บทที่ 3
ระเบียบวิธีวิจัย

3.1 บทนำ

ในบทนี้จะกล่าวถึงระเบียบวิธีวิจัยซึ่งประกอบด้วยการอธิบายแผนแบบการทดลอง (Experimental Design) การทดสอบสมมติฐาน การทำงานของเครื่องมือทดสอบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบต่างๆ ที่งานวิจัยกำหนด รวมทั้งวิธีการพัฒนาเครื่องมือทดสอบประสิทธิภาพของการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ ประเด็นความน่าเชื่อถือได้ (Reliability) ความถูกต้อง (Validity) และกรอบการวิเคราะห์ข้อมูล (Data Analysis Framework) ดังรายละเอียดต่อไปนี้

3.2 แผนแบบการทดลอง

งานวิจัยนี้มีวัตถุประสงค์ในการทดลองเพื่อศึกษาเปรียบเทียบประสิทธิภาพการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่น (Support-Confidence Model) ดั้งเดิมกับการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) ในสถานการณ์ของการให้คำแนะนำนักพัฒนาต่างๆกัน 3 สถานการณ์ได้แก่ 1) สถานการณ์การนำทาง (Navigation) คือ สถานการณ์ที่นักพัฒนามีการเปลี่ยนแปลงแก้ไขที่เอนทิตี้หนึ่งแล้ว ระบบจะให้คำแนะนำกับนักพัฒนาให้แก้ไขเอนทิตี้ใดต่อไปได้ถูกต้องหรือไม่ 2) สถานการณ์การป้องกันการเกิดข้อผิดพลาด (Error Prevention) คือ สถานการณ์ที่นักพัฒนามีการเปลี่ยนแปลงแก้ไขที่เอนทิตี้หลายๆเอนทิตี้ต่อเนื่องกันแต่ยังขาดการเปลี่ยนแปลงแก้ไขเอนทิตี้อีกหนึ่งเอนทิตี้จึงจะสมบูรณ์ ระบบจะให้คำแนะนำกับนักพัฒนาให้แก้ไขเอนทิตี้ที่เหลือนั้นได้ถูกต้องหรือไม่ 3) สถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว (Closure) คือ สถานการณ์ที่นักพัฒนามีการเปลี่ยนแปลงแก้ไขที่เอนทิตี้หลายๆเอนทิตี้ต่อเนื่องกันจนสมบูรณ์แล้ว ระบบจะให้คำแนะนำที่เป็นผลบวกลวง (False Positive) ออกมาแก่นักพัฒนาหรือไม่

จากวัตถุประสงค์งานวิจัยที่กล่าวข้างต้น ผู้วิจัยจึงเลือกใช้แผนแบบการทดลองแบบการเปรียบเทียบกลุ่มสถิต (Static Group Comparison) ซึ่งเป็นแผนแบบการทดลองที่เหมาะสมกับการทดลองที่ต้องการวัดค่าตัวแปรตามของกลุ่มควบคุมและกลุ่มทดลองหลังจากทำการทดสอบมีค่าแตกต่างกันอย่างไร นั่นคือ การวัดค่าประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นดั้งเดิม (กลุ่มควบคุม) กับค่าประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) (กลุ่มทดสอบ) หลังจากการทดสอบว่ามีค่าแตกต่างกันอย่างไร โดยกำหนดให้มีตัวแปรในการทดลองเปรียบเทียบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบที่ต้องการทดสอบ ดังต่อไปนี้

3.2.1 ตัวแปรต้น

งานวิจัยนี้สนใจว่าการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) สามารถเพิ่มประสิทธิภาพของระบบให้คำแนะนำนักพัฒนาในระหว่างการพัฒนาซอฟต์แวร์ของไอดีอี (IDE: Integrated Development Environment) ได้หรือไม่ ดังนั้นตัวแปรต้นของการศึกษาครั้งนี้คือตัวแบบของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์ทั้งหมด 2 ตัวแบบ ดังนี้

1) การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่น (Support-Confidence Model)
2) การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Support-New Confidence Model) (Liu et al., 2008)

จากการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ทั้ง 2 ตัวแบบข้างต้น ผู้วิจัยจะเรียกการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่น ด้วยคำว่า "การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1" ส่วนการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎ

ความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) จะเรียกว่า "การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2"

3.2.2 ตัวแปรตาม

เนื่องจากงานวิจัยนี้สนใจเปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ 2 ตัวแบบดังกล่าวข้างต้น ดังนั้นการเปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์จะพิจารณาจากความถูกต้องแม่นยำในการทำนายและให้คำแนะนำกับนักพัฒนาในระหว่างการพัฒนาซอฟต์แวร์มากน้อยเพียงใด โดยการวัดประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ในแต่ละสถานการณ์มีวิธีการในการประเมินที่แตกต่างกันออกไปดังนี้

- ในสถานการณ์ *การนำทาง (Navigation)* สามารถประเมินประสิทธิภาพจากค่าเอฟเมสเซอร์ (F-measure) ที่คำนวณมาจากค่าความถูกต้อง (Precision) และค่าเรียกคืน (Recall) ซึ่งรายละเอียดและวิธีการคำนวณค่าเอฟเมสเซอร์ ค่าความถูกต้องและค่าเรียกคืนสำหรับการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์นั้นได้กล่าวเอาไว้ในบทที่ 2
- ในสถานการณ์ *การป้องกันการเกิดข้อผิดพลาด (Error Prevention)* สามารถประเมินประสิทธิภาพจากค่าเอฟเมสเซอร์ (F-measure) ที่คำนวณมาจากค่าความถูกต้อง (Precision) และค่าเรียกคืน (Recall) ซึ่งรายละเอียดและวิธีการคำนวณค่าเอฟเมสเซอร์ ค่าความถูกต้องและค่าเรียกคืนสำหรับการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์นั้นได้กล่าวเอาไว้ในบทที่ 2
- ในสถานการณ์ *การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว (Closure)* สามารถประเมินประสิทธิภาพจากค่าผลสะท้อนกลับ (Feedback) ซึ่งรายละเอียดและวิธีการคำนวณค่าผลสะท้อนกลับสำหรับการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์นั้นได้กล่าวเอาไว้ในบทที่ 2

3.3 สมมติฐานงานวิจัย

จากวัตถุประสงค์ของงานวิจัย ผู้วิจัยต้องการทดสอบว่าการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) สามารถเพิ่มประสิทธิภาพของระบบให้คำแนะนำนักพัฒนาในสถานการณ์ต่างๆกัน 3 สถานการณ์คือ สถานการณ์การนำทาง (Navigation) สถานการณ์การป้องกันการเกิดข้อผิดพลาด (Error Prevention) และสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว (Closure) ได้หรือไม่ ดังนั้นงานวิจัยนี้จึงต้องการศึกษาประสิทธิภาพของการให้คำแนะนำนักพัฒนาในระหว่างการพัฒนาซอฟต์แวร์ที่ได้มาจากการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ทั้ง 2 ตัวแบบตามที่กำหนดไว้ในหัวข้อตัวแปรต้น โดยผู้วิจัยจะตั้งสมมติฐานในแต่ละสถานการณ์จะที่ทำการทดสอบไว้ดังนี้

1) วิเคราะห์เปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ทั้ง 2 ตัวแบบในสถานการณ์การนำทาง ว่ามีความแตกต่างกันหรือไม่

กำหนดให้ μ_1 คือ ค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การนำทาง

μ_2 คือ ค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ในสถานการณ์การนำทาง

ผู้วิจัยต้องการทราบว่าค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบใดที่มีค่ามากกว่ากัน ดังนั้นผู้วิจัยจึงเปรียบเทียบค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 กับค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ผู้วิจัยเห็นว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นสามารถให้เกิดผลลัพธ์ของการทำเหมืองข้อมูลที่เป็นผลบวกลวง (False Positive) เป็นจำนวนมาก โดยเฉพาะอย่างยิ่งกับข้อมูลซอฟต์แวร์อาร์ไคฟ์ (Li et al., 2005) ซึ่งน่าจะเป็นผลมาจากการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นจะให้ผลลัพธ์ที่ไม่สอดคล้องกับสหสัมพันธ์ (Correlation) แต่การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นให้ผลลัพธ์ของการทำเหมืองข้อมูลที่สอดคล้องกับสหสัมพันธ์ด้วย (Liu et al., 2008) ดังนั้นผู้วิจัยจึงคาดว่าการ

ค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นมีประสิทธิภาพที่ดีกว่าการการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การนำทาง จึงตั้งสมมติฐานไว้ ดังนี้

$H_0 : \mu_2 \leq \mu_1$

$H_1 : \mu_2 > \mu_1$

การยอมรับ H_0 หมายถึง การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ในสถานการณ์การนำทาง

การปฏิเสธ H_0 หมายถึง การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การนำทาง

2) วิเคราะห์เปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ทั้ง 2 ตัวแบบในสถานการณ์การป้องกันการเกิดข้อผิดพลาด ว่ามีความแตกต่างกันหรือไม่

กำหนดให้ μ_1 คือ ค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การป้องกันการเกิดข้อผิดพลาด

μ_2 คือ ค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ในสถานการณ์การป้องกันการเกิดข้อผิดพลาด

ผู้วิจัยต้องการทราบว่าค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบใดที่มีค่ามากกว่ากัน ดังนั้นผู้วิจัยจึงเปรียบเทียบค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 กับค่าเอฟเมสเซอร์ของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ผู้วิจัยเห็นว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นสามารถให้เกิดผลลัพธ์ของการทำเหมืองข้อมูลที่เป็นผลบวกลวง (False Positive) เป็นจำนวนมาก โดยเฉพาะอย่างยิ่งกับข้อมูลซอฟต์แวร์อาร์ไคฟ์ (Li et al., 2005) ซึ่งน่าจะเป็นผลมาจากการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นจะให้ผลลัพธ์ที่ไม่สอดคล้องกับสหสัมพันธ์ (Correlation) แต่การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นให้ผลลัพธ์ของการทำ

เหมืองข้อมูลที่สอดคล้องกับสหสัมพันธ์ด้วย (Liu et al., 2008) ดังนั้นผู้วิจัยจึงคาดว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การป้องกันการเกิดข้อผิดพลาด จึงตั้งสมมติฐานไว้ ดังนี้

$H_0 : \mu_2 \leq \mu_1$

$H_1 : \mu_2 > \mu_1$

การยอมรับ H_0 หมายถึง การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ในสถานการณ์การป้องกันการเกิดข้อผิดพลาด

การปฏิเสธ H_0 หมายถึง การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การป้องกันการเกิดข้อผิดพลาด

3) วิเคราะห์เปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว ว่ามีความแตกต่างกันหรือไม่

กำหนดให้ μ_1 คือ ค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว

μ_2 คือ ค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว

ผู้วิจัยต้องการทราบว่าค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบใดที่มีค่ามากกว่ากัน ดังนั้นผู้วิจัยจึงเปรียบเทียบค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 กับค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ผู้วิจัยเห็นว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นสามารถให้เกิดผลลัพธ์ของการทำเหมืองข้อมูลที่เป็นผลบวกลวง (False Positive) เป็นจำนวนมาก โดยเฉพาะอย่างยิ่งกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ (Li et al., 2005) ซึ่งน่าจะเป็นผลมาจาก

การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นจะให้ผลลัพธ์ที่ไม่สอดคล้องกับสหสัมพันธ์ (Correlation) แต่การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นให้ผลลัพธ์ของการทำเหมืองข้อมูลที่สอดคล้องกับสหสัมพันธ์ด้วย (Liu et al., 2008) ดังนั้นผู้วิจัยจึงคาดว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นจะให้ค่าผลสะท้อนกลับที่น้อยกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว จึงตั้งสมมติฐานไว้ ดังนี้

$H_0 : \mu_2 \geq \mu_1$

$H_1 : \mu_2 < \mu_1$

การยอมรับ H_0 หมายถึง การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว

การปฏิเสธ H_0 หมายถึง การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 นั้นมีประสิทธิภาพที่ดีกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว

3.4 ประชากรและหน่วยตัวอย่าง

สำหรับงานวิจัยนี้ข้อมูลซอฟต์แวร์อาร์ไคฟ์ที่ได้จากระบบคอนเคอเรนท์เวอร์ชั่นนั้นเป็นประชากรของการทดลองในงานวิจัยนี้ ซึ่งผู้วิจัยหวังว่าจะทดสอบระบบกับข้อมูลซอฟต์แวร์อาร์ไคฟ์ของโครงการพัฒนาซอฟต์แวร์ทั้งหมด แต่ในทางปฏิบัตินั้นผู้วิจัยไม่สามารถนำข้อมูลซอฟต์แวร์อาร์ไคฟ์ของโครงการพัฒนาซอฟต์แวร์ที่พัฒนาขึ้นในเชิงพาณิชย์ได้ ดังนั้นผู้วิจัยจึงเลือกใช้ข้อมูลซอฟต์แวร์อาร์ไคฟ์จากระบบคอนเคอเรนท์เวอร์ชั่นในโครงการพัฒนาซอฟต์แวร์ที่เป็นโครงการโอเพนซอร์ส (Open Source Project) จำนวน 1 โครงการคือโครงการพัฒนาซอฟต์แวร์ทางการบัญชีชื่อเคมายมันนี่ (KMyMoney) มาเพื่อเป็นหน่วยตัวอย่างของงานวิจัยนี้ รายละเอียดของซอฟต์แวร์นี้แสดงดังตารางต่อไปนี้

ตารางที่ 3-1 แสดงรายละเอียดของซอฟต์แวร์ทางการบัญชีชื่อเคมายมันนี่ (KMyMoney) เก็บข้อมูลในวันที่ 10 มกราคม พ.ศ. 2553

ข้อมูลทั่วไป	
ประเภทซอฟต์แวร์	การบัญชี การจัดทำและการพยากรณ์งบประมาณ
ผู้บำรุงรักษา	โทมัส โบมการ์ท และ มิเชล เอ็ดเวิร์ด
ประเภทลิขสิทธิ์	จีพีแอล (GPL)
ผู้ใช้เป้าหมาย	ผู้ใช้ทั่วไป องค์กรไม่แสวงหาผลประโยชน์
ขนาดซอฟต์แวร์	14.2 เมกะไบต์
จำนวนการถูกบรรจุลง (ครั้ง)	223,889
ข้อมูลด้านเทคนิค	
ภาษาที่ใช้ในการพัฒนา	ซีพลัสพลัส (C++)
แพลตฟอร์มที่รองรับ	ลินุกซ์ (Linux) ยูนิกซ์ (Unix) และฟรีบีเอสดี (FreeBSD)
ส่วนติดต่อผู้ใช้	เคดีอี (KDE), คิวท์ (Qt)
ฐานข้อมูลที่สนับสนุน	มายเอสคิวแอล (MySQL) โพสต์เกรสเอสคิวแอล (PostgreSQL) และ เอสคิวแอลไลท์ (SQLite)
ข้อมูลจากระบบคอนเคอเรนท์เวอร์ชั่น (CVS)	
วันที่เริ่มต้นโครงการ	16/04/2000
วันที่ปรับปรุงล่าสุด	28/12/2009
ระยะเวลา (ปี)	9.7
จำนวนนักพัฒนา (คน)	10
จำนวนทรานแซคชั่น (ทรานแซคชั่น)	28261
จำนวนแท็ก (แท็ก)	30777
จำนวนแฟ้มข้อมูล (แฟ้ม)	2583
จำนวนไดเรกทอรี่	118
จำนวนเอนทิตี้ (เอนทิตี้)	21660
จำนวนเอนทิตี้ต่อแฟ้มข้อมูลโดยเฉลี่ย (เอนทิตี้/แฟ้มข้อมูล)	8.4

จำนวนกิ่งก้าน (กิ่ง)	636
ข้อมูลโครงสร้างของซอฟต์แวร์	
จำนวนเนมสเปส (เนมสเปส)	7
จำนวนคลาส อินเตอร์เฟส สตรัคท์ และยูเนียน (คอมโพเนนท์)	543
จำนวนฟังก์ชั่น (ฟังก์ชั่น)	4472
จำนวนตัวแปร (ตัวแปร)	2896
จำนวนไทพ์ดีฟ (ไทพ์ดีฟ)	27
จำนวนอีนัมมูเรชั่น (อีนัมมูเรชั่น)	112
จำนวนอีนัมมูเรเตอร์ (อีนัมมูเรเตอร์)	962
จำนวนพรอพเพอร์ตี้ (พรอพเพอร์ตี้)	10

ซอฟต์แวร์เคมายมันนี่ (KMyMoney) คือ ซอฟต์แวร์จัดการการเงิน (Finance Management Software) ที่สามารถรองรับการใช้งานในระดับบุคคลและระดับองค์กรขนาดเล็กได้ ถูกพัฒนาขึ้นมาด้วยภาษาซีพลัสพลัส (C++) บนเครื่องมือที่ชื่อว่าคิวท์ (Qt) ทำให้เป็นโปรแกรมประยุกต์ที่พัฒนาขึ้นครั้งเดียวแต่ทำงานได้บนระบบปฏิบัติการตระกูลยูนิกซ์เกือบทุกรุ่น ซอฟต์แวร์เคมายมันนี่แบ่งกลุ่มของการจัดการข้อมูลออกเป็น 10 ส่วนดังนี้

1. สถาบันทางการเงิน (Institutions) คือ ส่วนที่ผู้ใช้สามารถบริหารจัดการบัญชีของผู้ใช้โดยแบ่งแยกตามสถาบันทางการเงินหรือธนาคารของแต่และบัญชีที่มีอยู่ ในส่วนนี้ผู้ใช้สามารถสร้าง แก้ไข ลบ หรือเปิดดูข้อมูลการเงินของแต่ละบัญชีในสถาบันทางการเงินหรือธนาคารได้ตามต้องการ
2. บัญชี (Accounts) คือ ส่วนที่ผู้ใช้สามารถบริหารจัดการบัญชีของผู้ใช้โดยแบ่งแยกตามประเภทของแต่และบัญชีที่มีอยู่ เช่น บัญชีทรัพย์สิน บัญชีหนี้สิน เป็นต้น ในส่วนนี้ผู้ใช้สามารถสร้าง แก้ไข ลบ หรือเปิดดูข้อมูลการเงินของแต่ละบัญชีได้ตามต้องการ
3. ตารางเวลา (Schedules) คือ ส่วนที่ผู้ใช้สามารถสร้างและจัดการตารางเวลาการทำธุรกรรมทางการเงินได้ ผู้ใช้สามารถสร้างตารางเวลาล่วงหน้า ตั้งตารางเวลา

แบบซ้ำเป็นรอบวัน รอบสัปดาห์ หรือรอบเดือน ช่วยให้ผู้ใช้ไม่ลืมและสามารถทำธุรกรรมได้ตรงตามเวลา นอกจากนั้นผู้ใช้สามารถกำหนดให้ธุรกรรมที่อยู่บนตารางเวลานั้นไปปรากฏบนบัญชีแยกประเภท (Ledgers) ได้ด้วย

4. หมวดการทำธุรกรรม (Categories) คือ ส่วนที่ผู้ใช้สามารถ แก้ไข หรือลบหมวดของการทำธุรกรรม เช่น หมวดธุรกรรมด้านศึกษา หมวดธุรกรรมทั่วไป เป็นต้น ผู้ใช้สามารถดึงข้อมูลธุรกรรมที่จัดทำไว้ในส่วนอื่นๆมาจัดหมวดหมู่ในส่วนนี้
5. ผู้ร่วมทำธุรกรรม (Payees) คือ ส่วนที่ผู้ใช้สามารถจัดการข้อมูลของบุคคล กลุ่มบุคคล หรือองค์กร ที่ผู้ใช้ไปทำธุรกรรมด้วย ในส่วนนี้ผู้ใช้สามารถเรียกดูรายการทำธุรกรรมทั้งหมดโดยแบ่งแยกตามบุคคล กลุ่มบุคคล หรือองค์กร ที่ผู้ใช้ไปทำธุรกรรมด้วยได้
6. บัญชีแยกประเภท (Ledgers) คือ ส่วนที่ผู้ใช้สามารถจัดการรายการธุรกรรมของผู้ใช้ ลักษณะเดียวกับโปรแกรมไมโครซอฟต์มันนี่ (Microsoft Money) นอกจากนั้นยังมีความสามารถพิเศษที่ชื่อว่าเลคเจอะเลนส์ (ledger lens) ใช้สำหรับเลือกรายการธุรกรรมตั้งแต่หนึ่งถึงสามรายการเพื่อขยายดูรายละเอียดภายในของรายการธุรกรรมนั้นๆ ข้อมูลรายการธุรกรรมทั้งหมดสามารถเลือกให้เรียงลำดับรายการตามคอลัมภ์ที่ต้องการได้
7. การลงทุน (Investments) คือ ส่วนที่ผู้ใช้สามารถติดตามการลงทุนพื้นฐานต่างๆ ได้ เช่นการติดตามราคาหุ้น ราคาทองคำ อัตราแลกเปลี่ยนเงินตราต่างประเทศ และกองทุนต่างๆ ผู้ใช้สามารถเลือกเพิ่ม แก้ไขหรือลบรายการการลงทุนที่ต้องการติดตามได้
8. รายงานทางการเงิน (Reports) คือ ส่วนที่ผู้ใช้สามารถจัดทำเอกสารรายงานทางการเงินต่างๆได้ โดยที่ผู้ใช้สามารถเลือกกำหนดค่าองค์ประกอบต่างๆได้อย่างอิสระ ในเวอร์ชั่นปัจจุบันผู้ใช้สามารถสร้างกราฟ แผนภาพแบบต่างๆในเอกสารรายงานได้ด้วย
9. งบประมาณ (Budgets) คือ ส่วนที่ผู้ใช้สามารถจัดการหมวดหมู่ของรายการรายได้และค่าใช้จ่ายที่คาดหวังไว้ในภายในช่วงของเวลาที่กำหนด ผู้ใช้สามารถกำหนดช่วงของเวลาได้ 3 แบบคือ รายปี รายเดือน หรือระยะเวลาเฉพาะที่กำหนดเอง นอกจากนั้นระบบยังสามารถสร้างรายงานแสดงการเปรียบเทียบรายได้และรายจ่ายจริงกับและรายได้และรายจ่ายที่คาดหวังไว้ได้

10. การพยากรณ์ทางการเงิน (Forecast) คือ ส่วนที่ผู้ใช้สามารถเรียกดูการพยากรณ์ทางการเงินของแต่ละบัญชีที่ผู้ใช้มีอยู่ได้ การพยากรณ์ทางการเงินในที่นี้หมายถึงการคาดการณ์ยอดเงินคงเหลือของแต่ละบัญชีในจุดเวลาอนาคต การพยากรณ์ของระบบเกิดขึ้นมาจากข้อมูลที่บันทึกในส่วนตารางเวลาและส่วนบัญชีแยกประเภทปัจจุบัน รวมถึงการรวบรวมธุรกรรมที่เคยเกิดขึ้นในอดีตมาพิจารณาด้วย

จากตารางแสดงรายละเอียดด้านต่างๆและหน้าที่การทำงานที่อธิบายไปในข้างต้นของซอฟต์แวร์ชื่อเคมายมันนี่ ผู้วิจัยเห็นว่าซอฟต์แวร์ชื่อเคมายมันนี่มีขนาดใหญ่ มีความซับซ้อนมากในระดับหนึ่ง มีนักพัฒนาผู้ร่วมโครงการเป็นจำนวนมาก และมีโอกาสที่จะมีนักพัฒนาใหม่เข้าร่วมโครงการได้เสมอ นอกจากนั้นซอฟต์แวร์ชื่อเคมายมันนี่นั้นพัฒนาขึ้นมาด้วยภาษาซีพลัสพลัส (C++) ซึ่งเป็นภาษาเชิงวัตถุ (Object Oriented Programming Language) ผู้วิจัยจึงหวังว่าซอร์สโค้ดของโครงการพัฒนาซอฟต์แวร์ชื่อเคมายมันนี่ (KMyMoney) จะสามารถเป็นตัวแทนที่ดีของซอฟต์แวร์ขนาดใหญ่ที่ถูกพัฒนาด้วยภาษาเชิงวัตถุ และมีนักพัฒนาผู้ร่วมโครงการเป็นจำนวนมากได้ ดังนั้นผู้วิจัยจึงเลือกข้อมูลซอฟต์แวร์อาร์ไคฟว์ของโครงการพัฒนาซอฟต์แวร์ชื่อเคมายมันนี่นี้เป็นข้อมูลที่ใช้ในการทดสอบของงานวิจัยนี้

สาเหตุที่งานวิจัยนี้เลือกตัวอย่างข้อมูลซอฟต์แวร์อาร์ไคฟว์เพียงตัวอย่างเดียว เนื่องจากงานวิจัยนี้ต้องการวิจัยเพื่อหาข้อมูลเบื้องต้น (Exploratory Research) เพื่อประโยชน์ในการวิจัยในอนาคตเท่านั้น ไม่ได้ต้องการสรุปผลให้ครอบคลุมการนำไปประยุกต์กับกรณีทั่วไป

3.5 แนวทางการทำวิจัย

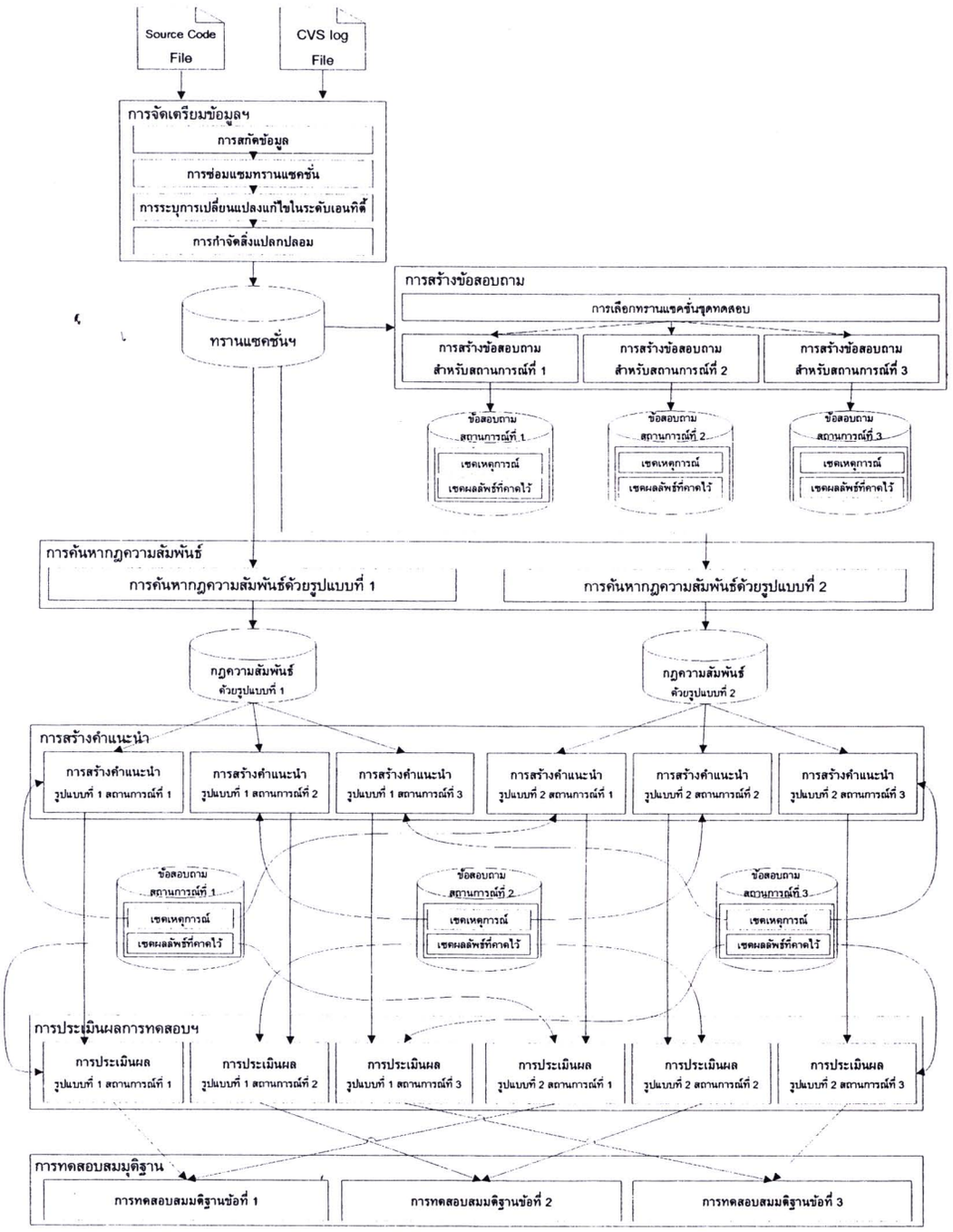
งานวิจัยนี้เป็นงานวิจัยเชิงทดลอง (Experimental Research) เนื่องจากเป็นการทดลองเพื่อเปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบ 2 ตัวแบบใน 3 สถานการณ์ สำหรับในสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาดนั้นตัวแบบที่มีประสิทธิภาพดีกว่าคือตัวแบบที่ให้ค่าเอฟเมสเซอร์ที่สูงกว่า และสำหรับในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้นตัวแบบที่มีประสิทธิภาพดีกว่าคือตัวแบบที่ให้ค่าผลสะท้อนกลับที่น้อยกว่า โดยในงานวิจัยนี้สนใจว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 หรือการใช้ตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่

ของ Liu และคณะ (Liu et al., 2008) ในการทำเหมืองข้อมูลนั้นจะสามารถช่วยเพิ่มประสิทธิภาพให้กับระบบให้คำแนะนำนักพัฒนาระหว่างการพัฒนาซอฟต์แวร์ที่ได้มาจากผลของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ได้ ในงานวิจัยเชิงทดลองนี้จะควบคุมตัวแปรอื่นๆ ให้เหมือนกันหมดนั่นคือ ข้อมูลซอฟต์แวร์อาร์ไคฟว์ ข้อสอบถามความถูกต้องระหว่างข้อมูลซอฟต์แวร์อาร์ไคฟว์และข้อสอบถาม และเครื่องมือที่ใช้ในการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูล (Data preprocessing tool) และเครื่องมือในการทำเหมืองข้อมูล (Data Mining tool) เพื่อให้ตัวแปรควบคุมที่กำหนดนั้นมีผลกระทบกับตัวแปรตามน้อยที่สุดและผลของงานวิจัยจะได้เป็นผลที่เกิดจากการเปลี่ยนแปลงตัวแปรต้นอย่างแท้จริง นั่นคืองานวิจัยจะทดลองว่าผลลัพธ์ของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์จะมีประสิทธิภาพเปลี่ยนแปลงไปอย่างไรเมื่อใช้ตัวแบบในการทำเหมืองข้อมูลแตกต่างกันสำหรับสถานการณ์ต่างๆ โดยได้สร้างเครื่องมือเพื่อทดสอบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ ดังนี้

1) การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่น
2) การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008)

งานวิจัยนี้ได้พัฒนาระบบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ออกเป็น 2 ตัวแบบดังกล่าว เนื่องจากผู้วิจัยสนใจว่าการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะนั้นสามารถเพิ่มประสิทธิภาพของระบบให้คำแนะนำนักพัฒนาในระหว่างการพัฒนาซอฟต์แวร์ของไอดีอีได้หรือไม่ ดังนั้นในการทดลองจึงต้องพัฒนาระบบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นดั้งเดิมไว้เป็นกลุ่มควบคุม เพื่อเป็นกลุ่มเปรียบเทียบกับระบบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่ของ Liu และคณะซึ่งเป็นกลุ่มทดสอบ สำหรับสถานการณ์ทั้ง 3 สถานการณ์

3.6 ขั้นตอนทดสอบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์



รูปที่ 3-1 แสดงขั้นตอนการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์

การออกแบบการทดสอบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ของงานวิจัยนี้มีขั้นตอนการทำงานทั้งหมดแสดงดังรูปที่ 3-1 ซึ่งการทดสอบในครั้งนี้จะแบ่งขั้นตอนในการทดสอบออกเป็น 6 ขั้นตอน คือ

1) การจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์
2) การสร้างข้อสอบถามสำหรับการทดสอบ 3 สถานการณ์
3) การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบสำหรับ 3 สถานการณ์
4) การสร้างเซตของคำแนะนำสำหรับเหตุการณ์
5) การประเมินผลการทดสอบ
6) การทดสอบสมมติฐาน

รายละเอียดข้อมูลเข้า กระบวนการทำงานและข้อมูลออกของแต่ละขั้นตอนการทดสอบอธิบายดังต่อไปนี้

3.6.1 การจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์

การจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ คือ ส่วนที่ทำหน้าที่ในการดึงแฟ้มข้อมูลซอร์สโค้ดและแฟ้มข้อมูลบันทึกจากรีพอสิโทริของระบบคอนเคอเรนท์เวอร์ชั่นแล้วนำแฟ้มข้อมูลเหล่านั้นมาผ่านกระบวนการ 4 กระบวนการเพื่อให้ได้ข้อมูลทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข ดังนี้

1) การสกัดข้อมูล (Data Extraction)

2) การซ่อมแซมทรานแซคชั่น (Restoring Transactions)

3) การระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้ (Mapping Changes to Entities)

4) การกำจัดสิ่งแปลกปลอม (Data Cleaning)

ขั้นตอนการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ 4 กระบวนการข้างต้นถูกเสนอขึ้นมาโดย Zimmermann และคณะ ในปี 2004 (Zimmermann et al., 2004) และเป็นขั้นตอนวิธีที่มีงานวิจัยที่เกี่ยวข้องกับการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์

ไคฟว์หลายงายวิจัย (Zimmermann et al., 2005; Livshits et al., 2005; Williams et al., 2005; Breu et al., 2006; Weißgerber et al, 2006) นำไปใช้ งานวิจัยนี้นำขั้นตอนการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 4 กระบวนการมาใช้เช่นกัน และกำหนดรายละเอียดต่างๆของแต่ละกระบวนการเหมือนกับที่ Zimmermann และคณะได้เสนอไว้ (Zimmermann et al., 2004) ผู้วิจัยจะอธิบายกระบวนการทำงานพร้อมกับยกตัวอย่างบางส่วนประกอบการอธิบาย เพื่อเพิ่มความเข้าใจในการทำงานของแต่ละกระบวนการมากขึ้นด้วย

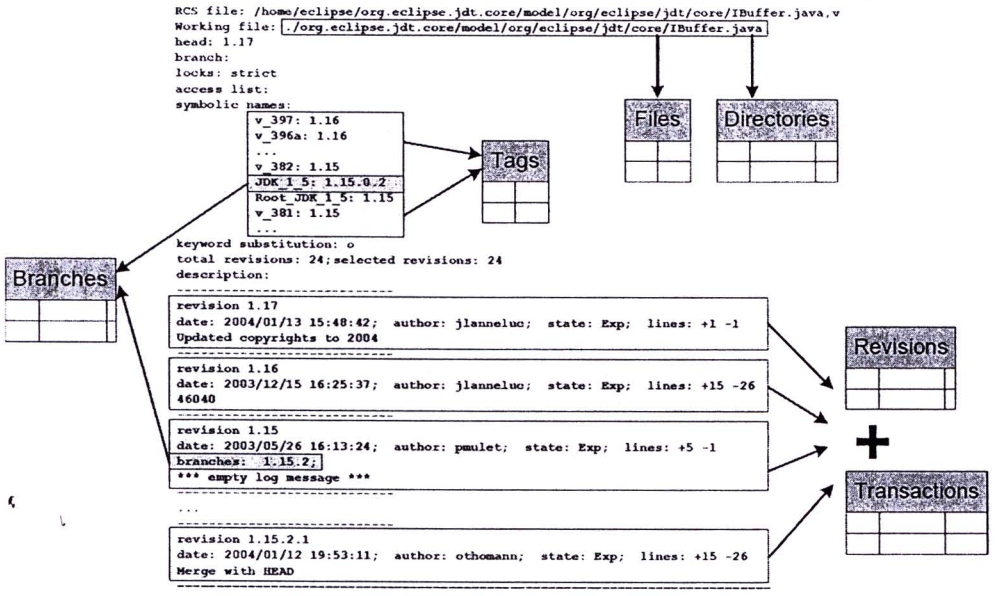
- **การสกัดข้อมูล** (Data Extraction)

การสกัดข้อมูลจากแฟ้มข้อมูลบันทึกของระบบคอนเคอเรนท์เวอร์ชั่นเริ่มต้นจากการเรียกใช้คำสั่ง CVS log จากไดเรคทอรี่ราก (root directory) ของโครงการพัฒนาซอฟต์แวร์ที่ต้องการ ผลลัพธ์ที่ส่งกลับคืนมาก็คือข้อมูลแฟ้มข้อมูลซอร์สโค้ดและแฟ้มข้อมูลบันทึก (Log File) ที่บันทึกอยู่บนรีพอสิโทริของระบบคอนเคอเรนท์เวอร์ชั่นนั้น แฟ้มข้อมูลบันทึกที่ได้มาจะถูกนำมาวิเคราะห์รูปแบบไวยากรณ์ (Parse) และถูกนำไปบันทึกลงฐานข้อมูล (Zimmermann et al., 2004)

ข้อมูลเข้า คือ แฟ้มข้อมูลบันทึกของระบบคอนเคอเรนท์เวอร์ชั่น (CVS Log File)

ข้อมูลออก คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (ฐานข้อมูลที่ประกอบด้วยตาราง 6 ตาราง คือตารางชื่อ Files ตารางชื่อ Directories ตารางชื่อ Tags ตารางชื่อ Branches ตารางชื่อ Revisions และตารางชื่อ Transactions)

กระบวนการทำงาน สามารถอธิบายได้โดยแผนภาพต่อไปนี้ (Zimmermann et al., 2004)



รูปที่ 3-2 แสดงตัวอย่างแฟ้มข้อมูลบันทึกของระบบคอนเคอเรนท์เวอร์ชั่น

รูปแบบไวยากรณ์ แอททริบิวต์ และความหมายของแต่ละแอททริบิวต์ อธิบายไว้ในบทที่ 2 รายละเอียดของข้อมูลทั้งหมดที่จะถูกบันทึกเอาไว้จากขั้นตอนการสกัดข้อมูลสามารถอธิบายได้ดังต่อไปนี้

- แอททริบิวต์ RCS file ของแต่ละส่วน (Sections) ในแฟ้มข้อมูลบันทึกสามารถสกัดข้อมูลออกมาเป็นรายชื่อและรายละเอียดของแฟ้มข้อมูล (Files) และไดเรคทอรี่ (Directories) ทั้งหมดของโครงการ จากตัวอย่างแฟ้มข้อมูลบันทึกข้างต้นจะทำให้เกิดระเบียน (Record) ใหม่ขึ้นมาในตารางชื่อ Files และ Directories ดังนี้

ตัวอย่างระเบียนของตารางชื่อ Directories

DirectoryID	DirectoryName	Depth
1	/org.eclipse.jdt.core/Model/org/elipse/jdt/core/	6

ตัวอย่างระเบียนของตารางชื่อ Files

FileID	FileName	DirectoryID	FileExtension	Depth	NumberOfRevisions
1	IBuffer.java	1	.java	7	0

- แอททริบิวต์ description ประกอบด้วยส่วนย่อยหลายส่วนแต่ละส่วนแสดงถึงการเปลี่ยนแปลงแก้ไขแต่ละครั้งที่เกิดขึ้น ข้อมูลในแต่ละส่วนย่อยนี้สามารถสกัดข้อมูลออกมาเป็นรายการการเปลี่ยนแปลงแก้ไขแฟ้มข้อมูล (Revisions) ได้ จากตัวอย่างแฟ้มข้อมูลบันทึกข้างต้นจะทำให้เกิดระเบียนใหม่ขึ้นมาในตารางชื่อ Revisions ดังนี้

ตัวอย่างระเบียนของตารางชื่อ Revisions

FileID	RevisionID	TransactionID	CheckinTime	Plus	Minus	State	BranchPrefix
1	1.17	1	2004-01-13 15:48:42	1	1	Exp	NULL

- รายการการเปลี่ยนแปลงแก้ไขแฟ้มข้อมูลที่ได้มาข้างต้นจะถูกนำมาพิจารณาว่ารายการใดบางที่เกิดขึ้นในเวลาเดียวกันและเกิดขึ้นโดยนักพัฒนาคนเดียวกันจะถูกรวมกันไว้เป็นทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (Transactions) เดียวกัน จากตัวอย่างแฟ้มข้อมูลบันทึกข้างต้นจะทำให้เกิดระเบียนใหม่ขึ้นมาในตารางชื่อ Transactions ดังนี้

ตัวอย่างระเบียนของตารางชื่อ Transactions

TransactionID	Author	Message	MessageMD5	BeginTime	EndTime	IsNoise
1	jlanneluc	Updated copyrights to 2004	817397A1A 8F94C3C8 1AF1C5DB E9F37F7	2004-01-13 15:48:42	2004-01-13 15:48:42	N

- แอททริบิวต์ symbolic name แต่ละส่วนย่อยของแอททริบิวต์ description ในแฟ้มข้อมูลบันทึกสามารถสกัดข้อมูลออกมาเป็นรายชื่อของแท็ก (Tags) ที่นักพัฒนาตั้งไว้ให้กับการเปลี่ยนแปลงแก้ไขนั้นๆได้ จากตัวอย่างแฟ้มข้อมูลบันทึกข้างต้นจะทำให้เกิดระเบียนใหม่ขึ้นมาในตารางชื่อ Tags ดังนี้

ตัวอย่างระเบียนของตารางชื่อ Tags

FileID	TagName	RevisionID
1	1.16	v_396a

– ตารางชื่อ Branches ในฐานข้อมูลนั้นจะบันทึกจุดต่อกิ่ง (Branch Points) และชื่อของการต่อกิ่ง (Branch Names) ข้อมูลทั้ง 2 ข้อมูลนี้ถูกเก็บมาจาก 2 ส่วนของข้อมูลที่ได้จากการเรียกคำสั่ง CVS log โดยที่ชื่อของการต่อกิ่งก็คือชื่อที่เป็นสัญลักษณ์ที่มีหมายเลขปรากฏอยู่ด้วย ตัวอย่างเช่น JDK_1_5 มีหมายเลขเวอร์ชั่นเป็น 1.15.0.2 จะได้ชื่อของการต่อกิ่งนี้เป็น 1.15.2 ส่วนจุดต่อกิ่งนั้นได้มาจากตารางแฮช (Hash Map) ที่ใช้ชื่อของการต่อกิ่งเป็นคีย์

ตัวอย่างระเบียนของตารางชื่อ Branches

FileID	BranchPrefix	OriginRevision	BranchName	InternalRevision
1	1.15.2	1.15	JDK_1_5	1.15.0.2

- **การซ่อมแซมทรานแซคชั่น** (Restoring Transactions)

การเข้าไปอ่านข้อมูลที่ถูกบันทึกเอาไว้ในแฟ้มข้อมูลบันทึก (Log File) ต่างๆบนเครื่องแม่ข่ายของระบบคอนเคอเรนท์เวอร์ชั่นว่ามีข้อความบันทึก (Log Message) ใดบ้างที่ระบุการเปลี่ยนแปลงแก้ไขทั้งหมดที่เกิดจากนักพัฒนาคนเดียวกันและเกิดขึ้นในเวลาเดียวกัน ข้อมูลการเปลี่ยนแปลงแก้ไขที่เกิดจากนักพัฒนาคนเดียวกันและเกิดขึ้นในเวลาเดียวกันจะถูกนำมาแปลงเป็นทรานแซคชั่น 1 ทรานแซคชั่นแล้วบันทึกลงในตาราง Transactions บนฐานข้อมูล คำว่า ในเวลาเดียวกัน ในที่นี้นั้นหมายรวมถึงในเวลาที่ใกล้เคียงกันด้วย เนื่องจากการคอมมิทในแต่ละครั้งอาจใช้เวลาในการดำเนินการหลายวินาทีหรือหลายนาที โดยเฉพาะอย่างยิ่งการคอมมิทที่ละหลายๆแฟ้มข้อมูล (Zimmermann et al., 2004) ดังนั้นในทางปฏิบัติแล้วนอกจากการพิจารณาที่การคอมมิทในเวลาเดียวกันแล้วยังต้องมีวิธีการพิจารณาที่การคอมมิทระหว่างช่วงของเวลา (Time Interval) เดียวกันด้วย วิธีการพิจารณาการเปลี่ยนแปลงแก้ไขระหว่างช่วงของเวลานั้นมี 2 วิธีคือวิธี

กำหนดกรอบเวลาที่แน่นอน (Fixed Time Windows) และวิธีเลื่อนกรอบเวลา (Sliding Time Windows) ตามที่ได้อธิบายอย่างละเอียดในบทที่ 2 (Zimmermann et al., 2004)

ข้อมูลเข้า คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไขและข้อมูลที่เกี่ยวข้องอื่นๆ (ฐานข้อมูลทั้ง 6 ตาราง คือตารางชื่อ Files ตารางชื่อ Directories ตารางชื่อ Tags ตารางชื่อ Branches ตารางชื่อ Revisions และตารางชื่อ Transactions)

ข้อมูลออก คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (เฉพาะตารางชื่อ Transactions ที่ถูกปรับปรุงข้อมูลใหม่)

กระบวนการทำงาน สำหรับในการทดสอบครั้งนี้ผู้วิจัยเลือกใช้วิธีเลื่อนกรอบเวลา (Sliding Time Windows) แบบเดียวกับที่งานวิจัยในอดีตหลายๆงานวิจัยเลือกใช้ (Zimmermann et al., 2004; Ying et al., 2004; Livshits et al., 2005; Weißgerber et al., 2005; Zimmermann et al., 2005; Kim et al., 2005; Breu et al 2006) หลักการของวิธีการนี้ คือ การกำหนดช่องว่างระหว่างการเปลี่ยนแปลงแก้ไข (Revision, Change) 2 ครั้งที่มากที่สุด จุดเริ่มต้นของกรอบของช่วงเวลาจะถูกเลื่อนไปที่การเปลี่ยนแปลงแก้ไขครั้งต่อไปเสมอตราบใดที่การเปลี่ยนแปลงแก้ไขครั้งต่อไปนั้นมีจุดเริ่มต้นอยู่ภายในกรอบเวลาของการเปลี่ยนแปลงแก้ไขครั้งก่อนหน้า

จากข้อมูลของการเปลี่ยนแปลงแก้ไขที่บันทึกไว้ในตารางชื่อ Revisions และข้อมูลทรานแซคชั่นที่อยู่ในตารางชื่อ Transactions จะต้องถูกดึงขึ้นมาพิจารณาระบุทรานแซคชั่นที่ถูกต้องใหม่ทั้งหมดตามวิธีเลื่อนกรอบเวลาข้างต้นโดยที่ในการทดสอบครั้งนี้ผู้วิจัยกำหนดให้ความกว้างของกรอบเวลาที่พิจารณาอยู่ที่ 200 วินาที เช่นเดียวกับงานวิจัยอื่นๆในอดีต (Zimmermann et al., 2004; Ying et al., 2004; Livshits et al., 2005; Weißgerber et al., 2005; Zimmermann et al., 2005; Kim et al., 2005; Breu et al 2006) สำหรับทุกการเปลี่ยนแปลงแก้ไข α_1, α_2, ..., α_k (เรียงตามลำดับเวลาที่บันทึก (time(α_i))) ที่เป็นส่วนหนึ่งของทรานแซคชั่น T เดียวกันนั้น จะต้องอยู่ภายใต้เงื่อนไข

$$\forall \alpha_i \in T : author(\alpha_i) = author(\alpha_1)$$

$$\forall \alpha_i \in T : \log\_message(\alpha_i) = \log\_message(\alpha_1)$$

$$\forall i \in \{2, \dots, k\} : |time(\alpha_i) - time(\alpha_{i-1})| \leq 200sec$$

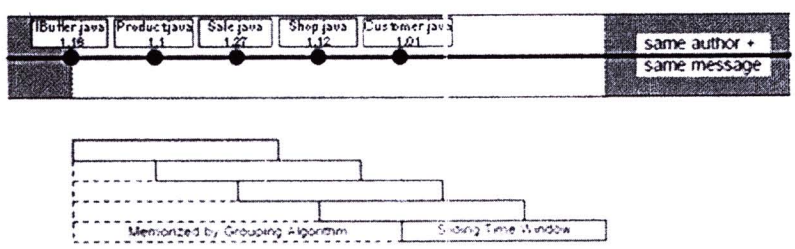
นอกจากนั้นการเปลี่ยนแปลงแก้ไขของแต่ละแฟ้มข้อมูลจะปรากฏอยู่บน 1 ทรานแซคชั่นได้เพียงครั้งเดียว เนื่องจากระบบคอนเคอเรนท์เวอร์ชั่นไม่อนุญาตให้มีการคอมมิทการ

เปลี่ยนแปลงเวอร์ชั่นของแฟ้มข้อมูลเดียวกัน 2 ครั้งในเวลาเดียวกันได้ ดังนั้นจึงมีเงื่อนไขเพิ่มมาอีก 1 ข้อดังนี้

$$\forall \alpha_a, \alpha_b \in T : \alpha_a \neq \alpha_b \rightarrow file(\alpha_a) \neq file(\alpha_b)$$

เมื่อพิจารณาทรานแซคชั่นทั้งหมดเรียบร้อยแล้วการทำการแก้ไขระเบียนภายในตารางชื่อ Transactions ใหม่ให้ถูกต้อง

ตัวอย่างการทำงานของวิธีเลื่อนกรอบเวลา แสดงได้ดังต่อไปนี้ (Zimmermann et al., 2004)



รูปที่ 3-3 แสดงตัวอย่างการพิจารณาการเปลี่ยนแปลงแก้ไขเวอร์ชั่นระหว่างช่วงของเวลาด้วยวิธีเลื่อนกรอบเวลา

รูปที่ 3-3 แสดงการพิจารณาด้วยวิธีเลื่อนกรอบเวลา โดยเริ่มต้นกรอบเวลาที่มีช่วงแน่นอนเริ่มต้นที่จุดของการเปลี่ยนแปลงแก้ไขที่แฟ้มข้อมูล IBuffer.java เวอร์ชั่นที่ 1.16 และกรอบเวลาถูกเลื่อนไปเรื่อยๆ จนสิ้นสุดดังรูป ดังนั้นการเปลี่ยนแปลงแก้ไขที่แฟ้มข้อมูล IBuffer.java เวอร์ชั่นที่ 1.16 แฟ้มข้อมูล Product.java เวอร์ชั่นที่ 1.1 แฟ้มข้อมูล Sale.java เวอร์ชั่นที่ 1.27 แฟ้มข้อมูล Shop.java เวอร์ชั่นที่ 1.12 และ แฟ้มข้อมูล Customer.java เวอร์ชั่นที่ 1.01 จะถูกพิจารณาว่าเกิดขึ้นพร้อมกันและอยู่ภายในทรานแซคชั่นเดียวกัน

- **การระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้** (Mapping Changes to Entities)

ข้อมูลที่ถูกจัดเก็บไว้ในรีพอสิโทริของระบบคอนเคอเรนท์เวอร์ชั่นนั้นมีเพียงข้อมูลแฟ้มข้อมูลทุกแฟ้มข้อมูลในโครงการและข้อมูลการเปลี่ยนแปลงแก้ไขในระดับแฟ้มข้อมูล (File) หรือคลาส (Class) ที่เก็บอยู่ในรูปของแฟ้มข้อมูลบันทึก (Log File) เท่านั้น แต่ไม่มีบันทึกว่าการ

เปลี่ยนแปลงแก้ไขที่เกิดขึ้นนั้นเกิดขึ้นกับฟังก์ชั่น (Function) หรือ เมธอด (Method) ใดบ้าง มีตัวแปร (Variable) ใดถูกเพิ่มเข้ามา แก้ไข หรือถูกลบออกไปบ้าง วิธีการที่มีความแม่นยำสูงกว่าแต่ก็มีค่าใช้จ่ายในการคำนวณสูงวิธีหนึ่ง คือการกำหนดเอนทิตี้ (ตัวแปร ฟังก์ชั่นหรือเมธอด) ทั้งหมดภายในแฟ้มข้อมูลทั้ง 2 เวอร์ชั่นโดยการนำไปผ่านตัววิเคราะห์ไวยากรณ์ (Parser) จากนั้นก็ทำการเปรียบเทียบซอร์สโค้ดของเอนทิตี้เดียวกันใน 2 เวอร์ชั่น หรือกล่าวคือเป็นการประยุกต์ใช้ฟังก์ชั่นดิฟฟ์ (diff) (Miller et al., 1985) ในระดับของเอนทิตี้นั่นเอง (Zimmermann et al., 2004)

ข้อมูลเข้า คือ แฟ้มข้อมูลบันทึกของระบบคอนเคอเรนท์เวอร์ชั่น (CVS Log File) แฟ้มข้อมูลซอร์สโค้ดทุกเวอร์ชั่นของโครงการ และฐานข้อมูลตารางชื่อ Files ตารางชื่อ Directories ตารางชื่อ Tags ตารางชื่อ Branches ตารางชื่อ Revisions และตารางชื่อ Transactions

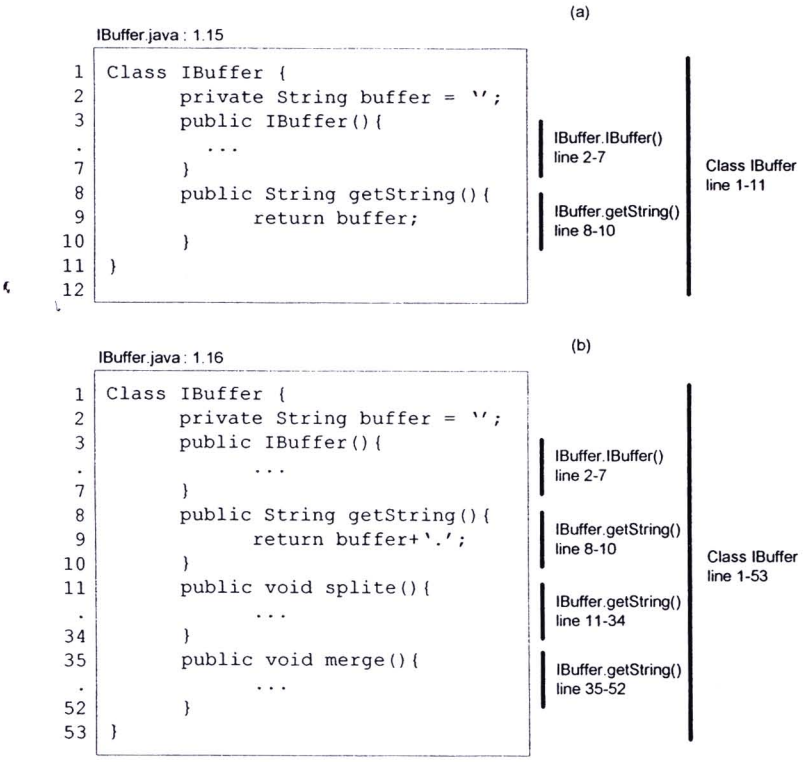
ข้อมูลออก คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (เฉพาะตารางชื่อ Transactions และตารางชื่อ Revisions ที่ถูกเพิ่มระเบียบใหม่เข้าไป หรือถูกปรับปรุงข้อมูลใหม่)

กระบวนการทำงาน สามารถดำเนินการได้ดังต่อไปนี้ (Zimmermann et al., 2004)

1) กำหนดเซต E_1 คือเซตของเอนทิตี้ที่มีอยู่ทั้งหมดในเวอร์ชั่น r_1 ของแฟ้มข้อมูล และกำหนดเซต E_2 คือเซตของเอนทิตี้ที่มีอยู่ทั้งหมดในเวอร์ชั่น r_2 ของแฟ้มข้อมูลเดียวกัน
2) เอนทิตี้ที่ถูกเพิ่มเข้ามาใหม่สามารถหาได้จาก $E_2 - E_1$
3) เอนทิตี้ที่ถูกลบออกไปสามารถหาได้จาก $E_1 - E_2$
4) ทุกๆ เอนทิตี้ที่อยู่ในเซต $E_1 \cap E_2$ อาจจะเป็นเอนทิตี้ที่มีการเปลี่ยนแปลงภายใน การตัดสินว่าเอนทิตี้ใดบ้างที่มีการเปลี่ยนแปลงแก้ไขสามารถทำได้โดยการประยุกต์ใช้ฟังก์ชั่นดิฟฟ์ (diff) กับซอร์สโค้ดของเอนทิตี้นั้นๆของทั้ง 2 เวอร์ชั่น

ภายในแพลตฟอร์ม (Platform) ของอิคลิพซ์ (Eclipse) นั้นมีการจัดเตรียมโครงร่าง (Framework) สำหรับการเปรียบเทียบความแตกต่างระหว่าง 2 ซอร์สโค้ดใดๆที่มีประสิทธิภาพสูงและสามารถนำไปประยุกต์เพิ่มเติมได้ทั้งหมด 2 โครงร่างได้แก่โครงร่างเรนจ์ดิฟเฟอร์เรนเซอร์ (Range Differencer) และโครงร่างสตรัคเชอร์เมิร์จวิวเวอร์ (Structure Merge Viewer) ซึ่งได้กล่าวไปแล้วในบทที่ 2 วิธีการเปรียบเทียบซอร์สโค้ดของเอนทิตี้เดียวกันใน 2 เวอร์ชั่นของงานวิจัยชิ้นนี้ผู้วิจัยเลือกใช้โครงร่างเรนจ์ดิฟเฟอร์เรนเซอร์ (Zimmermann et al., 2004)

ตัวอย่างของการระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้ตั้งแต่ขั้นตอนการระบุเอนทิตี้ของทั้ง 2 เวอร์ชั่นของแฟ้มข้อมูลจนถึงขั้นตอนวิธีการเปรียบเทียบความแตกต่างของซอร์สโค้ดของเอนทิตี้เดียวกันใน 2 เวอร์ชั่น ดังต่อไปนี้



รูปที่ 3-4 แสดงตัวอย่างการระบุเอนทิตี้ภายในซอร์สโค้ด 2 เวอร์ชั่นของแฟ้มข้อมูล IBuffer.java

จากรูปที่ 3-4 (a) แสดงซอร์สโค้ดของแฟ้มข้อมูล IBuffer.java ในเวอร์ชั่นที่ 1.15 และรูปที่ 3-4 (b) แสดงซอร์สโค้ดของแฟ้มข้อมูล IBuffer.java ในเวอร์ชั่นที่ 1.16 เมื่อนำซอร์สโค้ดทั้ง 2 เวอร์ชั่นข้างต้นไปผ่านการวิเคราะห์ไวยากรณ์ (Parse) โดยตัววิเคราะห์ไวยากรณ์ (Parser) ทำให้ได้เซต E_1 คือเซตของเอนทิตี้ที่มีอยู่ทั้งหมดในเวอร์ชั่น 1.15 ของแฟ้มข้อมูล IBuffer.java และได้เซต E_2 คือเซตของเอนทิตี้ที่มีอยู่ทั้งหมดในเวอร์ชั่น 1.16 ของแฟ้มข้อมูลเดียวกัน ดังนี้

E_1 = { (variable, buffer, (Class, IBuffer, (file, IBuffer.java, ...))),

(method, IBuffer(), (Class, IBuffer, (file, IBuffer.java, ...))),

(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

(Class, IBuffer, (file, IBuffer.java, ...)),

(file, IBuffer.java, ...)

}

E_2 = { (variable, buffer, (Class, IBuffer, (file, IBuffer.java, ...))),

(method, IBuffer(), (Class, IBuffer, (file, IBuffer.java, ...))),

(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

(method, splite(), (Class, IBuffer, (file, IBuffer.java, ...))),

(method, Merge(), (Class, IBuffer, (file, IBuffer.java, ...))),

(Class, IBuffer, (file, IBuffer.java, ...)),

(file, IBuffer.java, ...)

}

เอนทิตี้ที่ถูกเพิ่มเข้ามาใหม่สามารถหาได้จาก E_2 - E_1 ดังนี้

E_2 - E_1 = { (method, splite(), (Class, IBuffer, (file, IBuffer.java, ...))),

(method, Merge(), (Class, IBuffer, (file, IBuffer.java, ...)))

}

จากเซต E_2 - E_1 ข้างต้นทำให้ทราบว่ามีการเพิ่มเอนทิตี้ประเภทเมธอดชื่อ splite() และ Merge() เข้าสู่เอนทิตี้ประเภทคลาสชื่อ IBuffer นั่นคือเกิดเซตของการเปลี่ยนแปลงแก้ไขดังนี้

{ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

add_to(Class, IBuffer, (file, IBuffer.java, ...))

}

เอนทิตี้ที่ถูกลบออกไปสามารถหาได้จาก E_1 - E_2 ดังนี้

$E_1 - E_2 = \{\}$

จากเซต $E_1 - E_2$ ข้างต้นเป็นเซตว่างดังนั้นทำให้เกิดเซตของการเปลี่ยนแปลงแก้ไขที่เป็นเซตว่างเช่นกัน

เอนทิตี้ที่อาจจะถูกเปลี่ยนแปลงแก้ไขภายในสามารถหาได้จาก $E_1 \cap E_2$ ดังนี้

$E_1 \cap E_2 = \{$ (variable, buffer, (Class, IBuffer, (file, IBuffer.java, ...))),

(method, IBuffer(), (Class, IBuffer, (file, IBuffer.java, ...))),

(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

(Class, IBuffer, (file, IBuffer.java, ...)),

(file, IBuffer.java, ...)

}

การตัดสินว่าเอนทิตี้ใดบ้างที่มีการเปลี่ยนแปลงแก้ไขภายในจริงๆสามารถทำได้โดยการประยุกต์ใช้โครงร่างเรนจ์ดิฟเฟอร์เรนเซอร์ (Range Differencer) สำหรับการเปรียบเทียบความแตกต่างระหว่าง 2 ซอร์สโค้ดใดๆ ผลของการเปรียบเทียบคือมีการเปลี่ยนแปลงภายในเพียงเอนทิตี้เดียวคือเอนทิตี้ประเภทเมธอดชื่อ getString() จาก return buffer; ในเวอร์ชั่น 1.15 เป็น return buffer+'.'; ในเวอร์ชั่น 1.16 นั่นคือมีเซตของการเปลี่ยนแปลงแก้ไขดังนี้

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))) }

ดังนั้นเซตของการเปลี่ยนแปลงแก้ไขทั้งหมดที่เกิดขึ้นจากเวอร์ชั่น 1.15 ไปเป็นเวอร์ชั่น 1.16 ของแฟ้มข้อมูล IBuffer.java คือ

{ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...)))

}

ดำเนินการขั้นตอนทั้งหมดนี้ซ้ำกับแฟ้มข้อมูล Product.java จากเวอร์ชั่น 1.07 ไปเป็นเวอร์ชั่นที่ 1.1 แฟ้มข้อมูล Sale.java จากเวอร์ชั่น 1.26 ไปเป็นเวอร์ชั่นที่ 1.27 แฟ้มข้อมูล Shop.java จากเวอร์ชั่น 1.11 ไปเป็นเวอร์ชั่นที่ 1.12 และ แฟ้มข้อมูล Customer.java จากเวอร์ชั่น 1.0 ไปเป็นเวอร์ชั่นที่ 1.01 เมื่อได้เซตของการเปลี่ยนแปลงแก้ไขทั้งหมดมาแล้วให้นำการเปลี่ยนแปลงแก้ไขเหล่านั้นมารวมกันเป็นเซตเดียวแล้วเรียกเซตนั้นว่าเซตทรานแซคชั่น 1 ทรานแซคชั่น เพื่อความสะดวกผู้วิจัยจึงสมมุติเซตของการเปลี่ยนแปลงแก้ไขของแฟ้มข้อมูลทั้งหมดที่ได้หลังจากทำขั้นตอนข้างต้นเรียบร้อยแล้ว ดังนี้

เซตของการเปลี่ยนแปลงแก้ไขทั้งหมดที่เกิดขึ้นจากเวอร์ชั่น 1.07 ไปเป็นเวอร์ชั่น 1.1 ของแฟ้มข้อมูล Product.java คือ

```
{ alter(method, getPrice(), (Class, Product, (file, Product.java, ...))),
  alter(method, getDetail(), (Class, Product, (file, Product.java, ...))),
  del_from(Class, Product, (file, Product.java, ...))
}
```

เซตของการเปลี่ยนแปลงแก้ไขทั้งหมดที่เกิดขึ้นจากเวอร์ชั่น 1.26 ไปเป็นเวอร์ชั่น 1.27 ของแฟ้มข้อมูล Sale.java คือ

```
{ alter(method, add(), (Class, Sale, (file, Sale.java, ...))),
  alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),
  alter(Class, Sale, (file, Sale.java, ...)),
  alter(Class, Sale, (file, Sale.java, ...)))
}
```

เซตของการเปลี่ยนแปลงแก้ไขทั้งหมดที่เกิดขึ้นจากเวอร์ชั่น 1.11 ไปเป็นเวอร์ชั่น 1.12 ของแฟ้มข้อมูล Shop.java คือ

```
{ alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),
```

alter(Class, Shop, (file, Shop.java, ...)))

}

เซตของการเปลี่ยนแปลงแก้ไขทั้งหมดที่เกิดขึ้นจากเวอร์ชั่น 1.0 ไปเป็นเวอร์ชั่น 1.01 ของแฟ้มข้อมูล Customer.java คือ

{ alter(method, add(), (Class, Customer, (file, Customer.java, ...))),

alter(method, getID(), (Class, Customer, (file, Customer.java, ...))),

alter(Class, Customer, (file, Customer.java, ...)),

del_from(Class, Customer, (file, Customer.java, ...)))

}

สำหรับงานวิจัยนี้ผู้วิจัยสนใจทรานแซคชั่นของการเปลี่ยนแปลงแก้ไขเฉพาะในระดับของแฟ้มข้อมูล (file) และคลาส (Class) เท่านั้น แต่อย่างไรก็ตามขั้นตอนการระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้นี้ก็ยังคงจำเป็นสำหรับงานวิจัยนี้เนื่องจากในภาษาซีพลัสพลัส (C++) นั้นอนุญาตให้ใน 1 แฟ้มข้อมูลสามารถมีคลาสได้มากกว่า 1 คลาส ดังนั้นขั้นตอนการระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้นี้จะช่วยในการระบุเอนทิตี้ระดับคลาสของกรณีดังกล่าวได้

- **การกำจัดสิ่งแปลกปลอม (Data Cleaning)**

ขั้นตอนการกำจัดสิ่งแปลกปลอม (Data Cleaning) เป็นขั้นตอนที่เข้าไปตรวจสอบข้อมูลทั้งหมดเพื่อค้นหาสิ่งแปลกปลอมและกำจัดสิ่งแปลกปลอมเหล่านั้นออกไป ลักษณะของข้อมูลทรานแซคชั่นที่จะถูกระบุว่าเป็นสิ่งแปลกปลอมมีอยู่ 2 ลักษณะคือ 1) ทรานแซคชั่นขนาดใหญ่ (Large Transactions) และ 2) ทรานแซคชั่นการผสานกิ่ง (Merge Transactions) ตามที่ได้อธิบายรายละเอียดไว้ในบทที่ 2

ข้อมูลเข้า คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (เฉพาะตารางชื่อ Transactions และตารางชื่อ Revisions)

ข้อมูลออก คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (ฐานข้อมูลเฉพาะตารางชื่อ Transactions และตารางชื่อ Revisions ที่ถูกปรับปรุงข้อมูลใหม่)

กระบวนการทำงาน คือ พิจารณาทรานแซคชั่นที่ได้มาจากข้อมูลออกของขั้นตอนการระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้ข้างต้นว่ามีทรานแซคชั่นใดมีลักษณะเข้าข่ายที่จะเป็นสิ่งแปลกปลอมทั้ง 2 ลักษณะหรือไม่ ถ้าพบทรานแซคชั่นที่เข้าข่ายดังกล่าวจะทำการลบทรานแซคชั่นเหล่านั้นออกไป

จากตัวอย่างของเซตรายการการเปลี่ยนแปลงแก้ไขที่ได้มาจากขั้นตอนการระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้ในข้างนั้นมีทรานแซคชั่นที่มีลักษณะเป็นทรานแซคชั่นการผสานกิ่งตามเงื่อนไขที่ได้กล่าวไปในบทที่ 2 จึงทำให้ผลลัพธ์หลังจากผ่านขั้นตอนการกำจัดสิ่งแปลกปลอมเหลือเซตรายการการเปลี่ยนแปลงแก้ไข 3 เซต ดังนี้

```
{ add_to(Class, IBuffer, (file, IBuffer.java, ...)),
  add_to(Class, IBuffer, (file, IBuffer.java, ...)),
  alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...)))
}
{ alter(method, add(), (Class, Sale, (file, Sale.java, ...))),
  alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),
  alter(Class, Sale, (file, Sale.java, ...)),
  alter(Class, Sale, (file, Sale.java, ...))),
}
{ alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),
  alter(Class, Shop, (file, Shop.java, ...)))
}
```

เซตของการเปลี่ยนแปลงแก้ไขเหล่านี้จะถูกนำมารวมกันและตัดรายการการเปลี่ยนแปลงแก้ไขที่ซ้ำซ้อนกันออกแล้วเรียกเซตนี้ว่าทรานแซคชั่น กล่าวคือจากตัวอย่างที่ยกมาข้างต้นนั้นเมื่อนำมาผ่านส่วนการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟ์ทั้ง 4 ขั้นตอนแล้วทำให้เกิดทรานแซคชั่นขึ้น 1 ทรานแซคชั่นที่มีรายการของการเปลี่ยนแปลงแก้ไข 7 รายการดังนี้

T = { add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

alter(Class, Shop, (file, Shop.java, ...)))

}

ข้อมูลทรานแซคชั่นทั้งหมดที่ได้จากส่วนนี้คือข้อมูลทรานแซคชั่นของการเปลี่ยนแปลงแก้ไขที่จะนำไปใช้ในการสร้างกฎความสัมพันธ์ในส่วนที่ 3 นอกจากนั้นทรานแซคชั่นเหล่านี้จะถูกนำไปวิเคราะห์เพื่อคัดเลือกขึ้นมาสร้างเป็นข้อสอบถามสำหรับการทดสอบทั้ง 3 สถานการณ์ในส่วนที่ 2 ด้วย

3.6.2 การสร้างข้อสอบถาม

การสร้างข้อสอบถาม คือ ส่วนที่นำทรานแซคชั่นของการเปลี่ยนแปลงแก้ไขทั้งหมดที่มาจากส่วนการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟ์มาคัดเลือกทรานแซคชั่นจำนวนหนึ่งที่มีความหมายทางสถิติ เรียกว่าทรานแซคชั่นชุดทดสอบ (Test set) เพื่อ

นำทรานแซคชั่นเหล่านั้นมาใช้ในการทดสอบ หลังจากนั้นจึงนำทรานแซคชั่นที่คัดเลือกไว้ไปสร้างเป็นข้อสอบถาม (Query) สำหรับการทดสอบสถานการณ์ 3 สถานการณ์ คือ 1) ข้อสอบถามสำหรับการทดสอบสถานการณ์การนำทาง 2) ข้อสอบถามสำหรับการทดสอบสถานการณ์การป้องกันการเกิดข้อผิดพลาด และ 3) ข้อสอบถามสำหรับการทดสอบสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว ตามข้อกำหนดที่กำหนดไว้ในหัวข้อตัวแปรควบคุม

ข้อมูลเข้า คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (เฉพาะตารางชื่อ Revisions และตารางชื่อ Transactions)

ข้อมูลออก คือ ข้อสอบถามสำหรับการทดสอบสถานการณ์การนำทาง ข้อสอบถามสำหรับการทดสอบสถานการณ์การป้องกันการเกิดข้อผิดพลาด และข้อสอบถามสำหรับการทดสอบสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว (ตารางชื่อ Queries)

กระบวนการทำงานของการสร้างข้อสอบถามสำหรับแต่ละสถานการณ์มีข้อกำหนดในการสร้างที่แตกต่างกันออกไป ข้อกำหนดในการสร้างข้อสอบสำหรับทั้ง 3 สถานการณ์สามารถแบ่งออกเป็น 2 ขั้นตอนย่อยดังนี้

1) การเลือกทรานแซคชั่นชุดทดสอบ
2) การสร้างข้อสอบถามแต่ละสถานการณ์

- **การเลือกทรานแซคชั่นชุดทดสอบ**

ข้อมูลซอฟต์แวร์อาร์ไคฟ์ที่นำมาใช้ในการวิจัยนี้คือข้อมูลซอฟต์แวร์อาร์ไคฟ์จากโครงการพัฒนาซอฟต์แวร์ทางการบัญชีชื่อเคมายมันนี่ (KMyMoney) ที่มี Thomas Baumgart และ Michael Edwardes เป็นผู้ก่อตั้งโครงการและปัจจุบันมีนักพัฒนาในโครงการนี้ทั้งหมด 10 คน ช่องทางการติดต่อสื่อสารระหว่างนักพัฒนาภายในโครงการคือการใช้ระบบไออาร์ซี (IRC Channel) ที่ต้องเข้าสู่ระบบโดยใช้รหัสผู้ใช้ (Username) และรหัสผ่าน (Password) ของนักพัฒนาซึ่งต้องร้องขอและได้รับอนุมัติในการร้องขอเพื่อเข้าร่วมเป็นนักพัฒนาของโครงการ นอกจากนั้นในกลุ่มนักพัฒนาทั้งหมดมีเพียง Thomas Baumgart และ Martin Preuss ที่เปิดเผยข้อมูลส่วนตัว ด้วยเหตุนี้ทำให้ผู้วิจัยมีข้อจำกัดในการสร้างข้อสอบถาม (Query) ที่ได้รับการประเมินชุดทดสอบจากผู้เชี่ยวชาญ (ในกรณีนี้ ผู้เชี่ยวชาญคือนักพัฒนาที่อยู่ในโครงการนี้) ผู้วิจัยจึงใช้วิธีการเลือก

ทรานแซคชั่นตัวแทนที่มีความหมายทางสถิติมาจำนวนหนึ่งเพื่อนำทรานแซคชั่นเหล่านั้นมาสร้างเป็นข้อสอบถามที่ใช้ในการวิจัยครั้งนี้

ทรานแซคชั่นชุดทดสอบที่ผู้วิจัยเลือกมาเป็นตัวแทนเพื่อสร้างข้อสอบถามสำหรับการทดสอบนี้หรือที่เรียกว่า ทรานแซคชั่นชุดทดสอบ (Test set) ผู้วิจัยกำหนดให้มีทรานแซคชั่นชุดทดสอบทั้งหมด 60 ทรานแซคชั่น ซึ่งถือว่าเพียงพอสำหรับข้อสอบถามในงานวิจัยทางด้านการค้นคืนข้อมูล (Information Retrieval) เนื่องจากข้อสอบถามในงานวิจัยทางด้านการค้นคืนข้อมูลควรมีอย่างน้อย 30 หน่วยทดสอบ (Baeza-Yates and Riberio-Neto, 1999) จากข้อจำกัดที่กล่าวไปข้างต้นการสร้างข้อสอบถามจึงต้องเลือกจากทรานแซคชั่นในฐานข้อมูลที่มีความหมายทางสถิติ โดยเลือกทรานแซคชั่นชุดดทดสอบ 60 ทรานแซคชั่นเป็นตัวแทนของทรานแซคชั่นที่มีขนาดสั้น กลาง ยาวและแบ่งเป็นทรานแซคชั่นที่พบบ่อยและพบไม่บ่อยด้วย ดังนั้นทรานแซคชั่นชุดทดสอบที่จะเลือกมาจะแบ่งได้เป็น 6 กลุ่มคือ 1) ทรานแซคชั่นขนาดสั้นและพบบ่อย 2) ทรานแซคชั่นขนาดสั้นและพบไม่บ่อย 3) ทรานแซคชั่นขนาดกลางและพบบ่อย 4) ทรานแซคชั่นขนาดกลางและพบไม่บ่อย 5) ทรานแซคชั่นขนาดยาวและพบบ่อย และ 6) ทรานแซคชั่นขนาดยาวและพบไม่บ่อย ผู้วิจัยเลือกกำหนดให้แต่ละกลุ่มมีจำนวนที่เท่ากันคือกลุ่มละ 10 ทรานแซคชั่น หรือสามารถแสดงได้ดังตารางต่อไปนี้

ตารางที่ 3-2 แสดงจำนวนของทรานแซคชั่นในแต่ละกลุ่มที่จะเลือกขึ้นมาสร้างเป็นข้อสอบถาม

การปรากฏ \ ขนาด	สั้น	กลาง	ยาว
พบบ่อย	10	10	10
พบไม่บ่อย	10	10	10

ขนาดของทรานแซคชั่นนั้นนับจากจำนวนของการเปลี่ยนแปลงแก้ไขทั้งหมดในทรานแซคชั่น ส่วนวิธีการนับจำนวนการปรากฏของรูปแบบทรานแซคชั่นจะนับจากทรานแซคชั่นที่เป็นซูเปอร์เซตของรูปแบบทรานแซคชั่นนั้นได้

รายละเอียดของการเลือกทรานแซคชั่นชุดทดสอบที่เฉพาะเจาะจงกับโครงการพัฒนาซอฟต์แวร์เคมายมันนี่ (KMyMoney) นั้นอธิบายไว้ในภาคผนวก ก และนำทรานแซคชั่นที่ได้มาทำการสร้างข้อสอบถามของแต่ละสถานการณ์ด้วยวิธีการสร้างข้อสอบถามที่อธิบายในหัวข้อถัดไป

- **การสร้างข้อสอบถามแต่ละสถานการณ์**

ข้อสอบถาม (Query) สำหรับงานวิจัยนี้ คือ เซตที่ประกอบไปด้วยเซตเหตุการณ์การเปลี่ยนแปลงแก้ไขและเซตผลลัพธ์ที่คาดไว้ โดยจะมีข้อสอบถามทั้งหมด 3 แบบสำหรับ 3 สถานการณ์ที่แตกต่างกันคือสถานการณ์การนำทาง สถานการณ์การป้องกันข้อผิดพลาด และสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว และถูกเขียนในรูปแบบ q = (Q, E) เช่นเดียวกับงานวิจัยของ Zimmermann และคณะในปีค.ศ. 2005 (Zimmermann et al., 2005) และงานวิจัยของ Methanias และคณะในปีค.ศ. 2009 (Methanias et al., 2009) ที่ทำการทดสอบประสิทธิภาพของการค้นหากฎความสัมพันธ์บนข้อมูลซอฟต์แวร์อาร์ไคฟ์สำหรับระบบให้คำแนะนำนักพัฒนาระหว่างการพัฒนาซอฟต์แวร์ ขั้นตอนของการสร้างข้อสอบถามในแต่ละสถานการณ์อธิบายพ้อมยกตัวอย่างได้ดังต่อไปนี้ (Zimmermann et al., 2005; Methanias et al., 2009)

กำหนดให้ ข้อสอบถาม (Query) q ใดๆ ถูกเขียนในรูปแบบ q = (Q, E)

โดยที่ เซต Q คือเซตเหตุการณ์ (Situation) ซึ่งเป็นเซตย่อยของเซตทรานแซคชั่น T

เซต E คือเซตผลลัพธ์ที่คาดไว้ (Expected Result) ซึ่งเป็นเซตย่อยของเซตทรานแซคชั่น T

และเท่ากับเซต T - Q

เซต T คือทรานแซคชั่นที่ประกอบด้วยรายการของการเปลี่ยนแปลงแก้ไขจำนวน |T| รายการ

สถานการณ์การนำทาง (Navigation)

ข้อสอบถามสำหรับสถานการณ์การนำทาง กำหนดให้มีข้อสอบถามทั้งหมด |T| ข้อสอบถามต่อ 1 ทรานแซคชั่น โดยที่ในแต่ละข้อสอบถามนั้นมีสมาชิกในเซตเหตุการณ์ Q เพียง 1 รายการคือรายการ e โดยที่รายการ e เป็นสมาชิกของ ทรานแซคชั่น T และสมาชิกในเซตผลลัพธ์ที่คาดไว้ E ก็คือรายการอีก |T| - 1 รายการที่เหลือที่ไม่ใช่รายการ e (Zimmermann et al., 2005)

สำหรับทรานแซคชั่นชุดทดสอบนั้นประกอบด้วยรายการการเปลี่ยนแปลงแก้ไขทั้งหมด 7 รายการทำให้ได้ข้อสอบถามสำหรับทรานแซคชั่นชุดทดสอบทั้งหมด 7 ข้อสอบถาม โดยที่ในแต่ละ

ข้อสอบถามนั้นมีเซตเหตุการณ์ Q ที่มีสมาชิกเพียง 1 รายการคือ e โดยที่ e เป็นสมาชิกของทรานแซคชั่น T ข้อสอบถามสำหรับการทดสอบสถานการณ์การนำทางจากทรานแซคชั่นชุดทดสอบแสดงดังต่อไปนี้

กำหนดให้ q^n_i คือ ข้อสอบถามสำหรับการทดสอบสถานการณ์การนำทางข้อสอบถามที่ i ในรูป $q^n_i = (Q, E)$

q^n_1 = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)) } — เซตเหตุการณ์

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))) ,

alter(method, add(), (Class, Sale, (file, Sale.java, ...))) ,

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))) ,

alter(Class, Sale, (file, Sale.java, ...)) , — เซตผลลัพธ์ที่คาดไว้

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))) ,

alter(Class, Shop, (file, Shop.java, ...))) }

)

q^n_2 = ({ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...)))},

{ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))) })

q^n_3 = ({ alter(method, add(), (Class, Sale, (file, Sale.java, ...)))},

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))) })

q^n_4 = ({ alter(method, remove(), (Class, Sale, (file, Sale.java, ...)))},

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))) })

q^n_5 = ({ alter(Class, Sale, (file, Sale.java, ...))},

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))) })

q^n_6 = ({ alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...)))},

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(Class, Shop, (file, Shop.java, ...))) })

q^n_7 = ({ alter(Class, Shop, (file, Shop.java, ...)))},

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

add_to(Class, IBuffer, (file, IBuffer.java, ...)) })

ตัวอย่างระเบียนของตารางชื่อ Queries ในการบันทึกข้อสอบถาม q^n_1

QueryID	QueryType	QAntcSet (Ref:RevisionID)	QConqSet (Ref:RevisionID)
1	Navgation	11	34, 37, 46, 47, 51, 52

สถานการณ์การป้องกันการเกิดข้อผิดพลาด (Error Prevention)

ข้อสอบถามสำหรับสถานการณ์การป้องกันการเกิดข้อผิดพลาด กำหนดให้มีข้อสอบถามทั้งหมด |T| ข้อสอบถามต่อ 1 ทรานแซคชั่น โดยที่ในแต่ละข้อสอบถามนั้นมีสมาชิกในเซต

เหตุการณ์ Q เท่ากับ |T| - 1 รายการนั่นคือ สมาชิกในเซตเหตุการณ์ Q คือสมาชิกในทรานแซคชั่น T ทุกรายการยกเว้นรายการ e โดยที่รายการ e เป็นสมาชิกของทรานแซคชั่น T และสมาชิกในเซตผลลัพธ์ที่คาดไว้ E มี 1 รายการคือรายการ e นั่นเอง (Zimmermann et al., 2005)

สำหรับทรานแซคชั่นชุดทดสอบนั้นทำให้ได้ข้อสอบถามทั้งหมด 7 ข้อสอบถาม โดยที่ในแต่ละข้อสอบถามนั้นมีเซตเหตุการณ์ Q ที่มีจำนวนสมาชิกเท่ากับ 6 รายการนั่นคือ สมาชิกในเซตเหตุการณ์ Q คือสมาชิกในทรานแซคชั่น T ทุกรายการยกเว้น e โดยที่ e เป็นสมาชิกของทรานแซคชั่น T ข้อสอบถามสำหรับการทดสอบสถานการณ์การนำทางจากทรานแซคชั่นชุดทดสอบแสดงดังต่อไปนี้

กำหนดให้ q^p_i คือ ข้อสอบถามสำหรับการทดสอบสถานการณ์การป้องกันการเกิดข้อผิดพลาดข้อสอบถามที่ i ในรูป $q^p_i = (Q, E)$

q^p_1 = (

{ add_to(Class, IBuffer, (file, IBuffer.java, ...)) ,

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))) ,

alter(method, add(), (Class, Sale, (file, Sale.java, ...))) ,

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))) ,

alter(Class, Sale, (file, Sale.java, ...)) ,

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ..))) } ,

เซตเหตุการณ์

{ alter(Class, Shop, (file, Shop.java, ...))) }

เซตผลลัพธ์ที่คาดไว้

)

q^p_2 = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(Class, Shop, (file, Shop.java, ...))) },

{ alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...)))})

q^p_3 = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))) },

{ alter(Class, Sale, (file, Sale.java, ...))})

q^p_4 = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))) },

{ alter(method, remove(), (Class, Sale, (file, Sale.java, ...)))})

q^p_5 = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))) },

{ alter(method, add(), (Class, Sale, (file, Sale.java, ...)))})

q^p_6 = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(Class, Shop, (file, Shop.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))) },

{ alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...)))})

q^p_7 = ({ alter(Class, Shop, (file, Shop.java, ...))),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, add(), (Class, Sale, (file, Sale.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))) },

{ add_to(Class, IBuffer, (file, IBuffer.java, ...))})

ตัวอย่างระเบียนของตารางชื่อ Queries ในการบันทึกข้อสอบถาม q^{p}_{1}

QueryID	QueryType	QAntcSet (Ref:RevisionID)	QConqSet (Ref:RevisionID)
35	Prevention	34, 37, 46, 47, 51, 52	11

สถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว (Closure)

ข้อสอบถามสำหรับสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว กำหนดให้มีข้อสอบถามทั้งหมด 1 ข้อสอบถามต่อ 1 ทรานแซคชั่น โดยที่ในแต่ละข้อสอบถามนั้นมีสมาชิกในเซตเหตุการณ์ Q เท่ากับ |T| รายการนั่นคือ สมาชิกในเซตเหตุการณ์ Q คือสมาชิกทุกรายการในทรานแซคชั่น T และเซตผลลัพธ์ที่คาดไว้ E เป็นเซตว่าง (Zimmermann et al., 2005)

สำหรับทรานแซคชั่นชุดทดสอบนั้นทำให้ได้ข้อสอบถามทั้งหมด 1 ข้อสอบถาม โดยที่ในแต่ละข้อสอบถามนั้นมีเซตเหตุการณ์ Q ที่มีจำนวนสมาชิกเท่ากับ 7 รายการนั่นคือ สมาชิกในเซตเหตุการณ์ Q คือสมาชิกทุกรายการในทรานแซคชั่น T ข้อสอบถามสำหรับการทดสอบสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วจากทรานแซคชั่นชุดทดสอบแสดงดังต่อไปนี้

กำหนดให้ q^{c}_{i} คือ ข้อสอบถามสำหรับการทดสอบสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วข้อสอบถามที่ i

q^{c}_{1} = ({ add_to(Class, IBuffer, (file, IBuffer.java, ...)) ,
alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),
alter(method, add(), (Class, Sale, (file, Sale.java, ...))) , เซตเหตุการณ์
alter(method, remove(), (Class, Sale, (file, Sale.java, ...))) ,
alter(Class, Sale, (file, Sale.java, ...)) ,
alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),
alter(Class, Shop, (file, Shop.java, ...))) } ,
ø เซตผลลัพธ์ที่คาดไว้

)

ตัวอย่างระเบียนของตารางชื่อ Queries ในการบันทึกข้อสอบถาม q^p_1

QueryID	QueryType	QAntcSet (Ref:RevisionID)	QConqSet (Ref:RevisionID)
41	Prevention	11, 34, 37, 46, 47, 51, 52	*null*

ข้อสอบถามทั้งหมดที่ได้มานั้นแบ่งออกเป็น 3 ชุด ตามสถานการณ์ต่างกัน 3 สถานการณ์ เพื่อนำไปใช้ในการทดสอบของแต่ละสถานการณ์ในส่วนที่ 4 (หัวข้อ 3.6.4)

3.6.3 การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบ

การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบ คือ ขั้นตอนที่นำทรานแซคชั่นของการเปลี่ยนแปลงแก้ไขทั้งหมดมาทำเหมืองข้อมูลเพื่อค้นหากฎความสัมพันธ์ของการเปลี่ยนแปลงแก้ไข โดยใช้ขั้นตอนวิธีของ Zimmermann และคณะ (Zimmermann et al., 2005) ที่อธิบายอย่างละเอียดไว้ในบทที่ 2 หัวข้อ 2.8.2 มาประยุกต์ใช้สำหรับการค้นหากฎความสัมพันธ์ทั้ง 2 ตัวแบบ

ข้อมูลเข้า คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (เฉพาะตารางชื่อ Revisions และตารางชื่อ Transactions)

ข้อมูลออก คือ กฎความสัมพันธ์ที่ได้จากการทำเหมืองข้อมูลทั้ง 2 ตัวแบบรวมถึงค่าสนับสนุนค่าความเชื่อมั่น/ค่าความเชื่อมั่นใหม่ของกฎความสัมพันธ์นั้นๆ (ตารางชื่อ Rules)

กระบวนการทำงานของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบ นั้นคือการนำขั้นตอนวิธีในการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์ที่ชื่อว่า ขั้นตอนวิธีอพริโอริ (Apriori algorithm) ที่ถูกนำเสนอโดย Agrawal และ Srikant (Agrawal and Srikant, 1994)

เนื่องจากงานวิจัยนี้สนใจศึกษาว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 หรือการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบค่าสนับสนุน-ค่าความเชื่อมั่นใหม่

ของ Liu และคณะ (Liu et al., 2008) สามารถเพิ่มประสิทธิภาพของระบบให้คำแนะนำนักพัฒนาในระหว่างการพัฒนาซอฟต์แวร์ของไอดีอี (IDE: integrated development environment) ได้หรือไม่ ดังนั้นในขั้นตอนการระบุความน่าสนใจของกฎความสัมพันธ์แต่ละกฎในทั้ง 2 ตัวแบบของการค้นหากฎความสัมพันธ์จึงแตกต่างกันดังนี้

- การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 กำหนดให้ใช้การคำนวณค่าสนับสนุนนับ (Support Count) และค่าความเชื่อมั่น (Confidence) ในการระบุความน่าสนใจของกฎความสัมพันธ์ สำหรับการทดสอบนี้ผู้วิจัยกำหนดให้ค่าสนับสนุนนับขั้นต่ำ (Minimum Support Count) เท่ากับ 3 และค่าความเชื่อมั่นขั้นต่ำ (Minimum Confidence) เท่ากับ 0.1
- การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 กำหนดให้ใช้การคำนวณค่าสนับสนุนนับและค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) ในการระบุความน่าสนใจของกฎความสัมพันธ์ สำหรับการทดสอบนี้ผู้วิจัยกำหนดให้ค่าสนับสนุนนับขั้นต่ำเท่ากับ 3 และค่าความเชื่อมั่นใหม่ของ Liu และคณะ (Liu et al., 2008) ขั้นต่ำเท่ากับ 0.1

สำหรับตัวอย่างของการทำเหมืองข้อมูลด้วยเทคนิคค้นหากฎความสัมพันธ์ทั้ง 2 ตัวแบบนั้น เนื่องจากผู้วิจัยไม่สามารถสมมุติทรานแซคชั่นทั้งหมดออกมาได้ได้ ผู้วิจัยจึงสมมุติกฎความสัมพันธ์จากการค้นหากฎความสัมพันธ์ ที่เป็นผลลัพธ์ที่ได้มาจากกระบวนการข้างต้น เพื่อประโยชน์ในการอ้างอิงถึงในส่วนต่อไป ดังนี้

กำหนดให้ R คือ เซตของกฎความสัมพันธ์ที่แต่ละรายการอยู่ในรูปแบบ Q -> {x} s;c

Q คือ เซตเหตุการณ์ที่ประกอบด้วยรายการการเปลี่ยนแปลงแก้ไขตั้งแต่ 1 รายการขึ้นไป

x คือ รายการการเปลี่ยนแปลงแก้ไขใดๆ

s คือ ค่าสนับสนุนนับของกฎความสัมพันธ์ Q -> {x}

c คือ ค่าความเชื่อมั่นของกฎความสัมพันธ์ Q -> {x}

R = { alter(method, add(), ...) -> alter(method, getString(), ...) 6:0.5,

alter(method, add(), ...) -> alter(Class, Shop, ...) 5:0.71,

alter(method, getString(), ...) -> alter(method, getString(), ...) 5:0.63,

alter(method, add(), ...) -> alter(method, remove(), ...) 3:0.5,

alter(method, remove(), ...) -> alter(Class, Shop, ...) 2:0.29,

alter(Class, Sale, ...) -> alter(method, remove(), ...) 2:0.5,

alter(method, getDescription(), ...) -> alter(method, getString(), ...)1:0.14

}

ตัวอย่างระเบียนของตารางชื่อ Rules ในการบันทึกกฎความสัมพันธ์แรกในเซตข้างต้น

RuleID	Model	RAntcSet (Ref:RevisionID)	RConqSet (Ref:RevisionID)	Support	Confidence
1	1	11	34	6	0.5

3.6.4 การสร้างเซตของคำแนะนำสำหรับเหตุการณ์

การสร้างเซตของคำแนะนำสำหรับเหตุการณ์ คือ การนำเซตของกฎความสัมพันธ์ R สำหรับเหตุการณ์ Q มาสร้างเป็นเซตของคำแนะนำ (Suggestions) นำเสนอให้กับนักพัฒนาเมื่อนักพัฒนาได้ทำให้เกิดเหตุการณ์ Q ใดๆขึ้นมา เซตของคำแนะนำสำหรับเหตุการณ์ Q สามารถนิยามให้อยู่ในรูปของการยูเนียน (Union) ของเซตรายการที่ตามมาของกฎความสัมพันธ์ R ที่มีเซตรายการที่มาก่อน ตรงกับเซตเหตุการณ์ Q ได้ ดังต่อไปนี้ (Zimmermann et al., 2005)

$$apply_R(Q) = \mathrm{U}_{(\mathrm{Q}\rightarrow\{x_2\})\in R}\, x_2$$

ในงานวิจัยของ Zimmermann และคณะ (Zimmermann et al., 2005) ได้ตั้งข้อสันนิษฐานไว้ว่า การให้คำแนะนำในการเปลี่ยนแปลงแก้ไขกับนักพัฒนานั้น คำแนะนำที่จะได้รับความสนใจก็คือคำแนะนำที่อยู่ใน 10 อันดับแรก ดังนั้นในการสร้างเซตของคำแนะนำจึงควรให้ความสนใจกฎความสัมพันธ์ที่อยู่ใน 10 อันดับแรกโดยเรียงจากค่าสนับสนุนนับและค่าความ

เชื่อมั่นเท่านั้น ดังนั้นผู้วิจัยจึงกำหนดสมการในการสร้างเซตของคำแนะนำดังแสดงในสมการต่อไปนี้

กำหนดให้ q คือ ข้อสอบถาม (Query) ที่ประกอบด้วยเซตเหตุการณ์ (Situation) Q และเซตผลลัพธ์ที่คาดไว้ (Expected Result) E และเขียนให้อยู่ในรูป q = (Q, E)

R คือ เซตของกฎความสัมพันธ์ที่อยู่ในรูปแบบ Q→{x} โดยที่ x คือรายการการเปลี่ยนแปลงแก้ไข และกำหนดให้ R_{10} คือเซตของกฎความสัมพันธ์ที่มีระดับความน่าสนใจสูงสุด 10 กฎแรกซึ่งเรียงลำดับด้วยค่าความเชื่อมั่น โดยที่ $R_{10} \subset R$

A_q คือ เซตของรายการการเปลี่ยนแปลงแก้ไข x ที่ได้จากกฎความสัมพันธ์ในเซต R_{10} ที่สอดคล้องกับเซตเหตุการณ์ Q ของข้อสอบถาม q ซึ่งสามารถเขียนในรูป $A_q = apply_{R_{10}}(Q)$

$$A_q = apply_{R_{10}}(Q)$$

ข้อมูลเข้า คือ เซตของกฎความสัมพันธ์ และเซตเหตุการณ์ของข้อสอบถามชุดทดสอบ (ตารางชื่อ Rules และตารางชื่อ Queries)

ข้อมูลออก คือ เซตของคำแนะนำสำหรับเหตุการณ์ที่นำมาทดสอบ (ตารางชื่อ Suggestions)

กระบวนการทำงานของการสร้างเซตของคำแนะนำสำหรับเหตุการณ์ มีขั้นตอนวิธีดังต่อไปนี้

1) เลือกเหตุการณ์ Q ที่ต้องการนำมาหาคำแนะนำ
2) ค้นหากฎความสัมพันธ์ที่มีเซตรายการที่มาก่อนตรงกับเซตเหตุการณ์ Q มาจากเซตของกฎความสัมพันธ์ทั้งหมด
3) นำเซตรายการที่ตามมาของกฎความสัมพันธ์ที่ได้จากข้อที่ 2 มารวมกัน (Union) เป็นเซตใหม่ให้ชื่อว่า เซตของคำแนะนำสำหรับเหตุการณ์ Q โดยที่แต่ละรายการของเซต

คำแนะนำนี้จะเรียงลำดับตามค่าสนับสนุนนับและค่าความเชื่อมั่นของกฎความสัมพันธ์นั้นๆ

จากเซตของกฎความสัมพันธ์ที่ได้มาจากขั้นตอนที่แล้ว ตัวอย่างของการสร้างเซตของคำแนะนำสำหรับเหตุการณ์ alter(method, add(), (Class, Sale, (file, Sale.java, ...))) แสดงได้ดังต่อไปนี้

1) เหตุการณ์ที่เกิดขึ้นคือ เหตุการณ์ alter(method, add(), (Class, Sale, (file, Sale.java, ...)))
2) จากเซตของกฎความสัมพันธ์ที่ได้มาจากขั้นตอนที่แล้วมีกฎความสัมพันธ์ที่มีเซตรายการที่มาก่อน เป็น alter(method, add(), (Class, Sale, (file, Sale.java, ...))) อยู่ทั้งหมด 3 กฎความสัมพันธ์ดังนี้

alter(method, add(), ...) -> alter(method, getString(), ...) 6:0.5

alter(method, add(), ...) -> alter(Class, Shop, ...) 5:0.71

alter(method, add(), ...) -> alter(method, remove(), ...) 3:0.5

3) นำเซตรายการที่ตามมาของกฎความสัมพันธ์ทั้ง 3 กฎความสัมพันธ์ข้างต้นมารวมกันเป็นเซตของคำแนะนำเมื่อเกิดเห็นการ เหตุการณ์ alter(method, add(), (Class, Sale, (file, Sale.java, ...))) ได้ดังต่อไปนี้

A_q = { alter(method, getString(), ...),
alter(Class, Shop, ...),
alter(method, remove(), ...)
}

ตัวอย่างระเบียนของตารางชื่อ Suggestions ในการบันทึกเซตของคำแนะนำข้างต้น

SuggestionID	ForSituation (Ref:RevisionID)	SuggestionSet (Ref:RevisionID)
1	11	34, 46, 47

3.6.5 การประเมินผลการทดสอบ

เมื่อได้ทำการทดสอบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์เสร็จสิ้นแล้ว ขั้นตอนต่อไปนี้ก็คือการประเมินผลการทดสอบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ จุดมุ่งหมายหลักของการทดสอบของงานวิจัยนี้คือ การเปรียบเทียบค่าประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบที่ 1 กับตัวแบบที่ 2 ในสถานการณ์ที่ต่างกัน 3 สถานการณ์ ดังนั้นหลังจากการทดสอบเสร็จสิ้นผู้วิจัยจึงต้องนำผลการทดสอบเหล่านั้นมาคำนวณหาค่าประสิทธิภาพ ซึ่งงานวิจัยนี้จะใช้ค่าเอฟเมสเซอร์ (F-measure) มาเป็นค่าแสดงประสิทธิภาพของสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาด และค่าผลสะท้อนกลับ (Feedback) มาเป็นค่าแสดงประสิทธิภาพของสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว

ข้อมูลเข้า คือ เซตของคำแนะนำสำหรับเหตุการณ์ของการทดสอบทั้ง 6 การทดสอบ (ตารางชื่อ Suggestions)

ข้อมูลออก คือ ค่าประสิทธิภาพของการทดสอบทั้ง 6 การทดสอบ (แฟ้มข้อมูลประเภทข้อความที่บันทึกค่าประสิทธิภาพของการทดสอบทั้งหมดแยกตามการทดสอบ)

หลังจากขั้นตอนการสร้างเซตของคำแนะนำสำหรับเหตุการณ์แล้ว ผู้วิจัยจะได้เซตของคำแนะนำสำหรับเหตุการณ์ใดๆ ที่สร้างมาจากกฎความสัมพันธ์ของทรานแซคชั่นทั้งหมด เซตของคำแนะนำนั้นจะถูกนำไปเปรียบเทียบกับเซตผลลัพธ์ที่คาดไว้ของชุดทดสอบ สำหรับในสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาดจะนำมาคำนวณค่าความถูกต้อง (Precision) ค่าเรียกคืน (Recall) และค่าเอฟเมสเซอร์ (F-measure)

ในปี 2005 งานวิจัยของ Zimmermann และคณะ (Zimmermann et al., 2005)) ทำการเปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลบนข้อมูลซอฟต์แวร์อาร์ไคฟว์กับโครงการพัฒนาระบบปฏิบัติการคอมพิวเตอร์ต่างๆ (Operating System) และกล่าวว่า จุดมุ่งหมายของการทดสอบระบบให้คำแนะนำนักพัฒนาในระหว่างการพัฒนาซอฟต์แวร์คือค่าความถูกต้องที่สูง (ค่าใกล้เคียง 1) และค่าเรียกคืนที่สูง (ค่าใกล้เคียง 1) นั่นคือต้องการให้ระบบสามารถแนะนำคำแนะนำทั้งหมด (ค่าเรียกคืนเท่ากับ 1) และแต่ละคำแนะนำนั้นถูกต้องหรือตรงกับเซตผลลัพธ์ที่คาดไว้ทั้งหมด (ค่าความถูกต้องเท่ากับ 1) ดังนั้นงานวิจัยของ Zimmermann และคณะจึงใช้

ค่าเฉลี่ยฮาร์โมนินิคของค่าความถูกต้องและค่าเรียกคืน (Harmonic mean of Precision and Recall) หรือค่าเอฟเมสเซอร์ที่ให้น้ำหนักของค่าความถูกต้องและค่าเรียกคืนอย่างสมดุล เป็นค่าประเมินประสิทธิภาพของการทำเหมืองข้อมูล (Zimmermann et al., 2005)

ต่อมาในปี 2009 งานวิจัยของ Methanias และคณะ (Methanias et al., 2009) ทำการเปรียบเทียบประสิทธิภาพของการทำเหมืองข้อมูลบนข้อมูลซอฟต์แวร์อาร์ไคฟว์ของโครงการซอฟต์แวร์สิ่งแวดล้อมอุตสาหกรรม (Industrial Environment) ใน 3 สถานการณ์เช่นเดียวกัน และแนะนำให้ใช้ค่าเอฟเมสเซอร์ที่ให้น้ำหนักของค่าความถูกต้องและค่าเรียกคืนอย่างสมดุลสำหรับการประเมินประสิทธิภาพในสถานการณ์การนำทางและสถานการณ์การป้องกันข้อผิดพลาด (Methanias et al., 2009)

ดังนั้นงานวิจัยนี้จึงใช้ค่าเอฟเมสเซอร์ที่ $\beta = 1$ หรือค่าเอฟเมสเซอร์ที่ให้น้ำหนักของค่าความถูกต้องและค่าเรียกคืนอย่างสมดุลมาใช้ในการวัดประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ในสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาด ค่าเอฟเมสเซอร์ (F-measure) ที่ถ่วงน้ำหนักค่าความถูกต้องแต่ค่าเรียกคืนอย่างสมดุล แสดงดังสมการต่อไปนี้

กำหนดให้ F_1 ค่าเอฟเมสเซอร์ (F-measure) ที่ถ่วงน้ำหนักค่าความถูกต้องแต่ค่าเรียกคืนอย่างสมดุล

Precision ค่าความถูกต้อง

Recall ค่าเรียกคืน

q คือ ข้อสอบถาม (Query) ที่ประกอบด้วยเซตเหตุการณ์ (Situation) Q และเซตผลลัพธ์ที่คาดไว้ (Expected Result) E และเขียนให้อยู่ในรูป q = (Q, E)

R คือ เซตของกฎความสัมพันธ์ที่อยู่ในรูปแบบ Q→{x} โดยที่ x คือรายการการเปลี่ยนแปลงแก้ไข และกำหนดให้ R_{10} คือเซตของกฎความสัมพันธ์ที่มีระดับความน่าสนใจสูงสุด 10 กฎแรกซึ่งเรียงลำดับด้วยค่าความเชื่อมั่น โดยที่ $R_{10} \subset R$

A_q คือ เซตของรายการการเปลี่ยนแปลงแก้ไข x ที่ได้จากกฎความสัมพันธ์ในเซต R_{10} ที่สอดคล้องกับเซตเหตุการณ์ Q ของข้อสอบถาม q ซึ่งสามารถเขียนในรูป $A_q = apply_{R_{10}}(Q)$ เสมอ หรือเรียกว่า เซตของคำแนะนำ

$|A_q \cap E|$ คือ จำนวนรายการการเปลี่ยนแปลงแก้ไขในเซตคำแนะนำที่ตรงกับรายการการเปลี่ยนแปลงแก้ไขที่อยู่ในเซตผลลัพธ์ที่คาดไว้

$|A_q|$ คือ จำนวนรายการการเปลี่ยนแปลงแก้ไขที่อยู่ในเซตของคำแนะนำ

$|E|$ คือ จำนวนรายการการเปลี่ยนแปลงแก้ไขที่อยู่ในเซตผลลัพธ์ที่คาดไว้

$$precision = \frac{|A_q \cap E|}{|A_q|}, \qquad recall = \frac{|A_q \cap E|}{|E|}$$

และ $$F_1 = \frac{2*precision*recall}{precision+recall}$$

เนื่องจากค่าเอฟเมสเซอร์นั้นจะอยู่ในช่วง 0 ถึง 1 ค่าเอฟเมสเซอร์ที่มีค่าเป็น 0 ในงานวิจัยนี้จะหมายถึงประสิทธิภาพของการค้นหากฎความสัมพันธ์ต่ำหรือรายการการเปลี่ยนแปลงแก้ไขที่ถูกดึงขึ้นมาไม่ตรงกับรายการการเปลี่ยนแปลงแก้ไขใดๆในเซตผลลัพธ์ที่คาดไว้ของข้อสอบถามเลย และค่าเอฟเมสเซอร์ที่มีค่าเท่ากับ 1 ในงานวิจัยนี้จะหมายถึงประสิทธิภาพของการค้นหากฎความสัมพันธ์สูงหรือรายการการเปลี่ยนแปลงแก้ไขที่ถูกดึงขึ้นมาตรงกับเซตผลลัพธ์ที่คาดไว้ของข้อสอบถามทุกรายการ

ตัวอย่างเช่น สมมุติข้อมูลสอบถามชุดทดสอบสำหรับสถานการณ์การนำทาง 1 ข้อสอบถาม และเซตของคำแนะนำสำหรับเหตุการณ์ alter(method, add(), (Class, Sale, (file, Sale.java, ...))) ดังนี้

กำหนดให้ q^n คือ ข้อสอบถามชุดทดสอบสำหรับสถานการณ์การนำทาง ในรูป $q^n = (Q, E)$

A_q คือ เซตของคำแนะนำสำหรับเหตุการณ์ alter(method, add(), (Class, Sale, (file, Sale.java, ...)))

q^n = ({ alter(method, add(), (Class, Sale, (file, Sale.java, ...))) },

{ add_to(Class, IBuffer, (file, IBuffer.java, ...)),

alter(method, getString(), (Class, IBuffer, (file, IBuffer.java, ...))),

alter(method, remove(), (Class, Sale, (file, Sale.java, ...))),

alter(Class, Sale, (file, Sale.java, ...)),

alter(method, getDescription(), (Class, Shop, (file, Shop.java, ...))),

alter(Class, Shop, (file, Shop.java, ...))) })

A_q = { alter(method, getString(), ...),

alter(Class, Shop, ...),

del_from(Class, Product, ...),

alter(method, remove(), ...)

}

จากข้อสอบถามชุดทดสอบและเซตของคำแนะนำข้างต้น นำมาคำนวณค่าความถูกต้อง (Precision) ค่าเรียกคืน (Recall) และค่าเอฟเมสเซอร์ (F-measure) ได้ดังนี้

$$precision = \frac{|A_q \cap E|}{|A_q|} = \frac{3}{4} = 0.75$$

$$recall = \frac{|A_q \cap E|}{|E|} = \frac{3}{6} = 0.5$$

$$F_1 = \frac{2*precision*recall}{precision+recall} = \frac{2*0.75*0.5}{0.75+0.5} = 0.6$$

ค่าเอฟเมสเซอร์ที่คำนวณมาได้นั้นถ้ามีค่าใกล้เคียง 1 มากก็หมายความว่ายิ่งมีประสิทธิภาพมาก

สำหรับสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว เซตของคำแนะนำนั้นถูกนำไปเปรียบเทียบกับเซตผลลัพธ์ที่คาดไว้ของชุดทดสอบและค่าคำนวณผลสะท้อนกลับ (Feedback) ตามสมการต่อไปนี้

กำหนดให้ $|Z^*|$ คือ จำนวนข้อสอบถามที่อยู่ในเซตของข้อสอบถามที่มีเซตของคำแนะนำที่ไม่เป็นเซตว่าง ($|A_q| \neq 0$)

$|Z|$ คือ จำนวนข้อสอบถามทั้งหมด

$$feedback = \frac{|Z^*|}{|Z|}$$

เมื่อนำค่าผลสะท้อนกลับมาใช้วัดในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วจะสามารถแสดงให้เห็นถึงร้อยละของการเกิดการแจ้งเตือนที่ผิด (False alarm) หรือการให้คำแนะนำที่เป็นผลบวกลวง (False Positive) นั่นเอง เนื่องจากค่าผลสะท้อนกลับมีพิสัยอยู่ระหว่าง 0 กับ 1 ค่าผลสะท้อนกลับที่มีค่าเท่ากับ 0 หมายถึงไม่มีข้อสอบถามใดเลยที่ให้คำแนะนำที่เป็นผลบวกลวงออกมาในสถานการณ์นี้นั่นคือมีประสิทธิภาพดีที่สุด และค่าผลสะท้อนกลับที่มีค่าเท่ากับ 1 หมายถึงข้อสอบถามทั้งหมดให้คำแนะนำที่เป็นผลบวกลวงออกมานั่นคือมีประสิทธิภาพไม่ดีที่สุด ดังนั้นการวัดประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้นจะต้องเปรียบเทียบค่าผลสะท้อนกลับและค่าผลสะท้อนกลับที่น้อยกว่าจะมีความหมายว่ามีประสิทธิภาพดีกว่า

ตัวอย่างเช่น สมมุติให้มีข้อสอบถามในชุดทดสอบมีทั้งหมด 100 ข้อสอบถาม และสมมุติให้มีข้อสอบถามที่ได้เซตของคำแนะนำไม่เป็นเซตว่างทั้งหมด 66 ข้อสอบถาม จะสามารถคำนวณค่าคำนวณผลสะท้อนกลับ (Feedback) ได้ดังนี้

$$feedback = \frac{|Z^*|}{|Z|} = \frac{66}{100} = 0.66$$

การพิจารณาประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ สำหรับสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาดนั้นสามารถทำได้โดยการนำค่าเอฟเมสเซอร์ที่คำนวณได้มาเปรียบเทียบโดยใช้กราฟในรูปแบบที่เหมาะสม เพื่อง่ายต่อการพิจารณาเปรียบเทียบค่าเอฟเมสเซอร์ของการทำเหมืองข้อมูลด้วยรูปแบบทั้ง 2 ตัวแบบ สำหรับสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้นสามารถทำได้โดยการนำค่าคำนวณผลสะท้อนกลับที่คำนวณได้มาเปรียบเทียบค่ากันได้โดยตรง

3.6.6 การทดสอบสมมติฐาน

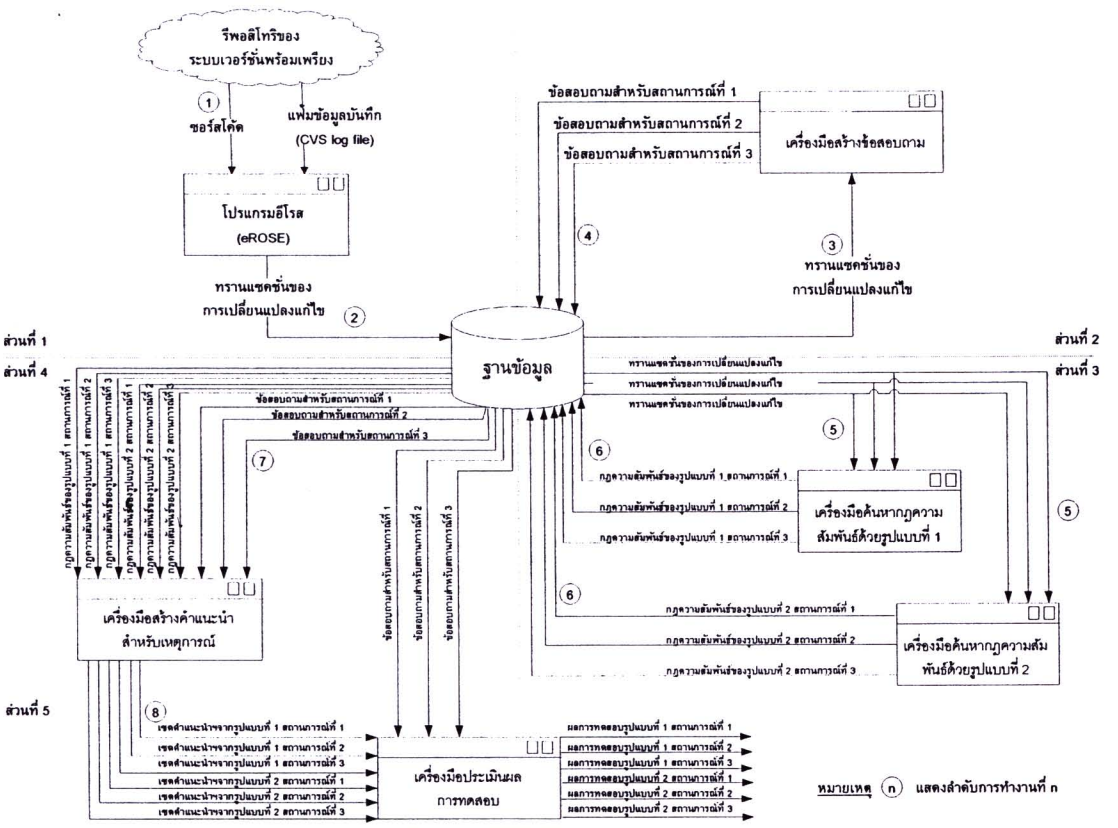
สำหรับกรณีของการทดสอบในสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาดนั้นใช้ค่าเอฟเมสเซอร์เป็นค่าที่แสดงถึงประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ เมื่อการทดสอบประสิทธิภาพเสร็จสิ้นแล้ว ทำให้ได้ค่าเอฟเมสเซอร์ออกมาเท่าจำนวนของข้อสอบถามชุดทดสอบที่สร้างขึ้นในขั้นตอนการสร้างข้อสอบถาม จากนั้นในขั้นตอนแรกจะตรวจสอบการแจกแจงของค่าประสิทธิภาพที่ได้มาว่ามีการแจกแจงปกติหรือไม่ ด้วยการใช้สถิติทดสอบ Kolmogorov-Smirnov เพื่อเลือกทางเลือกในการทดสอบสมมติฐานได้ว่าจะให้การทดสอบสมมติฐานแบบใช้พารามิเตอร์ (Parametric Test)

หรือแบบไม่อิงกับพารามิเตอร์ (Non Parametric Test) ถ้าผลการทดสอบพบว่าประชากรมีการแจกแจงแบบปกติ จึงใช้การวิเคราะห์โดยสถิติทดสอบที (t-test) เพื่อทดสอบสมมติฐานของผลต่างระหว่างค่าเฉลี่ยของค่าเอฟเมสเซอร์ของหน่วยทดลอง 2 กลุ่ม ถ้าค่า Sig. (Significance) ที่คำนวณได้น้อยกว่า 0.05 และค่าสถิติทีมากกว่า 0 จึงจะสามารถปฏิเสธ H_0 ได้ แต่ถ้าผลการแจกแจงประชากรพบว่ามีการแจกแจงไม่ปกติ ต้องใช้วิธีการทดสอบสมมติฐานแบบไม่อิงกับพารามิเตอร์ (Non Parametric Test) ต่อไป โดยในที่นี้คือการวิเคราะห์โดยสถิติทดสอบเครื่องหมายลำดับที่ของวิลคอกซันสำหรับการทดสอบแบบจับคู่ (The Wilcoxon Signed Rank Sum Test for the Matched Paired Difference) เพื่อทดสอบสมมติฐานของผลต่างระหว่างค่าเฉลี่ยของค่าเอฟเมสเซอร์ของหน่วยทดลอง 2 กลุ่ม ถ้าค่า Sig. (Significance) ที่คำนวณได้น้อยกว่า 0.05 และค่าสถิติซีมากกว่า 0 ในกรณีที่ผลการวิเคราะห์ตั้งอยู่บนพื้นฐานทางบวก (Based on positive ranks) จึงจะสามารถปฏิเสธ H_0 ได้

สำหรับการทดสอบในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้นใช้ค่าผลสะท้อนกลับ เป็นค่าที่แสดงถึงประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ เมื่อการทดสอบประสิทธิภาพเสร็จสิ้นแล้ว ทำให้ได้ค่าผลสะท้อนกลับออกมาหนึ่งค่าต่อหนึ่งการทดสอบ ค่าผลสะท้อนกลับที่ได้มานั้นแสดงให้เห็นถึงร้อยละของการเกิดการแจ้งเตือนที่ผิด (False Alarm) หรือการให้คำแนะนำที่เป็นผลบวกลวง (False Positive) นั่นเอง ดังนั้นการวัดประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้น ค่าผลสะท้อนกลับที่น้อยกว่าจะมีความหมายว่ามีประสิทธิภาพมากกว่า นั่นคือถ้าค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 มากกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 แล้วจึงสามารถปฏิเสธ H_0 ได้

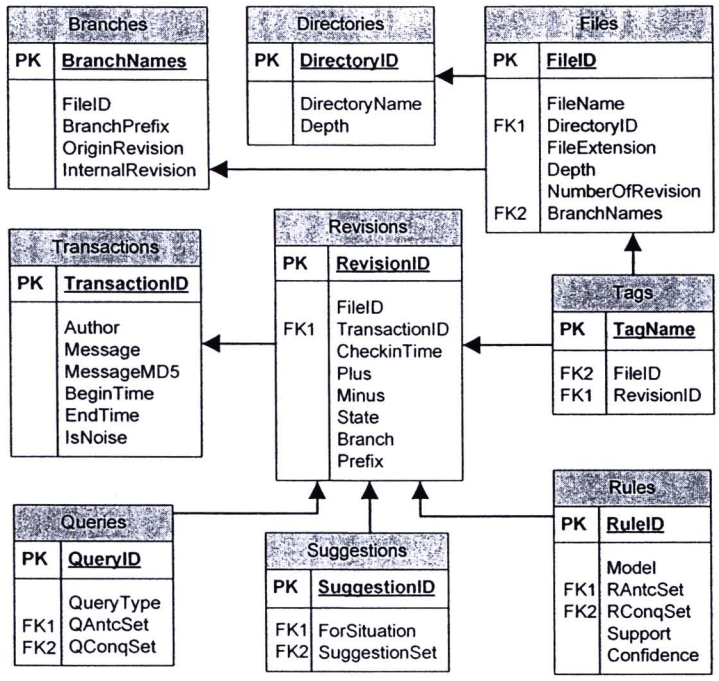
3.7 เครื่องมือที่ใช้ในงานวิจัย

ตามที่ได้กล่าวมาในหัวข้อแนวทางการทำวิจัยแล้วว่าผู้วิจัยได้พัฒนาเครื่องมือทดสอบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์วทั้ง 2 ตัวแบบใน 3 สถานการณ์ ผู้วิจัยเลือกใช้โปรแกรมประยุกต์อีโรส (eROSE) (Zimmermann et al., 2005) ร่วมกับสคริปท์ (Srcript) ที่ผู้วิจัยสร้างขึ้นมาเองด้วยภาษาพีเอชพี (PHP) และระบบจัดการฐานข้อมูลชื่อพีเอชพีมายแอดมิน (PHPMyAdmin Database Management System) ซึ่งเป็นฐานข้อมูลแบบเปิดสามารถนำมาใช้งานได้โดยไม่เสียค่าใช้จ่ายและเข้ากันได้ดีกับภาษาพีเอชพี (PHP) ผู้วิจัยออกแบบเครื่องมือทดสอบตามขั้นตอนในหัวข้อ 3.6 และแผนภาพในรูป 3-1 ภาพรวมของเครื่องมือการทดสอบทั้งหมดแบ่งออกเป็น 5 ส่วนดังนี้



รูปที่ 3-5 แสดงภาพรวมของเครื่องมือที่ใช้ในการทดสอบทดสอบการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์

การออกแบบฐานข้อมูลที่ใช้ในงานวิจัยนี้ ออกแบบตามฐานข้อมูลของโปรแกรมประยุกต์อีโรส (eROSE) (Zimmermann et al., 2005) เนื่องจากผู้วิจัยใช้บางส่วนของโปรแกรมประยุกต์อีโรสมาใช้ในขั้นตอนแรกของการวิจัย ฐานข้อมูลประกอบด้วยตารางทั้งหมด 9 ตาราง คือ ตารางชื่อ Directories ตารางชื่อ Files ตารางชื่อ Revisions ตารางชื่อ Transactions ตารางชื่อ Branches ตารางชื่อ Tags ตารางชื่อ Queries ตารางชื่อ Rules และตารางชื่อ Suggestions ตัวอย่างระเบียนของแต่ละตารางแสดงในหัวข้อ 3.6 ความสัมพันธ์ของแต่ละตารางแสดงดังแผนภาพต่อไปนี้



รูปที่ 3-6 แสดงแผนภาพอีอาร์ (ER Diagram) ฐานข้อมูลของเครื่องมือที่ใช้ในการทดสอบ

รายละเอียดของเครื่องมือทั้ง 5 เครื่องมือ แสดงดังต่อไปนี้

- **ส่วนของการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์**

ขั้นตอนแรกของการทดสอบคือการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ (Preparing Data for Mining in Software Archives) ของโครงการพัฒนาซอฟต์แวร์ชื่อเคมายมันนี่ (KMyMoney) ที่พัฒนาด้วยภาษาซีพลัสพลัส (C++) ในส่วนนี้ผู้วิจัยเลือกใช้ส่วนการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ซึ่งเป็นส่วนหนึ่งโปรแกรมประยุกต์อีโรส (eROSE) (Zimmermann et al., 2005) เนื่องจากโปรแกรมประยุกต์อีโรสสามารถรับรองการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลของโครงการพัฒนาซอฟต์แวร์ที่พัฒนาด้วยภาษาซีพลัสพลัส (C++) และยังเป็นเครื่องมือที่ถูกนำไปใช้งานวิจัยที่เกี่ยวข้องกับการวิเคราะห์ข้อมูลซอฟต์แวร์อาร์ไคฟว์เช่น งานวิจัยของ Zimmermann และคณะในปี 2005 (Zimmermann et al., 2005) และในงานวิจัยของ Methanias และคณะในปี 2009 (Methanias et al., 2009)

ขั้นตอนวิธีสำหรับส่วนการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์ของโปรแกรมประยุกต์อีโรสประกอบด้วย 4 ขั้นตอนคือ 1) การสกัดข้อมูล (Data Extraction) 2) การซ่อมแซมทรานแซคชั่น (Restoring Transactions) 3) การระบุการเปลี่ยนแปลงแก้ไขในระดับเอนทิตี้ (Mapping Changes to Entities) และ 4) การกำจัดสิ่งแปลกปลอม (Data Cleaning) อธิบายไว้อย่างละเอียดหัวข้อที่ 3.6.1 เมื่อได้ทรานแซคชั่นที่สมบูรณ์แล้วเก็บลงฐานข้อมูลของงานวิจัยเพื่อจัดเตรียมข้อมูลไว้ก่อนจะนำข้อมูลเหล่านี้ไปทำการทดสอบในส่วนการสร้างข้อสอบถามสำหรับการทดสอบสำหรับ 3 สถานการณ์ต่อไป

ข้อมูลเข้า คือ ข้อมูลซอฟต์แวร์อาร์ไคฟว์ซึ่งประกอบด้วย แฟ้มข้อมูลซอร์สโค้ดทั้งโครงการทุกเวอร์ชั่น และแฟ้มข้อมูลบันทึกของระบบคอนเคอเรนท์เวอร์ชั่น (CVS Log File) ตัวอย่างของข้อมูลเข้าแสดงดังรูป 3-2 ภายใต้หัวข้อ 3.6.1

ข้อมูลออก คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข ที่ประกอบด้วยตาราง 6 ตาราง คือตารางชื่อ Files ตารางชื่อ Directories ตารางชื่อ Tags ตารางชื่อ Branches ตารางชื่อ Revisions และตารางชื่อ Transactions ตัวอย่างของข้อมูลออกแสดงในหัวข้อ 3.6.1

- **ส่วนของการสร้างข้อสอบถามสำหรับการทดสอบสำหรับ 3 สถานการณ์**

ส่วนที่ 2 คือส่วนของการสร้างข้อสอบถามสำหรับการทดสอบสำหรับแต่ละสถานการณ์ทั้ง 3 สถานการณ์ การทำงานของส่วนนี้ประกอบด้วยส่วนย่อย 2 ส่วนคือ 1) ส่วนการเลือกทรานแซคชั่นชุดทดสอบ และ 2) ส่วนการสร้างข้อสอบถามแต่ละสถานการณ์

ในส่วนย่อยที่ 1 ผู้วิจัยต้องวิเคราะห์สถิติเชิงพรรณาและสุ่มเลือกทรานแซคชั่นชุดทดสอบโดยใช้โปรแกรมตารางคำนวณไมโครซอฟต์เอ็กซ์เซล (Microsoft Excel) ทั้งหมดไม่ได้พัฒนาเครื่องมือสำหรับส่วนนี้ ขั้นตอนวิธีในการวิเคราะห์และเลือกทรานแซคชั่นชุดทดสอบจากโครงการพัฒนาซอฟต์แวร์ชื่อเคมายมันนี่ (KMyMoney) ที่อธิบายไว้ในหัวข้อ 3.6.2 และภาคผนวก ก

ในส่วนย่อยที่ 2 เป็นส่วนที่ผู้วิจัยพัฒนาเครื่องมือขึ้นมาเองและเรียกว่าเครื่องมือนี้ว่าเครื่องมือสร้างข้อสอบถามสำหรับ 3 สถานการณ์ โดยพัฒนาขึ้นมาในลักษณะของสคริปท์ด้วยภาษาพีเอชพี (PHP) ผู้วิจัยทำการทดสอบความถูกต้องของเครื่องมือนี้โดยการตรวจสอบแบบเดินผ่าน (Walkthrough) กับผลลัพธ์ทั้งหมดของเครื่องมือนี้ นั่นก็คือข้อสอบถามสำหรับ 3 สถานการณ์รายละเอียดแสดงในภาคผนวก ข

ขั้นตอนวิธีในส่วนย่อยที่ 2 การสร้างข้อสอบถามที่อธิบายไว้อย่างละเอียดในหัวข้อ 3.6.2

ข้อมูลเข้า คือ ทรานแซคชั่นของการเปลี่ยนแปลงแก้ไข (เฉพาะตารางชื่อ Revisions และตารางชื่อ Transactions) ตัวอย่างของข้อมูลเข้าแสดงในหัวข้อ 3.6.1

ข้อมูลออก คือ ข้อสอบถามสำหรับสถานการณ์นำทาง ข้อสอบถามสำหรับสถานการณ์ป้องกันการเกิดข้อผิดพลาด และข้อสอบถามสำหรับสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้ว (ตารางชื่อ Queries) ตัวอย่างของข้อมูลออกแสดงในหัวข้อ 3.6.2

- **ส่วนของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์**

ส่วนที่ 3 คือส่วนการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบ นั่นคือ การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 และการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 เครื่องมือที่ผู้วิจัยพัฒนาขึ้นมาสำหรับส่วนนี้เรียกว่าเครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 และเครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 โดยพัฒนาเครื่องมือทั้ง 2 นี้ในลักษณะของสคริปท์ด้วยภาษาพีเอชพี (PHP)

ขั้นตอนวิธีการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 และการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 ใช้ขั้นตอนวิธีที่ชื่อว่า วิธีอพริโอริ (Apriori algorithm) อธิบายในหัวข้อ 2.8.2 และกำหนดค่าองค์ประกอบต่างๆตามที่ระบุไว้ในหัวข้อ 3.6.3 เช่นเดียวกับงานวิจัยของ Zimmermann และคณะในปี ค.ศ. 2004 และ 2005 (Zimmermann et al., 2004; Zimmermann et al., 2005) และงานวิจัยของ Michail ในปี ค.ศ. 2000 (Michail, 2000) ผู้วิจัยทำการทดสอบความถูกต้องของเครื่องมือนี้โดยการสุ่มตรวจผลลัพธ์หรือกฎความสัมพันธ์ที่ได้มาจากเครื่องมือทั้ง 2 เครื่องมือรายละเอียดแสดงในภาคผนวก ข

ข้อมูลเข้าของเครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 และเครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 คือ ทรานแซคชั่นทั้งหมดในฐานข้อมูล (เฉพาะเฉพาะตารางชื่อ Revisions และตารางชื่อ Transactions ทั้งหมด) ตัวอย่างของข้อมูลเข้าแสดงในหัวข้อ 3.6.2

ข้อมูลออกของเครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 คือ กฎความสัมพันธ์ด้วยตัวแบบที่ 1 รวมถึงค่าสนับสนุนค่าความเชื่อมั่นของกฎความสัมพันธ์ (ตารางชื่อ Rules) ข้อมูลออกของเครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 คือ กฎความสัมพันธ์ด้วยตัวแบบที่ 2 รวมถึง

ค่าสนับสนุนค่าความเชื่อมั่นของกฎความสัมพันธ์ (ตารางชื่อ Rules) ตัวอย่างของข้อมูลออกแสดงในหัวข้อ 3.6.3

- **ส่วนของการสร้างเซตของคำแนะนำสำหรับเหตุการณ์ในข้อสอบถาม**

ส่วนที่ 4 ของการทดสอบคือส่วนของการสร้างเซตของคำแนะนำสำหรับเหตุการณ์ในข้อสอบถามชุดทดสอบ ในส่วนนี้ผู้วิจัยพัฒนาเครื่องมือขึ้นมาเองและให้ชื่อว่าเครื่องมือสร้างคำแนะนำสำหรับเหตุการณ์ ผู้วิจัยพัฒนาขึ้นมาในลักษณะของสคริปท์ด้วยภาษาพีเอชพี (PHP)

ขั้นตอนวิธีการสร้างเซตของคำแนะนำสำหรับเหตุการณ์ในข้อสอบถามอธิบายไว้ในหัวข้อ 3.6.4 ซึ่งเป็นขั้นตอนวิธีเดียวกันกับที่ใช้ในงานวิจัยของ Zimmermann และคณะในปี 2005 (Zimmermann et al., 2005) และในงานวิจัยของ Methanias และคณะในปี 2009 (Methanias et al., 2009) ผู้วิจัยทำการทดสอบความถูกต้องของเครื่องมือนี้โดยการสุ่มตรวจผลลัพธ์หรือเซตของคำแนะนำสำหรับเหตุการณ์ที่ได้มาจากการใช้เครื่องมือนี้ รายละเอียดแสดงในภาคผนวก ข

ข้อมูลเข้า คือ เซตเหตุการณ์ในข้อสอบถาม (ตารางชื่อ Queries) และกฎความสัมพันธ์ด้วยตัวแบบที่ 1 และกฎความสัมพันธ์ด้วยตัวแบบที่ 2 (ตารางชื่อ Rules) ตัวอย่างของข้อมูลเข้าแสดงในหัวข้อ 3.6.2 และหัวข้อ 3.6.3 ตามลำดับ

ข้อมูลออก คือ เซตของคำแนะนำของการทดสอบทั้ง 6 การทดสอบ (ตารางชื่อ Suggestions) ตัวอย่างของข้อมูลออกแสดงในหัวข้อ 3.6.4

- **ส่วนของการประเมินผลการทดสอบ**

ส่วนสุดท้ายของการทดสอบคือส่วนของการประเมินผลการทดสอบเป็นส่วนที่นำเซตของคำแนะนำสำหรับเหตุการณ์ที่ได้มาจากการทดสอบทั้ง 6 การทดสอบมาคำนวณหาค่าประสิทธิภาพของการทำเหมืองข้อมูล ในส่วนนี้ผู้วิจัยพัฒนาเครื่องมือขึ้นมาเองและให้ชื่อว่าเครื่องมือประเมินผลการทดสอบ โดยการทดสอบที่ทดสอบในสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาดนั้นจะคำนวณค่าเอฟเมสเซอร์ ส่วนการทดสอบที่ทดสอบในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วจะคำนวณค่าสะท้อนกลับ เช่นเดียวกับงานวิจัยของ Zimmermann และคณะในปี 2005 (Zimmermann et al., 2005) และในงานวิจัยของ Methanias และคณะในปี 2009 (Methanias et al., 2009) ผู้วิจัยพัฒนาขึ้นมาในลักษณะของสคริปท์ด้วยภาษาพีเอชพี (PHP) ตามขั้นตอนวิธีในการประเมินผลการทดสอบแต่ละ

การทดสอบอธิบายอย่างละเอียดในหัวข้อ 3.6.5 ผู้วิจัยทำการทดสอบความถูกต้องของเครื่องมือนี้โดยการสุ่มตรวจผลลัพธ์หรือค่าประสิทธิภาพของแต่ละข้อสอบถามในแต่ละการทดสอบที่ได้มาจากการใช้เครื่องมือนี้ รายละเอียดแสดงในภาคผนวก ข

ข้อมูลเข้า คือ เซตของคำแนะนำสำหรับเหตุการณ์ของการทดสอบทั้ง 6 การทดสอบ (ตารางชื่อ Suggestions) และข้อสอบถามของแต่ละสถานการณ์

ข้อมูลออก คือ ค่าประสิทธิภาพของการทดสอบทั้ง 6 การทดสอบที่อยู่ในรูปแบบ (format) ของแฟ้มข้อมูลตัวอักษร (Text file)

ผู้วิจัยจะนำข้อมูลออกที่ได้จากเครื่องมือประเมินผลการทดสอบไปเข้าสู่ขั้นตอนการทดสอบสมมติฐานซึ่งเป็นขั้นตอนสุดท้ายของการวิจัยนี้ ผู้วิจัยต้องใช้การวิเคราะห์และเลือกสถิติทดสอบที่เหมาะสมตามข้อกำหนดที่อธิบายในหัวข้อ 3.6.6 และนำไปวิเคราะห์ด้วยโปรแกรมสถิติเอสพีเอสเอสต่อไป

3.8 ความถูกต้อง (Validity) และค่าความน่าเชื่อถือ (Reliability) ของข้อมูลที่เก็บ

การตอบวัตถุประสงค์ของข้อมูลงานวิจัยให้เชื่อถือได้ (Reliability) และถูกต้อง (Validity) จำเป็นต้องควบคุมปัจจัยที่เกี่ยวข้องอันได้แก่ การเลือกโครงการพัฒนาซอฟต์แวร์ การสร้างข้อสอบถามและการทดสอบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์

เนื่องจากงานวิจัยนี้มีวัตถุประสงค์ในการทดลองเพื่อศึกษาผลกระทบจากตัวแปรต้น คือ การทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟ์ด้วยตัวแบบต่างๆ กัน ซึ่งตัวแปรต้นนี้เป็นปัจจัยที่ต้องเปลี่ยนค่าไปตามแบบแผนการทดลองเพื่อดูความแตกต่างอันเกิดขึ้นจากการทดลอง นอกจากนั้นยังต้องสามารถควบคุมปัจจัยในด้านต่างๆ ให้มีความเหมือนกันหรือมีความคงที่ภายใต้สภาวะเดียวกัน เพื่อผลการทดลองที่สะท้อนเป็นค่าของตัวแปรต้นของทั้งกลุ่มควบคุมและกลุ่มทดสอบ นั่นคือ การค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 และการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 เท่านั้น โดยในการทดลองมีปัจจัยที่ต้องควบคุมดังนี้

1) การเลือกโครงการพัฒนาซอฟต์แวร์ที่นำมาใช้ในการทดสอบ ผู้วิจัยกำหนดให้ข้อมูลซอฟต์แวร์อาร์ไคฟว์และข้อสอบถามที่จะทดสอบนั้นเป็นข้อมูลที่มาจากโครงการพัฒนาซอฟต์แวร์โครงการเดียวกันเพื่อให้ค่าประสิทธิภาพที่วัดออกมานั้นเป็นประสิทธิภาพที่เกิดมาจากตัวแบบของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ที่แตกต่างกันอย่างแท้จริง
2) โครงการพัฒนาซอฟต์แวร์ที่นำมาใช้ในงานวิจัยนี้เป็นโครงการพัฒนาซอฟต์แวร์ชื่อเคมายมันนี่ (KMyMoney) ซึ่งเป็นซอฟต์แวร์ทางการบัญชี ซึ่งเริ่มต้นการเผยแพร่โครงการตั้งแต่ปี ค.ศ. 2000 มีทรานแซคชั่นของการเปลี่ยนแปลงแก้ไขกว่า 28261 ทรานแซคชั่น ทำให้ข้อมูลซอฟต์แวร์อาร์ไคฟว์ของโครงการนี้สามารถเป็นตัวแทนของข้อมูลซอฟต์แวร์อาร์ไคฟว์ของโครงการพัฒนาซอฟต์แวร์ที่มีความหลากหลายได้
3) การกำหนดเซตเหตุการณ์และเซตผลลัพธ์ที่คาดไว้ของข้อสอบถามทั้งหมดถูกกำหนดมาจากข้อมูลซอฟต์แวร์อาร์ไคฟว์ที่นำมาทดสอบเอง จึงสามารถแน่ใจได้ว่าผลลัพธ์ที่จะได้ออกมานั้นมาจากเหตุการณ์ที่เคยเกิดขึ้นมาแล้วจริงๆในอดีต ซึ่งทำให้กระบวนการพิจารณาผลลัพธ์ที่ระบบแสดงออกมาในขั้นตอนการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ด้วยตัวแบบทั้ง 2 ตัวแบบในงานวิจัยนี้จะสามารถเชื่อถือความถูกต้องของผลลัพธ์ซึ่งเป็นเซตของคำแนะนำสำหรับเหตุการณ์นั้นๆได้
4) การทดสอบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ในงานวิจัยนี้จะใช้ค่าเอฟเมสเซอร์ (F-measure) ที่เป็นการคำนวณร่วมกันระหว่างค่าความถูกต้อง (Precision) และค่าเรียกคืน (Recall) เป็นมาตรวัดที่นิยมใช้ในการทดลองในงานวิจัยด้านการค้นคืนสารสนเทศ (Baeza-Yates and Riberio-Neto, 1999) โดยจะเป็นการวัดว่าระบบสามารถให้ผลลัพธ์ของการค้นคืนออกมาได้ถูกต้องหรือไม่
5) เครื่องมือที่ใช้ในการทดสอบประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ทั้ง 2 ตัวแบบเป็นเครื่องมือเดียวกันทั้งหมด ต่างกันเพียงเครื่องในการค้นหากฎความสัมพันธ์ที่ใช้ตัวแบบในการระบุความน่าสนใจของกฎความสัมพันธ์ (ตัวแปรต้น) เท่านั้น
6) เครื่องมือที่ใช้ในการจัดเตรียมข้อมูลเพื่อการทำเหมืองข้อมูลกับข้อมูลซอฟต์แวร์อาร์ไคฟว์คือเครื่องมือที่เป็นส่วนหนึ่งโปรแกรมประยุกต์อีโรส (eROSE) (Zimmermann et

al., 2005) ที่ได้รับการยอมรับและถูกนำไปใช้งานวิจัยที่เกี่ยวข้องกับการวิเคราะห์ข้อมูลซอฟต์แวร์อาร์ไคฟว์เช่น งานวิจัยของ Zimmermann และคณะในปี 2005 (Zimmermann et al., 2005) และในงานวิจัยของ Methanias และคณะในปี 2009 (Methanias et al., 2009)

7) เครื่องมือที่ผู้วิจัยพัฒนาขึ้นมาเองได้แก่ เครื่องมือสร้างข้อสอบถามสำหรับ 3 สถานการณ์ เครื่องมือค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 และตัวแบบที่ 2 เครื่องมือสร้างคำแนะนำสำหรับเหตุการณ์ และเครื่องมือประเมินผลการทดสอบ ผู้วิจัยได้ทำการทดสอบความถูกต้องของเครื่องมือทั้งด้วยวิธีการสุ่มตรวจความถูกต้องของผลลัพธ์ที่เป็นตัวแทนของผลลัพธ์ทั้งหมด

3.9 **กรอบการวิเคราะห์ข้อมูล** (Data Analysis Framework)

สำหรับกรณีของการทดสอบในสถานการณ์การนำทางและสถานการณ์การป้องกันการเกิดข้อผิดพลาดนั้นใช้ค่าเอฟเมสเซอร์เป็นค่าที่แสดงถึงประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ เมื่อการทดสอบประสิทธิภาพเสร็จสิ้นแล้ว ทำให้ได้ค่าเอฟเมสเซอร์ออกมาเท่าจำนวนของข้อสอบถามชุดทดสอบที่สร้างขึ้นในขั้นตอนการสร้างข้อสอบถาม จากนั้นในขั้นตอนแรกจะตรวจสอบการแจกแจงของค่าประสิทธิภาพที่ได้มาว่ามีการแจกแจงปกติหรือไม่ ด้วยการใช้สถิติทดสอบ Kolmogorov-Smirnov เพื่อเลือกทางเลือกในการทดสอบสมมติฐานได้ว่าจะให้การทดสอบสมมติฐานแบบใช้พารามิเตอร์ (Parametric Test) หรือแบบไม่อิงกับพารามิเตอร์ (Non Parametric Test) ถ้าผลการทดสอบพบว่าประชากรมีการแจกแจงแบบปกติ จึงใช้การวิเคราะห์โดยสถิติทดสอบที (t-test) เพื่อทดสอบสมมติฐานของผลต่างระหว่างค่าเฉลี่ยของค่าเอฟเมสเซอร์ของหน่วยทดลอง 2 กลุ่ม ถ้าค่า Sig. (Significance) ที่คำนวณได้น้อยกว่า 0.05 และค่าสถิติทีมากกว่า 0 จึงจะสามารถปฏิเสธ H_0 ได้ แต่ถ้าผลการแจกแจงประชากรพบว่ามีการแจกแจงไม่ปกติ ต้องใช้วิธีการทดสอบสมมติฐานแบบไม่อิงกับพารามิเตอร์ (Non Parametric Test) ต่อไป โดยในที่นี้คือการวิเคราะห์โดยสถิติทดสอบเครื่องหมายลำดับที่ของวิลคอกซันสำหรับการทดสอบแบบจับคู่ (The Wilcoxon Signed Rank Sum Test for the Matched Paired Difference) เพื่อทดสอบสมมติฐานของผลต่างระหว่างค่าเฉลี่ยของค่าเอฟเมสเซอร์ของหน่วยทดลอง 2 กลุ่ม ถ้าค่า Sig. (Significance) ที่คำนวณได้น้อย

กว่า 0.05 และค่าสถิติซีมากกว่า 0 ในกรณีที่ผลการวิเคราะห์ตั้งอยู่บนพื้นฐานทางบวก (Based on positive ranks) จึงจะสามารถปฏิเสธ H_0 ได้

สำหรับการทดสอบในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้นใช้ค่าผลสะท้อนกลับ เป็นค่าที่แสดงถึงประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ เมื่อการทดสอบประสิทธิภาพเสร็จสิ้นแล้ว ทำให้ได้ค่าผลสะท้อนกลับออกมาหนึ่งค่าต่อหนึ่งการทดสอบ ค่าผลสะท้อนกลับที่ได้มานั้นแสดงให้เห็นถึงร้อยละของการเกิดการแจ้งเตือนที่ผิด (False Alarm) หรือการให้คำแนะนำที่เป็นผลบวกลวง (False Positive) นั่นเอง ดังนั้นการวัดประสิทธิภาพของการทำเหมืองข้อมูลด้วยเทคนิคการค้นหากฎความสัมพันธ์กับข้อมูลซอฟต์แวร์อาร์ไคฟว์ในสถานการณ์การเปลี่ยนแปลงแก้ไขที่สมบูรณ์แล้วนั้น ค่าผลสะท้อนกลับที่น้อยกว่าจะมีความหมายว่ามีประสิทธิภาพมากกว่า นั่นคือถ้าค่าผลสะท้อนกลับของการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 1 มากกว่าการค้นหากฎความสัมพันธ์ด้วยตัวแบบที่ 2 แล้วจึงสามารถปฏิเสธ H_0 ได้