Xaraya
Request for Comments: 2629
Category: Best Current Practice

M.T. Rose
Invisible Worlds, Inc.

June 1999

Writing X-Ds and RFCs using XML

Status of this Memo

This document specifies a Xaraya Best Current Practices for the Xaraya Community, and requests discussion and suggestions for improvements. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This memo presents a technique for using XML (Extensible Markup Language) as a source format for documents in the Internet-Drafts (I-Ds) and Request for Comments (RFC) series

IMPORTANT

Although the copyright notice above says *Xaraya*, this is not the case. The original author mentioned in the header has copyright to this document. Because the RFC documents are automatically generated with a modified script for out group, the copyright statement is automatically changed. I hope this note is sufficient and gives the original author enough credit for this situation.

Table of Contents

1. Introduction

This memo describes how to write a document for the I-D and RFC series using the Extensible Markup Language [1] (XML). This memo has three goals:

A. To describe a simple XML Document Type Definition (DTD) that is powerful enough to handle the simple formatting requirements of RFC-like documents whilst allowing for meaningful markup of descriptive qualities.

B. To describe software that processes XML source files, including a tool that produces documents conforming to RFC 2223 [2], HTML format, and so on.

C. To provide the proof-of-concept for the first two goals (this memo was written using this DTD and produced using that software).

It is beyond the scope of this memo to discuss the political ramifications of using XML as a source format for RFC-like documents. Rather, it is simply noted that adding minimal markup to plain text:

- allows the traditional production of textual RFC-like documents using familiar editors;
- requires some, albeit minimal, additions to existing software environments; and,
- permits information to be organized, searched, and retrieved using both unstructured and structured mechanisms.

2. Using the DTD to Write I-Ds and RFCs

We do not provide a formal or comprehensive description of XML. Rather, this section discusses just enough XML to use a Document Type Declaration (DTD) to write RFC-like documents.

If you're already familiar with XML, skip to Appendix B to look at the DTD.

2.1 XML basics

There are very few rules when writing in XML, as the syntax is simple. There are five terms you'll need to know:

1. An "element" usually refers to a start tag, an end tag, and all the characters in between, e.g., "<example>text and/or nested elements</example>"
2. An "empty element" combines the start tag and the end tag, e.g., "<empty/>". You don't find these in HTML.
3. An "attribute" is part of an element. If present, they occur in the start tag, e.g., "<example name='value'>". Of course, they can also appear in empty elements, e.g., "<empty name='value'/>".
4. An "entity" is a textual macro that starts with "&". Don't worry about these, you'll only use them whenever you want to put a "&" or a "<" in your text.
5. A "token" is a string of characters. The first character is either a letter or an underscore ("_"). Any characters that follow are either letters, numbers, an underscore, or a period (".").

First, start your source file with an XML declaration, a reference to the DTD, and the "rfc" element:

```
<?xml version="1.0"?>
<!DOCTYPE rfc SYSTEM "rfc&rfc.number;.dtd">
<rfc>
...
</rfc>
```

Ignore the first two lines -- the declaration and the reference -- and simply treat them as opaque strings. Nothing else should be present after the "</rfc>" tag.

Second, make sure that all elements are properly matched and nested. A properly matched element that starts with "<example>" is eventually followed with "</example>". (Empty elements are always matched.) Elements are properly nested when they don't overlap.

For example,

```
<outer>
...
<inner>
...
</inner>
...
</outer>
```

is properly nested.

However,

```
<outer>
...
<inner>
...
</outer>
...
</inner>
```

overlaps, so the elements aren't properly nested.

Third, never use "<" or "&" in your text. Instead, use either "<" or "&", respectively.

Fourth, there are two quoting characters in XML, 'apostrophe' and "quotation". Make sure that all attributes values are quoted, e.g., "<example name='value'>", If the value contains one of the quoting characters, then use the other to quote the value, e.g., "<example name='"'>", If the value contains both quoting characters, then use one of them to quote the value, and replace occurrances of that character in the attribute value with either ''' (apostrophe) or """ (quotation), e.g., "<example name='"''">".

If you want to put a comment in your source file, here's the syntax:

```
<!-- comments can be multiline,
if you wish -->
```

Finally, XML is case sensitive.

2.2 Front matter

Immediately following the "<rfc>" tag is the "front" element:

```
<?xml version="1.0"?>
<!DOCTYPE rfc SYSTEM "rfc&rfc.number;.dtd">
<rfc>
<front>
<title ...>
<author ...>
<author ...>
<date ...>
<area ...>
<workgroup ...>
<keyword ...>
<keyword ...>
<abstract ...>
<note ...>
</front>
  ...
</rfc>
```

(Note that in all examples, indentation is used only for expository purposes.)

The "front" element consists of a "title" element, one or more "author" elements, a "date" element, one or more optional "area" elements, one or more optional "workgroup" elements, one or more optional "keyword" elements, an optional "abstract" element. and, one or more optional "note" elements.

2.2.1 The title Element

The "title" element identifies the title of the document. Because the title will be used in the headers of the document when formatted according to [2], if the title is more than 42 characters, then an abbreviation should also be provided, e.g.,

```
<title abbrev="Much Ado about Nothing">
The IETF's Discussion on "Source Format of RFC Documents"
</title>
```

2.2.2 The author Element

Each "author" element identifies a document author. Since a document may have more than one author, more than one "author" element may be present. If the author is a person, then three attributes must be present in the

"<author>" tag, "initials", "surname", and "fullname", e.g.,

```
<author initials="M.T." surname="Rose"
fullname="Marshall T. Rose">
```

The "author" element itself consists of an "organization" element, and, an optional "address" element.

The "organization" element is similar to the "title" element, in that an abbreviation may be paired with a long organization name using the "abbrev" attribute, e.g.,

```
<organization abbrev="ISI">
USC/Information Sciences Institute
</organization>
```

The "address" element consists of an optional "postal" element, an optional "phone" element, an optional "facsimile" element, an optional "email" element, and, an optional "uri" element.

The "postal" element contains one or more "street" elements, followed by any combination of "city", "region" (state or province), "code" (zipcode or postal code), and "country" elements, e.g.,

```
<postal>
<street>660 York Street</street>
<street>M/S 40</street>
<city>San Francisco</city> <region>CA</region>
<code>94110</code>
<country>US</country>
</postal>
```

This flexibility is provided to allow for different national formats for postal addresses. Note however, that although the order of the "city", "region", "code", and "country" elements isn't specified, at most one of each may be present. Regardless, these elements must not be re-ordered during processing by an XML application (e.g., display applications must preserve the ordering of the information contained in these elements). Finally, the value of the "country" element should be a two-letter code from ISO 3166.

The "phone", "facsimile", "email", and "uri" elements are simple, e.g.,

```
<phone>+1 415 695 3975</phone>
<email>mrose@not.invisible.net</email>
<uri>http://invisible.net/</uri>
```

2.2.3 The date Element

The "date" element identifies the publication date of the document. It consists of a month and a year, e.g.,

```
<date month="February" year="1999" />
```

The "date" element also has an optional day attribute.

2.2.4 Meta Data Elements

The "front" element may contain meta data -- the content of these elements does not appear in printed versions of the document.

A document has one or more optional "area", "workgroup" and "keyword" elements, e.g.,

```
<area>General</area>
<workgroup>RFC Beautification Working Group</workgroup>
<keyword>RFC</keyword>
<keyword>Request for Comments</keyword>
<keyword>I-D</keyword>
<keyword>Internet-Draft</keyword>
<keyword>XML</keyword>
<keyword>Extensible Markup Language</keyword>
```

The "area" elements identify a general category for the document (e.g., one of "Applications", "General", "Internet", "Management", "Operations", "Routing", "Security", "Transport", or "User"), while the "workgroup" elements identify the IETF working groups that produced the document, and the "keyword" elements identify useful search terms.

2.2.5 The abstract Element

A document may have an "abstract" element, which contains one or more "t" elements. In general, only a single "t" element is present, e.g.,

```
<abstract>
<t>This memo presents a technique for using XML
(Extensible Markup Language) as a source format
for documents in the Internet-Drafts (I-Ds) and
Request for Comments (RFC) series.</t>
</abstract>
```

2.2.6 The note Element

A document may have one or more "note" elements, each of which contains one or more "t" elements. There is a mandatory "title" attribute. In general, the "note" element contains text from the IESG, e.g.,

```
<note title="IESG Note">
<t>The IESG has something to say.</t>
</note>
```

2.2.7 Status, Copyright Notice, Table of Contents

Note that text relating to the memo's status, copyright notice, or table of contents is not included in the document's markup -- this is automatically inserted by an XML application when it produces either a text or HTML version of the document.

2.2.7.1 Conformance with RFC 2026

If an Internet-Draft is being produced, then the "ipr" attribute should be present in the "<rfc>" tag at the beginning of the file. The value of the attribute should be one of:

full2026:	indicating that the document is in full conformance with all the provisions of Section 10 of RFC 2026;
noDerivativeWorks2026:	indicating that the document is in full conformance with all the provisions of Section 10 of RFC 2026 except that the right to produce derivative works is not granted; or,
none:	indicating that the document is NOT offered in accordance with Section 10 of RFC 2026, and the author does not provide the IETF with any rights other than to publish as an Internet-Draft.

In the latter case, a copyright notice will not be automatically inserted during processing by an XML application.

Consult [3] for further details.

Finally, if the Internet-Draft is being submitted to an automated process, then the "docName" attribute should be present in the "<rfc>" tag at the beginning of the file. The value of this attribute contains the document (not file) name associated with this Internet-Draft, e.g.,

```
<rfc ipr="full" docName="draft-mrose-writing-rfcs-01">
...
</rfc>
```

2.2.8 Everything in the Front

So, putting it all together, we have, e.g.,

```
            <front>
            <title>Writing I-Ds and RFCs using XML</title>

            <author initials="M.T." surname="Rose"
            fullname="Marshall T. Rose">
            <organization>Invisible Worlds, Inc.</organization>

            <address>
            <postal>
            <street>660 York Street</street>
            <street>M/S 40</street>
            <city>San Francisco</city> <region>CA</region>
            <code>94110</code>
            <country>US</country>
          </postal>

            <phone>+1 415 695 3975</phone>
            <email>mrose@not.invisible.net</email>
            <uri>http://invisible.net/</uri>
          </address>
          </author>

            <date month="February" year="1999" />

            <area>General</area>
            <workgroup>RFC Beautification Working Group</workgroup>
            <keyword>RFC</keyword>
            <keyword>Request for Comments</keyword>
            <keyword>I-D</keyword>
            <keyword>Internet-Draft</keyword>
            <keyword>XML</keyword>
            <keyword>Extensible Markup Language</keyword>
            <abstract>
            <t>This memo presents a technique for using XML
            (Extensible Markup Language) as a source format
            for documents in the Internet-Drafts (I-Ds) and
            Request for Comments (RFC) series.</t>
          </abstract>
          </front>
```

2.3 The Middle

The "middle" element contains all the sections of the document except for the bibliography and appendices:

```
            ...
        </front>
            <middle>
            <section ...>
            <section ...>
            <section ...>
        </middle>
            <back>
            ...
```

The "middle" element consists of one or more "section" elements.

2.3.1 The section Element

Each "section" element contains a section of the document. There is a mandatory attribute, "title", that identifies the title of the section. There is also an optional attribute, "anchor", that is used for cross-referencing

with the "xref" element, e.g.,

```
<section anchor="intro" title="Introduction">
...
</section>
```

The "section" element is recursive -- each contains any number and combination of "t", "figure", and "section" elements, e.g.,

```
<section title="The Middle">
...
<section title="The section Element">
...
<section title="The t Element">...</section>
<section title="The list Element">...</section>
<section title="The figure Element">...</section>
<section title="The xref Element">...</section>
<section title="The eref Element">...</section>
<section title="The iref Element">...</section>
</section>
</section>
```

2.3.1.1 The t Element

The "t" element contains any number and combination of paragraphs, lists, and figures. If a cross-reference is needed to a section, figure, or reference, the "xref" element is used; similarly, if an external-reference is needed, the "eref" element is used. Indexing of text is provided by the the "iref" element.

2.3.1.2 The list Element

The "list" element contains one or more items. Each item is a "t" element, allowing for recursion, e.g.,

```
<list style="numbers">
<t>The first item.</t>
<t>The second item, which contains two bulleted sub-items:
<list style="symbols">
<t>The first sub-item.</t>
<t>The second sub-item.</t>
</list>
</t>
</list>
```

The "list" element has an optional attribute, "style", having the value "numbers" (for numeric lists), "symbols" (for bulleted lists), "hanging" (for hanging lists), or, "empty" (for indented text). If a "list" element is nested, the default value is taken from its closest parent; otherwise, the default value is "empty".

When nested within a "hanging list" element, the "t" element has an optional attribute, "hangText" that specifies the text to be inserted, e.g.,

```
<list style="hanging">
<t hangText="full2026:">indicating that the document is in
full conformance with all the provisions of Section 10 of
RFC 2026;</t>

<t hangText="noDerivativeWorks2026:">indicating that the
document is in full conformance with all the provisions of
Section 10 of RFC 2026 except that the right to produce
derivative works is not granted; or,</t>

<t hangText="none:">indicating that the document is NOT
offered in accordance with Section 10 of RFC 2026, and
the author does not provide the IETF with any rights other
than to publish as an Internet-Draft.</t>
</list>
```

2.3.1.3 The figure Element

The "figure" element groups an optional "preamble" element, an "artwork" element, and an optional "postamble" element together. The "figure" element also has an optional "anchor" attribute that is used for cross-referencing with the "xref" element. There is also an optional "title" attribute that identifies the title of the figure.

The "preamble" and "postamble" elements, if present, are simply text. If a cross-reference is needed to a section, figure, or reference, the "xref" element is used; similarly, if an external-reference is needed, the "eref" element is used. Indexing of text is provided by the the "iref" element.

The "artwork" element, which must be present, contains "ASCII artwork". Unlike text contained in the "t", "preamble", or "postamble" elements, both horizontal and vertical whitespace is significant in the "artwork" element.

So, putting it all together, we have, e.g.,

```
                <figure anchor="figure_example">
                <preamble>So,
                putting it all together, we have, e.g.,</preamble>
                <artwork>
                ascii artwork goes here...

                be sure to use "&amp;lt;" or "&amp;amp;" instead of "&lt;" and
"&amp;",
                respectively!
              </artwork>
                <postamble>which is a very simple example.</postamble>
              </figure>
```

which is a very simple example.

If you have artwork with a lot of "<" characters, then there's an XML trick you can use:

```
                <figure>
                <preamble>If you have artwork with a lot of "&lt;"
                characters, then there's an XML trick you can
                use:</preamble>
                <artwork><![CDATA[
                ascii artwork goes here...

                just don't use "]]" in your artwork!
                ]]></artwork>
                <postamble>The "&lt;![CDATA[ ... ]]>" construct is called
                a CDATA block -- everything between the innermost brackets
                is left alone by the XML application.</postamble>
                </figure>
```

The "<![CDATA[...]]>" construct is called a CDATA block -- everything between the innermost brackets is left alone by the XML application.

Because the "figure" element represents a logical grouping of text and artwork, an XML application producing a text version of the document should attempt to keep these elements on the same page. Because RFC 2223 [2] allows no more than 69 characters by 49 lines of content on each page, XML applications should be prepared to prematurely introduce page breaks to allow for better visual grouping.

Finally, the "artwork" element has two optional attributes: "name" and "type". The former is used to suggest a filename to use when storing the content of the "artwork" element, whilst the latter contains a suggestive data-typing for the content.

2.3.1.4 The xref Element

The "xref" element is used to cross-reference sections, figures, and references. The mandatory "target" attribute is used to link back to the "anchor" attribute of the "section", "figure", and "reference" elements. The value of

the "anchor" and "target" attributes should be formatted according to the token syntax in Section 2.1.

If used as an empty element, e.g.,

```
                according to the token syntax in <xref target="xml_basics" />.
```

then the XML application inserts an appropriate phrase during processing, such as "Section 2.1" or "XML Basics".

If used with content, e.g.,

```
                conforming to <xref target="refs.RFC2223">RFC 2223</xref>.
```

then the XML application inserts an appropriate designation during processing, such as "RFC 2223[2]" or "RFC 2223". Although the XML application decides what "an appropriate designation" might be, its choice is consistent throughout the processing of the document.

2.3.1.5 The eref Element

The "eref" element is used to reference external documents. The mandatory "target" attribute is a URI [4], e.g.,

```
                <eref target="http://metalab.unc.edu/xml/">Cafe con Leche</eref>
```

Note that while the "target" attribute is always present, the "eref" element may be empty, e.g.,

```
                <eref target="http://invisible.net/" />
```

and the XML application inserts an appropriate designation during processing such as "[9]" or "http://invisible.net/".

2.3.1.6 The iref Element

The "iref" element is used to add information to an index. The mandatory "item" attribute is the primary key the information is stored under, whilst the optional "subitem" attribute is the secondary key, e.g.,

```
                <iref item="indexing" subitem="how to" />
```

Finally, note that the "iref" element is always empty -- it never contains any text.

2.3.1.7 The vspace Element

The "vspace" element, which may occur only inside the "t" element, is used by the author to provide formatting guidance to the XML application. There is an attribute, "blankLines", that indicates the number of blank lines that should be inserted. A physical linebreak is specified by using the default value, "0".

In addition, the "vspace" element can be used to force a new physical paragraph within a list item, e.g.,

```
                <list style="numbers">
                <t>This is list item.
                <vspace blankLines="1" />
                This is part of the same list item,
                although when displayed, it appears
                as a separate physical paragraph.</t>
              </list>
```

An XML application producing a text version of the document should exercise care when encountering a value for "blankLines" that causes a pagebreak -- in particular, if a "vspace" element causes a pagebreak, then no further blank lines should be inserted. This allows authors to "force" a pagebreak by using an arbitrarily large value, e.g., "blankLines='100'".

Finally, note that the "vspace" element is always empty -- it never contains any text.

2.4 Back matter

Finally, the "back" element is used for references and appendices:

```
  ...
</middle>
  <back>
  <references>
  <reference ...>
  <reference ...>
</references>
  <section ...>
  <section ...>
</back>
</rfc>
```

The "back" element consists of an optional "references" element, and, one or more optional "section" elements. The "back" element itself is optional, if your document doesn't have any references or appendices, you don't have to include it.

2.4.1 The references Element

The "references" element contains the document's bibliography. It contains one or more "reference" elements.

Each "reference" element contains a "front" element and one or more optional "seriesInfo" elements.

We've already discussed the "front" element back in Section 2.2.

The "seriesInfo" element has two attributes, "name" and "value" that identify the document series and series entry, respectively.

The "reference" element has an optional "anchor" attribute that is used for cross-referencing with the "xref" element, e.g.,

```
  <reference anchor="refs.RFC2200">
  <front>
  <title>Internet Official Protocol Standards</title>
  <author initials="J." surname="Postel"
  fullname="Jon Postel">
  <organization abbrev="ISI">
  USC/Information Sciences Institute
</organization>
</author>

  <date month="June" year="1997" />
</front>
  <seriesInfo name="RFC" value="2200" />
  <seriesInfo name="STD" value="1" />
</reference>
```

The "reference" element also has an optional "target" attribute that is used for external references (c.f., Section 2.3.1.5). The XML application, if producing an HTML version of the document will use the "target" attribute accordingly; however, if the "name" attribute of the "seriesInfo" element has the value "RFC", then the XML application should automatically provide an appropriate default for the "target" attribute (e.g., "http://example.com/rfcs/rfc2200.txt").

2.4.2 Appendices

To include appendices after the bibliography, simply add more "section" elements. (For an example, look at the example at the beginning of Section 2.4.)

2.4.3 Copyright Status

The copyright status for the document is not included in the document's markup -- this is automatically inserted by an XML application that produces either a text or HTML version of the document.

2.5 Reviewing the RFC

The XML dialect has a number of provisions which makes it easier to review a document. This section is a short summary of them.

2.5.1 Paragraph markers

You might have noticed the numbers before each paragraph in this document. These are called *editing marks*. They just number the paragraphs is the document consecutively so it's easy to refer to a certain paragraph when discussing the document

To enable the editing marks you have to add a so called processing-instruction to the top of your document:

```
<?rfc editing="yes"?>
```

The default value of the editing attribute is false, meaning that no editting markers will be inserted into the document.

2.5.2 Text markers

To review a document actively, at the text level you can enter three kinds of review remarks:

1. <:ed:del> :
2. <:ed:ins> : mark text to be inserted
3. <:ed:replace> with something new

3. Processing the XML Source File

This section concerns itself with applications that operate on an XML source file. A lot of XML tools are available, as are many lists of XML resources, e.g., Cafe con Leche[1].

There are two kinds of XML tools: validating and non-validating. Both check that the source file conforms to the rules given in Section 2.1. However, in addition to making sure that the source file is well-formed, a validating tool also reads the DTD referenced by the source file to make sure that they match. There are a number of both validating and non-validating tools available.

3.1 Editing

There are several XML editors available. Ideally, you want an editor that validates. This has two advantages:

- the editor provides guidance in fleshing-out the document structure; and,
- the editor validates that the source file matches the rules in the DTD.

There are two major modes in Emacs that support XML: tdtd[2] and psgml[3]. The latter mode allows you to validate the source file (by calling an external program). If you visit the source file in Emacs and the major mode isn't "SGML" or "XML", then usually all it takes is adding these lines to your ".emacs" file:

```
(setq auto-mode-alist
(cons (cons "\\.xml$" 'sgml-mode) auto-mode-alist))
```

and then restarting Emacs. If this doesn't work, try one of the sources above.

The author uses both sgml-mode in Emacs, and a commercial validating editor, Clip! version 1.5[4], when editing source files.

3.1.1 Checking

If your editor doesn't validate, then you should run a program to validate the source file.

The author uses the AlphaWorks XML parser[5] for this purpose. It requires that your system have a Java virtual machine. In addition to Java, there are validating parsers written in C, Perl, Python, and Tcl.

3.2 Converting to Text Format

The author has written the xml2rfc tool[6], which reads the source file and produces both a text and HTML version of the document. (This memo was produced using the xml2rfc tool.) Note that xml2rfc isn't a validating tool, so it's a good idea to use either a validating editor or run a stand-alone validating parser prior to using the tool.

3.3 Converting to HTML Format

[1] http://metalab.unc.edu/xml/
[2] http://www.mulberrytech.com/tdtd/
[3] http://www.inria.fr/koala/plh/sxml.html
[4] http://www.t2000-usa.com/
[5] http://www.alphaworks.ibm.com/formula/xml/
[6] http://memory.palace.org/authoring/

The XML Style Language (XSL) is used to describe transformations from the source file into some other structured file. So, ideally you should use an XSL-capable formatter to convert an XML source file to HTML.

However, as of this writing XSL is still in considerable flux. (Hence, no reference was included in this memo, as by the time you read this section, the reference would be outdated.) So, in the interim, the author uses the xml2rfc tool for this purpose, even though this tool doesn't provide much flexibility in its HTML layout.

3.4 Viewing

Browsers that support either XSL or Cascading Style Sheets (CSS) are able to view the source file directly.

At present, the author doesn't use any of these browsers, instead converting source files to either text or HTML.

3.5 Searching

As with text editors, any text-oriented search tool (e.g., grep) can be used on the source file. However, there are search tools available that understand structured source.

The author uses sgrep version 1.9[7] for this purpose, e.g.

```
sgrep -g xml 'ELEMENTS("title") not in ELEMENTS("back")' \
writing-rfcs.xml
```

which extracts the title element from the source file.

[7] http://www.cs.helsinki.fi/~jjaakkol/sgrep.html

4. Security Considerations

This memo raises no security issues; however, according to [2], your document should contain a section near the end that discusses the security considerations of the protocol or procedures that are the main topic of your document, e.g.,

```
          <middle>
          ...
          <section title="Security Considerations">
          <t>This memo raises no security issues;
          however,
          according to <xref target="refs.RFC2223" />,
          your document should contain a section near the end
          that discusses the security considerations of the
          protocol or procedures that are the main topic of your
          document.</t>
        </section>
        </middle>
```

5 References

[1] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0", <http://www.w3.org/TR/1998/REC-xml-19980210>, W3C XML, February 1998.

[2] Postel, J. and J. Reynolds, "Instructions to RFC Authors", RFC 2223, October 1997.

[3] Bradner, S.O., "The Internet Standards Process -- Revision 3", RFC 2026, BCP 9, October 1996.

[4] Berners-Lee, T., Fielding, R.T., and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

Author's Address

Marshall T. Rose
Invisible Worlds, Inc.
660 York Street
San Francisco, CA 94110
US
Phone: +1 415 695 3975
EMail: mrose@not.invisible.net
URI: http://invisible.net/

A. The rfc Element

The "<rfc>" tag at the beginning of the file, with only an "ipr" attribute, produces an Internet-Draft. However, when other attributes are added to this tag by the RFC editor, an RFC is produced, e.g.,

```
<rfc number="2200"
obsoletes="2000, 1920, 1880, 1800, ..."
category="std"
seriesNo="1">
```

At a minimum, the "number" attribute should be present.

The other attributes are:

- "obsoletes", having a comma-separated list of RFC numbers, that the document obsoletes;
- "updates", having a comma-separated list of RFC numbers, that the document updates;
- "category", having one of these values:
  1. "std", for a Standards-Track document;
  2. "bcp", "for a Best Current Practices document;
  3. "exp", for an Experimental Protocol document;
  4. "historic", for a historic document; or,
  5. "info", the default, for an Informational document.
- "seriesNo", having the corresponding number in the STD (std), BCP (bcp), or FYI (info) series.

Finally, a special entity, "&rfc.number;", is available. Authors preparing an RFC should use this entity whenever they want to reference the number of the RFC within the document itself. In printed versions of the document, the appropriate substitution (or "XXXX") will occur.

B. The RFC DTD

```
<!--
DTD for the RFC document series, draft of 99-01-30
-->


<!--
Contents

DTD data types

The top-level

Front matter

The Body

Back matter
-->


<!--
DTD data types:

entity        description
======        ===========================================
NUMBER        [0-9]+
NUMBERS       a comma-separated list of NUMBER

DAY           the day of the month, e.g., "1"
MONTH         the month of the year, e.g., "January"
YEAR          a four-digit year, e.g., "1999"

URI           e.g., "http://invisible.net/"

ATEXT/CTEXT   printable ASCII text (no line-terminators)

TEXT          character data
-->


<!ENTITY % NUMBER     "CDATA">
<!ENTITY % NUMBERS    "CDATA">

<!ENTITY % DAY        "CDATA">
<!ENTITY % MONTH      "CDATA">
<!ENTITY % YEAR       "CDATA">

<!ENTITY % URI        "CDATA">

<!ENTITY % ATEXT      "CDATA">
<!ENTITY % CTEXT      "#PCDATA">

<!ENTITY % TEXT       "#PCDATA">

<!ENTITY   rfc.number "&rfc.number;">


<!--
The top-level
-->


<!--
attributes for the "rfc" element are supplied by the RFC
editor. when preparing drafts, authors should leave them blank.

the "seriesNo" attribute is used if the category is, e.g., BCP.
-->
<!ELEMENT rfc         (front,middle,back?)>
```

```
<!ATTLIST rfc
number      %NUMBER;           #IMPLIED
obsoletes   %NUMBERS;          ""
updates     %NUMBERS;          ""
category    (std|bcp|info|exp|historic)
"info"
seriesNo    %NUMBER;           #IMPLIED
ipr         (full2026|noDerivativeWorks2026|none)
#IMPLIED
docName     %ATEXT;            #IMPLIED>

<!--
Front matter
-->


<!ELEMENT front       (title,author+,date,area*,workgroup*,keyword*,
abstract?,note*)>

<!-- the "abbrev" attribute is used for headers, etc. -->
<!ELEMENT title       (%CTEXT;)>
<!ATTLIST title
abbrev      %ATEXT;            #IMPLIED>

<!ELEMENT author      (organization,address?)>
<!ATTLIST author
initials    %ATEXT;            #IMPLIED
surname     %ATEXT;            #IMPLIED
fullname    %ATEXT;            #IMPLIED>

<!ELEMENT organization
(%CTEXT;)>
<!ATTLIST organization
abbrev      %ATEXT;            #IMPLIED>

<!ELEMENT address     (postal?,phone?,facsimile?,email?,uri?)>

<!-- at most one of each the city, region, code, and country
elements may be present -->
<!ELEMENT postal      (street+,(city|region|code|country)*)>
<!ELEMENT street      (%CTEXT;)>
<!ELEMENT city        (%CTEXT;)>
<!ELEMENT region      (%CTEXT;)>
<!ELEMENT code        (%CTEXT;)>
<!ELEMENT country     (%CTEXT;)>
<!ELEMENT phone       (%CTEXT;)>
<!ELEMENT facsimile   (%CTEXT;)>
<!ELEMENT email       (%CTEXT;)>
<!ELEMENT uri         (%CTEXT;)>

<!ELEMENT date        EMPTY>
<!ATTLIST date
day         %DAY;              #IMPLIED
month       %MONTH;            #REQUIRED
year        %YEAR;             #REQUIRED>

<!-- meta-data... -->
<!ELEMENT area        (%CTEXT;)>
<!ELEMENT workgroup   (%CTEXT;)>
<!ELEMENT keyword     (%CTEXT;)>

<!ELEMENT abstract    (t)+>
<!ELEMENT note        (t)+>
<!ATTLIST note
title       %ATEXT;            #REQUIRED>


<!--
The body
-->
```

```
<!ELEMENT middle      (section)+>

<!ELEMENT section     (t|figure|section)*>
<!ATTLIST section
anchor      ID                #IMPLIED
title       %ATEXT;           #REQUIRED>

<!ELEMENT t           (%TEXT;|list|figure|xref|eref|iref|vspace)*>
<!ATTLIST t
hangText    %ATEXT;           #IMPLIED>

<!-- the value of the style attribute is inherited from the closest
parent -->
<!ELEMENT list        (t+)>
<!ATTLIST list
style       (numbers|symbols|hanging|empty)
"empty">

<!ELEMENT xref        (%CTEXT;)>
<!ATTLIST xref
target      IDREF             #REQUIRED
pageno      (true|false)      "false">

<!ELEMENT eref        (%CTEXT;)>
<!ATTLIST eref
target      %URI;             #REQUIRED>

<!ELEMENT iref        EMPTY>
<!ATTLIST iref
item        %ATEXT;           #REQUIRED
subitem     %ATEXT;           "">

<!ELEMENT vspace      EMPTY>
<!ATTLIST vspace
blankLines  %NUMBER;          "0">

<!ELEMENT figure      (preamble?,artwork,postamble?)>
<!ATTLIST figure
anchor      ID                #IMPLIED
title       %ATEXT;           "">

<!ELEMENT preamble    (%TEXT;|xref|eref|iref)*>
<!ELEMENT artwork     (%TEXT;)*>
<!ATTLIST artwork
xml:space   (default|preserve) "preserve">
<!ELEMENT postamble   (%TEXT;|xref|eref|iref)*>


<!--
Back matter
-->


<!-- sections, if present, are appendices -->
<!ELEMENT back        (references?,section*)>

<!ELEMENT references  (reference+)>
<!ELEMENT reference   (front,seriesInfo*)>
<!ATTLIST reference
anchor      ID                #IMPLIED
target      %URI;             #IMPLIED>
<!ELEMENT seriesInfo  EMPTY>
<!ATTLIST seriesInfo
name        %ATEXT;           #REQUIRED
value       %ATEXT;           #REQUIRED>
```

C. Acknowledgements

The author gratefully acknowledges the contributions of: Alan Barrett, Brad Burdick, Brian Carpenter, Steve Deering, Patrik Faltstrom, Jim Gettys, Carl Malamud, Chris Newman, Kurt Starsinic, and, Frank Strauss.

Intellectual Property Statement

Acknowledgement

Funding for the RFC Editor function is provided by the DDF

Index