutefois complexe d'y redéfinir l'espace nominatif par défaut chaque fois que c'est nécessaire pour utiliser des préfixes.

8.2.1 Exposé du problème

Le choix du préfixe est local et peut être effectué par l'auteur du document XML concerné. On choisit d'ordinaire des préfixes d'espace nominatif "qui parlent d'eux-mêmes", ce qui rend le document plus

facile à lire. Un choix uniforme du préfixe pour tous les documents utilisant les noms d'un espace nominatif déterminé rend ces documents plus faciles à reconnaître et à comparer.

8.2.2 Recommandations

- **SHOULD**: Les espaces nominatifs devraient être utilisés avec leurs préfixes recommandés ou habituels (s'ils sont définis ou usuels). On ne dérogera à cette règle qu'exceptionnellement (par ex. pour éviter un conflit entre deux préfixes).
- **SHOULD**: En l'absence de préfixes définis ou usuels, on devrait choisir un nom référençant l'espace nominatif de la manière la plus claire et concise possible.
- **MAY**: S'il existe, dans un document, une sorte d'espace nominatif "principal", celui-ci peut être déclaré comme espace nominatif par défaut, ce qui rendra le document plus court et plus clair.

9 La langue des contenus

Les documents XML utilisent le balisage (Markup), c'est-à-dire qu'ils combinent des informations structurelles (éléments et attributs) à des contenus (contenus élémentaires et valeurs d'attribut). La langue du balisage fait l'objet de recommandations formulées au chapitre 7. Le présent chapitre traite de la langue des contenus.

9.1 Contenus textuels

Les documents XML comportent de nombreux contenus textuels, que l'on retrouve en principe partout où le contenu n'est pas restreint par le schéma à des types plus précisément définis (tels que les chiffres, les dates ou les énumérations de valeurs traitées au paragraphe 9.2).

9.1.1 Exposé du problème

Dans les documents XML, les contenus peuvent se présenter dans différentes langues. Sous la forme de l'attribut `xml:lang` (<http://www.w3.org/TR/REC-xml/#sec-lang-tag>), XML définit un mécanisme pour l'indication de la langue des contenus. L'attribut `xml:lang` doit être défini dans un schéma (par ex. DTD ou schéma XML), comme tout autre attribut, mais il est, de par sa provenance de l'espace nominatif XML lui-même, reconnaissable dans tous les schémas comme étant l'attribut d'identification de la langue.

9.1.2 Recommandations

- **MUST**: Le codage de la langue doit être indiqué dans l'attribut `xml:lang`.
- **MUST**: Le codage de la langue doit être représenté conformément à la RFC 3066 [RFC3066]. Les applications doivent utiliser les codes ISO 639 [ISO639] à deux positions (de, fr, it, rm, en) ou les codes ISO 639/3166 [ISO3166] combinés (de-CH, en-US).
- **SHOULD**: Les codes ISO 639/3166 (de-CH, en-US) doivent être préférés, lorsque cela est utile, aux codes ISO 639 (de, fr, en), qui sont moins détaillés.

9.2 Valeurs d'énumération

Contrairement aux contenus textuels décrits au paragraphe 9.1, des contenus prescrits par le schéma peuvent également exister. Il peut s'agir non seulement de valeurs indépendantes de toute langue, comme les chiffres et les dates, mais aussi d'énumérations dont les valeurs sont définies dans une langue déterminée.

9.2.1 Exposé du problème

Les énumérations définissent des noms qui, comme les noms d'éléments et d'attributs décrits au chapitre 7, sont utilisés dans des documents. C'est pourquoi les valeurs correspondantes devraient respecter la langue du schéma et les recommandations ci-après correspondent à celles du paragraphe 7.1.2.

9.2.2 Recommandations

- **SHOULD**: La langue des valeurs d'énumération est l'anglais.
- **MAY**: Les valeurs d'énumération utilisées dans le schéma qui sont intraduisibles parce qu'elles ont une signification spécifique à la langue concernée (par ex. termes juridiques) peuvent être écrites dans la langue pour laquelle leur sens doit être conservé.
- **SHOULD NOT**: L'exception ci-dessus ne devrait pas être mise à profit pour réaliser parallèlement des schémas en trois langues.

Partie II: Schémas XML

10 Définitions dans le schéma XML

Les schémas XML peuvent être utilisés sous une forme très simple. Il n'est en particulier pas nécessaire d'indiquer un espace nominatif pour les noms définis. En outre, aucune information relative à la version n'est requise, XML Schema n'offrant ici aucun soutien à la gestion des versions. Les directives ci-après traitent des aspects en rapport avec cette problématique.

10.1 Espaces nominatifs

Les espaces nominatifs XML (tels qu'ils sont présentés au chapitre 8) peuvent être décrits par des mécanismes quelconques, notamment par un schéma XML.

10.1.1 Exposé du problème

Tout schéma utilisé dans un environnement ouvert devrait définir un espace nominatif et rendre ainsi clairement référençables les noms qu'il définit. La structure des noms d'espaces nominatifs est décrite plus en détail dans le document eCH correspondant [eCH-0033].

10.1.2 Recommandations

- MUST: Tout schéma XML doit définir un espace nominatif XML.
- MUST: Pour faire référence à l'espace nominatif défini par un schéma, on utilisera l'attribut `targetNamespace` de l'élément `xs:schema`.

10.2 Gestion des versions

Les schémas se modifient au fil du temps, ils subissent des corrections et des extensions, sont adaptés à de nouveaux besoins, et devraient quand même rester utilisables en toute fiabilité. Dans un environnement ouvert, il faut compter avec la coexistence de plusieurs versions, avec des instances de versions différentes et avec des composants logiciels supportant des versions distinctes. Les directives ci-après définissent les exigences minimales à respecter à cet égard.

10.2.1 Exposé du problème

Selon le document eCH [eCH-0033], la distinction entre les versions majeures (donc celles qui ne sont pas compatibles vers le bas) s'effectue à travers le nom de l'espace nominatif. La distinction des versions mineures doit encore être résolue. Bien que XML Schema connaisse, pour l'élément `xs:schema`, l'attribut `version`, celui-ci n'a aucune signification définie et, surtout, n'est pas non plus visible dans l'instance. C'est pourquoi eCH définit un schéma spécifique (avec un espace nominatif

propre) ayant un attribut `minorVersion` qui devrait être importé dans tous les schémas eCH. Ce schéma spécifique permet de reconnaître de manière standardisée quelle est la version mineure dans les documents eCH. Le concepteur est libre de choisir dans son application s'il veut donner à la version mineure un numéro simple ou un identificateur plus structuré (p. ex. "2.3.5"), afin de pouvoir affiner davantage l'identification des versions.

10.2.2 Recommandations

- **MUST:** Les schémas doivent être écrits de manière que la version mineure doive être indiquée au moyen d'un attribut sur le "Document Element", et cet attribut doit être mis à la valeur `use="required"`.
- **SHOULD:** Pour l'attribut exigé ci-dessus, on devrait utiliser l'attribut `minorVersion` de l'espace nominatif <http://www.ech.ch/xmlns/minorversion/v1>. Cet espace nominatif est documenté sous le nom d'espace nominatif mentionné.

11 Noms des composantes d'un schéma

Le chapitre 7 décrit les directives concernant la forme des noms dans les documents XML. XML Schema prévoit des noms pour toutes les composantes, à savoir les éléments, les attributs, les notations, les types, les Identity Constraints et les Named Groups (<http://www.w3.org/TR/xmlschema-1/#concepts-data-model>). D'une manière générale, les noms des composantes d'un schéma sont aussi les noms figurant dans les documents XML. C'est pourquoi les directives formulées au chapitre 7 pour la langue (paragraphe 7.1) et la forme (paragraphe 7.2) des noms s'appliquent également aux noms des composantes d'un schéma.

11.1 Noms des types

11.1.1 Exposé du problème

XML Schema prévoit des types, soit simples soit complexes, qui sont utilisés comme base pour d'autres types (dérivation de type) ou pour des éléments et des attributs. Les types reçoivent un nom (s'ils ne sont pas utilisés anonymement, donc directement contenus dans une autre définition) qui sert à y faire référence.

11.1.2 Recommandations

- **MUST:** Dans XML Schema, les types sont définis selon les mêmes règles de dénomination que pour les noms d'éléments et d'attributs.
- **MUST:** La dénomination "Type" doit être ajoutée à la fin du nom.

- **SHOULD:** Le nom du type devrait faire référence à son utilisation. Un type utilisé exclusivement ou essentiellement pour un élément ou un attribut devrait en porter le nom (avec le préfixe "Type").

Exemple:

- personType → juste
- personTyp → faux ("Typ" au lieu de "Type")
- CustomerType → faux (doit commencer par une minuscule)

11.2 Noms des Named Groups

11.2.1 Exposé du problème

Dans XML Schema, les Named Groups sont des composantes réutilisables pouvant contenir certaines parties d'un type complexe, à savoir soit des groupes d'attributs ("Attribute Groups"), soit des parties de modèles de contenus ("Named Model Groups").

11.2.2 Recommandations

- **MUST:** Dans XML Schema, les Named Groups sont définis selon les mêmes règles de dénomination que pour les éléments et les attributs.

11.2.3 Recommandations concernant les groupes d'attributs

- **MUST:** Le nom des groupes d'attributs se termine toujours par le suffixe "AttributeGroup".

Exemple:

- genericAttributeGroup → juste
- transactionAttributGroupe → faux ("AttributGroupe" au lieu de "AttributeGroup")
- TransactionAttributeGroup → faux (doit commencer par une minuscule)

11.2.4 Recommandations concernant les Named Model Groups

- **MUST:** Le nom des Named Model Groups se termine par le suffixe "Group".

Exemple:

- keyValueGroup → juste
- keyValueGroupe → faux ("Groupe" au lieu de "Group")
- TransactionFragmentGroup → faux (doit commencer par une minuscule)

12 Documentation,

12.1 Exposé du problème

Un schéma XML définit uniquement la structure syntaxique de documents XML et n'apporte aucune information d'ordre sémantique. La signification de la syntaxe devrait être définie en partie par une documentation apportée dans le schéma lui-même (lorsqu'il s'agit de celle des unités les plus petites et des éléments qui en sont composés) ainsi que dans une documentation d'accompagnement (lorsqu'il s'agit des relations entre les unités et du contexte de l'application). La documentation interne peut être réalisée à l'aide de commentaires à l'intérieur du schéma XML, soit sous forme de commentaires XML soit comme éléments spéciaux de XML Schema, ces deux types de commentaire n'ayant aucune signification formelle pour le schéma XML.

La langue de la documentation doit être choisie en fonction du contexte. Selon l'affectation du schéma, il n'est pas toujours nécessaire ni financièrement avantageux de réaliser la documentation pour chaque groupe linguistique. Pour choisir la ou les langues à utiliser, on trouvera la mesure qui convient pour le champ d'application concerné. Les contextes d'application suivants ont été identifiés:

- général (concernant plusieurs offices de l'administration publique),
- spécifique au projet (localisé),
- technique.

12.2 Recommandations

- **MUST:** Le targetNamespace du schéma doit pointer sur une définition d'espace nominatif conformément à [eCH-0033] à travers laquelle le schéma lui-même, la documentation qui l'accompagne et d'autres ressources requises sont rendus accessibles.
- **MUST:** La documentation contenue dans les schémas XML doit être réalisée par des éléments `xs:annotation` et `xs:documentation`, ces derniers étant contenus dans les premiers.
- **MUST NOT:** Dans les schémas XML, la documentation ne doit pas être réalisée dans des commentaires XML (`<!-- ... -->`).

- **MUST:** Toutes les composantes globales (se trouvant donc au niveau supérieur du schéma, c'est-à-dire les descendants directs de l'élément `xs:schema`) doivent être documentées.
- **SHOULD:** Toutes les composantes existantes devraient être documentées dans la mesure où leur signification n'est pas triviale ou évidente.
- **MUST:** Les éléments `xs:documentation` doivent être définis au niveau de la langue de documentation par des attributs `xml:lang` (voir chapitre 9 sur les règles générales du marquage de la langue dans les documents XML). Si la documentation est réalisée en une seule langue, le codage de celle-ci dans l'attribut `xml:lang` de l'élément `xs:schema` est suffisant.
- **MUST:** Les schémas dont le contexte couvre toute la Suisse doivent être documentés en plusieurs langues. La langue de chaque documentation doit être indiquée dans l'attribut `xml:lang` de l'élément `xs:documentation`.
 - Allemand (impératif si pas anglais)
 - Français (impératif si pas anglais)
 - Italien (en option)
 - Il est aussi possible de rédiger la documentation en anglais exclusivement.
 - Quelle que soit la langue choisie pour la documentation, on veillera à appliquer ce choix systématiquement à tout le schéma, de sorte à n'y avoir qu'une seule langue de documentation.
- **MAY:** Les schémas spécifiques à un projet (localisés) peuvent être documentés dans la langue de ce dernier.
- **MAY:** Les schémas techniques peuvent être documentés, en plus ou exclusivement, en anglais.
- **MUST:** Tous les schémas doivent être dotés d'une description au plus haut niveau. Cette description doit offrir une introduction concise et bien compréhensible, avec indication de l'affectation et du contexte du schéma.

Partie III: Instances de XML Schema

13 Information sur le schéma dans les instances

Les documents XML qui sont des instances d'un schéma XML devraient contenir cette information d'une manière qui facilite autant que possible leur utilisation. Cette remarque concerne d'une part la manière de signaler l'appartenance à un schéma et d'autre part la façon de traiter l'information relative à la version.

13.1 Appartenance à un schéma XML

L'appartenance d'un document XML à un schéma peut en principe être rendue visible au moyen de deux mécanismes: premièrement, par l'espace nominatif utilisé (si le schéma définit un nom d'espace nominatif conformément à la recommandation du paragraphe 10.1), secondement, par la possibilité du document de faire directement référence au schéma au moyen de l'attribut `xsi : schemaLocation` .

13.1.1 Exposé du problème

L'appartenance d'un document XML à un schéma peut être décrite de deux façons, à savoir à l'aide du nom de l'espace nominatif ou à l'aide de l'attribut `xsi : schemaLocation` . Notons que l'utilisation de l'espace nominatif est impérative pour que le document XML soit correct.

Le nom de l'espace nominatif est un nom abstrait pour le vocabulaire utilisé et ne fait pas obligatoirement référence à une ressource existante. Dans le cadre de eCH, le document [eCH-0033] définit toutefois que le nom de l'espace nominatif doit faire référence à une description du schéma et comment s'effectue la référence au schéma à partir de cette description. Dans les applications concrètes, on travaille cependant presque toujours de manière que l'application concernée reconnaisse le nom de l'espace nominatif, l'identifie comme nom d'espace nominatif connu et utilise le schéma local pour la validation. La manière dont le schéma local est enregistré (en configuration fixe ou dans une antémémoire qui peut être étendue dynamiquement) est entièrement du ressort du concepteur de l'application.

13.1.2 Recommandations

- **MUST:** Les documents XML doivent déclarer l'espace nominatif utilisé (cela devrait être réalisé si les directives du chapitre 8 sont respectées).
- **SHOULD NOT:** Les documents XML ne devraient pas utiliser l'attribut `xsi : schemaLocation` pour faire référence au schéma correspondant. L'identification de ce dernier devrait s'effectuer au moyen du nom de l'espace nominatif.

- **SHOULD NOT:** Si l'instance contient un attribut `xsi : schemaLocation` , les applications ne devraient pas utiliser ce dernier pour retrouver effectivement le schéma XML à l'URI indiquée, mais donner la priorité à l'interprétation du nom de l'espace nominatif.

13.2 Gestion des versions

Un autre problème encore en suspens est la question de savoir comment la version mineure est mentionnée dans l'instance. Le nom de l'espace nominatif règle le cas de la version majeure parce que celle-ci devrait faire partie intégrante du nom de l'espace nominatif conformément au document [eCH-0033].

13.2.1 Exposé du problème

La version mineure devrait être mentionnée dans l'instance pour que les applications puissent mettre en œuvre le traitement qui convient, également en fonction de la version mineure (et non pas seulement en fonction de la version majeure figurant dans le nom de l'espace nominatif). Les recommandations ci-après doivent être comprises notamment en rapport avec les recommandations du paragraphe 10.2.

13.2.2 Recommandations

- **MUST:** Les documents XML qui sont des instances de schémas XML doivent contenir l'information complète relative à la version. Comme seule la version majeure est comprise dans le nom de l'espace nominatif dans le cas des schémas XML de eCH, la version mineure doit être indiquée au moyen d'un attribut du "Document Element".
- **SHOULD:** Comme attribut servant à indiquer le numéro de la version mineure (voir ci-dessus), il faudrait utiliser l'attribut `minorVersion` de l'espace nominatif <http://www.ech.ch/xmlns/minorversion/v1>. Cet espace nominatif est documenté sous le nom d'espace nominatif mentionné.

D'autres recommandations à ce propos figurent au chapitre 10.

14 Utilisation de composantes d'un schéma

14.1 Noms des éléments et des attributs

Dans XML Schema, les éléments et les attributs sont définis soit globalement (c'est-à-dire au niveau supérieur du schéma, donc directement sous l'élément `xs: schema`) soit localement (c'est-à-dire comme partie d'autres composantes et donc définis dans celles-ci). Les noms globaux d'un schéma doivent toujours être utilisés à l'aide de noms qualifiés (au moyen du `targetNamespace` du schéma), alors que les noms locaux sont utilisés par défaut de manière non qualifiée (donc provenant d'aucun espace nominatif).

ATTENTION: L'espace nominatif par défaut se réfère uniquement à des éléments, et non pas à des attributs. Cela signifie que les attributs auxquels il faut faire référence à l'aide d'un nom d'espace nominatif (parce qu'ils ont été définis globalement ou que le schéma XML contient la déclaration `attributeFormDefault="qualified"`) doivent toujours être dotés d'un préfixe. D'une manière générale, les attributs sont toutefois définis localement et la déclaration `attributeFormDefault="unqualified"` s'applique à eux (par défaut), si bien qu'ils peuvent être utilisés de manière non qualifiée.

Exemples:

```
<e1 attr1="x" xmlns="http://example.com/" />
<ex:e12 attr2="x" xmlns:ex="http://example.com/" />
<ex:e13 ex:attr3="x" xmlns:ex="http://example.com/" />
```

Dans cet exemple, `e1` fait partie de l'espace nominatif `http://example.com/` et `attr1` ne fait partie d'aucun espace nominatif; de même, `e12` fait partie de l'espace nominatif `http://example.com/` et `attr2` ne fait partie d'aucun espace nominatif, alors que `e13` fait partie de l'espace nominatif `http://example.com/`, tout comme `attr3` dans ce cas précis. Ce troisième exemple se présente rarement, presque exclusivement dans les cas utilisant des attributs qui sont définis sur de nombreux éléments, qui ont de ce fait une certaine autonomie et qui proviennent de leur propre espace nominatif, tel que l'attribut `xml:lang` décrit au chapitre 9.

14.1.1 Exposé du problème

Les attributs `elementFormDefault` et `attributeFormDefault` du schéma permettent d'indiquer dans quelle forme les noms définis localement du schéma doivent être utilisés (les noms définis globalement doivent toujours être référencés de manière qualifiée). Ces attributs ont donc une grande influence sur l'utilisation des espaces nominatifs par les instances d'un schéma.

14.1.2 Recommandations

- **MUST:** Si la déclaration `elementFormDefault="qualified"` est présente, le `targetNamespace` du schéma peut (si ce schéma est celui qui est principalement utilisé pour ce document) être déclaré comme espace nominatif par défaut, et tous les éléments peuvent alors être utilisés sans préfixe.
- **SHOULD:** Si la déclaration `attributeFormDefault="qualified"` est présente, le `targetNamespace` du schéma devrait être déclaré avec un préfixe, et les attributs définis localement devraient être dotés de ce dernier.

14.2 Valeurs par défaut pour les éléments et les attributs

14.2.1 Exposé du problème

XML Schema permet d'indiquer des valeurs par défaut pour les éléments et les attributs (dans les DTD, ce mécanisme n'est prévu que pour les attributs). Ainsi, il est possible d'indiquer dans le schéma

des valeurs qui sont utilisées dans le document si les composantes correspondantes n'y sont pas présentes (ou présentes mais vides, ce qui ne s'applique qu'aux éléments).

Les valeurs par défaut rendent impérative une validation selon le schéma, parce que cette validation change l'interprétation du document (les valeurs par défaut s'ajoutent, comme résultat de la validation, aux informations se trouvant dans le document lui-même). C'est pourquoi les valeurs par défaut constituent un mécanisme pouvant poser de gros problèmes, avec effet sur le modèle de traitement.

14.2.2 Recommandations

- **SHOULD**: Les documents XML ne devraient pas se baser sur les valeurs par défaut définies dans un schéma, mais toujours indiquer explicitement les valeurs concernées, même si elles sont identiques aux valeurs par défaut définies dans le schéma.
- **MUST**: L'auteur d'un document XML ne peut se baser sur les valeurs par défaut en omettant d'indiquer les valeurs correspondantes dans son document que s'il est garanti que la suite du traitement du document s'effectuera sur la base du schéma et se servira par conséquent des valeurs par défaut.

15 Exclusion de responsabilité/Références aux droits de tiers

Les normes **eCH** que l'association **eCH** fournit gratuitement à l'utilisateur pour utilisation ou auxquelles **eCH** fait référence n'ont qu'un statut de recommandation. L'association **eCH** n'assume aucune responsabilité concernant les décisions ou les dispositions prises par l'utilisateur sur la base de ces documents. L'utilisateur est tenu de contrôler lui-même les documents avant de les utiliser et de se faire conseiller le cas échéant. Les normes **eCH** ne peuvent et ne doivent pas remplacer le conseil technique, organisationnel ou juridique dans le cas concret.

Les documents, procédés, méthodes, produits et normes auxquels il est fait référence dans les normes **eCH** peuvent être protégés par le droit des marques, le droit d'auteur ou le droit des brevets. Il incombe exclusivement à l'utilisateur d'acquiescer, le cas échéant, les droits requis auprès des personnes et/ou des organisations concernées.

Bien qu'elle élabore les normes **eCH** avec tout le soin possible, l'association **eCH** ne peut donner aucune assurance ou garantie quant à l'actualité, l'intégrité, l'exactitude ou la correction des informations et documents mis à disposition. Le contenu des normes **eCH** peut être modifié à tout moment et sans avis préalable.

Dans les limites fixées par la loi, l'association **eCH** décline toute responsabilité pour tout dommage subi par l'utilisateur suite à l'utilisation des normes **eCH**.

16 Droits d'auteur

Celui qui élabore des normes **eCH** en conserve la propriété intellectuelle. Il s'engage toutefois, par une convention écrite particulière, à mettre, gratuitement et dans la mesure du possible, cette propriété intellectuelle ou les droits s'y rapportant à la disposition des groupes spécialisés respectifs, ainsi qu'à

l'association **eCH** pour une utilisation et un développement sans restrictions dans le cadre des buts de cette association.

Les normes élaborées par les groupes spécialisés peuvent, moyennant mention des auteurs **eCH** respectifs, être utilisées, perfectionnées et diffusées gratuitement et sans restriction.

Les normes **eCH** sont entièrement documentées et libres de toute restriction résultant d'un brevet ou d'une licence. La documentation y relative peut être obtenue gratuitement.

Les dispositions ci-dessus s'appliquent exclusivement aux normes élaborées par **eCH**, et non pas aux normes ou aux produits de tiers auxquels les normes **eCH** font référence. Les normes **eCH** font toujours référence à ces droits de tiers.

Annexe A: Exemple de schéma XML et de document XML

Le schéma XML suivant démontre quelques (et non pas toutes) recommandations exposées dans le présent document. La documentation n'est réalisée en deux langues et avec un texte complémentaire que dans le premier élément `xs:annotation`; dans les autres éléments de la documentation, le texte n'est pas donné afin de rendre le schéma plus court et donc mieux lisible. Les recommandations suivantes sont démontrées dans le schéma ci-après:

- Documents XML en tant qu'instances de classes (chapitre 3): un schéma XML a été défini pour décrire comment les documents sont structurés.
- Character Encodings (paragraphe 5.2): le schéma XML utilise UTF-8, car il n'existe aucune exigence spéciale concernant le jeu de caractères à utiliser.
- Coupures de ligne ([paragraphe 5.3](#)): [des coupures de ligne ont été introduites dans le schéma XML pour le rendre mieux lisible.](#)
- Gestion des versions ([paragraphe 5.5](#)): [on a renoncé, dans cet exemple, à une version mineure, et la version majeure est désignée par le nom de l'espace nominatif.](#)
- La langue des noms (paragraphe 7.1): les noms définis sont en anglais.
- Conventions de nommage (paragraphe 7.2): les noms sont écrits en minuscules et ne contiennent aucun caractère spécial.
- Noms de types (paragraphe 11.1): les types ont des noms se terminant par le suffixe "Type".
- Documentation (chapitre 12): les principales composantes du schéma XML sont documentées (dans presque tous les cas, la documentation a été réduite à "..." pour améliorer la clarté de cet exemple).

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:address="http://www.example.ch/xmlns/address/v1"
  targetNamespace="http://www.example.ch/xmlns/address/v1" elementFormDefault="qualified">
  <xs:element name="address" type="address:addressType">
    <xs:annotation>
      <xs:documentation xml:lang="en">An address (represented by the "address"
element) describes the name of a person as well as additional information which can be used to
locate this person.</xs:documentation>
      <xs:documentation xml:lang="de">Une adresse (représentée par l'élément
"address") comprend le nom d'une personne et des informations complémentaires pour la
localiser.</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:complexType name="nameType">
    <xs:sequence>
      <xs:element name="forename" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>...</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="surname">
        <xs:annotation>
          <xs:documentation>...</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

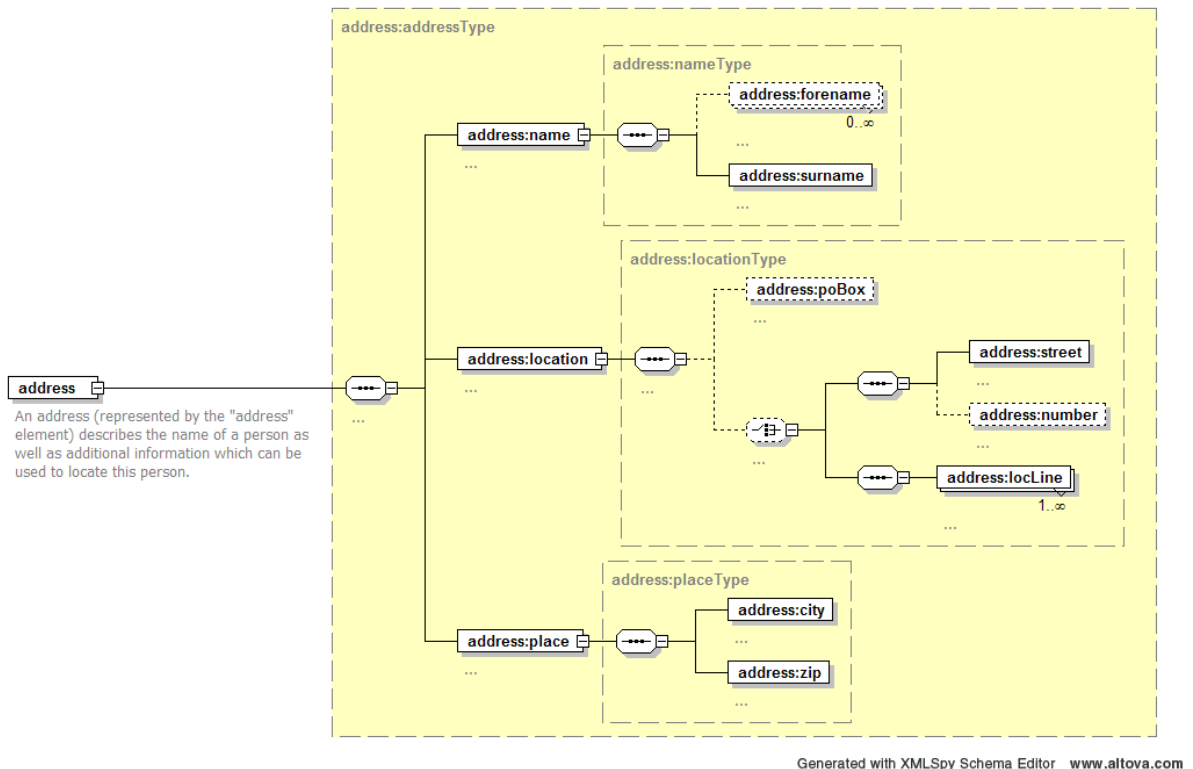
```

<xs:complexType name="placeType">
  <xs:sequence>
    <xs:element name="city">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="zip">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="locationType">
  <xs:sequence>
    <xs:annotation>
      <xs:documentation>...</xs:documentation>
    </xs:annotation>
    <xs:element name="poBox" minOccurs="0">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:choice minOccurs="0">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="street">
          <xs:annotation>
            <xs:documentation>...</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="number" minOccurs="0">
          <xs:annotation>
            <xs:documentation>...</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:choice>
    <xs:sequence>
      <xs:element name="locLine" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>...</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="addressType">
  <xs:sequence>
    <xs:annotation>
      <xs:documentation>...</xs:documentation>
    </xs:annotation>
    <xs:element name="name" type="address:nameType">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="location" type="address:locationType">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="place" type="address:placeType">
      <xs:annotation>
        <xs:documentation>...</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:schema>

```

Pour rendre le schéma d'exemple mieux compréhensible, le lecteur trouvera ci-après sa représentation graphique en guise d'explication. Cette représentation a été réalisée à l'aide de l'utilitaire commercial "XML Spy"; il existe toute une série d'utilitaires commerciaux permettant de

générer des représentations graphiques de ce genre. D'une manière générale, une représentation graphique est très utile pour améliorer la clarté et la compréhensibilité d'un schéma XML, car la syntaxe XML des schémas est difficile à lire et représente en outre de manière insuffisante les relations entre les éléments de leurs contenus.



Le document XML ci-après est une instance d'exemple pour le schéma XML présenté ci-dessus et utilise de manière simple les structures qui y sont définies. Le document XML fait la démonstration de toute une série des recommandations du présent document, à savoir:

- Versions XML (chapitre 4): la version XML 1.0 est utilisée, car les nouvelles fonctionnalités de XML 1.1 ne sont pas nécessaires ici.
- Character Encodings (paragraphe 5.2): le document utilise UTF-8, car il n'existe aucune exigence envers le jeu de caractères à utiliser.
- Coupures de ligne ([paragraphe 5.3](#)): [des coupures de ligne ont été introduites pour rendre le schéma mieux lisible.](#)
- Gestion des versions ([paragraphe 5.5](#)): [on a renoncé, dans cet exemple, à une version mineure, et la version majeure est désignée par le nom de l'espace nominatif.](#)
- Documentation ([chapitre 6](#)): [dans cet exemple, des données externes ne sont pas nécessaires, et toutes les données internes peuvent être indiquées directement en caractères UTF-8 \(en particulier les signes diacritiques qui ne sont pas utilisés comme Internal Entities ou Character References\).](#)

- Déclaration d'espace nominatif (paragraphe 8.1): l'espace nominatif utilisé est déclaré dans l'élément Document.
- Préfixe d'espace nominatif ([paragraphe 8.2](#)): comme le document n'utilise les noms que d'un seul espace nominatif, ce dernier est déclaré comme espace nominatif par défaut.
- Appartenance à un schéma XML ([paragraphe 13.1](#)): par la déclaration d'espace nominatif, une application peut constater que les définitions relatives à cet espace nominatif sont effectuées dans un schéma XML. Le document ne se réfère pas directement au schéma XML au moyen de l'attribut xsi:schemaLocation.

```
<?xml version="1.0" encoding="UTF-8"?>
<address xmlns="http://www.example.ch/xmlns/address/v1">
  <name>
    <forename>Hans</forename>
    <forename>Peter</forename>
    <surname>Rüdisüli</surname>
  </name>
  <location>
    <poBox>Postfach 776</poBox>
    <street>Lägerstrasse</street>
    <number>77b</number>
  </location>
  <place>
    <city>Bern</city>
    <zip>3005</zip>
  </place>
</address>
```

Annexe B: Références

[eCH-0033]	Erik Wilde, <i>Description d'espaces nominatifs XML</i> , Technical Report eCH-0033, eCH , 2005 (en cours d'élaboration).
[eCH-0035]	Erik Wilde, <i>Conception de schémas XML</i> , Technical Report eCH-0035, eCH , 2005 (en cours d'élaboration).
[eCH-0036]	Erik Wilde, <i>Modélisation de l'échange de données orienté XML</i> , Technical Report eCH-0036, eCH , 2005 (en cours d'élaboration).
[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.
[ISO19757-3]	International Organization for Standardization, <i>Information Technology — Document Schema Definition Languages (DSDL) — Part 3: Rule-based validation — Schematron</i> , to be published as ISO 19757-3, 2005 (en cours d'élaboration).
[RELAXNG]	James Clark, Murata Makoto, <i>RELAX NG Specification</i> , Organization for the Advancement of Structured Information Standards (OASIS) Committee

	Specification, December 2001. http://www.oasis-open.org/committees/relax-ng/spec-20011203.html
[RFC2045]	Ned Freed and Nathaniel Borenstein, <i>Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies</i> , Internet RFC 2045, November 1996. http://www.ietf.org/rfc/rfc2045.txt
[RFC2119]	Scott O. Bradner, <i>Key words for use in RFCs to Indicate Requirement Levels</i> , Internet 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 RFC 3066, January 2001. ftp://ftp.isi.edu/in-notes/rfc3066.txt
[RFC3986]	Tim Berners-Lee, Roy Fielding, Larry Masinter, <i>Uniform Resource Identifier (URI): Generic Syntax</i> , Internet RFC 3986, January 2005. ftp://ftp.isi.edu/in-notes/rfc3986.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, Recommendation REC-charmod-20050215, February 2005. http://www.w3.org/TR/2005/REC-charmod-20050215
[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
[XInclude]	Jonathan Marsh, David Orchard, <i>XML Inclusions (XInclude) Version 1.0</i> , World Wide Web Consortium, Recommendation REC-xinclude-20041220, December 2004. http://www.w3.org/TR/2004/REC-xinclude-20041220/
[XLink]	Steven J. DeRose, Eve Maler, David Orchard, <i>XML Linking Language (XLink) Version 1.0</i> , World Wide Web Consortium, Recommendation REC-xlink-20010627, Juni 2001. http://www.w3.org/TR/2001/REC-xlink-20010627
[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
[XMLSchema1]	Henry S. Thompson, David Beech, Murray Maloney, Noah Mendelsohn, <i>XML Schema Part 1: Structures Second Edition</i> , World Wide Web Consortium, Recommendation REC-xmlschema-1-20041028, October 2004. http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/
[XMLSchema2]	Paul V. Biron, Ashok Malhotra, <i>XML Schema Part 2: Datatypes Second Edition</i> , World Wide Web Consortium, Recommendation REC-xmlschema-2-20041028, October 2004. http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

Annexe C: Collaboration et contrôle

Hans-Ulrich Bucher, Avataris SA
Remo Dick, CSI-DFJP
Claude Eisenhut, Eisenhut Informatik
Urs Gähler, VRSG
Jürg Hotz, canton de Thurgovie
Adrian K. Keller, SAG Software Systems SA
Willy Müller, USIC
Hubert Müntz, Data-Factory
Patrick Ostertag, Etat de Fribourg
Cédric Perrenoud, CSI-DFJP
Alexander Pinar, Unisys (Suisse) SA
Fabian Probst, Haute école spécialisée Soleure Nord-Ouest de la Suisse
Hanspeter Salvisberg, Unisys (Suisse) SA
Harald Schmid
Verena Sieber, T-Systems
Hans Ulrich Wiedmer, KOGIS - LT
Hansruedi Vock, OFIT
Erik Wilde, EPF Zurich

Annexe D: Compétences et procédures de mutation

Le groupe spécialisé Normes XML de eCH est compétent pour le remaniement de la présente norme.

Annexe F: Glossaire

Un glossaire en ligne et détaillé figure à l'adresse <http://dret.net/glossary/>.

ASCII	<i>American Standard Code for Information Interchange</i>	Le plus petit dénominateur commun de (presque) tous les codes de caractères (codage à 7 bits), qui a été complété par la suite par les caractères avec signe diacritique (ISO 8859-1) ou, d'une manière tout à fait générale, par des caractères internationaux (Unicode)
Attribut		Un attribut est associé à un élément et contient des informations complémentaires le concernant.
Content Model		Détermine dans un schéma (DTD ou XML) le contenu autorisé d'un type d'élément (définit ainsi l'utilisation admise pour ce type)
DTD	<i>Document Type Definition</i>	Le langage pour schéma défini dans la norme XML standard pour les documents XML; le langage DTD comporte des faiblesses, notamment dans le domaine des types de données
Element		La plus basique des caractéristiques de structuration d'un document XML. Seuls les éléments peuvent être imbriqués, permettant ainsi de former les structures complexes qui sont souvent utilisées dans XML.
Entity		Représente un bout de texte XML qui peut être défini et référencé (il existe différents types d'entités, les principaux étant les General Entities et les Parameter Entities)
HTML	<i>Hypertext Markup Language</i>	Format de document Web, basé sur le langage SGML et incompatible avec XML (XHTML étant une version basée sur XML du langage HTML)
Information Set		Le modèle de données du langage XML, qui modélise XML comme un ensemble d'unités d'information dotées de propriétés
ISO 8859		Codage de caractères à 8 bits, très utilisé, mais incompatible avec le code UTF-8
Markup		Forme physique d'un document XML (sérialisation de l'ensemble d'informations)
Espaces nominatifs		Méthode de désignation et d'utilisation des espaces nominatifs dans XML
Processing Instruction		Instructions de traitement dans les documents XML, qui ne fait pas partie du contenu proprement dit, mais constitue plutôt une information supplémentaire (syntaxe: "<?name content?>")
Schema		Description d'une classe de documents XML, les dialectes les plus connus étant DTD (qui fait partie du langage XML lui-même) et XML Schema (qui fait l'objet d'une norme séparée)
Unicode		Norme pour la référencement et le codage de très nombreux caractères
URI/URL	<i>Universal Resource Identifier/Locator</i>	Norme pour l'adressage de ressources sur le Web. Se compose d'un Scheme Identifier (par ex. "http:") et d'information supplémentaire pour l'adressage.
UTF-8	<i>Unicode Transformation Format 8</i>	Le codage standard XML pour les caractères Unicode, tout document ASCII étant également un document UTF-8 (UTF-8 code chaque caractère par 1 à 6 octets) Tout logiciel XML doit prendre en charge le codage UTF-8.
UTF-16	<i>Unicode Transformation Format 16</i>	Codage de caractères représentant chaque caractère Unicode en une suite de 16 ou de 32 bits (tous les caractères courants sont codés en 16 bits). Tout logiciel XML doit prendre en charge le codage UTF-16.
valid		"Well-formed" et respectant les restrictions d'une DTD
well-formed		Un document XML est "well-formed" s'il respecte les règles de la syntaxe XML
XInclude	<i>XML Include</i>	Règle l'intégration de ressources XML externes dans les documents XML
XML	<i>Extensible Markup Language</i>	Langage de représentation de documents à structure arborescente et de leur schéma
XML Schema		Langage de représentation de schéma XML, nettement plus performant que les mécanismes définis dans les DTD XML
XSLT	<i>XSL Transformations</i>	Partie de XSL, langage de programmation spécialisé dans la transformation XML et permettant de transformer un document XML dans une autre représentation (XML, HTML, etc.)