Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia
technology

เอกสาร

รายงานผลการปรับแก้ด้านภาษาไทย ใน Qt/KDE และ Applications

เป็นส่วนหนึ่งของโครงการ

Suriyan Live CD

เสนอ

สำนักงานส่งเสริมอุตสาหกรรมซอฟต์แวร์แห่งชาติ
(องค์การมหาชน)

โดย

บริษัท เมตามีเดีย เทคโนโลยี จำกัด

วันที่ 27 กุมภาพันธ์ 2550

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia
technology

สารบัญ

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia
technology

บทนำ

เอกสารฉบับนี้นำเสนอการสรุปผลการปรับแก้ในด้านภาษาไทยใน Qt/KDE Version 4 (เวอร์ชั่นที่ใช้ทดสอบคือ Qt 4.3.2, KDE 4.0.1) โดยจะนำเสนอปัญหาและวิธีการแก้ไขปัญหาในส่วนต่างๆ

การสนับสนุนภาษาไทยพื้นฐานใน Qt Library และ KDE

การแสดงผลภาษาไทย

KDE/Qt 4 สามารถแสดงฟอนต์ภาษาไทยทั่วๆ ไปได้ แต่ยังมีปัญหาในรายละเอียดการแสดงตำแหน่งสระและวรรณยุกต์ในกรณีต่างๆ ดังนี้

1. กรณีที่วรรณยุกต์หรือสระบนผสมกับพยัญชนะที่มีหางบนยาว

กรณีที่ถูกต้อง คือ วรรณยุกต์หรือสระบนต้องหลบหางพยัญชนะหางบนยาว

กรณีที่ผิด เช่น วรรณยุกต์หรือสระบนไม่มีการหลบหาง พยัญชนะที่มีหางบนยาว

KDE/Qt 4

Loma	ป่า เป็น ปี่ ฝ้า ฟ้า ป่า ปัด ฝึก	ถูก	วรรณยุกต์หรือสระบนมีการหลบหางพยัญชนะที่มีหางบนยาว
Tahoma	ป่า เป็น ปี่ ฝ้า ฟ้า ป่า ปัด ฝึก	ผิด	วรรณยุกต์หรือสระบนไม่มีการหลบหาง พยัญชนะที่มีหางบนยาว

Gnome/Gtk+

Loma	ป่า เป็น ปี่ ฝ้า ฟ้า ป่า ปัด ฝึก	ถูก	วรรณยุกต์หรือสระบนมีการหลบหางพยัญชนะที่มีหางบนยาว
Tahoma	ป่า เป็น ปี่ ฝ้า ฟ้า ป่า ปัด ฝึก	ถูก	วรรณยุกต์หรือสระบนมีการหลบหางพยัญชนะที่มีหางบนยาว

Windows XP

Loma	ป่า เป็น ปี่ ฝ้า ฟ้า ป่า ปัด ฝึก	ถูก	วรรณยุกต์หรือสระบนมีการหลบหางพยัญชนะที่มีหางบนยาว

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th

Tahoma	ป่า เป็น ปี่ ฝ้า ฟ้า ป่า ปัด ฝึก	ถูก	วรรณยุกต์หรือสระบนมีการหลบหางพยัญชนะที่มีหางบนยาว

2. กรณีที่สระผสมกับพยัญชนะไม่มีหาง

กรณีที่ถูกต้อง คือ วรรณยุกต์ต้องมีการดึงขึ้นเมื่อมีการผสมกับสระบน

กรณีที่ผิด เช่น วรรณยุกต์และสระบนซ้อนกัน

KDE/Qt 4

Loma	กิ กี่ รึ ถือ ตัด อิ่ม บำ อี๋ จี๊	ถูก	วรรณยุกต์มีการดึงขึ้นเมื่อมีการผสมกับ สระบน
Tahoma	กิ กี่ รึ ถือ ตัด อิ่ม บำ อี๋ จี๊	ถูก	วรรณยุกต์มีการดึงขึ้นเมื่อมีการผสมกับ สระบน

Gnome/Gtk+

Loma	กิ กี่ รึ ถือ ตัด อิ่ม บำ อี๋ จี๊	ถูก	วรรณยุกต์มีการดึงขึ้นเมื่อมีการผสมกับ สระบน
Tahoma	กิ กี่ รึ ถือ ตัด อิ่ม บำ อี๋ จี๊	ถูก	วรรณยุกต์มีการดึงขึ้นเมื่อมีการผสมกับ สระบน

Windows XP

Loma	กิ กี่ รึ ถือ ตัด อิ่ม บำ อี๋ จี๊	ถูก	วรรณยุกต์มีการดึงขึ้นเมื่อมีการผสมกับ สระบน
Tahoma	กิ กี่ รึ ถือ ตัด อิ่ม บำ อี๋ จี๊	ถูก	วรรณยุกต์มีการดึงขึ้นเมื่อมีการผสมกับ สระบน

3. กรณีที่สระล่างผสมกับพยัญชนะที่มีหางล่างยาว

กรณีที่ถูกต้อง คือ สระล่างต้องถูกจัดระดับลงล่างเพื่อให้ หลบหางล่างของพยัญชนะ

กรณีที่ผิด เช่น สระล่างไม่กดลงต่ำเพื่อหลบหางล่างของพยัญชนะ

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia technology

KDE/Qt

Loma	กฎุม ฎู ฎู	ถูก	สระล่างต้องถูกจัดระดับลงล่างเพื่อให้หลบหางล่างของพยัญชนะ
Tahoma	กฎุม ฎู ฎู	ถูก	สระล่างต้องถูกจัดระดับลงล่างเพื่อให้หลบหางล่างของพยัญชนะ

Gnome/Gtk+

Loma	กฎุม ฎู ฎู	ถูก	สระล่างต้องถูกจัดระดับลงล่างเพื่อให้หลบหางล่างของพยัญชนะ
Tahoma	กฎม ฎู ฎู	ผิด	สระล่างไม่กดลงต่ำเพื่อหลบหางล่างของ พยัญชนะ

Windows XP

Loma	กฎม ฎ ฎ	ผิด	สระล่างไม่กดลงต่ำเพื่อหลบหางล่างของ พยัญชนะ
Tahoma	กฎม ฎ ฎ	ถูก	สระล่างไม่กดลงต่ำเพื่อหลบหางล่างของ พยัญชนะ

4. กรณีที่ในคำนั้นมีวรรณยุกต์ที่ไม่ได้มีการผสมกับพยัญชนะหางบนยาวหรือสระบน จะทำให้สระหรือวรรณยุกต์อื่นๆในคำนั้นลอย

กรณีที่ถูกต้อง คือ วรรณยุกต์ต้องอยู่ในตำแหน่งต่ำตั้งแต่แรก

กรณีที่ผิด เช่น วรรณยุกต์หรือสระบนลอยในคำที่ไม่ได้มีการผสมกับพยัญชนะหางบนยาวหรือสระบน

KDE/Qt

Loma	ด้ายสี ด้ายสีสี่	ผิด	วรรณยุกต์หรือสระบนลอยในคำที่ไม่ได้มีการผสมกับพยัญชนะหางบนยาวหรือสระบน
Tahoma	ด้ายสี ด้ายสีสี่	ถูก	วรรณยุกต์หรือสระบนไม่ลอย

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia technology

Gnome/Gtk+

Loma	ด้ายสี ด้ายสีสี่	ถูก	วรรณยุกต์หรือสระบนไม่ลอย
Tahoma	ด้ายสี ด้ายสีสี่	ถูก	วรรณยุกต์หรือสระบนไม่ลอย

Windows XP

Loma	ด้ายสี ด้ายสีสี่	ถูก	วรรณยุกต์หรือสระบนไม่ลอย
Tahoma	ด้ายสี ด้ายสีสี่	ถูก	วรรณยุกต์หรือสระบนไม่ลอย

5. กรณีที่สระ อำ ผสมกับวรรณยุกต์ วรรณยุกต์จะไม่ถูกปรับให้อยู่ในตำแหน่งสูง และถ้ามีวรรณยุกต์ขี่สระในคำนั้นๆ จะทำให้วรรณยุกต์ถูกกดลงต่ำ

กรณีที่ถูกต้อง คือ วรรณยุกต์ถูกดึงขึ้นเมื่อมีการผสมกับสระบน

กรณีที่ผิด เช่น วรรณยุกต์ไม่ถูกดึงขึ้นเมื่อมีการผสมกับสระบน

KDE/Qt

Loma	น้ำน่ำน้ำ น้ำน่ำน้ำที่	ผิด	วรรณยุกต์ไม่ถูกดึงขึ้นเมื่อมีการผสมกับ สระบน
Tahoma	น้ำน่ำน้ำ น้ำน่ำน้ำที่	ผิด	วรรณยุกต์ไม่ถูกดึงขึ้นเมื่อมีการผสมกับ สระบน

Gnome/Gtk+

Loma	น้ำน่ำน้ำ น้ำน่ำน้ำที่	ถูก	วรรณยุกต์ถูกดึงขึ้นเมื่อมีการผสมกับสระบน
Tahoma	น้ำน่ำน้ำ น้ำน่ำน้ำที่	ถูก	วรรณยุกต์ถูกดึงขึ้นเมื่อมีการผสมกับสระบน

Windows XP

Loma	น้ำน่ำน๊ำ น้ำน่ำน๊ำที่	ถูก	วรรณยุกต์ถูกดึงขึ้นเมื่อมีการผสมกับสระบน
Tahoma	น้ำน่ำน๊ำ น้ำน่ำน๊ำที่	ถูก	วรรณยุกต์ถูกดึงขึ้นเมื่อมีการผสมกับสระบน

6. กรณีที่มีวรรณยุกต์ในคำนั้นๆ จะทำให้สระล่างถูกกดลงต่ำ ยกเว้นวรรณยุกต์นั้นผสมกับพยัญชนะมีหางบนยาว

กรณีที่ถูกต้อง คือ สระล่างจะต้องอยู่ตำแหน่งเดิมตลอด

กรณีที่ผิด เช่น สระล่างถูกกดลงต่ำ เมื่อมีวรรณยุกต์ ในคำนั้น ยกเว้นวรรณยุกต์ผสมกับ พยัญชนะหางบนยาว

KDE/Qt

Loma	ถูบาน ถูบ้าน ถูบ้านที่ ถูฝา ถูฝ้า	ผิด	สระล่างถูกกดลงต่ำ เมื่อมีวรรณยุกต์ในคำนั้น ยกเว้นวรรณยุกต์ผสมกับพยัญชนะหางบนยาว
Tahoma	ถูบาน ถูบ้าน ถูบ้านที่ ถูฝา ถูฝ้า	ถูก	สระล่างไม่ถูกกดลงต่ำ

Gnome/Gtk+

Loma	ถูบาน ถูบ้าน ถูบ้านที่ ถูฝา ถูฝ้า	ถูก	สระล่างไม่ถูกกดลงต่ำ
Tahoma	ถูบาน ถูบ้าน ถูบ้านที่ ถูฝา ถูฝ้า	ถูก	สระล่างไม่ถูกกดลงต่ำ

Windows XP

Loma	ถูบาน ถูบ้าน ถูบ้านที่ ถูฝา ถูฝ้า	ถูก	สระล่างไม่ถูกกดลงต่ำ
Tahoma	ถูบาน ถูบ้าน ถูบ้านที่ ถูฝา ถูฝ้า	ถูก	สระล่างไม่ถูกกดลงต่ำ

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th

7. กรณีที่ ตัว ญ, ฐ ผสมกับสระล่าง เมื่อทำการ normalize แล้วสระล่างจะถูกกดลงต่ำ

กรณีที่ถูกต้อง คือ เมื่อทำการ normalize แล้วสระล่างอยู่ในระดับปกติ

กรณีที่ผิด เช่น เมื่อทำการ normalize แล้วสระล่างจะถูกกดลงต่ำ หรือไม่มีการ normalization

KDE/Qt

Loma	วิญญู ทุฏฐุลละ ทุก	ผิด	เมื่อทำการ normalize แล้วสระล่างจะถูกกดลงต่ำ
Tahoma	วิญญู ทุฏฐุลละ ทุก	ผิด	ไม่มีการ normalization ตัว ญ, ฐ

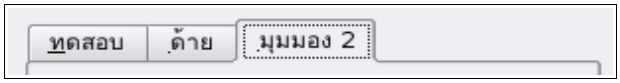
Gnome/Gtk+

Loma	วิญญู ทุฏฐุลละ ทุก	ถูก	เมื่อทำการ normalize แล้วสระล่างอยู่ในระดับปกติ
Tahoma	วิญญู ทุฏฐุลละ ทุก	ถูก	เมื่อทำการ normalize แล้วสระล่างอยู่ในระดับปกติ

Windows XP

Loma	วิญญู ทุฏฐุลละ ทุก	ถูก	เมื่อทำการ normalize แล้วสระล่างอยู่ในระดับปกติ
Tahoma	วิญญู ทุฏฐุลละ ทุก	ถูก	เมื่อทำการ normalize แล้วสระล่างอยู่ในระดับปกติ

8. กรณีการแสดงผล shortcut (แสดงโดยขีดเส้นใต้) ที่เป็นภาษาไทย ถ้าเกิดว่ามีสระบนหรือสระล่าง อยู่ด้วยจะไม่มีการขีดเส้นใต้



Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia
technology

สรุปปัญหาและวิธีการแก้ไขในส่วนการแสดงผล

ปัญหา	วิธีการแก้ไข
1. ในคำที่มีวรรณยุกต์ที่ไม่ได้มีการผสมกับพยัญชนะหางบนยาวหรือสระบน จะทำให้สระหรือวรรณยุกต์อื่นๆในคำนั้นลอย	ดูที่หัวข้อการแก้ไขที่ 1.1
2. การแสดงผลวรรณยุกต์เมื่อมีการผสมกับ สระ อำ ทำวรรณยุกต์จะไม่ถูกปรับ ให้อยู่ในตำแหน่งสูง	
3. การแสดงสระล่างถูกกดลงต่ำ เมื่อมีวรรณยุกต์ในคำนั้นๆ ยกเว้นวรรณยุกต์ผสมกับพยัญชนะมีหางบนยาว	
4. การแสดงผลของตัว ญ, ฐ เมื่อมีการผสมกับสระล่าง เมื่อทำการ normalize แล้วสระล่างถูกกดลงต่ำ	
5. ใน font Tahoma วรรณยุกต์หรือสระบนไม่มีการหลบหาง พยัญชนะที่มีหางบนยาว	ดูที่หัวข้อการแก้ไขที่ 1.1
6. ใน font Tahoma ไม่มีการ normalization ตัว ญ, ฐ เมื่อผสมกับสระล่าง	
7. กรณีการแสดงผล shortcut (แสดงโดยขีดเส้นใต้) ที่เป็นภาษาไทย ถ้าเกิดว่ามีสระบนหรือสระล่าง อยู่ด้วยจะไม่มีการขีดเส้นใต้	ดูที่หัวข้อการแก้ไขที่ 1.2

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia technology

Input Method

Input Method ผ่าน XIM

KDE/Qt

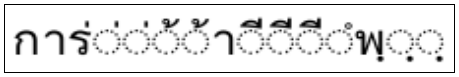
1. เมื่อลำดับของการอินพุตผิดจะเตือนด้วยเสียงและไม่มีการ normalization
2. การย้อนกลับมาใส่วรรณยุกต์หรือสระที่มีตำแหน่งอยู่ด้านบนและด้านล่าง ในตำแหน่งที่ cursor พิมพ์ผ่านมาจะใส่ทันทีไม่ได้จะต้องลบตัวอักษรแล้วพิมพ์อักษรเข้าไปใหม่จึงจะใส่ได้

**หมายเหตุ

ต้องปิด NumLock ก่อนถึงจะใช้ XIM ได้(ไม่รู้ทำไม)

Gnome/Gtk+

1. เมื่อลำดับของการอินพุตผิดจะพิมพ์วรรณยุกต์นั้นๆ ใน cell ใหม่



2. สามารถย้อนกลับมาใส่วรรณยุกต์หรือสระบน-ล่าง ในตำแหน่งที่ cursor ผ่านมาได้

วิธีการแก้ไข

ดูที่หัวข้อการแก้ไขที่ 2

Input Method โดยใช้ SCIM

KDE/Qt

1. เมื่อลำดับของการอินพุตผิดจะเตือนด้วยเสียงและในตำแหน่งที่สามารถทำ normalization ได้ก็จะทำการแก้ไขให้ถูกต้องให้
2. สามารถย้อนกลับมาใส่วรรณยุกต์หรือสระบน-ล่าง ในตำแหน่งที่ cursor พิมพ์ผ่านมาได้และในตำแหน่งที่สามารถทำ normalization ได้ก็จะทำการแก้ไขให้ถูกต้องให้

Gnome/Gtk+

1. เมื่อลำดับของการอินพุตผิดจะเตือนด้วยเสียงและในตำแหน่งที่สามารถทำ normalization ได้ก็จะทำการแก้ไขให้ถูกต้องให้
2. สามารถย้อนกลับมาใส่วรรณยุกต์หรือสระบน-ล่าง ในตำแหน่งที่ cursor พิมพ์ผ่านมาได้และในตำแหน่งที่สามารถทำ normalization ได้ก็จะทำการแก้ไขให้ถูกต้องให้

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

Input Method ใน WindowsXP (NotePad)

1. เมื่อลำดับของการอินพุตผิดจะเตือนด้วยเสียงและไม่มีการ normalization
2. การย้อนกลับมาใส่วรรณยุกต์หรือสระที่มีตำแหน่งบน-ล่าง ในตำแหน่งที่ cursor พิมพ์ผ่านมา สามารถใส่ได้ ยกเว้นใน cell ที่ใส่ไปแล้วทำให้เกิดลำดับการอินพุตผิด

การเลื่อน Cursor

KDE/Qt 4 มีปัญหาในการเลื่อน cursor และปัญหาในการใช้งานปุ่ม Delete, Backspace, Ctrl, Arrow ในกรณีต่างๆ ดังนี้

1. กรณีการเลื่อน cursor ใน cell ที่มีการผสมสระหรือวรรณยุกต์

KDE/Qt 4	cursor เลื่อนทีละอักขระ (เลื่อนเข้าสระหรือวรรณยุกต์ที่ผสมอยู่ด้วย)
Gnome/Gtk+	cursor เลื่อนทั้ง cell (ข้ามสระหรือวรรณยุกต์ที่ผสมอยู่)
Windows	cursor เลื่อนทั้ง cell (ข้ามสระหรือวรรณยุกต์ที่ผสมอยู่)

แบบที่ถูกต้อง คือ cursor ควรเลื่อนทั้ง cell

2. กรณีการกดปุ่ม Delete เมื่อ cursor อยู่หน้า cell ที่มีการผสมสระหรือวรรณยุกต์

KDE/Qt 4	ลบทีละตัวอักขระ (ลบที่ละสระหรือวรรณยุกต์ที่ผสมอยู่)
Gnome/Gtk+	ลบทั้ง cell ที่มีการผสมสระหรือวรรณยุกต์
Windows	ลบทั้ง cell ที่มีการผสมสระหรือวรรณยุกต์

แบบที่ถูกต้อง คือ ควรลบทั้ง cell ที่มีการผสมสระหรือวรรณยุกต์

3. กรณีการเลื่อน cursor เมื่อมีการกดปุ่ม Ctrl+Arrow Left-Right cursor

KDE/Qt 4	cursor เลื่อนไปตาม space
Gnome/Gtk+	cursor เลื่อนไปทีละคำ
Windows	cursor เลื่อนไปทีละคำ

แบบที่ถูกต้อง คือ cursor ควรเลื่อนไปทีละคำ

4. กรณีการใช้ Ctrl+Delete ในการลบคำที่อยู่หลัง cursor ทีละคำ

KDE/Qt 4	ลบตาม space
Gnome/Gtk+	ลบคำที่อยู่หลัง cursor ทีละคำ
Windows	ลบทั้งบรรทัด

แบบที่ถูกต้อง คือ ควรลบคำที่อยู่หลัง cursor ทีละคำ

5. กรณีการใช้ Ctrl+Backspace ในการลบคำที่อยู่หน้า cursor ทีละคำ

KDE/Qt 4	ลบตาม space
Gnome/Gtk+	ลบคำที่อยู่หน้า cursor ทีละคำ
Windows	ลบทั้งบรรทัด

แบบที่ถูกต้อง คือ ควรลบคำที่อยู่หน้า cursor ทีละคำ

สรุปปัญหาและวิธีการแก้ไข

ปัญหา	วิธีการแก้ไข
1. Cursor เลื่อนเข้าทีละอักขระใน cell ที่มีการผสม	ดูที่หัวข้อการแก้ไขที่ 3
2. Delete ทีละอักขระใน cell ที่มีการผสม	
3. Ctrl+Left-Right	
4. Ctrl+Delete	
5. Ctrl+Backspace	

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

การ copy และ paste ระหว่าง widget/applications

การ copy-paste ระหว่าง widget/applications ใน KDE/Qt 4 นั้นส่วนมากทำงานได้ปกติยกเว้นการ copy-paste ของ konsole ซึ่งจะ copy-paste ได้เฉพาะอักขระที่อยู่ใน Base level เท่านั้น วรรณยุกต์และ สระบน-ล่างไม่ถูก copy-paste

สรุปปัญหาและวิธีการแก้ไข

ปัญหา	วิธีแก้ไข
1. การ copy-paste ของ konsole จะ copy-paste ได้เฉพาะอักขระที่อยู่ใน Base level เท่านั้น	ดูที่หัวข้อการแก้ไขที่ 6

การพิมพ์

การพิมพ์ภาษาไทยออกทางเครื่องพิมพ์หรือสร้าง PS/PDF file จะให้ผลเหมือนกับการแสดงผล ผ่านทางหน้าจอ ซึ่งก็มีปัญหาการแสดงวรรณยุกต์ สระบน-ล่าง ลอยเช่นเดียวกัน

สรุปปัญหาและวิธีการแก้ไข

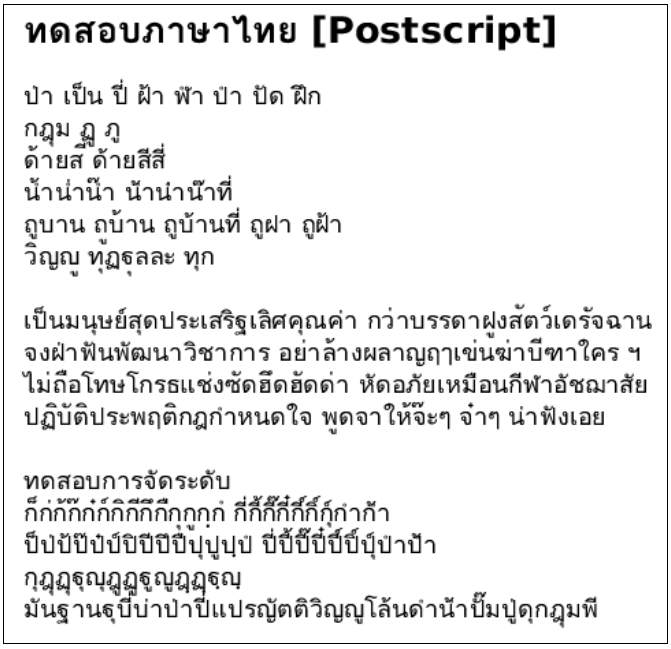
ปัญหา	วิธีแก้ไข
1. มีปัญหาการ render ภาษาไทยในการพิมพ์ในลักษณะเดียวกับปัญหาที่พบในการแสดงผลทางหน้าจอของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th

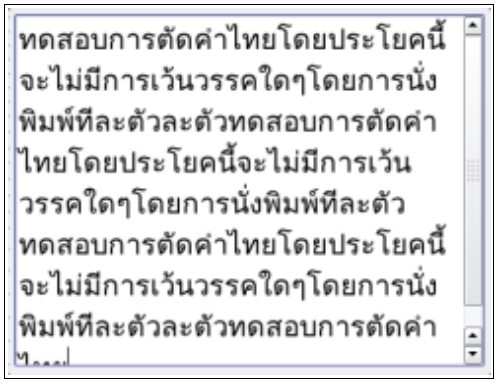


ภาพแสดงผลภาษาไทยจากไฟล์ Postscript

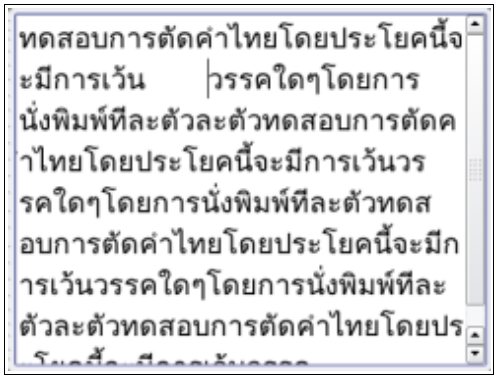
การตัดคำภาษาไทย

KDE/Qt 4 สามารถตัดคำไทยได้ แต่ยังมีกรณีที่ยังเกิดปัญหาทำให้ไม่มีการตัดคำ ดังนี้

1. ในกรณีที่ประโยคนั้นมี space การตัดคำจะผิดพลาด

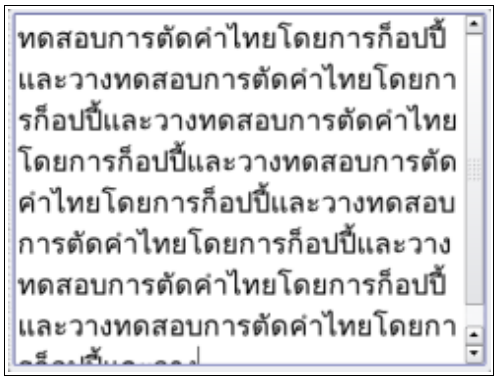


ไม่มี space ตัดคำได้ถูกต้อง



มี space การตัดคำผิดพลาด

2. ในกรณีที่มีการ copy-paste ในปริมาณที่มากจะทำให้การตัดคำผิดพลาด



สรุปปัญหาและวิธีการแก้ไข

ลำดับ	ปัญหา	วิธีแก้ไข
1.	ในกรณีที่ประโยคนั้นมี space	ดูที่หัวข้อการแก้ไขที่ 4
2.	กรณีที่มีการ copy-paste	

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia
technology

การสนับสนุนภาษาไทยพื้นฐานใน KDE Applications ที่สำคัญ

Dolphin และ Konqueror

การสนับสนุนภาษาไทยใน Dolphin และ Konqueror ส่วนมากจะมีปัญหาเดียวกันกับ ปัญหาที่เกิดขึ้นใน KDE/Qt 4 และปัญหาอื่นๆอีกเล็กน้อยโดยสรุปปัญหาและวิธีการแก้ไขดังนี้

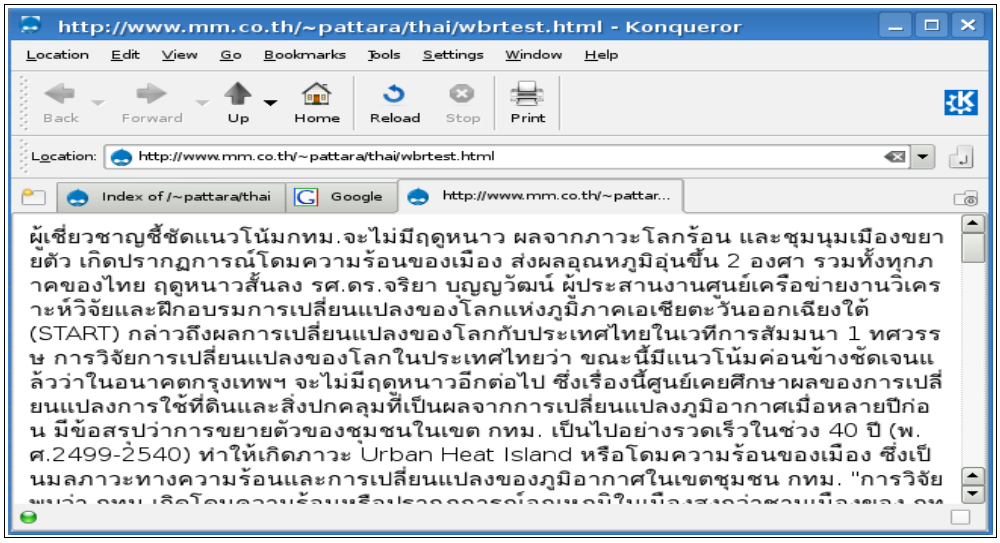
หัวข้อทดสอบ	ปัญหา	วิธีการแก้ไข
การแสดงผลภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
Input Method	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 2
การเลื่อน Cursor	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 3
การ Copy และ Paste	สามารถ copy-paste ได้ปกติ	-
การพิมพ์	การ render ภาษาไทยมีปัญหาเช่นเดียวกับการแสดงผลภาษาไทยใน KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
การตัดคำภาษาไทย	ไม่สามารถตัดคำไทยได้ (ไม่สามารถ load libthai ได้)	ดูที่หัวข้อการแก้ไขที่ 5

Metamedia Technology Co., Ltd.

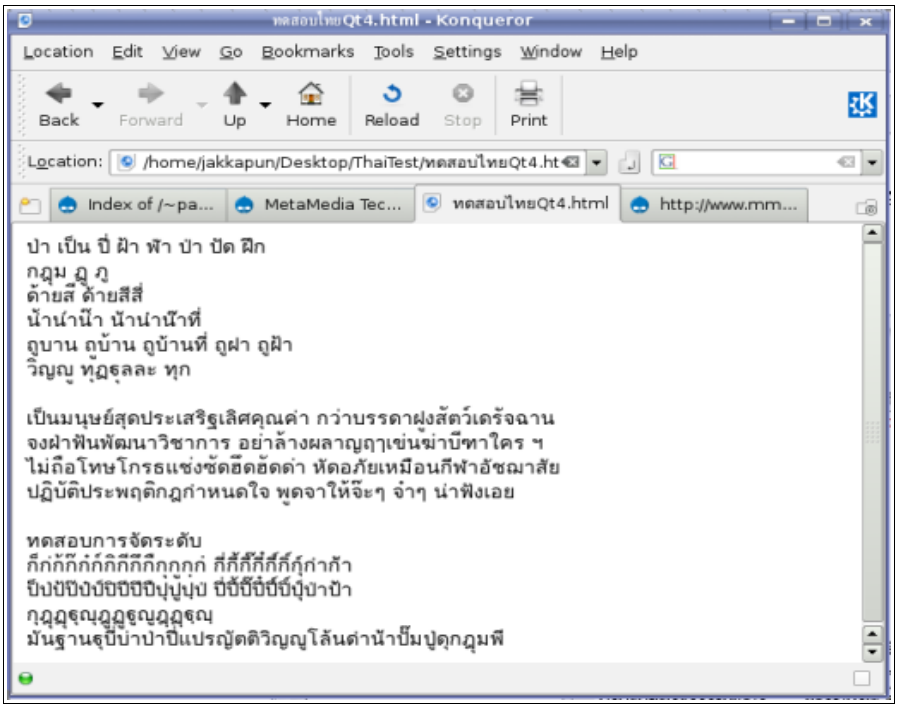
Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th



ภาพการตัดคำใน Konqueror

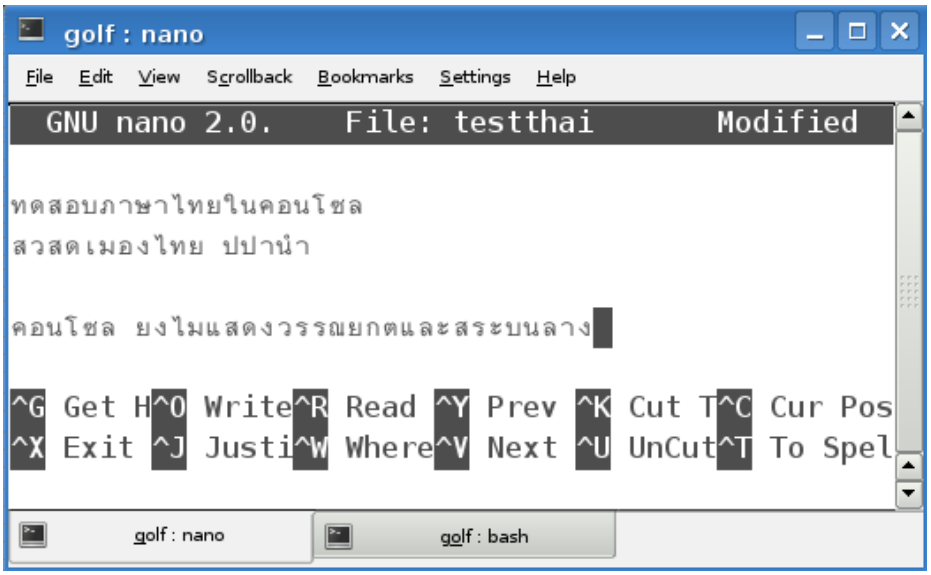


ภาพการแสดงผลใน Konqueror

Konsole

การสนับสนุนภาษาไทยใน Konsole ในหัวข้อต่างๆ สรุปปัญหาและวิธีการแก้ไขดังนี้

หัวข้อทดสอบ	ปัญหา	วิธีการแก้ไข
การแสดงผลภาษาไทย	ไม่มีการแสดงวรรณยุกต์และสระบน-สระล่าง	ดูที่หัวข้อการแก้ไขที่ 6
Input Method	-	-
การเลื่อน Cursor	-	-
การ Copy และ Paste	การ copy-paste จาก konsole ไปยัง application หรือจาก application อื่นมายัง konsole จะไม่ copy วรรณยุกต์และสระบน-ล่างมาด้วย	ดูที่หัวข้อการแก้ไขที่ 6
การพิมพ์	-	-
การตัดคำภาษาไทย	-	-

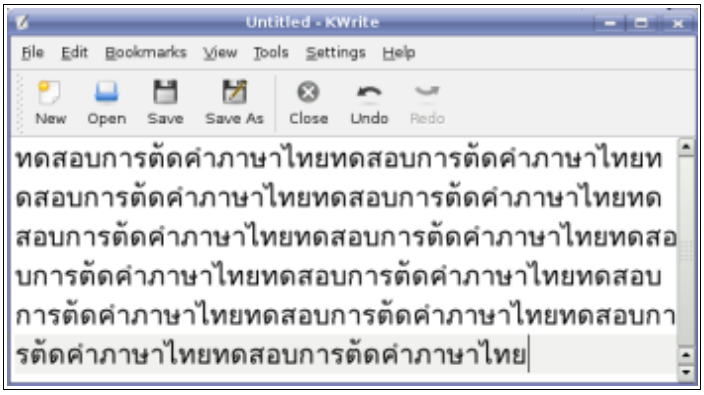


ภาพการแสดงผลภาษาไทยใน Konsole

Kate/Kwrite

การสนับสนุนภาษาไทยใน Kate/Kwrite ส่วนมากจะมีปัญหาเดียวกันกับ ปัญหาที่เกิดขึ้น ใน KDE/Qt 4 และปัญหาอื่นๆอีกเล็กน้อยโดยสรุปปัญหาและวิธีการแก้ไขดังนี้

หัวข้อทดสอบ	ปัญหา	วิธีการแก้ไข
การแสดงผลภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
Input Method	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 7
การเลื่อน Cursor	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 3
การ Copy และ Paste	สามารถ copy-paste ได้ปกติ	-
การพิมพ์	หน้าต่างการพิมพ์ไม่ขึ้น	-
การตัดคำภาษาไทย	ไม่มีการตัดคำไทย	-
อื่นๆ	ไม่สามารถเข้าเมนู print ได้	-



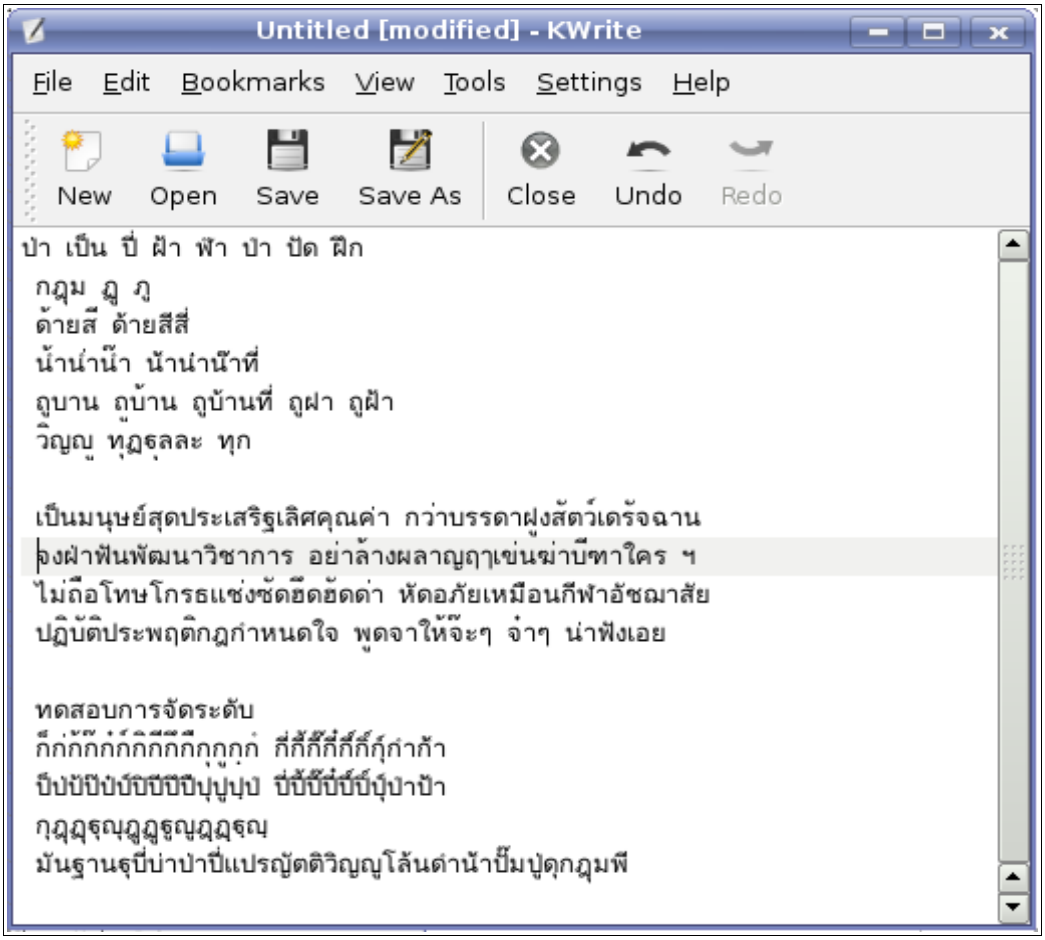
ภาพแสดงการตัดคำใน Kwrite

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500

Tel: +66-2-632-9700, Fax: +66-2-632 9703

Web: http://www.mm.co.th/, Email: info@mm.co.th



ภาพการแสดงผลภาษาไทยใน Kwrite

Metamedia Technology Co., Ltd.

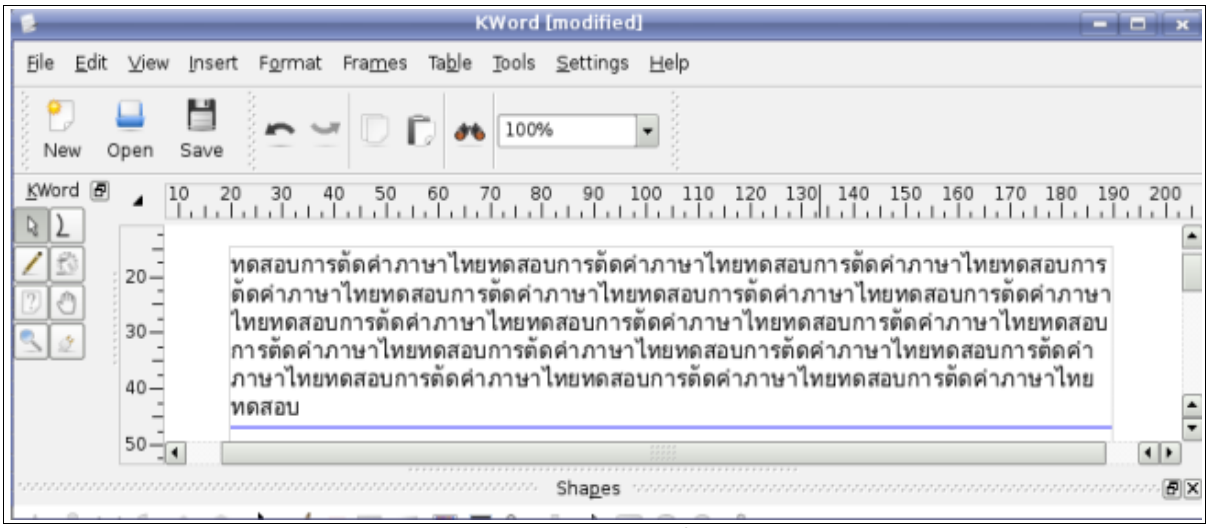
Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

KOffice

KWord

การสนับสนุนภาษาไทยใน Kword ส่วนมากจะมีปัญหาเดียวกันกับ ปัญหาที่เกิดขึ้น ใน KDE/Qt 4 และปัญหาอื่นๆอีกเล็กน้อยโดยสรุปดังนี้

หัวข้อทดสอบ	ปัญหา	วิธีการแก้ไข
การแสดงผลภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
Input Method	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 8
การเลื่อน Cursor	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 3
การ Copy และ Paste	สามารถ copy-paste ได้ปกติ	-
การพิมพ์	สั่ง print แล้วไม่เกิดอะไรขึ้นเลย	-
การตัดคำภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 4



ภาพการแสดงการตัดคำใน Kword

KSpread

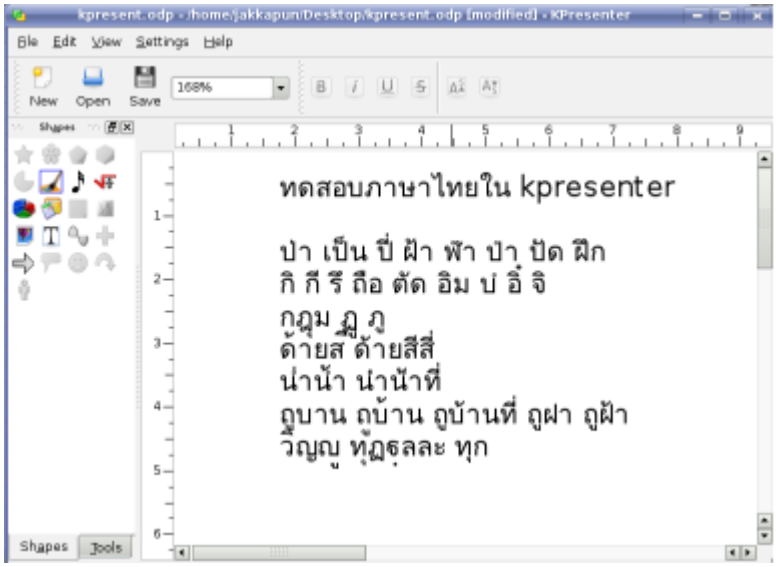
การสนับสนุนภาษาไทยใน KSpread ส่วนมากจะมีปัญหาเดียวกันกับ ปัญหาที่เกิดขึ้น ใน KDE/Qt 4 และปัญหาอื่นๆอีกเล็กน้อยโดยสรุปปัญหาและวิธีการแก้ไขดังนี้

หัวข้อทดสอบ	ปัญหา	วิธีการแก้ไข
การแสดงผลภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
Input Method	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 2
การเลื่อน Cursor	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 3
การ Copy และ Paste	สามารถ copy-paste ได้ปกติ	-
การพิมพ์	สั่ง print แล้วไม่เกิดอะไรขึ้นเลย	-
การตัดคำภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 4

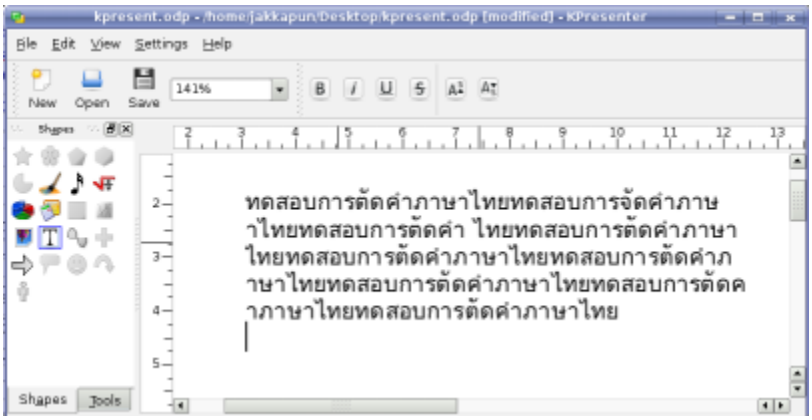
KPresenter

การสนับสนุนภาษาไทยใน KPresenter ส่วนมากจะมีปัญหาเดียวกันกับ ปัญหาที่เกิดขึ้น ใน KDE/Qt 4 และปัญหาอื่นๆอีกเล็กน้อยโดยสรุปปัญหาและวิธีการแก้ไขดังนี้

หัวข้อทดสอบ	ปัญหา	วิธีการแก้ไข
การแสดงผลภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
Input Method	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 2
การเลื่อน Cursor	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 3
การ Copy และ Paste	สามารถ copy-paste ได้ปกติ	-
การพิมพ์	สั่ง print แล้วไม่เกิดอะไรขึ้นเลย	-
การตัดคำภาษาไทย	มีปัญหาเหมือนกับในกรณีของ KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 4



ภาพการแสดงผลภาษาไทยใน KPresenter



ภาพการแสดงการตัดคำไทยใน KPresenter

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia
technology

สรุปปัญหาและวิธีการแก้ไขทั้งหมดที่ใช้ใน KDE/Qt4, KDE Applications

ปัญหา	วิธีการแก้ไข
1. การแสดงผล [KDE/Qt4]	
1.1 วรรณยุกต์ + พยัญชนะไม่มีหาง จะทำให้สระบนและวรรณยุกต์ทุกตัวลอย ในคำนั้น	ดูที่หัวข้อการแก้ไขที่ 1.1
1.2 สระ อำ + วรรณยุกต์ จะทำให้วรรณยุกต์ไม่ถูกปรับให้อยู่ในตำแหน่งสูง	
1.3 สระล่าง + วรรณยุกต์ จะทำให้สระล่างถูกดึงลงต่ำ	
1.4 ตัว ญ, ฐ + สระล่าง เมื่อทำการ normalization แล้วทำให้สระล่างถูกดึงลงต่ำ	
1.5 ใน font Tahoma วรรณยุกต์หรือ สระบนไม่มีการหลบหาง พยัญชนะที่มีหางบนยาว	ดูที่หัวข้อการแก้ไขที่ 1.1
1.6 ใน font Tahoma ไม่มีการ normalization ตัว ญ, ฐ เมื่อผสมกับสระล่าง	
1.7 กรณีการแสดงผล shortcut (แสดงโดยขีดเส้นใต้) ที่เป็นภาษาไทย ถ้าเกิดว่ามีสระบนหรือสระล่าง อยู่ด้วยจะไม่มีการขีดเส้นใต้	ดูที่หัวข้อการแก้ไขที่ 1.2
2. Input Method [KDE/Qt4]	
Input ผ่าน XIM	
2.1 เมื่อลำดับการอินพุตผิดจะไม่ให้ใส่หร้อมกับการเตือนด้วยเสียงไม่มีการ normalization	ดูที่หัวข้อการแก้ไขที่ 2
2.2 ไม่สามารถย้อนกลับมาใส่วรรณยุกต์หรือสระบน-ล่างได้	
3. การเลื่อน cursor [KDE/Qt4]	
3.1 Cursor เลื่อนเข้าทีละอัษรใน cell ที่มีการผสม	ดูที่หัวข้อการแก้ไขที่ 3
3.2 Delete ทีละอัษรใน cell ที่มีการผสม	
3.3 Ctrl+(Arrow Left-Right)	ดูที่หัวข้อการแก้ไขที่ 3

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

ปัญหา	วิธีการแก้ไข
3.4 Ctrl+Delete	
3.5 Ctrl+Backspace	
4. การ copy และ paste [KDE/Qt4]	
4.1 การ copy-paste กับ konsole จะไม่มีพวกวรรณยุกต์หรือสระบน-ล่าง ไปด้วย	ดูที่หัวข้อการแก้ไขที่ 6
5. การพิมพ์ภาษาไทย [KDE/Qt4]	
5.1 มีปัญหาการ render ภาษาไทยเช่นเดียวกับปัญหาการแสดงภาษาใน KDE/Qt 4	ดูที่หัวข้อการแก้ไขที่ 1
6. การตัดคำไทย [KDE/Qt4]	
6.1 ไม่ตัดคำในประโยคที่มี space	ดูที่หัวข้อการแก้ไขที่ 4
6.2 ไม่ตัดคำเมื่อมีการ copy-paste ข้อมูลมากๆ	
7. ปัญหาที่เกิดใน Application	
7.1 konqueror ไม่สามารถ load libthai ได้	ดูที่หัวข้อการแก้ไขที่ 5
7.2 ไม่มีการแสดงวรรณยุกต์หรือสระบน-ล่างใน konsole	ดูที่หัวข้อการแก้ไขที่ 6
7.3 การ input method ใน Kwrite	ดูที่หัวข้อการแก้ไขที่ 7
7.4 การ input method ใน Kword	ดูที่หัวข้อการแก้ไขที่ 8
7.5 ต้องปิด NumLock ก่อนถึงจะใช้ XIM ได้	เป็น bug ของ XIM แก้ไขโดยใช้ patch https://bugs.freedesktop.org/show_bug.cgi?id=12759

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

metamedia technology

การแก้ปัญหาภาษาไทยใน Qt Library

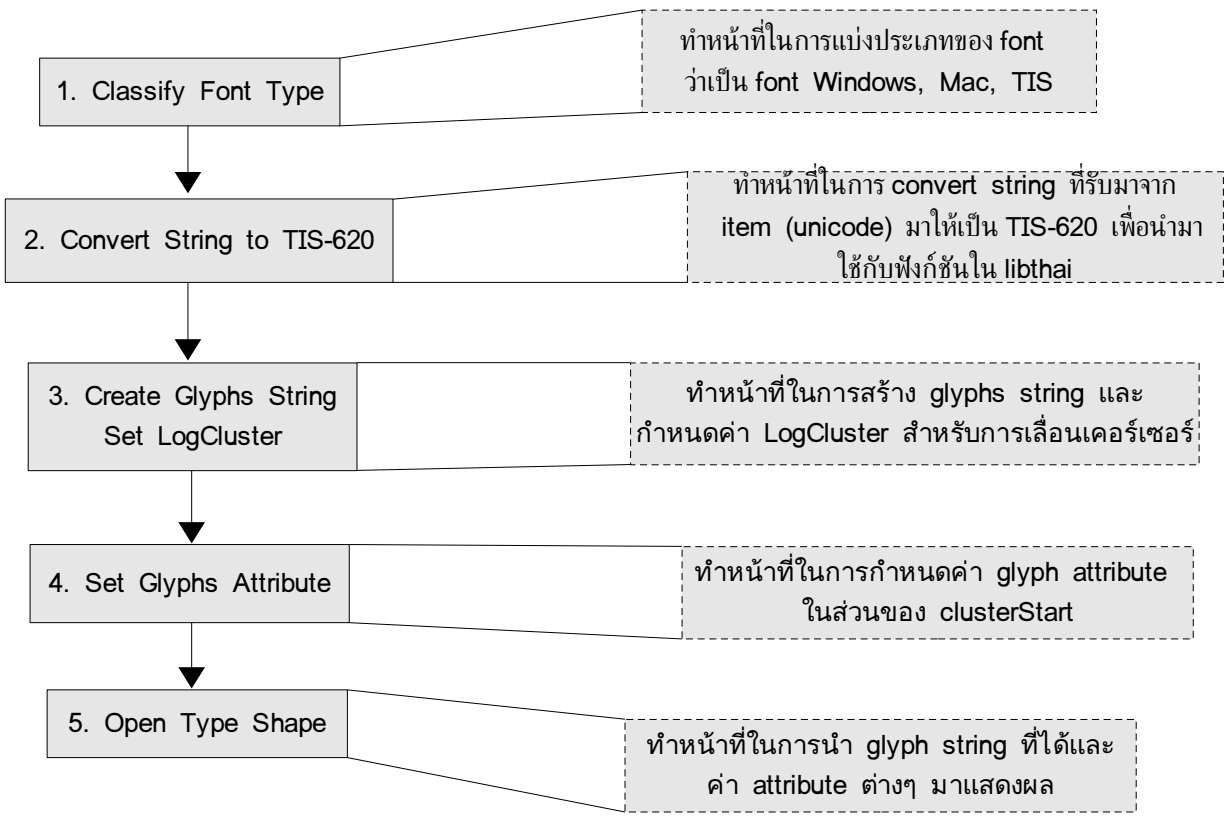
1. การแก้ปัญหาการแสดงผลภาษาไทยใน KDE4/Qt4

ปัญหาการแสดงผลภาษาไทยที่เกิดขึ้นใน Qt4 เกิดจากในไฟล์ /src/gui/text/qscriptengine.cpp ไม่มีฟังก์ชันในการแสดงผลภาษาไทยโดยตรงแต่ใช้ basic_shape ซึ่งเป็น default ในการแสดงผลของภาษาต่างๆ ที่ยังไม่มีฟังก์ชันในการแสดงผลของตัวเอง ดังนั้นเราจึงต้องสร้างฟังก์ชันการแสดงผลภาษาไทยโดยตรง (thai_shape) แทนการใช้ basic_shape ที่ Qt4 เตรียมมาให้

1.1 การ implement thai_shape(QShaperItem *item)

การ implement ฟังก์ชัน thai_shape(QShaperItem *item) จะทำที่ไฟล์ qt/src/gui/text/qscriptengine.cpp ช่วงบรรทัดที่ 3665-3781 โดยฟังก์ชันนี้จะทำหน้าที่ในการสร้าง glyph string เพื่อที่จะนำไปให้ opentype ทำการแสดงผล ซึ่ง glyph string นี้สร้างมาจาก string ที่ส่งผ่านมาทาง item และใช้ฟังก์ชันใน libthai ในการสร้าง

การทำงานของฟังก์ชัน thai_shape นี้จะแบ่งออกเป็นส่วนหลักได้ 5 ส่วน ดังนี้



1. Classify Font Type

ทำหน้าที่ในการแบ่งประเภทของฟอนต์ว่าเป็นฟอนต์ Windows, Mac, TIS โดยเราสร้างข้อมูลประเภท enum (บรรทัดที่ 3605-3609) มาเก็บประเภทของฟอนต์ WIN, MAC, TIS โดยให้ชื่อว่า ThaiFontType และเราจะต้องนำข้อมูลนี้เก็บเข้าไปเป็นข้อมูลของ item->font เพื่อความรวดเร็วในการจำแนกฟอนต์ ซึ่งจะทำให้เราไม่ต้องมาคำนวณ font type ใหม่ทุกครั้งที่มีการเรียก thai_shape โดยเราสามารถดึง attributes font_type มาจาก tem->font ได้เลย

```
+enum ThaiFontType {
+     TIS,
+     WIN,
+     MAC
+};
```

การเพิ่มข้อมูลเข้าไปใน item->font ทำโดยการเพิ่ม UserData เข้าไปใน item->font โดยวิธีการเราต้องสร้าง class ที่สืบทอดมาจาก QObjectUserData ซึ่งในที่นี้คือ ThaiFontUserData (บรรทัดที่ 3611-3616) ซึ่งเป็นชนิดข้อมูลที่จะถูกเพิ่มเข้าไปใน item->font ภายใน ThaiFontUserData จะประกอบไปด้วยเมมเบอร์ชนิด ThaiFontType ที่ใช้ในการเก็บประเภทของฟอนต์ และในการ set หรือ get ค่า UserData เราจะต้อง อ้างอิงหมายเลขที่เราทำการ register User Data จากฟังก์ชัน thaiFontTypeUDataID() (บรรทัดที่ 3645-3649) ฟังก์ชันนี้ทำการ registerUserData ให้กับ class QFontEngine และจะ return id ของ data นั้นกลับมาเพื่อใช้ในการอ้างอิง

```
+class ThaiFontUserData : public QObjectUserData
+{
+     public:
+     ThaiFontType font_type;
+     ThaiFontUserData(ThaiFontType font_t) : font_type(font_t) {}
+};
```

```
+static uint thaiFontTypeUDataID()
+{
+     static uint thaiFontTypeUDataID = QFontEngine::registerUserData();
+     return thaiFontTypeUDataID;
+}
```

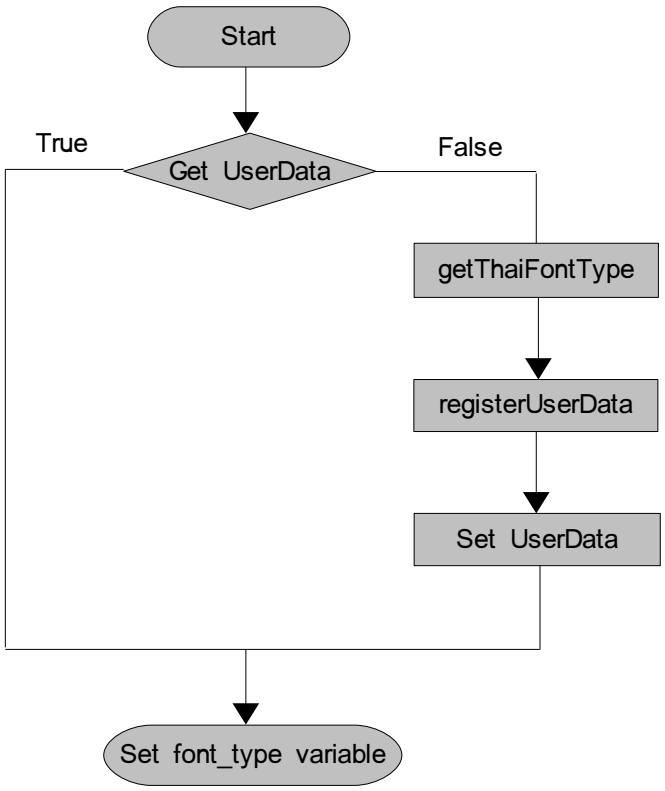
ขั้นตอนการทำงานจะเริ่มต้นจากการดึงค่า UserData ขึ้นมาจาก item->font->userData ก่อนถ้าไม่สามารถดึงได้ก็ให้ทำการคำนวณ font type และเซ็ตค่า UserData (item->font->setUserData) ให้กับ item->font หลังจากการเซ็ตค่านี้แล้วในการเรียก thai_shape ในครั้งถัดไปจะไม่ต้องมาคำนวณ font type ใหม่อีกครั้ง สามารถดึงได้จาก item->font->userData() ได้เลย

```
+#ifndef QT_NO_USERDATA
+     ThaiFontUserData *thaiFontData = reinterpret_cast<ThaiFontUserData *>(item->font-
>userData(thaiFontTypeUDataID()));
+     if (thaiFontData == NULL) {
+          thaiFontData = new ThaiFontUserData(getThaiFontType(item));
+          item->font->setUserData(thaiFontTypeUDataID(), thaiFontData);
+     }
+     ThaiFontType font_type = thaiFontData->font_type;
+#else
+     ThaiFontType font_type = getThaiFontType(item);
 #endif
```

ส่วนวิธีการจำแนกฟอนต์เราทำโดยตรวจสอบว่าฟอนต์นั้นๆมี glyph อยู่ในตาราง glyph table ที่เราสร้างขึ้นมาหรือไม่ โดย glyph table ที่เราสร้างขึ้นมี 3 ตาราง คือ tis620_0 สำหรับเก็บ glyph ที่เป็น TIS, tis620_1 สำหรับเก็บ glyph ของ Windows, tis620_2 สำหรับเก็บ glyph ของ MAC โดยมีฟังก์ชัน getThaiFontType(QShaperItem *item) สำหรับใช้ get ค่า font type ผ่าน item โดยภายในฟังก์ชันนี้จะมีฟังก์ชัน thai_contain_glyphs ในการตรวจสอบ font กับ glyph table ทั้ง 3 อยู่

```
+static ThaiFontType getThaiFontType(QShaperItem *item)
+{
+     QOpenType *ot = item->font->openType();
+
+     if (!(ot && ot->supportsScript(item->script))) {
+          if (thai_contain_glyphs(item,tis620_2))
+               return MAC;
+          else if (thai_contain_glyphs(item,tis620_1))
+               return WIN;
+          else return TIS;
+     }
+     return TIS;
+}
```

flow chart แสดงการทำงาน



สรุปชนิดข้อมูลและฟังก์ชัน
ชนิดข้อมูลที่ได้เพิ่มเติม

```
class ThaiFontUserData : public QObjectUserData
enum ThaiFontType
```

ตาราง glyphs tabel

```
static int tis620_0[128]
static int tis620_1[128]
static int tis620_2[128]
```

ฟังก์ชันที่ได้เพิ่มเติม

```
static uint thaiFontTypeUDataID()
static ThaiFontType getThaiFontType(QShaperItem *item)
static int thai_contain_glyphs(QShaperItem *item, const int glyph_map[128])
```

2. Convert String to TIS-620

ทำหน้าที่ในการ convert string ที่รับมาจาก item->string (unicode) มาให้เป็น TIS-620 (thchar_t) เพื่อนำมา ใช้กับฟังก์ชันใน libthai โดยจะต้อง convert ตั้งแต่ item->from จำนวน item->length ซึ่งเป็นส่วนที่เป็นภาษาไทย ดังนั้นแราจะได้ string tis_text ที่มีแต่ภาษาไทยอย่างเดียว และทำการจัดเตรียมตัวแปรต่างที่ใช้สำหรับการสร้าง glyph string ต่อไป โดยการทำงานส่วนนี้อยู่ในช่วงบรรทัดที่ 3679-3688

```
+       thchar_t *tis_text;
+       int tis_length;
+       static QTextCodec *thaiCodec = QTextCodec::codecForMib(2259);
+       QGlyphLayout *glyphs = item->glyphs;
+       unsigned short *logClusters = item->log_clusters;

+       QByteArray tis_str = thaiCodec->fromUnicode(item->string->data() + item->from, item-
>length);
+       tis_text = (thchar_t*)tis_str.data();
+       tis_length = tis_str.length();
+       QString glyphs_str;
```

3. Create Glyph String and Set LogCluster

ทำหน้าที่ในการสร้าง glyphs string และกำหนดค่า LogCluster สำหรับการเลื่อนเคอร์เซอร์ เพื่อที่จะส่งต่อไปให้ opentype แสดงผล โดยมีหลักการทำงานดังนี้จาก tis_text string ในกระบวนการที่สอง ให้นำมาแบ่งเซลโดยใช้ฟังก์ชัน th_next_cell ซึ่งเป็นฟังก์ชันใน libthai ซึ่งในการแบ่งเซลเราจะกำหนดให้ทำการแยกสระอำออก ยกเว้นกรณีที่เป็นฟอนต์แบบ monospace ให้รวมสระอำเป็นเซลเดียว (เอาไปใช้ใน konsole) หลังจากการแบ่งเซลแล้วเราจะได้ cell_length กับ tis_cell ที่จะต้องนำไปใช้งานต่อ ซึ่งค่า cell_length เป็นค่า return มาจากฟังก์ชัน th_next_cell โดย cell_length จะเป็นตัวบอกจำนวนอักษรในเซล ตัวอย่างเช่น คำว่า "ที่" cell_length จะเท่ากับ 3 ส่วนคำว่า "กำ" ในกรณีทำการแยกสระอำ cell_length จะเท่ากับ 2

```
+       for (int i = 0;i < tis_length;/* nop */) {
+           int cell_length;
+           int n = 0;
+           struct thcell_t tis_cell;
+           thglyph_t log_glyphs[4];
+
+           cell_length = th_next_cell (tis_text + i, tis_length - i, &tis_cell, !item->font-
>fontDef.fixedPitch);
```

ดังนั้นเราจึงนำค่า cell_length นี้มาช่วยกำหนด logClusters ในการกำหนดค่า logCluster เราจะใช้ความยาวของ glyph string ในการกำหนดในที่นี้คือ glyphs_str.length() หลังจากการกำหนดค่า logCluster แล้วเราจะทำการสร้าง logical glyph 8 bits เพื่อใช้ในการสร้าง glyphs string 16 bits ในการสร้างเราจะใช้ฟังก์ชันจาก libthai ได้แก่ th_render_cell_tis, th_render_cell_mac, th_render_cell_win โดยการเลือกใช้งานฟังก์ชัน เราต้องใช้ฟังก์ชันให้ถูกกับประเภทของฟอนต์ว่าเป็น win, mac, tis โดยแยกแยะได้จากกระบวนการที่หนึ่งที่ได้กล่าวไปแล้ว การใช้งานฟังก์ชันเหล่านี้เราจะต้องส่งค่า tis_cell ที่ได้จากฟังก์ชัน th_next_cell ซึ่งเป็นข้อมูลประเภท thcell_t ซึ่งข้อมูลนี้จะเก็บโครงสร้างของ cell เช่น base, top, hilo ว่าเป็นอะไร และหลังจากการทำงานของฟังก์ชันเราจะได้ค่า log_glyphs (8 bits) ของ tis_cell ที่ได้ส่งไปและได้ค่าที่ส่งกลับมา (n) ออกมาเป็นจำนวน glyphs ใน cell ที่ส่งไป

```
+           for(int j = 0;j < cell_length;j++)
+               logClusters[i + j] = glyphs_str.length();
+
+     //Make Logical Glyphs switch by font type
+           switch (font_type) {
+               case TIS: n = th_render_cell_tis (tis_cell, log_glyphs,
sizeof(log_glyphs)/sizeof(log_glyphs[0]), TRUE);break;
+               case WIN: n = th_render_cell_mac (tis_cell, log_glyphs,
sizeof(log_glyphs)/sizeof(log_glyphs[0]), TRUE);break;
+               case MAC: n = th_render_cell_win (tis_cell, log_glyphs,
sizeof(log_glyphs)/sizeof(log_glyphs[0]), TRUE);break;
+           }
```

เมื่อเราได้ logical glyph แล้วเราก็จะนำมาสร้างเป็น glyph string 16 bits (glyphs_str) ในการสร้าง glyph string เราจะต้องทำการแปลงจาก logical glyphs 8 bits มาเป็น glyphs แบบ 16 bits ก่อน โดยการแปลงเราใช้ฟังก์ชัน thai_get_glyph_index เมื่อแปลงเสร็จเราก็นำ glyph ที่ได้มาต่อกันเป็น glyph string (glyphs_str) โดยในการสร้าง glyph string จะต้องมีการตรวจสอบก่อนว่า log_glyphs ตัวนั้นๆ มีค่าเป็น TH_BLANK_BASE_GLYPH หรือไม่ ถ้ามีก็ให้ใส่ dotted circle เพิ่มลงไปใน glyph string ยกเว้นกรณีที่ฟอนต์เป็นแบบ monospace ไม่ต้องใส่ เมื่อเราได้ glyph string เรียบร้อยแล้วเราจะต้องจัดการกับกรณีพิเศษคือกรณีสระอำ โดยเราจะต้องทำการเซต logCluster ก่อนหน้า นิคหิต ทั้งในกรณีที่มีการผสมวรรณยุกต์ด้วย

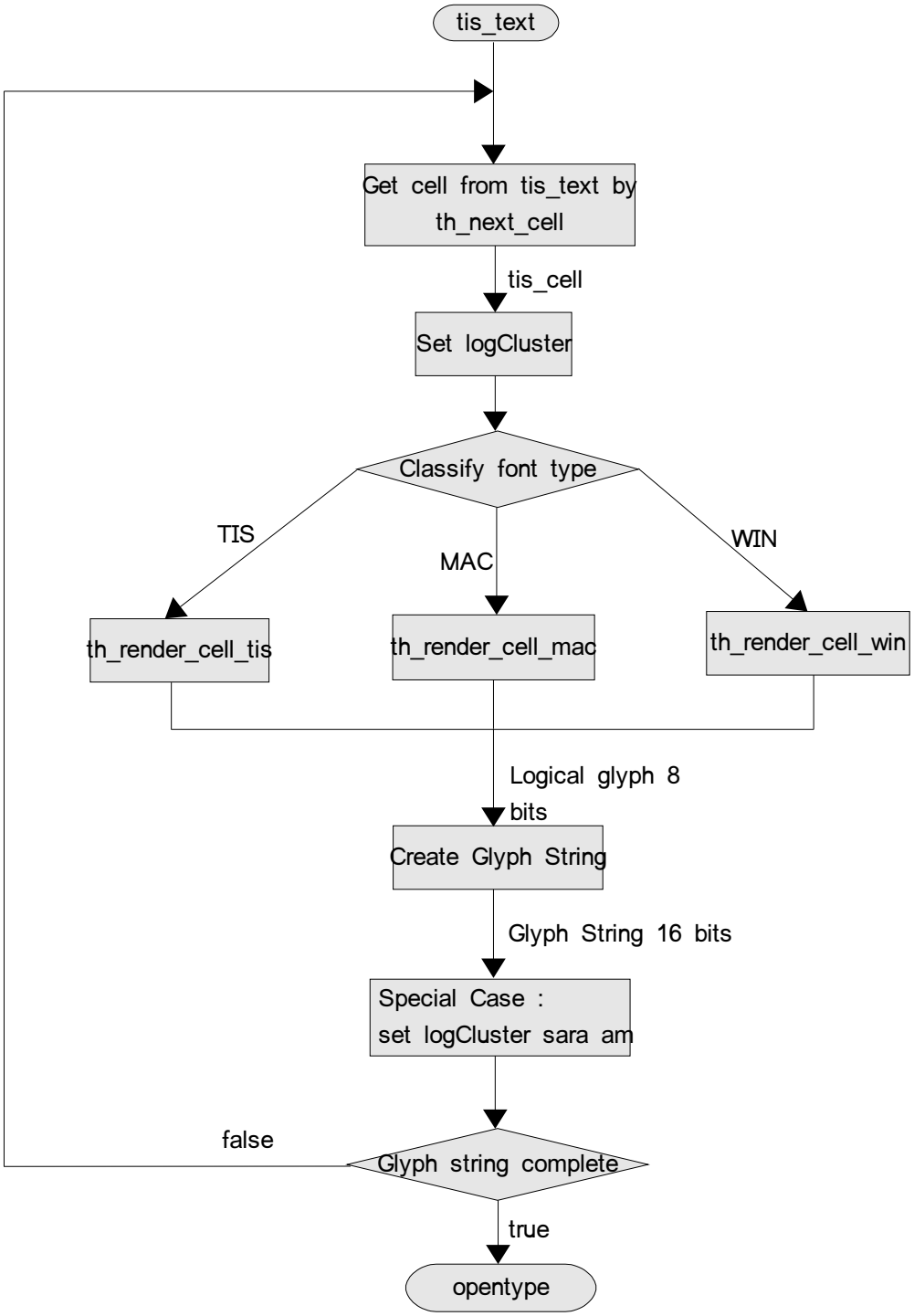
```
+     //Add glyphs to glyphs string
+           for (int a = 0;a < n;a++) {
+               if (log_glyphs[a] == TH_BLANK_BASE_GLYPH) {
+                   if (!item->font->fontDef.fixedPitch)
+                       glyphs_str.append(0x25cc);
+               }
```

```
+                else
+                    glyphs_str.append(QChar(thai_get_glyph_index(font_type, log_glyphs[a])));
+            }
+
+        //SARA AM
+            if (tis_cell.base != 0 && tis_cell.hilo == 0xd3) {
+                logClusters[i + cell_length - 1] = glyphs_str.length() - 2; // Set logClusters before
NIKAHIT
+                if (tis_cell.top != 0)
+                    logClusters[i + cell_length - 1]--; // Set logClusters before NIKAHIT when
tis_cell has top
+            }
+
+            i += cell_length;
+        }
```

จากนั้นเรานำ glyphs string ที่ได้นำมา map กับ item->font ด้วยฟังก์ชัน item->font->stringToCMap()

```
+        if (!item->font->stringToCMap(glyphs_str.unicode(), glyphs_str.length(),
+                                      item->glyphs, &item->num_glyphs, QFlag(item->flags)))
+            return false;
```

flow chart การทำงาน



ชนิดข้อมูลและฟังก์ชัน

ฟังก์ชันที่ได้เพิ่มเติม

```
static int thai_get_glyph_index (ThaiFontType font_type,uchar c)
```

ฟังก์ชันที่ใช้ใน libthai

```
extern size_t th_next_cell(const thchar_t *s, size_t len, struct thcell_t *cell, int is_decomp_am)
extern int th_render_cell_tis(struct thcell_t cell, thglyph_t res[], size_t res_sz, int is_decomp_am)
extern int th_render_cell_mac(struct thcell_t cell, thglyph_t res[], size_t res_sz, int is_decomp_am)
extern int th_render_cell_win(struct thcell_t cell, thglyph_t res[], size_t res_sz, int is_decomp_am)
```

header file: thcell.h, thrend.h

4. Set Attribute

ทำหน้าที่ในการกำหนดค่าของ glyph attribute ในส่วนของ clusterStart สำหรับเป็นตำแหน่งในการแก้ไข glyph ซึ่งจะต้องเป็นตำแหน่งเดียวกันกับตำแหน่งของเคอร์เซอร์ โดยการกำหนดค่าให้กับ clusterStart เราจะกำหนดให้ clusterStart มีค่าเป็น true เมื่อ glyph นั้นเป็นตำแหน่งที่เคอร์เซอร์หยุด ส่วน glyph ตำแหน่งอื่น (ตำแหน่งที่เคอร์เซอร์ทำการข้าม) ให้มีค่าเป็น false

```
+// Set Attribute
+     int iCluster = 0;
+     while (iCluster < item->length) {
+          int beginCluster = logClusters[iCluster];
+          ++iCluster;
+          while (iCluster < item->length && logClusters[iCluster]==beginCluster)
+               ++iCluster;
+          int endCluster = (iCluster < item->length) ? logClusters[iCluster]:glyphs_str.length();
+          glyphs[beginCluster].attributes.clusterStart = true;
+          for (int i = beginCluster+1;i < endCluster;i++)
+               glyphs[i].attributes.clusterStart = false;
+     }
```

5. Opentype Shape

หลังจากที่เราได้ glyph string และทำการเซตค่า attribute ต่างๆแล้วเราก็ส่งไปให้ openType (item->font->opentype()) แสดงผลและในขั้นตอนสุดท้ายในการเรียก openType->positionAndAdd จะต้องส่ง false ไปในพารามิเตอร์ตัวสุดท้ายเพื่อไม่ให้ทำการคำนวณ logCluster อีกครั้งใน doLogCluster() เพราะว่าเราจะใช้ค่า logCluster ที่เราคำนวณเอง

```
+     if (openType && openType->supportsScript(item->script)) {
+         openType->selectScript(item, item->script, basic_features);
+         openType->shape(item);
+         return openType->positionAndAdd(item, availableGlyphs, false);
+     }
```

1.2 การแก้ปัญหาการแสดงผล shortcut ที่เป็นภาษาไทย

แก้ไขในไฟล์ /src/gui/text/qtextlayout.cpp ที่ฟังก์ชัน drawMenuText ในส่วน draw underline (บรรทัดที่ 1804) โดยใน code เดิมจะทำการลากเส้นใต้โดยคิดจากตัวแรกที่ขีดเส้นใต้จนถึง logClusters ถัดไป แต่ในภาษาไทยในกรณีที่อักษรมีการผสมค่าของ logClusters ตัวถัดไปจะมีค่าเท่ากับตัวแรกดังนั้นการขีดเส้นใต้จึงขีดเป็นจุดแทน ดังนั้นเราจึงต้องแก้ไขการขีดเส้นใต้ให้คิดจากตัวแรกที่ขีดเส้นใต้ไปจนถึงค่า logCluster ตัวแรกบวกด้วยหนึ่งจึงเป็นการขีดเส้นทั้งเซลแทนซึ่งพูดง่ายๆก็คือการขีดเส้นตั้งแต่ตัวแรกจนไปถึงอักษรที่ติดกัน

```
            if (ul && *ul != -1 && *ul < end) {
                // draw underline
-               gtmp = (*ul == end-1) ? ge : logClusters[*ul+1-si.position];
+               gtmp = (*ul == end-1) ? ge : logClusters[*ul-si.position] + 1;
                ++stmp;
                gf.num_glyphs = gtmp - gs;
```

2. การแก้ปัญหา Input Method (XIM)

การแก้ไขเรื่อง inputmethod ใน qt จะแบ่งการแก้ไขออกเป็นสองส่วนคือ แก้ที่ qt inputmethod และ qlineedit ซึ่งเป็น widget ของ qt

2.1 การแก้ที่ qt inputmethod

ปัญหาการ input method โดยใช้ XIM เกิดจากการที่ XIM และ Qt ไม่สามารถติดต่อขอรับข้อมูลกันได้ จึงทำให้ XIM มีข้อมูลเฉพาะตัวที่พิมพ์ก่อนหน้าเท่านั้น ดังนั้นการแก้ปัญหานี้เราต้องสร้างฟังก์ชัน (conversion callback) ที่ทำการรับ request ของ XIM และทำการติดต่อกับ Qt เพื่อดึงข้อมูลไปให้ XIM ทำงานต่อ โดยการวิธีการนี้มีสอง ขั้นตอนคือ

1. Register Conversion CallBack Function
2. Implement Conversion CallBack Function

Register Conversion CallBack Function

การ register conversion callback function นั้นมีกระบวนการอยู่สองขั้นตอนคือ

1. check string conversion support
2. register string conversion callback

1. check string conversion support

เป็นการทดสอบว่า XIM รองรับการทำ string conversion หรือไม่ โดยการทดสอบเราทำในไฟล์ /src/gui/inputmethod/qximinputcontext_x11.cpp ในฟังก์ชัน QXIMInputContext::create_xim() ที่บรรทัด 475-489 โดยทำการ query XIM โดยใช้ฟังก์ชัน XGetIMValues หลังจากเรียกใช้เราจะได้ข้อมูลของ XIM ในรูปของ array ในที่นี้คือ ic_values ซึ่งเราจะทำการตรวจสอบค่าใน array นี้ว่ามีค่า XNStringConversionCallback หรือไม่ ถ้ามีแสดงว่ารองรับการทำ string conversion จากนั้นให้ทำการเซตค่า supports_string_conv flag โดย flag นี้เราจะต้องเพิ่มเข้าไปเป็นเมมเบอร์ของ class QXIMInputContext ในไฟล์ /src/gui/inputmethod/qximinputcontext_p.h บรรทัดที่ 119

ไฟล์ /src/gui/inputmethod/qximinputcontext_p.h

```
        ICData *icData() const;
+       bool supports_string_conv;
    protected:
        bool x11FilterEvent( QWidget *keywidget, XEvent *event );
```

ไฟล์ /src/gui/inputmethod/qximinputcontext_x11.cpp

```
+           // Check support string converion flag
+           XIMValuesList *ic_values = NULL;
+           XGetIMValues(xim, XNQueryInputStyle, &styles,
```

```
+                   XNQueryICValuesList,  &ic_values,
+                   NULL);
+          if (ic_values) {
+               for (int i = 0; i < ic_values->count_values; i++)
+                   if (strcmp (ic_values->supported_values[i],
+                        XNStringConversionCallback) == 0)
+                   {    //qDebug("Support string conversion");
+                       this->supports_string_conv = true;
+                       break;
+                   }
+               XFree (ic_values);
+          }
```

2. register string conversion callback

เป็นการ register string conversion callback ไปที่ XIM ซึ่งต้องแก้ไขในไฟล์ qximinputcontext_x11.cpp ที่ฟังก์ชัน QXIMInputContext::createICData บรรทัดที่ 765 และ 822-829 โดยเริ่มเราต้องสร้างตัวแปร XIMCallback ชื่อ string_conv_callback สำหรับทำการเซ็ตค่า XNStringConversionCallback ให้กับ XIM ซึ่งตัวแปรนี้จะเก็บข้อมูลสองอย่างคือ ข้อมูล client data ที่จะส่งไปให้ XIM และฟังก์ชันที่ให้ XIM callback กลับมา (ในที่นี้ คือฟังก์ชัน string_conversion_callback) และเมื่อเตรียมค่าให้กับตัวแปร string_conv_callback เสร็จแล้วก็นำตัวแปรนี้ไปเซ็ตค่าให้กับ XIM โดยใช้ฟังก์ชัน XSetICValues ในการเซ็ตค่า XNStringConversionCallback ของ XIM

```
      XVaNestedList preedit_attr = 0;
-     XIMCallback startcallback, drawcallback, donecallback;
+     XIMCallback startcallback, drawcallback, donecallback, string_conv_callback;
```

```
      if (data->ic) {
          // when resetting the input context, preserve the input state
          (void) XSetICValues(data->ic, XNResetState, XIMPreserveState, (char *) 0);
+         //Register string conversion callback
+         if (this->supports_string_conv) {
+                 string_conv_callback.client_data = (XPointer) this;
+                 string_conv_callback.callback = (XIMProc) string_conversion_callback;
+                 (void) XSetICValues(data->ic,
+                 XNStringConversionCallback,&string_conv_callback,
+                 NULL );
+         }
```

Implement String Conversion CallBack Function

การ implement ฟังก์ชัน string_conversion_callback นี้ทำที่ไฟล์ qximinputcontext_x11.cpp ในบรรทัดที่ 284-359 โดยฟังก์ชัน string_conversion_callback นี้ทำหน้าที่ในการตอบสนอง request ที่มาจาก XIM โดย request ที่มาจาก XIM จะมีอยู่สองแบบคือ XIMStringConversionRetrieval (สำหรับขอ surrounding text) และ XIMStringConversionSubstitution (สำหรับการแก้ไข string) โดยฟังก์ชันนี้มีพารามิเตอร์ที่ชื่อว่า client_data ทำหน้าที่ในการติดต่อในส่วนของ qt และพารามิเตอร์ที่ชื่อว่า call_data ทำหน้าที่ในการส่งข้อมูลของ XIM

ขั้นตอนในการสร้างฟังก์ชัน string_conversion_callback จะต้องสร้างส่วนที่รองรับกับ request ของ XIM ทั้งสอง XIMStringConversionRetrieval และ XIMStringConversionSubstitution

ส่วนรองรับกับ XIMStringConversionRetrieval Request

การทำงานส่วนนี้ทำหน้าที่ในการสร้าง string ที่ XIM request เข้ามา โดยเริ่มจากการสร้าง instance ของ XIMInputContext จากพารามิเตอร์ชนิด Xpointer ชื่อ client_data สำหรับการอ้างถึงข้อมูลในด้านของ qt และสร้าง instance ของ XIMStringConversionCallbackStruct จากพารามิเตอร์ชนิด Xpointer ชื่อ call_data สำหรับอ้างถึงข้อมูลของ XIM

```
+    QXIMInputContext *qic = reinterpret_cast<QXIMInputContext *>(client_data);
+    XIMStringConversionCallbackStruct *conv_data =
reinterpret_cast<XIMStringConversionCallbackStruct *>(call_data);
```

จากนั้นเราจะ get surrounding text และ cursor position จาก qt (widget) โดยใช้ตัวแปร qic ในการติดต่อกับ widget ใน qt โดยในการเรียกจะทำการเรียกผ่านฟังก์ชัน inputMethodQuery ใน widget นั้นโดยทำการส่งข้อมูลที่ต้องการ query ไปเป็นพารามิเตอร์ของฟังก์ชัน

```
+    QString surrounding = qic->icData()->widget->
inputMethodQuery(Qt::ImSurroundingText).toString();
+    int cursor_index = qic->icData()->widget->inputMethodQuery(Qt::ImCursorPosition).toInt();
```

และเมื่อเราได้ surrounding text และ cursor_index แล้วเราจะทำการสร้าง string ที่ XIM request มาโดยข้อมูลที่จำเป็นในการสร้างได้แก่ conv_data->position, conv_data->direction, conv_data->factor ซึ่งจะนำข้อมูลเหล่านี้มาใช้กับ surrounding string โดย

conv_data->position คือข้อมูลที่ใช้ในการกำหนดตำแหน่งเริ่มต้นของการสร้าง XIM request string โดยมีค่าอ้างอิงกับตำแหน่งของเคอร์เซอร์ (cursor_index)

```
+        short position = (short)conv_data->position;
```

conv_data->direction คือ ข้อมูลที่ใช้บอกทิศทางในการสร้าง request string จากตำแหน่ง conv_data-

>position ตัวอย่างในภาษาไทยจะต้องทำการ retrieve character ที่อยู่ด้านหน้าของเคอร์เซอร์ ดังนั้นค่าของ conv_data->direction จะเป็น XIMBackwardChar

conv_data->factor คือ ข้อมูลที่ใช้บอกจำนวนตัวอักษรจากตำแหน่ง conv_data->position ไปในทิศทางของ conv_data->direction

โดยเราจะได้ request string ตั้งแต่ตำแหน่ง conv_data->position ไปตามจำนวน conv_data->factor ในทิศทางของ conv_data->direction

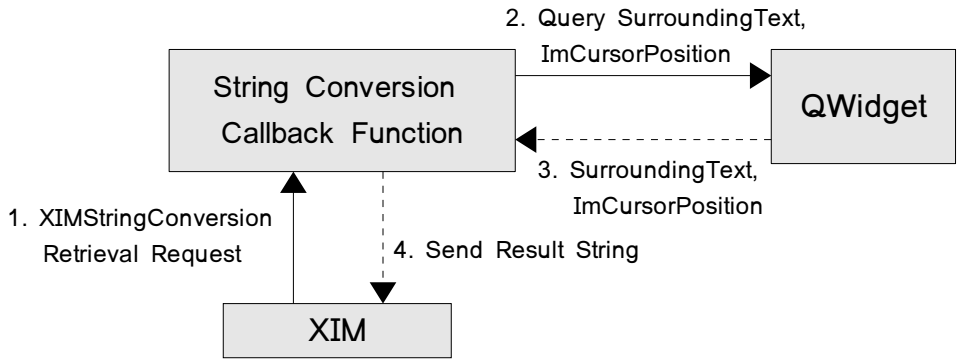
```
+          switch (conv_data->direction)
+          {
+               case XIMForwardChar:
+               if (cursor_index > 0 && cursor_index+position >= 0)
+                    for (i = cursor_index+position; i < cursor_index+position+conv_data->factor; i++)
+                         str_tmp.append(surrounding[i]);
+
+               subst_offset = position + conv_data->factor;
+               subst_nchars = conv_data->factor;
+               break;
+
+               case XIMBackwardChar:
+               if (cursor_index > 0 && cursor_index+position-conv_data->factor >= 0)
+                    for (i = cursor_index+position-conv_data->factor; i < cursor_index+position; i++)
+                         str_tmp.append(surrounding[i]);
+
+               subst_offset = position - conv_data->factor;
+               subst_nchars = conv_data->factor;
+               break;
+
```

และหลังจากได้ XIM request string แล้วเราจะทำการส่งค่า string นี้กลับไปให้ XIM โดยการส่งจะทำการส่งผ่านตัวแปร conv_data โดยจะต้องทำการเซ็ตค่าต่างๆ ใน conv_data->text และเราจะต้องกำหนด request string (str_tmp) ที่ตัดมาจาก surrounding text ให้กับค่า conv_data->text->string.wcs

```
+          /* block out any failure happenning to "text", including conversion */
+          if (!str_tmp.isNull()) {
+               conv_data->text = (XIMStringConversionText *)malloc (sizeof
(XIMStringConversionText));
+               if (conv_data->text) {
```

```
+                   conv_data->text->length = str_tmp.length();
+                   conv_data->text->feedback = NULL;
+                   conv_data->text->encoding_is_wchar = True;
+                   conv_data->text->string.wcs = (wchar_t *)malloc (str_tmp.length() * sizeof
(wchar_t));
+                       if (conv_data->text->string.wcs) {
+                           for (i = 0; i < str_tmp.length(); i++)
+                           conv_data->text->string.wcs[i] = str_tmp[i].unicode();
+                       }
+                       else {
+                           free (conv_data->text);
+                           conv_data->text = NULL;
+                       }
+               }
+           }
```

รูปแสดงการทำงานในการ Retrieve String ของ XIM



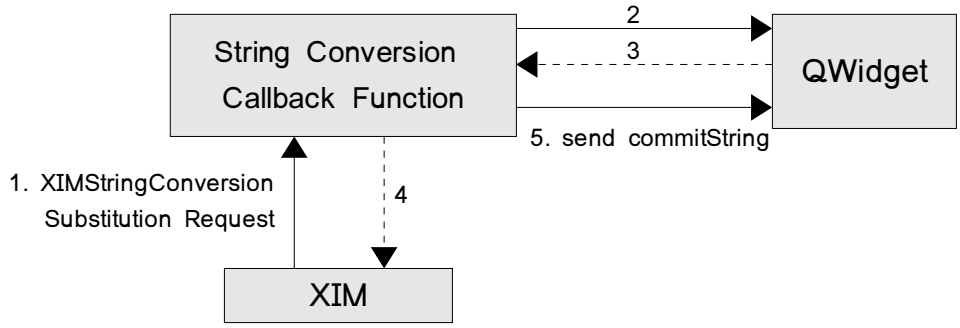
ส่วนรองรับกับ XIMStringConversionSubstitution Request

การทำงานในส่วนนี้จะทำหน้าที่ในส่งคำสั่งในการลบ string ไปให้ qt widget ทำการลบ โดยขั้นตอนการทำงานจะเริ่มการตรวจสอบ operation ของ XIM ว่าเป็นแบบ XIMStringConversionSubstitution หรือไม่ ถ้าใช่ให้ทำการส่ง QinputMethodEvent ที่ได้ทำการเซ็ต commitString (ตำแหน่งและจำนวนที่ต้องการลบ) ไปให้ qt widget ในการลบ string โดยใช้ฟังก์ชัน qic->sendEvent ในการส่ง event

```
+           if (conv_data->operation == XIMStringConversionSubstitution && subst_nchars > 0) {
+               QInputMethodEvent im_event;
+               im_event.setCommitString("",subst_offset,subst_nchars);
+               qic->sendEvent(im_event);
```

```
+           }
```

รูปแสดงการทำงานในการ Substitution String ของ XIM (หลังจากเสร็จสิ้นขั้นตอนที่ 4 ใน flow chart ข้างบน)



2.2 การแก้ปัญหา input method ใน qlineedit

แก้ไขที่ไฟล์ /src/gui/widgets/qlineedit.cpp ในฟังก์ชัน inputMethodEvent บรรทัดที่ 2122 ซึ่งเดิมตำแหน่งเคอร์เซอร์หลังจากใส่ commit string (c) จะถูกบวกด้วยค่า qMin(-e->replacementStart(), e->replacementLength()) ซึ่งฟังก์ชัน replacementStart() จะ return ค่าตำแหน่งที่เริ่มการแทนที่เทียบกับตำแหน่งเคอร์เซอร์ปัจจุบันและในกรณีที่ฟังก์ชันนี้ return ค่าที่เป็นลบจะทำให้ได้ค่า c ที่ผิดพลาด และในการแก้ไข เราแก้ไขจากเดิมให้นำค่า qMin(-e->replacementStart(), e->replacementLength()) ไปบวกกับค่า c ให้นำไปลบกับค่า c แทน

```
      if (e->replacementStart() <= 0)
-         c += e->commitString().length() + qMin(-e->replacementStart(), e->replacementLength());
+         c += e->commitString().length() - qMin(-e->replacementStart(), e->replacementLength());
```

3. การแก้ปัญหาการเลื่อนเคอร์เซอร์

ปัญหาการ Arrow Left, Arrow Right

ปัญหาการเลื่อนเคอร์เซอร์ ซ้าย-ขวา จะต้องไปแก้ไขที่ไฟล์ /src/gui/text/qscriptengine.cpp ในฟังก์ชัน thaiWordBreaks (ช่วงบรรทัดที่ 3505-3525) ซึ่งเดิมฟังก์ชันนี้ทำหน้าที่ในการกำหนด char attributes สำหรับการตัดคำไทยอย่างเดียว แต่สำหรับการเลื่อนเคอร์เซอร์แล้วจะต้องทำการกำหนด attributes charStop ด้วย ในการกำหนดเราจะใช้ประโยชน์จากฟังก์ชัน th_next_cell ใน libthai ซึ่งจะทำการแบ่ง cluster ไว้ให้แล้ว เราจึงกำหนด charStop ตาม cluster ที่แบ่งมาแต่ยังต้องจัดการกับกรณีพิเศษอีกคือกรณีที่เป็นสระอำจะต้องกำหนด charStop ที่สระอำด้วย (การกำหนด charStop เป็น true คือหยุด false คือ ข้าม)

```
+     //Set CharStop
+     thchar_t* tis_text = (thchar_t*)tis_str.data();
```

```
+     int tis_length = tis_str.length();
+
+     for (int i = 0;i < tis_length;/* nop */) {
+          int cell_length;
+          struct thcell_t tis_cell;
+          cell_length = th_next_cell (tis_text + i, tis_length - i, &tis_cell, TRUE);
+
+          attributes[i].charStop = true;
+          for(int j = 1;j < cell_length;j++)
+               attributes[i + j].charStop = false;
+
+          // Set Char Stop SARA AM
+          if (tis_text[i + cell_length - 1] == 0xd3) {
+               attributes[i + cell_length - 1].charStop = true;
+          }
+
+          i += cell_length;
+     }
```

ปัญหาการ Ctrl+Left-Right, Ctrl+Backspace

แก้ไขในไฟล์ /src/gui/text/qtextlayout.cpp ที่ฟังก์ชัน nextCursorPosition (Ctrl+Right) ในบรรทัดที่ 655 และ previousCursorPosition (Ctrl+Left) ในบรรทัดที่ 684 ซึ่งเป็นส่วนที่ตรวจสอบการ SkipCharacters โดยเราจะเพิ่มการตรวจสอบ attributes.lineBreakType สำหรับการตัดคำเข้าไป ดังนี้

```
          while (oldPos < len && !attributes[oldPos-1].whiteSpace
-                && !d->atWordSeparator(oldPos))
+                && !d->atWordSeparator(oldPos)
+                && attributes[oldPos-1].lineBreakType != QCharAttributes::Break)
               oldPos++;
```

ปัญหาการ Ctrl+Delete, Double Click Select word

แก้ไขในไฟล์ /src/gui/text/qtextcursor.cpp ที่ฟังก์ชัน EndOfWord บรรทัดที่ 416 ซึ่งเป็นส่วนที่ตรวจสอบการสิ้นสุดของคำ โดยเราจะเพิ่มการตรวจสอบ attributes.lineBreakType สำหรับการตัดคำเข้าไป ดังนี้

```
          while (relativePos < len
                 && !attributes[relativePos].whiteSpace
```

```
-                    &&  !engine->atWordSeparator(relativePos))
+                    &&  !engine->atWordSeparator(relativePos)
+                    &&  (attributes[relativePos-1].lineBreakType  !=  QCharAttributes::Break))
                 relativePos++;
```

4. การแก้ปัญหาในการตัดคำไทย

ปัญหาในการตัดคำไทยแบ่งออกเป็นสองปัญหาได้แก่

1. เมื่อประโยคนั้นมี space หรือมีการแบ่งประโยค
2. เมื่อประโยคนั้นมีคำไทยเรียงติดต่อกันเกิน 128 คำ

ปัญหาการตัดคำในประโยคที่มี space หรือมีการแบ่งประโยค

แก้ไขที่ไฟล์ /src/gui/text/qscriptengine.cpp ในฟังก์ชัน thai_attributes ในบรรทัดที่ 3540 โดยเดิมการส่งพารามิเตอร์ attributes ในการเรียกฟังก์ชัน thaiWordBreaks นั้นจะทำการส่ง attributes ตั้งแต่ต้นไปจึงทำให้การเซ็ตค่า attributes ในส่วนที่เพิ่มเข้ามาไปเซ็ตในช่วงต้นจนถึง len ที่ได้ส่งไป จึงทำให้ attributes ช่วงแรกถูกทับด้วย attributes ที่เพิ่มเข้ามา ดังนั้นการแก้ไขเราจึงต้องส่ง attributes ในตำแหน่งที่สอดคล้องกับตำแหน่ง text.unicode() + from โดยการส่ง attributes + from แทน

```
      Q_ASSERT(script == QUnicodeTables::Thai);
-     thaiWordBreaks(text.unicode() + from, len, attributes);
+     thaiWordBreaks(text.unicode() + from, len, attributes + from);
+}
```

ปัญหาการตัดคำในประโยคที่มีคำไทยเรียงติดต่อกันเกิน 128 คำ

แก้ไขที่ไฟล์ /src/gui/text/qscriptengine.cpp ในฟังก์ชัน thaiWordBreaks โดยเดิมฟังก์ชันนี้จะเก็บตำแหน่ง break position ไว้ใน int array ขนาด 128 จำนวนดังนั้นถ้ามีจำนวน break position มากว่านี้จะทำให้ไม่สามารถเซ็ต attributes.lineBreakType ได้ครบทุกตัว ดังนั้นเราจึงแก้ให้ break_positions allocate ตามจำนวน len ที่รับเข้ามาแทนจึงทำให้สามารถรับ text ที่มีความยาวติดต่อกันได้ และในขั้นตอนสุดท้ายเมื่อทำการเซ็ตค่า lineBreakType เรียบร้อยแล้วก็ทำการ clear ค่า break_positions ที่ allocate มาทิ้ง

```
-     QByteArray cstr = thaiCodec->fromUnicode(QString(string, len));
+     QByteArray tis_str = thaiCodec->fromUnicode(QString(string, len));

-     int brp[128];
-     int *break_positions = brp;
-     int numbreaks = th_brk(cstr.constData(), break_positions, 128);
-     if (numbreaks > 128) {
-         break_positions = new int[numbreaks];
-         numbreaks = th_brk(cstr.data(),break_positions, numbreaks);
-     }
+     int *break_positions = new int[len];
+     int numbreaks = th_brk((const thchar_t*)(tis_str.constData()), break_positions, len);
```

```
-       for (int i = 0; i < len - 1; ++i)
+       for (int i = 0; i < len; ++i)
            attributes[i].lineBreakType = QCharAttributes::NoBreak;

        for (int i = 0; i < numbreaks; ++i) {
            if (break_positions[i] > 0)
-               attributes[break_positions[i]-1].lineBreakType = QCharAttributes::Break;
+               attributes[break_positions[i] - 1].lineBreakType = QCharAttributes::Break;
        }

-       if (break_positions != brp)
+       if (break_positions != NULL)
            delete [] break_positions;
```

Metamedia Technology Co., Ltd.

Room 2104, 21F Silom Center Building 2 Silom Rd.,Bangrak, Bangkok 10500
Tel: +66-2-632-9700, Fax: +66-2-632 9703
Web: http://www.mm.co.th/, Email: info@mm.co.th

การแก้ปัญหาภาษาไทยใน KDE Applications

5. การแก้ปัญหาในการตัดคำไทยใน konqueror

ปัญหาการ load libthai ของ Klibrary แก้ไขโดยทำการระบุรุ่นที่ต้องการ load ลงไปด้วย (libthai.so.0) ในไฟล์ /kde4libs-4.0.0/khtml/rendering/break_lines.cpp ที่บรรทัดที่ 72-74

```
                printf("Try to load libthai dynamically...\n");
                 KLibLoader *loader = KLibLoader::self();
-                lib = loader->library(QLatin1String("libthai"));
+                 lib = loader->library(QLatin1String("libthai.so.0"));
+                 if ( !lib )
+                       lib = loader->library(QLatin1String("/usr/lib/libthai.so.0"));
                 if ( lib )
```

ปัญหาการตัดคำไทยใน konqueror เมื่อประโยคที่ส่งมามีคำมากกว่า 1024 คำจะทำให้ไม่มีการตัดคำในส่วนที่เกินออกไป ซึ่งเกิดจากการจัดเตรียมหน่วยความจำไม่เพียงพอ ดังนั้นวิธีแก้ไขสามารถทำได้โดยการจัดเตรียมหน่วยความจำให้เท่ากับขนาดของ string ที่ส่งมา โดยเข้าไปแก้ไขที่ไฟล์ /kde4libs-4.0.0/khtml/rendering/break_lines.cpp ที่บรรทัดที่ 106-107

```
                 //fprintf(stderr,"libthai returns with value %d\n",cache->numwbrpos);
-                if (cache->numwbrpos > cache->allocated) {
-                      cache->allocated = cache->numwbrpos;
+                 if (cache->numwbrpos >= cache->allocated) {
+                       cache->allocated = len;
                       cache->wbrpos = (int *)realloc(cache->wbrpos, cache->allocated*sizeof(int));
```

6. การแก้ปัญหาการแสดงผลใน konsole

การแก้ปัญหาวรรณยุกต์หรือสระบน-ล่างไม่แสดงผล แก้ไขโดยทำการเซ็ต charSequence ให้กับตัวอักษรที่มีการผสมสระบน-ล่าง โดยมีขั้นตอนการทำดังนี้

1. การเซ็ต charSequence ให้กับ Character
2. การแสดงผล Character
3. การ Copy & Paste
4. การ Input Method

การเซ็ต charSequence ให้กับ Character

การเซ็ตค่า charSequence ให้กับ Character นี้ทำในไฟล์ /kdebase-3.96.0/apps/konsole/src/Screen.cpp ที่ฟังก์ชัน Screen::ShowCharacter บรรทัดที่ 732-760 โดยวิธีในการเซ็ตค่า charSequence เราทำโดยนำตัวอักษรและสระหรือวรรณยุกต์บน-ล่างมาที่ผสมกันมาเข้า hash function แล้วนำ hash code ที่ได้มาเก็บเป็นค่า charSequence ในตำแหน่งนั้น (อักษรตำแหน่ง base) เพราะฉะนั้นจะเห็นได้ว่าในการเก็บ characters เราจะไม่เก็บพวกสระหรือวรรณยุกต์บน-ล่าง โดยตรง ตัวอย่างเช่นคำว่า "มาที่ม" เราจะเก็บตัวอักษร [ม],[า],[hash_code(ที่)],[ม] โดยใน code เรากำหนดตัวแปร u_char_combind เอาไว้สำหรับเก็บอักษรที่มีการผสมกันโดยกำหนดให้มีการผสมได้มากสุดอยู่ที่ 5 ตัวอักษร ตัวอย่าง คำว่า "ที่" มีการผสม 3 ตัว

```
+   //Set CharSequence
+   ushort u_char_combind[5];
+
```

จากนั้นเราจะทำการตรวจสอบ character ว่าเป็นสระหรือวรรณยุกต์บนล่างหรือไม่ โดยตรวจสอบได้จากความกว้างของ character โดยถ้ามีความกว้างเป็น 0 (เป็นสระบน-ล่าง) จะต้องทำการเซ็ต charSequence โดยวิธีการเซ็ต จะทำการตรวจสอบค่า character ตัวก่อนหน้าก่อนว่าเป็น charSequence หรือไม่ โดยทำการตรวจ bit ใน attribute rendition ใน class Character ว่าได้เซ็ตเป็น RE_EXTENDED_CHAR หรือไม่ ถ้าได้เซ็ตเป็น RE_EXTENDED_CHAR แล้วแสดงว่า character ตัวก่อนหน้าเป็น charSequence ดังนั้นในการคำนวณ charSequence ใหม่จะต้องทำการถอดค่า charSequence นั้นว่าประกอบไปด้วยอักษรตัวไหนบ้างโดยการใช้ฟังก์ชัน ExtendedCharTable::instance.lookupExtendedChar และทำการเพิ่มอักษรใหม่เข้าไป จากนั้นจึงทำการคำนวณค่า charSequence ใหม่อีกครั้งโดยการใช้ฟังก์ชัน ExtendedCharTable::instance.createExtendedChar แล้วนำค่าที่คำนวณได้ใส่ใน attribute charSequence ของ character ตัวก่อนหน้า

```
+   if (w == 0 && cuX > 0) {
+     if (screenLines[cuY][cuX-1].rendition & RE_EXTENDED_CHAR)
+     {
```

```
+          // sequence of characters
+          ushort extendedCharLength = 0;
+          ushort* chars = ExtendedCharTable::instance
+                                   .lookupExtendedChar(screenLines[cuY]
[cuX-1].charSequence,extendedCharLength);
+          if (extendedCharLength > 5)
+               return;
+          for ( int index = 0 ; index < extendedCharLength ; index++ )
+               u_char_combind[index] = chars[index];
+
+          u_char_combind[extendedCharLength] = c;
+          Character& th_char = screenLines[cuY][cuX-1];
+          th_char.charSequence =
ExtendedCharTable::instance.createExtendedChar(u_char_combind      ,extendedCharLength+1);
+     }
```

และในกรณีที่ character ตัวก่อนหน้าไม่ได้เป็น charSequence เราจะต้องทำการเซ็ต bit ใน attribute rendition ของ class Character ให้เป็น RE_EXTENDED_CHAR ก่อน จากนั้นจึงคำนวณค่า charSequence ของ character ก่อนหน้าและ character ที่เข้ามาใหม่ และนำค่าที่ได้ไปใส่ใน attribute charSequence ของ character ตัวก่อนหน้า

```
+     else
+     {
+          Character& th_char = screenLines[cuY][cuX-1];
+
+          th_char.rendition |= RE_EXTENDED_CHAR;
+          u_char_combind[0] = (ushort)screenLines[cuY][cuX-1].character;
+          u_char_combind[1] = c;
+          th_char.charSequence =
ExtendedCharTable::instance.createExtendedChar(u_char_combind      ,2);
+     }
+     return;
+  }
```

การแสดงผล Character

การแสดงผล characters ใน konsole นั้นจะใช้ฟอนต์ TlwgTypo ซึ่งเป็นฟอนต์ monospace ที่ทำการใส่ anchor เพื่อเชื่อมอักขระฐานกับสระบน-ล่าง ซึ่งจะทำให้ฟอนต์ monospace สามารถ compose charcell อัตโนมัติ จึงทำให้สามารถใช้ painter->drawText() ได้โดยตรง

ในส่วนต่อมาแก้ไขที่ ฟังก์ชัน TerminalDisplay::drawContents ในบรรทัดที่ 1263 เพื่อไม่ให้ทำการ render characters ที่เป็น charSequence ด้วยวิธีเดิมที่ใช้กับ characters ปกติ คือ render ตามรหัสที่เก็บใน characters นั้น ซึ่งถ้าเป็น characters แบบ charSequence จะเก็บค่า hash แทน

```
                _image[loc(x+len,y)].backgroundColor == currentBackground &&
                _image[loc(x+len,y)].rendition == currentRendition &&
+               _image[loc(x+len,y)].rendition & ~RE_EXTENDED_CHAR &&  //No CharSequence
                (_image[ qMin(loc(x+len,y)+1,_imageSize) ].character == 0) == doubleWidth &&
```

การ Copy & Paste

เมื่อเรามีการใช้ charSequence แล้วการ copy & paste จะต้องทำการถอดค่า charSequence ออกมาเป็น ค่ารหัสของ characters แต่ละตัวใน charSequence ก่อนถึงจะทำการ copy ได้ ไม่เช่นนั้นก็จะทำการ copy ค่า charSequence นั้นไปเลยซึ่งจะได้ค่า hash ไปแทน โดยการแก้ไขให้ไปแก้ไขที่ไฟล์ /kdebase-3.96.0.org/apps/konsole/src/TerminalCharacterDecoder.cpp ในฟังก์ชัน PlainTextDecoder::decodeLine ในบรรทัด 83-96

```
-    for (int i=0;i<outputCount;i++)
-    {
-        plainText.append( QChar(characters[i].character) );
-    }
+    for (int i=0;i<outputCount;i++)
+    {
+        if (characters[i].rendition & RE_EXTENDED_CHAR)
+        {
+            ushort extendedCharLength = 0;
+            ushort* chars = ExtendedCharTable::instance
+
.lookupExtendedChar(characters[i].charSequence,extendedCharLength);
+
+            for (int j = 0;j<extendedCharLength;j++)
+                plainText.append(QChar(chars[j]));
+
+        } else
```

```
+                plainText.append( QChar(characters[i].character) );
+    }
```

การ Input Method

การ input method ใน konsole นี้เราใช้ XIM โดยที่ไม่ให้มีการทำ string conversion เพราะว่าตัว konsole ไม่ได้เป็นเจ้าของ text buffer ซึ่งจะเกิดปัญหาในกรณีที่ทำการ remote และ network ที่ใช้ช้าจะทำให้ต้องมีเวลาที่ต้องคอยให้ XIM ไป query surrounding text มาก่อนจึงจะทำการใส่ตัวต่อไปได้เช่นในการพิมพ์คำว่า "ที่" จะพิมพ์ 'ท' และ สระอี ทันทีเลยไม่ได้ จะต้องรอให้ XIM ไป query surrounding text มาให้เสร็จก่อน เพราะฉะนั้นเพื่อความเหมาะสมเราจึงตัดส่วนการทำ string conversion ออกไป

ดังนั้นเราจะต้องทำการแก้ไขฟังก์ชัน inputMethodQuery ของ konsole ซึ่งอยู่ในไฟล์ TerminalDisplay.cpp ที่ฟังก์ชัน TerminalDisplay::inputMethodQuery โดยในส่วนการ query cursor position (ImCursorPosition) นั้นให้เราทำการคำนวณตำแหน่งเคอร์เซอร์ โดยคิดรวมจำนวน characters ที่ อยู่ใน charSequence ด้วย

```
        case Qt::ImCursorPosition:
+           {
+               Character *chars = &_image[loc(0,cursorPos.y())];
+               int tmp_count = 0;
+               for (int i = 0;i < cursorPos.x();i++)
+               {
+                   if (chars[i].rendition & RE_EXTENDED_CHAR)
+                   {
+                       ushort extendedCharLength = 0;
+                       ushort* chars_t = ExtendedCharTable::instance
+                           .lookupExtendedChar(chars[i].charSequence,extendedCharLength);
+                       tmp_count += extendedCharLength-1;
+                   }
+               }
                // return the cursor position within the current line
-              return cursorPos.x();
+               return cursorPos.x() + tmp_count;
+           }
            break;
```

และในส่วนการ query surrounding text (ImSurroundingText) เราจะทำการส่ง null string กลับไป เพื่อปิดการทำงานในส่วนของ string conversion

```
            case Qt::ImSurroundingText:
-               {
+               {
+                   return QString(QString::null);
                    // return the text from the current line
-                   QString lineText;
-                   QTextStream stream(&lineText);
-                   PlainTextDecoder decoder;
-                   decoder.begin(&stream);
-
decoder.decodeLine(&_image[loc(0,cursorPos.y())],_usedColumns,_lineProperties[cursorPos
    .y()]);
-                   decoder.end();
-                   return lineText;
```

7. การแก้ปัญหาใน Kate/Kwrite

7.1 การแก้ปัญหาการ input method ใน Kate/Kwrite

ปัญหาการ input method ใน Kate/Kwrite นั้นมีอยู่ สองปัญหา คือ

1. ปัญหาการพิมพ์แล้ว cursor ไม่เลื่อนตำแหน่ง
2. ปัญหาการทำ text normalize

ปัญหาการพิมพ์แล้ว cursor ไม่เลื่อนตำแหน่ง

ปัญหานี้แก้ไขที่ไฟล์ /kdelibs-3.96.0/kate/view/kateviewinternal.cpp ในฟังก์ชัน KateViewInternal::inputMethodEvent บรรทัดที่ 3432 โดยเดิมจะทำการกำหนด KtextEditor::Cursor start ด้วยค่า m_imPreedit ซึ่งเป็นค่าที่ใช้ในภาษาที่ต้องใช้การ preedit ในการพิมพ์เช่น ภาษาญี่ปุ่น ส่วนในภาษาไทยไม่ได้ใช่ค่านี้ดังนั้นจึงทำให้ค่า cursor start เป็น 0 ตลอด วิธีแก้ไขทำได้โดยเราจะต้องกำหนด KtextEditor::Cursor start ด้วยค่า เคอร์เซอร์จริงๆ ซึ่งคือ m_cursor แทนการกำหนดด้วย preedit

```
    if (!e->commitString().isEmpty() || e->replacementLength()) {
      KTextEditor::Range preeditRange = *m_imPreedit;

-     KTextEditor::Cursor start(m_imPreedit->start().line(), m_imPreedit->start().column() + e-
>replaceme
    ntStart());
+//     KTextEditor::Cursor start(m_imPreedit->start().line(), m_imPreedit->start().column() + e-
>replace
    mentStart());
+     KTextEditor::Cursor start(m_cursor.line(), m_cursor.column() + e->replacementStart());
      KTextEditor::Cursor removeEnd = start + KTextEditor::Cursor(0, e->replacementLength());
```

ปัญหาการทำ text normalize

ปัญหานี้แก้ไขที่ไฟล์ kateviewinternal.cpp ในฟังก์ชัน KateViewInternal::inputMethodQuery บรรทัดที่ 3382 ในส่วน Qt::ImCursorPosition ซึ่งเดิมเป็นการส่งค่าที่ return จาก cursorCoordinates(false) ซึ่งค่าที่ได้จะเป็นชนิด QPoint ซึ่งไม่ถูกต้อง ดังนั้นการแก้ไขเราจึงส่ง m_displayCursor.column() ซึ่งเป็นตำแหน่งของเคอร์เซอร์แทน

```
      case Qt::ImCursorPosition:
-       return cursorCoordinates(false);
+//       return cursorCoordinates(false);
```

```
+          return m_displayCursor.column();
```

การแก้ไขในส่วนถัดมาแก้ไขที่ไฟล์ kateviewinternal.cpp ในฟังก์ชัน KateViewInternal::inputMethodEvent บรรทัดที่ 3409 ซึ่งเดิมถ้ามีการ request XIMStringConversionSubstitution (commit string จะเป็น "") ดังนั้นเมื่อเข้าเงื่อนไขการตรวจสอบที่บรรทัดนี้จะทำให้เกิดการ return ดังนั้นจึงทำให้ไม่สามารถทำการ normalization text ได้เราจึงต้องเพิ่มกรณีในการตรวจสอบเพิ่มอีกกรณีคือกรณีที่ e->replacementLength()==0 ด้วย

```
    // Finished this input method context?
-   if (m_imPreedit && e->preeditString().isEmpty() && e->commitString().isEmpty()) {
+//   if (m_imPreedit && e->preeditString().isEmpty() && e->commitString().isEmpty()) {
+   if (m_imPreedit && e->preeditString().isEmpty() && e->commitString().isEmpty() && e-
>replacementLengt
h() == 0) {
        m_view->removeInternalHighlight(m_imPreedit);
```

7.2 การแก้ปัญหาการกด Delete ใน Kate/Kwrite

ปัญหาการ delete ใน kate/kwrite ที่เกิดขึ้นคือ กด delete แล้วทำการลบทีละตัวอักษร ไม่ทำการ delete ทั้ง cell โดยไปแก้ไขที่ไฟล์ kateviewinternal.cpp ในฟังก์ชัน KateViewInternal::doDelete() บรรทัดที่ 719-723 ซึ่งเราแก้ไขโดย ทำการ select ไปทางขวาหนึ่งครั้ง (m_view->shiftCursorRight()) และหลังจากนั้นให้ทำการ ลบ text ที่ถูก select อยู่ (m_view->removeSelectedText()) ดังนั้นเราก็สามารถลบทีละ cell ได้ โดยในขั้นตอนการทำ select text (m_view->shiftCursorRight()) จะต้องตรวจสอบก่อนว่ามีการ select text อยู่ก่อนหน้าหรือไม่

```
 void KateViewInternal::doDelete()
 {
-   m_doc->del( m_view, m_cursor );
+//   m_doc->del( m_view, m_cursor );
+   if ( m_view->config()->persistentSelection() || !m_view->selection() ) {
+      m_view->shiftCursorRight();
+   }
+   m_view->removeSelectedText();
+   return;
 }
```

7.3 การแก้ปัญหาการตัดคำใน kate/kwrite

ปัญหาการตัดคำไทยใน kate/kwrite สามารถแก้ไขโดยทำการเพิ่ม attribute ของการตัดคำ (m_breakLine) และฟังก์ชันที่ใช้ตรวจสอบค่าของ attribute m_breakLine นี้ (isBreakable(int pos)) เข้าไปในคลาส KateTextLine โดยเข้าไปเพิ่มที่ไฟล์ katetextline.h ในบรรทัดที่ 287-291 และ 384-387

```
      void insertText (int pos, const QString& insText);

      /**
+      * return breakline attribute m_breakLine
+      * @param pos insert position
+      */
+     bool isBreakable (int pos);
+
+     /**
       * remove text at given position
       * @param pos start position of remove
       * @param delLen length to remove
@@ -375,6 +381,11 @@ class KateTextLine : public KShared
       */
      QVector<int> m_attributesList;

+      /**
+      * Attribute for lineBreak value "true" for breakline
+      */
+     QVector<bool> m_breakLine;
+
      /**
       * context stack
       */
```

โดยค่า m_breakline จะเก็บเป็น vector ของ bool ซึ่งถ้าเป็น true หมายถึงเป็นตำแหน่งที่ตัดและเป็น false คือตำแหน่งที่ไม่มีการตัด

โดยฟังก์ชัน bool isBreakable(int pos) จะทำหน้าที่ในการดึงค่าของ m_breakline ในตำแหน่ง pos ซึ่งเพิ่มฟังก์ชันนี้เข้าไปในไฟล์ katetextline.cpp ในบรรทัดที่ 189-192

```
+bool KateTextLine::isBreakable (int pos)
+{
+   return m_breakLine[pos];
 }
```

และทำการเพิ่มส่วนการตัดคำไทยเข้าไปในไฟล์ katetextline.cpp ในบรรทัดที่ 31-192 โดยในส่วนนี้จะประกอบไปด้วย การประกาศ header และค่าต่างๆ ที่จำเป็นต้องใช้ในการตัดคำไทย การเพิ่ม struct ThaiCache สำหรับทำ cache การตรวจสอบการตัดคำไทย

การประกาศ header และค่าต่างๆ ที่จำเป็นในการตัดคำ เช่น header ต่างๆ ของ libthai โดยส่วนนี้จะอยู่บรรทัดที่ 31-40

```
+//Load Libthai
+#include <QtCore/QTextCodec>
+#include <klibloader.h>
+#ifndef HAVE_LIBTHAI
+typedef int (*th_brk_def)(const unsigned char*, int[], int);
+static th_brk_def th_brk;
+#else
+#include <thai/thailib.h>
+#include <thai/thbrk.h>
+#endif
+
```

ในการทำ cache จะทำการสร้าง ThaiCache ขึ้นสำหรับ cahce ในกรณี string ที่ส่งมาเหมือนกับ string ก่อนหน้านี้ (จะไม่ต้องคำนวณจุดตัดเดิมอีก) โดยเพิ่มเข้าไปในบรรทัดที่ 43-66

```
+//Thai Break
+struct ThaiCache
+{
+     ThaiCache() {
+          string = 0;
+          allocated = 0x400;
```

```
+            wbrpos = (int *) malloc(allocated*sizeof(int));
+            numwbrpos = 0;
+            numisbreakable = 0x400;
+            isbreakable = (int *) malloc(numisbreakable*sizeof(int));
+            library = 0;
+    }
+    ~ThaiCache() {
+        free(wbrpos);
+        free(isbreakable);
+        if (library) library->unload();
+    }
+    const QChar *string;
+    int *wbrpos;
+    int *isbreakable;
+    int allocated;
+    int numwbrpos,numisbreakable;
+    KLibrary *library;
+};
+static ThaiCache *cache = 0;
+
```

ส่วนถัดมาเป็นส่วนการ cleanup memory โดยฟังก์ชัน cleanup_thaibreaks() จะทำการ delete pointer ที่ได้สร้างขึ้นมาในโปรแกรม ส่วนนี้อยู่ในช่วงบรรทัดที่ 68-75

```
+void cleanup_thaibreaks()
+{
+    delete cache;
+    cache = 0;
+#ifndef HAVE_LIBTHAI
+    th_brk = 0;
+#endif
+}
+
```

ส่วนต่อมาเป็นส่วนการพิจารณาจุดตัดใน string โดยฟังก์ชัน isbreakWordThai จะ return

ค่าเป็น pointer ของ int โดยค่านี้จะเก็บค่าจุดตัดที่ได้คำนวณมาตรงกับตำแหน่งของ string ที่ได้ส่งมา โดยส่วนนี้อยู่ในบรรทัดที่ 78-139

```
+int* isbreakWordThai( const QChar *string, const int pos, const int len)
+{
+     string = string + pos;
+     static QTextCodec *thaiCodec = QTextCodec::codecForMib(2259);
+
+#ifndef HAVE_LIBTHAI
+
+     KLibrary *lib = 0;
+
+     /* load libthai dynamically */
+     if (!th_brk && thaiCodec) {
+          printf("Try to load libthai dynamically...\n");
+          KLibLoader *loader = KLibLoader::self();
+          lib = loader->library(QLatin1String("libthai.so.0"));
+          if ( !lib )
+              lib = loader->library(QLatin1String("/usr/lib/libthai.so.0"));
+          if ( lib )
+              th_brk = (th_brk_def) lib->resolveFunction("th_brk");
+          if ( !th_brk ) {
+              // indication that loading failed and we shouldn't try to load again
+              printf("Error, can't load libthai...\n");
+              thaiCodec = 0;
+              if (lib)
+                  lib->unload();
+           }
+     }
+
+     if (!th_brk ) {
+          return NULL;
+     }
+#endif
+     if (!cache ) {
+          cache = new ThaiCache;
+#ifndef HAVE_LIBTHAI
```

```
+            cache->library = lib;
+#endif
+      }
+      // build up string of thai chars
+      if ( string != cache->string ) {
+            //fprintf(stderr,"new string found (not in cache), calling libthai\n");
+            QByteArray cstr = thaiCodec->fromUnicode( QString::fromRawData(string,len));
+            //printf("About to call libthai::th_brk with str: %s",cstr.data());
+            cache->numwbrpos = th_brk((const unsigned char*) cstr.data(), cache->wbrpos, cache-
>allocated);
+            //fprintf(stderr,"libthai returns with value %d\n",cache->numwbrpos);
+            cache->isbreakable = (int *)realloc(cache->isbreakable, len*sizeof(int));
+           if (cache->numwbrpos >= cache->allocated) {
+                  cache->allocated = len;
+                  cache->wbrpos = (int *)realloc(cache->wbrpos, cache->allocated*sizeof(int));
+                  cache->numwbrpos = th_brk((const unsigned char*) cstr.data(), cache->wbrpos,
cache->allocated);
+            }
+            for (int i = 0 ; i < len ; ++i) {
+                  cache->isbreakable[i] = 0;
+            }
+            if ( cache->numwbrpos > 0 ) {
+                  for (int i = cache->numwbrpos-1; i >= 0; --i) {
+                        cache->isbreakable[cache->wbrpos[i]] = 1;
+                  }
+            }
+            cache->string = string;
+      }
+
+      return cache->isbreakable;
+}
+
 KateTextLine::KateTextLine ()
    : m_flags(0)
```

ต่อมาเป็นส่วนของการใส่ attribute m_breakline โดยการใส่ค่าของ m_breakline นี้เราจะใส่ค่าไปพร้อมกับการ insert text โดยเข้าไปเพิ่มที่ฟังก์ชัน insertText ใน KateTextLine ในช่วง

บรรทัดที่ 163-186

```
     m_text.insert (pos, insText);
+
+   m_breakLine.resize (m_text.size());
+   for (int i = pos; i < pos + insText.length(); i++) {
+      m_breakLine.insert (i, false);
+   }
+
+   int i = pos;
+   while (i != 0 && !m_text[i].isSpace()) {
+      --i;
+   }
+
+   while (i < pos + insText.length()) {
+      int th_len = 0;
+      while (m_text[i].unicode() < 0x0e00 && m_text[i].unicode() >= 0x0e80 && i < pos +
insText.length())
+         i++;
+      while (m_text[i + th_len].unicode() >= 0x0e00 && m_text[i + th_len].unicode() < 0x0e80 && i
+ th_len < pos + insText.length())
+         th_len++;
+
+      if (th_len > 0) {
+         int* th_break = isbreakWordThai(m_text.constData(), i, th_len);
+         for (int j = i;j < i + th_len; j++)
+            m_breakLine[j] = th_break[j - i];
+      }
+      i += th_len + 1;
+   }
+}
```

และในกรณีลบ m_breakline ก็ทำเช่นเดียวกันโดยทำการลบพร้อมกับการลบ text โดยทำที่ฟังก์ชัน removeText ใน katetextline.cpp บรรทัดที่ 212

```
    m_text.remove (pos, delLen);
+   m_breakLine.remove (pos, delLen);
```

```
}
```

ส่วนต่อมาเป็นการแก้ไขในส่วนของการกดปุ่ม Ctrl+Del และ Ctrl+Arrow(left-right) โดยการแก้ไขให้แก้ไขที่ไฟล์ kateviewinternal.cpp

การแก้ไขการกดปุ่ม Ctrl+Del แก้ที่บรรทัดที่ 735-739

```
void KateViewInternal::doDelete()
{
-   m_doc->del( m_view, m_cursor );
+   if ( m_view->config()->persistentSelection() || !m_view->selection() ) {
+       m_view->shiftCursorRight();
+   }
+   m_view->removeSelectedText();
+   return;
}
```

การแก้ไขการกดปุ่ม Ctrl+Arrow แก้ที่ฟังก์ชัน wordLeft และ wordRight ในบรรทัดที่ 1031-1034 (wordLeft)

```
     {
-       while( !c.atEdge( left ) && h->isInWord( m_doc->line( c.line() )[ c.column() - 1 ] ) )
-         --c;
+       if (m_doc->getBreakLine(c.line(),c.column()))
+           --c;
+
+       while( !c.atEdge( left ) && h->isInWord( m_doc->line( c.line() )[ c.column() - 1 ] ) && !
m_doc->getBreakLine(c.line(),c.column()))
+           --c;
    }Len);}
```

และบรรทัด 1072-1075 (wordRight)

```
    else if( h->isInWord( m_doc->line( c.line() )[ c.column() ] ) )
    {
-       while( !c.atEdge( right ) && h->isInWord( m_doc->line( c.line() )[ c.column() ] ) )
+       if (m_doc->getBreakLine(c.line(),c.column()))
```

```
+         ++c;
+      while( !c.atEdge( right ) && h->isInWord( m_doc->line( c.line() )[ c.column() ] ) && !m_doc-
>getBreakLine(c.line(),c.column()) )
          ++c;
    }
```

โดยการแก้นี้ทำโดยการเพิ่มการตรวจสอบ breakline ในตำแหน่ง c.line(), c.column() โดยใช้ฟังก์ชัน getBreakLine ของ KateDocument ในการตรวจสอบจุดตัด โดยฟังก์ชัน getBreakLine นี้จะเพิ่มเข้าไปในไฟล์ katedocument.cpp และ katedocument.h

เพิ่มฟังก์ชัน getBreakLine เข้าไปในไฟล์ katedocument.cpp ที่บรรทัด 563-571

```
+bool KateDocument::getBreakLine(int line, int col)
+{
+   KateTextLine::Ptr l = m_buffer->plainLine(line);
+
+   if(!l)
+      return 0;
+
+   return l->isBreakable(col);
+}
+
```

เพิ่มการประกาศฟังก์ชัน getBreakLine เข้าไปในไฟล์ katedocument.h ที่บรรทัดที่ 186

```
+      bool getBreakLine(int line, int col);
       virtual KTextEditor::Cursor documentEnd() const;
```

ส่วนต่อมาเป็นส่วนที่ใช้การตัดคำเมื่อกดที่เมนู Tools > Word Wrap Document โดยการแก้ไขให้แก้ไขที่ไฟล์ katedocument.cpp ในฟังก์ชัน wrapText บรรทัดที่ 1202-1216

```
          int z = 0;
          int nw = 0; // alternative position, a non word character
+         bool removeTrailingSpace = false;
+
          for (z=searchStart; z > 0; z--)
```

```
        {
-         if (t.at(z).isSpace()) break;
-         if ( ! nw && highlight()->canBreakAt( t.at(z) , l->attribute(z) ) )
-         nw = z;
+          if (t.at(z).isSpace()) {
+            removeTrailingSpace = true;
+            z++;
+            break;
+         }
+          if (l->isBreakable(z)) break;
+         if ( ! nw && highlight()->canBreakAt( t.at(z) , l->attribute(z) ) )
+            nw = z;
        }

-       bool removeTrailingSpace = false;
-       if (z > 0)
-       {
-         // So why don't we just remove the trailing space right away?
-         // Well, the (view's) cursor may be directly in front of that space
-         // (user typing text before the last word on the line), and if that
-         // happens, the cursor would be moved to the next line, which is not
-         // what we want (bug #106261)
-         z++;
-         removeTrailingSpace = true;
-       }
-       else
+       if (z == 0)
        {
```

ส่วนต่อมาเป็นส่วนที่ให้ kate/kwrite ทำการตัดคำเมื่อถึงท้ายประโยค โดยแก้ที่ไฟล์ /kate/render/renderer.cpp ในฟังก์ชัน layoutline ที่บรรทัดที่ 758

```
   opt.setTabStop(m_tabWidth  *  config()->fontMetrics().width(spaceChar));
-    opt.setWrapMode(QTextOption::WrapAnywhere);//QTextOption::WrapAtWordBoundaryOrAnywhere);
+
opt.setWrapMode(QTextOption::WrapAtWordBoundaryOrAnywhere);//QTextOption::WrapAnywhere);
```

8. การแก้ปัญหาการ input method ใน Kword

ปัญหาการ input method ใน kword คือ การไม่สามารถ normalization text ได้โดยมีสาเหตุมาจาก kword ยังไม่ได้มีการ implement ฟังก์ชัน inputMethodEvent เลยดังนั้นเราจึงต้องทำการ implement ฟังก์ชันนี้เพิ่มเข้าไปโดยเราเพิ่มฟังก์ชัน TextTool::inputMethodEvent เข้าไปที่ไฟล์ /koffice-1.9.95.org/shapes/text/TextTool.cpp บรรทัดที่ 905-917 ซึ่งฟังก์ชันนี้ทำหน้าในการแก้ไข characters ในการลบหรือการเพิ่ม commit string โดยเราจะใช้ตัวแปร m_caret (QTextCursor) ในการจัดการ cursor และ characters โดยในการลบ characters เราจะใช้ฟังก์ชัน m_caret.deleteChar() ลบตามจำนวนที่ ส่งมาจากพารามิเตอร์ event (event->replacementLength())

```
+void TextTool::inputMethodEvent (QInputMethodEvent * event) {
+     if (event->replacementLength() > 0) {
+          m_caret.setPosition(m_caret.position() + event->replacementStart());
+          for (int i = event->replacementLength(); i > 0; --i) {
+               m_caret.deleteChar();
+          }
+     }
```

ส่วนในการเพิ่ม commit sting (event->commitString()) เราทำโดยการส่ง commitString ผูกไปกับ QkeyEvent และส่งไปโดยฟังก์ชัน keyPressEvent

```
+     if(! event->commitString().isEmpty()) {
+          QKeyEvent ke(QEvent::KeyPress, -1, 0, event->commitString());
+          keyPressEvent(&ke);
+     }
+     event->accept();
+}
+
 void TextTool::ensureCursorVisible() {
```

ในการเพิ่มฟังก์ชัน inputMethodEvent เข้าไปใน class TextTool เราจะต้องทำการเพิ่มเข้าไปใน header file ด้วย ในไฟล์ koffice-1.9.95.org/shapes/text/TextTool.h ดังนี้

```
     virtual QVariant inputMethodQuery(Qt::InputMethodQuery query, const KoViewConverter
&converter) con     st;
+     /// reimplemented from superclass
+     virtual void inputMethodEvent(QInputMethodEvent * event);
```

การพัฒนาในส่วนอื่นๆ

9. การเพิ่ม libthai configuration เข้าไปใน qt configuration

การเพิ่ม libthai เข้าไปใน qt นี้จะทำให้ qt สามารถเรียกฟังก์ชันต่างๆจาก libthai ได้เลยโดยการเรียกแบบ static (include header file ของ libthai ได้เลย) ไม่ต้องทำการเรียกแบบ dynamic อย่างที่เคยทำ โดยวิธีการทำเราจะต้องไปแก้ไขไฟล์ configuration ในการติดตั้งของ qt ในทำการรวม libthai เข้าไปด้วย โดยการแก้ไขให้เข้าไปแก้ไขที่ไฟล์ /qt/configure

โดยการแก้ไขจะเริ่มจากสร้างตัวแปรต่างๆ สำหรับ libthai พวก flag ต่างที่ใช้ในการ config โดยการเซ็ตก็ให้เซ็ตเลียนแบบโมดูลอื่นๆ ที่เซ็ตไว้แล้ว ดังนี้

เซ็ต require version ของ libthai (บรรทัดที่ 479-480)

```
+# libthai version
+MIN_LIBTHAI_VERSION=0.1.8
+
 # initalize internal variables
```

เซ็ตค่า libthai variable (บรรทัดที่ 582)

```
 CFG_FRAMEWORK=auto
+CFG_LIBTHAI=auto
 CFG_MAC_ARCHS=
```

เซ็ต libthai flags (บรรทัดที่ 630-632)

```
+# flags for libthai
+QT_CFLAGS_LIBTHAI=
+QT_LIBS_LIBTHAI=
+
 # flags for Glib (X11 only)
```

เซ็ตค่าตัวแปรจาก command line (บรรทัดที่ 1431-1437)

```
            ;;
+      libthai)
+     if [ "$VAL" = "yes" ] || [ "$VAL" = "no" ]; then
+          CFG_LIBTHAI="$VAL"
+          else
+               UNKNOW_OPT=yes
+     fi
```

```
+    ;;
      nis)
          if [ "$VAL" = "yes" ] || [ "$VAL" = "no" ]; then
```

ทำการตรวจสอบและแสดงผลเมื่อไม่มี libthai (2693-2699)

```
      DBN=" "
 fi
+if [ "$CFG_LIBTHAI" = "no" ]; then
+     LTHY=" "
+     LTHN="+"
+else
+     LTHY="+"
+     LTHN=" "
+fi
+
       cat << EOF
```

ข้อความสำหรับแสดงในกรณีที่มีและไม่มี libthai (บรรทัดที่ 2818-2819)

```
   $DBY   -qdbus..... Compile the QtDBus module.
+ $LTHN   -no-libthai..... Do not use libthai for Thai language support.
+ $LTHY   -libthai..... Use libthai for Thai language support.
+
      -reduce-relocations ... Reduce relocations in the libraries through extra linker
```

ส่วนการตรวจสอบว่ามี libthai หรือไม่ (บรรทัดที่ 4063-4085)

```
+# auto-detect libthai support
+if [ "$CFG_LIBTHAI" != "no" ]; then
+     if [ "$QT_CROSS_COMPILE" = "no" ] && "$WHICH" pkg-config >/dev/null 2>&1 && \
+          pkg-config --atleast-version="$MIN_LIBTHAI_VERSION" libthai 2>/dev/null; then
+          QT_CFLAGS_LIBTHAI=`pkg-config --cflags libthai`
+          QT_LIBS_LIBTHAI=`pkg-config --libs libthai`
+     fi
+     if "$unixtests/compile.test" "$XQMAKESPEC" "$QMAKE_CONFIG" $OPT_VERBOSE
"$relpath" "$outpath" config.tests/unix/libthai "LibTha
    i" $L_FLAGS $I_FLAGS $l_FLAGS $QT_CFLAGS_LIBTHAI $QT_LIBS_LIBTHAI; then
+          CFG_LIBTHAI=yes
+          QMakeVar set QT_CFLAGS_LIBTHAI "$QT_CFLAGS_LIBTHAI"
```

```
+          QMakeVar set QT_LIBS_LIBTHAI "$QT_LIBS_LIBTHAI"
+      else
+          if [ "$CFG_LIBTHAI" = "yes" ] && [ "$CFG_CONFIGURE_EXIT_ON_ERROR" = "yes" ];
then
+              echo "Thai language support using LibThai cannot be enabled because libthai
version $MIN_LIBTHAI_VERSION was not found.
    "
+              echo " Turn on verbose messaging (-v) to $0 to see the final report."
+              echo " If you believe this message is in error you may use the continue"
+              echo " switch (-continue) to $0 to continue."
+              exit 101
+          else
+              CFG_LIBTHAI=no
+          fi
+      fi
+fi
+
```

โดยในการตรวจสอบนี้จะต้องทำการสร้างโปรแกรมทดสอบด้วย โดยสร้างที่ /qt/config.tests/unix/libthai/ โดยทำการสร้าง สองไฟล์ได้แก่ libthai.cpp และ libthai.pro

ไฟล์ libthai.cpp

```
+#include <thai/thailib.h>
+
+int main(int, char **)
+{
+      thchar_t c;
+      return 0;
+}
```

ไฟล์ libthai.pro

```
+SOURCES = libthai.cpp
+CONFIG -= qt
+CONFIG -= app_bundle
```

ส่วนการเซ็ตค่า configure อื่นๆ (บรรทัดที่ 4787)

```
 [ "$CFG_QDBUS" = "yes" ] && QT_CONFIG="$QT_CONFIG qdbus"
+[ "$CFG_LIBTHAI" = "yes" ] && QT_CONFIG="$QT_CONFIG libthai"
```

```
 [ "$CFG_NAS" = "system" ] && QT_CONFIG="$QT_CONFIG nas"
```

ส่วนการเซ็ตค่า configure อื่นๆ (บรรทัดที่ 5226)

```
 [ "$CFG_QDBUS" != "yes" ]     && QCONFIG_FLAGS="$QCONFIG_FLAGS QT_NO_DBUS"
+[ "$CFG_LIBTHAI" != "yes" ]   && QCONFIG_FLAGS="$QCONFIG_FLAGS QT_NO_LIBTHAI"
```

ส่วนแสดงข้อความในการรองรับ module libthai ในการ configuration (บรรทัดที่ 5539)

```
 echo "QtDBus module ....... $CFG_QDBUS"
+echo "LIBTHAI module ...... $CFG_LIBTHAI"
 echo "STL support ......... $CFG_STL"
```

สรุป patch ที่ส่งเข้าต้นน้ำ

Qt4 patches: ได้ส่งไปยัง Trolltech developers ตั้งแต่ 2007-12-21

KDE4 patches:

- kdelibs
  - https://bugs.kde.org/show_bug.cgi?id=156558 (khtml)
  - https://bugs.kde.org/show_bug.cgi?id=156561 (kate/kwrite)
- kdebase
  - http://bugs.kde.org/show_bug.cgi?id=156071 (konsole)
- koffice2 (accepted 20071220)

Webkit patches:

- http://bugs.webkit.org/show_bug.cgi?id=16981

KDE3 patches:

- https://bugs.launchpad.net/ubuntu/+source/kdelibs/+bug/190371 (ubuntu)
- http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=444284 (debian)

Patches ที่ได้พัฒนาทั้งหมดอยู่ที่ LTN CVS http://linux.thai.net/