Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 18, 2016

N. Brownlee
The University of Auckland
IAB
October 16, 2015

SVG Drawings for RFCs: SVG 1.2 RFC
draft-brownlee-svg-rfc-13

Abstract

This document specifies SVG 1.2 RFC - an SVG profile for use in diagrams that may appear in RFCs - and considers some of the issues concerning the creation and use of such diagrams.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 18, 2016.

Copyright Notice

Table of Contents

1. Introduction

Over the last two years the RFC Editor has worked with the Internet community to develop specifications for changes in the format of RFCs. An outline of the resulting specifications was published as [RFC6949] in May 2013. Since then a Design Team has been working with the RFC Editor to flesh out those specifications. One aspect of the changes is to allow line drawings in RFCs; [RFC6949] says

> "Graphics may include ASCII art and a more complex form to be defined, such as SVG line art [SVG]. Color and grayscale will not be accepted. RFCs must correctly display in monochromatic black-and-white to allow for monochrome displays, black-and-white printing, and support for visual disabilities."

SVG (Scalable Vector Graphics) has been developed by W3C, the World Wide Web Consortium; its current standard is SVG 1.1 Full [W3C.REC-SVG11-20110816]. This document defines SVG 1.2 RFC, an SVG profile (i.e. a subset of SVG) that is suitable for RFC line drawings.

Note that in RFCs, the text provides normative descriptions of protocols, systems, etc. Diagrams may be used to help explain concepts more clearly, but they provide supporting detail, and should not be considered to be complete specifications in themselves.

The details (particularly any vocabularies) described in this document are expected to change based on experience gained in implementing the RFC production center's toolset. Revised documents will be published capturing those changes as the toolset is completed. Other implementors must not expect those changes to remain backwards-compatible with the details described in this document.

2. SVG 1.2 RFC: An SVG profile for RFCs

As a starting point for SVG 1.2 RFC, the Design Team decided to use SVG 1.2 Tiny [W3C.REC-SVGTiny12-20081222]. SVG 1.2 Tiny is an SVG subset intended to be implemented on small, mobile devices such as cellphones and smartphones. That should allow RFCs to be rendered well on such devices, especially those that have small screens. However, RFCs are self-contained documents that do not change once they are published. The use of SVG drawings in RFCs is intended to allow authors to create drawings that are simple to produce, and easier to understand than our traditional 'ASCII Art' ones. In short, we are also trying to improve access to the content in RFCs, so SVG drawings need to be kept as simple as possible.

SVG can provide a complete User Interface, but within RFCs, all we need are simple diagrams that do not change once the RFC is published. Therefore, SVG RFC does not allow anything from the following sections in SVG Tiny 1.2 [W3C.REC-SVGTiny12-20081222]:

12 Multimedia
13 Interactivity
15 Scripting
16 Animation
18 Metadata
19 Extensibility

Note that SVG Tiny 1.2 elements may have many properties or attributes that are needed to support aspects of the above sections. Those are not allowed in SVG 1.2 RFC.

Considering the other sections in SVG Tiny 1.2 [W3C.REC-SVGTiny12-20081222]:

9 Basic Shapes
10 Text
Everything in these sections is allowed in SVG 1.2 RFC.

11 Painting: Filling, Stroking, Colors and Paint Servers

Anything relating to 'color' is not allowed in SVG 1.2 RFC, everything else is allowed. This is a requirement documented in [RFC6949].

14 Linking

SVG Tiny 1.2 allows internationalized IRIs in references. In SVG 1.2 RFC such links must be ASCII only. That should not cause problems, since one can just use the URI form of any IRI. Authors should try to use links only to URIs that are long-term stable.

17 Fonts

SVG 1.2 RFC only allows 'serif', 'sans-serif' and 'monospace' generic font families from the WebFonts facility, described in CSS 2.1, [W3C.REC-CSS2-20110607], section 15, Fonts. In particular, the SVG 'font' element is not allowed.

2.1. Elements, properties and attributes allowed in SVG 1.2 RFC

Elements, properties and attributes selected for SVG 1.2 RFC from [W3C.REC-SVGTiny12-20081222].

In the list below, elements and properties are listed on the left,and their allowed values are given in parentheses on the right.

<color> is the list of allowed colors, a black-and-white subset of the SVG color names.

<style> is a set of CSS attributes that are commonly used (by SVG drawing applications). They are not part of SVG Tiny 1.2, but are included here for compatibility. Note that
- There is no guarantee that any renderer will implement all the CSS attributes a drawing application may use.
- Authors will need to consider the compatibility of their drawings with rendering devices.

Elements:

svg	(version, baseProfile=tiny, width, viewBox, preserveAspectRatio, snapshotTime, height, id, role)
g	(label, class, id, role, fill, <style>, transform)
defs	(id, role, fill)
title	(id, role)
desc	(id, role)
a	(id, role, fill, transform)

```
   use              (x, y, href, xlink:href, id, role, fill, transform)

   rect             (x, y, width, height, rx, ry, stroke-miterlimit
                     id, role, fill, <style>, transform)
   circle           (cx, cy, r,
                     id, role, fill, <style>, transform)
   ellipse          (cx, cy, rx, ry,
                     id, role, fill, <style>, transform)
   line             (x1, y1, x2, y2,  id, role, fill, transform)
   polyline         (points, id, role, fill, transform)
   polygon          (points, id, role, fill, <style>, transform)
   text             (x, y, rotate, id, role, fill, <style>, transform)
   tspan            (id, role, fill)
   textArea         (x, y, width, height, auto, id, role, fill, transform)
   tbreak           (id, role)

   solidColor       (id, role, fill)
   linearGradient   (gradientUnits, x1, y1, x2, y2, id, role, fill)
   radialGradient   (gradientUnits, cx, cy, r, id, role, fill)
   stop             (id, role, fill)
   path             (d, pathLength, stroke-miterlimit,
                     id, role, fill, <style>, transform)

 Properties:  (most allow inherit as a value)

   <style>          (font-family, font-weight, font-style,
                     font-variant, direction, unicode-bidi, text-anchor,
                     fill, fill-rule)

   <color>          (black, white, #000000, #ffffff, #FFFFFF)

   stroke           (<color>, none, currentColor)
   stroke-width
   stroke-linecap   (butt, round, square)
   stroke-linejoin  (miter, round, bevel)
   stroke-mitrelimit
   stroke-dasharray
   stroke-dashoffset
   stroke-opacity
   vector-effect    (non-scaling-stroke, none)
   viewport-fill    (none, currentColor)
   viewport-fill-opacity

   display          (inline, block, list-item, run-in, compact,
                       marker, table, inline-table, table-row-group,
                       table-header-group, table-footer-group,
                       table-row, table-column-group,
                       table-column, table-cell, table-caption,
```

```
                    none)
   visibility       (visible, hidden, collapse)
   color-rendering  (auto, optimizeSpeed, optimizeQuality)
   shape-rendering  (auto, optimizeSpeed, crispEdges,
                     geometricPrecision)
   text-rendering   (auto, optimizeSpeed, optimizeLegibility,
                     geometricPrecision)
   buffered-rendering  (auto, dynamic, static)

   opacity
   solid-opacity
   solid-color    (currentColor, <color>)
   color          (currentColor, <color>)

   line-increment (auto)
   text-align     (start,end, center)
   display-align  (auto, before, center, after)

   font-size
   font-family    (serif, sans-serif, monospace)
   font-weight    (normal, bold, bolder, lighter)
   font-style     (normal, italic, oblique)
   font-variant   (normal, small-caps)
   direction      (ltr, rtl)
   unicode-bidi   (normal, embed, bidi-override)
   text-anchor    (start, middle, end)
   fill           (none, <color>)
   fill-rule      (nonzero, evenodd)
   fill-opacity
```

3. How to create SVG drawings

Many drawing packages can be used to create SVG drawings, for example Open Source packages Inkscape and Dia. Be aware that such packages may use SVG elements or attributes that are not allowed in SVG 1.2 RFC.

-For example, the ’marker’ attribute is often used to place symbols such as arrowheads on lines, but ’marker’ is not allowed in SVG 1.2 Tiny or SVG 1.2 RFC. In such cases one has to draw the arrowhead in another, simpler way.

-SVG clip paths are used to define a shape; objects outside that shape become invisible. The ’clipPath’ element is not allowed in SVG 1.2 Tiny or SVG 1.2 RFC.

Diagrams produced with these packages may contain elements, their attributes or properties, or values of attributes or properties that are not allowed in SVG 1.2 RFC. We will need to provide a tool to strip out anything that is not allowed in SVG 1.2 RFC, or to replace disallowed values, e.g., 'sans-serif' for 'Sans' as values for 'font-family'. Experience with a simple test version of a tool for this has shown that such deletion and replacement can be effective for making SVG files from drawing packages conform to SVG 1.2 RFC, without visibly changing the diagrams they produce.

The tool described above can also be used by Authors simply to check that their diagrams conform to SVG 1.2 RFC. To help with this, if visible changes do occur, the tool should produce a list of non-allowed keywords and the context in which they were found.

Another way to create SVG drawings is to write programs to draw them. For example, using python and its svgwrite module is a pleasant environment (for those who like writing code).

To include a diagram into an RFC, the xml2rfc (v3) tool will need to provide a way to include SVG drawings in Internet Drafts, as described in [XML2RFCv3] sections 2.5 and 2.67.

4. Accessibility Considerations

One of the long-term goals for RFCs is to make them more accessible, e.g. to sight-impaired readers. For diagrams, it would be useful for authors to provide alternative forms of the diagram, so that voice-reading software could be used to 'talk through' the diagram. Simply reading the SVG code for a complex diagram seems unlikely to work.

SVG 1.2 RFC allows SVG's 'title' and 'desc' elements. 'title' provides a brief text caption for an SVG object (much like a figure caption), and 'desc' provides a longer text description of what the object actually represents. As well, the SVG 'role' attribute can be used to indicate to a browser how an SVG object is to be interpreted. Good suggestions on how to use these elements are given in [SVG-ACCESS-TIPS].

ARIA is a W3C Recommendation for using SVG to create 'Accessible Rich Internet Applications.' A helpful introduction to ARIA is provided by [SVG-ARIA-PRIMER], while [SVG-USING-ARIA] gives examples of how to use ARIA to enhance SVG accessibility.

5. Meta-language for diagrams common in RFCs

This section presents a few examples of possible meta-languages which could be used to create the kinds of diagrams that are most common in RFCs. Note that they are merely examples, they do not imply that these particular experimental languages might be more widely implemented or used. Instead, they seem to show that designing meta-languages simple enough to serve as audible representations of complex diagrams is difficult indeed!

The SVG diagrams produced from the following examples can be seen at [This-I-D-with-SVG-included] along with an html version of this draft that includes the SVG diagrams.

5.1. Packet Layout Diagrams

Example: Figure 3 from [RFC0793].

In these examples the first line specifies the generated SVG filename. The scale factor determines the size of the SVG drawing; it needs to be set so that the drawing fits nicely into the final document.

'packet;' starts the packet description; it's followed by a description of the fields in each row.

```
   info;
     output "tcp-header.svg", scale 0.65;

   packet;
     row 0;
       field "Source Port", 0 to 15;
       field "Destination Port", 16 to 31;
     row 1;
       field "Sequence Number", 0 to 31;
     row 2;
       field "Acknowledgement Number", 0 to 31;
     row 3;
       field "Data Offset", 0 to 3;
       field "Reserved", 4 to 9;
       field "Urg", 10 to 10, fsize 14;  # 14 px font so the flags fit
       field "Ack", 11 to 11, fsize 14;
       field "Psh", 12 to 12, fsize 14;
       field "Rst", 13 to 13, fsize 14;
       field "Syn", 14 to 14, fsize 14;
       field "Fin", 15 to 15, fsize 14;
       field "Window", 16 to 31;
     row 4;
       field "Checksum", 0 to 15;
       field "Urgent Pointer", 16 to 31;
     row 5;
       field "Options", 0 to 23;
       field "Padding", 24 to 31;
     row 6;
       field "Data", 0 to 31;
```

5.2. Sequence Diagrams (1)

Example: Figure 6 from [ExpTrustedProxy].

In this example, columns are vertical lines with a text header above them. There are three columns, and columns 1 and 2 are spaced 250 pixels apart.

The rest of the file describes objects to be drawn; most of them are plines (polylines) from one column to another, but object 3 only extends across to 0.3 of the distance between columns 1 and 2.

```
   info;
     output "httpbis-proxy20-fig6.svg", scale 0.9;

   #Thu, 30 Jan 14 (NZDT)

   #Figure 6 of draft-loreto-httpbis-trusted-proxy20-00.txt
```

```
   column 1 width 250;  # columns have vertical line to bottom
     text above "user-agent";

   column 2 width 250;
     text "Proxy";

   column 3; # Last col
     text "Server";

   object 1;  # Only need polylines
     pline 1 to 2, arrowhead at end;
     text above "(1) TLS ClientHello";
     text below "(ALPN ProtocolName: http)";

   object 2;
     pline 1 to 2, arrowhead at start;
     text above "(2) TLS Error";
     text below "(Proxy Cert)";

   object 3;
     pline 1 to 1.3, down, back to 1, arrowhead at end;
     text seg 2 centre "(inform user of the SecureProxy)";

   object 4;
     pline 1 to 2, arrowhead at end;
     text above "(3) TLS ClientHello";

   object 5;
     pline 1 to 2, arrowhead at start;
     text above "(4) ServerHello";

   object 6;
     blank 1 to 2;

   object 7;
     block 1 to 2, objects 8 to 15;
     text above "HTTP2.0";

   object 8;
     pline 1 to 2, arrowhead at end;
     text seg 1 centre "(5) stream(X) GET";

   object 9;
     pline 2 to 3, arrowhead at end;
     text seg 1 above "(6) TLS ClientHello";

   object 10;
     pline 2 to 3, arrowhead at start;
```

```
    text seg 1 above "TLS ServerHello";

  object 11;
    blank 2 to 3;

  object 12;
    block 2 to 3, objects 13 to 15;
    text seg 1 above "HTTP2.0";

  object 13;
    pline 2 to 3, arrowhead at end;
    text seg 1 centre "(7) stream(Z) GET";

  object 14;
    pline 2 to 3, arrowhead at start;
    text seg 1 centre "(8) stream(Z) 200 OK";

  object 15;
    pline 1 to 2, arrowhead at start;
    text seg 1 centre "(9) stream(X) 200 OK";
```

5.3. Sequence Diagrams (2)

Example: Figure 3 from [RFC4321].

This example uses (x,y) coordinates to specify points in in plines. For these, the x units are columns and the y units are lines (positive means 'down the diagram').

both x and y may be absolute, e.g. 4.3, or relative, e.g. +1.5). For the first point of a pline, relative means 'relative to the starting point of the previous pline,' for other points in a pline it means 'relative to the last point.'

Note that column 1 is drawn in white, i.e. nothing is drawn for it. It's simply used to make a blank area where objects 8 and 9 can place text. For both those objects a pline is used to specify the text's position.

Last, the metalanguage allows simple macros, introduced by 'define foo = '. These make it easier to re-use definitions, for example of line types.

```
info;
  output "rfc4321-fig3.svg", scale 0.9;

# Sat,  5 Apr 14 (NZDT)
```

```
   #Figure 3 of RFC 4321

   define hw = width 110;  # Hop width

   column 1 width 130, color "white";  # No heading or vertical line

   column 2 hw;  text above "UAC";

   column 3 hw;  text "P1";

   column 4 hw;  text "P2";

   column 5 hw;  text "P3";

   column 6 hw;  text "UAS";

   define tgrey = width 5;  # Thick grey
   define ahe = arrowhead at end;

   object 1;
     pline 1.8
       to 2.3 tgrey, to (2.4,+0), to (2.6,+1.5), to (2.7,+0) ahe,
       to 3.3 tgrey, to (3.4,+0), to (3.6,+1.5), to (3.7,+0) ahe,
       to 4.3 tgrey, to (4.4,+0), to (4.6,+1.5), to (4.7,+0) ahe,
       to 5.3 tgrey, to (5.4,+0), to (5.6,+1.5), to (5.7,+0) ahe,
       to 6.3 tgrey;

   object 2;
     pline (1.8,+10) to 2.3 tgrey;

   object 3;
     pline (3.3,+2)
       to 2.85 tgrey, to (2.7,+0) tgrey,
       to (2.5,+0), to (2.25,+1.5), to (2.0,+0) ahe;
     text seg 2 centre "408";

   object 4;
     pline (4.3,+1.5)
       to 3.9 tgrey, to (3.7,+0) tgrey,
       to (3.5,+0), to (3.3,+1.5), to (3.1,+0) ahe,
       to 2.9 tgrey, to (2.7,+0) tgrey,
       to (2.5,+0), to (2.25,+1.5), to (2.0,+0) ahe;
     text seg 2 centre "408";
     text seg 7 centre "408";

   object 5;
     pline (5.3,+1.5)
       to 4.9 tgrey, to (4.7,+0) tgrey,
```

```
      to (4.5,+0), to (4.3,+1.5), to (4.1,+0) ahe,
      to 3.9 tgrey, to (3.7,+0) tgrey,
      to (3.5,+0), to (3.3,+1.5), to (3.1,+0) ahe,
      to 2.9 tgrey, to (2.7,+0) tgrey,
      to (2.5,+0), to (2.25,+1.5), to (2.0,+0) ahe;
    text seg  2 centre "408";
    text seg  7 centre "408";
    text seg 12 centre "408";

  object 6;
    pline (6.3,+1.5)
      to 5.9 tgrey, to (5.7,+0) tgrey,
      to (5.5,+0), to (5.3,+1.5), to (5.1,+0) ahe;
      to 4.9 tgrey, to (4.7,+0) tgrey,
      to (4.5,+0), to (4.3,+1.5), to (4.1,+0) ahe;
      to 3.9 tgrey, to (3.7,+0) tgrey,
      to (3.5,+0), to (3.3,+1.5), to (3.1,+0) ahe;
      to 2.9 tgrey, to (2.7,+0) tgrey,
      to (2.5,+0), to (2.25,+1.5), to (2.0,+0) ahe;
    text seg  2 centre "408";
    text seg  7 centre "408";
    text seg 12 centre "408";
    text seg 17 centre "408";

  object 7:
    pline (1.63,4.1) to (1.73,+0);

  object 8;
    pline (1.68,4.1) to (+0,14) arrowhead at end;
    text centre "64*T1";

  object 9;
    pline (1.2,13.1) to (1.5,+0) color "white";
    text centre "(timeout)";
```

6. IANA Considerations

This document does not create a new registry nor does it register any values in existing registries; no IANA action is required.

7. Acknowledgements

Thanks to the RSE and the Design Team members for their helpful comments and suggestions for SVG 1.2 RFC.

8. Revision History [RFC Editor please delete]

version -13, 16 Oct 15:
Added Informative Reference to XML2RFC v3 draft on ’how to include SVG diagrams’.
IAdded nformative Reference to Nevil’s home page for a version of this I-D with its SVG diagrams included.
Added Informative References to RFC0793 and I-Ds for the example diagrams.
Changed ’colour’ to ’color’ so as to be consistent.
Fixed other typos (thanks to Dave Thaler for all these)!
Removed ’grey’ color from example diagrams.

version -12, 24 Sep 15:
Appendix A added: a complete relax-ng compact (rnc) schema for SVG-1.2-RFC.
Section 2.1: Elements/attributes/properties table updated to match the schema in Appendix A.

version -11, 17 Aug 15:
Section 1: Fixed typo in "Details are expected to change" paragraph.

version -10, 14 Aug 15:
Section 1: Added "Details are expected to change" paragraph.

version -09, 31 Mar 15:
No changes, version number incremented to keep draft alive

version -08, 29 Sep 14:
Section 1: Changed comment about diagrams ’not being normative’ to ’not complete specifications in themselves.
Section 2.1: Added SVG 1.2 Tiny ’id’ attribute because most drawing packages use it in constructing drawings.
Section 2.1: Added SVG 1.2 Tiny ’role’ attribute so that ARIA can use it.
Section 3: added comment about changes to xml2rfc required to include SVG diagrams.
Section 4: Added reference to svg-aria-primer.

version -07, 3 Jul 14:
Expanded text about Accessibility in ’how to create SVG drawings’ section into ’Accessibility Considerations section. Added two SVG Accessibility references to support that.

version -06, 26 Jun 14:
Remove trailing / from URL in section 4; the html version on tools.ietf.org/html assumed the next word was part of that URL.

version -05, 25 Jun 14:
Improved section on 'how to create SVG drawings' By adding some text about which elements aren't allowed in SVG 1.2 RFC.
Added more text describing the tool for checking, stripping out or replacing incompatible elements and attributes from an SVG file.

version -04, 30 Apr 14:
Fixed typos, used full references for two of the w3c refs - each had an author name using UTF8 characters.
Moved the Elements and Attributes appendix up earlier to make it sub-section 2.1.
Disclaimer added to the Meta-languages section.

version -03, 14 Apr 14:
Added two more example diagrams; a simple packet layout, and a diagram that uses lots of diagonal lines.

version -02, 12 Feb 14:
Added metalanguage example to make time-sequence drawings.

version -01, 11 Feb 14:
Allow links to 'long-term stable URIs'
Link URIs must be ASCII only
Need for tools to check SVG 1.2 RFC compatibility and to strip 'unnecessary' attributes explicitly stated.
Statement that drawings can't be normative removed; Postscript-only RFCs already exist.
Added most attributes and elements to the Appendix.

version -00, 29 Jan 14:
Initial version, using content from Nevil's emails to the Design Team.

9. References

9.1. Normative References

[RFC6949] Flanagan, H. and N. Brownlee, "RFC Series Format Requirements and Future Development", RFC 6949, DOI 10.17487/RFC6949, May 2013, <http://www.rfc-editor.org/info/rfc6949>.

[W3C.REC-SVGTiny12-20081222]
Andersson, O., Berjon, R., Dahlstrom, E., Emmons, A., Ferraiolo, J., Grasso, A., Hardy, V., Hayman, S., Jackson, D., Lilley, C., McCormack, C., Neumann, A., Northway, C., Quint, A., Ramani, N., Schepers, D., and A. Shellshear, "Scalable Vector Graphics (SVG) Tiny 1.2 Specification", World Wide Web Consortium Recommendation REC-SVGTiny12-20081222, December 2008, <http://www.w3.org/TR/2008/REC-SVGTiny12-20081222>.

[W3C.REC-CSS2-20110607]
Bos, B., Celik, T., Hickson, I., and H. Lie, "Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification", World Wide Web Consortium Recommendation REC-CSS2-20110607, June 2011, <http://www.w3.org/TR/2011/REC-CSS2-20110607>.

9.2. Informative References

[W3C.REC-SVG11-20110816]
Dahlstrom, E., Dengler, P., Grasso, A., Lilley, C., McCormack, C., Schepers, D., Watt, J., Ferraiolo, J., Fujisawa, J., and D. Jackson, "Scalable Vector Graphics (SVG) 1.1 (Second Edition)", World Wide Web Consortium Recommendation REC-SVG11-20110816, August 2011, <http://www.w3.org/TR/2011/REC-SVG11-20110816>.

[SVG-ACCESS-TIPS]
Watson, L., "Tips for Creating Accessible SVG", SitePoint tips-accessible-svg, May 2014, <http://www.sitepoint.com/tips-accessible-svg>.

[SVG-ARIA-PRIMER]
Pappas, L., Schwerdtfeger, R., and M. Cooper, "WAI-ARIA 1.0 Primer", World Wide Web Consortium WD WD-wai-aria-primer-20100916, September 2010, <http://www.w3.org/TR/2010/WD-wai-aria-primer-20100916>.

[SVG-USING-ARIA]
Watson, L., "Using ARIA to enhance SVG accessibility", The Paciello Group 2013/12/using-aria-enhance-svg-accessibility, December 2013, <http://blog.paciellogroup.com/2013/12/using-aria-enhance-svg-accessibility>.

[XML2RFCv3]
Hoffman, P., "The XML2RFC version 3 Vocabulary", Work in Progress, draft-hoffman-xml2rfc-23, September 4 2015.

[This-I-D-with-SVG-included]
Brownlee, N., "Example html version of this I-D with its SVG diagrams included", Nevil's home page, https://www.cs.auckland.ac.nz/~nevil/SVG_RFC_1.2, October 16 2015.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[ExpTrustedProxy]
Loretto, S., Mattson, J., Skog, R., Spaak, H., Gus, G., Druta, D., and M. Hafeez, "Explicit Trusted Proxy in HTTP/2.0", Work in Progress, draft-loreto-httpbis-trusted-proxy20-01, February 14 2014.

[RFC4321] Sparks, R., "Problems Identified Associated with the Session Initiation Protocol's (SIP) Non-INVITE Transaction", RFC 4321, DOI 10.17487/RFC4321, January 2006, <http://www.rfc-editor.org/info/rfc4321>.

Appendix A. RELAX NG Compact (rnc) Schema for SVG 1.2 RFC

The following rnc schema can be used to check whether an svg file conforms to SVG 1.2 RFC. For example, if this schema were contained in a file called SVG-1.2-RFC.rnc, the following command will test whether svg file diagram.svg is a conformant SVG 1.2 RFC drawing.

```
jing -c SVG-1.2-RFC.rnc diagram.svg
```

```
#---  SVG 1.2 RFC rnc schema;  Nevil Brownlee, Thu 24 Sep 2015 (NZST)

default namespace = "http://www.w3.org/2000/svg"
namespace ns1 = "http://www.w3.org/1999/xlink"

rfc-color = (  # SVG-1.2-RFC doesn't allow "color or grey-scale"
  "black" | "white" | "#000000" | "#FFFFFF" | "#ffffff" | "inherit" )
start = svg
svg =
  element svg {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
```

```
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
    & attribute vector-effect {
        "none" | "non-scaling-stroke" | "inherit"
      }?
    & (attribute direction { "ltr" | "rtl" | "inherit" }?,
       attribute unicode-bidi {
         "normal" | "embed" | "bidi-override" | "inherit"
       }?)
    & (attribute solid-color { rfc-color }?,
       attribute solid-opacity { "inherit" | xsd:string }?)
    & (attribute display-align {
         "auto" | "before" | "center" | "after" | "inherit"
       }?,
       attribute line-increment { "auto" | "inherit" | xsd:string }?)
    & (attribute stop-color { rfc-color }?,
       attribute stop-opacity { "inherit" | xsd:string }?)
    & (attribute font-family { "inherit" | xsd:string }?,
       attribute font-size { "inherit" | xsd:string }?,
       attribute font-style {
         "normal" | "italic" | "oblique" | "inherit"
       }?,
       attribute font-variant { "normal" | "small-caps" | "inherit" }?,
       attribute font-weight {
         "normal"
         | "bold"
         | "bolder"
         | "lighter"
       }?,
       attribute text-anchor {
         "start" | "middle" | "end" | "inherit"


}?,
       attribute text-align {
         "start" | "center" | "end" | "inherit"
       }?)),
   (attribute id { xsd:NCName }
    | attribute xml:id { xsd:NCName })?,
```

```
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,

    attribute width { xsd:string }?,
    attribute height { xsd:string }?,
    attribute preserveAspectRatio {
      xsd:string { pattern = "\s*(none|xMidYMid)\s*(meet)?\s*" }
    }?,
    attribute viewBox { text }?,
    attribute zoomAndPan { "disable" | "magnify" }?,
    attribute version {
      xsd:string "1.0" | xsd:string "1.1" | xsd:string "1.2"
    }?,
    attribute baseProfile {
      xsd:string "none"
      | xsd:string "tiny"
      | xsd:string "basic"
      | xsd:string "full"
    }?,
    attribute contentScriptType { xsd:string }?,
    attribute snapshotTime { xsd:string "none" | xsd:string }?,
    attribute timelineBegin {
      xsd:string "onLoad" | xsd:string "onStart"
    }?,
    attribute playbackOrder {
      xsd:string "all" | xsd:string "forwardOnly"
    }?,
    (desc
     | title
     | path
     | rect
     | circle
     | line
     | ellipse
     | polyline
     | polygon
     | solidColor
```

```
      | textArea
      | linearGradient
      | radialGradient
      | \text
      | g
      | defs
      | use
      | a)*
  }
desc =
  element desc {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    ((attribute display {
        "inline"
        | "block"
        | "list-item"
        | "run-in"
        | "compact"
        | "marker"
        | "table"
        | "inline-table"
        | "table-row-group"
        | "table-header-group"
        | "table-footer-group"
        | "table-row"
        | "table-column-group"
        | "table-column"
        | "table-cell"
        | "table-caption"
```

```
        | "none"
        | "inherit"
      }?,
      attribute visibility { "visible" | "hidden" | "collapse" | "inherit" }?,
      attribute image-rendering {
        "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
      }?,
      attribute shape-rendering {
        "auto"
        | "optimizeSpeed"
        | "crispEdges"
        | "geometricPrecision"
        | "inherit"
      }?,
      attribute text-rendering {
        "auto"
        | "optimizeSpeed"
        | "optimizeLegibility"
        | "geometricPrecision"
        | "inherit"
      }?,
      attribute buffered-rendering {
        "auto" | "dynamic" | "static" | "inherit"
      }?)
     & (attribute viewport-fill { "none" | rfc-color }?,
        attribute viewport-fill-opacity { "inherit" | xsd:string }?)),
    text
  }
title =
  element title {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
```

```
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    ((attribute display {
        "inline"
        | "block"
        | "list-item"
        | "run-in"
        | "compact"
        | "marker"
        | "table"
        | "inline-table"
        | "table-row-group"
        | "table-header-group"
        | "table-footer-group"
        | "table-row"
        | "table-column-group"
        | "table-column"
        | "table-cell"
        | "table-caption"
        | "none"
        | "inherit"
      }?,
      attribute visibility { "visible" | "hidden" | "collapse" | "inherit" }?,
      attribute image-rendering {
        "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
      }?,
      attribute shape-rendering {
        "auto"
        | "optimizeSpeed"
        | "crispEdges"
        | "geometricPrecision"
        | "inherit"
      }?,
      attribute text-rendering {
        "auto"
        | "optimizeSpeed"
        | "optimizeLegibility"
        | "geometricPrecision"
        | "inherit"
      }?,
      attribute buffered-rendering {
        "auto" | "dynamic" | "static" | "inherit"
      }?)
     & (attribute viewport-fill { "none" | rfc-color }?,
        attribute viewport-fill-opacity { "inherit" | xsd:string }?)),
    text
  }
path =
```

```
  element path {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
```

```
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute d { xsd:string }?,
    attribute pathLength { xsd:string }?,
    attribute style { xsd:string }?,  # Added to SVG-1.2-RFC (Inkscape)
    (desc
     | title)*
  }
rect =
  element rect {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
```

```
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
```

```
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute x { xsd:string }?,
    attribute y { xsd:string }?,
    attribute width { xsd:string }?,
    attribute height { xsd:string }?,
    attribute rx { xsd:string }?,
    attribute ry { xsd:string }?,
    attribute style { xsd:string }?,  # Added to SVG-1.2-RFC (Inkscape)
    (desc
     | title)*
  }
circle =
  element circle {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
```

```
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
      & attribute vector-effect {
          "none" | "non-scaling-stroke" | "inherit"
        }?
      & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
      & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
      & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
      & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
      & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
```

```
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute cx { xsd:string }?,
    attribute cy { xsd:string }?,
    attribute r { xsd:string }?,
    attribute style { xsd:string }?,  # Added to SVG-1.2-RFC (Inkscape)
    (desc
     | title)*
  }
line =
  element line {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
```

```
          "none" | "non-scaling-stroke" | "inherit"
        }?
      & (attribute direction { "ltr" | "rtl" | "inherit" }?,
         attribute unicode-bidi {
           "normal" | "embed" | "bidi-override" | "inherit"
         }?)
      & (attribute solid-color { rfc-color }?,
         attribute solid-opacity { "inherit" | xsd:string }?)
      & (attribute display-align {
           "auto" | "before" | "center" | "after" | "inherit"
         }?,
         attribute line-increment { "auto" | "inherit" | xsd:string }?)
      & (attribute stop-color { rfc-color }?,
         attribute stop-opacity { "inherit" | xsd:string }?)
      & (attribute font-family { "inherit" | xsd:string }?,
         attribute font-size { "inherit" | xsd:string }?,
         attribute font-style {
           "normal" | "italic" | "oblique" | "inherit"
         }?,
         attribute font-variant { "normal" | "small-caps" | "inherit" }?,
         attribute font-weight {
           "normal"
           | "bold"
           | "bolder"
           | "lighter"
           | "inherit"
         }?,
         attribute text-anchor {
           "start" | "middle" | "end" | "inherit"
         }?,
         attribute text-align {
           "start" | "center" | "end" | "inherit"
         }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute x1 { xsd:string }?,
    attribute y1 { xsd:string }?,
    attribute x2 { xsd:string }?,
    attribute y2 { xsd:string }?,
    (desc
     | title)*
  }
ellipse =
  element ellipse {
    (attribute id { xsd:NCName }
```

```
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
```

```
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute rx { xsd:string }?,
    attribute ry { xsd:string }?,
    attribute cx { xsd:string }?,
    attribute cy { xsd:string }?,
    attribute style { xsd:string }?,  # Added to SVG-1.2-RFC (Inkscape)
    (desc
     | title)*
  }
polyline =
  element polyline {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
```

```
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
```

```
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute points { xsd:string }?,
    (desc
     | title)*
  }
polygon =
  element polygon {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute transform { xsd:string | "none" }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
```

```
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
      & attribute vector-effect {
          "none" | "non-scaling-stroke" | "inherit"
        }?
      & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
      & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
      & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
      & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
      & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute points { xsd:string }?,
    attribute style { xsd:string }?,  # Added to SVG-1.2-RFC (Inkscape)
    (desc
```

```
     | title)*
  }
solidColor =
  element solidColor {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
```

```
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    (desc
     | title)*
  }
textArea =
  element textArea {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
```

```
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
```

```
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute transform { xsd:string | "none" }?,
    attribute x { xsd:string }?,
    attribute y { xsd:string }?,
    attribute width { xsd:string | "auto" }?,
    attribute height { xsd:string | "auto" }?,
    (tspan
     | desc
     | title
     | tspan_2
     | text
     | a_2)+
  }
linearGradient =
  element linearGradient {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
```

```
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute gradientUnits { "userSpaceOnUse" | "objectBoundingBox" }?,
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute x1 { xsd:string }?,
    attribute y1 { xsd:string }?,
    attribute x2 { xsd:string }?,
    attribute y2 { xsd:string }?,
    (desc
     | title)*
  }
```

```
radialGradient =
  element radialGradient {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
```

```
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute gradientUnits { "userSpaceOnUse" | "objectBoundingBox" }?,
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute cx { xsd:string }?,
    attribute cy { xsd:string }?,
    attribute r { xsd:string }?,
    (desc
     | title)*
  }
\text =
  element text {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
```

```
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
```

```
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute transform { xsd:string | "none" }?,
    attribute x { xsd:string }?,
    attribute y { xsd:string }?,
    attribute rotate { xsd:string }?,
    attribute style { xsd:string }?,  # Added to SVG-1.2-RFC (Inkscape)
    (desc
     | title
     | tspan_2
     | text
     | a_2)+
  }
g =
  element g {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
```

```
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute transform { xsd:string | "none" }?,
    attribute style { xsd:string }?,
```

```
        # Added to SVG-1.2-RFC (for Inkscape)
    attribute visibility {
      "visible" | "hidden" | "collapse" | "inherit" }?,
        # Added to SVG-1.2-RFC (for LibreOffice)
    (desc
     | title
     | path
     | rect
     | circle
     | line
     | ellipse
     | polyline
     | polygon
     | solidColor
     | textArea
     | linearGradient
     | radialGradient
     | \text
     | g
     | defs
     | use
     | a)*
  }
defs =
  element defs {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
```

```
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    (desc
     | title
     | path
```

```
      | rect
      | circle
      | line
      | ellipse
      | polyline
      | polygon
      | solidColor
      | textArea
      | linearGradient
      | radialGradient
      | \text
      | g
      | defs
      | use
      | a)*
  }
use =
  element use {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
```

```
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute transform { xsd:string | "none" }?,
    attribute ns1:show { "embed" }?,
    attribute ns1:actuate { "onLoad" }?,
    attribute ns1:type { "simple" }?,
    attribute ns1:role { xsd:anyURI | xsd:string }?,
```

```
    attribute ns1:arcrole { xsd:anyURI | xsd:string }?,
    attribute ns1:title { text }?,
    attribute ns1:href { xsd:anyURI | xsd:string }?,
    attribute x { xsd:string }?,
    attribute y { xsd:string }?,
    (desc
     | title)*
  }
a =
  element a {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
```

```
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute transform { xsd:string | "none" }?,
    attribute ns1:show { "new" | "replace" }?,
    attribute ns1:actuate { "onRequest" }?,
    attribute ns1:type { "simple" }?,
    attribute ns1:role { xsd:anyURI | xsd:string }?,
    attribute ns1:arcrole { xsd:anyURI | xsd:string }?,
    attribute ns1:title { text }?,
    attribute ns1:href { xsd:anyURI | xsd:string }?,
    attribute target {
      "_replace" | "_self" | "_parent" | "_top" | "_blank" | xsd:Name
    }?,
    (desc
     | title
```

```
     | path
     | rect
     | circle
     | line
     | ellipse
     | polyline
     | polygon
     | solidColor
     | textArea
     | linearGradient
     | radialGradient
     | \text
     | g
     | defs
     | use)*
  }
stop =
  element stop {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "inherit" | xsd:string }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { "inherit" | xsd:string }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
     & attribute vector-effect {
         "none" | "non-scaling-stroke" | "inherit"
       }?
     & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
     & (attribute solid-color { "inherit" | xsd:string }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
     & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
```

```
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
     & (attribute stop-color { "inherit" | xsd:string }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
     & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute offset { xsd:string }?,
    (desc
     | title)*
  }
tspan =
  element tspan {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
```

```
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
      & attribute vector-effect {
          "none" | "non-scaling-stroke" | "inherit"
        }?
      & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
      & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
      & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
      & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
      & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
```

```
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute x { xsd:string }?,  # For SVG-1.2-RFC
    attribute y { xsd:string }?,
    (tbreak
     | desc
     | title
     | tspan_2
     | text
     | a_2)+
  }
tspan_2 =
  element tspan {
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
```

```
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
      & attribute vector-effect {
          "none" | "non-scaling-stroke" | "inherit"
        }?
      & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
      & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
      & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
      & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
      & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
```

```
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    attribute x { xsd:string }?,  # For SVG-1.2-RFC
    attribute y { xsd:string }?,
    (desc
     | title
     | tspan_2
     | text
     | a_2)+
  }
a_2 =
  element a {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?,
    ((attribute fill-opacity { "inherit" | xsd:string }?,
      attribute stroke-opacity { "inherit" | xsd:string }?)
     & (attribute fill { "none" | rfc-color }?,
        attribute fill-rule { "inherit" | "nonzero" | "evenodd" }?,
        attribute stroke { rfc-color }?,
        attribute stroke-dasharray { "inherit" | "none" | xsd:string }?,
        attribute stroke-dashoffset { "inherit" | xsd:string }?,
        attribute stroke-linecap {
          "butt" | "round" | "square" | "inherit"
```

```
        }?,
        attribute stroke-linejoin {
          "miter" | "round" | "bevel" | "inherit"
        }?,
        attribute stroke-miterlimit { "inherit" | xsd:string }?,
        attribute stroke-width { "inherit" | xsd:string }?,
        attribute color { rfc-color }?,
        attribute color-rendering {
          "auto" | "optimizeSpeed" | "optimizeQuality" | "inherit"
        }?)
      & attribute vector-effect {
          "none" | "non-scaling-stroke" | "inherit"
        }?
      & (attribute direction { "ltr" | "rtl" | "inherit" }?,
        attribute unicode-bidi {
          "normal" | "embed" | "bidi-override" | "inherit"
        }?)
      & (attribute solid-color { rfc-color }?,
        attribute solid-opacity { "inherit" | xsd:string }?)
      & (attribute display-align {
          "auto" | "before" | "center" | "after" | "inherit"
        }?,
        attribute line-increment { "auto" | "inherit" | xsd:string }?)
      & (attribute stop-color { rfc-color }?,
        attribute stop-opacity { "inherit" | xsd:string }?)
      & (attribute font-family { "inherit" | xsd:string }?,
        attribute font-size { "inherit" | xsd:string }?,
        attribute font-style {
          "normal" | "italic" | "oblique" | "inherit"
        }?,
        attribute font-variant { "normal" | "small-caps" | "inherit" }?,
        attribute font-weight {
          "normal"
          | "bold"
          | "bolder"
          | "lighter"
          | "inherit"
        }?,
        attribute text-anchor {
          "start" | "middle" | "end" | "inherit"
        }?,
        attribute text-align {
          "start" | "center" | "end" | "inherit"
        }?)),
    attribute transform { xsd:string | "none" }?,
    attribute ns1:show { "new" | "replace" }?,
    attribute ns1:actuate { "onRequest" }?,
    attribute ns1:type { "simple" }?,
```

```
    attribute ns1:role { xsd:anyURI | xsd:string }?,
    attribute ns1:arcrole { xsd:anyURI | xsd:string }?,
    attribute ns1:title { text }?,
    attribute ns1:href { xsd:anyURI | xsd:string }?,
    attribute target {
      "_replace" | "_self" | "_parent" | "_top" | "_blank" | xsd:Name
    }?,
    (desc
     | title
     | tspan_2
     | text)+
  }
tbreak =
  element tbreak {
    (attribute id { xsd:NCName }
     | attribute xml:id { xsd:NCName })?,
    attribute xml:base { xsd:anyURI | xsd:string }?,
    attribute xml:lang { xsd:language? }?,
    attribute class { xsd:NMTOKENS }?,
    attribute role { xsd:string }?,
    attribute rel { xsd:string }?,
    attribute rev { xsd:string }?,
    attribute typeof { xsd:string }?,
    attribute content { xsd:string }?,
    attribute datatype { xsd:string }?,
    attribute resource { xsd:string }?,
    attribute about { xsd:string }?,
    attribute property { xsd:string }?,
    attribute xml:space { "default" | "preserve" }?,
    attribute requiredFeatures { xsd:string }?,
    attribute requiredExtensions { xsd:string }?,
    attribute requiredFormats { xsd:string }?,
    attribute requiredFonts { xsd:string }?,
    attribute systemLanguage { xsd:string }?
  }

#---  End of SVG 1.2 RFC rnc schema
```

Authors' Addresses

Nevil Brownlee
The University of Auckland

Email: n.brownlee@auckland.ac.nz

Internet Architecture Board

Email: iab@iab.org