บทที่ 2

แนวคิดทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในการศึกษาครั้งนี้ประกอบด้วย 2 ส่วนหลักๆ ได้แก่หลักการและทฤษฎีที่เกี่ยวข้องและงานวิจัยที่เกี่ยวข้อง ซึ่งในส่วนแรก (2.1) แนวคิดทฤษฎีที่เกี่ยวข้องประกอบไปด้วยหลักการดังนี้ (2.1.1) วิศวกรรมความต้องการ (2.1.2) การวิเคราะห์ความต้องการ (2.1.3) การจัดลำดับความต้องการก่อนหลัง (2.1.4) ขั้นตอนการจัดลำดับความต้องการก่อนหลัง (2.1.5) การหาค่าน้ำหนักความสำคัญของปัจจัย (2.1.6) เทคนิคสำหรับการจัดลำดับความต้องการก่อนหลัง

ในส่วน (2.2) เอกสารและผลงานวิจัยที่เกี่ยวข้อง ทำการแบ่งตามวิธีการที่นำมาใช้ภายในงานวิจัยซึ่งได้แก่ (2.2.1) วิธีการจัดกลุ่มความต้องการ (2.2.2) การประเมินประสิทธิภาพการจัดลำดับก่อนหลัง (2.2.3) เครื่องมือที่เหมาะสมกับการจัดลำดับความต้องการขนาดใหญ่ (2.2.4) ขั้นตอนการจัดลำดับความต้องการก่อนหลัง (2.2.5) ปัจจัยที่ใช้ในการพิจารณาการจัดลำดับก่อนหลังโดยมีรายละเอียดดังต่อไปนี้

2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง

ในการศึกษาครั้งนี้ประกอบด้วยหลักการและทฤษฎีที่เกี่ยวข้องดังนี้

2.1.1 วิศวกรรมความต้องการ

2.1.2 การวิเคราะห์ความต้องการ

2.1.3 การจัดลำดับความต้องการก่อนหลัง

2.1.4 ขั้นตอนการจัดลำดับความต้องการก่อนหลัง

2.1.5 การหาค่าน้ำหนักความสำคัญของปัจจัย

2.1.6 เทคนิคสำหรับการจัดลำดับความต้องการก่อนหลัง

2.1.1 วิศวกรรมความต้องการ

เป็นที่ทราบกันว่าความสำเร็จของโครงการพัฒนาซอฟต์แวร์นั้นวัดได้จากความพึงพอใจของผู้ใช้งานเป็นสำคัญ หากซอฟต์แวร์ที่ผลิตนั้นสามารถตอบโจทย์ความต้องการของผู้ใช้งานได้อย่างเต็มประสิทธิภาพแล้ว โอกาสในการที่โครงการพัฒนาซอฟต์แวร์จะประสบผลสำเร็จย่อมมีสูงยิ่งขึ้น

เพื่อให้เข้าใจหลักการและความสำคัญของวิศวกรรมความต้องการซึ่งเป็นกระบวนการหลักของการพัฒนาซอฟต์แวร์ ดังนั้นในส่วนนี้ของบทจะกล่าวถึงเนื้อหาดังนี้ คำจำกัดความของความต้องการ นิยามของความต้องการขนาดใหญ่ คำจำกัดความของผู้ถือผลประโยชน์ร่วม คำจำกัดความของวิศวกรรมความต้องการ และกระบวนการทางวิศวกรรมความต้องการ โดยมีรายละเอียดดังต่อไปนี้

2.1.1.1 คำจำกัดความของความต้องการ

ความหมายของความต้องการ(Requirement) ตาม IEEE 610.12-1990 (IEEE Standard 610.12-1990, 1990) ได้ให้คำนิยามของความต้องการไว้ดังนี้

> *"(1) A condition or capability needed by a user to solve a problem or achieve an objective.*
>
> *(2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.*
>
> *(3) A documented representation of a condition or capability as in (1) or (2)."*

จากนิยามข้างต้น สามารถอธิบายได้ว่า ความต้องการคือเงื่อนไขหรือความสามารถที่กำหนดโดยผู้ใช้งานเพื่อแก้ปัญหาหรือเพื่อให้บรรลุวัตถุประสงค์ ซึ่งระบบหรือส่วนประกอบของระบบจะต้องนำเสนอออกมาเพื่อให้สอดคล้องกับมาตรฐาน รายละเอียด หรือเอกสารเป็นทางการ

(Sommerville, 2004) ได้กล่าวไว้ว่า ความต้องการคือรายละเอียดของสิ่งที่จะพัฒนา ซึ่งอธิบายว่าระบบที่พัฒนาจะทำงานอย่างไร หรืออธิบายคุณลักษณะหรือคุณสมบัติของระบบ ความต้องการอาจจะเป็นข้อบังคับของกระบวนการในการพัฒนาระบบก็ได้

ในขณะเดียวกัน (Young, 2004) ได้กล่าวถึงความหมายและความสำคัญของความต้องการไว้ว่า ความต้องการคือคุณสมบัติที่จำเป็นของระบบ ที่ระบุความสามารถ คุณลักษณะเฉพาะ หรือปัจจัยคุณภาพ

ของระบบเพื่อให้ระบบมีคุณค่าและประโยชน์ต่อลูกค้าหรือผู้ใช้งาน ซึ่งความต้องการมีความสำคัญเนื่องจากความต้องการนำมาซึ่งส่วนประกอบหลักของขั้นตอนการพัฒนาทั้งหมดที่จะตามมา

ทั้งนี้ (Wiegers, 2003) กล่าวว่าความต้องการคือคุณสมบัติที่ผลิตภัณฑ์ต้องมีเพื่อสร้างคุณค่าให้แก่ผู้ถือผลประโยชน์ร่วม (Stakeholder)

จากที่กล่าวมาข้างต้นสามารถสรุปได้ว่าความต้องการคือความสามารถ คุณสมบัติของระบบ หรืออาจเป็นข้อบังคับของกระบวนการพัฒนาก็ได้ ที่กำหนดโดยผู้ใช้งานหรือลูกค้า เพื่อทำให้ระบบสร้างคุณค่าและประโยชน์ให้แก่ผู้ถือผลประโยชน์ร่วมได้ ความต้องการมีความสำคัญเนื่องจากเป็นส่วนประกอบพื้นฐานของขั้นตอนการพัฒนาทั้งหมดที่จะตามมา ยกตัวอย่างเช่น การออกแบบ การเขียนโปรแกรม การทดสอบ เป็นต้น

2.1.1.2 ความต้องการขนาดใหญ่

(Ma, 2009) ได้กำหนดนิยามของความต้องการขนาดใหญ่ (Large Requirement) ไว้ดังนี้ ความต้องการขนาดใหญ่หมายถึงกลุ่มความต้องการที่ประกอบด้วยความต้องการตั้งแต่ 100 ความต้องการขึ้นไป ดังนั้น ในงานวิจัยนี้ที่ทำการศึกษาเครื่องมือสำหรับการจัดการความต้องการขนาดใหญ่และทำการทดสอบเครื่องมือที่นำเสนอกับโครงการพัฒนาซอฟต์แวร์ที่มีความต้องการตั้งแต่ 100 ความต้องการขึ้นไป

2.1.1.3 ผู้ถือผลประโยชน์ร่วม

(Young, 2001) ได้ให้นิยามของผู้ถือผลประโยชน์ร่วม (Stakeholder) ไว้ว่าบุคคล กลุ่มคน หรือองค์กรที่มีผลต่อระบบ หรือที่ระบบส่งผลกระทบให้ ซึ่งได้แก่ ลูกค้า ผู้ใช้งาน และนักพัฒนา

(Sommerville, 2004) ได้กล่าวไว้ว่า ผู้ถือผลประโยชน์ร่วมคือผู้ที่ได้รับผลกระทบจากระบบและผู้ที่มีอิทธิพลต่อความต้องการของระบบทั้งทางตรงและทางอ้อม ยกตัวอย่างเช่น ผู้ใช้งานระบบ ผู้จัดการและบุคคลอื่นๆที่เกี่ยวข้องกับกระบวนการขององค์กร วิศวกรรมที่ดูแลเรื่องการพัฒนาและบำรุงรักษาระบบ และลูกค้า เป็นต้น

ในขณะที่ (Wiegers, 2003) กล่าวว่า ผู้ถือผลประโยชน์ร่วม คือบุคคล กลุ่มบุคคล หรือองค์กรที่เกี่ยวข้องกับโครงการ ซึ่งได้รับผลกระทบจากผลลัพธ์ของโครงการนั้น หรืออาจจะมีอิทธิผลต่อผลลัพธ์ของโครงการนั้นๆได้เช่นเดียวกัน

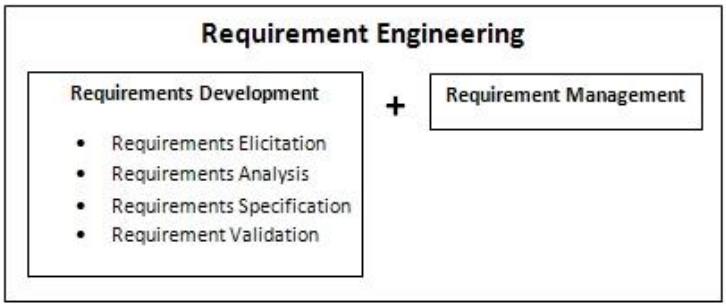
ดังนั้น จากที่กล่าวมาข้างต้นสามารถกล่าวได้ว่า ผู้ถือผลประโยชน์ร่วม คือบุคคล กลุ่มบุคคล หรือองค์กรที่มีอิทธิพลต่อผลลัพธ์ของระบบ หรือได้รับผลกระทบจากผลลัพธ์ของระบบ ทั้งทางตรงและทางอ้อม ยกตัวอย่างเช่น ผู้ใช้งานระบบ ลูกค้า และพัฒนาระบบ เป็นต้น

2.1.1.4 คำจำกัดความของวิศวกรรมความต้องการ

(Hull, 2010) ได้ให้ความหมายของวิศวกรรมความต้องการ ไว้ว่าวิศวกรรมความต้องการคือส่วนหนึ่งของวิศวกรรมระบบ (Systems Engineering) ซึ่งเกี่ยวข้องกับการค้นพบ การพัฒนา การติดตาม การวิเคราะห์ การจำแนกคุณสมบัติ การติดต่อสื่อสารและบริหารจัดการความต้องการที่ซึ่งกำหนดความสำเร็จของระบบในระดับนามธรรม

(Young, 2001) ได้นิยามความหมายของวิศวกรรมความต้องการว่าเป็นส่วนหนึ่งของวิศวกรรมระบบและวิศวกรรมซอฟต์แวร์ที่สนใจเรื่องกระบวนการความต้องการ ส่วน (Sommerville, 2004) ได้ให้ความหมายของวิศวกรรมความต้องการไว้ว่า วิศวกรรมความต้องการคือกิจกรรมทั้งหมดที่เกี่ยวข้องกับการค้นพบ การทำเอกสาร และการบำรุงรักษาชุดความต้องการสำหรับระบบคอมพิวเตอร์

ทั้งนี้ (Krams, 2010) กล่าวว่าวิศวกรรมความต้องการประกอบไปด้วยองค์ประกอบ 2 ส่วนคือ การวิเคราะห์ความต้องการ (Requirement Analysis) และการบริหารความต้องการ (Requirement Management) เช่นเดียวกับ (Wiegers, 2003) ที่ได้นิยามความหมายของวิศวกรรมความต้องการว่าเป็นขอบเขตของความรู้ที่รวมกิจกรรมทั้งวงจรชีวิตของโครงการ (Project Life Cycle) เข้าด้วยกัน ที่เกี่ยวข้องกับการทำความเข้าใจความสามารถที่จำเป็นและคุณลักษณะของผลิตภัณฑ์ ซึ่งประกอบไปด้วยสองกระบวนการหลักๆ ได้แก่ การพัฒนาความต้องการ (Requirement Development) และการบริหารความต้องการ (Requirement Management) และเป็นส่วนหนึ่งของวิศวกรรมระบบและวิศวกรรมซอฟต์แวร์ (Software Engineering) ดังแสดงตามภาพที่ 2.1

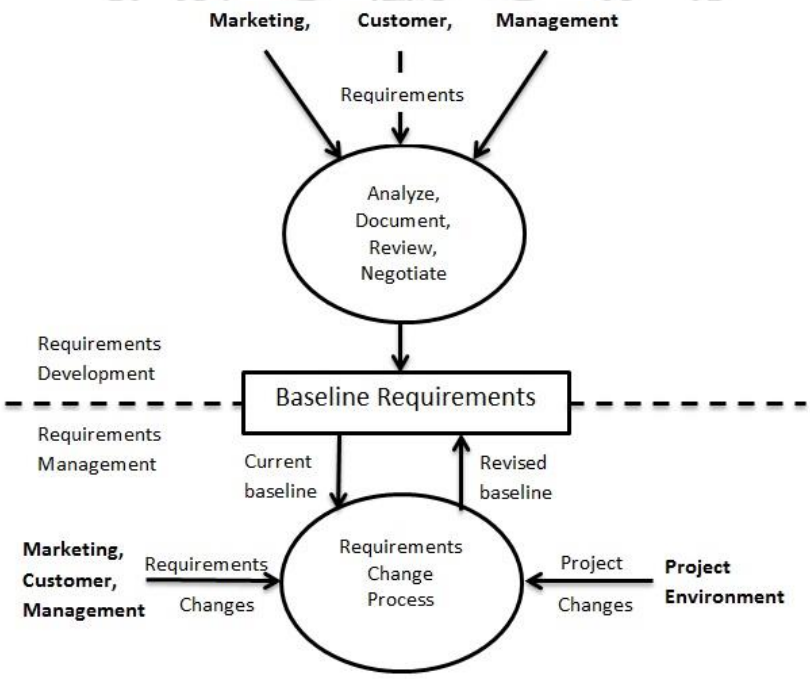


ภาพที่ 2.1 องค์ประกอบของวิศวกรรมความต้องการ (Weigers, 2003)

การพัฒนาความต้องการคือขั้นตอนที่เกี่ยวข้องกับกระบวนการที่เริ่มตั้งแต่การสกัดความต้องการ (Requirements Elicitation) หลังจากนั้น นำความต้องการที่ได้จากขั้นตอนแรกไปทำการวิเคราะห์ความต้องการ (Requirements Analysis) ถัดไปก็ทำการระบุรายละเอียดความต้องการ (Requirements Specification) และขั้นตอนสุดท้ายคือการตรวจรับรองความต้องการ (Requirements Validation)

ส่วนการบริหารความต้องการนั้นเกี่ยวข้องกับกระบวนการทั้งหมดที่เกี่ยวเนื่องกับการเปลี่ยนแปลงความต้องการของระบบ ซึ่งระหว่างการพัฒนาระบบซอฟต์แวร์นั้น อาจมีความต้องการใหม่ๆเกิดขึ้นมาหรือความต้องการที่มีอยู่แล้วอาจถูกเปลี่ยนแปลงไป ดังนั้นจึงต้องมีกระบวนการการบริหารความต้องการเพื่อให้แน่ใจความต้องการยังมีคุณภาพตามที่กำหนดไว้

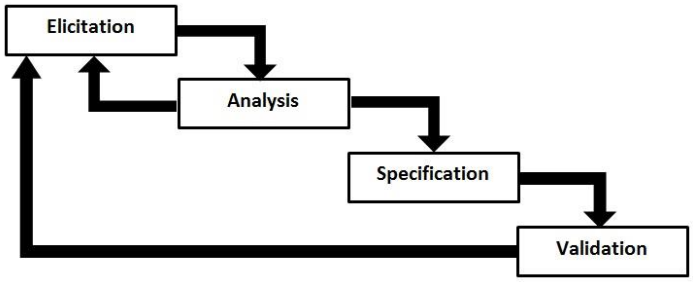
จากที่กล่าวมาข้างต้นสามารถสรุปได้ว่าวิศวกรรมความต้องการเป็นส่วนหนึ่งของวิศวกรรมระบบและวิศวกรรมซอฟต์แวร์ ที่แบ่งออกเป็นสองกระบวนการหลักๆ ได้แก่การพัฒนาความต้องการ (Requirements Development) และการบริหารความต้องการ (Requirements Management) ซึ่งการพัฒนาความต้องการเป็นกระบวนการที่เกี่ยวข้องกับการค้นพบ การวิเคราะห์ การจำแนกคุณสมบัติ และการตรวจสอบความถูกต้องของความต้องการ ส่วนการบริหารความต้องการเป็นกระบวนการที่เกี่ยวข้องกับการติดตามความต้องการไปยังทุกๆกระบวนการพัฒนา และการบริหารการเปลี่ยนแปลงที่เกิดขึ้นกับความต้องการ เพื่อให้เข้าใจภาพรวมและขอบเขตของระหว่างการพัฒนาความต้องการและการบริหารความต้องการ ดังแสดงในภาพ2.2



ภาพที่ 2.2 ขอบเขตระหว่างการพัฒนาความต้องการและการบริหารความต้องการ (Weigers, 2003)

2.1.1.5 กระบวนการทางวิศวกรรมความต้องการ

กระบวนการทางวิศวกรรมความต้องการ (Requirement Engineering Process) นั้นประกอบด้วย 4 กระบวนการย่อยได้แก่ การสกัดความต้องการ (Elicitation) การวิเคราะห์ความต้องการ (Analysis) การระบุรายละเอียดความต้องการ (Specification) และการตรวจรับรองความต้องการ (Validation) ดังแสดงในภาพ 2.3



ภาพที่ 2.3 กระบวนการทางวิศวกรรมความต้องการ (Pfleeger et al., 2006)

การสกัดความต้องการนั้นเป็นกระบวนการที่เกี่ยวข้องกับการระบุกลุ่มผู้ใช้งานและผู้ถือผลประโยชน์ของระบบ เพื่อทำการเก็บรวบรวมความต้องการมาจากบุคคลเหล่านั้น ซึ่งความต้องการที่เก็บมาได้สามารถอยู่ในรูปแบบความต้องการเชิงฟังก์ชั่น (Functional Requirement) และความต้องการที่ไม่ใช่ความต้องการเชิงฟังก์ชั่น (Nonfunctional Requirement)

ขั้นตอนการวิเคราะห์ความต้องการเกี่ยวข้องกับการกลั่นความต้องการเพื่อให้มั่นใจว่ามีความเข้าใจตรงกันกับผู้ถือผลประโยชน์ร่วมของระบบและทำการตรวจสอบอย่างละเอียดเพื่อหาข้อผิดพลาดความต้องการที่ตกหล่น และข้อบกพร่องอื่นๆ ซึ่งขั้นตอนนี้เป็นการแตกรายละเอียดความต้องการออกมา สร้างตัวต้นแบบ(Prototype) ประเมินความน่าจะเป็นที่จะพัฒนาระบบ และการต่อรองการจัดลำดับก่อนหลังของความต้องการ โดยจุดประสงค์ของขั้นตอนนี้คือการพัฒนาความต้องการที่มีคุณภาพ ที่ผู้บริหารโครงการสามารถประเมินความน่าจะเป็นของโครงการได้ตามจริงและนักพัฒนาสามารถนำความต้องการเหล่านี้ไปออกแบบ สร้าง และทดสอบได้

ขั้นตอนการระบุรายละเอียดความต้องการนั้นเกี่ยวข้องกับการทำเอกสารระบุความต้องการของซอฟต์แวร์ (Software Requirement Specification- SRS) ซึ่งต้องบันทึกความต้องการที่ได้มาให้สามารถอ่านและตรวจสอบได้ง่าย โดยที่ความต้องการทางธุรกิจ (Business Requirement) จะถูกระบุในเอกสารวิสัยทัศน์และขอบเขต ส่วนความต้องการจากผู้ใช้งาน (User Requirement) นั้นมักจะถูกจัดให้อยู่ในรูปแบบของยูสเคส (Use cases) ดังนั้นในเอกสารระบุความต้องการของซอฟต์แวร์จะต้องประกอบด้วยความต้องการเชิงฟังก์ชั่นและความต้องการที่ไม่ใช่ความต้องการเชิงฟังก์ชั่น

ขั้นตอนสุดท้ายของการพัฒนาความต้องการนั้นคือขั้นตอนการตรวจรับรองความต้องการ ซึ่งวัตถุประสงค์ของขั้นตอนนี้คือเพื่อสร้างความแน่ใจว่าความต้องการทั้งหมดนั้นมีความถูกต้อง มีคุณภาพตามที่กำหนดไว้ และจะต้องสอดคล้องกับสิ่งที่ลูกค้าต้องการเพื่อสร้างความพึงพอใจให้แก่ลูกค้า ดังนั้นขั้นตอนนี้จึงเป็นขั้นตอนที่สำคัญก่อนการนำความต้องการไปพัฒนาในขั้นตอนการออกแบบและพัฒนาซอฟต์แวร์ต่อไป

2.1.2 การวิเคราะห์ความต้องการ

เป็นที่ทราบกันดีว่าในการพัฒนาซอฟต์แวร์ในเชิงอุตสาหกรรมนั้น โครงการพัฒนาซอฟต์แวร์โครงการหนึ่งๆจำเป็นต้องจัดการกับความต้องการของผู้ที่เกี่ยวข้องกับโครงการเป็นจำนวนมาก แต่เนื่องจากทรัพยากรและงบประมาณที่จำกัด อีกทั้งยังมีข้อกำหนดเรื่องของเวลาการส่งมอบมาเกี่ยวข้อง ดังนั้นเป็นเรื่องยากอย่างยิ่งสำหรับผู้พัฒนาระบบซอฟต์แวร์ในการวิเคราะห์และเข้าใจความต้องการเพื่อจัดลำดับความสำคัญของความต้องการเพื่อให้สามารถนำส่งชิ้นงานที่มีคุณภาพแก่ผู้ใช้งานได้ ในส่วนนี้อธิบายถึงการวิเคราะห์ความต้องการ ดังนี้

(Young, 2001) ได้นิยามการวิเคราะห์ความต้องการ (Requirement Analysis) ไว้ว่าการวิเคราะห์ความต้องการคือวิธีการที่มีโครงสร้างเพื่อให้เข้าใจคุณสมบัติที่จะสร้างความพึงพอใจให้กับความต้องการของลูกค้า ทั้งนี้ (Sommerville, 2004) กล่าวว่าผลลัพธ์ที่ได้จากการวิเคราะห์ความต้องการคือ การระบุความต้องการที่ตกหล่น ความต้องการที่ไม่สอดคล้องกัน และความต้องการที่ขัดแย้งกัน

ในขณะที่ (Weigers, 2003)ได้กล่าวถึงการวิเคราะห์ความต้องการว่า การวิเคราะห์ความต้องการเกี่ยวข้องกับการกลั่นกรองความต้องการเพื่อทำให้แน่ใจว่าผู้ถือผลประโยชน์ร่วมเข้าใจความต้องการและการพิจารณาตรวจสอบข้อผิดพลาด สิ่งที่ละเลย การขาดตกปกพร่องของความต้องการ ซึ่งการวิเคราะห์ความต้องการเกี่ยวข้องกับกระบวนการหลักๆ เช่น การแตกรายละเอียดของความต้องการ การสร้างต้นแบบ การศึกษาความเป็นไปได้ และการเจรจาต่อรองความสำคัญของความต้องการ ดังมีรายละเอียดดังนี้

2.1.2.1 การเขียน Context Diagram

การเขียน Context Diagram ช่วยให้นักวิเคราะห์มองเห็นภาพรวมของระบบที่พัฒนาที่มีต่อระบบในสภาวะแวดล้อมรอบข้างอย่างไร ซึ่งแผนภาพนี้สามารถอธิบายขอบเขตของระบบที่กำลังพัฒนาและการเชื่อมต่อกับสิ่งอื่นๆ ไม่ว่าจะเป็น การเชื่อมต่อกับระบบอื่น การเชื่อมต่อกับผู้ใช้งาน การเชื่อมต่อกับฮาร์ดแวร์ เป็นต้น

2.1.2.2 การจัดลำดับความต้องการก่อนหลัง

การจัดเรียงลำดับความต้องการก่อนหลัง (Requirement Prioritization) นั้นเกี่ยวข้องกับการวิเคราะห์เพื่อหาความสำคัญในการพัฒนาของความสามารถของผลิตภัณฑ์ ยูสเคส (Use Case) หรือความต้องการใดๆ เพื่อให้ทราบว่าในการพัฒนาในแต่ละรอบจะประกอบด้วยความต้องการหรือความสามารถของผลิตภัณฑ์อะไรบ้าง

2.1.2.3 การสร้างส่วนติดต่อผู้ใช้งานและการสร้างตัวต้นแบบทางเทคนิค

การสร้างส่วนติดต่อผู้ใช้งานและการสร้างตัวต้นแบบทางเทคนิค (Create user interface and technical prototypes) ช่วยให้นักพัฒนาและผู้ใช้งานมองเห็นภาพระบบในทิศทางเดียวกัน และในบางครั้งที่ผู้ใช้งานไม่สามารถระบุความต้องการได้ การสร้างตัวต้นแบบจะช่วยให้ผู้ใช้งานมองเห็นภาพระบบที่กำลังจะพัฒนามากยิ่งขึ้น ช่วยให้เข้าใจว่าระบบที่จะพัฒนานั้นมีรูปร่างหน้าตาและความสามารถในการแก้ปัญหาด้านใดบ้าง ซึ่งช่วยให้สามารถเก็บความต้องการที่ตกหล่นไปได้

2.1.2.4 การกำหนดความต้องการตามระบบย่อย

ในระบบที่มีความซับซ้อนสูงและประกอบไปด้วยระบบย่อยเป็นจำนวนมากนั้น นักพัฒนามักจะแบ่งความต้องการของระบบออกไปตามระบบย่อย (Allocate requirements to subsystems) ได้แก่ แบ่งตามระบบซอฟต์แวร์ย่อย ฮาร์ดแวร์ (Hardware) และตามผู้ใช้งาน หรือส่วนประกอบย่อย เป็นต้น

2.1.2.5 การจำลองความต้องการ

การจำลองความต้องการ (Model the requirement) โดยการใช้รูปแบบการวิเคราะห์แบบกราฟิกในการจำลองความต้องการนั้นช่วยทำให้เข้าใจความต้องการมากยิ่งขึ้น โดยการใส่รายละเอียดให้แก่ความต้องการที่อยู่ในรูปแบบนามธรรมให้สามารถเข้าใจได้ ซึ่งรายละเอียดเหล่านี้อาจจะเป็นส่วนประกอบของ SRS (Software Requirement Specification) หรืออาจจะอยู่ในรูปแบบของส่วนติดต่อผู้ใช้งาน (User Interface) ที่แสดงในตัวต้นแบบ (Prototype)

การใช้รูปแบบการวิเคราะห์แบบกราฟิกช่วยให้มองเห็นความต้องการที่ไม่ถูกต้อง ที่ขัดแย้งกัน ที่ตกหล่น และที่มีมากเกินความจำเป็น ซึ่งรูปแบบการวิเคราะห์แบบกราฟิกนี้สามารถอยู่ในรูปแบบของแผนภาพการไหลของข้อมูล (Data Flow Diagram) แผนภาพแสดงความสัมพันธ์ของกลุ่มข้อมูล (Entity-Relationship diagram) แผนภาพการเปลี่ยนสถานะ (State-Transition diagram หรือ state

chart) แผนภาพคลาส (Class diagram) แผนภาพแสดงลำดับการทำงานของระบบ (Sequence Diagram) และแผนภาพแสดงความสัมพันธ์ (Interaction Diagram) เป็นต้น

2.1.2.6 การสร้างพจนานุกรมข้อมูล

การสร้างพจนานุกรมข้อมูล (Create a data dictionary) เป็นการนิยามข้อมูลและโครงสร้างข้อมูลทั้งหมดที่เกี่ยวข้องกับระบบเพื่อช่วยให้ผู้ที่เกี่ยวข้องกับการทำโครงการมีความเข้าใจเกี่ยวกับความหมายของข้อมูลใดๆได้ตรงกัน ซึ่งในขั้นตอนทางวิศวกรรมความต้องการนั้น พจนานุกรมข้อมูลเป็นตัวที่ช่วยที่ใช้ในการสื่อสารระหว่างนักพัฒนาและลูกค้าเพื่อสร้างความเข้าใจที่ตรงกัน

จากที่กล่าวมาข้างต้นสามารถสรุปได้ว่าการวิเคราะห์ความต้องการคือกระบวนการที่เกี่ยวข้องกับการวิเคราะห์คุณสมบัติต่างๆเพื่อให้เข้าใจความต้องการของลูกค้าและผู้ใช้งาน และเพื่อก่อให้เกิดประโยชน์สูงสุดให้แก่ผู้ถือผลประโยชน์ร่วมโครงการ อีกทั้งยังเป็นกระบวนการเพื่อตรวจสอบหาข้อผิดพลาดของความต้องการและหาความต้องการที่ขาดตกบกพร่องไป ซึ่งกระบวนการนี้เกี่ยวข้องกับกิจกรรมหลายๆกิจกรรมได้แก่การแตกรายละเอียดของความต้องการ การสร้างต้นแบบ การศึกษาความเป็นไปได้ และการเจรจาต่อรอง เป็นต้น

2.1.3 การจัดลำดับความต้องการก่อนหลัง

การจัดลำดับความต้องการก่อนหลัง (Requirement Prioritization) นั้นเป็นขั้นตอนที่เกี่ยวข้องกับการประเมินลำดับของความต้องการที่จะถูกผลิตก่อนหลัง จากที่ (Stanislav, 2012) ได้กล่าวไว้ว่าหนึ่งในเหตุผลที่ทำให้โครงการซอฟต์แวร์ล้มเหลว คือ การทำการเรียงลำดับความสำคัญที่ผิด หากเราเรียงลำดับความสำคัญผิด จะทำให้มีโอกาสสูงที่จะออกแบบโครงสร้างซอฟต์แวร์ผิด ก่อให้เกิดข้อผิดพลาดในการพัฒนาและมีค่าใช้จ่ายสูงขึ้น

เนื่องจากความต้องการมีลำดับความสำคัญที่ไม่เท่าเทียมกันจำเป็นต้องทำการประเมินเพื่อตัดสินใจว่าความต้องการใดบ้างที่จะถูกพัฒนารวมเข้าไปกับตัวผลิตภัณฑ์ตามเงื่อนไขต่างๆดังนี้ งบประมาณ เวลา ทรัพยากร- กำลังคน วัตถุดิบ และคุณภาพตามที่ลูกค้ากำหนด ดังนั้นในส่วนนี้ของบทจะกล่าวถึงการจัดลำดับความสำคัญของความต้องการ ซึ่งเนื้อหาประกอบด้วย ความหมายของการเรียงลำดับความต้องการ ประโยชน์ของการเรียงลำดับความต้องการ และเกณฑ์ที่ใช้ในการเรียงลำดับความต้องการ

2.1.3.1 คำจำกัดความของการจัดลำดับความสำคัญของความต้องการ

การจัดลำดับความสำคัญของความต้องการมีวัตถุประสงค์เพื่อกำหนดความต้องการที่สำคัญที่สุดและความต้องการที่เร่งด่วนที่สุด (M. Aasem et al., 2010) อีกทั้งยังช่วยให้การทำตารางกำหนดการนำความต้องการไปปฏิบัติสะดวกขึ้น (Firesmith, 2004)

ทั้งนี้ (Young, 2001) ได้กล่าวถึงการจัดลำดับความสำคัญของความต้องการว่าคือการจำแนกความต้องการออกเป็นกลุ่มๆตามความจำเป็นของระบบหรือตามความสามารถของระบบ

ดังนั้นสามารถกล่าวโดยสรุปได้ว่า การเรียงลำดับความต้องการตามความสำคัญนั้นคือขั้นตอนที่เกี่ยวข้องกับการระบุความต้องการหลัก โดยต้องคำนึงถึงข้อจำกัดในด้านต่างๆ ไม่ว่าจะเป็นงบประมาณ ระยะเวลาในการผลิต และทรัพยากรที่มีอยู่ ซึ่งช่วยในการทำตารางการผลิต เพื่อให้ผลิตภัณฑ์ที่ผลิตนั้นสามารถช่วยแก้ปัญหาของผู้ใช้งานได้และมีคุณภาพตามที่ลูกค้ากำหนดไว้

2.1.3.2 ประโยชน์ของการจัดลำดับความสำคัญของความต้องการ

(Sommerville, 2004) ได้กล่าวถึงประโยชน์ของการทำการเรียงลำดับความสำคัญของความต้องการไว้ว่า การจัดลำดับความสำคัญช่วยให้ผู้ถือผลประโยชน์ร่วมในโครงการสามารถตัดสินใจเพื่อระบุความต้องการหลักของระบบได้ อีกทั้งยังช่วยลดข้อโต้แย้งและช่วยในการเจรจาต่อรองระหว่างผู้ถือผลประโยชน์ร่วมอีกด้วย นอกจากนั้นการเรียงลำดับความสำคัญของความต้องการยังช่วยให้นักออกแบบสามารถตัดสินใจเรื่องสถาปัตยกรรมระบบและช่วยลดข้อขัดแย้งในการออกแบบที่จะเกิดขึ้นอีกด้วย

ในขณะเดียวกัน (Wiegers, 2003) ได้กล่าวว่าการเรียงลำดับความสำคัญของความต้องการเป็นวิธีการจัดการกับความต้องการด้วยทรัพยากรที่จำกัด ซึ่งช่วยให้สามารถวางแผนการผลิตเพื่อให้เกิดประโยชน์สูงสุดโดยใช้ต้นทุนต่ำสุด และลดข้อขัดแย้ง

จากที่กล่าวมาข้างต้นสามารถสรุปได้ว่าการจัดลำดับความสำคัญของความต้องการนั้นเป็นวิธีการในการค้นหาความต้องการหลักที่สำคัญและเร่งด่วนที่สุด เพื่อใช้ในการจัดเตรียมแผนดำเนินการพัฒนาให้เกิดประโยชน์สูงสุดแก่ผู้ถือประโยชน์ร่วมโดยใช้ต้นทุนที่ต่ำสุด ซึ่งการทำการจัดลำดับความสำคัญของความต้องการนั้นมีประโยชน์ช่วยในการตัดสินใจ สนับสนุนการเจรจาต่อรอง และลดข้อขัดแย้งระหว่างผู้ถือผลประโยชน์ร่วม

2.1.4 ขั้นตอนการจัดลำดับความต้องการก่อนหลัง

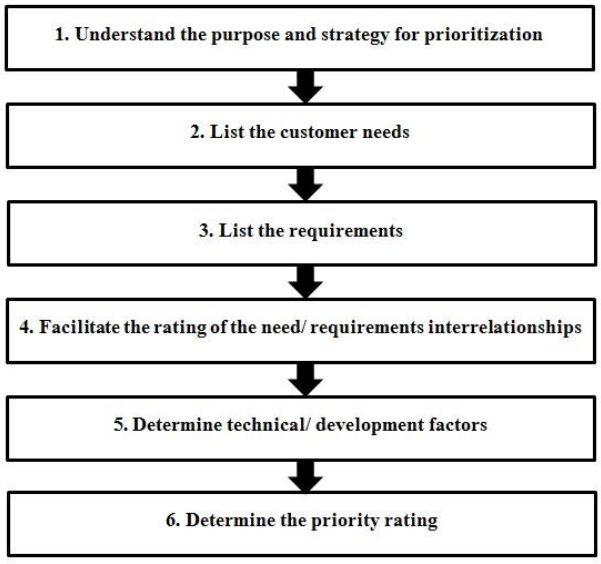
การจัดลำดับความสำคัญของความต้องการมีวัตถุประสงค์เพื่อระบุความต้องการหลักเพื่อการจัดลำดับการพัฒนาก่อนหลัง เพื่อให้การจัดลำดับความต้องการก่อนหลังมีประสิทธิภาพมากยิ่งขึ้น

ในส่วนนี้ของบทได้ทำการศึกษาขั้นตอนการจัดลำดับความต้องการก่อนหลังที่นำเสนอโดย บริษัท IAG Consulting ที่มีทั้งหมด 6 ขั้นตอนและ (Deneva, 2008) ที่มีทั้งหมด 3 ขั้นตอน โดยมีรายละเอียดดังนี้

2.1.4.1 ขั้นตอนการจัดลำดับความต้องการก่อนหลังที่นำเสนอโดยบริษัท IAG Consulting

ในส่วนนี้จะนำเสนอขั้นตอนการจัดลำดับความต้องการก่อนหลังที่นำเสนอโดยบริษัทIAG Consulting โดยมีรายละเอียดดังนี้

IAG Consulting ได้กำหนดขั้นตอนการจัดลำดับความต้องการก่อนหลังไว้ 6 ขั้นตอนดังนี้ การเข้าใจวัตถุประสงค์และกลยุทธ์ของการจัดลำดับก่อนหลัง (Understand the purpose and strategy for prioritization) การทำรายการความต้องการของลูกค้า (List the customer needs) การทำรายการความต้องกร (List the requirements) การจัดเตรียมข้อมูลก่อนการจัดลำดับก่อนหลัง (Facilitate the rating of the need/ requirements interrelationships) การกำหนดปัจจัย (Determine technical/ development factors) และการกำหนดลำดับก่อนหลัง (Determine the priority rating) ดังมีรายละเอียดดังนี้



ภาพที่ 2.4 ส่วนประกอบของการวิเคราะห์ความต้องการ (IAG Consulting, 2014)

2.1.4.1.1 เข้าใจวัตถุประสงค์และกลยุทธ์ของการจัดลำดับก่อนหลัง (Understand the purpose and strategy for prioritization)

ขั้นตอนแรกสุดของการทำการจัดลำดับก่อนหลังที่ IAG Consulting ซึ่งเป็นบริษัทที่มีประสบการณ์ด้านการบริหารจัดการ และให้คำแนะนำเกี่ยวกับการบริหารจัดการความต้องการมาเป็นเวลายาวนาน ได้กล่าวไว้ว่า แรกสุดต้องเข้าใจวัตถุประสงค์ก่อน ยกตัวอย่างวัตถุประสงค์ของการจัดลำดับความต้องการก่อนหลัง ได้แก่ การจัดลำดับก่อนหลังเพื่อตัดความต้องการบางอันออกไป เพื่อทำให้แน่ใจว่าสอดคล้องกับวัตถุประสงค์ เพื่อวางแผนตารางการทำงาน หรือเพื่อบริหารจัดการวัตถุประสงค์ที่เพิ่มขึ้น (Scope Creep) เป็นต้น หลังจากนั้นให้ทำการระบุสิ่งที่จะนำมาจัดลำดับก่อนหลังและระบุกลยุทธ์ของการจัดลำดับความต้องการก่อนหลัง

2.1.4.1.2 ทำรายการความต้องการของลูกค้า (List the customer needs)

ขั้นตอนต่อไปคือขั้นตอนการทำรายการความต้องการของลูกค้า ซึ่งเป็นขั้นตอนที่เกี่ยวข้องกับการคัดเลือกปัจจัยที่มีผลต่อการตัดสินใจต่อผู้ถือผลประโยชน์ร่วมของโครงการในการเลือกจัดลำดับก่อนหลังของความต้องการ โดยที่ปัจจัยเหล่านี้จะเป็นตัวกำหนดคุณลักษณะ ขอบเขต และฟังก์ชันของผลิตภัณฑ์ซอฟต์แวร์ที่กำลังจะพัฒนา ซึ่งขั้นตอนนี้รวมไปถึงการระบุผลประโยชน์ที่มีต่อธุรกิจ (Business Benefits) หรือวัตถุประสงค์ของการพัฒนาโครงการนี้ ยกตัวอย่างเช่น การลดค่าใช้จ่าย การส่งผลต่อรายได้ของธุรกิจ การใช้งานที่ง่ายขึ้น การสร้างประโยชน์ใช้สอย และอื่นๆ

2.1.4.1.3 ทำรายการความต้องการ (List the requirements)

ขั้นตอนการทำรายงานความต้องการนั้นเกี่ยวข้องกับการระบุความต้องการทั้งในรูปแบบของความต้องการเดี่ยวและกลุ่มของความต้องการ ซึ่งความต้องการอาจจะอยู่ในรูปแบบของกิจกรรม ฟังก์ชันการทำงาน หรืออื่นๆ โดยการเก็บไว้เป็นแถวและระบุตัวเลขเพื่อใช้ในการอ้างอิงถึงความต้องการนั้นๆ หากความต้องการมีมากกว่า 20-30 ความต้องการขึ้นไป ให้ทำการแยกกลุ่ม ซึ่งจะช่วยในการบริหารจัดการความต้องการได้ดีขึ้น

2.1.4.1.4 จัดเตรียมข้อมูลก่อนการจัดลำดับความต้องการก่อนหลัง (Facilitate the rating of the need/ requirements interrelationships)

การจัดเตรียมข้อมูลก่อนการจัดลำดับความต้องการก่อนหลังช่วยทำให้การทำงานมีศักยภาพมากยิ่งขึ้น ซึ่งประกอบด้วย 2 ทางเลือกได้แก่ การสำรวจก่อนการอภิปราย (Pre-session survey) และการออกเสียงระหว่างการอภิปราย (In-session voting)

ในขั้นตอนของการจัดเตรียมข้อมูลก่อนการจัดลำดับความต้องการก่อนหลังนั้นเกี่ยวข้องกับ 3 ขั้นตอนหลักๆดังนี้

1) การเลือกใช้ Numerically assigned scales เพื่อจัดกลุ่มของความต้องการ
2) การกำหนดปัจจัยความสำเร็จที่สำคัญสำหรับการจัดลำดับความต้องการ โดยมีขั้นตอนทั้งหมด 4 ขั้นตอนย่อยดังนี้
    a. ขั้นตอนเริ่มต้น (Set-up) คือขั้นตอนที่กำหนดความต้องการที่ถูกต้องและคัดเลือกความต้องการ
    b. ขั้นตอนการมีส่วนร่วม (Involvement) คือการผู้ถือผลประโยชน์ร่วมที่เกี่ยวข้องกับโครงการ
    c. ขั้นตอนการอำนวยความสะดวก (Facilitation) คือการจัดเตรียมข้อมูลก่อนการจัดลำดับความต้องการก่อนหลัง
    d. ขั้นตอนการเลือกใช้เครื่องมือ (Tools) การเลือกใช้วิธีการหรือเครื่องมือที่เหมาะสมสำหรับการออกความเห็นเรื่องการจัดลำดับก่อนหลัง
3) เมื่อเสร็จแล้วให้คำนวณค่าน้ำหนักความสำคัญของแต่ละปัจจัย

2.1.4.1.5 กำหนดปัจจัยทางเทคนิคหรือปัจจัยที่มีผลทางการพัฒนา (Determine technical/ development factors)

ขั้นตอนนี้มักจะแยกมาทำภายในกลุ่มนักพัฒนาในทีมพัฒนาและผู้เชี่ยวชาญด้าน SMEs เพื่อประมาณการความต้องการด้านทรัพยากรที่ใช้และราคา ยกตัวอย่างปัจจัยโดยทั่วไป เช่น ระดับของค่าความพยายาม (หรือความต้องการด้านทรัพยากร) ความยากทางเทคนิค และราคา เป็นต้น

2.1.4.1.6 กำหนดลำดับก่อนหลัง (Determine the priority rating)

การจัดลำดับก่อนหลังของแต่ละความต้องการโดยอาศัยค่าความสำคัญ และปัจจัยทางเทคนิคหรือปัจจัยที่มีผลต่อการพัฒนา โดยที่บันทึกการตัดสินใจก่อนหน้านี้ที่เกี่ยวข้องกับการจัดอันดับและการให้คะแนนเพื่อมีการลงมติเป็นเอกฉันท์ได้ง่ายขึ้น อีกทั้งช่วยให้การตัดสินใจมีความถูกต้องมากขึ้นและมีประสิทธิภาพตามเกณฑ์และปัจจัยที่ตกลงกันไว้ อีกทั้งไม่ลืมที่จะอ้างอิง ปรับปรุง และตรวจทานความต้องการและรายละเอียดของความต้องการ โดยอาศัยเทคนิคต่างๆได้แก่ การลงคะแนนแบบ Blind voting การคำนวณทางสถิติ การทำแผนที่ การอภิปรายกลุ่ม การตัดสินใจแบบกลุ่มร่วมกัน การใช้

กราฟแบบ 4 Quadrant graph หรือการอาศัยการจัดสินใจของผู้บริหาร เป็นต้น หลังจากนั้นทำการปรับปรุงเอกสารความต้องการตามคะแนนลำดับ รวมถึงดำเนินการต่อในเรื่องของการประเมินการจัดสินใจเรื่องความต้องการ การจัดตารางเวลา และความรับผิดชอบ อย่างต่อเนื่องเท่าที่จำเป็น

2.1.4.2 ขั้นตอนการจัดลำดับก่อนหลังที่ถูกนำเสนอโดย Deneva (Deneva, 2008)

ในส่วนนี้อธิบายถึงขั้นตอนการจัดลำดับก่อนหลังที่ถูกนำเสนอโดย Deneva (Deneva, 2008) โดยประกอบด้วยขั้นตอน 3 ขั้นตอนหลักๆดังต่อไปนี้ การหาความต้องการรองจากความต้องการหลัก การคำนวณหาค่าความสำคัญจากประโยชน์และราคา และการจัดเรียงลำดับความต้องการก่อนหลัง โดยมีรายละเอียดของแต่ละขั้นตอนดังนี้

1) การหาความต้องการรองจากความต้องการหลัก (Derive secondary requirements from primary requirements): ในขั้นตอนนี้เป็นขั้นตอนการหาความต้องการรองจากความต้องการหลักเพื่อใช้ในการหาค่าความสำคัญตามปัจจัยที่มีผลต่อการจัดลำดับก่อนหลังในขั้นตอนต่อไป
2) การคำนวณหาค่าความสำคัญจากประโยชน์และราคา (Determine importance based on benefit/cost): เมื่อได้ความต้องการมาแล้ว ขั้นตอนต่อไปคือการคำนวณค่าความสำคัญตามปัจจัยที่มีผลต่อการจัดลำดับก่อนหลัง ซึ่งในที่นี้หมายถึง ประโยชน์ (Benefit) และราคา (Cost)
3) การจัดเรียงลำดับความต้องการก่อนหลัง (Prioritize requirement): จากขั้นตอนที่ 2 เมื่อได้ค่าความสำคัญของแต่ละความต้องการมาแล้ว ขั้นตอนสุดท้ายคือการจัดเรียงลำดับความต้องการก่อนหลัง ผลลัพธ์ที่ได้จากขั้นตอนนี้คือความต้องการที่ถูกเรียงลำดับตามความสำคัญ

2.1.5 การหาค่าน้ำหนักความสำคัญของปัจจัย

ในงานวิจัยนี้มีขั้นตอนที่เกี่ยวข้องกับการหาค่าน้ำหนักความสำคัญของปัจจัยที่มีผลจากการจัดลำดับความต้องการก่อนหลัง ซึ่งมีรายละเอียดของวิธีการหาค่าน้ำหนักปัจจัยต่างๆดังนี้

2.1.5.1 การหาค่าน้ำหนักความสำคัญของปัจจัยด้วยวิธีการวิเคราะห์เชิงลำดับชั้น

การหาค่าน้ำหนักความสำคัญของตัวแปรด้วยวิธีการวิเคราะห์เชิงลำดับชั้น (Analysis Hierarchy Process; AHP) เป็นวิธีการหาลำดับความสำคัญของแต่ละปัจจัยที่เกี่ยวข้องโดยอาศัยแผนภูมิลำดับขั้น เพื่อเปรียบเทียบความสัมพันธ์ของปัจจัยทีละคู่ สำหรับการหาค่าน้ำหนักด้วยวิธีนี้นั้นได้แบ่งระดับการเปรียบเทียบออกเป็น 9 ระดับ ดังแสดงรายละเอียดในตารางที่ 2.1

ตารางที่ 2.1 แสดงมาตราที่ใช้ในการเปรียบเทียบแทนปัจจัยแต่ละคู่ด้วยวิธีการ AHP

(โสภาแดง, 2551)

ระดับความเข้มข้นของความสำคัญ	รายละเอียด
1	มีความสำคัญเท่ากัน
2	มีความสำคัญเท่ากันถึงปานกลาง
3	มีความสำคัญปานกลาง
4	มีความสำคัญปานกลางถึงค่อนข้างมาก
5	มีความสำคัญมากกว่าค่อนข้างมาก
6	มีความสำคัญค่อนข้างมากถึงมากกว่า
7	มีความสำคัญมากกว่า
8	มีความสำคัญมากกว่าถึงมากที่สุด
9	มีความสำคัญมากกว่าที่สุด

ทั้งนี้ ในการคำนวณหาค่าน้ำหนักความสำคัญของแต่ละปัจจัยด้วยวิธี AHP นั้นสามารถแบ่งขั้นตอนออกเป็น 3 ขั้นตอนดังนี้

1) กรอกคะแนนการเปรียบเทียบปัจจัยโดยใช้ตารางเมตริกซ์
2) คำนวณค่าอัตราความสอดคล้อง
3) การหาค่าน้ำหนักความสำคัญแต่ละปัจจัยด้วยการหาค่า Geometric Mean

ขั้นตอนแรกเริ่มจากการกรอกคะแนนการเปรียบเทียบปัจจัยโดยใช้เมตริกซ์ดังแสดงรายละเอียดการกรอกในตารางด้านล่างนี้

ตารางที่ 2.2 แสดงตัวอย่างการกรอกคะแนนการเปรียบเทียบปัจจัยโดยใช้ตารางเมตริกซ์

(โสภาแดง, 2551)

เกณฑ์ที่ใช้สำหรับการตัดสินใจ		**ปัจจัย**				
$C_1, C_2, C_3, \ldots, C_n$		A_1	A_2	A_3	…	A_n
ปัจจัย	A1	1	a_{12}	a_{13}	…	a_{1n}
	A2	$1/a_{12}$	1	a_{23}	…	a_{2n}
	A3	$1/a_{13}$	$1/a_{23}$	1	…	a_{3n}
	…	…	…	…	…	…
	An	$1/a_{1n}$	$1/a_{2n}$	$1/a_{3n}$	…	1

ขั้นตอนต่อมาคือการคำนวณหาค่าอัตราความสอดคล้องของเหตุผล เพื่อทำการตรวจสอบความสอดคล้องของการให้คะแนนค่าน้ำหนักของปัจจัยทั้งหมดว่ามีความสอดคล้องกันของเหตุผลหรือไม่ โดยอาศัยขั้นตอนย่อยทั้งหมด 4 ขั้นตอนย่อยดังนี้

1) คำนวณค่า λ_{max}

ในการคำนวณหาค่า λ_{max} นั้นสามารถคำนวณได้จากการนำเอาผลรวมของคะแนนเปรียบเทียบของแต่ละปัจจัยในแถวตั้งแต่ละแถว มาคูณด้วยผลรวมค่าเฉลี่ยในแถวนอนแต่ละแถว แล้วนำเอาผลคูณที่ได้มารวมกัน โดยผลลัพธ์ที่ได้จะมีค่าเท่ากับจำนวนปัจจัยทั้งหมด *(n)* ที่ถูกนำมาเปรียบเทียบ ซึ่งในกรณีที่คะแนนเปรียบเทียบในปัจจัยนั้นมีความสอดคล้องกันอย่างสมบูรณ์นั้น ค่าของ $\lambda_{max} = n$

2) คำนวณหาค่าดัชนีวัดความสอดคล้อง (Consistency Index: C.I.)

ในการคำนวณหาค่าดัชนีวัดความสอดคล้องสามารถใช้สูตรในการคำนวณหาค่าดัชนีวัดความสอดคล้องได้ดังนี้

$$C.I. = (\lambda_{max} - n) / (n-1) \qquad (2.1)$$

3) หาค่าดัชนีความสอดคล้องเชิงสุ่ม (Random Consistency Index: R.I.)

ค่าดัชนีความสอดคล้องเชิงสุ่มซึ่งเป็นค่าคงที่ที่ขึ้นอยู่กับขนาดของเมตริกซ์ โดยมีขนาดเริ่มตั้งแต่ 1x1 ไปจนถึงขนาด 15x15 โดยสามารถพิจารณาค่า R.I. ได้จากตารางที่ 2.3 นี้

ตารางที่ 2.3 ค่าของดัชนีความสอดคล้องตามขนาดของเมตริกซ์ (โสภาแดง, 2551)

N														
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59
R.I.														

4) คำนวณหาค่าความสอดคล้องกันของเหตุผล

ค่าความสอดคล้องกันของเหตุผลหรือค่า C.R. นั้นคือค่าอัตราส่วนการเปรียบเทียบระหว่างค่า C.I. ที่คำนวณได้จากเมตริกซ์ กับค่า R.I. ที่ได้จากการสุ่มตัวอย่างจากตาราง โดยสามารถแสดงได้ตามสูตรนี้

$$\text{C.R.} = \frac{C.I.}{R.I.} \qquad (2.2)$$

ซึ่งหากผลการคำนวณค่า C.R. พบว่าค่า C.R. น้อยกว่าหรือเท่ากับร้อยละ 10 หรือ C.R. <=0.10 สามารถสรุปได้ว่าการเปรียบเทียบคู่นั้นมีความสอดคล้องกันของเหตุผลอยู่ในเกณฑ์ที่ยอมรับได้

แต่ถ้า C.R. > 0.10 แล้ว จะถือว่าการเปรียบเทียบคู่นั้นอยู่ในเกณฑ์ที่ไม่สามารถยอมรับได้ ทั้งนี้แล้วผู้ที่ให้ข้อมูลนี้จำต้องทำการทบทวนการตัดสินใจและให้คะแนนการเปรียบเทียบปัจจัยและการจำลำดับความสำคัญในการเปรียบเทียบปัจจัยทีละคู่ใหม่

ดังนั้นแล้วหากค่าอัตราความสอดคล้องที่คำนวณได้อยู่ในเกณฑ์ที่ยอมรับได้แล้ว จะนำผลของการเปรียบเทียบปัจจัยเหล่านี้เข้าสู่กระบวนการคำนวณหาค่าน้ำหนักความสำคัญด้วยวิธีการ AHP ต่อไป

ขั้นตอนสุดท้ายเมื่อเสร็จสิ้นการกรอกคะแนนการเปรียบเทียบปัจจัยแต่ละคู่โดยใช้เมตริกซ์และคำนวณหาค่าอัตราความสอดคล้องแล้ว ขั้นตอนต่อมาคือการหาค่าน้ำหนักความสำคัญของแต่ละปัจจัยด้วยการหาค่า Geometric Mean ซึ่งค่าน้ำหนักนั้นจะพิจารณาจากค่า Geometric Mean ของแต่ละปัจจัยต่อค่า Geometric Mean รวมของทุกปัจจัย ซึ่งมีวิธีการคิดคำนวณแสดงดังตัวอย่างข้างล่างนี้

ตารางที่ 2.4 ตัวอย่างการคำนวณหาค่าน้ำหนักความสำคัญของแต่ละปัจจัย (โสภาแดง, 2551)

	Geometric Mean		**ค่าน้ำหนัก**
A_1	$(1 \times a_{12} \times a_{13} \times \ldots \times a_{1n})^{1/n}$	W_1	$W_1 / \mathbf{W_{total}}$
A_2	$((1/a_{12}) \times 1 \times a_{13} \times \ldots \times a_{1n})^{1/n}$	W_2	$W_2 / \mathbf{W_{total}}$
A_3	$((1/a_{13}) \times (1/a_{23}) \times 1 \times \ldots \times a_{1n})^{1/n}$	W_3	$W_3 / \mathbf{W_{total}}$
…	…	…	…
A_n	$((1/a_{1n}) \times (1/a_{2n}) \times (1/a_{3n}) \times \ldots \times 1)^{1/n}$	W_n	$W_n / \mathbf{W_{total}}$
	ผลรวม	$\mathbf{W_{total}}$	

2.1.5.2 การหาค่าน้ำหนักตัวแปรโดยใช้ลำดับความสำคัญ (Weight from Rank)

การหาค่าน้ำหนักของตัวแปรโดยใช้ลำดับความสำคัญเป็นวิธีการหาค่าน้ำหนักของตัวแปรหรือปัจจัย โดยคำนวณจากการลำดับความสำคัญของตัวแปรนั้นๆ ซึ่งสามารถคำนวณได้ตามขั้นตอน 2 ขั้นตอน ได้แก่ ขั้นตอนการจัดเรียงลำดับตัวแปรและขั้นตอนการคำนวณหาค่าน้ำหนักของตัวแปร ดังแสดงรายละเอียดนี้

ให้เริ่มจากการเรียงลำดับตัวแปรตามความสำคัญ โดยคำนึงถึงตัวแปรที่มีความสำคัญมากที่สุดจะมีลำดับความสำคัญที่ 1 ตัวแปรต่อมาที่มีความสำคัญรองลงมา จะมีลำดับความสำคัญที่ 2 ไปเรื่อยๆ จนถึงลำดับที่ n ซึ่ง n ในที่นี้คือจำนวนตัวแปรทั้งหมดที่ต้องการนำมาหาค่าน้ำหนัก

ยกตัวอย่างเช่น ปัจจัยที่มีผลต่อการจัดลำดับก่อนหลังความสำคัญของบริษัทหนึ่งมีดังต่อไปนี้คือ

- ความสำคัญลำดับที่ 1 คือ มูลค่าทางธุรกิจ
- ความสำคัญลำดับที่ 2 คือ ค่าใช้จ่าย
- ความสำคัญลำดับที่ 3 คือ เวลา
- ความสำคัญลำดับที่ 4 คือ ความเสี่ยง
- ความสำคัญลำดับที่ 5 คือ บทลงโทษ

หลังจากที่ได้ลำดับความสำคัญจากขั้นตอนที่ 1 แล้วนั้น ขั้นตอนต่อไปคือการคำนวณหาค่าน้ำหนักของตัวแปร (W_j) จากสมการ

$$W_j = \frac{\frac{1}{rj}}{\sum_{k=1}^{n}(\frac{1}{rk})} \quad (2.3)$$

โดยที่ rj คือลำดับของปัจจัยที่ j

rk คือลำดับของปัจจัยใดๆ ที่ k

และ n คือจำนวนปัจจัยทั้งหมด

ยกตัวอย่างการคำนวณหาค่าน้ำหนักของแต่ละปัจจัยมูลค่าทางธุรกิจที่ได้รับการจัดลำดับความสำคัญเป็นลำดับที่ 1 ได้ดังนี้

โดยที่ $r_1 = 1$ และ n=5

ดังนั้น $\frac{1}{r1} = \frac{1}{1}$

$\sum_{1}^{5}(\frac{1}{rk}) = \left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5}\right) = 2.28$

เพราะฉะนั้นค่าน้ำหนักของปัจจัยที่ได้ความสำคัญเป็นลำดับที่ 1 จึงมีค่าดังนี้

$w_1 = \frac{\frac{1}{1}}{2.28} = 0.439$

จากการคำนวณข้างต้นสามารถสรุปค่าน้ำหนักของแต่ละปัจจัยได้ตามตารางข้างล่างนี้

ตารางที่ 2.5 ค่าน้ำหนักของแต่ละปัจจัย (ที่มา: นักวิจัย)

ปัจจัย	**ลำดับความสำคัญ**	**ค่าน้ำหนักของปัจจัย**
มูลค่าทางธุรกิจ	1	0.439
ค่าใช้จ่าย	2	0.219
เวลา	3	0.146
ความเสี่ยง	4	0.110
บทลงโทษ	5	0.088

เมื่อพิจารณาจากตารางซึ่งแสดงค่าน้ำหนักของแต่ละปัจจัยซึ่งได้จากการคำนวณด้วยวิธีการคำนวณหาค่าน้ำหนักโดยใช้ลำดับความสำคัญ พบว่ามูลค่าทางธุรกิจซึ่งเป็นปัจจัยอันดับแรกมีน้ำหนักมากที่สุดที่ 0.439 รองลงมาคือ ค่าใช้จ่าย เวลา ความเสี่ยง และบทลงโทษ ซึ่งมีค่าน้ำหนักคือ 0.219, 0.146, 0.11, และ 0.088 ตามลำดับ

2.1.6 เทคนิคสำหรับการจัดลำดับความต้องการก่อนหลัง

เป็นที่ชัดเจนที่ว่าการคำนึงถึงการเรียงลำดับความต้องการจะไม่มีความจำเป็นเลยถ้าความต้องการทั้งหมดมีลำดับความสำคัญเท่ากันและทรัพยากรมีไม่จำกัด แต่ในความเป็นจริงแล้ว โครงการพัฒนาซอฟต์แวร์ล้วนเกี่ยวข้องกับข้อจำกัดเรื่องทรัพยากรและระยะเวลาโครงการที่ถูกกำหนดไว้ ดังนั้นการเรียงลำดับความต้องการจึงมีความจำเป็นอย่างยิ่ง ซึ่งการเรียงลำดับความต้องการนั้นเกี่ยวข้องกับการประเมินความสำคัญของความต้องการ หลังจากนั้นจึงนำความต้องการเหล่านั้นมาจัดลำดับตามความสำคัญ

ในส่วนนี้จะกล่าวถึงเครื่องมือที่ใช้ในการเรียงลำดับความสำคัญของความต้องการโดยจำแนกออกเป็นประเภทตามมาตรการวัดข้อมูล ซึ่งในการศึกษานี้ได้กล่าวถึงมาตรการวัด 3 แบบ ได้แก่มาตรนามบัญญัติ (Nominal Scale) มาตราเรียงลำดับ(Ordinal Scale) และมาตราอัตราส่วน (Ratio Scale) ดังแสดงในตารางที่ 2.6

ตารางที่ 2.6 เครื่องมือที่ใช้ในการเรียงลำดับความสำคัญของความต้องการโดยจำแนกออกเป็นประเภทตามมาตรการวัดข้อมูล

Scale	**Requirement prioritize techniques**
Nominal Scale	Numerical assignment (Grouping)(priority groups)
	MoScoW
	Quality Function Deployment (QFD)
Ordinal Scale	Simple Ranking
	Bubble Sort
	Binary Search Tree

ตารางที่ 2.6 เครื่องมือที่ใช้ในการเรียงลำดับความสำคัญของความต้องการโดยจำแนกออกเป็นประเภทตามมาตรการวัดข้อมูล (ต่อ)

Scale	**Requirement prioritize techniques**
Ratio Scale	Hundred Dollar Method or Cumulative Voting
	AHP
	Hierarchy AHP
	Minimal Spanning Tree
	Cost-Value Approach
	win-win (Theory W)

มาตรนามบัญญัติ (Nominal Scale) เป็นวิธีการที่ช่วยในการแบ่งกลุ่มความสำคัญของความต้องการ ซึ่งความต้องการที่อยู่ในกลุ่มเดียวกันจะมีความสำคัญที่เท่ากัน ข้อจำกัดของวิธีการนี้คือภายในกลุ่มความต้องการเดียวกัน จะไม่มีรายละเอียดที่บอกว่าความต้องการไหนสำคัญกว่ากัน ดังปัจจุบันมีเครื่องมือในการเรียงลำดับความสำคัญของความต้องการหลายอย่างที่จัดได้ว่าอยู่ในกลุ่มมาตรนามบัญญัติ ยกตัวอย่างเช่น Numerical Assignment ที่บางครั้งจะถูกเรียกว่าวิธีการแบบแบ่งกลุ่ม (Grouping) หรือ Priority Group วิธีการแบบ MoSCow และวิธีการแบบ Quality Function Deployment (QFD) เป็นต้น

Numerical Assignment หรือ Grouping หรือ Priority Groups เป็นวิธีการสำหรับเรียงลำดับความสำคัญของความต้องการหนึ่งที่ง่ายมาก ซึ่งมีการแบ่งความต้องการออกเป็นกลุ่มๆ ซึ่งจำนวนกลุ่มจะมีจำนวนหลากหลาย ข้อดีของวิธีการนี้คือไม่ซับซ้อน แต่มีข้อเสียคือความต้องการที่อยู่ในกลุ่มเดียวกันจะถือว่ามีความสำคัญเท่ากันโดยไม่มีการให้รายละเอียดว่าความต้องการไหนมากกว่ากันภายในกลุ่มเดียวกัน และวิธีการนี้เหมาะสำหรับความต้องการที่ชัดเจน

MoScoW เป็นเทคนิคที่พัฒนาโดย Dai Clegg ซึ่งเป็นเทคนิคที่ใช้ร่วมกับการพัฒนาโดยมีเวลาเป็นเครื่องจำกัด (Time boxing) โดยจะมีการกำหนดวันส่งงานไว้แล้วและเลือกทำเฉพาะความต้องการที่มีความสำคัญสูงสุดเท่านั้น ซึ่งจะพบมากในวิธีการพัฒนาโปรแกรมแบบเร็ว (Rapid Application Development: RAD) เช่น วิธีการพัฒนาระบบแบบพลวัติ (Dynamic Systems Development Method: DSDM) และวิธีการพัฒนาซอฟแวร์แบบคล่องแคล่วว่องไว (Agile Software Development) MoSCow เป็นเทคนิคที่มีพื้นฐานมาจาก Nominal Scale โดยการแบ่งความต้องการออกเป็น 4 กลุ่มหลักๆ ได้แก่

Must Have หมายถึงความต้องการที่จำเป็นต้องมีและขาดไม่ได้, Should Have หมายถึงความต้องการที่ควรจะมีในระบบ, Could Have หมายถึงความต้องการที่ควรจะมีในระบบแต่มีความสำคัญน้อยกว่า Should Have, และ Won't Have หมายถึงความต้องการที่พึงประสงค์ (Wish Lists) เป็นความต้องการที่ดี แต่ยังจะไม่ถูกดำเนินการในช่วงเวลานี้ ซึ่งข้อดีของวิธีการนี้คือ ใช้งานง่าย ใช้กำลังคนน้อย สามารถจัดการกับความต้องการขนาดใหญ่ได้ ผู้ถือผลประโยชน์ร่วมไม่จำเป็นต้องให้ความสนใจในความต้องการที่อยู่ในกลุ่มสุดท้ายหรือความต้องการพึงประสงค์ แต่มีข้อจำกัดตรงที่ไม่สามารถลำดับความสำคัญของความต้องการภายในกลุ่มได้

มาตราเรียงลำดับ (Ordinal Scale) เป็นมาตราช่วยในการเรียงลำดับความสำคัญของความต้องการ ซึ่งผลลัพธ์ที่ได้จากวิธีการนี้คือลำดับของความต้องการที่ถูกเรียงตามความสำคัญ จากความสำคัญมากไปยังความต้องการที่มีความสำคัญน้อยกว่า ถึงอย่างไรก็ตามวิธีการเรียงลำดับความสำคัญแบบมาตรเรียงลำดับนั้นยังมีข้อเสียเปรียบ ยกตัวอย่างเช่น วิธีการแบบมาตรเรียงลำดับนั้นสามารถระบุได้ว่าความต้องการหนึ่งมีความสำคัญมากกว่าอีกความต้องการ แต่ไม่ได้ระบุว่ามีความสำคัญมากกว่าเท่าไหร่ ทั้งนี้มีเครื่องมือหลายอย่างที่จัดว่าอยู่ในกลุ่มของมาตรเรียงลำดับ ได้แก่ วิธีการแบบ Simple Ranking วิธีการแบบ Bubble Sort และวิธีการแบบ Binary Search Tree เป็นต้น

Simple Ranking วิธีการเรียงแบบง่ายนี้เป็นวิธีการสำหรับเรียงลำดับความสำคัญของความต้องการจากมากไปหาน้อยตามลำดับ ซึ่งสามารถแทนเป็นตัวเลขจาก 1 ไปถึง n ซึ่งความต้องการที่มีความสำคัญมากจะอยู่ในอันดับที่ 1 และความสำคัญที่มีความสำคัญน้อยกว่าจะถูกเรียงลำดับถัดมาเรื่อยๆจนถึงความต้องการที่ n แต่วิธีการนี้มีข้อจำกัดคือเราไม่สามารถจำความต้องการในปริมาณมากๆได้ Miller 1956) กล่าวว่าเราไม่สามารถจดจำความต้องการได้มากกว่า 7 ความต้องการ ซึ่งบวกลบไม่เกิน 2 ความต้องการ และ (Hatton, 2007) กล่าวว่าเราไม่สามารถจำความต้องการได้มากกว่าหรือเท่ากับ 15 ความต้องการขึ้นไป ดังนั้นวิธีการนี้จึงไม่เหมาะสมสำหรับความต้องการขนาดใหญ่

Bubble Sort เป็นวิธีการหนึ่งในการจัดลำดับความสำคัญของความต้องการ โดยอาศัยหลักการที่ว่า นำความต้องการสองความต้องการมาเปรียบเทียบกัน และจะทำการสลับตำแหน่งของความต้องการเมื่อความต้องการอยู่ในตำแหน่งที่ผิด ทำการเปรียบเทียบและสลับตำแหน่งไปเรื่อยๆจนหมด ซึ่งผลลัพธ์จากการวิธีการนี้คือ ความต้องการที่ถูกเรียงลำดับแล้วจากมากไปหาน้อย โดยวิธีการนี้จะมีค่าความซับซ้อนของกรณีที่เลวร้ายที่สุดดังสมการ

$$\frac{n*(n-1)}{2} \text{ หรือ } O(n^2) \qquad (2.4)$$

Binary Search Tree เป็นวิธีการหนึ่งในการจัดลำดับความสำคัญของความต้องการ ซึ่งวิธีการนี้ช่วยสนับสนุนซอฟต์แวร์ที่มีการเปลี่ยนแปลงของความต้องการที่มีการเปลี่ยนแปลงเสมอ อีกทั้งยังสนับสนุนการเรียงลำดับความสำคัญของความต้องการปริมาณมากอีกด้วย ซึ่งให้ความแม่นยำมากกว่าเมื่อเปรียบเทียบกับการเรียงลำดับแบบ Simple Ranking ที่สำคัญคือเทคนิคนี้ยังมีความสามารถที่จะเพิ่มความต้องการเรื่อย และสามารถหาคำตอบได้โดยที่ไม่จำเป็นต้องรอให้เรียงลำดับครบทุกความต้องการ (Run time capability) โดยวิธีการนี้จะมีค่าความซับซ้อนของกรณีที่เลวร้ายที่สุดคือ

$$O(n \log n) \tag{2.5}$$

<u>มาตราอัตราส่วน (Ratio Scale)</u> เป็นหลักการในการเรียงลำดับความสำคัญของความต้องการที่สามารถระบุได้ว่าความต้องการหนึ่งๆมีความสำคัญมากกว่าความต้องการอีกอันหนึ่งเท่าไหร่ ซึ่งรายละเอียดความสำคัญที่มากกว่านี้จะปรากฏอยู่ในรูปแบบเปอร์เซ็นต์ ซึ่งมีเครื่องมือมากมายที่จัดอยู่ในรูปแบบมาตราอัตราส่วน ยกตัวอย่างเช่น วิธีการ Hundred Dollar Method หรือ Cumulative Voting วิธีการ Analytical Hierarchy Process (AHP) วิธีการแบบ Hierarchy AHP วิธีการ Minimal Spanning Tree วิธีการแบบ Cost-Value Approach และวิธีการแบบ win-win (Theory W) เป็นต้น

Hundred Dollar Method หรือ Cumulative Voting เป็นวิธีการที่สมมติให้เงินแก่ผู้ถือผลประโยชน์ร่วมของโครงการคนละ 100 $ หลังจากนั้นให้ผู้ถือผลประโยชน์ร่วมนำเงินไปกระจายลงกับความต้องการที่ต้องการสร้าง ผลลัพธ์ที่ได้จากวิธีการนี้คือลำดับความสำคัญของความต้องการที่มีข้อมูลบอกว่าความต้องการมีความสำคัญต่างกันเท่าไหร่ ซึ่งวิธีการนี้มีข้อจำกัดคือ ผู้ถือผลประโยชน์ส่วนใหญ่มักจะวางเงินไปกับความต้องการที่ชอบมากกว่าความต้องการที่มีความสำคัญจริงๆ และวิธีการนี้พบว่าจะมีความต้องการหลายๆความต้องการที่มีความสำคัญเท่ากัน

กระบวนการวิเคราะห์ตามลำดับชั้น (Analytic Hierarchy Process: AHP) เป็นวิธีการที่นำเสนอโดย Saaty (Saaty, 1980) ซึ่งมีหลักการเรียงลำดับความสำคัญโดยการเปรียบเทียบคู่ความสำคัญที่เป็นไปได้ทั้งหมด โดยเริ่มจากการที่ผู้ใช้งานระบุคุณลักษณะและตัวเลือกของแต่ละความต้องการเพื่อนำมาจัดเรียงลำดับขั้น หลังจากนั้นให้ผู้ใช้งานระบุความชอบในแต่ละคู่ความต้องการซึ่งมีค่าตั้งแต่ 1 จนถึง 9 โดยที่ 1 หมายถึง ความต้องการทั้งสองมีคุณค่าเท่ากัน และ 9 หมายถึงมีความชอบความต้องการนี้มากกว่าอีกความต้องการเป็นอย่างมาก หลังจากนั้นก็จะทำการแปลงค่าตัวเลขจากลำดับชั้นเพื่อเรียงลำดับความต้องการต่อไปโดยวิธีการ AHP เป็นวิธีการเหมาะสำหรับการพัฒนาแบบเส้นตรง (Linear Development Model) และความต้องการที่ชัดเจน มีการเปรียบเทียบแบบจับคู่ (Pair wise evaluation) เป็นวิธีการที่มีความซับซ้อนสูง แต่วิธีการนี้มีข้อจำกัดคือใช้เวลาและความพยายามมากในการเรียงลำดับความสำคัญ ซึ่งมีจำนวนการเปรียบเทียบความต้องการอยู่ที่

$$\frac{n*(n-1)}{2} \quad (2.6)$$

และมีค่าความซับซ้อนอยู่ที่

$$O(n^2) \quad (2.7)$$

ดังนั้นวิธีการนี้จึงไม่เหมาะกับการเรียงลำดับความสำคัญของความต้องการขนาดใหญ่ๆ

Hierarchy AHP เป็นวิธีการที่นำเสนอโดย Karlson (Karlsson, 1998) เป็นวิธีการที่มีการจัดเรียงตามลำดับขั้นของความต้องการ ซึ่งความต้องการทั่วไปจะอยู่ด้านบน และความต้องการที่เฉพาะเจาะจงจะอยู่ด้านล่าง หลังจากนั้นจะใช้วิธีการของ AHP ในการเรียงลำดับความสำคัญของความต้องการในแต่ละลำดับขั้น ซึ่งเป็นวิธีการลดเวลาในการเรียงลำดับเป็นอย่างมาก อีกทั้งยังช่วยลดการเปรียบเทียบที่ซ้ำซ้อนลง แต่วิธีการนี้มีข้อจำกัดคือมีค่าความสามารถในการระบุความไม่สอดคล้องกันของการตัดสินใจต่ำ และมีผลที่ยังไม่น่าเชื่อถือ

Minimal Spanning Tree เป็นวิธีการที่นำเสนอโดย Karlsson (Karlsson, 1998) ซึ่งเป็นวิธีการที่จับคู่เปรียบเทียบความต้องการเพื่อหาการเชื่อมต่อที่สั้นที่สุด ซึ่งวิธีการนี้สามารถลดจำนวนการเปรียบเทียบเป็นอย่างมากเมื่อเทียบกับวิธีการแบบ AHP ซึ่งจำนวนครั้งของการเปรียบเทียบอยู่ที่

$$n-1 \quad (2.8)$$

แต่วิธีการนี้มีข้อจำกัดคือมีค่าความสามารถในการระบุความไม่สอดคล้องกันของการตัดสินใจต่ำ และมีผลที่ยังไม่น่าเชื่อถือ

Cost-Value Approach เป็นวิธีการที่นำเสนอโดย Karlsson (Karlsson, 1997) ซึ่งเป็นวิธีการเพื่อเรียงลำดับความสำคัญของความต้องการโดยการตัดสินใจความต้องการแต่ละความต้องการจากประโยชน์แก่ผู้ใช้งาน (Value) และค่าใช้จ่ายในการดำเนินการ (Cost) ซึ่งวิธีการนี้เป็นวิธีการที่อาศัยหลักการของเทคนิค AHP ในการเปรียบเทียบมูลค่าและค่าใช้จ่าย ข้อจำกัดของวิธีการนี้คือใช้เวลาในการจัดเรียงลำดับมาก

<u>เทคนิครวม (Combination Techniques)</u> เป็นวิธีการเรียงลำดับความต้องการโดยการนำหลายๆ เทคนิคมารวมกัน เช่น Planning Game เป็นต้น

Planning Game เป็นวิธีการเรียงลำดับความสำคัญของความต้องการโดยการแบ่งความต้องการออกเป็น 3 กลุ่มหลักๆคือ ความต้องการที่สำคัญและจำเป็นต่อระบบ ความต้องการที่จำเป็นต้องมีใน

ระบบ และความต้องการที่ควรจะมีในระบบ เมื่อจัดกลุ่มความต้องการเสร็จแล้ว ก็จะทำการเรียงลำดับความสำคัญของความต้องการ โดยอาศัยวิธี Basic Ranking ซึ่งวิธีการนี้ได้รับความนิยมมากในการพัฒนาแบบ Agile ซึ่งวิธีนี้มีข้อจำกัดคือ ถ้าความต้องการภายในกลุ่มมีมากกว่า 15 ความต้องการขึ้นไป จะทำให้เกิดความยุ่งยากในการจัดเรียงลำดับความสำคัญ และความน่าจะเป็นที่จะเรียงผิดสูงขึ้น ซึ่งวิธีการนี้ไม่เหมาะสำหรับความต้องการขนาดใหญ่และระบบที่มีการเปลี่ยนแปลงของความต้องการสูงได้

2.2 งานวิจัยที่เกี่ยวข้อง

ในการศึกษาครั้งนี้ประกอบด้วยงานวิจัยที่เกี่ยวข้องดังนี้

2.2.1 วิธีการจัดกลุ่มความต้องการ

2.2.2 การประเมินประสิทธิภาพการจัดลำดับก่อนหลัง

2.2.3 เครื่องมือที่เหมาะสมกับการจัดลำดับความต้องการขนาดใหญ่

2.2.4 ขั้นตอนการจัดลำดับความต้องการก่อนหลัง

2.2.5 ปัจจัยที่ใช้ในการพิจารณาการจัดลำดับก่อนหลัง

โดยมีรายละเอียดดังนี้

2.2.1 วิธีการจัดกลุ่มความต้องการ

ในส่วนนี้กล่าวถึงทฤษฎีวิธีการการแบ่งกลุ่มความต้องการ โดยจะแสดงรายละเอียดของการแบ่งกลุ่มความต้องการแบบ 3 กลุ่ม และการแบ่งกลุ่มความต้องการออกเป็น 4 กลุ่ม ดังนี้

2.2.1.1 การแบ่งกลุ่มความต้องการออกเป็น 3 กลุ่ม

(Sommervile, 1997) ที่ได้ทำการแบ่งกลุ่มความต้องการออกเป็น 3 กลุ่มดังนี้ Essential Requirements, Useful capabilities และ Desirable capabilities ดังมีรายละเอียดดังนี้

ความต้องการที่จำเป็นต่อระบบ (Essential Requirements) คือความต้องการสำคัญที่จำเป็นต้องถูกนำไปพัฒนาเป็นระบบ ถ้าขาดความต้องการประเภทนี้ ระบบจะไม่สามารถทำงานได้ ได้แก่ ความต้องการที่เป็นการทำงานหนักของระบบ (Core Functionality) ความต้องการที่เป็นข้อกำหนดทางธุรกิจ (Business Constraints) ความต้องการที่เกี่ยวข้องกับข้อกฎหมายหรือระเบียบข้อบังคับต่างๆ (Legal and Regulation)

ความต้องการประเภทที่สองคือความต้องการที่เป็นความสามารถมีเป็นประโยชน์ต่อระบบ (Useful Capabilities) ดังนั้นถ้าความต้องการประเภทนี้ ไม่ได้ถูกพัฒนา จะส่งผลให้ระบบมีประสิทธิภาพในการทำงานลดลง ถ้ามีทรัพยากร งบประมาณ และเวลาเพียงพอ ความต้องการประเภทนี้ก็จะถูกพัฒนา

ความต้องการประเภทสุดท้ายคือความต้องการที่พึงประสงค์ (Desirable Capabilities) ซึ่งเป็นความต้องการที่ไม่ใช่การทำงานหลักของระบบและไม่ได้ช่วยเพิ่มมูลค่าตอบแทนการลงทุนหรือระดับความพึงพอใจของผู้ใช้งานเลย แต่เป็นความต้องการประเภทนี้เป็นความต้องการที่ช่วยให้ระบบมีความน่าสนใจมากยิ่งขึ้น ดังนั้นความต้องการประเภทนี้จะไม่ถูกนำไปพัฒนาเป็นส่วนหนึ่งของระบบในเวอร์ชันนี้ ซึ่งสามารถเก็บความต้องการประเภทนี้ไว้พิจารณาได้ในเวอร์ชันถัดไปของการพัฒนาระบบได้

2.2.1.2 การแบ่งกลุ่มความต้องการออกเป็น 4 กลุ่ม

(Wiegers, 2003) ที่ได้ทำการแบ่งกลุ่มความต้องการออกเป็น 4 กลุ่ม โดยคำนึงถึงความสำคัญ (Important) และความรีบเร่ง (Urgent) ดังนี้ ความต้องการที่สำคัญและรีบเร่ง (Important and Urgent), ความต้องการที่สำคัญแต่ไม่รีบเร่ง (Important and not Urgent), ความต้องการที่ไม่สำคัญแต่รีบเร่ง (not Important and Urgent) และ ความต้องการที่ไม่สำคัญและไม่รีบเร่ง (not Important and not Urgent) ดังมีรายละเอียดตามตารางนี้

ตารางที่ 2.7 ความสำคัญของความต้องการแบ่งตามความสำคัญและความเร่งรีบ

(Wiegers, 2003)

	Important	**Not Important**
Urgent	High Priority	Don’t do these!
Not Urgent	Medium Priority	Low Priority

ความต้องการที่สำคัญและรีบเร่ง ความต้องการกลุ่มนี้เป็นความต้องการที่มีความสำคัญสูง (High Priority Requirements) โดยที่ลูกค้าต้องการใช้งานและต้องการใช้งานในเวอร์ชันซอฟต์แวร์ที่กำลังพัฒนานี้ ซึ่งอาจจะเกี่ยวข้องกับเรื่องกฎหมายที่บังคับให้ความต้องการนี้ถูกพัฒนาหรืออาจเป็นเหตุผลทางธุรกิจที่ทำให้ความต้องการนี้มีความสำคัญและต้องพัฒนาโดยทันที

ความต้องการที่สำคัญแต่ไม่รีบเร่ง ความต้องการกลุ่มนี้เป็นความต้องการที่มีความสำคัญปานกลาง (Medium Priority Requirements) ซึ่งเป็นความต้องการที่ลูกค้าต้องการใช้งานแต่สามารถรอได้ในเวอร์ชันถัดไปของการพัฒนา

ความต้องการที่ไม่สำคัญแต่รีบเร่ง เป็นความต้องการที่ไม่มีความสำคัญแต่เร่งรีบ แนะนำให้ไม่ต้องใส่ใจกับความต้องการประเภทนี้ เนื่องจากความต้องการประเภทนี้ไม่ได้สร้างมูลค่าให้กับธุรกิจเลย

ความต้องการที่ไม่สำคัญและไม่รีบเร่ง ความต้องการกลุ่มนี้เป็นความต้องการที่มีความสำคัญต่ำ (Low Priority Requirements) เป็นความต้องการที่ผู้ใช้งานไม่จำเป็นต้องใช้และไม่มีความเร่งรีบในการพัฒนา

2.2.1.3 การแบ่งกลุ่มความต้องการในงานวิจัยนี้

(Hasan, 2010) ได้กล่าวว่าการแบ่งกลุ่มความต้องการนั้นสามารถแบ่งได้เป็นหลายกลุ่ม โดยจำนวนกลุ่มขึ้นอยู่กับสถานการณ์และความรู้ความเข้าใจของผู้ที่ทำการจัดลำดับความต้องการก่อนหลัง แต่อย่างไรก็ตาม Hasan ได้ทำการแนะนำให้เลือกวิธีการแบ่งกลุ่มความต้องการออกเป็น 3 กลุ่มเพื่อความง่ายและสะดวกต่อการจัดลำดับก่อนหลัง

ในขณะเดียวกันมีนักวิจัยหลายท่านได้ทำการแบ่งกลุ่มความต้องการออกเป็น 3 กลุ่ม ได้แก่ Sommervile (1997) ที่ได้ทำการแบ่งกลุ่มความต้องการออกเป็น 3 กลุ่มดังนี้ Essential Requirements, Useful capabilities และ Desirable capabilities

ทั้งนี้มีนักวิจัยอีกกลุ่มหนึ่งได้แก่ IEEE (1998) และ Weigers (1999) ทำการแบ่งความต้องการออกเป็น 3 กลุ่มเช่นกัน ได้แก่ Essential, Conditional และ Optional

ซึ่งในงานวิจัยในครั้งนี้จะแบ่งกลุ่มความต้องการออกเป็น 3 กลุ่มเช่นเดียวกัน ได้แก่ (1) ความต้องการที่จำเป็นต่อระบบ (Essential Requirements) คือความต้องการที่จำเป็นต้องนำมาพัฒนา (2) ความต้องการที่มีประโยชน์ต่อระบบ (Useful Capabilities) คือความต้องการที่ช่วยให้ระบบทำงานดีขึ้น และ (3) ความต้องการที่พึงประสงค์ (Desirable Capabilities) คือความต้องการที่ยังไม่จำเป็นต้องพัฒนาในรอบการผลิตนี้

2.2.2 การประเมินประสิทธิภาพการจัดลำดับก่อนหลัง

ในการศึกษาครั้งนี้คำนึงถึงการประเมินประสิทธิภาพการจัดลำดับก่อนหลังไว้ 2 ด้าน ได้แก่ การประเมินประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความพยายาม (Effort) และการประเมินประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความถูกต้อง Accuracy โดยมีรายละเอียดดังนี้

2.2.2.1 การประเมินประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความพยายาม (Effort)

ตามที่ Qiao (2009) ได้กล่าวถึงวิธีการวัดค่าความพยายามในการจัดลำดับความต้องการก่อนหลังไว้ 2 ค่าดังนี้คือ การวัดจากจำนวนความต้องการที่นำมาจัดเรียง (The number of Requirement) และวัดจากจำนวนการเปรียบเทียบของความต้องการ (The number of Comparison) ดังนั้นแล้วในการศึกษาครั้งนี้จะประเมินค่าความพยายามในการจัดลำดับก่อนหลังความต้องการด้วย 3 วิธีการได้แก่

1) การวัดจากจำนวนความต้องการที่นำมาจัดเรียง (The number of Requirement)
2) วัดจากจำนวนการเปรียบเทียบของความต้องการ (The number of Comparison)
3) วัดได้จากเวลาที่ใช้ในการเรียงลำดับความต้องการ

2.2.2.2 การประเมินประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความถูกต้อง Accuracy

ถึงแม้ Karlsson กล่าวว่าในการเปรียบเทียบประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความถูกต้อง มีการกำหนดนิยามที่ยังไม่ค่อยจะดีนัก เนื่องจากยังไม่มีแก่นสารที่ชัดเจนในการทำการเปรียบเทียบเพื่อวัดค่าความถูกต้อง (Karlsson, 2007)

อย่างไรก็ตาม Perini กล่าวว่าการประเมินประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความถูกต้อง นั้นสามารถวัดโดยการถามผู้ที่เกี่ยวข้องกับการจัดลำดับก่อนหลังว่ารายการความต้องการที่สร้างจากเครื่องมือใดที่ให้ลำดับความถูกต้องใกล้เคียงกับมุมมองของพวกเขา (Perini, 2003) ในกรณีที่ต้องการเปรียบเทียบผลลัพธ์ของความถูกต้องของเครื่องมือที่ต่างกัน Perini แนะนำให้ใช้เทคนิคแบบ Blind Test โดยให้ผู้ที่เกี่ยวข้องกับการจัดลำดับก่อนหลังทำการเลือกรายการความต้องการที่ใกล้เคียงกับมุมมองของพวกเขา โดยที่พวกเขาไม่ทราบว่ารายการความต้องการนั้นเกิดจากเครื่องมือใด

แต่ในงานวิจัยของ Karlsson ที่กล่าวถึงการวัดประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความถูกต้องนั้น Karlsson ก็ได้ทำการวัดค่าความถูกต้องด้วยวิธีการเดี่ยวกับ Perini ซึ่งใช้วิธีการเดียวกันและทำการทดสอบหลังจากที่ทำการจัดเรียงลำดับความต้องการแล้วเสร็จไปมากกว่าหนึ่งสัปดาห์

ดังนั้นแล้ว ในการศึกษาครั้งนี้จะทำการประเมินประสิทธิภาพการจัดลำดับก่อนหลังด้านค่าความถูกต้องโดยอ้างอิงวิธีการของ Perini และ Karlsson

2.2.3 เครื่องมือที่เหมาะสมกับการจัดลำดับความต้องการขนาดใหญ่

ในหัวข้อนี้ประกอบด้วยเนื้อหาดังนี้

2.2.3.1 การคัดเลือกเครื่องมือที่เหมาะสมกับการจัดลำดับความต้องการขนาดใหญ่

2.2.3.2 การจัดลำดับความต้องการด้วยวิธี Binary Search Tree

2.2.3.1 การคัดเลือกเครื่องมือที่เหมาะสมกับการจัดลำดับความต้องการขนาดใหญ่

ความสำคัญที่ไม่เท่ากันของความต้องการเป็นสิ่งที่ท้าทายผู้พัฒนาในการประเมินลำดับการพัฒนาความต้องการโดยเฉพาะอย่างยิ่งโครงการพัฒนาซอฟต์แวร์ แม้จะมีงานวิจัยหลายชิ้นที่ศึกษาเกี่ยวกับการจัดลำดับความสำคัญของความต้องการ แต่ก็ยังไม่มีหลักฐานที่เพียงพอที่แสดงให้เห็นว่าสามารถจัดการกับความต้องการขนาดใหญ่ได้ แม้นว่าจะมีเครื่องมือและวิธีการเรียงลำดับความสำคัญมากมาย แต่ยังเป็นเรื่องที่ยากสำหรับวิศวกรซอฟต์แวร์ที่จะเลือกเครื่องมือที่เหมาะสมที่สุดสำหรับความต้องการขนาดใหญ่ ในส่วนนี้ของบทจะกล่าวถึงวิธีการเลือกเครื่องมือในการเรียงลำดับความสำคัญของความต้องการที่สามารถจัดการกับความต้องการขนาดใหญ่ได้ ซึ่งเครื่องมือนี้จะต้องใช้ค่าความพยายามในการเรียงลำดับความสำคัญให้น้อยที่สุดด้วย

ถึงแม้จะมีวิธีการในการเรียงลำดับความต้องการมากมาย แต่ในทางปฏิบัตินั้น นักพัฒนามักจะเลือกวิธีการที่เหมาะสมที่สุดที่สอดคล้องกับลักษณะที่กำหนดไว้ ในส่วนนี้จะอภิปรายถึงวิธีการที่ใช้ในการเลือกเครื่องมือที่เหมาะสมที่สุดโดยคำนึงถึงค่าความพยายามที่ใช้ ซึ่งเครื่องมือที่ได้นั้นจะต้องใช้ค่าความพยายามน้อยลงและสนับสนุนการจัดลำดับสำหรับความต้องการขนาดใหญ่ได้

ในส่วนนี้ของการศึกษาวิธีการเลือกเครื่องมือจะแบ่งวิธีการในการเลือกเครื่องมือออกเป็น 2 ส่วนใหญ่ๆได้แก่ ขั้นตอนการประเมินเบื้องต้นและขั้นตอนในการตัดสินใจในการเลือกเครื่องมือ

ในขั้นตอนแรกของการเลือกเครื่องมือที่จะใช้ในการเรียงลำดับความต้องการนั้นมีจุดมุ่งหมายเพื่อจำแนกเครื่องมือในการเรียงลำดับที่สนับสนุนความยืดหยุ่นของจำนวนความต้องการได้ อีกทั้งเป็นเครื่องมือที่ใช้ค่าความพยายามน้อยสุดด้วย ซึ่งจากการศึกษาบทความเรื่องการจัดลำดับความต้องการก่อนหน้านี้สามารถสรุปได้ตามตารางที่ 2.8

ตารางที่ 2.8 คุณลักษณะของเทคนิคการจัดลำดับความสำคัญ (ที่มา: นักวิจัย)

	SR (IEEE ,1998)	BS (Karls-son, 1998)	BST (Beg, 2008)	$100 (Leffing well, 2003)	AHP (Karlsson, 1996; Saaty, 1990)	H-AHP (Karlsson, 1998)	MST (Karls -son, 1998)	CV (Karlsson and Ryan, 1997)	WW (Grüen bacher, 2000)
Support large scale of require-ments			/				/		
Pairwise Evaluation	/	/	/		/	/	/	/	/
Clear of Requirements									
Clear					/			/	/
Ambiguity				/					
Degree of change of requirements									
low					/	/		/	/
dynamic			/						
Criteria					Multiple			2	

โดยที่ SR คือ Simple Ranking, BS คือ Bubble Sort, BST คือ Binary Search Tree, $100 คือ Cumulative voting (100 dollars), AHP คือ Analytical Hierarchy Process, H-AHP คือ Hierarchy AHP, MST คือ Minimal Spanning Tree, CV คือ Cost-Value Approach, WW คือ Win-win (Theory W).

จากตารางที่ 2.8 สามารถสรุปได้ว่า มีเทคนิคหรือเครื่องมือในการเรียงลำดับความต้องการ 2 ประเภทที่สามารถสนับสนุนความต้องการขนาดใหญ่ได้เหล่านั้นคือ Binary search tree และ Minimal spanning tree

ขั้นตอนที่สองของการเลือกเครื่องมือคือขั้นตอนในการตัดสินใจในการเลือกเครื่องมือ ในขั้นตอนนี้เกี่ยวข้องกับการคิดคำนวณจำนวนการเปรียบเทียบความต้องการเพื่อประเมินวิธีที่ใช้ค่าความพยายามน้อยที่สุด

ตามที่ Qiao (2009) กล่าวว่ามีวิธีการในการวัดค่าความพยายามอยู่ 3 วิธี วิธีแรกนั้น ค่าความพยายามสามารถวัดได้จากจำนวนความต้องการที่จะถูกเรียงลำดับ วิธีที่สองในการวัดค่าความพยายามคือวัดจากจำนวนการเปรียบเทียบของความต้องการ และวิธีสุดท้ายคือค่าความพยายามสามารถวัดได้จากเวลาที่ใช้ในการเรียงลำดับความต้องการ ในส่วนนี้จะกล่าวถึงวิธีที่สองที่ใช้ในการวัดค่าความพยายามนั้นคือการวัดจากจำนวนการเปรียบความต้องการ

ตารางที่ 2.9 สมการที่แสดงจำนวนค่าความพยายามที่ใช้ (Beg, 2008)

วิธีการ/ เครื่องมือ/ เทคนิคในการจัดลำดับความต้องการก่อนหลัง	**จำนวนการเปรียบเทียบของความต้องการ (Number of requirement comparison)**
AHP	$\frac{n*(n-1)}{2}$ or $O(n^2)$
Hierarchy AHP	Less than $\frac{n*(n-1)}{2}$
Minimal Spanning Tree	$n-1$
Bubble sort	$\frac{n*(n-1)}{2}$ or $O(n^2)$
Binary search tree	$O(n\log n)$
Priority Groups	$O(n\log n)$

จากตารางข้างต้นแสดงให้เห็นถึงสมการที่ใช้คำนวณจำนวนการเปรียบเทียบความต้องการของแต่ละเครื่องมือ ซึ่งสมการเหล่านี้สามารถบ่งบอกถึงค่าความพยายามที่ใช้สำหรับการเรียงลำดับความต้องการได้ อ้างอิงจากตารางข้างต้นสามารถกล่าวได้ว่า Binary search tree มีจำนวนการเปรียบเทียบความต้องการน้อยที่สุด

อนึ่งในงานวิจัยของ Ahl (2005) ยังได้ทำการทดลองเปรียบเทียบเทคนิคการจัดลำดับความต้องการโดยวิธีการ Binary search tree เปรียบเทียบกับวิธีการจัดลำดับความต้องการด้วยวิธีอื่นๆอีก 4 วิธีการ Ahl สามารถสรุปได้ว่าวิธีการแบบ Binary search tree เป็นวิธีการที่ดีที่สุดสำหรับการจัดลำดับความต้องการก่อนหลังสำหรับความต้องการขนาดใหญ่ในมุมมองของความน่าเชื่อถือของผลลัพธ์การ

จัดลำดับก่อนหลัง ความง่ายในการใช้งานเครื่องมือ เวลาที่ใช้ในการจัดลำดับ และการรองรับจำนวนความต้องการจำนวนมาก

จากตารางข้างต้นทั้งสองและผลการทดลองของ Ahl (2005) สามารถกล่าวโดยสรุปได้ว่า Binary search tree เป็นเครื่องมือที่เหมาะสมที่สุดที่สอดคล้องกับเกณฑ์ที่ว่าสามารถลดค่าความพยายามและเหมาะสมกับความต้องการขนาดใหญ่

2.2.3.2 การจัดลำดับความต้องการด้วยวิธี Binary Search Tree

ในส่วนนี้อธิบายถึงทฤษฎีของ Binary Search Tree โดยประกอบด้วยเนื้อหาดังต่อไปนี้ คำจำกัดความของ Binary Search Tree ขั้นตอนการจัดลำดับความต้องการก่อนหลังด้วยวิธีการ Binary Search Tree โดยมีรายละเอียดดังนี้

B-TREE ประกอบด้วย Root ในที่นี้แทนด้วย root[T] และประกอบด้วยคุณสมบัติดังต่อไปนี้ (Beg, 2008)

1. ทุกๆ Node x ประกอบด้วยข้อมูลดังนี้
    a. n[x] คือจำนวน Key ที่เก็บไว้ใน node x
    b. ค่า key ของ n[x] ซึ่งค่านี้จะถูกเก็บในรูปแบบของลำดับที่มีค่าไม่ลดลง ดังนั้นแล้ว

        $$key_1[x] <= key_2[x] <= \ldots <= key_{n[x]}$$

    c. ค่า leaf[x] จะอยู่ในรูป Boolean ซึ่งถ้าเป็น leaf จะมีค่าเป็น TRUE และจะเป็น FALSE เมื่อ Node x นี้เป็น internal node.
2. ทุกๆ Internal Node x ประกอบด้วย pointer ทั้งหมด n[x] + 1 ไปยัง Node ลูกของมัน

    $$c_1[x], c_1[x], \ldots, c_{n[x]+1}[x]$$

3. $key_i[x]$ แยกช่วงของ key ที่ถูกจัดเก็บไว้ในแต่ละ sub tree

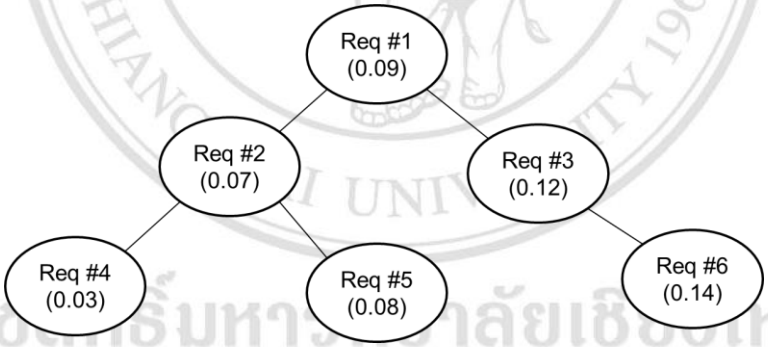
    ถ้า k_i คือ key ใดๆที่ถูกเก็บไว้ใน sub tree ที่มี root $c_1[x]$

    ดังนั้น $K_1 <= key_1[x] <= k_2 <= key_2[x] <= \ldots <= key_{n[x]}[x] <= key_{n[x]+1}$
4. ทุกๆ leaf มีความลึก (Depth) ซึ่งคือค่าความสูงของ Tree ซึ่งแทนด้วย h
5. Node จะประกอบด้วย key โดยที่มีขอบเขตค่าที่ต่ำสุดและค่าสูงสุดของจำนวน key ที่ node นั้นๆสามารถบรรจุได้ ซึ่งค่าขอบเขตนี้จะอยู่ในรูปของจำนวนเต็ม t >= 2 ซึ่งเรียกว่า ระดับต่ำสุดของ B-TREE (Minimum Degree)

a. ทุกๆ node ยกเว้น root ต้องมี key อย่างน้อย t-1 key โดยที่ internal node นอกเหนือจาก root ต้องมีลูก (Children) อย่างน้อย t ถ้า tree ไม่ใช่ nonempty ดังนั้นแล้ว root จะต้องมีอย่างน้อย 1 key

b. ทุกๆ node สามารถมี key มากสุดได้แค่ 2t-1 ดังนั้นแล้ว internal node สามารถมีลูกมากสุดได้ที่ 2t ถ้า node นั้นประกอบด้วย key ทั้งหมด 2t-1 แล้วเราจะเรียกว่า full

ในการจัดลำดับด้วย Binary Search Tree ประกอบด้วยขั้นตอนการจัดลำดับความต้องการก่อนหลัง โดยแทนแต่ละโหนด (Node) ของต้นไม้ด้วยความต้องการ โหนดที่อยู่ทางซ้ายประกอบด้วยความต้องการที่มีความสำคัญน้อยกว่าโหนดที่อยู่ฝั่งขวาของต้นไม้ ซึ่งการเรียงลำดับความสำคัญจะเริ่มต้นจากโหนดแรกบนสุด หากต้องการหาความต้องการที่มีค่าความสำคัญน้อยสุด จะทำการเปรียบเทียบความต้องการตามโหนดฝั่งซ้ายไปจนกระทั่งไม่มีโหนดใดๆให้เปรียบเทียบอีกต่อไป ในทางกลับกันหากต้องการค้นหาความต้องการที่มีความสำคัญสูงสุด จะเริ่มต้นทำการเปรียบเทียบตั้งแต่โหนดแรกไปทางฝั่งขวาจนกระทั่งไม่มีโหนดใดๆให้เปรียบเทียบอีกต่อไป ซึ่งผลลัพธ์ของขั้นตอนนี้คือความต้องการที่ถูกเรียงลำดับจากความต้องการที่มีความสำคัญมากสุดไปยังความต้องการที่มีความสำคัญน้อยสุด ดังแสดงในภาพที่ 2.5



ภาพที่ 2.5 Binary Search Tree

ภาพที่ 2.5 แสดงตัวอย่างการจัดลำดับความต้องการด้วย Binary Search Tree โดยประกอบด้วยความต้องการทั้งหมด 6 ความต้องการ ได้แก่ความต้องการที่ 1 (Req#1) จนถึงความต้องการที่ 6 (Req #6) โดยที่ความต้องการที่ 1-6 มีค่าความสำคัญดังนี้ 0.09, 0.07, 0.12, 0.03, 0.08, และ 0.14 ตามลำดับ ซึ่งหลังจากจัดลำดับด้วยวิธีการ Binary Search Tree ได้ผลลัพธ์ดังนี้ Req #6, Req #3, Req # 1, Req #5, Req #2, Req #4

2.2.4 ขั้นตอนการจัดลำดับความต้องการก่อนหลัง

จากการศึกษาทฤษฎีที่เกี่ยวข้องต่างๆเกี่ยวกับขั้นตอนการจัดลำดับความต้องการก่อนหลังแล้ว จึงได้มีการพัฒนาแนวคิดสำหรับขั้นตอนการจัดลำดับความต้องการก่อนหลัง ดังแสดงการพัฒนาแนวคิดดังภาพที่ 2.6

ภาพที่ 2.6 แสดงที่มาของแนวคิดขั้นตอนการจัดลำดับความต้องการก่อนหลังสำหรับความต้องการขนาดใหญ่ โดยการรวมวิธีการของ IAG และ Deneva เข้าด้วยกัน โดยประกอบด้วยขั้นตอนหลักๆ ดังนี้ ได้แก่ ขั้นตอนการกำหนดปัจจัย ขั้นตอนการเตรียมความต้องการที่จะนำมาจัดเรียง ขั้นตอนการแบ่งความต้องการออกเป็น 3 กลุ่ม ขั้นตอนการให้น้ำหนักปัจจัย ขั้นตอนการให้ค่าคะแนนแต่ละความต้องการตามปัจจัยต่างๆ ขั้นตอนการคำนวณหาค่าความสำคัญจากสมการ และขั้นตอนการจัดเรียงลำดับความต้องการ ซึ่งมีรายละเอียดของแต่ละขั้นตอนดังนี้

2.2.4.1 ขั้นตอนการกำหนดปัจจัย (List the factors)

เป็นขั้นตอนที่เกี่ยวข้องกับการศึกษาวิเคราะห์ปัจจัยที่มีผลต่อการจัดลำดับความต้องการก่อนหลังทั้งหมด การกำหนดปัจจัยที่ถูกต้องช่วยให้การจัดลำดับความต้องการก่อนหลังมีความถูกต้องแม่นยำยิ่งขึ้น

2.2.4.2 ขั้นตอนการเตรียมความต้องการที่จะนำมาจัดเรียง (List the requirements)

ในการจัดเตรียมความต้องการที่จะนำมาจัดเรียงนั้น ความต้องการเหล่านี้อาจจะอยู่ในรูปแบบของฟังก์ชันการทำงาน (Fuction) ยูสเคส (Usecase) หรือฟีเจอร์ (Feature) ซึ่งอาจจะเป็นความต้องการเดี่ยวๆ หรือเป็นกลุ่มของความต้องการก็ได้

2.2.4.3 ขั้นตอนการจัดกลุ่มความต้องการ (Facilitate the rating of the need/ requirements interrelationships)

ก่อนการทำการจัดลำดับความต้องการก่อนหลังนั้น การเตรียมข้อมูลความต้องการเป็นวิธีการที่ช่วยให้การจัดลำดับความต้องการก่อนหลังมีประสิทธิภาพมากยิ่งขึ้น ซึ่งในการนี้มีการจัดกลุ่มความต้องการออกเป็นกลุ่มๆเพื่อช่วยให้การจัดเรียงง่ายยิ่งขึ้น โดยการเริ่มจากการจัดเรียงความต้องการในกลุ่มที่มีความสำคัญมากสุดและสามารถนำความต้องการในกลุ่มที่มีความสำคัญน้อยสุดไปทำการจัดเรียงในรอบการผลิตต่อไป เพื่อช่วยลดภาระในการจัดลำดับความต้องการก่อนหลังลงได้

2.2.4.4 ขั้นตอนการให้น้ำหนักปัจจัย (Determines technical/ development factors)

หลังจากได้ปัจจัยที่มีผลต่อการจัดลำดับความต้องการก่อนหลังจากขั้นตอนที่ 1 (ขั้นตอนการกำหนดปัจจัย) แล้วนั้น จากนั้นผู้ที่ถือผลประโยชน์ร่วมของโครงการพัฒนาซอฟต์แวร์ต้องทำการระดมความคิดเห็นใจการกำหนดค่าน้ำหนักปัจจัยที่มีผลต่อการจัดลำดับความต้องการก่อนหลัง ซึ่งค่าน้ำหนักของปัจจัยที่ได้นี้ จะถูกนำมาคำนวณหาค่าความสำคัญของความต้องการเพื่อใช้ในการจัดลำดับความต้องการก่อนหลังต่อไป

2.2.4.5 ขั้นตอนการให้ค่าคะแนนแต่ละความต้องการตามปัจจัย (Determine importance)

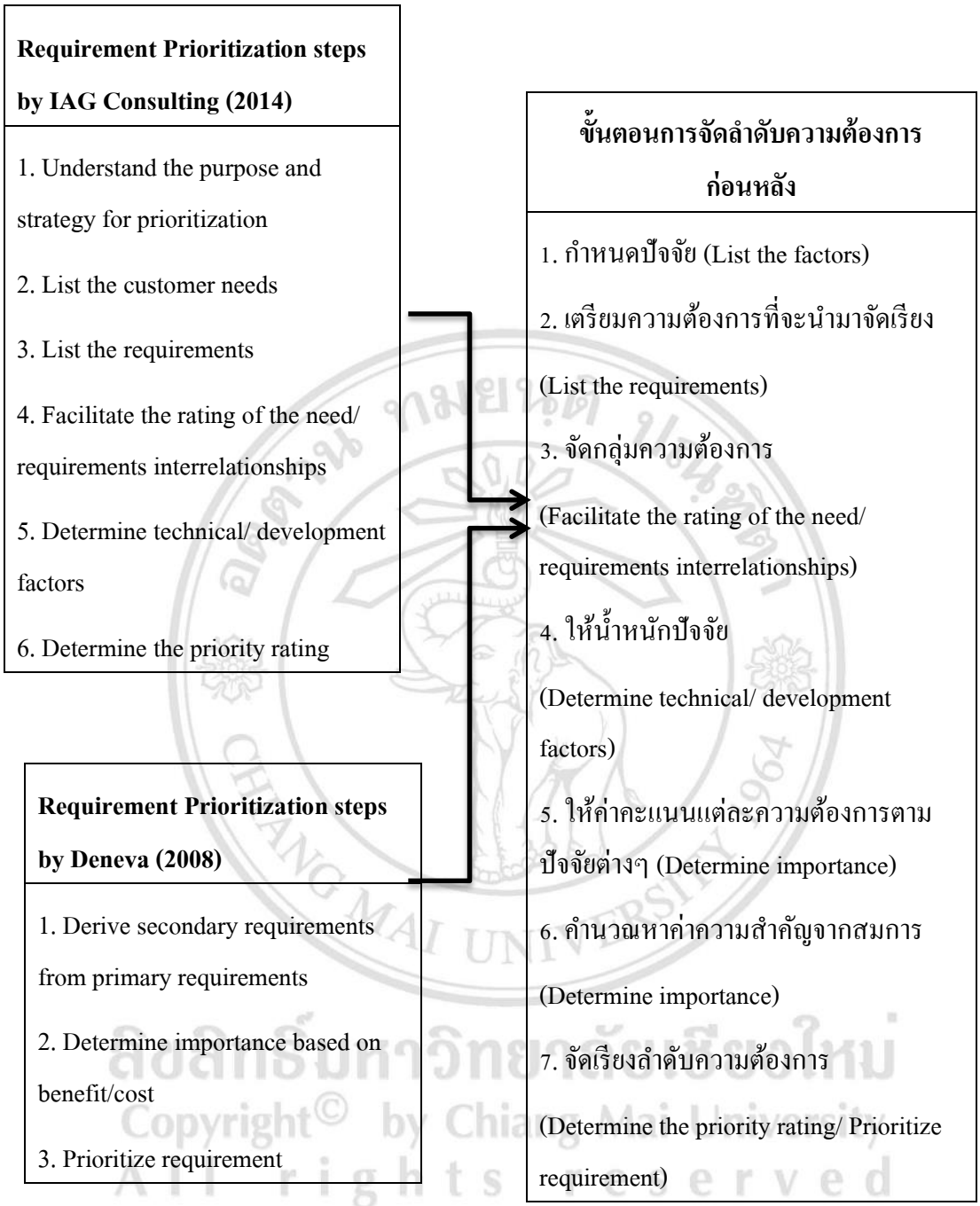
นำความต้องการที่ผ่านการจัดกลุ่มความต้องการแล้วมาทำการให้คะแนนตามปัจจัยที่ได้กำหนดไว้ ซึ่งผู้ที่ถือผลประโยชน์ร่วมจะต้องทำการให้คะแนนความต้องการ โดยพิจารณาตามเกณฑ์ที่กำหนดไว้ ซึ่งค่าคะแนนนี้จะถูกนำไปใช้ในการคำนวณหาค่าความสำคัญของความต้องการในขั้นตอนต่อไป

2.2.4.6 ขั้นตอนการคำนวณหาค่าความสำคัญจากสมการ (Determine importance)

เมื่อได้ปัจจัยที่มีผลต่อการจัดลำดับก่อนหลังจากขั้นตอนที่ 1 (ขั้นตอนการกำหนดปัจจัย) และทำการให้น้ำหนักปัจจัยในขั้นตอนที่ 4 (ขั้นตอนการให้น้ำหนักปัจจัย) และได้ค่าคะแนนของแต่ละความต้องการตามปัจจัยจากขั้นตอนที่ 5 แล้วนั้น (ขั้นตอนการให้ค่าคะแนนแต่ละความต้องการตามปัจจัย) ขั้นตอนต่อไปคือการคำนวณหาค่าความสำคัญของความต้องการ เพื่อนำค่าที่ได้นี้ ไปใช้ในขั้นตอนสุดท้ายนั้นคือขั้นตอนการจัดเรียงลำดับความต้องการ

2.2.4.7 ขั้นตอนการจัดเรียงลำดับความต้องการ (Determine the priority rating/ Prioritize requirement)

ขั้นตอนสุดท้ายหลังจากขั้นตอนการคำนวณหาค่าความสำคัญของความต้องการแล้วนั้น นำค่าที่ได้นี้มาทำการจัดลำดับก่อนหลัง ซึ่งผลลัพธ์ที่ได้จากขั้นตอนนี้คือรายการความต้องการที่ถูกจัดลำดับก่อนหลังตามความสำคัญแล้ว



ภาพที่ 2.6 ที่มาของแนวคิดขั้นตอนการจัดลำดับความต้องการก่อนหลัง (ที่มา: นักวิจัย)

2.2.5 ปัจจัยที่ใช้ในการพิจารณาการจัดลำดับก่อนหลัง

หลังจากที่ได้ทำการเก็บข้อมูลจากขั้นตอนที่ 4.1 ขั้นตอนต่อมาคือการวิเคราะห์หาปัจจัยที่ใช้ในการพิจารณาการจัดเรียงลำดับความต้องการก่อนหลัง

จากการศึกษาจากการทบทวนวรรณกรรมต่างๆที่เกี่ยวข้องกับการจัดลำดับความต้องการก่อนหลัง สามารถกำหนดปัจจัยที่มีผลต่อการจัดลำดับก่อนหลังที่ใช้ในการศึกษาวิจัยในครั้งนี้ ได้ดังตารางที่ 2.10

จากตารางที่ 2.10 พบว่าปัจจัยที่มีผลต่อการจัดลำดับความต้องการก่อนหลังมีดังต่อไปนี้ ผลประโยชน์ (Relative Benefit) บทลงโทษ (Relative Penalty) ราคา (Relative Cost) และความเสี่ยง (Relative Risk) โดยมีรายละเอียดแต่ละปัจจัยดังต่อไปนี้

2.2.5.1 ผลประโยชน์ (Relative Benefit)

ผลประโยชน์เป็นปัจจัยสำคัญที่มักจะถูกพิจารณาเมื่อจะทำการพัฒนาซอฟต์แวร์หนึ่งๆ ซึ่งผลประโยชน์ในที่นี้หมายถึงความต้องการนั้นๆสามารถสร้างประโยชน์ให้แก่ผู้ถือผลประโยชน์ร่วมของโครงการใดได้บ้าง ยกตัวอย่างเช่น ความต้องการที่สร้างมูลค่าให้กับธุรกิจ และความต้องการที่สร้างมูลค่าให้กับลูกค้า

2.2.5.2 บทลงโทษ (Relative Penalty)

บทลงโทษจะถูกหยิบยกขึ้นมาประเมินเมื่อความต้องการที่กำหนดไม่บรรลุเป้าหมาย ในบางครั้งบทลงโทษอาจจะเป็นสิ่งที่ไม่อยู่ตรงข้ามกับความสำคัญเสมอไป ยกตัวอย่างเช่น ความล้มเหลวที่จะทำให้สอดคล้องกับมาตรฐานซึ่งอาจจะเผชิญกับบทลงโทษที่รุนแรงได้ถึงแม้ว่าความต้องการนี้ๆมีความสำคัญต่ำสำหรับลูกค้า เป็นต้น หรือบทลงโทษจะเกิดขึ้นเมื่อความต้องการนั้นๆไม่สามารถสร้างรายได้ให้กับลูกค้าและสามารถสร้างผลิตภัณฑ์ที่สอดคล้องกับความต้องการหรือไม่เหมาะสมกับตลาด เป็นต้น

2.2.5.3 ราคา (Relative Cost)

ราคาสำหรับการพัฒนาโครงการนั้นจะถูกประมาณการโดยทีมนักพัฒนา โดยที่การประมาณการด้านราคานั้นวัดจาก ความซับซ้อนของความต้องการ ความสามารถในการนำสิ่งที่มีอยู่แล้วกลับมาใช้ใหม่ จำนวนเอกสารและจำนวนการทำการทดสอบที่จำเป็นต้องเตรียม ความยากง่ายของเทคโนโลยีที่ใช้ในการพัฒนา เป็นต้น ซึ่งราคานั้นมักจะอยู่ในรูปแบบของจำนวนเวลาที่พัฒนา (Staff hours) หรือที่เรียกว่าค่าความพยายามในการพัฒนา (Effort) ดังนั้นราคาซอฟต์แวร์หลักๆมักจะแปรผันกับจำนวนชั่วโมงที่ใช้ในการพัฒนาซอฟต์แวร์นั้นๆ

2.2.5.4 ความเสี่ยง (Relative Risk)

ทุกๆโครงการพัฒนาซอฟต์แวร์มักจะเผชิญกับความเสี่ยง ซึ่งถ้าพูดถึงความเสี่ยงในด้านของการบริหารจัดการความเสี่ยงนั้นมักจะครอบคลุมถึงความเสี่ยงจากปัจจัยภายในและความเสี่ยงจากปัจจัยภายนอก ซึ่งความเสี่ยงจากปัจจัยภายในได้แก่ ความเสี่ยงทางเทคนิค และความเสี่ยงทางด้านการตลาด เป็นต้น ส่วนความเสี่ยงจากปัจจัยภายนอกนั้น ได้แก่กฎข้อบังคับต่างๆหรือความเสี่ยงจากซัพพลายเออร์ เป็นต้น โดยที่ระดับของความน่าจะเป็นในการเกิดขึ้นของความเสี่ยงและผลกระทบที่จะเกิดขึ้นจากความเสี่ยงมักจะถูกพิจารณา

การบริหารจัดการความเสี่ยงมักจะถูกกล่าวถึงเมื่อทำการวางแผนการพัฒนาความต้องการโดยการระบุความเสี่ยงที่อาจจะเป็นสาเหตุที่ทำให้เกิดความยุ่งยากระหว่างการพัฒนาซอฟต์แวร์ ยกตัวอย่างความเสี่ยงที่เกิดขึ้นได้แก่ ความเสี่ยงด้านประสิทธิภาพ (Performance Risks) ความเสี่ยงด้านกระบวนการ (Process Risks) ความเสี่ยงของการจัดกำหนดการ (Schedule Risks) เป็นต้น

ตารางที่ 2.10 ปัจจัยที่มีผลต่อการจัดลำดับความต้องการก่อนหลัง (ที่มา: ทบทวนวรรณกรรม)

	Benefit		**Penalty**	**Cost**			
Authors	**Business value**	**Customer value**	**Law**	**Impl. cost**	**Impl. time**	**Easy of technology impl.**	**Risk**
Robertson, 2006	/	/	/	/	/	/	
Wiegers, 2003	/		/	/			/
Berander			/	/	/		/
Akao, 1990		/					
Ruhe et al., 2002	/			/	/	/	
Davis, 2005	/			/	/	/	
Karlson, 1997		/		/			
Gilb, 2005	/	/					
Finkelstein et al., 2009	/	/					
Beck, 1999		/			/		
Daveva, 2008	/	/		/			
Wiegers, 1999		/		/		/	/