

Rendering/Layout Engine for Complex script

Pema Geyleg pgeyleg@dit.gov.bt



Overview

- What is the Layout Engine/ Rendering?
 - What is complex text?
 - Types of rendering engine?
 - How does it work?
 - How does it support the display of Dzongkha text?
-

What is Layout Engine / Rendering?

- How different scripts are displayed by the particular software.
 - It identifies the script that the user wants, and displays the text using that script correctly.
 - The Latin script, is least complex script to display especially when used to write English.
 - Mainly used to display complex scripts properly /correctly.
-

What Is Complex Text?

- Unicode: not just a bigger character set
 - Bidirectionality: mixed directions on a line
 - Shaping: character shapes depend on context
 - Ligatures: mandatory special forms, and no Unicode equivalent
 - Positioning: vertical and horizontal adjustments
 - Reordering: character positions depend on context
 - Split characters: some characters appear in more than one position
-

Bidirectional Text

- Visual order differs from storage order
- Arabic and Hebrew read right to left, but numbers still read left to right

So, תא הרש is a sentence.

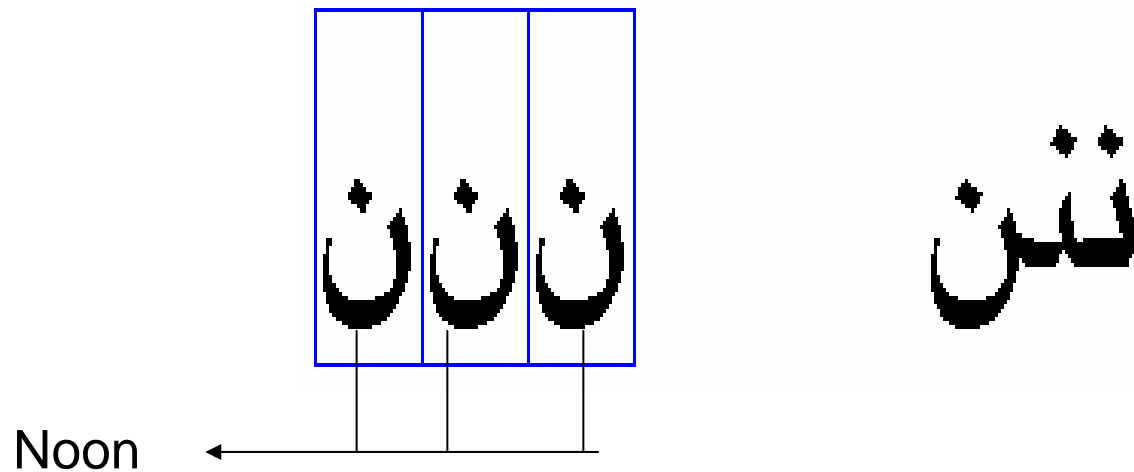
Memory

So, את שרה is a sentence.

Reading order

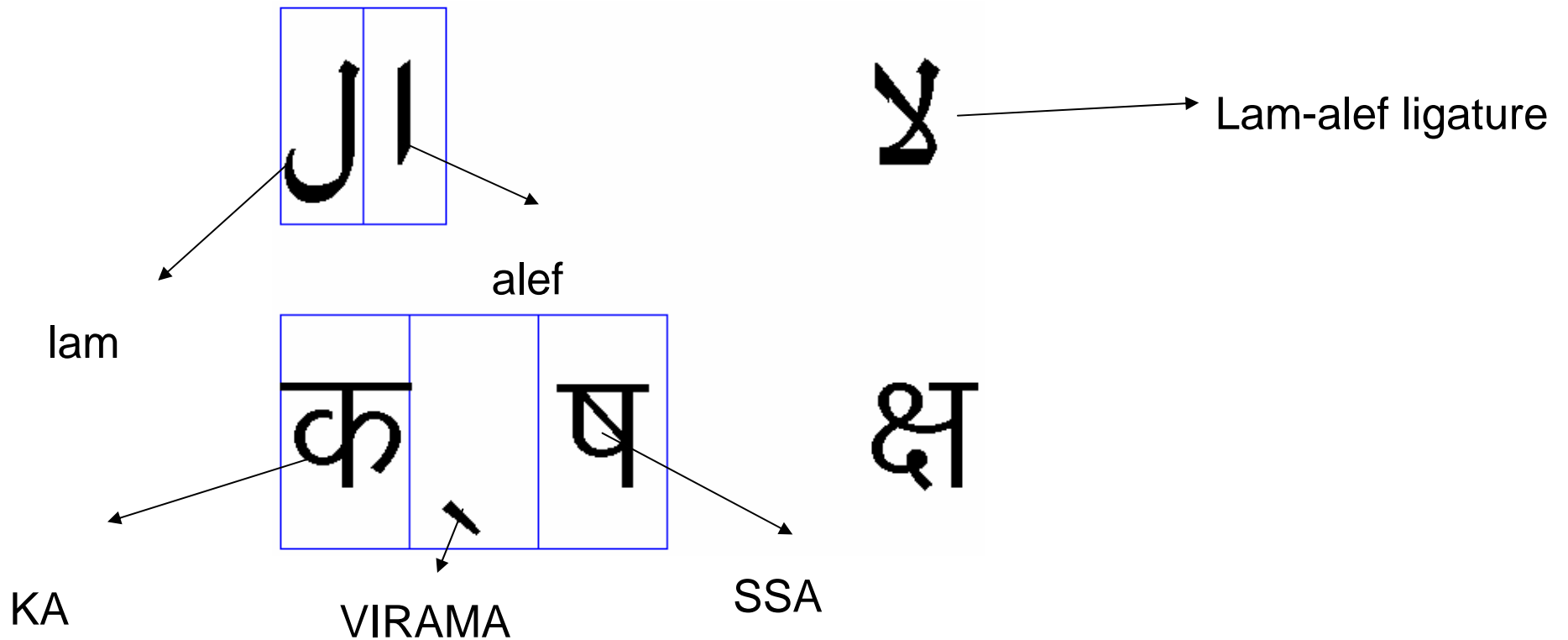
Character Shaping

- Arabic character shapes change to connect adjacent characters



Ligatures

- Arabic and Devanagari represent some character sequences with ligatures



Character Positioning

- Thai (and other scripts) require characters to reposition

MAI THO



KO KAI



SARA UEE



Reordering

- Some Hindi characters reorder based on context

Logical Order

प	त	दि	व
---	---	----	---

Visual Order

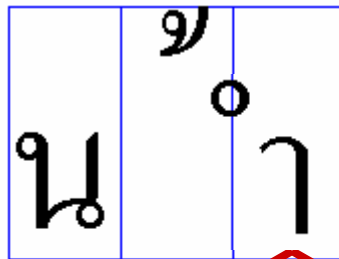
प	ति	द	व
---	----	---	---

A red 'X' with two arrows pointing from the top row to the bottom row, indicating the reordering of characters. The top row is 'पतदि व' and the bottom row is 'पतिद व'. The arrows point from the 'त' and 'दि' characters in the top row to the 'ति' and 'द' characters in the bottom row.

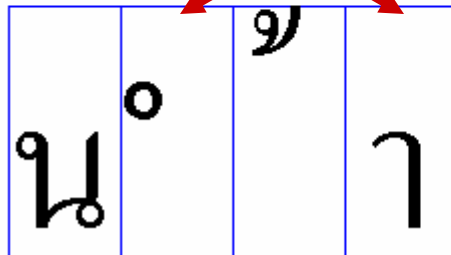
Split Characters

- Thai and many Indic languages display a single character in multiple positions

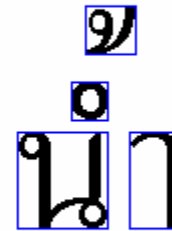
Logical Characters



Visual Glyphs



Displayed Result



Types of rendering/ Layout engine?

- Uniscribe

- This is the rendering engine used by the Microsoft software.

- Pango

- Pan in Greek means “all” and go in Japanese means “language”.
- It is an Open-source framework for the layout and rendering of internationalized text. Gnome applications use it for rendering.

- ICU Layout engine

- ICU stands for International component for Unicode. Maintained by IBM and this rendering engine is being used in Open office application.
-

Prerequisite.

- ❑ The particular script should be supported by the software. Unicode & ISO 10646 Standards.
 - ❑ A working font for that script should exist. Open type fonts are preferred.
 - ❑ A keyboard driver for that script should be developed
-

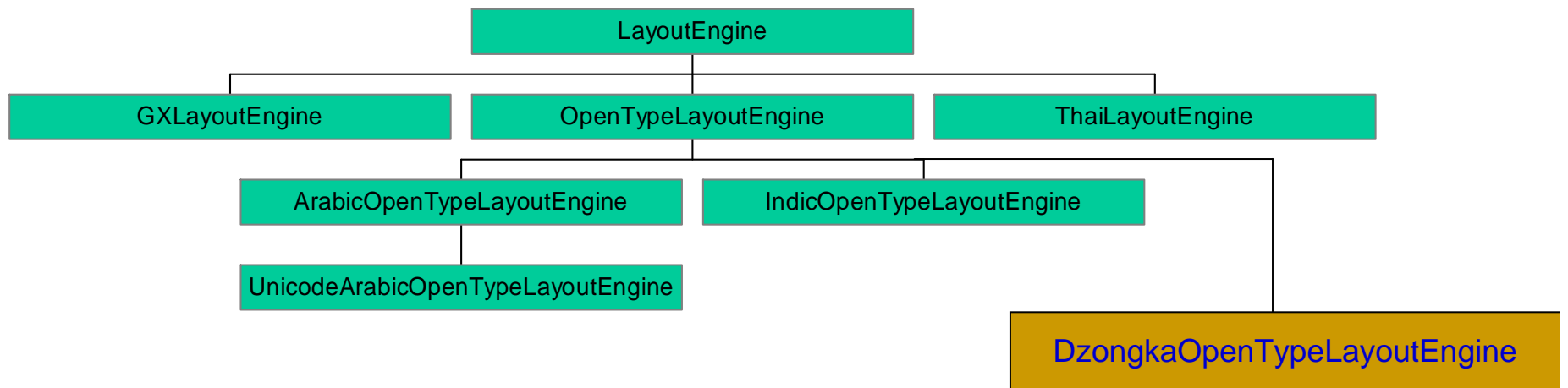
Overview on working of Layout Engine

- The font for a particular script contains rules.
 - Two main categories called “GPOS” (glyph positioning) and “GSUB” (glyph substitution).
 - There are features like “ccmp” (composition and decomposition), “blws” (below base substitution) etc. falling under GSUB rule. Other features like “blwm” (below base mark positioning), “abvm” (above base mark positioning) “kern” etc. fall under GPOS rule.
 - The fonts may contain language tags for the languages they support.
 - All combinations of characters used by particular languages are accessed by rules or lookups defined in the fonts.
 - The rendering engine has to identify the script, select the fonts, apply correct rules from the fonts and display it.
-

working of Layout Engine

- User input is stored in a buffer/memory.
 - Identify a script by looking at the Unicode values in the buffer.
 - Determine the bidirectional levels for the text.
 - Update the language tag using information.
 - Determine a language engine from the updated language tag and script.
 - Determine a set of possible fonts from the updated language tag and the font properties for the character. These fonts are sorted according to how well they match the language tag and font properties.
 - Apply the rules defined in the font to the Unicode values stored in the buffer.
 - Do character, word, line boundary analysis.
 - The output of this process is usually per line. These are then fed into the renderer.
-

LayoutEngine Class Hierarchy in ICU



How does it support Dzongkha Text

- Encoding Model for Dzongkha script
- OpenType Features for Dzongkha Fonts



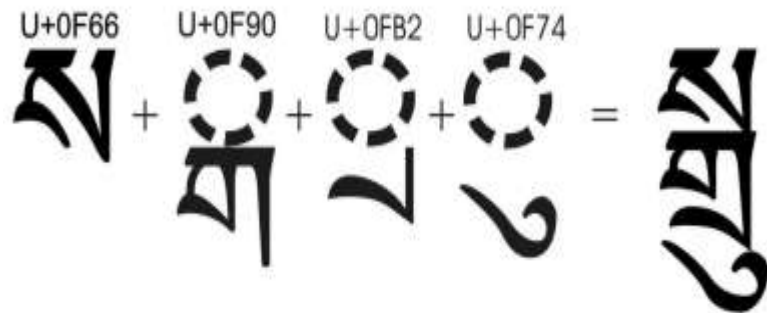
Encoding Model for Dzongkha script

■ Regular & Combining Consonants

- Vertically combined conjuncts of consonants and vowels.
 - Neighboring characters should stack vertically or be written left to right, not always determined by contextual or grammatical rules.
 - explicitly stacking model. In UCS two complete sets of consonants are encoded as separate characters.
 - i.e headline consonant characters [U+0F40-U+0F6A] , and combining consonant characters [U+0F90–U+0FBC]
-

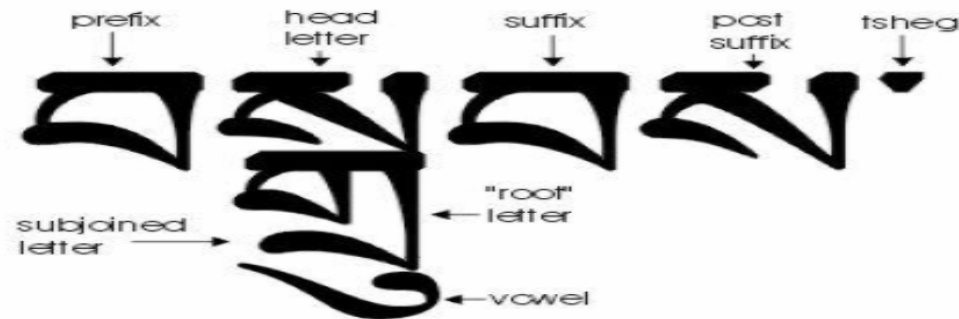
■ Character Order

- ❑ Conjunct stacks are encoded in the order in which the parts are written.
- ❑ consonant in the topmost or headline position , followed by characters for any combining consonants and then by the character(s) for any vowel(s).



■ Syllables & Encoding

- ❑ The basic unit of meaning or morpheme in Dzongkha is the *tsheg bar* usually referred to as a “syllable”.
- ❑ Each syllable contains a root letter (*ming zhi*) and may additionally have any/or all of the following parts: prefix, head letter, sub-fixed letter, vowel sign, suffix, and post-suffix.
- ❑ Syllables are normally delimited by a *tsheg* or another punctuation character.
- ❑ There are no inter-word spaces in Dzongkha



■ Special Characters

- U+0F0C NON BREAKING *TSHEG*.
 - *In case of a *tsheg* occurring after the letter *nga* and before a *shad*, it is desirable to suppress this behavior.*
- U+0F6A FIXED FORM *RA*.
 - override the normal contextual shaping of RA

Normal RA

A large, bold Tibetan character 'RA' in its normal form, which is a stylized '3' shape with a horizontal bar at the top.

U+0F62
+
U+0FA0







Fixed Form RA

A large, bold Tibetan character 'RA' in its fixed form, which is a stylized '3' shape with a horizontal bar at the top, but with a different internal structure than the normal form.

U+0F6A
+
U+0FA0

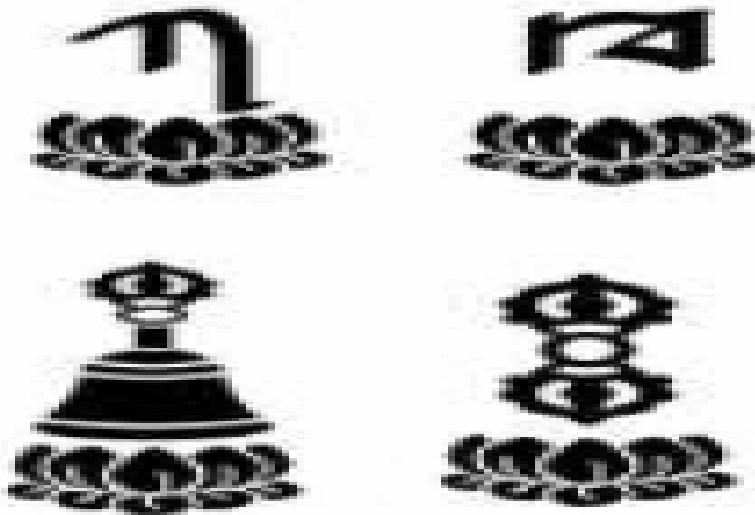
ཀ ཁ ར ལ ཅ ཆ ཇ ཉ ཊ ཋ ཌ ཌྷ ཏ

- U+0FBA, U+0FBB, U+0FBC: FIXED FORM SUB-JOINED WA, YA & RA.
 - WA YA and RA occurring mid-stack are often normally written in their full form.

Normal			
	U+0F5D U+0FAD	U+0F61 U+0FB1	U+0F62 U+0FB2
Fixed Form			
	U+0F5D U+0FBA	U+0F61 U+0FBB	U+0F62 U+0FBC

- U+0FC6 DZONGKHA SYMBOL *PADMA GDAN*

- This is an unusual *combining* symbol character - it may be used to combine with letters or other symbols.



OpenType Features for Dzongkha Fonts

- An Open Type shaping engine for Dzongkha processes text in stages:
 - 1. Analyzing syllables.
 - 2. Identification of correct cluster of characters.
 - 3. Shaping (substituting) glyphs using GSUB features & lookups in the font
 - 4. Positioning glyphs using GPOS features & lookups in the font.
-

-
- ❑ The Dzongkha syllable strings of UCS characters, in a sequence.
 - ❑ These characters are not necessarily ordered within the sequence.
 - ❑ The shaping engine first needs to identify the first consonant.
 - ❑ Identification of the correct stacks.
 - ❑ shaping engine apply contextual shaping or glyph substitution (GSUB) features to the glyph string.
 - ❑ applies OpenType positioning (GPOS) features to position glyphs.
-

■ SHAPING FEATURES:

- ❑ Glyph Composition Decomposition:
 - Apply lookups under '*ccmp*' feature
 - ❑ Conjuncts:
 - Apply lookups under '*b/lws*' feature to create conjuncts or ligatures
 - ❑ Below-base Marks:
 - Apply additional lookups under '*b/lws*' to get any additional below-base combining consonants and any below-base vowel marks; and other below-base marks.
 - ❑ Above-base Marks:
 - Apply lookups under '*abvs*' feature to get any above-base vowel conjuncts; above-base vowel modifiers; and above-base marks.
-

Refernces

- Pango :
www.pango.org
 - Uniscribe:
<http://www.microsoft.com/typography/developers/uniscribe/default.htm>
 - ICU:
<http://oss.software.ibm.com/icu>
 - OpenType Specifications:
<http://www.microsoft.com//typography/tt/tt.htm>
 - TrueType Font File Specification:
<http://fonts.apple.com/TTRefMan/RM06/Chap6.html>
-