สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย

วิศวกรรมซอฟต์แวร์
Software Engineering

สุวรรณี อัศวกุลชัย

คำนำ

วิศวกรรมซอฟต์แวร์ เป็นพื้นฐานสำคัญในกระบวนการพัฒนาซอฟต์แวร์ที่จำเป็นต้องดำเนินการในรูปแบบกระบวนการทางวิศวกรรมซอฟต์แวร์ อาจเป็นการปรับเปลี่ยนระบบงานเดิมที่มีอยู่แล้ว หรือการสร้างระบบงานใหม่ สามารถแก้ปัญหาต่าง ๆ เพื่อให้การดำเนินงานทางธุรกิจมีประสิทธิภาพสูงสุด มุ่งเน้นตามความต้องการของผู้ใช้งานเป็นสำคัญ **ทั้งนี้หน้าที่สำคัญของวิศวกรซอฟต์แวร์ก็คือกระบวนการพัฒนาซอฟต์แวร์ให้มีประสิทธิภาพ**

หนังสือเล่มนี้กล่าวถึงกระบวนการพัฒนาซอฟต์แวร์ที่สามารถนำไปประยุกต์การใช้งานได้จริงในทางปฏิบัติ ภายใต้การดำเนินการตามวงจรการพัฒนาระบบ เนื้อหาในหนังสือเล่มนี้แบ่งออกเป็น 12 บท บทที่ 1 ความรู้เบื้องต้นของวิศวกรรมซอฟต์แวร์ และหลักการเบื้องต้นของวิศวกรรมซอฟต์แวร์ บทที่ 2 การบริหารโครงการพัฒนาซอฟต์แวร์ บทที่ 3 แบบจำลองกระบวนการของซอฟต์แวร์ บทที่ 4 กระบวนการของการพัฒนาซอฟต์แวร์ เพื่อความเข้าใจมากยิ่งขึ้น นับว่าเป็นหัวใจของหนังสือเล่มนี้ และเป็นสิ่งที่ผู้อ่านอยากทราบ ไม่ว่าจะเป็น ทำอย่างไร ทำอะไร เมื่อใด เน้นตามวงจรพัฒนาซอฟต์แวร์ ซึ่งมีรายละเอียดแต่ละกระบวนการในบทถัดไป บทที่ 5 วิศวกรรมความต้องการ บทที่ 6 การวิเคราะห์และออกแบบระบบเชิงโครงสร้าง บทที่ 7 การวิเคราะห์และออกแบบระบบเชิงอ็อบเจกต์ บทที่ 8 การออกแบบซอฟต์แวร์ บทที่ 9 การพัฒนาซอฟต์แวร์ บทที่ 10 การทดสอบซอฟต์แวร์ บทที่ 11 การประกันคุณภาพซอฟต์แวร์ และบทที่ 12 ขอนำเสนอ กรณีศึกษา จากทฤษฎีสู่การปฏิบัติ เป็นการประมวลความรู้เพื่อเป็นแนวทางปฏิบัติที่เรียกว่า “HOW TO” นั่นเอง

หากมีข้อบกพร่อง ข้อผิดพลาด ผู้เขียนขอน้อมรับเพื่อนำไปปรับปรุงต่อไป ขอขอบคุณทุกข้อแนะนำ ข้อเสนอแนะของผู้อ่านทุกท่าน และหวังเป็นอย่างยิ่งว่า ถ้าได้อ่านแล้ว จะเข้าใจว่ากระบวนการพัฒนาซอฟต์แวร์ไม่ใช่เรื่องยาก

สุวรรณี อัศวกุลชัย
ตุลาคม 2561

สารบัญ

บทที่ 1

ความรู้เบื้องต้นของวิศวกรรมซอฟต์แวร์

วัตถุประสงค์

1. เพื่อเข้าใจความรู้พื้นฐานของวิศวกรรมซอฟต์แวร์
2. เพื่อเข้าใจหลักการเบื้องต้นของวิศวกรรมซอฟต์แวร์
3. เพื่ออธิบายแบบจำลองกระบวนการของซอฟต์แวร์
4. เพื่อเข้าใจความท้าทายต่อวิศวกรรมซอฟต์แวร์

ยิ่งนับวัน คอมพิวเตอร์ได้เข้ามามีบทบาทในชีวิตการทำงานมากยิ่งขึ้น ลำพังเครื่องคอมพิวเตอร์ฮาร์ดแวร์อย่างเดียว คงจะไม่สามารถก่อให้เกิดงานต่าง ๆ มากมายได้ ต้องอาศัยทั้งคอมพิวเตอร์ซอฟต์แวร์ และมนุษย์เพื่อใช้ซอฟต์แวร์ในการสร้างงานต่าง ๆ ดังนั้น ซอฟต์แวร์ต้องมีการพัฒนาเพื่อให้ตรงตามความต้องการของผู้ใช้ในแต่ละธุรกิจของแต่ละประเภท จึงนับว่าการพัฒนาซอฟต์แวร์เป็นสิ่งที่สำคัญ

วิศวกรรมซอฟต์แวร์ เป็นศาสตร์หนึ่งของวิศวกรรม ในการเน้นถึงการพัฒนาระบบซอฟต์แวร์ให้มีประสิทธิภาพ ซอฟต์แวร์เป็นสิ่งที่จับต้องไม่ได้ แต่เข้าใจได้ง่าย นอกจากนั้นวิศวกรรมซอฟต์แวร์ยังเป็นอิสระจากฮาร์ดแวร์

คำว่า "วิศวกรรมซอฟต์แวร์"[1] เป็นคำที่กำเนิดขึ้นมาในปี 2511 ในการประชุมสัมมนาเกี่ยวกับวิกฤตการณ์ของซอฟต์แวร์ ซึ่งเกิดขึ้นเนื่องจากการพัฒนาระบบคอมพิวเตอร์ฮาร์ดแวร์ใหม่ มีผลทำให้ซอฟต์แวร์ต้องมีขนาดใหญ่และซับซ้อนกว่าเดิมมาก จากอดีตที่ผ่านมา การพัฒนาซอฟต์แวร์ยังไม่เป็นระบบเพียงพอ ก่อให้เกิดปัญหาการส่งงานที่ไม่ตรงต่อเวลา ต้นทุนที่เพิ่มขึ้น รวมไปถึงการบำรุงรักษาโปรแกรมที่ทำได้ยาก ทั้งหมดนี้จึงส่งผลให้เกิดวิกฤตการณ์ของซอฟต์แวร์นั่นเอง

การพัฒนาเทคนิคต่าง ๆ ซึ่งเป็นส่วนหนึ่งของวิศวกรรมซอฟต์แวร์ เพื่อให้ได้ซอฟต์แวร์ที่เชื่อถือได้ ส่งมอบงานให้ลูกค้าได้ตรงตามเวลา และอยู่ในกรอบของงบประมาณที่กำหนด ดังนั้น วิศวกรรมซอฟต์แวร์ ทำให้เกิดความเข้าใจเกี่ยวกับกิจกรรมของการพัฒนาซอฟต์แวร์ ตั้งแต่วิธีการกำหนดข้อกำหนดของซอฟต์แวร์ การออกแบบ และนำไปสู่การปฏิบัติ อย่างไรก็ตาม ยังมีความหลากหลายในด้านเทคนิคต่าง ๆ ในการพัฒนาวิศวกรรมซอฟต์แวร์ จึงนับว่าเป็นสิ่งสำคัญมาก

ส่งผลให้วิศวกรรมซอฟต์แวร์ยังคงเป็นศาสตร์ที่ต้องพัฒนาอย่างต่อเนื่อง เพื่อพัฒนาการออกแบบซอฟต์แวร์ต่าง ๆ เช่น การติดต่อสื่อสารทางอินเทอร์เน็ต รวมถึงอินเทอร์เน็ตของสรรพสิ่ง (Internet of thing) การทำธุรกรรมต่าง ๆ ผ่านโทรศัพท์เคลื่อนที่ ดังนั้น วิศวกรรมซอฟต์แวร์จึงมีประโยชน์อย่างมาก และจะมากขึ้นต่อไปในศตวรรษที่ 21

1.1 ซอฟต์แวร์คืออะไร

ซอฟต์แวร์[2] คือ ชุดของคำสั่งที่เขียนให้เครื่องคอมพิวเตอร์ดำเนินงานต่าง ๆ ตามความต้องการของผู้ใช้ ประกอบด้วย โปรแกรมคอมพิวเตอร์ โครงสร้างของแฟ้มข้อมูล (File organization)

---

[1] https://th.wikipedia.org

[2] https://en.wikipedia.org/wiki/Software

โครงสร้างข้อมูล (Data structure) คลังโปรแกรม (library) ต่าง ๆ นอกจากนั้นยังมีเอกสารของระบบที่อธิบายโครงสร้างของระบบ การติดตั้งระบบ และคู่มือการใช้งาน เพื่ออธิบายขั้นตอนการทำงานของโปรแกรม

1.2 นิยามของวิศวกรรมซอฟต์แวร์

วิศวกรรมซอฟต์แวร์[3] คือ ศาสตร์หนึ่งของทางวิศวกรรมที่ว่าด้วยการพัฒนาซอฟต์แวร์ ในทุกมุมมองตั้งแต่ต้นทางจนไปถึงปลายทาง จากนิยามพบว่ามีคำหลักอยู่ 2 ประเด็น คือ

1. ศาสตร์ทางวิศวกรรม หมายถึง งานที่วิศวกรต้องประยุกต์ใช้ทั้งทฤษฎี วิธีการ และเครื่องมือต่าง ๆ เพื่อนำมาใช้ในการแก้ไขปัญหาต่าง ๆ ถึงแม้ว่าอาจจะไม่มีทฤษฎีที่เหมาะสมก็ตาม วิศวกรเหล่านี้ก็ต้องทำงานอย่างเป็นระบบภายใต้สภาวะขีดจำกัด ทั้งเวลาและงบประมาณ

2. ทุกมุมมองของการพัฒนาซอฟต์แวร์ หมายถึง วิศวกรรมซอฟต์แวร์ต้องคำนึงถึงกระบวนการพัฒนาซอฟต์แวร์ทั้งในด้านเทคนิคและกิจกรรมอื่น ๆ คือ การบริหารจัดการ เครื่องมือต่าง ๆ วิธีการ และทฤษฎีในการสนับสนุนการพัฒนาซอฟต์แวร์

อาจกล่าวได้ว่า วิศวกรซอฟต์แวร์ต้องทำงานอย่างเป็นระบบและจัดการงานต่าง ๆ ได้ โดยต้องเป็นวิธีการที่มีประสิทธิภาพเพื่อให้ได้มาซึ่งซอฟต์แวร์ที่มีคุณภาพ

1.3 ความแตกต่างระหว่างวิศวกรรมซอฟต์แวร์กับวิทยาศาสตร์คอมพิวเตอร์

วิทยาการคอมพิวเตอร์ หรือ วิทยาศาสตร์คอมพิวเตอร์[4] **(Computer science)** เป็นศาสตร์เกี่ยวกับการศึกษาและค้นคว้าทฤษฎีการคำนวณสำหรับคอมพิวเตอร์ และทฤษฎีการประมวลผลสารสนเทศ ทั้งด้านซอฟต์แวร์ ฮาร์ดแวร์ และเครือข่าย วิทยาการคอมพิวเตอร์นั้นประกอบด้วยหลายหัวข้อที่เกี่ยวข้องกับคอมพิวเตอร์ ตั้งแต่ระดับนามธรรม หรือความคิดเชิงทฤษฎี เช่น การวิเคราะห์และสังเคราะห์ขั้นตอนวิธี ไปจนถึงระดับรูปธรรม เช่น ทฤษฎีภาษาโปรแกรม ทฤษฎีการพัฒนาซอฟต์แวร์ ทฤษฎีฮาร์ดแวร์คอมพิวเตอร์ ทฤษฎีเครือข่าย

วิศวกรรมซอฟต์แวร์ (Software engineering)[5] เป็นศาสตร์เกี่ยวกับวิศวกรรมด้านซอฟต์แวร์ มีเนื้อหาเกี่ยวข้องกับการใช้กระบวนการทางวิศวกรรมในการดูแลการผลิต ตั้งแต่การรวบรวม

---

[3] https://en.wikipedia.org/wiki/Software_engineering

[4] https://th.wikipedia.org/wiki/วิทยาการคอมพิวเตอร์

[5] https://th.wikipedia.org/wiki/วิศวกรรมซอฟต์แวร์

ความต้องการ การตั้งเป้าหมายของระบบ การออกแบบ กระบวนการพัฒนา การตรวจสอบ การประเมินผล การติดตามโครงการ การประเมินต้นทุน การรักษาความปลอดภัย ไปจนถึงการคิดราคาซอฟต์แวร์ เป็นต้น

1.4 หลักการเบื้องต้นของวิศวกรรมซอฟต์แวร์

เพื่อความสำเร็จในการพัฒนาซอฟต์แวร์ จึงควรยึดหลักการเบื้องต้นของวิศวกรรมซอฟต์แวร์ ประกอบด้วย 6 ประเด็น ดังนี้

1. การวิเคราะห์ปัญหาให้ชัดเจน
2. การแบ่งงานออกเป็นส่วนย่อย ๆ เพื่อความง่ายในการติดตามตรวจสอบ
3. การจัดลำดับความสำคัญของงาน
4. การแก้ไขปัญหาต่าง ๆ ได้อย่างเป็นระบบ
5. การนำวิธีการแก้ปัญหาหนึ่งไปใช้แก้ปัญหาได้หลาย ๆ ปัญหา
6. การเพิ่มเติมแก้ไขได้ง่าย

1.4.1 การวิเคราะห์ปัญหาให้ชัดเจน

ในการพัฒนาซอฟต์แวร์เพื่อให้สามารถทำงานได้ตามความต้องการ จำเป็นต้องวิเคราะห์ระบบงานปัจจุบัน เพื่อให้ทราบขั้นตอนการทำงานอย่างชัดเจน รวมทั้งทราบถึงขั้นตอนการทำงานที่ไม่มีประสิทธิภาพ จากนั้นต้องวิเคราะห์ปัญหาเพื่อนำไปสู่การพัฒนาซอฟต์แวร์ สามารถวิเคราะห์ปัญหาได้จากการสัมภาษณ์ ตั้งแต่ผู้บริหาร เพื่อเข้าใจภาพรวมของการดำเนินงานทั้งระบบ และขอการสนับสนุนการให้ความร่วมมือจากทุก ๆ ฝ่าย จนถึงผู้ปฏิบัติงาน เพื่อทราบขั้นตอนการดำเนินงานอย่างละเอียด รวมไปถึงการศึกษาจากเอกสารต่าง ๆ ด้วย และต้องกำหนดองค์ประกอบของโครงสร้างข้อมูล ซึ่งโครงสร้างข้อมูลต้องมีการกำหนดพฤติกรรมอย่างเฉพาะเจาะจง การสนับสนุน ความถูกต้องของข้อมูล ส่วนประกอบโครงสร้างข้อมูล

ตัวอย่าง การวิเคราะห์ปัญหาในงานวิจัยการพัฒนาระบบการเฝ้าระวังติดตามผู้ป่วยโรคเบาหวานผ่านโทรศัพท์เคลื่อนที่

จากการศึกษาและติดตามผลระดับน้ำตาลในเลือดของผู้ป่วยโรคเบาหวานในปัจจุบัน พบว่ายังไม่มีการเชื่อมต่ออย่างเป็นระบบ ถึงแม้จะมีการพัฒนาระบบติดต่อผ่านโทรศัพท์เคลื่อนที่แล้วก็ตาม ก็ยังไม่เป็นที่นิยม เนื่องจากต้องอาศัยเทคโนโลยีทางคอมพิวเตอร์หลายระบบ ตั้งแต่อุปกรณ์กลูโคสมิเตอร์ที่ต้องเป็นระบบดิจิทัล และมีระบบการส่งข้อมูลแบบอินฟราเรด ปัญหาก็คือ การใช้งานสำหรับผู้ป่วยโรคเบาหวานยุ่งยากและไม่สะดวก

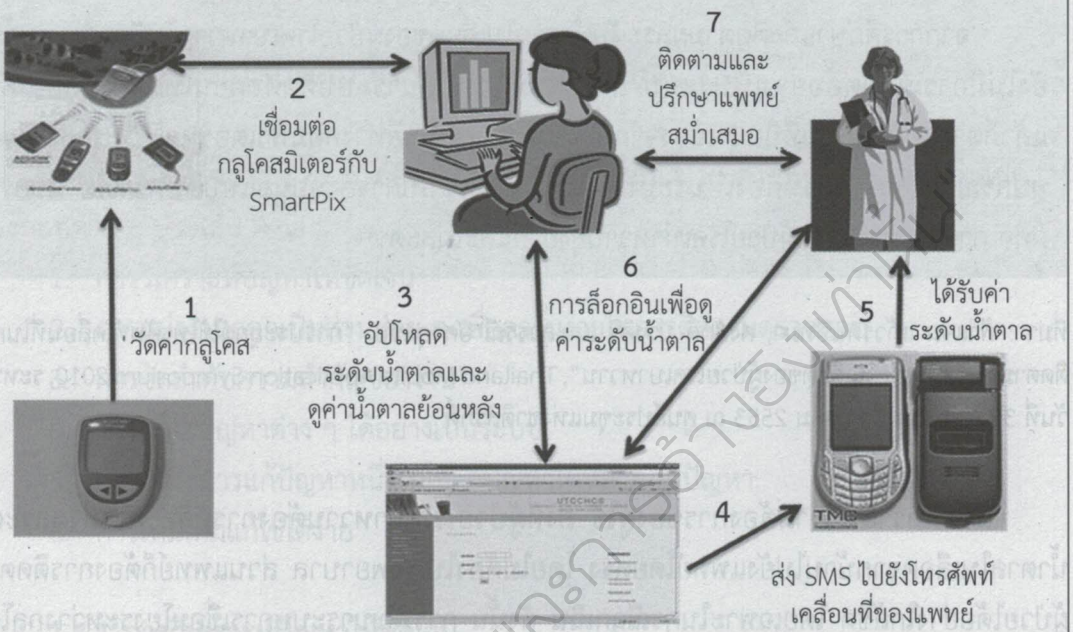
ที่มา : ฟ้าแจ่ม แก้วรัตนปัทมา, คงศักดิ์ ม่วงศิริ และสุวรรณี อัศวกุลชัย, “การประยุกต์ใช้โทรศัพท์เคลื่อนที่ในการติดตามระดับน้ำตาลในเลือดของผู้ป่วยโรคเบาหวาน”, Thailand Mobile Application Symposium 2010, ระหว่างวันที่ 30 กันยายน-3 ตุลาคม 2553 ณ ศูนย์ประชุมแห่งชาติสิริกิติ์

การวิเคราะห์ความต้องการของผู้ใช้ สิ่งที่ผู้ป่วยโรคเบาหวานต้องการก็คือ การส่งผลระดับน้ำตาลในเลือดจากบ้านไปยังแพทย์โดยตรง โดยไม่ต้องไปโรงพยาบาล ส่วนแพทย์ก็ต้องการติดตามผู้ป่วยได้อย่างใกล้ชิด โดยเฉพาะในกรณีฉุกเฉิน ดังนั้น การพัฒนาระบบการเชื่อมโยงระหว่างกลูโคสมิเตอร์ให้เข้ากับคอมพิวเตอร์ผ่านเว็บแอปพลิเคชัน พร้อมกับส่งค่าระดับน้ำตาลในเลือดผ่านระบบโทรศัพท์เคลื่อนที่ของผู้ป่วยและแพทย์โดยแสดงผลเป็นกราฟ สามารถแสดงแนวโน้มของระดับน้ำตาลในเลือดได้ จากการวิเคราะห์ปัญหา สิ่งที่ต้องดำเนินการต่อไปคือ การแบ่งงานออกเป็นส่วนย่อย ๆ ซึ่งจะได้กล่าวต่อไป

1.4.2 การแบ่งงานออกเป็นส่วนย่อย ๆ

ระบบที่ซับซ้อน ต้องแบ่งออกเป็นส่วนย่อย ๆ ซึ่งต้องมีการเขียนโปรแกรมย่อย ๆ ที่เรียกว่ามอดูล (module) แบ่งตามหน้าที่และความรับผิดชอบที่มีความเกี่ยวข้องกัน ประโยชน์ของการแบ่งงานออกเป็นส่วนย่อย ๆ คือ ทำให้ง่ายในการติดตามและตรวจสอบ

ตัวอย่าง การแบ่งงานออกเป็นส่วนย่อย ๆ ในงานวิจัยการพัฒนาระบบการเฝ้าระวังติดตามผู้ป่วยโรคเบาหวานผ่านโทรศัพท์เคลื่อนที่ แสดงดังรูปที่ 1.1



รูปที่ 1.1 การแบ่งงานออกเป็นส่วนย่อย ๆ ของการพัฒนาระบบการเฝ้าระวังติดตามผู้ป่วยโรคเบาหวานผ่านโทรศัพท์เคลื่อนที่

จากรูปที่ 1.1 การทำงานของระบบการเฝ้าระวังติดตามผู้ป่วยโรคเบาหวานผ่านโทรศัพท์เคลื่อนที่แบ่งงานออกเป็นส่วนย่อย ๆ ดังนี้

1. งานพัฒนาระบบการเชื่อมต่อเครื่องกลูโคสมิเตอร์กับ SmartPix และ SmartPix กับคอมพิวเตอร์ เพื่อส่งค่าระดับน้ำตาลในเลือดด้วยเครื่องกลูโคสมิเตอร์จากผู้ป่วยโรคเบาหวาน เพื่อบันทึกผลลงฐานข้อมูลในเซิร์ฟเวอร์ (server)

2. งานพัฒนาระบบการบันทึกค่าระดับน้ำตาลในเลือดที่ตรวจได้จากเครื่องกลูโคสมิเตอร์เข้าสู่ระบบฐานข้อมูลของเครื่องเซิร์ฟเวอร์ เพื่อดูค่าระดับน้ำตาลในเลือดย้อนหลัง

3. งานพัฒนาระบบทำการส่งค่าระดับน้ำตาลในเลือดของผู้ป่วยให้กับแพทย์ประจำตัวด้วยการส่ง SMS ไปยังโทรศัพท์เคลื่อนที่ของแพทย์

4. เมื่อแพทย์ได้รับค่าระดับน้ำตาลในเลือดของผู้ป่วยแล้ว แพทย์สามารถทำการลงบันทึก (login) เข้าสู่ระบบ เพื่อติดตามผลที่ได้รับมาล่าสุดและย้อนหลังของผู้ป่วยโรคเบาหวานได้

1.4.3 การจัดลำดับความสำคัญของงาน

ขั้นตอนการพัฒนาซอฟต์แวร์ หลังจากทราบการทำงานของระบบงานปัจจุบันแล้ว ทำให้เลือกได้ว่าจะสร้างระบบงานใดก่อน ดังนั้น ความสามารถในการจัดลำดับความสำคัญของงาน จากการศึกษากระบวนการไหลของกระบวนการ (Process flow) และการไหลของข้อมูล (Data flow) จึงพัฒนาแบบจำลองของงาน และพิจารณาองค์ประกอบของซอฟต์แวร์ เพื่อจัดลำดับความสำคัญในการพัฒนา

ตัวอย่าง การจัดลำดับความสำคัญของงาน ในงานวิจัยการพัฒนาระบบการเฝ้าระวังติดตามผู้ป่วยโรคเบาหวานผ่านโทรศัพท์เคลื่อนที่

การพัฒนาระบบการเฝ้าระวังติดตามผู้ป่วยโรคเบาหวานผ่านโทรศัพท์เคลื่อนที่ แบ่งออกเป็น 2 ส่วน ส่วนแรกเป็นโปรแกรมการโอนถ่ายข้อมูลจากเครื่องกลูโคสมิเตอร์นำมาจัดเก็บไว้ในรูปของไฟล์ข้อมูล ส่วนที่สองเป็นส่วนของเว็บแอปพลิเคชัน ในส่วนนี้ให้นำไฟล์ข้อมูลมาบันทึกลงฐานข้อมูล รวมไปถึงส่ง SMS และดึงข้อมูลจากฐานข้อมูลมาแสดงเป็นกราฟ

การจัดลำดับความสำคัญของงาน ต้องเริ่มจากการโอนถ่ายข้อมูลจากเครื่องกลูโคสมิเตอร์ให้ได้ก่อน เพราะถ้าไม่สามารถโอนถ่ายข้อมูลจากเครื่องกลูโคสมิเตอร์ได้ จะดำเนินการขั้นตอนถัดไปไม่ได้ ต่อมาจึงพัฒนาระบบฐานข้อมูลในการจัดเก็บข้อมูลที่เกี่ยวข้องทั้งระบบ และพัฒนาระบบเชื่อมโยงเข้าด้วยกัน

1.4.4 การแก้ไขปัญหาต่าง ๆ ได้อย่างเป็นระบบ

โปรแกรมคอมพิวเตอร์ เป็นเครื่องมือที่ใช้ในการแก้ไขปัญหาได้อย่างเป็นระบบ ตราบเท่าที่นักพัฒนาซอฟต์แวร์สามารถจัดการกับโครงสร้างและขั้นตอนวิธีการ โดยคำนึงถึงความหมายหรือความสำคัญของข้อมูลที่มีส่วนเกี่ยวข้อง เนื่องจากนักพัฒนาซอฟต์แวร์ใช้ขั้นตอนวิธี (algorithm) ในการแก้ไขปัญหาต่าง ๆ โดยเฉพาะการประมวลผลข้อมูลที่ต้องการคำตอบแบบทันที เป็นต้น

สามารถยืมและติดตามหนังสือใหม่ได้ที่ ระบบห้องสมุดอัตโนมัติ WALAI AutoLib

http://lib.rmutp.ac.th/catalog/BibItem.aspx?BibID=b00104561

Multi view View map

วิศวกรรมซอฟต์แวร์ = Software engineering / สุวรรณี อัศวกุลชัย.

Author	สุวรรณี อัศวกุลชัย
Edition	พิมพ์ครั้งที่ 1
Detail	571 หน้า : ภาพประกอบ ; 26 ซม
Subject	วิศวกรรมซอฟต์แวร์(+) ซอฟต์แวร์(+)
ISBN	9789740338093
ประเภทแหล่งที่มา	Book