บทที่ 1
บทนำ

1.1 ที่มาของปัญหาที่นำไปสู่การค้นคว้าวิจัย และแนวทางการแก้ปัญหา

ในโลกปัจจุบันนี้เป็นโลกยุคแห่งการแข่งขัน ความสามารถทางการแข่งขันและการอยู่รอดขององค์กรผลิตซอฟท์แวร์ใด ขึ้นอยู่กับปัจจัยทางการผลิตซอฟท์แวร์ที่สามารถตอบสนองความต้องการของลูกค้าได้อย่างแม่นยำ ความต้องการเหล่านี้ทำให้เกิดการสร้างมาตรฐานซอฟต์แวร์ต่างๆเพื่อใช้ในการชี้วัดการพัฒนาด้านซอฟท์แวร์ตามมุมมองด้านวิศวกรรมศาสตร์ ซึ่งมาตรฐานที่ได้รับความนิยมได้แก่ เช่น Capability Maturity Model Integration (CMMI), ISO/IEC, IEEE และ PSM ซึ่งตัวชี้วัดมาตรฐานนี้มุ่งเน้นถึงการลดข้อผิดพลาดที่เกิดขึ้นในกระบวนการผลิตซอฟท์แวร์ที่มีขั้นตอนการพัฒนาอยู่ 6 ขั้น คือ ขั้นเริ่มต้นโครงการ, ขั้นวิเคราะห์ระบบ, ขั้นออกแบบระบบ, ขั้นพัฒนาโปรแกรม, ขั้นตรวจสอบ และ ขั้นนำไปใช้

หากขั้นเริ่มต้นโครงการกระบวนการผลิตซอฟท์แวร์ได้สกัดความต้องการซอฟต์แวร์ของลูกค้าโครงการผิด ผลิตภัณฑ์ซอฟท์แวร์ที่ได้เมื่อสิ้นสุดโครงการนั้น ถึงจะมีคุณภาพตรงตามตัวชี้วัดของมาตรฐานที่กำหนดแต่ก็ไม่ได้ตอบสนองความต้องการของลูกค้าที่แท้จริง ถึงแม้ว่าซอฟท์แวร์ที่ได้จะมีคุณภาพมากแต่ก็ถือได้ว่าล้มเหลวในการผลิต เพราะไม่ตรงกับความต้องการลูกค้าโครงการแล้วยังก่อให้เกิดปัญหา และ ต้นทุนการทำงานที่สูญเปล่าตามกระบวนการผลิตซอฟท์แวร์ในขั้นตอนต่อมา [1][3][7][10]

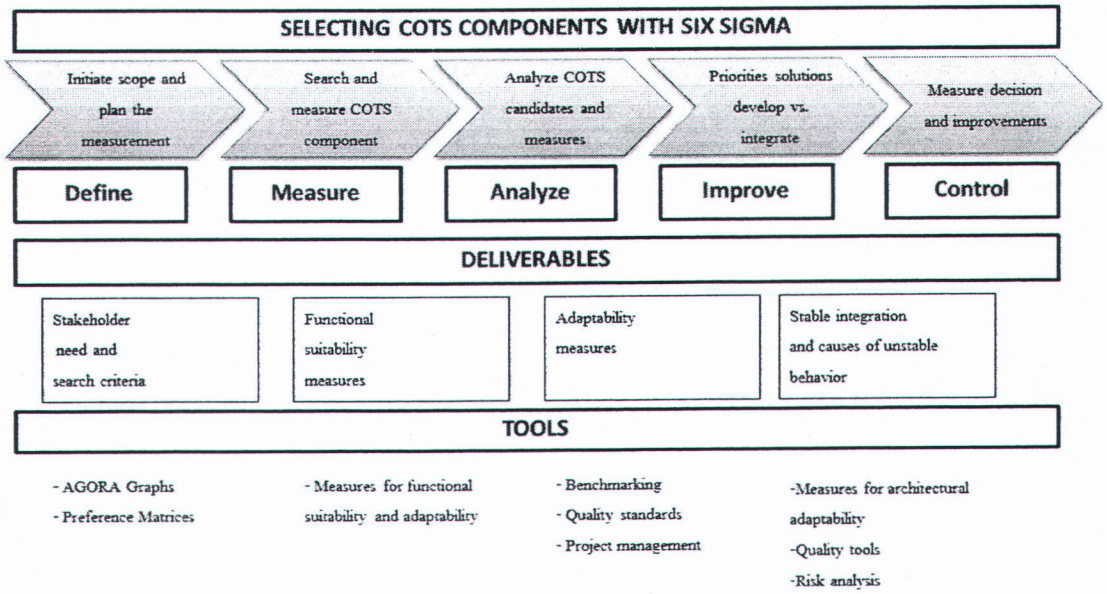
งานวิจัยนี้จึงมุ่งเน้นกระบวนการเชิงคุณภาพในการสกัดความต้องการซอฟท์แวร์ซึ่งเป็นกระบวนการเริ่มต้นโครงการ ด้วยวิธีการประเมินคุณภาพลักษณะความต้องการซอฟท์แวร์ตามมาตรฐาน ISO 9126 และออกแบบเน้นถึงความต้องของลูกค้าโครงการเป็นสิ่งสำคัญ โดยความต้องการทางซอฟต์แวร์จากความคาดหวังอันหลากหลายของลูกค้าโครงการถูกเรียงลำดับความสำคัญด้วยเทคนิคการกระจายหน้าที่เชิงคุณภาพ ซึ่งผลลัพธ์นำไปสู่ความต้อการซอฟต์แวร์เพียงเซ็ตเดียวและนำมาเป็นตัวผลักดันทางด้านข้อมูลลักษณะความต้องการซอฟท์แวร์เพื่อให้การผลิตซอฟท์แวร์ที่ได้ผลลัพธ์ตรงตามความต้องการลูกค้าโครงการได้อย่างเหมาะสม

1.2 สรุปสาระสำคัญจากเอกสารที่เกี่ยวข้อง

งานวิจัยที่เกี่ยวข้องกับงานวิจัยนี้มีอยู่ 2 ส่วน คือ งานวิจัยที่เกี่ยวข้องกับกระบวนการเก็บความต้องการทางซอฟต์แวร์เชิงคุณภาพตามมาตรฐานสากล และการประเมินจัดลำดับความต้องการทางซอฟต์แวร์จากหลากหลายความคาดหวังของลูกค้าโครงการมาใช้ร่วมปรับปรุงกระบวนพัฒนาซอฟต์แวร์

การปรับปรุงกระบวนผลิตซอฟต์แวร์โดยใช้Six Sigma มีงานวิจัยหลายงานที่นำ Six Sigma มาใช้ร่วมกับการพัฒนาซอฟต์แวร์เพื่อที่จะลดความผิดพลาด สำหรับเป็นพื้นฐานในการตัดสินใจ มุมมองของลูกค้า และ การทำงานเป็นทีม [1][2][3][7]

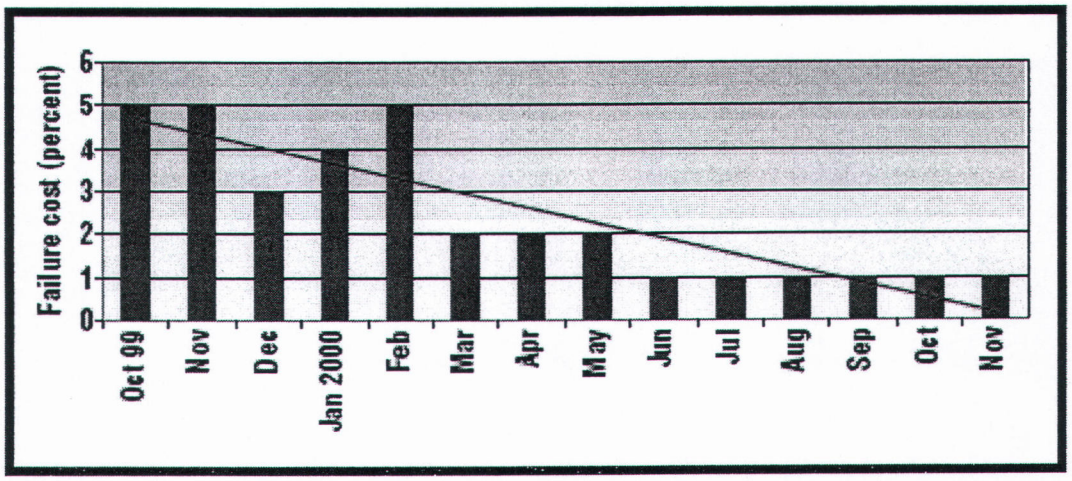
[1] ได้นำเสนอ Six Sigma มาร่วมสร้างซอฟต์แวร์ลักษณะ COTS(Commercial off the shelf) โดยนำเฟสทำงานของSix Sigma มาเลือกใช้ให้ตรงกับขั้นตอนของCOTS และมีการแนะนำเครื่องมือวัดมาใช้ร่วมประยุกต์กับการผลิตในเฟสนั้น ดังรูป1.1 แต่อย่างไรก็ตามก็ยังยากที่จะระบุใช้งานและไม่มีมาตรฐานการพัฒนาซอฟต์แวร์รองรับแต่ละขั้นตอนการทำงาน



รูปที่ 1.1 5 ขั้นตอนของSix Sigma มาร่วมสร้างซอฟต์แวร์ลักษณะ COTS

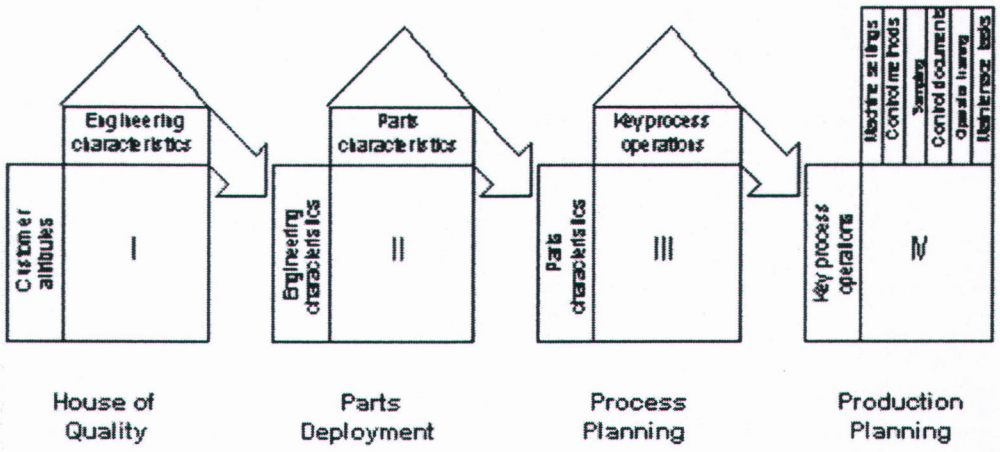
[2][3] ได้พยายามเพิ่มมาตรฐานCMMI(Capability Maturity Model) ลงในขั้นตอนการผลิตซอฟต์แวร์ โดยให้ Six Sigma เป็นวิธีการปรับปรุงธุรกิจ ลดปัญหาข้อบกพร่องในผลิตภัณฑ์โดยใช้

มุมมองของลูกค้าเป็นข้อระบุ และ CMMI เป็นตัวควบคุณภาพการพัฒนาซอฟต์แวร์ ซึ่งมีการทำงานร่วมแบบTQM(Total Quality management) งานวิจัย[3]ได้เสนอประโยชน์ที่ดีหลายด้านเมื่อนำวิธีการข้างต้นมาใช้ในบริษัท "Tata Consultancy Services" ดังรูป 1.2 แต่อย่างไรก็ตามการทำงานร่วมกับระหว่าง Six Sigma และ CMMI ก็ยังเป็นแนวทางที่ใช้ต้นทุนที่สูงในการบริหารบริษัทซอฟต์แวร์ทั่วๆไป



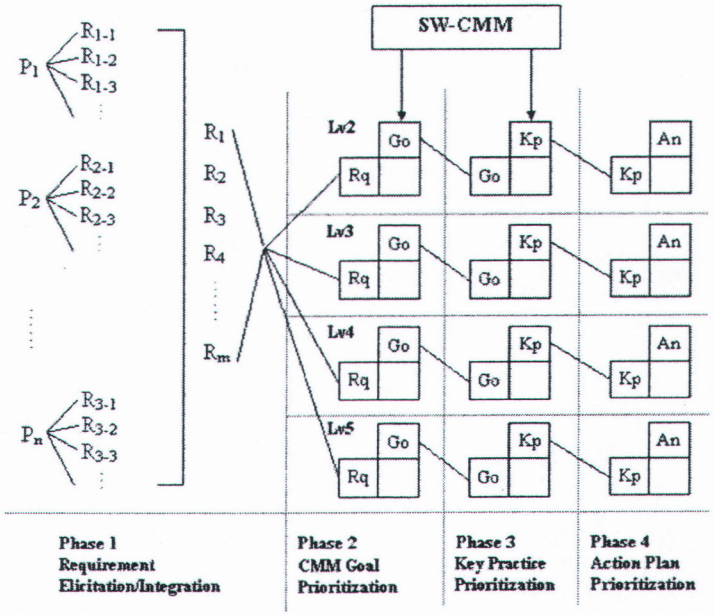
รูปที่ 1.2 แนวโน้มของต้นทุนความผิดพลาด

ในแบบจำลองการปรับปรุงกระบวนการผลิตซอฟต์แวร์สำหรับธุรกิจขนาดเล็ก [4] ได้เสนองานวิธีการ SPM(Software Process Matrix) ที่มีการใช้พื้นฐานบนเทคนิคการกระจายหน้าที่เชิงคุณภาพในการระบุความต้องการลูกค้า งานวิจัยนี้ได้นำ "Four-Phase Model" ดังรูป 1.3 ที่ปกติใช้ในอุตสาหกรรมการผลิต มาใช้การปรับปรุงกระบวนการซอฟต์แวร์ในธุรกิจขนาดเล็ก แต่อย่างไรก็ตามการวัดในโมเดลตัวนี้ยังอยู่บนพื้นฐานการประเมินซอฟต์แวร์ ซึ่งโมเดลตัวนี้ไม่เหมาะสมกับบริษัทขนาดใหญ่เพราะไม่สามารถประเมินในลักษณะที่ข้ามแผนกได้



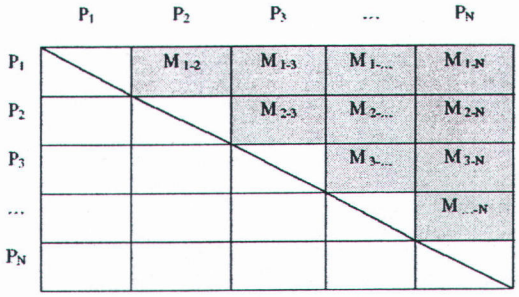
รูปที่ 1.3 Four-Phase model สำหรับอุตสาหกรรมการผลิตซอฟต์แวร์

ในการนำเทคนิคการกระจายหน้าที่เชิงคุณภาพ ใช้ร่วมการจัดการกระบวนการซอฟต์แวร์ [10] นำเสนอกรอบ SPI(Software Process Improvement) ใช้ CMMI เป็นโมเดลเป็นพื้นฐานโดยที่เทคนิคการกระจายหน้าที่เชิงคุณภาพ เป็นเครื่องมือ ดังรูป 1.4 แต่อย่างไรก็ตามโมเดลนี้ไม่ได้เสนอการใช้ประโยชน์จากการใช้สถิติการผลิตซอฟต์แวร์ครั้งก่อนๆมาร่วมพิจารณา



รูปที่ 1.4 แสดงการปรับปรุงกระบวนการซอฟต์แวร์ผ่าน CMM โดยใช้เทคนิคการกระจายหน้าที่เชิงคุณภาพ

ในการเรียงลำดับความสำคัญความต้องการ-ซอฟต์แวร์ และ การนำเทคนิคการกระจายหน้าที่เชิงคุณภาพ [11] นำเสนอโมเดล CBPA(Correlation-Base Priority) เพื่อที่จะรวมมุมมองความคาดหวังซอฟต์แวร์ด้วยกัน ดังรูป 1.5 อย่างไรก็ตามการนำข้อมูลจะคาดหวังลูกค้าหลายๆรายการก่อนนี้ น่าจะนำมาใช้ให้เกิดประโยชน์กับลูกค้าคนถัดไปเช่นการพยากรณ์ความต้องการซอฟต์แวร์รายถัดไปได้



รูปที่ 1.5 การรวมความคาดหวัง

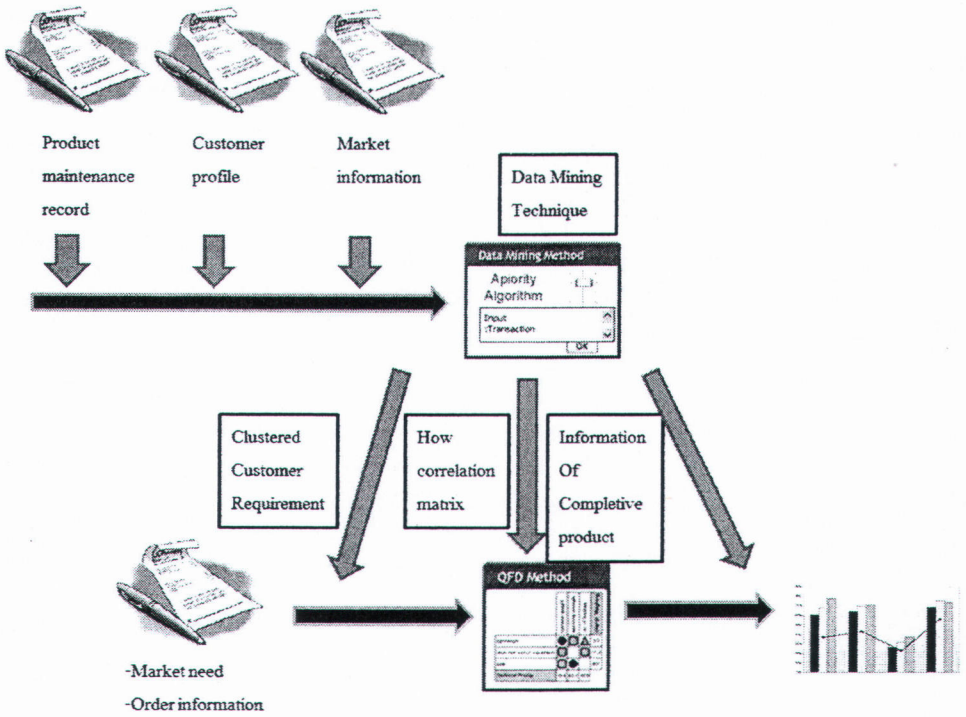
ในการนำเทคนิคเทคนิคการกระจายหน้าที่เชิงคุณภาพ ใช้ร่วมกับเทคนิคเหมืองข้อมูล [6] นำเสนอโมเดลที่นำเทคนิคเหมืองข้อมูลมาใช้งานร่วมกับเทคนิคการกระจายหน้าที่เชิงคุณภาพ เพื่อจัดซื้อวัตถุดิบให้กับโรงงานอุตสาหกรรมได้อย่างเหมาะสม อย่างไรก็ตามงานวิจัยนี้ไม่ได้กล่าวถึงกระบวนการวิเคราะห์ข้อมูลเชิงคุณภาพที่มีมาตรฐานสากลรองรับ ดังรูป 1.6

1.3 หลักการและเหตุผล

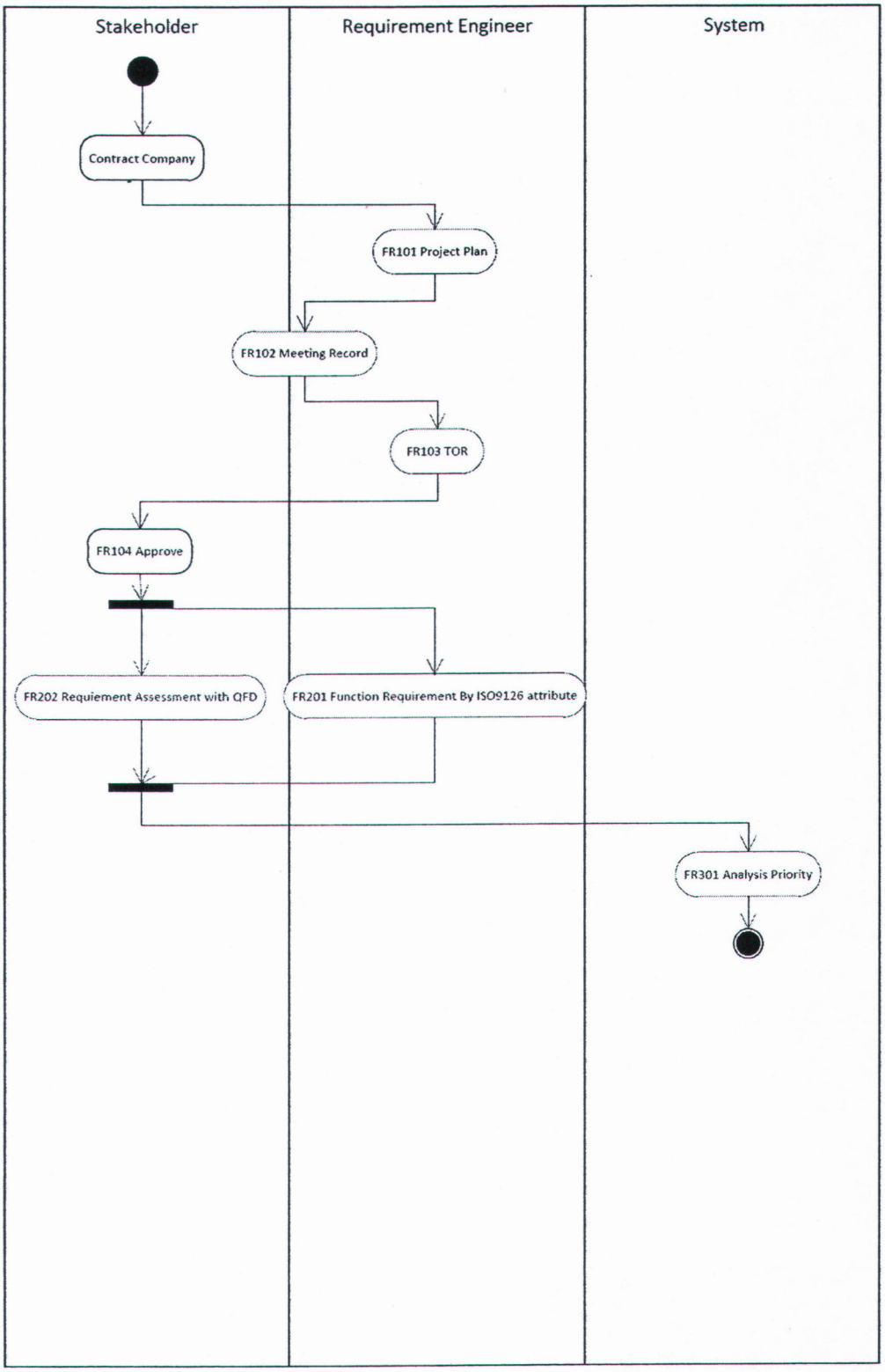
งานวิจัยนี้นำเสนอโครงสร้าง "Correlation Matrix Union Standard " (CMUS) เพื่อวิเคราะห์และเรียงลำดับความสำคัญ ความต้องการซอฟต์แวร์จากลูกค้าโครงการ ซึ่งช่วยต่อการประเมินงานโครงการซอฟต์แวร์และการว่าจ้างงานได้อย่างเหมาะสม ก่อนการอนุมัติตอบรับความต้องการซอฟต์แวร์ที่ยินยอมจากผู้มีส่วนได้ส่วนเสียโครงการซอฟต์แวร์ ก่อนที่จะนำความต้องการซอฟต์แวร์พัฒนาต่อ โดยโครงสร้าง CMUS ดังรูป 1.7 มีองค์ประกอบดังนี้

1. Project Plan ขั้นตอนวิธีการเก็บข้อมูลพื้นฐานของโครงการซอฟต์แวร์
2. Meeting Record ขั้นตอนวิธีการเก็บรายละเอียดการประชุม ระหว่างผู้มีส่วนได้ส่วนเสียทางธุรกิจ และ ทีมผู้พัฒนาซอฟต์แวร์
3. TOR(Term of requirement) ขั้นตอนวิธีการทำสัญญาข้อตกลงและขอบเขตในการทำงานตามโครงการซอฟต์แวร์
4. Approve ขั้นตอนวิธีการยินยอมอนุมัติ ตามขั้นตอน TOR เสนอ

5. Function Requirement ขั้นตอนวิธีการแปลงความต้องการลูกค้าสู่ความต้องการฟังก์ชั่นซอฟต์แวร์ที่ทำในโครงการซอฟต์แวร์
6. Software Requirement Assessment ขั้นตอนวิธีการนำฟังก์ชั่นซอฟต์แวร์ที่ได้ออกแบบให้ลูกค้าประเมินความสำคัญ จากมุมมองอันหลายความคาดหวังของลูกค้าโครงการต่อโครงการซอฟต์แวร์ และยังค่าผลกระทบความต้องการซอฟต์แวร์ระหว่างกัน
7. Analysis Function Requirement Priority ขั้นตอนวิธีการระบบวิเคราะห์ เรียงลำดับความสำคัญความต้องการซอฟต์แวร์อันหลากหลายมุมมองให้เหลือเพียงกลุ่มเดียวโดยพิจารณาจากคุณลักษณะความต้องการซอฟต์แวร์จากมาตรฐาน ISO9126 เพื่อสะดวกต่อการพัฒนาโครงการซอฟต์แวร์ต่อไป และวิเคราะห์ถึงความสัมพันธ์ความต้องการซอฟต์แวร์ที่เกี่ยวเนื่องกัน ด้วยเทคนิคการกระจายหน้าที่เชิงคุณภาพ



รูปที่ 1.6 โครงสร้างระบบสำหรับการเลือกผู้จำหน่ายบนพื้นฐานของเทคนิคการกระจายหน้าที่เชิงคุณภาพ และ เหมืองข้อมูล



รูปที่1.7 แสดงการทำงานของระบบ CMUS

8. TOR(Term of requirement) ขั้นตอนวิธีการทำสัญญาข้อตกลงและขอบเขตในการทำงานตามโครงการซอฟต์แวร์
9. Approve ขั้นตอนวิธีการยินยอมอนุมัติ ตามขั้นตอน TOR เสนอ
10. Function Requirement ขั้นตอนวิธีการแปลงความต้องการลูกค้าสู่ความต้องการฟังก์ชั่นซอฟต์แวร์ที่ทำในโครงการซอฟต์แวร์
11. Software Requirement Assessment ขั้นตอนวิธีการนำฟังก์ชั่นซอฟต์แวร์ที่ได้ออกแบบให้ลูกค้าประเมินความสำคัญ จากมุมมองอันหลายความคาดหวังของลูกค้าโครงการต่อโครงการซอฟต์แวร์ และยังค่าผลกระทบความต้องการซอฟต์แวร์ระหว่างกัน
12. Analysis Function Requirement Priority ขั้นตอนวิธีการระบบวิเคราะห์ เรียงลำดับความสำคัญความต้องการซอฟต์แวร์อันหลากหลายมุมมองให้เหลือเพียงกลุ่มเดียวโดยพิจารณาจากคุณลักษณะความต้องการซอฟต์แวร์จากมาตรฐาน ISO9126 เพื่อสะดวกต่อการพัฒนาโครงการซอฟต์แวร์ต่อไป และวิเคราะห์ถึงความสัมพันธ์ความต้องการซอฟต์แวร์ที่เกี่ยวเนื่องกัน ด้วยเทคนิคการกระจายหน้าที่เชิงคุณภาพ

1.4 วัตถุประสงค์ของการศึกษา

1 . เพื่อวิเคราะห์ความต้องการและลำดับความสำคัญความต้องการซอฟต์แวร์ ที่ตรงกับความต้องการของลูกค้าโครงการด้วยเทคนิค เทคนิคการกระจายหน้าที่เชิงคุณภาพ

2. เพื่อพัฒนาระบบจัดเก็บความต้องการซอฟต์แวร์ ตามมาตรฐาน ISO 9126

1.5 ขอบเขตการทำวิจัย

เพื่อให้บรรลุวัตถุประสงค์ในการศึกษาจึงกำหนดขอบเขตวิธีการศึกษาไว้ดังต่อไปนี้

1.5.1 ขอบเขตด้านประชากรและกลุ่มตัวอย่าง

ประชากร ที่ใช้ในการศึกษาครั้งนี้ เป็นนักศึกษาสาขาวิชาระบบสารสนเทศทางคอมพิวเตอร์-พัฒนาซอฟต์แวร์ ชั้นปีที่ 4 คณะบริหารธุรกิจและศิลปศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ตาก ภาคเรียนที่ 2 ปีการศึกษา 2554 จำนวน 60 คน

กลุ่มตัวอย่าง ที่ใช้ในการศึกษาครั้งนี้ นักศึกษาสาขาวิชาระบบสารสนเทศทางคอมพิวเตอร์-พัฒนาซอฟต์แวร์ เทียบโอน ชั้นปีที่ 4 คณะบริหารธุรกิจและศิลปศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ตาก ภาคเรียนที่ 2 ปีการศึกษา 2554 จำนวน 32 คน

1.5.2 ขอบเขตด้านตัวแปร

ตัวแปรต้น คือ เครื่องมือตามมาตรฐาน

ตัวแปรตาม ผลสัมฤทธิ์หลังจากการนักศึกษาวิเคราะห์ความต้องการซอฟต์แวร์ด้านความถูกต้อง และผลสัมฤทธิ์หลังจากการนักศึกษาวิเคราะห์ความต้องการซอฟต์แวร์ด้านการใช้เวลา

1.5.3 ขอบเขตด้านเนื้อหาศึกษาระบบการทำงานและคุณสมบัติทางด้านเทคโนโลยีของนักศึกษาในรายวิชาโครงการศึกษาเฉพาะบุคคล โดยมีข้อมูลที่ต้องการศึกษาดังต่อไปนี้

1) ขั้นตอนการทำงานด้านการสร้างโครงการซอฟท์แวร์ร่วมกัน ระหว่าง ลูกค้าโครงการและนักศึกษาในรายวิชาโครงการศึกษาเฉพาะบุคคล

2) ขั้นตอนการทำงานด้านการประชุมร่วมกัน ระหว่างลูกค้าโครงการและนักศึกษาในรายวิชาโครงการศึกษาเฉพาะบุคคล

3) ขั้นตอนการทำงานด้านการเก็บความต้องการซอฟท์แวร์ ระหว่างลูกค้านักศึกษาในรายวิชาโครงการศึกษาเฉพาะบุคคล

4) โครงสร้างระบบฐานข้อมูล

1.5.4 ขอบเขตด้าน วิเคราะห์ ออกแบบ และพัฒนาระบบจัดเก็บความต้องการซอฟท์แวร์ ตามมาตรฐาน ISO9126 ในการพัฒนา และระบบสามารถทำงานได้ดังต่อไปนี้

1) ระบบสามารถจัดการข้อมูลโครงการซอฟท์แวร์
   - ชื่อโครงการ
   - ขอบเขตโครงการ
   - ลูกค้าโครงการ
   - ผู้ร่วมทำโครงการ
2) ระบบสามารถจัดการข้อมูลการประชุมโครงการซอฟท์แวร์
   - หัวข้อการประชุม
   - วาระการประชุม
3) ระบบสามารถจัดการข้อมูลความต้องการซอฟท์แวร์
   - ลักษณะความต้องการซอฟท์แวร์ตามโครงสร้าง ISO9126
   - ป้อนค่าคะแนนความสำคัญ
4) ระบบสามารถรายงานและแบบฟอร์ม
   - รายงานโครงการซอฟท์แวร์

- รายงานการประชุม
- รายงานความต้องการซอฟท์แวร์
- รายงานสรุปลำดับความสำคัญความต้องการซอฟท์แวร์

1.5.5 ขอบเขตด้าน วิเคราะห์ ออกแบบ และพัฒนาระบบลำดับความต้องการซอฟต์แวร์ด้วยเทคนิคการกระจายหน้าที่เชิงคุณภาพในการพัฒนา และระบบสามารถทำงานได้ดังต่อไปนี้

1) ระบบสามรถจัดการวิเคราะห์ความสัมพันธ์ที่มีผลกระทบระหว่างความต้องการซอฟต์แวร์จากความคาดหวังอันหลากหลายมุมมองของลูกค้าโครงการ

2) ระบบสามารถจัดการลำดับความสำคัญความต้องการซอฟต์แวร์จากความคาดหวังอันหลากหลายมุมมองของลูกค้าโครงการ

1.5.6 พัฒนาโปรแกรมโดยยึดหลักการพัฒนาระบบตามกระบวนการทางวิศวกรรมซอฟต์แวร์ให้ครอบคลุม 15 กระบวนการดังต่อไปนี้

1) การเริ่มโครงการ (Project Initiation)
2) การสำรวจความต้องการ (Requirement Elicitation)
3) การวิเคราะห์ความต้องการของระบบ (System Requirement Analysis)
4) การออกแบบสถาปัตยกรรมของระบบ (System Architecture Design)
5) การวิเคราะห์ความต้องการของซอฟต์แวร์ (Software Requirement Analysis)
6) การออกแบบซอฟต์แวร์ (Software Design)
7) การสร้างซอฟต์แวร์ (Software Construction)
8) การประกอบซอฟต์แวร์ (Software Integration)
9) การทดสอบซอฟต์แวร์ (Software Testing)
10) การติดตั้งซอฟต์แวร์ (Software Installation)
11) การบำรุงรักษาซอฟต์แวร์ (Software & System Maintenance)
12) การบริหาร โครงร่างซอฟต์แวร์ (Configuration Management)
13) การบริหาร โครงการ (Project Management)
14) การประกันคุณภาพ (Quality Assurance)
15) การบริหารการเปลี่ยนแปลง (Change Request Management)

1.6 อุปกรณ์ที่ใช้พัฒนา

1.6.1 สถานที่ที่ใช้ดำเนินการ

คณะบริหารธุรกิจและศิลปศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา ตาก

1.6.2 อุปกรณ์ที่ใช้ในการวิจัย

1.6.2.1 ฮาร์ดแวร์ (Hardware)

- เครื่องคอมพิวเตอร์ส่วนบุคคล (PC)
- เครื่องคอมพิวเตอร์แม่ข่าย (Server)
- เครื่องพิมพ์

1.6.2.2 ซอฟต์แวร์ (Software)

- ไมโครซอฟต์วินโดว์เอ็กซ์พี (Microsoft Windows 7)
- ชุดไมโครซอฟต์ออฟฟิสแอ็กเซ็สเวอร์ชั่น 2010 (Microsoft Office Access)

1.7 ระยะเวลาในการดำเนินการวิจัย

ตั้งแต่เดือนมิถุนายน 2554 ถึง เดือนเมษายน 2555 รวมเป็นระยะเวลา 11 เดือน

ตาราง 1.1 ระยะเวลาดำเนินการวิจัย

ระยะเวลา / การดำเนินการ	2554							2555			
	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.
1. ศึกษาระบบงานจากระบบเดิม	■	■	■	■							
2. ศึกษาวิธีพัฒนาระบบ			■	■	■						
3. วิเคราะห์ระบบ					■	■					
4. ออกแบบระบบ						■	■				
5. พัฒนาระบบ								■	■	■	
6. ทดสอบระบบ								■	■	■	■
7. จัดทำเอกสาร	■	■	■	■	■	■	■	■	■	■	■
8. นำเสนอผลงานการค้นคว้าอิสระ			■	■	■						