บทที่ 2
เอกสารและงานวิจัยที่เกี่ยวข้อง

2.1 บทนำ

ในบทนี้เป็นการนำเสนอวรรณกรรมในอดีตที่เกี่ยวข้องเพื่อชี้ให้เห็นถึงการศึกษาหรือสำรวจในเรื่องเกี่ยวกับความสำคัญของการยอมรับซอฟต์แวร์ ตัวแปรที่มีผลต่อการยอมรับซอฟต์แวร์ของผู้ใช้ กระบวนการพัฒนาซอฟต์แวร์ที่เกี่ยวข้องกับความยากง่ายและประโยชน์ในการใช้งาน การประเมินการยอมรับซอฟต์แวร์ วิธีการวัดความยากง่ายและประโยชน์ของซอฟต์แวร์ และการศึกษาที่อธิบายถึงไซต์แมพในบริบทต่าง ๆ ซึ่งประกอบด้วยทฤษฎีสาระสำคัญ งานวิจัยในอดีต โดยผู้วิจัยได้รวบรวมนำมารายงานให้เห็นถึงความเป็นไป บทสรุป ข้อจำกัด รวมถึงข้อมูลจากการวิจัยต่าง ๆ ที่เป็นประโยชน์ซึ่งจะช่วยให้การศึกษาครั้งนี้คลี่คลาย

2.2 การยอมรับซอฟต์แวร์ (Adoption of software)

การศึกษาในอดีตพบสาเหตุสำคัญของความล้มเหลวในการพัฒนาซอฟต์แวร์ต่าง ๆ เนื่องมาจากขาดการยอมรับจากผู้ใช้ เช่น การศึกษาจำนวนมากพบว่าสามเหตุผลหลักที่ทำให้การนำระบบ CRM (Customer Relation Management) ไปใช้แล้วล้มเหลวประกอบด้วย (1) ขาดการยอมรับจากผู้ใช้ (2) ขาดการประสานงาน (Integration) กับระบบอื่น ๆ และ (3) ยากในการนำไปติดตั้ง (ADAPT Software Applications, Inc. (ASA), 2001) ซึ่งแสดงให้เห็นว่าซอฟต์แวร์ที่ขาดการยอมรับจากผู้ใช้เป็นสาเหตุอันดับต้น ๆ ที่ก่อให้เกิดความล้มเหลวมาสู่การพัฒนาซอฟต์แวร์ไม่เว้นแม้แต่ซอฟต์แวร์ที่นำมาใช้กันอย่างแพร่หลาย หรือในการศึกษาของ Warren (2002) กล่าวว่าการยอมรับของผู้ใช้เป็นลักษณะที่สำคัญที่สุดซึ่งสนับสนุนความสำเร็จของการพัฒนาซอฟต์แวร์ และในทางกลับกันหากขาดการยอมรับจากผู้ใช้ก็เป็นเหตุผลเดียวที่ก่อให้เกิดความล้มเหลวต่อการพัฒนาซอฟต์แวร์บ่อยที่สุด

อีกทั้ง Griffin (1999) ผู้ซึ่งพัฒนาซอฟต์แวร์ปัญญาธุรกิจ (Business Intelligence) และระบบคลังสินค้า (Warehouse) ให้กับบริษัทจำนวนมาก อ้างว่าปัจจัยที่นำมาสู่ความล้มเหลวของซอฟต์แวร์คือการที่ผู้ใช้ไม่นำซอฟต์แวร์หรือระบบที่พัฒนาขึ้นนั้นไปใช้ และผู้ใช้ยังขาดการยอมรับซอฟต์แวร์อยู่ นอกจากนี้ Griffin (1999) ยังเสนอว่าหากผู้ใช้ส่วนหนึ่งในองค์กรแสดงความไม่มั่นใจในซอฟต์แวร์ที่ใช้อยู่ จะขยายความไม่มั่นใจนี้ไปสู่ผู้ใช้ทั้งองค์กร ทำให้ผู้ใช้ทั้งหมดไม่พร้อมที่จะยอมรับซอฟต์แวร์ไปด้วย ซึ่งความก้าวหน้าของโครงการ และการได้รับการยอมรับจากผู้ใช้นี้มีความสำคัญต่อองค์กรมากทีเดียว

เมื่อระบบสารสนเทศที่ช่วยในการตัดสินใจของผู้บริหารแบบอินทราเน็ตที่เรียกว่า Executive Dashboards ซึ่งประกอบด้วยซอฟต์แวร์หลายรูปแบบ เช่น Crystal Report ใช้

สำหรับแสดงกราฟ หรือ ซอฟต์แวร์ปัญญาธุรกิจที่ใช้ในการตัดสินใจ เป็นต้น ถูกค้นพบว่าหากนำเข้ามาใช้จะมีโอกาสล้มเหลวตั้งแต่เริ่มต้นเมื่อขาดการยอมรับจากผู้ใช้ มีรายจ่ายสูง และขาดการสนับสนุนจากผู้บริหาร (Ordway, 2005)

บางงานวิจัยยังกล่าวเพิ่มเติมว่าผู้ใช้เป็นเหตุผลสำคัญในการพัฒนาซอฟต์แวร์ ซึ่งเป็นการแนะนำไปสู่วิธีพัฒนาซอฟต์แวร์ที่ให้ผู้ใช้เป็นศูนย์กลาง (User-Centred approach) เช่น การศึกษาของ Jiang และคณะ (1996) แนะนำ 13 ปัจจัยที่นำไปสู่ความสำเร็จของโครงการพัฒนาซอฟต์แวร์หรือระบบสารสนเทศ หรือเรียกว่า Jiang's 13 ซึ่งประกอบด้วย การระบุเป้าหมายที่ชัดเจน การมีหัวหน้าโครงการ (Project manager) ที่มีความสามารถ ได้รับการสนับสนุนจากผู้บริหาร สมาชิกในทีมมีความสามารถ มีการจัดสรรทรัพยากรที่ดี มีช่องทางการสื่อสารอย่างเพียงพอ มีกลไกการควบคุมที่เหมาะสม นำผลป้อนกลับ (Feedback) ที่ได้รับมาปรับปรุง แจ้งความคืบหน้าของงานที่ทำให้ผู้ใช้รับทราบ สอบถามความเห็นของผู้ใช้ นำเทคโนโลยีที่ดีมาใช้ ได้รับการยอมรับจากลูกค้า และริบหาทางแก้ไขเมื่อเกิดปัญหา ทั้งนี้ในตอนท้ายของข้อสรุปงานวิจัย Jiang และคณะ (1996) กล่าวว่าผู้ใช้ซอฟต์แวร์หรือเทคโนโลยีสารสนเทศนั้นถือว่าเป็นปัจจัยที่สำคัญอย่างมากต่อความสำเร็จในการพัฒนาซอฟต์แวร์

Jiang และคณะ (1996) แสดงให้เห็นว่าผู้พัฒนาซอฟต์แวร์ต้องเกี่ยวข้องในกระบวนการพัฒนาซอฟต์แวร์ตั้งแต่ขั้นเริ่มต้น เพื่อที่จะสามารถรับรู้ความรู้สึกของผู้ใช้งาน การทำงานกับผู้ใช้เพื่อพัฒนาโปรแกรมประยุกต์ที่ประสบความสำเร็จนั้น นอกจากได้รับข้อมูลป้อนกลับ (Feedback) ว่าผู้ใช้ชอบหน้าจอที่พัฒนาขึ้นมาหรือไม่ แล้วยังต้องสามารถเข้าใจข้อผูกมัดซึ่งเกิดขึ้นเมื่อผู้ใช้ส่วนใหญ่ทำงานกับซอฟต์แวร์อัตโนมัติ นั่นคือต้องเข้าใจแรงจูงใจเบื้องหลังการทำงานของผู้ใช้ว่าทำไมผู้ใช้จึงลังเลใจที่จะใช้ซอฟต์แวร์ หรือ การกระทำของผู้ใช้เป็นไปตามแนวทางที่องค์กรตั้งเป้าหมายไว้อย่างไร เป็นต้น

เพื่อที่จะตอบปัญหาเหล่านี้ต้องชักนำการออกแบบซอฟต์แวร์ให้เป็นไปในทางที่มีความเข้าใจลักษณะผู้ใช้มากขึ้น จะทำให้ซอฟต์แวร์ที่ได้มีประสิทธิภาพ มีความหมายและได้รับการยอมรับจากผู้ใช้ นั่นคือผู้ใช้เป็นหัวใจแห่งความสำเร็จในการพัฒนาซอฟต์แวร์ (Ross, 2005)

2.3 ตัวแปรที่มีผลต่อการยอมรับซอฟต์แวร์

โครงการพัฒนาซอฟต์แวร์จะประสบความสำเร็จ หรือไม่เกิดความล้มเหลวจนกระทั่งต้องเสียค่าใช้จ่ายตามมาอีกมากมายได้ จำเป็นต้องมีกฎสำคัญที่ผู้พัฒนาจะต้องยึดถือไว้นั่นคือพัฒนาซอฟต์แวร์ให้ได้รับการยอมรับจากลูกค้าหรือผู้ใช้เสียก่อน ดังนั้นเป็นสิ่งที่น่าศึกษาถึงปัจจัยที่ส่งผลต่อการยอมรับซอฟต์แวร์ เพื่อให้ผู้พัฒนานำปัจจัยเหล่านี้ไปวัดหรือทดสอบก่อนที่จะส่งมอบซอฟต์แวร์ให้กับผู้ใช้ต่อไป

Coskun และ Grabowski (2005) ได้ศึกษาผลกระทบของความซับซ้อนของส่วนเชื่อมต่อผู้ใช้ (User interface) ต่อการยอมรับของผู้ใช้ในระบบรักษาความปลอดภัยฉุกเฉิน

(Safety-critical system) กล่าวว่าส่วนเชื่อมต่อผู้ใช้ของระบบรักษาความปลอดภัยฉุกเฉินทวีความสำคัญมากขึ้น เมื่อผู้ใช้มีระยะเวลาจำกัดในขณะที่ต้องการการตัดสินใจที่ถูกต้อง รวดเร็ว ดังนั้นระบบจึงต้องถูกออกแบบมาให้ง่ายต่อการใช้งาน เพื่อที่จะได้รับการยอมรับจากผู้ใช้ซึ่งต้องการตัดสินใจในภาวะเร่งด่วน มิเช่นนั้นแล้วจะก่อให้เกิดความเสียหายที่รุนแรง เช่น สูญเสียทางการเงิน สิ่งแวดล้อมได้รับความเสียหาย หรือแม้แต่สูญเสียชีวิต ซึ่งระบบรักษาความปลอดภัยฉุกเฉินในการทดลองของ Coskun และ Grabowski (2005) เลือกใช้ระบบผู้เชี่ยวชาญในการนำทางบนเรือ (Navigation and Piloting Expert System; NPES) มาศึกษา สอดคล้องกับระบบจำแนกลายนิ้วมืออัตโนมัติ (National Automated Fingerprint Identification System: NAFIS) ที่รัฐบาลของสหรัฐอเมริกาพัฒนาขึ้น เป็นเครื่องมือช่วยตำรวจในการจับโจรผู้ร้ายและระบุตัวตนของตำรวจได้ โดยบังคับใช้ในอังกฤษ และเวลส์ ซึ่งโครงการพัฒนาระบบจำแนกลายนิ้วมืออัตโนมัตินี้ต้องการแสดงให้เห็นความสำคัญของปฏิสัมพันธ์ระหว่างมนุษย์และคอมพิวเตอร์ (Human-Computer Interaction) ที่มีต่อวิศวกรรมซอฟต์แวร์ โดยทดสอบความสามารถในการใช้งานในห้องทดลองก่อนการใช้จริง และวัดประสิทธิภาพของระบบจากเทปวิดีโอ รวมทั้งมีบรรทัดฐานว่าการทำให้ผู้ใช้ยอมรับระบบ ก็คือต้องทำระบบให้ง่ายต่อการใช้งาน (Followell et al., 1996)

เช่นเดียวกับ Risley (2004) ให้ข้อสรุปว่าการพัฒนาซอฟต์แวร์ที่ทำหน้าที่เกี่ยวกับการสอนวิชาฟิสิกส์ (Physics Academic Software) จะถูกยอมรับและนำไปเผยแพร่ให้ใช้งานได้ขึ้นอยู่กับคุณภาพของโปรแกรม ความชัดเจนของการแสดงหน้าจอ ความถูกต้องในเชิงตัวเลข ความถูกต้องของอัลกอริทึม ความมีประสิทธิภาพทางการศึกษาหรืองานวิจัยที่นำมารวมในซอฟต์แวร์ และมีความง่ายในการใช้งานซอฟต์แวร์

หลายครั้งพบว่าเทคโนโลยีจำนวนมากไม่ได้รับการยอมรับอันเนื่องมาจากซอฟต์แวร์ไม่มีความง่ายในการใช้งาน โดยการยอมรับ ความสามารถในการใช้งาน และความสามารถทางการตลาดของซอฟต์แวร์ มักขึ้นอยู่กับความรู้สึกของผู้ใช้ว่าง่ายต่อการใช้งาน การออกแบบส่วนเชื่อมต่อผู้ใช้จะเพิ่มระดับความมีประสิทธิภาพของซอฟต์แวร์ได้จากการที่ซอฟต์แวร์นั้นได้รับการยอมรับ (Usernomics company, 2003)

ขณะที่การพัฒนาซอฟต์แวร์มุ่งเน้นไปที่ความยากง่ายและประโยชน์ในการใช้งาน แต่ดูเหมือนว่าการมีประโยชน์จะขับเคลื่อนให้เกิดการยอมรับซอฟต์แวร์ที่ใช้มากกว่า โดยหากพิจารณาจากชีวิตประจำวันของคนในยุคปัจจุบันที่มีเทคโนโลยีให้เลือกใช้มากมาย ผู้ใช้ต้องอาศัยความพยายามที่จะใช้ซอฟต์แวร์ซึ่งประกอบด้วยเทคโนโลยีมากมายในแต่ละวัน (ซอฟต์แวร์มีความสามารถในการใช้งานต่ำ) แต่ผู้ใช้ก็มักจะใช้อย่างต่อเนื่อง เพราะผู้ใช้มีความเห็นว่าซอฟต์แวร์เหล่านั้นมีประโยชน์ต่อการใช้งานสูง (Ross, 2005) โดยมีงานวิจัยอีกจำนวนหนึ่งที่อ้างว่าการยอมรับซอฟต์แวร์ขึ้นอยู่กับความมีประโยชน์ของซอฟต์แวร์ ดังนี้

การศึกษาปัจจัยที่กระทบต่อการยอมรับเครื่องมือสนับสนุนการสอน โดยใช้คอมพิวเตอร์ของผู้เรียน (Computer-based training support tool) ตามสมมติฐานของ Wagner และคณะ (2004) ประกอบด้วย ความตั้งใจทางพฤติกรรม (Behavioural intensions) ความกังวลใจในคอมพิวเตอร์ (Computer anxiety) ความกังวลใจในซอฟต์แวร์ (Software anxiety) ทัศนคติในการใช้งาน (Attitudes) ลักษณะส่วนบุคคลหรือลักษณะขององค์กร (Individual and organizational characteristics) การควบคุมความรู้สึกส่วนบุคคลเมื่อได้รับการสอน (Self-determination) ความสามารถส่วนบุคคลในการจัดการหรือควบคุมแผนการสอน (Self-management) และการรับรู้ถึงประโยชน์ (Perceived usefulness) โดยผลการศึกษาพบว่าปัจจัยทั้งหมดมีผลกระทบต่อการยอมรับซอฟต์แวร์ดังกล่าว ซึ่งหนึ่งในปัจจัยเหล่านั้นก็คือการมีประโยชน์ อีกทั้ง Jiang และคณะ (1996) แนะนำว่าประโยชน์ของการพัฒนาซอฟต์แวร์มักเกี่ยวข้องกับการยอมรับจากผู้ใช้ และซอฟต์แวร์ที่มีประโยชน์นั้นจะเป็นจุดขายที่สำคัญต่อลูกค้า

Vassar และคณะ (1999) กล่าวว่าการมีประโยชน์จะมีอิทธิพลต่อการยอมรับของผู้ใช้ที่มีต่อซอฟต์แวร์หรือโปรแกรมประยุกต์ ซึ่งประโยชน์นี้สามารถแบ่งการพิจารณาออกเป็นสี่ด้าน คือ ลักษณะของซอฟต์แวร์ ผู้ใช้ กระบวนการทำงาน และสิ่งแวดล้อม

หากจะมองการยอมรับซอฟต์แวร์ในแง่ที่เป็นคุณภาพของซอฟต์แวร์ (Software Quality) แล้ว การทดสอบการยอมรับ (Acceptance testing) ก็ถือว่าเป็นการประเมินคุณภาพของซอฟต์แวร์ที่สำคัญ และหน่วยที่นำมาทดสอบการยอมรับซอฟต์แวร์ก็คือประโยชน์ (Kan, 2002)

จากการศึกษาที่ได้รายงานมาข้างต้น ทำให้ผู้วิจัยทราบถึงปัจจัยหลักที่ส่งผลต่อการยอมรับซอฟต์แวร์ในบริบทต่าง ๆ สรุปได้เป็นสองปัจจัยคือ (1) ความยากง่าย และ (2) ประโยชน์ในการใช้งาน ซึ่งจะเป็นแนวทางไปสู่การศึกษาวิธีการวัดปัจจัยดังกล่าวในหัวข้อ 2.6 ต่อไป

2.4 กระบวนการพัฒนาซอฟต์แวร์ที่เกี่ยวข้องกับความยากง่ายและประโยชน์ในการใช้งาน

ในอดีตผู้พัฒนาซอฟต์แวร์มักถูกพูดถึงจากบุคคลที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ (Stakeholders) โดยเฉพาะผู้ใช้ ในทางที่ไม่ค่อยดีนักหลังจากพัฒนาซอฟต์แวร์เสร็จสิ้นและถูกติดตั้งให้ใช้งานแล้ว เนื่องจากผู้ใช้และผู้สนับสนุนโครงการ (Project sponsor) นั้นมองการทำงานของซอฟต์แวร์กันคนละมุม รวมทั้งผู้ที่มีอิทธิพลในการกำหนดทิศทางของซอฟต์แวร์ที่พัฒนาขึ้นนั้นมิใช่ผู้ใช้ แต่เป็นผู้สนับสนุนโครงการ ต่อมาวิธีการเช่นนี้ได้เปลี่ยนแปลงไป โดยผู้ใช้เริ่มเข้ามาเป็นบุคคลที่เกี่ยวข้องหลักของโครงการพัฒนาซอฟต์แวร์มากขึ้น โดยมีเสียงในการออกความเห็นเท่ากับบุคคลอื่น ๆ ที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ เริ่มตั้งแต่ขั้นตอนตั้งวัตถุประสงค์และเป้าหมายในการพัฒนาซอฟต์แวร์ไปจนกระทั่งติดตั้งและสอนการใช้งาน

ซอฟต์แวร์ มีการพูดถึงในวงกว้างเกี่ยวกับการพัฒนาซอฟต์แวร์ในลักษณะเช่นนี้ว่าเป็นกระบวนการพัฒนาโดยให้ผู้ใช้เป็นศูนย์กลาง (User-centric software engineering) ในการพัฒนาซอฟต์แวร์ ซึ่งริเริ่มมาจากเทคโนโลยีสารสนเทศของสแกนดิเนเวีย (Scandinavian IT) เป็นปรัชญาการพัฒนาที่มุ่งเน้นเป้าหมายของธุรกิจ และจดจำความต้องการ รวมทั้งข้อจำกัดของผู้ใช้เป็นสำคัญ (Ross, 2005)

การออกแบบโดยให้ผู้ใช้เป็นศูนย์กลาง (User-centred design) เป็นกระบวนการพัฒนาซอฟต์แวร์ที่มักถูกให้คำนิยามโดยอาศัยกฎสามข้อ (Gould and Lewis, 1985) กฎข้อแรกมุ่งเน้นไปที่ผู้ใช้และงานที่ผู้ใช้ต้องทำ ซึ่งกระบวนการจะถูกจัดโครงสร้างการรวบรวมข้อมูลจากผู้ใช้เกี่ยวกับงานที่ผู้ใช้ทำ โดยให้ผู้ใช้เข้าร่วมในการพัฒนาตั้งแต่ขั้นแรก ๆ ของวงจรชีวิตการพัฒนา (Development lifecycle) และต่อเนื่องไปจนการพัฒนาเสร็จสิ้นลง กฎข้อที่สอง เป็นกฎข้อที่สำคัญที่สุดนั่นคือวัดการใช้งานของซอฟต์แวร์ มักจะวัดความยากง่ายและประโยชน์ในการใช้งานตั้งแต่ขั้นเริ่มคิดซอฟต์แวร์ (Concept software) ทำต้นแบบ (Prototype) และวัดต่อเนื่องไปถึงขั้นการพัฒนา กฎข้อสามเป็นกฎพื้นฐานของการออกแบบโดยให้ผู้ใช้เป็นศูนย์กลางคือการออกแบบแบบวนกลับ (Iterative design) ซึ่งกระบวนการพัฒนาจะเป็นวงจรของการออกแบบ (Design) ทดสอบ (Test) ออกแบบซ้ำ (Redesign) และทดสอบซ้ำ (Retest) ทำเช่นนี้เรื่อยไปจนกระทั่งบรรลุวัตถุประสงค์ด้านความสามารถในการใช้งาน และพร้อมที่จะใช้จริง (Alexander, 2003)

กระบวนการออกแบบโดยให้ผู้ใช้เป็นศูนย์กลางมีผลกระทบต่อความสำเร็จของโครงการพัฒนาซอฟต์แวร์เป็นอย่างมาก ความสำเร็จของระบบใด ๆ วัดจากการมีผู้ใช้ซอฟต์แวร์ที่พัฒนาขึ้นหรือไม่นั่นเอง เป้าหมายของธุรกิจอย่างเช่น การมีประสิทธิภาพหรือการเพิ่มผลผลิตจะไม่ปรากฏขึ้นหากว่าผู้ใช้ปฏิเสธการทำงานกับซอฟต์แวร์นั้น ถึงแม้บริษัทจะบังคับใช้ซอฟต์แวร์ที่ติดตั้ง แต่ผู้ใช้ก็อาจจะยังไม่รับมาใช้ (Adoption) ในเวลาอันรวดเร็วหากซอฟต์แวร์นั้นไม่ตรงกับความต้องการของผู้ใช้ (Ross, 2005)

ตัวอย่างกรณีศึกษาเว็บไซต์ของ Monash University (ภาคผนวก ก) ที่นำกระบวนการออกแบบโดยให้ผู้ใช้เป็นศูนย์กลางมาใช้เพื่อแก้ปัญหาที่เกิดขึ้นจากการขาดการยอมรับจากผู้ใช้ โดยวัดการยอมรับของผู้ใช้จากการวัดความยากง่าย และประโยชน์ในการใช้งานของเว็บไซต์

2.5 การประเมินการยอมรับของซอฟต์แวร์

การรับซอฟต์แวร์มาใช้นั้นส่วนใหญ่จะขึ้นอยู่กับการรับรู้ของผู้ใช้ จึงมีผู้คิดค้นแนวคิดเพื่อประเมินการยอมรับซอฟต์แวร์ นั่นคือทฤษฎีตัวแบบการยอมรับทางเทคโนโลยี (Technology Acceptance Model หรือ TAM) หัวใจสำคัญของตัวแบบนี้คือการรับรู้ถึงประโยชน์และความยากง่ายในการใช้งานของซอฟต์แวร์ นั่นคือสามารถบอกได้ว่า ซอฟต์แวร์สามารถแก้ปัญหาของผู้ใช้ได้หรือไม่ และซอฟต์แวร์ง่ายต่อการปฏิบัติงานของผู้ใช้หรือไม่

ทฤษฎีตัวแบบการยอมรับทางเทคโนโลยี เป็นทฤษฎีซึ่งอธิบายว่าผู้ใช้จะยอมรับและใช้เทคโนโลยีได้อย่างไร โดยตัวแบบนี้อ้างว่าเมื่อผู้ใช้ได้รับการแนะนำซอฟต์แวร์ใหม่ ๆ แล้ว ปัจจัยที่มีอิทธิพลต่อการตัดสินใจว่าเมื่อไรที่ผู้ใช้จะใช้ซอฟต์แวร์ และยอมใช้ซอฟต์แวร์นั้นได้อย่างไร มีสองปัจจัยคือ (1) รับรู้ถึงประโยชน์ของซอฟต์แวร์ เป็นการคาดหวังทางจิตใจของผู้ใช้ว่าการใช้โปรแกรมประยุกต์ (Application) ที่เฉพาะเจาะจงจะเพิ่มประสิทธิภาพ (Performance) ของการทำงาน (2) รับรู้ถึงความยากง่ายในการใช้งาน อ้างถึงระดับความคาดหวังของผู้ใช้ว่าจะไม่ต้องอาศัยความพยายามในการใช้ซอฟต์แวร์

ทฤษฎีตัวแบบการยอมรับทางเทคโนโลยีเป็นหนึ่งในส่วนขยายของทฤษฎีการกระทำที่ถูกให้เหตุผล (Theory of Reasoned Action; TRA) พัฒนาขึ้นโดย Davis et al. (1998) และ Bagozzi et al. (1992) ทั้งทฤษฎีการกระทำที่ถูกให้เหตุผล และทฤษฎีตัวแบบการยอมรับทางเทคโนโลยีเป็นทฤษฎีทางพฤติกรรมซึ่งคาดการณ์ล่วงหน้าว่าเมื่อบุคคลตั้งใจกระทำบางสิ่ง ย่อมต้องการความอิสระที่จะกระทำโดยปราศจากข้อจำกัดใด ๆ ซึ่งในโลกแห่งความจริงนั้นมีข้อจำกัดมากมาย ไม่ว่าจะเป็นข้อจำกัดในความสามารถ ข้อจำกัดทางด้านเวลา ข้อจำกัดทางสิ่งแวดล้อม หรือแม้แต่ข้อจำกัดทางด้านความเคยชินของมนุษย์ก็สามารถจำกัดความอิสระต่อการกระทำใด ๆ

เนื่องจากซอฟต์แวร์ที่พัฒนาขึ้นใหม่มีความซับซ้อน สร้างความไม่แน่นอนว่าควรรับมาใช้หรือไม่ขึ้นในจิตใจของผู้ใช้ สิ่งแรกผู้ใช้จะเริ่มสร้างทัศนคติและความตั้งใจในการเรียนรู้ที่จะใช้ซอฟต์แวร์ใหม่นั้นก่อนที่จะอาศัยความพยายามในการใช้จริง โดยทัศนคติและความตั้งใจนี้หากขาดความเชื่อหรือต้องอาศัยความพยายามในการเรียนรู้ที่จะทำในครั้งแรกแล้ว การใช้จริงอาจไม่เกิดขึ้นตามทัศนคติหรือความตั้งใจตั้งแต่แรก เป็นส่วนที่สนับสนุนทฤษฎีที่กล่าวว่า การที่ผู้ใช้จะยอมรับซอฟต์แวร์ใหม่ ๆ ได้นั้นอาศัยความตั้งใจ (Intention) ที่จะใช้เป็นเครื่องช่วยตัดสิน ซึ่งความตั้งใจนี้ได้รับการสนับสนุนจากทัศนคติ (Attitude) ที่มีต่อซอฟต์แวร์นั้น โดยทัศนคติวัดจากตัวแปรสองรูปแบบคือ ความรับรู้ถึงความยากง่ายและความรับรู้ถึงประโยชน์ในการใช้งาน

ดังนั้นการรับรู้ถึงประโยชน์ การรับรู้ถึงความยากง่ายในการใช้งาน และความสามารถในการทำงานที่ตรงตามความต้องการของผู้ใช้ ร่วมกันสร้างทัศนคติเกี่ยวกับซอฟต์แวร์ในจิตใจของผู้ใช้ ซึ่งทัศนคตินี้ส่งผลไปสู่การตัดสินใจว่าผู้ใช้จะนำซอฟต์แวร์ไปใช้หรือไม่ การใช้ซอฟต์แวร์อย่างต่อเนื่องจะช่วยขยายให้ความรู้สึกของผู้ใช้เกี่ยวกับประโยชน์และความยากง่ายในการใช้งานกับการเพิ่มปฏิสัมพันธ์ด้านบวกและลบต่อการวัดการยอมรับให้กว้างขึ้น

2.6 วิธีการวัดความยากง่าย และประโยชน์ในการใช้งาน

จากหัวข้อ 2.3 ได้แสดงตัวแปรที่ส่งผลต่อการยอมรับของซอฟต์แวร์มาแล้ว ต่อไปนี้เป็นการนำเสนอวิธีการวัดปัจจัยเหล่านั้น ประกอบด้วยปัจจัยความยากง่ายในการใช้งาน และประโยชน์ ซึ่งจากการสำรวจงานวิจัยที่ผ่านมาสามารถแบ่งวิธีการวัดออกเป็นสองรูปแบบ คือ

2.6.1 วิธีการวัดด้วยแบบสอบถามที่ใช้มาตราการประเมิน

ในส่วนของวิธีการวัดความรับรู้ (Perception) ตาม TAM นั้น Davis (1989) ได้ประยุกต์แบบสอบถามที่ใช้มาตราการประเมิน (Rating scale) หรือลิเคิร์ทสเกลขึ้น เพื่อให้สามารถเข้าใจความรู้สึกที่ผู้ใช้มีต่อซอฟต์แวร์ได้ วิธีการวัดดังกล่าวถูกนำไปใช้อย่างแพร่หลายโดยนักวิจัยจำนวนมาก ซึ่งนำแบบสอบถามที่ใช้มาตราการประเมินนี้ไปใช้วัดความรับรู้ของผู้ใช้ต่อซอฟต์แวร์อื่น ๆ จนกลายเป็นงานวิจัยเรียกว่า ทฤษฎีตัวแบบการยอมรับทางเทคโนโลยีที่ขยายขึ้น (Extended TAM หรือ TAM2) (Venkatesh and Davis, 2000) ดังเช่น วัดการยอมรับในเครื่องมือการรวบรวมข้อมูลที่ใช้เว็บ (Web-based Inspection Data collection Tool) ของ Dryer และ Laitenberger (1998) หรือวัดการยอมรับในเน็ตสเคป (Netscape) ของ Dillon และ Morris (1997) เป็นต้น ทำให้ความถูกต้องและความน่าเชื่อถือของสเกลถูกการันตีได้จากงานวิจัยที่นำทฤษฎีตัวแบบการยอมรับทางเทคโนโลยีไปพัฒนาจนได้ผลวิจัยชิ้นต่อ ๆ มาจนกระทั่งถึงปัจจุบัน

เมื่อสำรวจประโยชน์ของการเข้าถึงวารสารส่วนบุคคลที่เป็นอิเล็กทรอนิกส์ (E-journal) เว็บไซต์ที่นำเสนอวารสารที่มีชื่อเสียงอย่างไซเอ็นซ์ไดเร็กซ์ (Sciencedirect) นั้นได้วิเคราะห์และระบุถึงประโยชน์ด้วยวิธีการให้หน่วยตัวอย่างเข้าเว็บไซต์แล้วจึงสอบถามความคิดเห็นโดยใช้แบบสอบถามที่มีข้อถามเป็นมาตราการประเมิน (Hyldegaard and Seiden, 2004)

อีกทั้ง Lund (2001) พบว่าการนำความต้องการ (Requirement) ของลูกค้ามาออกแบบส่วนเชื่อมต่อผู้ใช้ (User interface) นั้น มักพบมีปัญหามากมายเกิดขึ้นไม่ว่าจะเป็น สามารถตัดสินระดับการยอมรับข้อผิดพลาดหรือความเร็วที่เกิดขึ้นกับงานที่ทำได้อย่างไร จะทราบได้อย่างไรว่าการปรับปรุงส่วนเชื่อมต่อผู้ใช้นั้นเพียงพอและไม่ต้องแก้ไขอีกแล้ว หรือแม้แต่จะทราบได้อย่างไรว่าการปรับปรุงส่วนติดต่อผู้ใช้มีผลกระทบกับการขายหรือไม่ ปัญหาเหล่านี้ต้องการหาคำตอบและมิติสำคัญในการตอบปัญหาคือต้องทราบความรู้สึกของผู้ใช้ที่มีต่อความยากง่ายและประโยชน์ในการใช้งานซอฟต์แวร์ที่พัฒนาขึ้น โดยอาศัยเครื่องมือที่เป็นแบบสอบถามที่มีข้อถามเป็นมาตราการประเมิน เนื่องจากผู้ใช้สามารถแยกแยะได้เองว่าแบบใดจึงง่าย หรือมีประโยชน์ในการใช้งาน และแบบใดที่ไม่มีประโยชน์ โดยวิธีการวัดที่ใช้แบบสอบถามนี้สามารถนำไปประยุกต์ใช้ในโครงการพัฒนาซอฟต์แวร์ ฮาร์ดแวร์ บริการ และเครื่องมือสนับสนุนผู้ใช้อื่น ๆ ได้เป็นอย่างดี

นอกจากนี้ Nielsen (2003) ถือว่าความยากง่ายในการใช้งานเป็นมิติหนึ่งของการวัดความสามารถในการใช้งาน (Usability) ของเว็บไซต์ โดยให้ความหมายว่าเป็นความยากง่ายที่ผู้ใช้สามารถหาข้อมูลที่เฉพาะเจาะจงในเว็บไซต์พบ ตัดสินจากการตอบแบบสอบถามที่ประกอบด้วยคำถามรูปแบบจัดอันดับคุณภาพ 10 ระดับของลิเคิร์ท (Likert's rating scale) ซึ่งเป็นการถามความเห็นผู้ใช้ว่าเว็บไซต์ทำงานได้ดีเพียงใด ไม่ใช่เป็นการประเมินตัวผู้ใช้ว่าทำงาน

ได้ดีเพียงใด นั่นคือ ถึงแม้ผู้ใช้จะเสียเวลากับการค้นหาข้อมูลในเว็บไซต์แต่ผู้ใช้อาจจะคิดว่าเว็บไซต์ยังคงง่ายต่อการใช้งานได้

ยังมีงานวิจัยอีกจำนวนมากที่วัดความยากง่าย และประโยชน์ในการใช้งานจากการใช้แบบสอบถามที่ประกอบด้วยมาตราการประเมิน ดังที่รวบรวมไว้ในตารางที่ 2.1

ตารางที่ 2.1 แสดงผลสรุปที่ได้จากการศึกษางานวิจัยในอดีตซึ่งเกี่ยวข้องกับวิธีการวัดความยากง่ายและประโยชน์ในการใช้งานโดยใช้แบบสอบถามที่ใช้มาตราการประเมิน

การศึกษา	ตัวแปรที่ใช้	ซอฟต์แวร์
Benslimane, Bernard และ Plaisent (2004)	ความยากง่าย และประโยชน์ในการใช้งาน	Web systems for E-procurement
Brown (2002)	ความยากง่าย และประโยชน์ในการใช้งาน	Web-based Learning Technologies
Ciba และ Rakestraw (1998)	ประโยชน์ในการใช้งาน	Internet-based Tools in Traditional Teaching Situations
Corbitt และ Sandhu (2003)	ความยากง่าย และประโยชน์ในการใช้งาน	Web-based Electronic Services Adoption Model (E-SAM)
Dillon และ Morris (1997)	ความยากง่าย และประโยชน์ในการใช้งาน	Netscape
Dryer และ Laitenberger (1998)	ความยากง่าย และประโยชน์ในการใช้งาน	Web-based Inspection Data collection Tool
Gardner และ Amoroso (2004)	ความยากง่าย และประโยชน์ในการใช้งาน	Internet technology
Hildreth (2001)	ประโยชน์ในการใช้งาน	Information Retrieval (IR) system
Chorng-Shyong Onga และคณะ (2003)	ความยากง่าย และประโยชน์ในการใช้งาน	E-learning systems
Liu และคณะ (2003)	ความยากง่าย และประโยชน์ในการใช้งาน	Online E-learning
Lopez และ Manson (1997)	ประโยชน์ในการใช้งาน	Desktop Information System
Mohd และ Mohamad (2005)	ความยากง่าย และประโยชน์ในการใช้งาน	Electronic Medical Record system (EMR)
Nelson (1991)	ประโยชน์ในการใช้งาน	General Information system

ถึงแม้จะมีการศึกษามากมายข้างต้นที่ยืนยันว่าการวัดความยากง่าย และประโยชน์ในการใช้งานด้วยแบบสอบถามที่ใช้มาตราการประเมินนั้นเพียงพอและสามารถยืนยันความน่าเชื่อถือในเรื่องการยอมรับซอฟต์แวร์ของผู้ใช้ได้ แต่ก็ยังพบงานวิจัยที่กล่าวว่าการวัดด้วยวิธีการดังกล่าวนั้นมีความน่าเชื่อถือไม่เพียงพอ (Dillon, 2005) ดังนั้นผู้วิจัยจึงศึกษาวิธีการวัดอื่นควบคู่ด้วย ดังในหัวข้อ 2.6.2

2.6.2 วิธีการวัดโดยอาศัยเครื่องมือวัดเชิงพฤติกรรม

เมื่อมีข้อโต้แย้งเกี่ยวกับวิธีการวัดด้วยแบบสอบถามที่ใช้มาตราการประเมินเพียงอย่างเดียวว่าไม่เพียงพอต่อการสรุปการยอมรับซอฟต์แวร์ ดังนั้นผู้วิจัยจึงนำวิธีการวัดจากพฤติกรรมการใช้งานจริงของผู้ใช้มาศึกษา แต่เนื่องจากความยากง่าย และประโยชน์ในการใช้งาน อาศัยเครื่องมือวัดเชิงพฤติกรรมที่แตกต่างกัน จึงขอแยกเป็นสองหัวข้อ ดังนี้

2.6.2.1 วิธีการวัดความยากง่ายในการใช้งานโดยอาศัยเครื่องมือวัดเชิงพฤติกรรม มีหลายงานวิจัยที่ศึกษาถึงความยากง่ายในการใช้งานที่มีต่อซอฟต์แวร์ต่าง ๆ จากพฤติกรรมของผู้ใช้ โดยผลสรุปของการศึกษาวรรณกรรมในอดีต (Review literature) ที่พิจารณาถึงปัจจัยในการวัดความยากง่ายในการใช้งานของผู้ใช้ แสดงดังตารางที่ 2.2

ตารางที่ 2.2 แสดงผลสรุปที่ได้จากการศึกษางานวิจัยในอดีตซึ่งเกี่ยวข้องกับวิธีการวัดความยากง่ายในการใช้งานซอฟต์แวร์ต่าง ๆ จากพฤติกรรมการใช้งานจริง

การศึกษา	ตัวแปรที่ใช้วัดความยากง่ายในการใช้งาน	ซอฟต์แวร์
Bernard และ Hull (2002)	- เวลาที่ใช้เพื่อทำงานที่กำหนดให้	Embedded and Framed/Non-Framed Links for Website
Carbone (2006)	- เวลาที่ใช้ในการทำงานที่กำหนดให้เสร็จสิ้น - ข้อผิดพลาดที่พบ	Real-time operating system (RTOS)
Corley, Miller และ Rainer (2003)	- เวลาที่ใช้ในการทำงานกับโมดูลที่ติดต่อกับผู้ใช้	Online learning
Good (1982)	- เวลาที่ผู้เพิ่งฝึกหัดใช้ในการพิมพ์และแก้ไขงานที่ทำกับระบบ - เวลาที่ใช้ในการเรียนรู้การใช้ระบบ	Integrated document processing system

ตารางที่ 2.2 (ต่อ) แสดงผลสรุปที่ได้จากการศึกษางานวิจัยในอดีตซึ่งเกี่ยวข้องกับวิธีการวัดความยากง่ายในการใช้งานซอฟต์แวร์ต่าง ๆ จากพฤติกรรมการใช้งานจริง

การศึกษา	ตัวแปรที่ใช้วัดความยากง่ายในการใช้งาน	ซอฟต์แวร์
Hix และ Hartson (1993); Shneiderman (1998)	- ความเร็วในการทำงาน - ความสามารถในการเรียนรู้ - อัตราข้อผิดพลาดที่พบ	General software
Miller (1971)	- จำนวนข้อผิดพลาดที่พบ - ระยะเวลาที่ใช้ในการเรียนรู้	General software
Sauro (2004)	- เวลาที่ใช้ในการทำงานที่กำหนดให้เสร็จสิ้น - ข้อผิดพลาด (Error) ที่พบ	General software
Xie (2003)	- เวลาที่ใช้สำหรับการได้รับข้อมูลจากระบบ	Web-based online information retrieval (IR) system

เนื่องจากการศึกษาในอดีตที่เกี่ยวข้องกับการวัดความยากง่ายในการใช้งานซอฟต์แวร์ต่าง ๆ ส่วนมากเป็นไปในแนวทางเดียวกัน นั่นคือวัดความยากง่ายในการใช้งานซอฟต์แวร์จากเวลาที่ใช้ และจำนวนข้อผิดพลาด (Error) ที่พบจากการทำงานที่ได้รับมอบหมาย จึงเป็นแนวทางสำหรับงานวิจัยนี้ให้นำปัจจัยทั้งสองมาใช้

2.6.2.2 วิธีการวัดประโยชน์ในการใช้งานโดยอาศัยเครื่องมือวัดเชิงพฤติกรรม พบจากการศึกษางานวิจัยในอดีตว่ามีนักวิจัยจำนวนมากที่วัดประโยชน์ในการใช้งานซอฟต์แวร์ต่าง ๆ จากพฤติกรรมของผู้ใช้ โดยผลสรุปของการศึกษาดังกล่าวมีปัจจัยที่ใช้วัดประโยชน์ แสดงดังตารางที่ 2.3

ตารางที่ 2.3 แสดงผลสรุปที่ได้จากการศึกษางานวิจัยในอดีตซึ่งเกี่ยวข้องกับวิธีการวัดประโยชน์ของซอฟต์แวร์ต่าง ๆ จากพฤติกรรมการใช้งานจริง

การศึกษา	ตัวแปรที่ใช้วัดประโยชน์	ซอฟต์แวร์
Bernard (1999)	- จำนวนโหนดทั้งหมดที่ผู้ใช้ใช้ในการค้นหาคำตอบของงานที่ได้รับลบด้วยจำนวนโหนดที่ใช้ในการทำงานเสร็จสิ้นอย่างถูกต้อง	Sitemaps for Website

ตารางที่ 2.3 (ต่อ) แสดงผลสรุปที่ได้จากการศึกษางานวิจัยในอดีตซึ่งเกี่ยวข้องกับวิธีการวัดประโยชน์ของซอฟต์แวร์ต่าง ๆ จากพฤติกรรมการใช้งานจริง

การศึกษา	ตัวแปรที่ใช้วัดประโยชน์	ซอฟต์แวร์
Bieman (2000)	- จำนวนครั้งในการนำกลับมาใช้ใหม่ของโมดูลเชิงวัตถุ (Object oriented module)	Object oriented system
Bronson (2005)	- ความถูกต้องของสารปนเปื้อนที่วัดได้จากระบบ	Gamma measurement systems
Bucksch (2004)	- ความถูกต้องของการรายงานจุดบกพร่อง (Bug) ที่ได้รับจากการใช้ซอฟต์แวร์ - ความรุนแรงของจุดบกพร่อง	Bug report software
Chrisman (1990)	- ความถูกต้องของข้อมูลที่ได้จากการใช้ระบบ	Geographic Information System (GIS)
Edwards (1995)	- เนื้อหาและเทคนิคการเรียนรู้ในสื่ออิเล็กทรอนิกส์ถูกต้อง	Courseware
Eysenbach (2000)	- ความถูกต้องของข้อมูลที่ค้นหา - จำนวนครั้งที่ใช้ในการค้นหาการศึกษาที่ถูกตีพิมพ์ในอินเตอร์เน็ต - เวลาที่ใช้ในการค้นหางานที่ได้รับ - จำนวนการศึกษาที่พบในอินเตอร์เน็ต	Internet search engine for Website
Gow (2003)	- ความถูกต้องของผลลัพธ์	Translation memory software
Lazar และ Eisenbrey (2000)	- เวลาที่หน่วยตัวอย่างใช้ในการทำงานแต่ละอย่าง - ความเป็นไปได้ที่หน่วยตัวอย่างยอมแพ้ต่องานที่ได้รับ - ความเร็วในการเชื่อมต่ออินเตอร์เน็ต	Structural navigation for Website
Oh (1997)	- ความถูกต้องของเนื้อหา - ความทันสมัยของเนื้อหา	Personal Digital Assistant (PDA) Software

ตารางที่ 2.3 (ต่อ) ตารางแสดงผลสรุปที่ได้จากการศึกษางานวิจัยในอดีตเกี่ยวข้องกับวิธีการวัดประโยชน์ของซอฟต์แวร์ต่าง ๆ จากพฤติกรรมการใช้งานจริง

การศึกษา	ตัวแปรที่ใช้วัดประโยชน์	ซอฟต์แวร์
Olson (2002)	- ความถูกต้องของข้อมูลที่ได้จากการใช้โปรแกรมประยุกต์	Data mining application
Wells (2000)	- จำนวนรายละเอียดที่ได้รับจากเครื่องมือสำรวจคุณภาพของซอฟต์แวร์	Software quality surveyor tool

จากการศึกษาในตารางที่ 2.3 พบว่าซอฟต์แวร์หรือเครื่องมือนำทางบางรูปแบบวัดประโยชน์จากปัจจัยที่หลากหลายและแตกต่างกันออกไป นั่นขึ้นอยู่กับวัตถุประสงค์ของผู้วิจัยว่าต้องการศึกษาประโยชน์ในแง่มุมใด และต้องการคำตอบของงานวิจัยในทิศทางใด เช่น การศึกษาในเรื่องการนำกลับไปใช้ใหม่ของโมดูล (Module) ในระบบเชิงวัตถุ (Object oriented system) (Bieman, 2000) เมื่อต้องการวัดประโยชน์จึงใช้จำนวนครั้งของการนำกลับไปใช้ใหม่ (Amount of reuse) มาเป็นปัจจัยในการวัดเพื่อความเหมาะสมกับวัตถุประสงค์ของงานวิจัยดังกล่าว เป็นต้น อย่างไรก็ตามงานวิจัยส่วนใหญ่ได้นำปัจจัยเรื่องความถูกต้องจากพฤติกรรมการใช้งานจริงมาวัดประโยชน์ของซอฟต์แวร์ต่าง ๆ สอดคล้องกับคำนิยามของ Davis (1989) ที่กล่าวว่าประโยชน์ คือ ระดับขั้นของความเชื่อส่วนบุคคลที่มีต่อการใช้ซอฟต์แวร์ว่าจะเพิ่มประสิทธิภาพ (Performance) ให้กับงานที่ทำ โดย ประสิทธิภาพ (Performance) ในที่นี้หมายถึงผลลัพธ์ของกิจกรรม หรือลักษณะที่ได้จากกระบวนการทำงาน ซึ่งประโยชน์จะเป็นการประเมินบรรทัดฐานที่เป็นเป้าหมายของผู้ใช้เชื่อมโยงกับคุณสมบัติของผลผลิต เช่น ผู้ใช้จะมีบรรทัดฐานในใจ คาดหวังว่าทำงานกับซอฟต์แวร์แล้วจะช่วยให้งานถูกต้องมากขึ้น จากนั้นจึงดูผลลัพธ์ที่ได้หลังการใช้จริงว่าเป็นไปตามเป้าหมาย หรือบรรทัดฐานที่ตั้งไว้หรือไม่ เป็นต้น โดยในงานวิจัยนี้คำว่า ประสิทธิภาพ ที่จะวัดหมายถึงความถูกต้องของงานที่ได้รับเมื่อผู้ใช้ทำงานโดยใช้ซอฟต์แวร์ที่พัฒนาขึ้น

การศึกษาวรรณกรรมในอดีตที่เกี่ยวกับวิธีการวัดความยากง่าย และประโยชน์ในการใช้งานนั้น นำมาใช้กันทั้งในการศึกษา งานวิจัย รวมทั้งบริษัทมีชื่อเสียงหลายบริษัทอย่างไม่มีกฎเกณฑ์ กล่าวคือบางงานวิจัยใช้เครื่องมือวัดพฤติกรรมการใช้งาน บางงานวิจัยใช้แบบสอบถามที่มีมาตราการประเมิน หรือบางงานวิจัยใช้วิธีการวัดทั้งสองแบบควบคู่กัน ทั้งที่การใช้เครื่องมือวัดเชิงพฤติกรรมทำให้ผู้พัฒนาต้องสิ้นเปลืองค่าใช้จ่าย และเสียเวลาในแง่การเก็บข้อมูลการทำงานจริงของผู้ใช้หรือพัฒนาเครื่องมือวัดอย่างมากมาย แต่ยังไม่มีงานวิจัยใดตระหนักถึงความสำคัญในเรื่องนี้ ซึ่งหากผลของงานวิจัยสรุปได้ว่าความสัมพันธ์ของวิธีการวัดทั้งสองแบบเป็นไปในทิศทางเดียวกัน จะให้คุณประโยชน์ต่อโครงการพัฒนาซอฟต์แวร์อย่างมาก เนื่องจากไม่ต้องพัฒนาเครื่องมือเพื่อใช้วัดพฤติกรรมการใช้งานของผู้ใช้สำหรับทดสอบการ

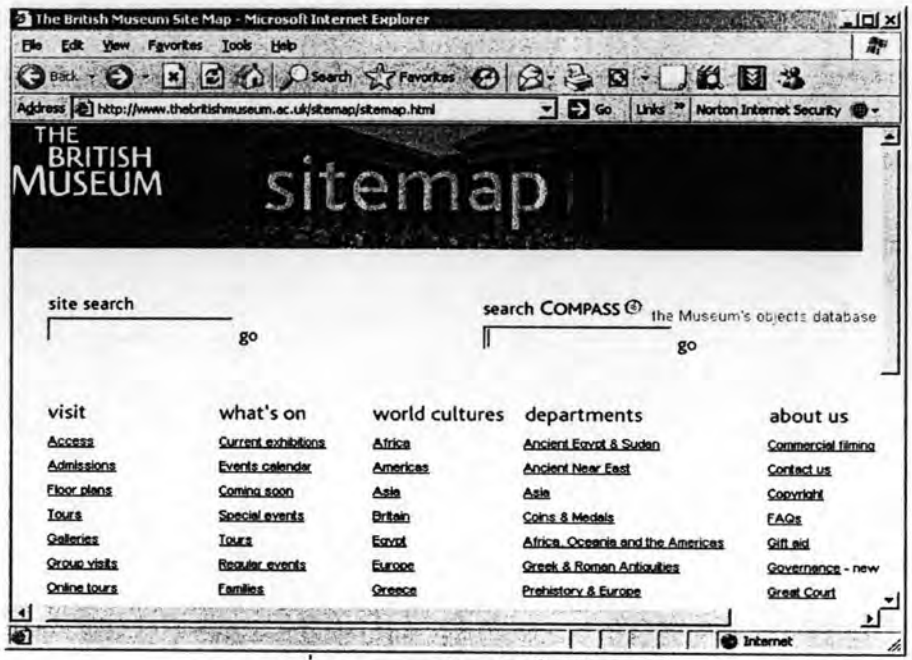
ยอมรับซอฟต์แวร์ อีกทั้งย่นระยะเวลาของกระบวนการพัฒนาซอฟต์แวร์ให้ถึงมือลูกค้าอย่างรวดเร็วยิ่งขึ้น แต่หากสรุปว่าไม่มีความสัมพันธ์ระหว่างวิธีการวัดทั้งสองแบบ จะทำให้ผู้ที่ต้องการวัดความยากง่ายหรือวัดประโยชน์ในการใช้งานมีข้อมูลสำหรับเลือกวิธีการวัดเพื่อให้ได้ข้อมูลที่ถูกต้อง น่าเชื่อถือยิ่งขึ้น ดังนั้นงานวิจัยชิ้นนี้จึงนำวิธีการวัดทั้งสองแบบมาวิเคราะห์เพื่อให้เห็นความสัมพันธ์ระหว่างกัน

2.7 การศึกษาที่อธิบายถึงไซต์แมพในบริบทต่าง ๆ

เนื่องจากไซต์แมพจะประกอบด้วยลิงค์ (link) แสดงหัวข้อเนื้อหาต่างๆ จึงอธิบายความหมายที่เกี่ยวข้องกับลิงค์ดังนี้

ลิงค์ (link) หมายถึง ข้อความ หรือรูปภาพที่วางไว้ตามส่วนต่างๆ ในหน้าเว็บเพื่อเชื่อมโยงไปยังเนื้อหาอ้างอิงส่วนอื่นที่อยู่ในหน้าเอกสารเดียวกัน หรือเอกสารต่างหน้ากัน การแสดงลิงค์ส่วนมากมักจะเป็นข้อความที่ขีดเส้นใต้ หรือตัวอักษรที่มีสีแตกต่างจากสีตัวอักษรปกติเพื่อแสดงให้ผู้ใช้เห็นได้ชัดเจน (Thomason, 2001)

ไซต์แมพ (ดังรูปที่ 2.1) เป็นเครื่องมือช่วยนำทาง (Navigation tool) ให้ลิงค์ที่แสดงเนื้อหา (Content) ในเว็บไซต์ทั้งหมด (Shneiderman, 1998) สามารถช่วยให้เว็บไซต์ที่เป็นเครื่องมือสำหรับค้นหา (Search engine) ต่าง ๆ ส่งสไปเดอร์ (Spider หรือ Robot) เข้ามาหาหน้าเว็บที่ผู้ใช้ต้องการค้นหาจากไซต์แมพได้รวดเร็วและมีคำสำคัญ (Keyword) ครบถ้วนมากกว่าการหาจากโค้ด (Code) ในส่วนเมต้าแท็ก (Meta tag) นอกจากนี้ไซต์แมพยังทำให้ผู้ใช้เข้าใจว่าเว็บไซต์ที่กำลังเข้าใช้นั้นมีโครงสร้างอย่างไร ช่วยให้ผู้ใช้สามารถเข้าถึงส่วนของเว็บไซต์ที่ต้องการอย่างรวดเร็วยิ่งขึ้น จุดเด่นที่ทำให้ไซต์แมพแก้ไขจุดด้อยของเครื่องมือช่วยนำทางอื่น ๆ ได้ ดังตารางที่ 2.4 (ข้อมูลจากเว็บไซต์ http://www.webopedia.com/TERM/s/site_map.html ปี 2005)



รูปที่ 2.1 แสดงตัวอย่างไซต์แมพ

ตารางที่ 2.4 ตารางแสดงคุณสมบัติของเครื่องมือช่วยนำทางต่าง ๆ เปรียบเทียบกับไซต์แมพ

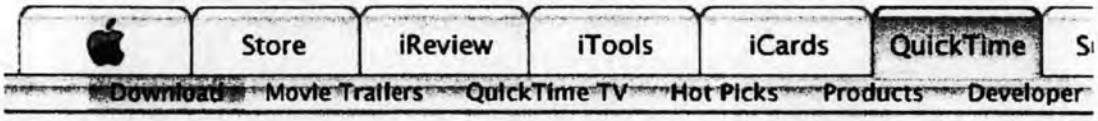
เครื่องมือนำทางรูปแบบต่าง ๆ (Navigation tool)	ไซต์แมพ
1. ปุ่มต่อไปหรือย้อนกลับ (Back/Forward button) ดังรูปที่ 2.2 เป็นปุ่มที่มีในหน้าต่างของเบราเซอร์ (Browser) อยู่แล้ว หรือหากผู้พัฒนาเว็บไซต์ต้องการสร้างไว้ให้อยู่ภายในเว็บไซต์ของตนเองก็ได้ ใช้เมื่อต้องการย้อนกลับไปยังหน้าที่เพิ่งผ่านมา มีวิธีใช้ที่ง่ายไม่ซับซ้อน เพียงแค่กดคลิกจนกระทั่งถึงหน้าที่ต้องการ (Greenberg and Cockburn, 1999) แต่มีข้อด้อยคือ สามารถจดจำหน้าเว็บที่เคยผ่านมาแล้วได้จำนวนจำกัด ไม่สามารถคลิกไปยังหน้าที่เคยเปิดได้ทั้งหมด (Golovchinsky, 2002)	ไม่ต้องอาศัยหน่วยความจำในการจดจำหน้าเว็บที่เคยเปิดมาแล้ว เนื่องจากผู้ใช้สามารถกลับมาเลือกหน้าที่ต้องการเข้าถึงได้ตลอดเวลา และไม่จำกัดจำนวนหน้า
2. เบรดครัมเทรล (Breadcrumbs trail) เป็นเครื่องมือช่วยนำทางที่แสดงพาท (Path) ซึ่งผู้ใช้เปิดขึ้นมา เริ่มตั้งแต่หน้าเว็บหลัก (Home) ไปจนกระทั่งถึงหน้าเว็บที่ผู้ใช้กำลังเปิดอยู่ในปัจจุบัน (Rudl, 2004) เช่น home >> furniture >> dining room >> tables เป็นตัวอย่างที่แสดงว่าตอนนี้ผู้เยี่ยมชมกำลังอยู่ที่หน้า tables ซึ่งเข้ามาโดยผ่านหน้า home, furniture, dining room มาก่อน ซึ่งผู้เยี่ยมชมสามารถกลับไปยังหน้า home หรือ furniture หรือ dining room ได้โดยการคลิกที่ลิงค์ของชื่อนั้น ๆ ที่อยู่ตามพาท (Path) มีข้อด้อยคือเป็นเครื่องมือช่วยนำทางที่แสดงให้เห็นลำดับขั้นของหัวข้อเพียงส่วนหนึ่งจากทั้งหมดของเว็บไซต์เท่านั้น	แสดงให้เห็นหัวข้อเรื่องทั้งหมดที่มีอยู่ในเว็บไซต์ เปรียบเสมือนการแสดงสารบัญเว็บไซต์

ตารางที่ 2.4 (ต่อ) ตารางแสดงคุณสมบัติของเครื่องมือช่วยนำทางต่าง ๆ เปรียบเทียบกับไซต์แมพ

เครื่องมือนำทางรูปแบบต่าง ๆ (Navigation tool)	ไซต์แมพ
3. แท็บเนวิเกชัน (Tab Navigation) ดังรูปที่ 2.3 เปรียบเสมือนแถบเมนูสำหรับเว็บไซต์นั้น ๆ โดยไม่ว่าผู้เยี่ยมชมกำลังอยู่หน้าใดภายในเว็บไซต์ก็สามารถเข้าถึงหน้าเว็บอื่น ๆ ได้ทั้งสิ้น สามารถใช้เป็นเครื่องมือช่วยนำทางหลักภายในเว็บไซต์ได้ (Guibert, 2005) มีข้อด้อยคือในกรณีที่เนื้อหาภายในเว็บไซต์มีจำนวนมากจนกระทั่งไม่สามารถแสดงได้เพียงพอภายใน 6-8 แท็บ ก็สามารถแสดงได้มากกว่านั้น เรียกว่ามัลติโรลแท็บ (Multi-rows Tab) ดังรูปที่ 2.4 แต่การสร้างแท็บมากมายเช่นนี้ทำให้ผู้เยี่ยมชมเกิดความสับสน จนเป็นสาเหตุให้ละทิ้งการเยี่ยมชมเว็บไซต์ได้ (Hobart and Harden, 2003)	สามารถแสดงหัวข้อของเนื้อหาได้ไม่จำกัดจำนวน เนื่องจากไม่จำกัดเนื้อที่ในการแสดงหัวข้อ และไม่ก่อให้เกิดความสับสนต่อผู้ใช้
4. เครื่องมือช่วยนำทางที่วางด้านบน ด้านข้าง หรือด้านล่างของหน้าเว็บ (Top/Side/Bottom Navigation) หรือแถบเมนู (Menu) ดังรูปที่ 2.5 เป็นเครื่องมือช่วยนำทางหลักที่แสดงอยู่ทุกหน้าภายในเว็บไซต์ อาจมีลักษณะเป็นตัวอักษร หรือไอคอน ข้อด้อยคือหากผู้ใช้ต้องการเปิดหน้าเว็บใดที่อยู่ลึกลงไปจากหน้าแถบเมนูหลัก ต้องเข้าผ่านเครื่องมือช่วยนำทางหลักนี้แล้วจึงหาสิ่งที่ต้องการต่อไป เนื่องจากไม่แสดงโครงสร้างหรือลำดับขั้นของเว็บไซต์นั้นให้ผู้ใช้สามารถเข้าใจได้ (Webopedia, 2002)	สามารถเข้าถึงหน้าเว็บที่อยู่ลึกลงไปจากหน้าเว็บหลักได้โดยตรง เปรียบเสมือนทางลัด (Shortcut) ไปสู่หน้าที่ต้องการได้ทันที

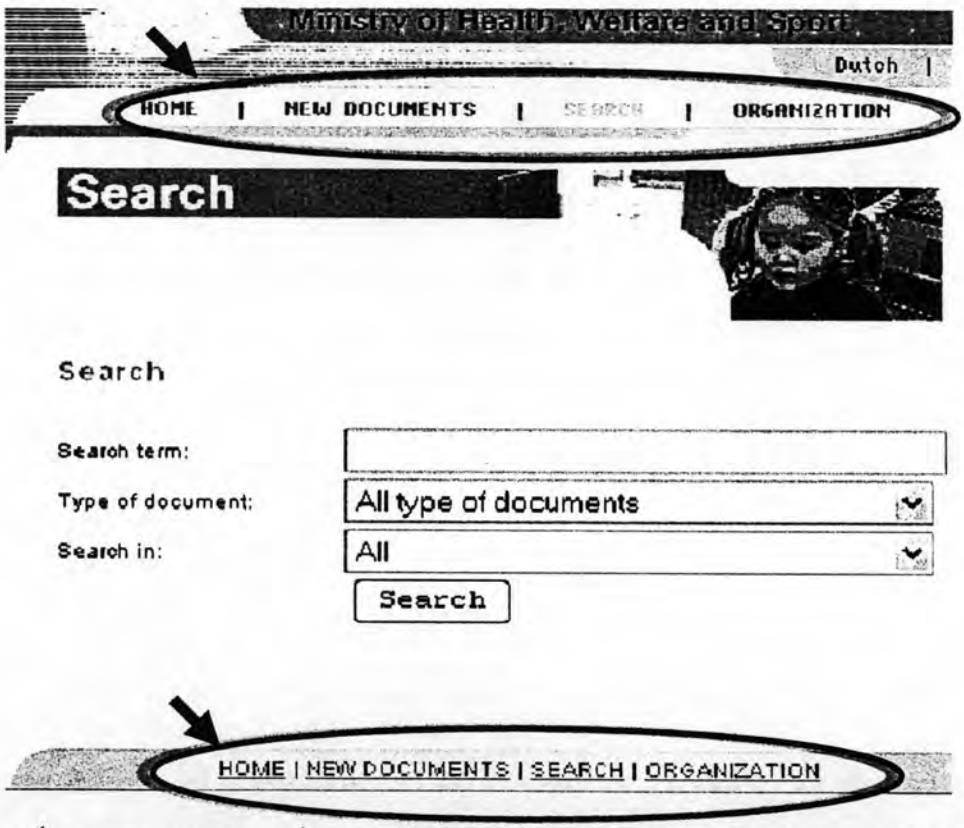
นอกจากไซต์แมพจะสามารถแก้จุดด้อยของเครื่องมือช่วยนำทางอื่น ๆ แล้วยังมีความสามารถสูงแตกต่างจากเครื่องมือนำทางอื่น ๆ ตรงที่สามารถอธิบายให้ผู้ใช้เข้าใจโครงสร้างครอบคลุมทั้งเว็บไซต์ได้ดีกว่าเครื่องมือช่วยนำทางรูปแบบอื่นที่ถูกจำกัดด้วยการแสดงบางส่วนของหัวข้อที่มีอยู่ในเว็บไซต์ทั้งหมด

รูปที่ 2.2 แสดงตัวอย่างปุ่มต่อไปหรือย้อนกลับ (Back/Forward button)



รูปที่ 2.3 แสดงตัวอย่างแท็บเนวิเกชัน (Tab Navigation)

รูปที่ 2.4 แสดงตัวอย่างมัลติโรลแท็บ (Multi-rows Tab)



รูปที่ 2.5 แสดงตัวอย่างเครื่องมือช่วยนำทางที่วางด้านบน หรือด้านล่างของหน้าเว็บ (Top และ Bottom Navigation)

ตามที่เว็บไซต์ของ The Edna McConnell Clark Foundation (2005) กล่าวถึงประโยชน์ของการมีไซต์แมพ และปรัชญาของการออกแบบเว็บไซต์ คือ 1) ต้องสามารถเข้าถึงได้จากทุกหน้าของเว็บไซต์ (Accessibility) 2) มีโครงร่าง (Layout) ที่คงที่ ไม่เปลี่ยนแปลงหากต้องการไปยังหน้าเว็บใดหน้าเว็บหนึ่ง ไม่ว่าจะลิงค์มาจากส่วนใด ๆ ของเว็บไซต์ก็ต้องสามารถ

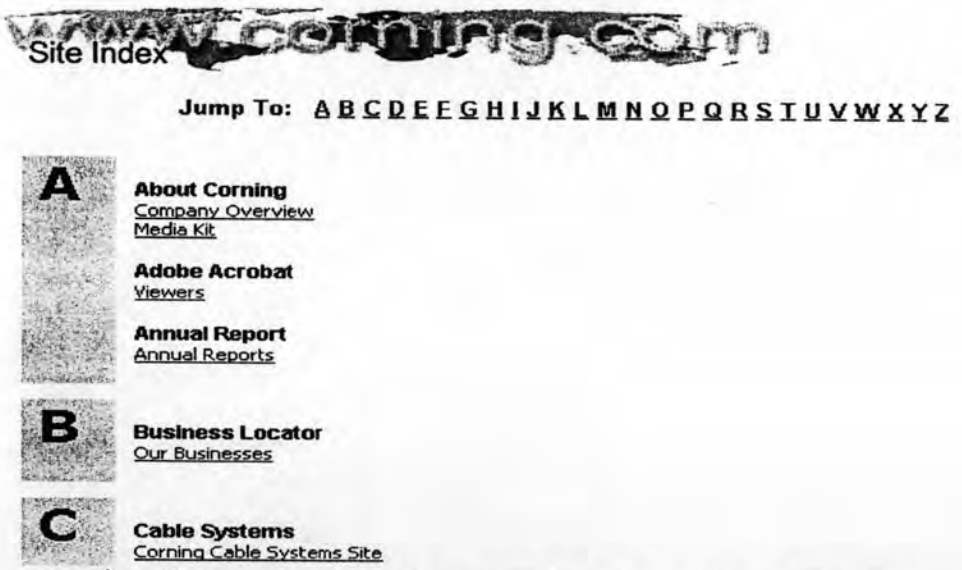
กลับมายังหน้าที่มีโครงร่างเหมือนเดิม (Consistency) 3) สามารถเปลี่ยนแปลงแก้ไข (Change) เนื้อหาในเว็บไซต์ ตรวจดู (Review) และแสดงการเปลี่ยนแปลงแก้ไขนั้นแก่ผู้ใช้ได้ (Publish) รวมเรียกว่า Maintainability โดยทั้งสามข้อหมายถึงความสามารถที่มีอยู่ในไซต์แมพนั่นเอง ยังมีงานวิจัยที่ศึกษาเกี่ยวกับเครื่องมือช่วยนำทางแบบต่าง ๆ รวมทั้งไซต์แมพในด้านต่าง ๆ ดังนี้ Russell (2002) แบ่งไซต์แมพออกเป็นสี่รูปแบบดังต่อไปนี้ (ตัวอย่างประกอบไซต์แมพทั้งสี่รูปแบบอธิบายในหัวข้อ 1.1 ความเป็นมาและความสำคัญของปัญหา)

1. ไซต์แมพแบ่งตามประเภทหัวข้อเรื่องที่ถูกจัดเรียงตามหมวดหมู่ เรียกว่า Categorical

2. ไซต์แมพแบ่งตามลำดับชั้น มีรูปแบบเหมือนรายการที่แบ่งตามประเภทหัวข้อเรื่อง แต่จะมีเส้น (Lines) ประกอบแสดงความสัมพันธ์ระหว่างแต่ละหัวข้อ เรียกว่า Hierarchical

3. ไซต์แมพแสดงแบบกราฟิก มีรูปแบบที่เป็นทางการมากกว่าไซต์แมพแบ่งตามลำดับชั้น (Hierarchical) โดยมักใช้เส้นและมีข้อความประกอบเล็กน้อย แสดงเหมือนแผนภูมิองค์กร (Organizational chart) เรียกว่า Graphical

4. ไซต์แมพซึ่งแสดงรายการของหัวข้อเรื่องที่มีอยู่ในเว็บไซต์เรียงตามลำดับตั้งแต่ A-Z เรียกว่า Alphabetical Index ดังรูปที่ 2.6



รูปที่ 2.6 ไซต์แมพในลักษณะเรียงตามตัวอักษร (Alphabetical Index)

สำหรับงานวิจัยนี้แตกต่างไปบ้างจาก Russell (2002) ตรงที่ให้ความสำคัญกับไซต์แมพเพียงสามรูปแบบแรก โดยไม่ครอบคลุมไซต์แมพแบบเรียงตามตัวอักษร (Alphabetical Index) เนื่องจากในงานวิจัยอื่น เช่น Fox (2003) ไม่จัดว่าการเรียงรายการหัวข้อตามตัวอักษร (Alphabetical Index) เป็นไซต์แมพ เนื่องจากมีการแสดงเฉพาะหัวข้อ (Topic) ของเนื้อหาทั้งหมดภายในเว็บไซต์แบบเรียงตามตัวอักษร ไม่ได้แสดงให้เห็นว่าภายในเว็บไซต์นั้นมีการ

เชื่อมโยงระหว่างแต่ละหัวข้ออย่างไรบ้าง หรือไม่แสดงว่าแต่ละหัวข้อประกอบด้วยหัวข้อย่อยใดบ้าง จึงยากต่อการทำความเข้าใจในโครงสร้างของเว็บไซต์สำหรับผู้เยี่ยมชม อีกทั้งผลสรุปงานวิจัยของ Russell (2002) สนับสนุนว่าไซต์แมพแบบเรียงตามตัวอักษร (Alphabetical Index) มีความถี่ของการใช้งานและได้รับความนิยมจากผู้ใช้น้อยที่สุด (มีความถี่ในการใช้งานเพียง 1 เปอร์เซ็นต์ของการใช้งานไซต์แมพทุกรูปแบบ) นอกจากนี้การจัดตามตัวอักษร (Index) ไม่สามารถแสดงการเชื่อมโยง (Link) ของแต่ละหน้าเว็บขณะที่ไซต์แมพแบบอื่น ๆ สามารถแสดงให้เห็นความสัมพันธ์ระหว่างแต่ละหน้าได้ (McDonald and Stevenson, 1998)

Bernard (1999) ได้ทดลองวัดความยากง่ายในการใช้งานของไซต์แมพสามรูปแบบ คือ ไซต์แมพแบบเรียงตามตัวอักษร (Alphabetical Index) ไซต์แมพแบ่งตามประเภทหัวข้อเรื่องอย่างละเอียด (Full categorical) และไซต์แมพแบ่งตามประเภทหัวข้อเรื่องอย่างคร่าว ๆ (Restricted categorical) โดยผลสรุปที่ได้ของ Bernard (1999) คือ ไซต์แมพแบ่งตามประเภทหัวข้อเรื่องอย่างละเอียด (Full categorical) มีความง่ายในการใช้งานมากที่สุด เนื่องจากข้อมูลถูกจัดเรียงตามประเภทเป็นหมวดหมู่ไม่ก่อให้เกิดความสับสนในตัวโครงสร้างของเว็บไซต์เหมือนไซต์แมพแบบเรียงตามตัวอักษร (Alphabetical Index) และง่ายสำหรับผู้ใช้ในการเปรียบเทียบข้อมูลในแต่ละหมวดหมู่มากกว่าไซต์แมพแบ่งตามประเภทหัวข้อเรื่องอย่างคร่าว ๆ

สำหรับงานวิจัยนี้ผู้วิจัยเห็นว่าการแบ่งไซต์แมพตามประเภทหัวข้อเรื่องอย่างละเอียดหรือไม่นั้นไม่มีนัยสำคัญ เนื่องจากถือว่าเป็นไซต์แมพแบ่งตามประเภทหัวข้อเรื่อง (Categorical) เหมือนกัน รวมทั้งผู้วิจัยต้องการควบคุมให้การแบ่งหัวข้อเนื้อหา (Content) ในไซต์แมพแต่ละรูปแบบมีจำนวนเท่ากัน เพื่อลดความเอนเอียง (Bias) ที่จะเกิดขึ้นกับงานวิจัยจึงไม่นำการแบ่งไซต์แมพในลักษณะดังกล่าวมาพิจารณา

นอกจากนี้ยังมีงานวิจัยที่กล่าวถึงไซต์แมพในด้านที่เกี่ยวกับวิธีการวัดประโยชน์ดังเช่น งานวิจัยของ Bernard (1999) สำรวจประโยชน์ของไซต์แมพแบ่งตามลำดับชั้น (Hierarchical) และไซต์แมพแสดงแบบกราฟิก (Graphical) เปรียบเทียบกัน โดยอาศัยการวัดจริงจากจากจำนวนโหนด (Node) ที่ใช้ในการทำงานที่ได้รับมอบหมายให้ถูกต้อง ซึ่งโหนดหมายถึงจำนวนหน้าเว็บที่ถูกหน่วยทดลองเปิดขึ้นมานั่นเอง โดยได้ผลสรุปว่าไซต์แมพแบ่งตามลำดับชั้น (Hierarchical) และไซต์แมพแสดงแบบกราฟิก (Graphical) สามารถใช้งานได้เหมือนกัน แต่ไซต์แมพแบ่งตามลำดับชั้น (Hierarchical) มีประโยชน์ในการใช้งานมากกว่าเนื่องจากไม่ต้องเสียเวลารอคอยการโหลดเหมือนไซต์แมพแสดงแบบกราฟิก (Graphical) แต่งานวิจัยนี้มีข้อจำกัดอยู่หลายประการคือ (1)ประโยชน์ของไซต์แมพแบ่งตามลำดับชั้น (Hierarchical) วัดจากเว็บไซต์เพียงสองแห่งเท่านั้น ได้แก่ http://www.uprr.com/ และ http://www.exxon.com/ (2) รูปแบบของไซต์แมพน้อยเกินไป (3) อาจมีปัจจัยอื่น ๆ อีกที่เหมาะสำหรับการวัดประโยชน์ของไซต์แมพ นอกจากจำนวนโหนดที่ใช้ทำงาน ดังนั้นการวัดประโยชน์ในการใช้ไซต์แมพควรถูกศึกษาและวิจัยในด้านอื่นอีกเพื่อแก้ข้อจำกัดดังกล่าว