บทที่ 3

ระเบียบวิธีวิจัย

ในบทนี้จะกล่าวถึงระเบียบวิธีวิจัย ซึ่งประกอบด้วย แนวทางในการวิจัยโดยรวม แผนแบบการทดลอง (Experimental Design) การทดสอบสมมติฐาน ขั้นตอนการเก็บรวบรวมข้อมูล (Data Gathering Execution) กรอบของการวิเคราะห์ข้อมูล (Data Analysis Framework) และประเด็นของความเชื่อถือได้ (Reliability) และความถูกต้อง (Validity) เพื่อเอื้อประโยชน์ต่อการดำเนินการพัฒนาการวิจัยในลำดับต่อไป ดังนี้

3.1 แนวทางการวิจัยโดยรวม

งานวิจัยนี้ ผู้วิจัยได้ออกแบบให้เป็นงานวิจัยเชิงทดลอง (Experimental Design) เนื่องจากต้องการศึกษาถึงความสัมพันธ์เชิงเหตุผลระหว่างตัวแปร (ศิริวรรณ และคณะ, 2541) โดยตัวแปรที่สนใจคือ โครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน (Association Relationship) และโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน (Generalization Relationship) กล่าวคือ ผู้วิจัยต้องการ (1) เปรียบเทียบประสิทธิภาพของซอฟต์แวร์ (Software Performance) ขณะประมวลผล (Run – Time) และ (2) เปรียบเทียบผลกระทบ (Impact) จากการเปลี่ยนแปลงซอฟต์แวร์ (Software Change) ตามความต้องการเชิงฟังก์ชัน ระหว่างโครงสร้างของคลาสที่แตกต่างกันว่ามีผลกับคุณภาพของซอฟต์แวร์ในด้านการแก้ไข ปรับปรุง และบำรุงรักษาซอฟต์แวร์ (Software Maintainability) อย่างไร เพื่อรองรับการเปลี่ยนแปลงความต้องการทางธุรกิจ โดยตัวแปรอื่น ๆ เช่น เครื่องมือสำหรับอ่านกระบวนการทำงานของซอฟต์แวร์หน่วยตัวอย่างและหาค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลและซอฟต์แวร์ซอร์สโค้ดของหน่วยตัวอย่างที่นำมาใช้ในการทดลองต้องถูกพัฒนาด้วยภาษาโปรแกรมเดียวกัน เพื่อให้ผลการทดลองที่เกิดขึ้น เป็นผลมาจากการเปลี่ยนแปลงโครงสร้างของคลาสอย่างแท้จริง และให้ผลการวิจัยที่น่าเชื่อถือเนื่องจากการควบคุมสิ่งแวดล้อมและทรัพยากรในการทดลองที่เหมือนกัน

3.2 แผนแบบการทดลอง (Experimental Design)

เนื่องจากผู้วิจัยต้องการวัดค่าตัวแปรตามของหน่วยตัวอย่างว่ามีผลกับตัวแปรต้นอย่างไร และผลที่ได้มีค่าที่แตกต่างกันอย่างไร โดยงานวิจัยนี้จะแบ่งแผนแบบการทดลองออกเป็น 2 แผน

แบบ คือ (1) การเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล และ (2) การเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงของซอฟต์แวร์ ซึ่งมีรายละเอียดปัจจัย ดังนี้

3.2.1 แผนแบบการทดลองการเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล

ปัจจัยที่เกี่ยวข้องกับการเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล ประกอบด้วย

- **ตัวแปรต้น (Independent Variable)** โครงสร้างของคลาส ซึ่งได้แก่ โครงสร้างของคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างของคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน
- **ตัวแปรตาม (Dependent Variable)** การวัดประสิทธิภาพของโครงสร้างคลาสของซอฟต์แวร์ระหว่างโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ภายใต้การประมวลผลซอฟต์แวร์ ซึ่งสามารถกำหนดตัวแปรตามได้ดังนี้
  - P_x คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากจำนวนการส่งข้อความหรือเมสเสจ
  - P_y คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากจำนวนการส่งข้อความหรือเมสเสจ
  - T_x คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์
  - T_y คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์
- **ตัวแปรควบคุม (Control Variable)** เพื่อให้ผลของการทดลองเกิดจากการปรับเปลี่ยนค่าตัวแปรต้นอย่างแท้จริง ทำให้มีการควบคุมปัจจัยอื่นที่ส่งผลกระทบกับการทดลองให้คงที่และให้คล้ายคลึงกันมากที่สุด ดังนี้

1. การเลือกหน่วยตัวอย่างที่นำมาใช้ในงานวิจัย ผู้วิจัยกำหนดหน่วยตัวอย่างจำนวน 5 หน่วยตัวอย่าง โดยแต่ละหน่วยตัวอย่างต้องเป็นซอฟต์แวร์ทางธุรกิจที่ออกแบบ

โดยใช้หลักการเชิงวัตถุ และมีจำนวนคลาสการทำงานในเชิงธุรกิจอย่างน้อย 5 คลาส เช่น ระบบออนไลน์เอทีเอ็ม ระบบการขายหน้าร้าน เป็นต้น ซึ่งแต่ละหน่วยตัวอย่างที่นำมาใช้ในงานวิจัยต้องเป็นซอฟต์แวร์พัฒนาภายใต้โครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันที่มีการสืบทอดคุณสมบัติ (Inheritance) จากซูปเปอร์คลาส (Super Class) และมีซับคลาส (Subclass) ที่รับการถ่ายทอดคุณสมบัติจากซูปเปอร์คลาสอย่างน้อยหนึ่งซับคลาสในโครงสร้างคลาสของซอฟต์แวร์

2. จากโครงสร้างของคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ผู้วิจัยนำมาปรับเปลี่ยนให้เป็นโครงสร้างของคลาสในความสัมพันธ์แบบแอสโซซิเอชัน และนำโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันและแอสโซซิเอชันมาพัฒนาเป็นซอฟต์แวร์ซอร์สโค้ดสำหรับนำไปประมวลผล โดยซอร์สโค้ดที่ใช้ในการทดลองต้องถูกพัฒนาด้วยภาษาโปรแกรมเดียวกัน เนื่องจากผู้วิจัยไม่ต้องการให้เกิดปัจจัยทางด้านภาษาโปรแกรมที่ส่งผลกระทบต่อประสิทธิภาพของซอฟต์แวร์ จึงจำเป็นต้องกำหนดการใช้ภาษาโปรแกรมที่พัฒนาให้เหมือนกัน โดยผู้วิจัยเลือกโปรแกรมภาษาจาวา (Java Programming Language) เป็นภาษาโปรแกรมที่ใช้สำหรับพัฒนาซอฟต์แวร์ซอร์สโค้ดหน่วยตัวอย่าง
3. เครื่องมือสำหรับหาค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล ผู้วิจัยจะพัฒนาเป็นซอฟต์แวร์สำหรับช่วยในการอ่านกระบวนการทำงานของซอฟต์แวร์หน่วยตัวอย่างและหาค่าประสิทธิภาพของซอฟต์แวร์ของโครงสร้างของคลาสขณะประมวลผลอย่างอัตโนมัติ เพื่อให้การคำนวณและจัดเก็บค่าถูกกระทำในสิ่งแวดล้อมและทรัพยากรเดียวกัน โดยซอฟต์แวร์ที่พัฒนาต้องพัฒนาภายใต้โปรแกรมภาษาเดียวกับหน่วยตัวอย่าง ดังนั้น เครื่องมือจะต้องถูกพัฒนาภายใต้โปรแกรมภาษาจาวาด้วย

3.2.2 แผนแบบการทดลองการเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงของซอฟต์แวร์

ปัจจัยที่เกี่ยวข้องกับการเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงของซอฟต์แวร์ประกอบด้วย

- ตัวแปรต้น **(Independent Variable)** คือ โครงสร้างของคลาส ซึ่งได้แก่ โครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

- **ตัวแปรตาม (Dependent Variable)** การวัดการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ โดยพิจารณาผลกระทบจากการเปลี่ยนแปลงโครงสร้างและคุณลักษณะของคลาส ภายใต้การเปลี่ยนแปลงความต้องการของซอฟต์แวร์ (Software Requirement Specification) ที่เพิ่มขึ้น ว่ามีผลกับคุณภาพของซอฟต์แวร์ในเชิงของการแก้ไขปรับปรุงและการบำรุงรักษาซอฟต์แวร์ โดยคำนวณจากจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของซอฟต์แวร์กรณีศึกษา ซึ่งสามารถกำหนดตัวแปรตามได้ดังนี้

  C_x คือ จำนวนคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน

  C_y คือ จำนวนคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

  M_x คือ จำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน

  M_y คือ จำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

โดยการกำหนดหน่วยตัวอย่างเพื่อใช้ในการเปลี่ยนแปลงซอฟต์แวร์กรณีศึกษา ต้องอยู่ในรูปแบบของความต้องการเชิงฟังก์ชัน (เพื่อให้ซอฟต์แวร์กรณีศึกษามีการเปลี่ยนแปลง) โดยกำหนดให้มีจำนวนหน่วยตัวอย่าง 30 หน่วยตัวอย่าง

- **ตัวแปรควบคุม (Control Variable)** เพื่อให้ผลของการทดลองเกิดจากการปรับเปลี่ยนค่าตัวแปรต้นอย่างแท้จริง จึงมีการควบคุมปัจจัยอื่นที่ส่งผลกระทบกับการทดลองให้คงที่และให้คล้ายคลึงกันมากที่สุด ดังนี้

  1. การเลือกกรณีศึกษาสำหรับใช้ในงานวิจัย กำหนดให้เป็นซอฟต์แวร์ทางธุรกิจ ที่พัฒนาด้วยหลักการเชิงวัตถุ และมีจำนวนคลาสการทำงานในเชิงธุรกิจอย่างน้อย 5 คลาส โดยกรณีศึกษาที่เลือกใช้ต้องสามารถนำมาปรับให้อยู่ในรูปแบบของโครงสร้างในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างในความสัมพันธ์แบบเจเนอรัลไลเซชันได้

2. เครื่องมือสำหรับจัดเก็บจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ ผู้วิจัยจะพัฒนาเป็นซอฟต์แวร์แพคเกจ (Package) สำหรับช่วยในการจัดเก็บค่าการเปลี่ยนแปลงคุณลักษณะของคลาสต่าง ๆ อย่างอัตโนมัติ เพื่อให้การคำนวณและการจัดเก็บค่า ถูกกระทำในสิ่งแวดล้อมและทรัพยากรเดียวกัน

3.3 สมมติฐานการวิจัย

จากวัตถุประสงค์ของงานวิจัย ผู้วิจัยต้องการ (1) เปรียบเทียบถึงประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล และ (2) การเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ระหว่างโครงสร้างของคลาสที่แตกต่างกัน ว่ามีผลกับคุณภาพของซอฟต์แวร์ในด้านประสิทธิภาพของซอฟต์แวร์ รวมถึงการแก้ไข ปรับปรุงและบำรุงรักษาซอฟต์แวร์อย่างไร โดยงานวิจัยนี้สามารถแบ่งการทดสอบสมมติฐานออกเป็น 2 ส่วน ดังนี้

3.3.1 สมมติฐานการเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล

จากวัตถุประสงค์ของงานวิจัย สามารถตั้งสมมติฐานสำหรับงานวิจัยภายใต้ขอบเขตของปัญหา (Problem Domain) ที่กำหนด ดังนี้คือ โครงสร้างของคลาสมีผลต่อประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลอย่างไร

กำหนดให้ P_x คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากจำนวนการส่งข้อความหรือเมสเสจ

P_y คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากจำนวนการส่งข้อความหรือเมสเสจ

T_x คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์

T_y คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์

จากตัวแปรที่กำหนด สามารถนำมาตั้งสมมติฐานเพื่อหาความสัมพันธ์ระหว่างตัวแปร เพื่อนำมาตอบวัตถุประสงค์ของงานวิจัยภายใต้ขอบเขตของปัญหาที่กำหนด ได้ดังนี้

1. เปรียบเทียบประสิทธิภาพของ P_x และประสิทธิภาพของ P_y

   H_0 : P_x และ P_y มีค่าไม่แตกต่างกัน

   H_1 : P_x และ P_y มีค่าแตกต่างกัน

2. เปรียบเทียบประสิทธิภาพของ T_x และประสิทธิภาพของ T_y

   H_0 : T_x และ T_y มีค่าไม่แตกต่างกัน

   H_1 : T_x และ T_y มีค่าแตกต่างกัน

3.3.2 สมมติฐานการเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์

จากวัตถุประสงค์ของงานวิจัย สามารถตั้งสมมติฐานสำหรับงานวิจัยภายใต้ขอบเขตของปัญหา (Problem Domain) ที่กำหนด ดังนี้คือ โครงสร้างของคลาสมีผลกระทบต่อการปรับปรุงแก้ไขและบำรุงรักษาซอฟต์แวร์อย่างไร

กำหนดให้ C_x คือ จำนวนคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน

C_y คือ จำนวนคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

M_x คือ จำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน

M_y คือ จำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

จากตัวแปรที่กำหนด สามารถนำมาตั้งสมมติฐานเพื่อหาความสัมพันธ์ระหว่างตัวแปร เพื่อนำมาตอบวัตถุประสงค์ของงานวิจัยภายใต้ขอบเขตของปัญหาที่กำหนด ได้ดังนี้

1. เปรียบเทียบค่าผลกระทบของ C_x และค่าผลกระทบ C_y

   H_0 : C_x และ C_y มีค่าไม่แตกต่างกัน

   H_1 : C_x และ C_y มีค่าแตกต่างกัน

2. เปรียบเทียบค่าผลกระทบของ M_x และค่าผลกระทบ M_y

   H_0 : M_x และ M_y มีค่าไม่แตกต่างกัน

   H_1 : M_x และ M_y มีค่าแตกต่างกัน

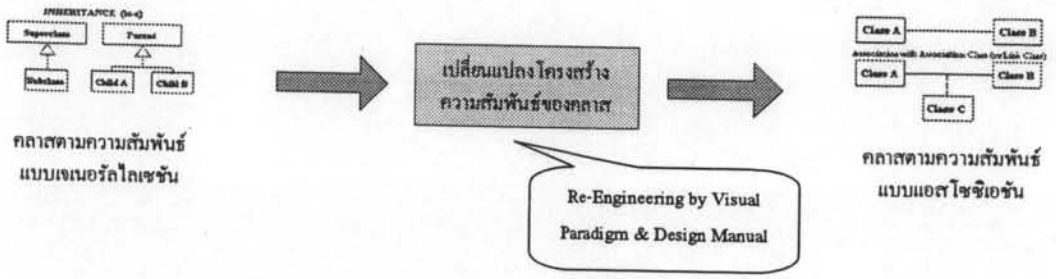
3.4 ขั้นตอนการเก็บรวบรวมข้อมูล (Data gathering execution)

งานวิจัยนี้ผู้วิจัยต้องการทราบถึงลักษณะของโครงสร้างของคลาสที่แตกต่างกันจะมีผลต่อ (1) ประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล และ (2) ผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ ว่ามีผลกับคุณภาพของซอฟต์แวร์ในด้านการแก้ไข ปรับปรุงและบำรุงรักษาซอฟต์แวร์อย่างไร โดยงานวิจัยนี้สามารถแบ่งขั้นตอนการเก็บรวบรวมข้อมูลออกเป็น 2 ขั้นตอน ดังนี้

3.4.1 การเก็บรวบรวมข้อมูลเพื่อเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล

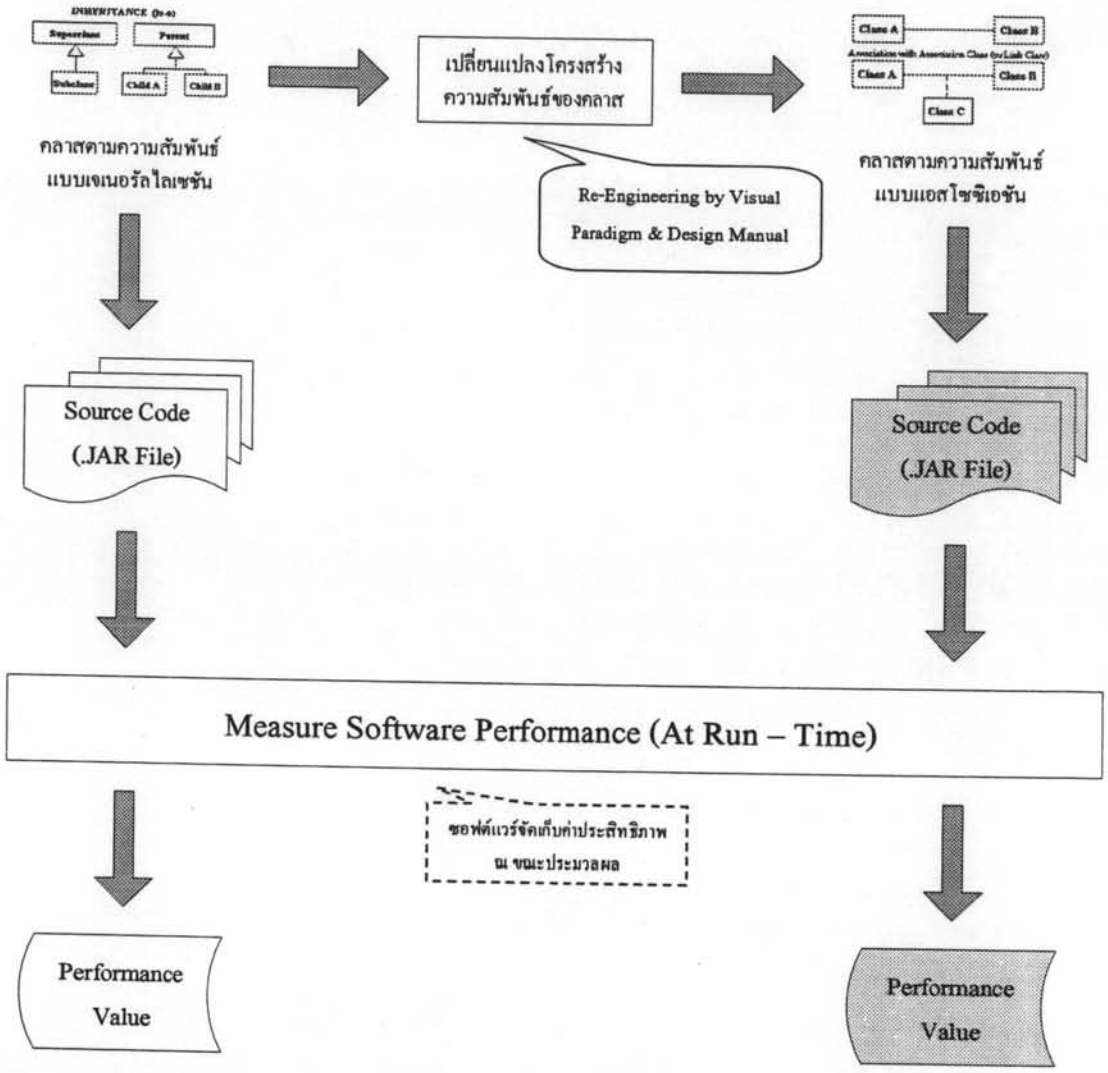
การเก็บรวบรวมข้อมูลในขั้นตอนนี้ จะพิจารณาประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล ระหว่างโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ซึ่งมีขั้นตอนการเก็บรวบรวมข้อมูล ดังนี้

1. เลือกหน่วยตัวอย่างที่จะใช้สำหรับงานวิจัย ซึ่งหน่วยตัวอย่างที่เลือกต้องมีข้อจำกัดดังที่กล่าวมาแล้วข้างต้น โดยโครงสร้างคลาสที่จะนำมาใช้ในงานวิจัย คือ โครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ซึ่งจะต้องมีความสัมพันธ์แบบที่มีการสืบทอดคุณสมบัติจากซูปเปอร์คลาสและมีซับคลาสที่รับการถ่ายทอดคุณสมบัติจากซูปเปอร์คลาสอย่างน้อยหนึ่งซับคลาสในโครงสร้างคลาสของหน่วยตัวอย่าง
2. จากโครงสร้างของคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ผู้วิจัยจะนำโครงสร้างความสัมพันธ์นั้นมาปรับเปลี่ยนเป็นโครงสร้างของคลาสในความสัมพันธ์แบบแอสโซซิเอชัน โดยอาศัยหลักการออกแบบคลาสไดอะแกรมดังที่กล่าวมาแล้วใน บทที่ 2 รวมทั้งใช้กระบวนการรีเวิร์สเอนจิเนียร์ริง (Reverse - Engineering) โดยใช้เครื่องมือวิสชวลแพระไดม์ (Visual Paradigm Tool) ดังรูปที่ 3-1



รูปที่ 3-1 การแปลงโครงสร้างคลาสแบบเจเนอรัลไลเซชันเป็นโครงสร้างคลาสแบบแอสโซซิเอชัน

3. จากโครงสร้างของคลาสในความสัมพันธ์แบบแอสโซซิเอชัน (X) และโครงสร้างของคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน (Y) จะนำมาพัฒนาเป็นซอฟต์แวร์ทางธุรกิจ (ซอร์สโค้ด) จากนั้นจะนำซอร์สโค้ดของทั้งสองโครงสร้างไปประมวลผล เพื่อวัดค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล โดยจะกระทำกับทุกหน่วยตัวอย่างที่ใช้ในงานวิจัย ดังรูปที่ 3-2 ซึ่งจะได้ค่าประสิทธิภาพของซอฟต์แวร์ ดังนี้
   - ประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน โดยคำนวณจากจำนวนการส่งข้อความหรือเมสเสจระหว่างคลาสทั้งหมดที่เป็นองค์ประกอบของซอฟต์แวร์ (P_x) และระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์ (T_x)
   - ประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน โดยคำนวณจากจำนวนการส่งข้อความหรือเมสเสจระหว่างคลาสทั้งหมดที่เป็นองค์ประกอบของซอฟต์แวร์ (P_y) และระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์ (T_y)



รูปที่ 3-2 การจัดเก็บค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล

โดยที่

คือ ซอร์สโค้ดจากกลุ่มหน่วยตัวอย่าง

คือ กระบวนการทำงานของเครื่องมือวัดประสิทธิภาพของซอฟต์แวร์

คือ ค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลที่คำนวณได้

คือ การแสดงทิศทางการทำงาน

4. เครื่องมือที่ใช้สำหรับหาค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล ผู้วิจัยจะพัฒนาเป็นซอฟต์แวร์ (Software) สำหรับช่วยในการอ่านการทำงานของซอฟต์แวร์หน่วยตัวอย่างและหาค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลอย่างอัตโนมัติ โดยรายละเอียดการออกแบบและพัฒนาเครื่องมือผู้วิจัยจะกล่าวถึงในบทที่ 4 ซึ่งความต้องการของซอฟต์แวร์ที่ใช้สำหรับวัดประสิทธิภาพของซอฟต์แวร์ ผู้วิจัยได้กำหนดความต้องการ โดยพิจารณาจากความเป็นไปได้ในการปฏิบัติการ (Operational Feasibility) คือ ความเป็นไปได้ของซอฟต์แวร์ที่ถูกต้องตรงความต้องการของผู้ใช้และพิจารณาถึงความเป็นไปได้ในด้านการจัดการ (Management) โดยที่ความต้องการของซอฟต์แวร์จะต้องช่วยสนับสนุนให้สามารถทำงานได้อย่างมีประสิทธิภาพ และเป็นทางเลือกที่เป็นไปได้ในการแก้ปัญหา ให้สำเร็จตามวัตถุประสงค์ที่ต้องการ (Sommerville, 2001) จากวัตถุประสงค์ของงานวิจัย ต้องการเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล ดังนั้น ความต้องการของซอฟต์แวร์ (Software Requirement Specification) ที่จะพัฒนา ประกอบด้วย

ความต้องการเชิงฟังก์ชัน (Functional Requirement)

- ซอฟต์แวร์สามารถอ่านซอร์สโค้ดจากหน่วยตัวอย่างที่จะใช้ในงานวิจัย เพื่อเป็นข้อมูลนำเข้าได้
- ซอฟต์แวร์สามารถอ่านกระบวนการทำงานของหน่วยตัวอย่างได้
- สามารถวัดประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลซอฟต์แวร์ได้
- ซอฟต์แวร์สามารถรายงานประสิทธิภาพของของซอฟต์แวร์ขณะประมวลผล โดยสามารถแสดงผลออกทางหน้าจอคอมพิวเตอร์ได้

ความต้องการเชิงนอนฟังก์ชัน (Non – Functional Requirement)

- ซอฟต์แวร์สามารถอ่านกระบวนการทำงานของซอร์สโค้ดหน่วยตัวอย่าง และวัดค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลได้อย่างถูกต้องและแม่นยำ
- ซอฟต์แวร์สามารถทำงานได้อย่างมีประสิทธิภาพและรวดเร็ว
- ซอฟต์แวร์สามารถใช้งานได้ง่าย

ข้อกำหนดของซอฟต์แวร์ (System Requirement)

- ซอฟต์แวร์นี้ต้องพัฒนาภายใต้ภาษาโปรแกรมเดียวกับหน่วยตัวอย่างที่ใช้ใน โดยหน่วยตัวอย่างที่เลือกใช้ในงานวิจัยนี้ ผู้วิจัยได้พัฒนาซอร์สโค้ดหน่วยตัวอย่างภายใต้โปรแกรมภาษาจาวา (Java Programming Language) ดังนั้นเครื่องมือ

สำหรับวัดประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลจะถูกพัฒนาภายใต้โปรแกรมภาษาจาวาเช่นกัน

3.4.2 การเก็บรวบรวมข้อมูลเพื่อเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์

การเก็บรวบรวมข้อมูลในขั้นตอนนี้ ผู้วิจัยจะพิจารณาผลกระทบที่เกิดจากการเปลี่ยนแปลงซอฟต์แวร์ ระหว่างโครงสร้างของคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างของคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ซึ่งมีขั้นตอนการเก็บรวบรวมข้อมูล ดังนี้

1. เลือกซอฟต์แวร์กรณีศึกษาหนึ่งกรณีศึกษาสำหรับใช้ในงานวิจัย โดยกำหนดให้เป็นซอฟต์แวร์ทางธุรกิจ ที่พัฒนาด้วยหลักการเชิงวัตถุ
2. นำซอฟต์แวร์กรณีศึกษาที่เลือกมาปรับให้อยู่ในรูปแบบของซอฟต์แวร์ที่พัฒนาภายใต้โครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน
3. กำหนดหน่วยตัวอย่างเพื่อเปลี่ยนแปลงซอฟต์แวร์กรณีศึกษา โดยงานวิจัยนี้กำหนดให้หน่วยตัวอย่างคือ ความต้องการเชิงฟังก์ชัน (Functional Requirement) เพื่อใช้ในการพิจารณาจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ ซึ่งความต้องการที่กำหนดจะอยู่ภายใต้ลักษณะการเปลี่ยนแปลงของซอฟต์แวร์เชิงวัตถุ โดยจะพิจารณาการเปลี่ยนแปลงที่เกิดขึ้นที่คลาส เมธธอดและแอททริบิวต์ โดยการเปลี่ยนแปลงที่เกิดขึ้นจะต้องเป็นการเปลี่ยนแปลงแบบสถิตย์และทำให้มีผลกระทบกับการเปลี่ยนแปลงแบบวากยสัมพันธ์ของซอฟต์แวร์ (Li, 1996) ซึ่งรูปแบบการเปลี่ยนแปลงจะอยู่ภายใต้ขอบเขตการเปลี่ยนแปลง ดังตารางที่ 3-1

ตารางที่ 3 – 1 รูปแบบของการเปลี่ยนแปลงที่คลาส แอททริบิวต์และเมธธอด

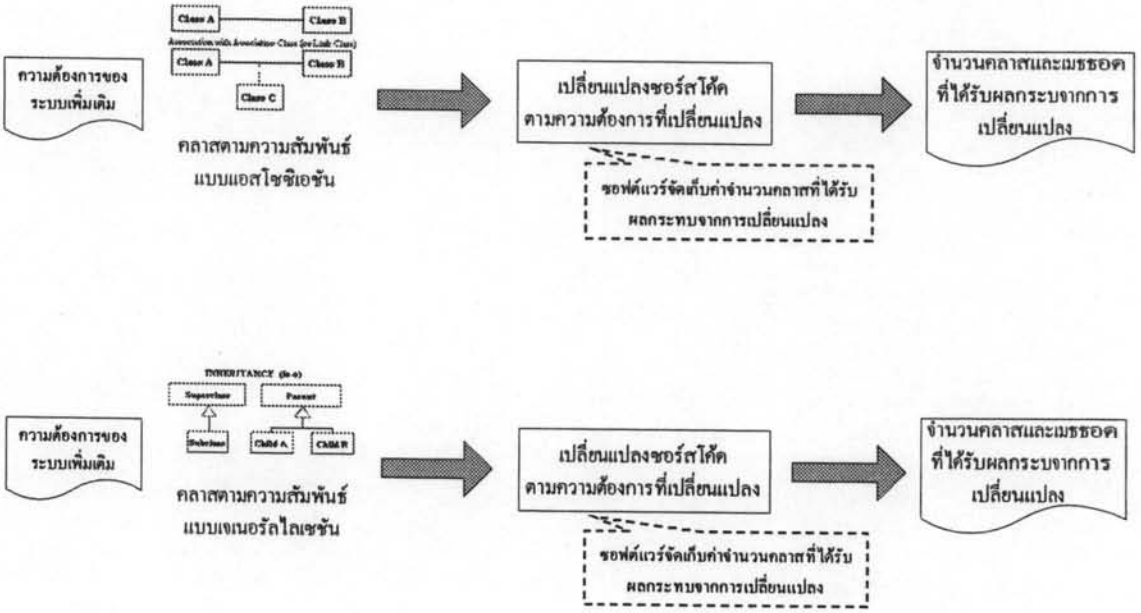
ลักษณะการเปลี่ยนแปลง	รูปแบบการเปลี่ยนแปลง
การเปลี่ยนแปลงที่ความสัมพันธ์ระหว่างคลาส	การเพิ่มการสืบทอดคุณสมบัติ
	การลบการสืบทอดคุณสมบัติ
การเปลี่ยนแปลงที่คลาส	การเพิ่มคลาส
	การลบคลาส
	การเปลี่ยนแปลงขอบเขตของคลาส

ตารางที่ 3 – 1 รูปแบบของการเปลี่ยนแปลงที่คลาส แอททริบิวต์และเมธธอด (ต่อ)

ลักษณะการเปลี่ยนแปลง	รูปแบบการเปลี่ยนแปลง
การเปลี่ยนแปลงที่แอททริบิวต์	การเพิ่มแอททริบิวต์
	การลบแอททริบิวต์
	การเปลี่ยนแปลงขอบเขตของแอททริบิวต์
การเปลี่ยนแปลงที่เมธธอด	การเพิ่มเมธธอด
	การลบเมธธอด
	การเปลี่ยนแปลงขอบเขตของเมธธอด

โดยการกำหนดหน่วยตัวอย่างที่เป็นความต้องการเชิงฟังก์ชันของซอฟต์แวร์ จะต้องกำหนดหน่วยตัวอย่างให้อยู่ภายใต้ขอบเขตของการเปลี่ยนแปลงคุณลักษณะของโครงสร้างคลาสดังตารางที่ 3-1

4. เมื่อตรวจสอบความต้องการเชิงฟังก์ชันเรียบร้อยแล้ว ผู้วิจัยจะแก้ไขโครงสร้างคลาสในความสัมพันธ์ของคลาสแบบแอสโซซิเอชันและความสัมพันธ์แบบเจเนอรัลไลเซชัน โดยการออกแบบความต้องการเชิงฟังก์ชันที่เพิ่มเข้ามาในซอฟต์แวร์กรณีศึกษา
5. ทำการทดลองโดยการปรับปรุง แก้ไขซอฟต์แวร์กรณีศึกษา ตามหน่วยตัวอย่างที่กำหนด เพื่อใช้ในการเปลี่ยนแปลงซอฟต์แวร์ในรูปของความต้องการเชิงฟังก์ชัน (เพื่อให้ซอฟต์แวร์กรณีศึกษามีการเปลี่ยนแปลง) โดยมีหน่วยตัวอย่างจำนวน 30 หน่วยตัวอย่าง โดยจะกระทำกับทุกหน่วยตัวอย่างที่ใช้ในงานวิจัย ดังรูปที่ 3-3 ซึ่งจะได้ค่าผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ ดังนี้
   - ค่าผลกระทบที่เกิดจากการเปลี่ยนแปลงคุณลักษณะของโครงสร้างคลาสตามความสัมพันธ์แบบแอสโซซิเอชัน จากความต้องการของซอฟต์แวร์ที่เปลี่ยนแปลงไป โดยคำนวณจากจำนวนคลาส (C_x) และจำนวนเมธธอด (M_x) ที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชัน
   - ค่าผลกระทบที่เกิดจากการเปลี่ยนแปลงคุณลักษณะของโครงสร้างคลาสตามความสัมพันธ์แบบเจเนอรัลไลเซชัน จากความต้องการของซอฟต์แวร์ที่เปลี่ยนแปลงไป โดยคำนวณจากจำนวนคลาส (C_y) และจำนวนเมธธอด (M_y) ที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชัน



รูปที่ 3-3 การจัดเก็บจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลง

6. เครื่องมือที่ใช้สำหรับจัดเก็บจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ ผู้วิจัยจะพัฒนาเป็นซอฟต์แวร์แพคเกจ (Software Package) สำหรับช่วยในการจัดเก็บค่าการเปลี่ยนแปลงต่าง ๆ อย่างอัตโนมัติ โดยรายละเอียดการออกแบบและพัฒนาเครื่องมือผู้วิจัยจะกล่าวถึงในบทที่ 4 ซึ่งความต้องการของซอฟต์แวร์ที่ใช้สำหรับวัดผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ ผู้วิจัยได้กำหนดความต้องการโดยพิจารณาจากความเป็นไปได้ในการปฏิบัติการ (Operational Feasibility) คือ ความเป็นไปได้ของซอฟต์แวร์ที่ถูกต้องตรงความต้องการของผู้ใช้ และพิจารณาถึงความเป็นไปได้ในด้านการจัดการ (Management) โดยที่ความต้องการของซอฟต์แวร์จะต้องช่วยสนับสนุนให้สามารถทำงานได้อย่างมีประสิทธิภาพและเป็นทางเลือกที่เป็นไปได้ในการแก้ปัญหา ให้สำเร็จตามวัตถุประสงค์ที่ต้องการ (Sommerville, 2001) จากวัตถุประสงค์ของงานวิจัย ต้องการเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ ดังนั้น ความต้องการของซอฟต์แวร์ (Software Requirement Specification) ที่จะพัฒนา ประกอบด้วย

ความต้องการเชิงฟังก์ชัน (Functional Requirement)

- ซอฟต์แวร์แพคเกจสามารถจัดเก็บจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงได้
- ซอฟต์แวร์แพคเกจสามารถรายงานผลกระทบจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงได้ โดยสามารถแสดงผลออกทางฐานข้อมูล (Database) ที่สร้างขึ้น

ความต้องการเชิงนอนฟังก์ชัน (Non – Functional Requirement)

- ซอฟต์แวร์แพคเกจสามารถจัดเก็บจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงได้อย่างถูกต้องและแม่นยำ
- ซอฟต์แวร์แพคเกจสามารถทำงานได้อย่างมีประสิทธิภาพและรวดเร็ว
- ซอฟต์แวร์แพคเกจสามารถใช้งานได้ง่าย

ข้อกำหนดของซอฟต์แวร์ (System Requirement)

- ซอฟต์แวร์แพคเกจนี้ต้องพัฒนาภายใต้ภาษาโปรแกรมเดียวกับซอฟต์แวร์กรณีศึกษาที่ใช้ในงานวิจัย และจะถูกติดตั้งเพิ่ม (Add-in) เข้าไปกับเครื่องมือที่ใช้ในการพัฒนาซอฟต์แวร์ โดยซอฟต์แวร์กรณีศึกษาที่เลือกใช้ในงานวิจัยนี้ ผู้วิจัยได้พัฒนาภายใต้โปรแกรมภาษาจาวา (Java Programming Language) ดังนั้นเครื่องมือสำหรับจัดเก็บค่าผลกระทบที่เกิดจากการเปลี่ยนแปลงจะถูกพัฒนาภายใต้โปรแกรมภาษาจาวาเช่นกัน

3.5 กรอบของการวิเคราะห์ข้อมูล (Data analysis framework)

งานวิจัยนี้จะเก็บรวบรวมข้อมูล (1) เพื่อเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล และ (2) เพื่อเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ ดังนั้น สามารถพิจารณากรอบการวิเคราะห์ข้อมูล ได้เป็น 2 ส่วน ดังนี้

3.5.1 กรอบการวิเคราะห์ข้อมูลเพื่อเปรียบเทียบประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล

จากนิยามประสิทธิภาพของซอฟต์แวร์ที่กล่าวมาแล้วในบทที่ 2 ผู้วิจัยสามารถกำหนดนิยามของประสิทธิภาพของโครงสร้างคลาสได้ดังนี้ (1) เมื่อนำโครงสร้างคลาสไปสร้างซอฟต์แวร์ แล้วได้ซอฟต์แวร์ที่มีความซับซ้อนต่ำ โครงสร้างคลาสนั้นจะมีประสิทธิภาพที่ดี ซึ่งสามารถวัดได้จากจำนวนการส่งข้อความหรือเมสเสจ (Message Calling) ระหว่างคลาสทั้งหมดที่เป็นองค์ประกอบของซอฟต์แวร์ และ (2) เมื่อประมวลผลการทำงานของซอฟต์แวร์ ในการทำงานที่ทรานแซคชัน

เดียวกันและได้ผลลัพธ์ของการประมวลผลทรานแซคชันเหมือนกัน โครงสร้างคลาสที่ดีจะต้องได้ระยะเวลาในการทำงานที่เร็วกว่า ซึ่งสามารถวัดได้จากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์ (Response Time)

การทดสอบสมมติฐานของการเปรียบเทียบประสิทธิภาพของโครงสร้างคลาส โดยการวิเคราะห์ว่าโครงสร้างของคลาสที่ต่างกันมีผลต่อประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลหรือไม่ เพื่อตอบวัตถุประสงค์ที่ตั้งไว้ โดยกำหนดข้อมูลที่จะนำมาวิเคราะห์ ดังนี้

กำหนดให้ P_x คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากจำนวนการส่งข้อความหรือเมสเสจ

P_y คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากจำนวนการส่งข้อความหรือเมสเสจ

T_x คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์

T_y คือ ประสิทธิภาพของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของซอฟต์แวร์ขณะประมวลผล โดยวัดจากระยะเวลาการทำงานที่ซอฟต์แวร์ถูกประมวลผลโดยสมบูรณ์

ซึ่งผู้วิจัยคาดหวังว่าโครงสร้างของคลาสที่ออกแบบต่างกัน จะมีผลต่อประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลต่างกัน โดยโครงสร้างของคลาสที่มีจำนวนครั้งของการส่งข้อความหรือเมสเสจระหว่างคลาสที่ทำงานร่วมกันน้อย โครงสร้างของคลาสนั้นจะมีประสิทธิภาพสูง ในทางตรงกันข้ามถ้าโครงสร้างของคลาสที่มีจำนวนครั้งของการส่งข้อความหรือเมสเสจระหว่างคลาสที่ทำงานร่วมกันมาก โครงสร้างของคลาสนั้นจะมีประสิทธิภาพต่ำ และระยะเวลาการทำงานของซอฟต์แวร์ขณะประมวลผล โครงสร้างคลาสที่มีประสิทธิภาพดีกว่าจะต้องได้ระยะเวลาในการทำงานของซอฟต์แวร์ขณะประมวลที่เร็วกว่า

ดังนั้น ค่า P และ T จึงเป็นค่าที่คำนวณได้จากประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันของแต่ละหน่วยตัวอย่างที่เลือกมาทำการทดลอง ซึ่งจะได้เป็นกลุ่มข้อมูลสำหรับนำไปวิเคราะห์ผลต่อไป

โดยผู้วิจัยจะนำผลต่างของค่า P และ T ระหว่างโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันและโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันมาตรวจสอบการแจกแจงของข้อมูลที่ได้ว่าเป็นการแจกแจงปกติหรือไม่ (Normal Distribution) ถ้าเป็นการแจกแจงแบบปกติจะใช้การทดสอบแบบใช้พารามิเตอร์ (Parametric Test) ซึ่งใช้วิธีการทดสอบสมมติฐานเกี่ยวกับผลต่างระหว่างค่าเฉลี่ย 2 ประชากรแบบจับคู่ (Paired t-test)

แต่ถ้าตรวจสอบแล้วว่าการแจกแจงของข้อมูลที่ได้ไม่เป็นการแจกแจงปกติจะใช้การทดสอบสมมติฐานแบบไม่อิงกับพารามิเตอร์ (Nonparametric Test) ซึ่งผู้วิจัยเลือกใช้วิธีการทดสอบสมมติฐานเกี่ยวกับการแจกแจงของข้อมูล 2 ชุดที่สัมพันธ์กัน (Two-Related-Samples Tests) โดยเลือกใช้วิธีการทดสอบประเภทวิลคอกสัน ไซน์-แรงค์ เทส (Wilcoxon signed-rank test)

3.5.2 กรอบการวิเคราะห์ข้อมูลเพื่อเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์

งานวิจัยนี้ได้เก็บรวบรวมข้อมูลจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของซอฟต์แวร์ ซึ่งขอบเขตของการเปลี่ยนแปลงคุณลักษณะของโครงสร้างคลาสจะเปลี่ยนแปลงตามขอบเขตคุณลักษณะของคลาสดังตารางที่ 3-1 โดยผลกระทบจากการเปลี่ยนแปลงนี้จะพิจารณาจากจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงหน่วยตัวอย่างที่เป็นความต้องการเชิงฟังก์ชัน โดยกำหนดกรอบการวิเคราะห์ข้อมูล ได้ดังนี้

การทดสอบสมมติฐานของการเปรียบเทียบผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของซอฟต์แวร์ โดยการวิเคราะห์ว่าโครงสร้างของคลาสที่ต่างกันมีผลต่อการแก้ไขปรับปรุงและการบำรุงรักษาซอฟต์แวร์ไม่ เพื่อตอบวัตถุประสงค์ที่ตั้งไว้ โดยกำหนดข้อมูลที่จะนำมาวิเคราะห์ ดังนี้

กำหนดให้ C_x คือ จำนวนคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน

C_y คือ จำนวนคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

M_x คือ จำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน

M_y คือ จำนวนเมธธอดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการเชิงฟังก์ชันของโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน

จากงานวิจัยของ Daly และคณะ (1995) พบว่า ซอฟต์แวร์ที่มีการสืบทอดคุณสมบัติที่ระดับความลึก 3 ระดับมีความสามารถในการปรับปรุงและแก้ไขซอฟต์แวร์ระหว่างการอิมพลีเมนต์ได้รวดเร็วกว่าซอฟต์แวร์ที่ไม่มีระดับความลึกในการสืบทอดคุณสมบัติ ดังนั้น ผู้วิจัยคาดหวังว่าโครงสร้างของคลาสที่ออกแบบต่างกัน จะมีผลต่อการแก้ไข ปรับปรุงและบำรุงรักษาซอฟต์แวร์ต่างกัน โดยโครงสร้างของคลาสที่ให้ผลกระทบในการปรับเปลี่ยนความต้องการของซอฟต์แวร์น้อย น่าจะมีความสามารถในการรองรับการเปลี่ยนแปลงโครงสร้างของซอฟต์แวร์ทางธุรกิจได้ดี และทำให้ทราบว่าโครงสร้างและความสัมพันธ์ของคลาสแบบใดที่รองรับการเปลี่ยนแปลงได้ดี

ดังนั้น ค่า C และ M จึงเป็นค่าที่คำนวณได้จากผลกระทบจากการเปลี่ยนแปลงซอฟต์แวร์ตามความต้องการเชิงฟังก์ชัน (หน่วยตัวอย่าง) ของซอฟต์แวร์กรณีศึกษาของโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันและโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ซึ่งจะได้เป็นกลุ่มข้อมูลสำหรับนำไปวิเคราะห์ผลต่อไป

โดยผู้วิจัยจะนำผลต่างของค่า C และ M ระหว่างโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันและโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชันมาตรวจสอบการแจกแจงของข้อมูลที่ได้ว่าเป็นการแจกแจงปกติหรือไม่ (Normal Distribution) ถ้าเป็นการแจกแจงแบบปกติจะใช้การทดสอบแบบใช้พารามิเตอร์ (Parametric Test) ซึ่งใช้วิธีการทดสอบสมมติฐานเกี่ยวกับผลต่างระหว่างค่าเฉลี่ย 2 ประชากรแบบจับคู่ (Paired t-test)

แต่ถ้าตรวจสอบแล้วว่าการแจกแจงของข้อมูลที่ได้ไม่เป็นการแจกแจงปกติจะใช้การทดสอบสมมติฐานแบบไม่อิงกับพารามิเตอร์ (Nonparametric Test) ซึ่งผู้วิจัยเลือกใช้วิธีการทดสอบสมมติฐานเกี่ยวกับการแจกแจงของข้อมูล 2 ชุดที่สัมพันธ์กัน (Two-Related-Samples Tests) โดยเลือกใช้วิธีการทดสอบประเภทวิลคอกสัน ไซน์-แรงค์ เทส (Wilcoxon signed-rank test)

จากกรอบการวิเคราะห์ข้อมูล (1) ประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล และ (2) ผลกระทบจากการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ ผู้วิจัยจะนำผลลัพธ์ที่ได้ มาพิจารณาว่าโครงสร้างของคลาสมีอิทธิพลต่อการปรับปรุง บำรุงรักษาซอฟต์แวร์และประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลอย่างไร

3.6 ประเด็นของความเชื่อถือได้ (Reliability) และความถูกต้อง (Validity)

การตอบวัตถุประสงค์ให้มีความถูกต้องและแม่นยำ ต้องควบคุมปัจจัยที่มีผลเกี่ยวข้องกับงานวิจัย ได้แก่ เครื่องมือสำหรับอ่านกระบวนการทำงานของซอฟต์แวร์หน่วยตัวอย่างและหาค่าประสิทธิภาพของซอฟต์แวร์ขณะประมวลผลและซอร์สโค้ดที่นำมาใช้ในการทดลองต้องถูกพัฒนาด้วยภาษาโปรแกรมเดียวกัน เนื่องจากงานวิจัยต้องการเปรียบเทียบ (1) ประสิทธิภาพของซอฟต์แวร์ขณะประมวลผล และ (2) ผลกระทบจากการเปลี่ยนแปลงของซอฟต์แวร์ ว่ามีผลกับคุณภาพของซอต์แวร์ในด้านของการแก้ไข ปรับปรุง และบำรุงรักษาซอฟต์แวร์ ซึ่งต้องมีการควบคุมปัจจัย ดังนี้

1. การเลือกหน่วยตัวอย่างที่นำมาใช้ทดสอบประสิทธิภาพของโครงสร้างคลาส ต้องเป็นโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชัน ที่มีการสืบทอดคุณสมบัติ จากซูปเปอร์คลาสและมีซับคลาสที่รับการถ่ายทอดคุณสมบัติจากซูปเปอร์คลาสอย่างน้อยหนึ่งซับคลาสในโครงสร้างคลาส โดยผู้วิจัยเลือกหน่วยตัวอย่างที่มีความหลากหลายทางธุรกิจและมีจำนวนคลาสการทำงานในเชิงธุรกิจอย่างน้อย 5 คลาส และหน่วยตัวอย่างที่นำมาใช้ทดสอบผลกระทบจากการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ต้องเป็นโครงสร้างคลาสในความสัมพันธ์แบบแอสโซซิเอชัน ที่สามารถปรับเปลี่ยนให้เป็นโครงสร้างคลาสในความสัมพันธ์แบบเจเนอรัลไลเซชันได้
2. การนำหน่วยตัวอย่างมาพัฒนาเป็นซอฟต์แวร์ซอร์สโค้ด จะพัฒนาภายใต้ภาษาโปรแกรมเดียวกัน และการพิจารณาการเปลี่ยนแปลงซอฟต์แวร์ตามความต้องการเชิงฟังก์ชันของซอฟต์แวร์ที่เพิ่มขึ้น จะพิจารณาที่ซอร์สโค้ดของซอฟต์แวร์ โดยการเปลี่ยนแปลงจะกระทำครั้งละหนึ่งความต้องการเชิงฟังก์ชันเท่านั้น
3. เครื่องมือที่พัฒนาขึ้นเพื่อใช้สำหรับอ่านกระบวนการทำงานของซอฟต์แวร์ และวัดประสิทธิภาพของซอฟต์แวร์ รวมทั้งเครื่องมือที่ใช้สำหรับวัดจำนวนคลาสและจำนวนเมธธอดที่ได้รับผลกระทบที่เกิดจากการเปลี่ยนแปลงความต้องการของซอฟต์แวร์ จะพัฒนาภายใต้ภาษาโปรแกรมเดียวกันกับหน่วยตัวอย่างที่นำใช้ในงานวิจัย