Proposal for Adding Font Feature Support to the Open Document Format

Keith Stribley (ThanLwinSoft) &
Martin Hosken (SIL International & Payap University)

1 Introduction

The 3 major font technologies in use in most Open Document Format implementations are OpenType, Apple Advanced Typography (AAT) and Graphite. Each of these technologies supports "Font Features" as a way to control rendering of a specific font-family beyond that allowed by just font-weight and font-style. This proposal suggests two alternative ways of storing font feature information in the Open Document Format, so that font features may be utilized in ODF documents.

Most of the time, font features will only be used by advanced users knowledgeable in typography, who want more control over the appearance of their document. However, there may also be cases where ethnic groups without explicit software support for their locale or language require a feature to be applied to get correct rendering.

It might be thought that features could be avoided by just creating separate fonts for each feature combination. However, some fonts have a very large number of features, for which it would not be practical to create individual fonts for each permutation.

1.1 Examples

The Doulos SIL font has 27 different Graphite features (which would require over 100 million fonts to represent all combinations). For example, one feature allows alternate glyphs to be used for Capital R Tail; another changes the glyph of Uppercase Eng:

- ꞃ normal appearance
- ꞃ feature `1039=1` and feature `1024=2`

The Padauk font uses features to enable different linguistic preferences to be displayed by the same font:

- normal မိ `utal=1` မိ (tall u vowel)
- normal တ `wtri=1` တ (triangular wasway)

These examples used Graphite, but similar capabilities are possible in OpenType and AAT fonts.

2 Font Technologies

The font feature characteristics of each font technology are described below. Each technology uses an identifier for each feature. Each feature can be set a numeric value. In Open Type, the feature identifier can be a 4 byte ASCII tag, which is represented by a 32 bit integer using big endian encoding. For example, the 4 character ASCII tag "dflt" has a numeric value of 0x64666c74 (1684434036 decimal). In AAT feature identifiers are numeric, and are given a more understandable English name in the Apple

Feature Registry. Graphite has a 32 bit feature identifier which may be interpretted as an ASCII tag (if the highest byte is set) or as an unsigned number.

2.1 Open Type

Open Type is probably the most widely used of the font technologies and was developed jointly by Microsoft and Adobe. It uses features both for linguistic shaping determined by the script and language of the text and for discretional features. It is probably only useful to allow the discretional features to be specified in an ODF document, since the others will be controlled by the renderer such as Uniscribe on Windows and ICU[1] or Harfbuzz on other platforms. The script information can in most cases be determined from the Unicode code points of a piece of text and the language is already storable in ODF within the `fo:language`, `style:language-asian` and `style:language-complex`, XML attributes (but note section 4.1).

OpenType Features are given an ASCII 4 character tag and their values are mostly boolean, though some features such as alternate glyph selection allow unsigned 16 bit values.[2] Feature names are controlled by a central registry. Most registered features have a clear English definition, which could be localized by the application. However, a few features such as character variants `cv01` to `cv99` and stylistic sets `ss01` to `ss20` do have font specific localized names held in the name table.

2.2 Apple Advanced Typography

AAT is a technology developed by Apple on top of TrueType. AAT uses font features to allow additional control of rendering in addition to that available by font weight, style and variant. Apple maintains a central Tag Registry of font features which may be used. The feature names and settings names are held in the font's `name` table, but for registered features, the names could be localized by the application itself instead.

In some cases an AAT feature may be allowed to have multiple setting values for the same feature. This is determined by the feature flag in the `feat` table.

2.3 Graphite

The features in Graphite fonts are controlled by the font designer, and so are only known to the application at runtime. Font designers or foundries may choose to share font feature definitions between similar fonts, but that is not enforced. The list of features and their available settings is held inside the font in the `Feat` table (note upper case initial). The `Sill` table may change the default values of features based on the language of the text (see also section 4.1).

Graphite features are identified by an unsigned 32 bit integer. A 4 byte ASCII character tag can also be used, in which case, the tag is Big Endian encoded.

2.4 Comparison of Features used by different Font Technologies

The following table compares some of the main characteristics of font features between the different technologies.

1 International Components in Unicode

2 http://www.microsoft.com/typography/otspec/gsub.htm#ASF1 see the GlyphCount field in the AlternateSet table.

Technology	OpenType	Apple Advanced Typography	Graphite
Set of Font Features	Microsoft Tag registry http://www.microsoft.com/typography/otspec/featuretags.htm.	AAT Font Feature Registry http://developer.apple.com/fonts/Registry/index.html	Controlled by font designer for each font. Accessible from font.
Feature identifier	4 byte ASCII tag, space 0x20 padded if necessary.	Unsigned 16 bit integer	Unsigned 32 bit integer or 4 byte ASCII tag, 0x0 padded if necessary.
Feature Values	Normally 0 or 1, but may be unsigned 16 bit integer[3]	Unsigned 16 bit integer	Signed 16 bit integer
Default Feature Values	Determined by layout engine implementation.	Feature flags in `feat` table.	First value listed in Feat table, but may be overridden for a specific language code in the `Sill` table.
Localization of feature names & settings	External, except for `cvXX`, `ssXX`	In name table or external	In name table
Applicable TrueType Tables	See individual feature descriptions. http://www.microsoft.com/typography/otspec/featuretags.htm	`feat`	`Feat`, `Sill`

3 Options for storing font feature information in ODF

Two options are presented here for storing font feature information in an ODF document:

1. Extend the existing face name to include a list of font features appended to it.
2. Add an additional style attribute to store font feature name, value pairs.

The first option does not require a change to the ODF schema, but it ought to be documented properly. An application, which is not aware of the extended naming scheme will fail to match the font correctly, even ignoring the feature settings. The second option requires a change to the ODF specification, but is much clearer and more consistent with other font parameters.

3.1 Extended Face Names

In this case, the ODF schema remains unmodified, but the description of the font-family name attributes needs to state that a list of features can be appended to the `svg:font-family`, `fo:font-family`, `style:font-family-complex` and `style:font-family-asian` attributes' values.

3 http://msdn.microsoft.com/en-us/library/dd319096.aspx

OpenOffice since 3.2 has supported Graphite font features using an extended naming scheme. A colon ':' is used to separate the family name from the feature list. Individual features are separated using ampersand '&'. '=' is used to separate the feature identifier from the feature value, which is a decimal integer. For example, for the face “Doulos SIL”, with the `1039` feature set to a value of 1 and the `1024` feature set to a value of 2, the extended' face name would become:

```
"Doulos SIL:1039=1&1024=2"
```

This results in a font-face entry of:

```
<style:font-face style:name="Doulos SIL:1039=1&amp;1024=2" svg:font-
family="&apos;Doulos SIL:1039=1&amp;1024=2&apos;"/>
```

or, in a graphics document, specified directly in text-properties:

```
<style:text-properties fo:font-family="&apos;Doulos SIL:1039=1&amp;1024=2&apos;"/>
```

The “Padauk” font uses ASCII tag names for it's Graphite features, so the triangular wa feature `wtri=1` and the tall U feature `utal=1` would combine to give a face name of:

```
"Padauk:wtri=1&utal=1"
```

If extended face names are adopted, then the OpenDocument specification should be modified to include a comment describing such names and recommending that applications without feature support should just drop everything following a colon from the family name, since otherwise they will fall-back to a completely different font-family.

':' was chosen to start the feature list since it is very unlikely to appear in a face-name. ',' and ';' cannot be used to separate features, since they are already used to delimit a list of fallback fonts, hence '&' was chosen. '&' is not found in any of the currently registered OpenType feature names and is unlikely to be used in a feature identifier tag.

If multiple values are required for the same feature, as in AAT without the mutually exclusive flag set, then each value should be specified sequentially e.g `15=6&15=8`. The order of setting features is not significant for Graphite or AAT, though it might in certain circumstances affect the result with OpenType fonts, depending on how the renderer handles the order of processing OpenType lookups.

3.2 Additional Style Attributes

The current draft of CSS3 has a `font-feature-settings` property, which uses a comma separated list of feature_identifier=feature_value pairs. This has already been implemented for at least the Mozilla web browser.[4] CSS2 is already referenced by the ODF specification, so it might be reasonable to borrow from CSS3 to improve consistency between file formats. A reasonable solution, might therefore be to use attributes something like:

```
css3:font-feature-settings
style:font-feature-settings-complex
style:font-feature-settings-asian
```

These refer to the Latin, Complex Text Layout and Asian script fonts respectively.

For Open Type and Graphite fonts using ASCII tag names, feature_ identifier would be an ASCII tag. For AAT and Graphite fonts with purely numeric feature identifiers, then the feature_ identifier will be an unsigned integer. The feature_value will always be a decimal integer and in the case of Graphite fonts, may be signed.

4 https://wiki.mozilla.org/Platform/2010-08-03#Layout

```
<style:text-properties style:font-name="Doulos SIL" css3:font-feature-
settings="1024=2,1039=1" style:font-name-complex="Padauk" style:font-feature-
settings-complex="wtri=1,utal=1"/>
```

4 Other considerations

4.1 Font Language Override

The CSS3 proposal contains a `font-language-override` property,[5] which can be used to specify a different language for the font renderer than that used by the text of the document for say spell-checking. This may be useful if a font defines rendering variants for one language, which are also applicable for another language which is not explicitly supported by the font. In this case the user might want to specify the `font-language-override` to be the language supported by an OpenType or Graphite feature set, but have the text language to be the actual language for spell-checking purposes. Another possibility is where the font supports features for a language, but the ODF application does not yet support that language in it's language list. The list of language codes supported by a font, is accessible from the font's tables.

Allowing `font-language-override` would give more flexibility, but risks giving a confusing user-interface. However, it may be worth adding for consistency with CSS3, in which case it will also need Asian and Complex Text Layout variants. The interim extended face name support for Graphite features in OpenOffice 3.2 currently allows for overriding the language name with a special `lang=xyz` feature setting, where `xyz` is the ISO639-3 language code as used by the font to select a specific combination of features.

4.2 Existing ODF mark-up implemented using Font Features

There are cases where existing ODF attributes overlap with some specific font features. For example, ODF already uses the `svg:font-variant` attribute, which allows for `small-caps` and Open Type has a `smcp` “Small Captials” feature and AAT has the “Letter Case” feature (3), which supports a “Small Caps” selector (3). The application may choose to implement the small-caps feature variant using a relevant font-feature, even when it has not been specified explicitly. If the alternative ODF markup and the `font-feature-settings` attribute explicitly conflict, then the authors suggest that the `font-feature-settings` value should win. In the case of `small-caps`, this would allow the user to enable the application's synthetic small-caps, if the font's small-caps feature appearance is undesired.

4.3 Feature identifiers which overlap between rendering technologies

An OpenDocument file may be rendered using one font technology on one platform/application and with another on another platform/application. There may be a theoretical case where a numeric font identifier is used both in a Graphite version of the font and an AAT version of the font, but with a different meaning for the value. It is up to the Graphite font designer to avoid such a problem. Similarly, a Graphite ASCII tag could in theory conflict with an OpenType tag with a different meaning. It might be possible to add a prefix to feature names which specified a fixed technology. However, this issue would be best addressed by the font designer not using the same tag name for different purposes in separate rendering technologies. The other ODF parameters are font-technology

5 http://dev.w3.org/csswg/css3-fonts/#propdef-font-language-override

agnostic, so it is probably best to keep that true for font-features as well. It is up to the application to ignore `font-feature-settings` which are inappropriate for the font rendering technology which is currently in use.

4.4 User Interface for specifying features

An ODF application supporting font features will need a way to allow the user to select which features are applied to text and set the features' values. This means that feature names and where more than one exists, feature setting labels, will need to be localized. This is allowed for in Graphite and AAT fonts already, but for most of the OpenType features, the list of registered feature descriptions would need to be localized as part of the application's normal localization process.

Documents containing a mix of Latin, CTL and Asian scripts will need the features to be selectable according to the font in use for each script type.

5 Conclusion

This proposal has presented two possible solutions for storing font feature information in the Open Document Format. The extended face names described in 3.1 have already been implemented for Graphite fonts as an interim measure in OpenOffice 3.2, however the authors would recommend 3.2 as the best long term solution. Alternative XML mark-up to feature mappings could be used instead of those presented in 3.2, but the `font-feature-settings` attribute already has a precedent in CSS3 and gives a high level of control, with minimum implementation cost. For full flexibility, it is worth considering whether the `font-language-override` attribute from CSS3 should also be added.

6 References

The OASIS Open Document Format for Office Applications, OASIS OpenDocument TC, 8 July 2010, http://docs.oasis-open.org/office/v1.2/cd05/.

Adding Font Feature Support to ODF, A Proposal, Martin Hosken & Tim Eves, 6 August 2008.

OpenType Layout tag registry, Microsoft, 4 April 2002, http://www.microsoft.com/typography/otspec/featuretags.htm.

The 'feat' table, Apple, 18 May 2000, http://developer.apple.com/fonts/TTRefMan/RM06/Chap6feat.html.

Font Feature Registry, Apple, 25 April 1996, http://developer.apple.com/fonts/Registry/index.html.

Graphite Table Format : Extending TrueType for Graphite, Version 4 , 18 May 2010, Martin Hosken and Sharon Correll , http://scripts.sil.org/svn-public/graphite/graphite/trunk/engine/doc/GTF.odt.

Uniscribe Functions (Windows), Microsoft, 6 July 2010, http://msdn.microsoft.com/en-us/library/dd374093.aspx.

CSS Fonts Module Level 3, John Daggett, Editor's Draft 10 August 2010, http://dev.w3.org/csswg/css3-fonts/#propdef-font-feature-settings.