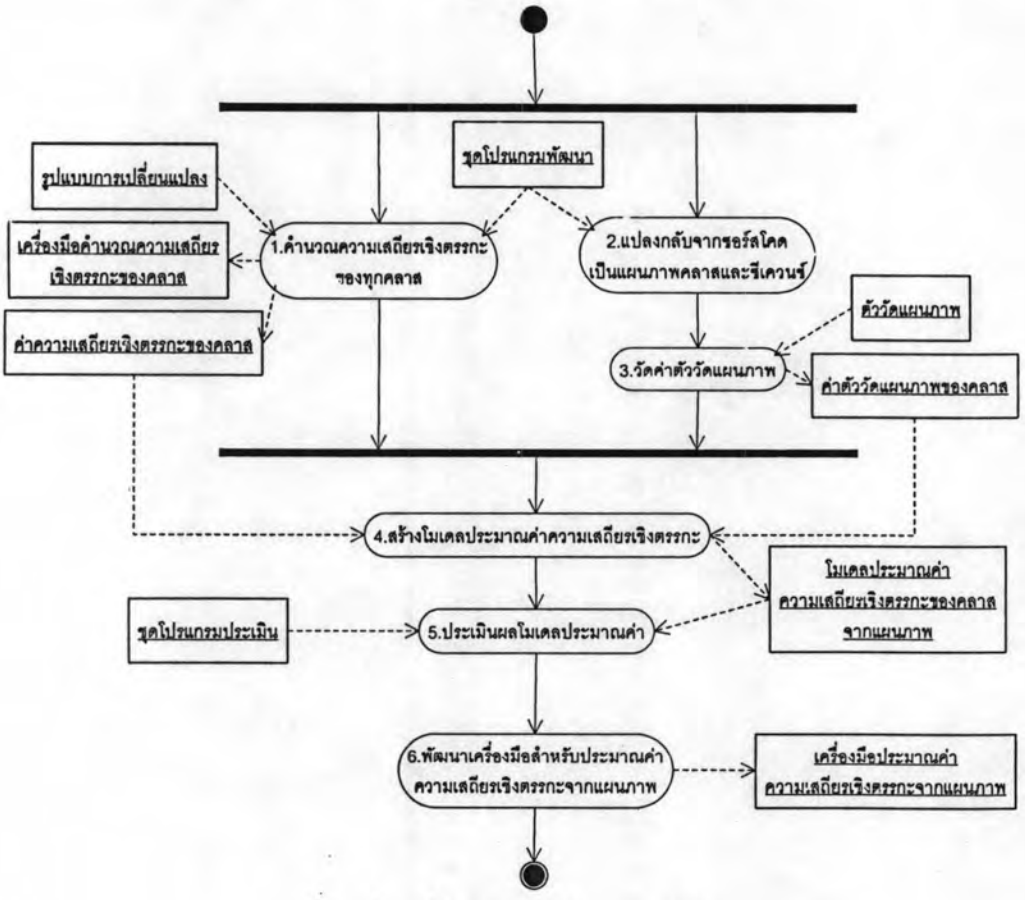
บทที่ 3

การสร้างโมเดลสำหรับประมาณค่าความเสถียรเชิงตรรกะ

จากการศึกษางานวิจัยต่างๆ พบว่า การคำนวณความเสถียรเชิงตรรกะในงานวิจัยส่วนใหญ่จะทำการคำนวณจากซอร์สโคด ซึ่งทำให้การแก้ไขจะต้องใช้เวลาและกำลังงานมากกว่าการแก้ไขการออกแบบ ดังนั้นผู้วิจัยจึงมีแนวคิดที่จะทำการสร้างโมเดลสำหรับประมาณค่าความเสถียรเชิงตรรกะของคลาสจากแผนภาพคลาสและแผนภาพซีเควนซ์ ซึ่งจะช่วยให้สามารถทำการประมาณค่าความเสถียรเชิงตรรกะได้จากแบบจำลองที่ใช้ในการออกแบบซอฟต์แวร์ และจะช่วยลดเวลาที่ใช้ในการแก้ไขเมื่อเทียบกับการแก้ไขที่ซอร์สโคดอีกด้วย

ในบทนี้จะกล่าวถึงการสร้างโมเดลสำหรับประมาณค่าความเสถียรเชิงตรรกะของคลาสจากแผนภาพคลาสและแผนภาพซีเควนซ์ แผนภาพกิจกรรมในรูปที่ 3.1 แสดงขั้นตอนการดำเนินงานเพื่อสร้างโมเดลสำหรับประมาณค่าความเสถียรเชิงตรรกะ



รูปที่ 3.1 ขั้นตอนการดำเนินงานวิจัย

ขั้นตอนในการดำเนินงานเพื่อสร้างโมเดลประมาณค่าความเสถียรเชิงตรรกะจากแผนภาพมีด้วยกัน 6 ขั้นตอน โดยเริ่มจากการคำนวณความเสถียรเชิงตรรกะของคลาสจากซอร์สโคดของชุดโปรแกรมทดสอบซึ่งมีรายละเอียดในหัวข้อ 3.1 ในขณะเดียวกันก็นำซอร์สโคดของชุดโปรแกรมทดสอบแปลงกลับเป็นแผนภาพคลาสและแผนภาพซีเควนซ์ดังรายละเอียดในหัวข้อ 3.2 แล้วจึงทำการวัดค่าตัววัดแผนภาพจากแผนภาพที่ได้จากการแปลงกลับ ในหัวข้อที่ 3.3 เมื่อได้ค่าความเสถียรเชิงตรรกะของคลาสจากซอร์สโคด และค่าตัววัดแผนภาพแล้วจึงทำการสร้างโมเดลเพื่อประมาณค่าความเสถียรเชิงตรรกะจากแผนภาพ โดยอาศัยข้อมูลทั้ง 2 โดยรายละเอียดอยู่ในบทที่ 4 จากนั้นทำการประเมินผลโมเดลที่ได้เพื่อหาค่าความคลาดเคลื่อนของโมเดลในการประมาณค่า ซึ่งมีรายละเอียดในบทที่ 4 ขั้นตอนสุดท้ายทำการพัฒนาเครื่องมือเพื่อประมาณค่าความเสถียรเชิงตรรกะโดยมีรายละเอียดในบทที่ 5

3.1 คำนวณความเสถียรเชิงตรรกะของคลาสของชุดโปรแกรมพัฒนา

ความเสถียรเชิงตรรกะของแต่ละคลาส คือค่าความต้านทานการได้รับผลกระทบจากการเปลี่ยนแปลงที่คลาสใดๆ ของโปรแกรม โดยค่าความเสถียรเชิงตรรกะของแต่ละคลาสสามารถคำนวณได้จากการนำรูปแบบการเปลี่ยนแปลงที่กำหนดไว้มาจำลองการเปลี่ยนแปลงกับโปรแกรมโดยดำเนินการจำลองทีละคลาสและทีละการเปลี่ยนแปลง จากนั้นทำการเก็บข้อมูลจำนวนครั้งการเปลี่ยนแปลงที่เป็นไปได้ทั้งหมด พร้อมทั้งจำนวนครั้งที่คลาสหนึ่งๆ จะได้รับผลกระทบตามเงื่อนไขของรูปแบบการเปลี่ยนแปลง

รูปแบบการเปลี่ยนแปลงที่ใช้ในวิทยานิพนธ์นี้ ได้ทำการรวบรวมและดัดแปลงมาจากงานวิจัยอื่นๆ [2][7][11][12] โดยรวบรวมมาเฉพาะรูปแบบการเปลี่ยนแปลงที่ส่งผลกระทบเชิงตรรกะซึ่งจะทำให้โปรแกรมเกิดการผิดพลาด ไม่สามารถทำงานได้หรือทำงานผิดไปจากเดิม โดยรูปแบบการเปลี่ยนแปลงที่รวบรวมมาสามารถแบ่งออกเป็น 2 ระดับด้วยกัน คือการเปลี่ยนแปลงในระดับของระบบ (System-level Change) และการเปลี่ยนแปลงในระดับของคลาส (Class-level Change) การเปลี่ยนแปลงในระดับของระบบได้แก่ การเพิ่ม ลบ คลาส และการเพิ่ม ลบ ความสัมพันธ์ต่างๆ ส่วนการเปลี่ยนแปลงในระดับคลาสแบ่งออกเป็น การเปลี่ยนแปลงที่เกิดแก่แอททริบิวต์และ เมทธอดของคลาส เช่น การเพิ่ม ลบ แอททริบิวต์ หรือ เมทธอด การแก้ไขประเภทแอททริบิวต์ และ การแก้ไขขอบเขต เป็นต้น โดยรายละเอียดของรูปแบบการเปลี่ยนแปลงทั้งหมด รวมทั้งคลาสที่จะได้รับผลกระทบจากแต่ละการเปลี่ยนแปลงแสดงอยู่ในตารางที่ 3.1

การเปลี่ยนแปลงในระดับของระบบเป็นการเปลี่ยนแปลงที่เกี่ยวข้องกับความสัมพันธ์ของคลาส และภาพรวมของระบบ ซึ่งเป็นการเปลี่ยนแปลงที่สามารถกระทำได้ทั้งที่ซอร์สโคด และแบบจำลองในการออกแบบซอฟต์แวร์ ทั้งนี้สาเหตุในการเลือกรูปแบบการเปลี่ยนแปลงในระดับของระบบมาใช้ในวิทยานิพนธ์นี้ เนื่องจากวิทยานิพนธ์นี้มีวัตถุประสงค์ในการสร้างโมเดลเพื่อประมาณค่าความเสถียรเชิงตรรกะจากแบบจำลองการออกแบบซอฟต์แวร์ ซึ่งแสดงถึงภาพรวมและรายละเอียดของซอฟต์แวร์ ทว่ารูปแบบการเปลี่ยนแปลงในระดับของคลาสไม่เพียงพอที่จะสะท้อนภาพรวมของระบบ จึงต้องนำรูปแบบการเปลี่ยนแปลงในระดับของระบบมาใช้ในการคำนวณความเสถียรเชิงตรรกะด้วย

ทั้งนี้โอกาสในการเกิดการเปลี่ยนแปลงแต่ละรูปแบบมีค่าไม่เท่ากัน อย่างไรก็ตาม เนื่องจากวิทยานิพนธ์นี้มีวัตถุประสงค์ในการสร้างโมเดลประมาณค่าความเสถียรเชิงตรรกะ เพื่อนำไปใช้ในระหว่างการออกแบบซอฟต์แวร์ซึ่งไม่สามารถทราบได้ว่าจะเกิดการเปลี่ยนแปลงใดได้บ้าง จึงให้โอกาสในการเกิดการเปลี่ยนแปลงแต่ละรูปแบบเท่ากันหมด

ตารางที่ 3.1 รูปแบบการเปลี่ยนแปลง

หมายเลข	รูปแบบการเปลี่ยนแปลง	คลาสที่อาจได้รับผลกระทบ
การเปลี่ยนแปลงระดับระบบ		
1	เพิ่มความสัมพันธ์คลาสแม่ของ C	C,S
2	ลบความสัมพันธ์คลาสแม่ของ C	C,S,R
3	เพิ่มความสัมพันธ์คลาสลูกของ C	
4	ลบความสัมพันธ์คลาสลูกของ C	
5	เพิ่มความสัมพันธ์แอกกรีเกตคลาสของ C	C,S
6	ลบความสัมพันธ์แอกกรีเกตคลาสของ C	C,S,R
7	เพิ่มความสัมพันธ์คลาส C เป็นแอกกรีเกตคลาส	
8	ลบความสัมพันธ์คลาส C ที่เป็นแอกกรีเกตชันคลาส	
9	เพิ่มความสัมพันธ์แอสโซซิเอชันคลาสของ C	C,S
10	ลบความสัมพันธ์แอสโซซิเอชันคลาสของ C	C,S,R

ตารางที่ 3.1 รูปแบบการเปลี่ยนแปลง (ต่อ)

หมายเลข	รูปแบบการเปลี่ยนแปลง	คลาสที่อาจได้รับผลกระทบ
12	ลบความสัมพันธ์คลาส C เป็นแอสโซซิเอชันคลาส	
13	เพิ่มคลาส C	
14	ลบคลาส C	C,S,R
การเปลี่ยนแปลงระดับคลาส		
	แอททริบิวต์ A ของ C	
15	เพิ่มแอททริบิวต์	S (เมื่อขอบเขตเป็นพับบลิคเท่านั้น)
16	ลบแอททริบิวต์	
	ขอบเขตเป็นพับบลิค	C,X,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
17	เปลี่ยนค่าของแอททริบิวต์	
	ขอบเขตเป็นพับบลิค	C,X,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
18	เปลี่ยนขอบเขตการเข้าถึง	
	พับบลิคเป็นไพรเวต	S,X
	พับบลิคเป็นโพรเทคต์	X
	โพรเทคต์เป็นพับบลิค	
	โพรเทคต์เป็นไพรเวต	S
	ไพรเวตเป็นโพรเทคต์	

ตารางที่ 3.1 รูปแบบการเปลี่ยนแปลง (ต่อ)

หมายเลข	รูปแบบการเปลี่ยนแปลง	คลาสที่อาจได้รับผลกระทบ
19	เปลี่ยนประเภทแอททริบิวต์	
	ขอบเขตเป็นพับบลิค	C,X,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
20	เปลี่ยนชื่อแอททริบิวต์	
	ขอบเขตเป็นพับบลิค	C,X,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
	เมทธอด M ของ C	
21	เพิ่มเมทธอด	S (เมื่อขอบเขตเป็นพับบลิคเท่านั้น)
22	ลบเมทธอด	
	ขอบเขตเป็นพับบลิค	C,Y,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
23	เปลี่ยนชื่อเมทธอด	
	ขอบเขตเป็นพับบลิค	C,Y,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
24	เปลี่ยนซิกเนเจอร์	
	ขอบเขตเป็นพับบลิค	C,Y,S
	ขอบเขตเป็นโพรเทคต์	C,S

ตารางที่ 3.1 รูปแบบการเปลี่ยนแปลง (ต่อ)

หมายเลข	รูปแบบการเปลี่ยนแปลง	คลาสที่อาจได้รับผลกระทบ
	ขอบเขตเป็นไพรเวต	C
25	เปลี่ยนชนิดข้อมูลคืนกลับ	
	ขอบเขตเป็นพับบลิค	C,Y,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
26	เปลี่ยนรายละเอียดการทำงาน	
	ขอบเขตเป็นพับบลิค	C,Y,S
	ขอบเขตเป็นโพรเทคต์	C,S
	ขอบเขตเป็นไพรเวต	C
27	เปลี่ยนขอบเขตการเข้าถึง	
	พับบลิคเป็นไพรเวต	S,Y
	พับบลิคเป็นโพรเทคต์	Y
	โพรเทคต์เป็นพับบลิค	
	โพรเทคต์เป็นไพรเวต	S
	ไพรเวตเป็นพับบลิค	
	ไพรเวตเป็นโพรเทคต์	
C คือคลาสที่สนใจ		
S คือคลาสลูกของ C		
R คือคลาสที่มีความสัมพันธ์กับ C ในฐานะผู้กระทำ เช่น เป็นแอสโซซิเอชันคลาสของ C		
X คือคลาสที่เรียกใช้แอททริบิวต์ A ของ C		
Y คือคลาสที่เรียกใช้เมทธอด M ของ C		

ตารางที่ 3.1 แสดงรูปแบบการเปลี่ยนแปลงที่เป็นไปได้ และอาจส่งผลกระทบเชิงตรรกะแก่ซอฟต์แวร์ พร้อมทั้งกลุ่มของคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงนั้นๆ เพื่อใช้ในการวัดค่าความเสถียรเชิงตรรกะจากซอร์สโคดของซอฟต์แวร์ โดยแต่ละรูปแบบของการเปลี่ยนแปลงจะส่งผลกระทบต่อกลุ่มของคลาสต่างกันไปขึ้นอยู่กับรูปแบบการเปลี่ยนแปลงและขอบเขตของแอททริบิวต์และเมทธอด ตัวอย่างเช่น การเปลี่ยนแปลงที่ 16 การลบแอททริบิวต์ หากแอททริบิวต์มีขอบเขตเป็นพับบลิค คลาสที่จะได้รับผลกระทบจากการเปลี่ยนแปลงคือ คลาสที่มีแอททริบิวต์นั้นอยู่ คลาสลูก และคลาสที่มีการอ้างถึงแอททริบิวต์นั้นๆ เพราะเมื่อขอบเขตของแอททริบิวต์เป็นพับบลิคทุกๆ คลาสในซอฟต์แวร์จะสามารถเข้าถึงได้โดยตรง และเมื่อขอบเขตของแอททริบิวต์เป็นโพรเทคต์คลาสที่ได้รับผลกระทบคือ คลาสที่มีแอททริบิวต์อยู่ และคลาสลูก ส่วนคลาสอื่นๆ ไม่ได้รับผลกระทบเพราะไม่สามารถเข้าถึงแอททริบิวต์ได้ เป็นต้น

การเปลี่ยนแปลงในระดับของระบบที่แสดงในตารางที่ 3.1 ที่เกี่ยวข้องกับความสัมพันธ์ระหว่างคลาส จะแยกพิจารณาเป็นความสัมพันธ์แบบทางเดียวทั้งสิ้น และในการจำลองการเปลี่ยนแปลงเพื่อวิเคราะห์และคำนวณความเสถียรเชิงตรรกะ จะทำการจำลองทีละคลาสและพิจารณาผลกระทบที่เกิดกับคลาสที่สนใจเป็นหลัก ทำให้การเปลี่ยนแปลงบางรูปแบบถือว่าไม่เกิดผลกระทบขึ้น เช่นการเพิ่มความสัมพันธ์ให้คลาสที่สนใจเป็นแอสโซซิเอชันคลาส หรือมีคลาสอื่นมาเรียกใช้ ในกรณีนี้ถือว่าไม่มีผลกระทบกับคลาสที่สนใจ ในขณะที่เพิ่มความสัมพันธ์แอสโซซิเอชันคลาสของคลาสที่สนใจ หรือเรียกใช้คลาสอื่น กรณีนี้จะเกิดผลกระทบกับคลาสที่สนใจและคลาสลูก เป็นต้น อย่างไรก็ตามในการคำนวณความเสถียรเชิงตรรกะจะต้องทำการจำลองการเปลี่ยนแปลงทุกรูปแบบให้แก่ทุกๆ คลาส ดังนั้นทุกคลาสจะได้รับการวิเคราะห์เหมือนกันหมดทั้งสิ้น

รูปแบบการเปลี่ยนแปลงและคลาสที่ได้รับผลกระทบเชิงตรรกะในตารางที่ 3.1 ใช้เป็นพื้นฐานในการวิเคราะห์เพื่อวัดค่าความเสถียรเชิงตรรกะของคลาสใดๆ โดยเมื่อทำการตรวจสอบการเปลี่ยนแปลงหนึ่งๆ ของคลาสใดๆ (C) แล้วทำการเปรียบเทียบกับตารางที่ 3.1 ว่าคลาสที่อาจได้รับผลกระทบจากการเปลี่ยนแปลงนั้นๆ มีคลาสใดบ้าง

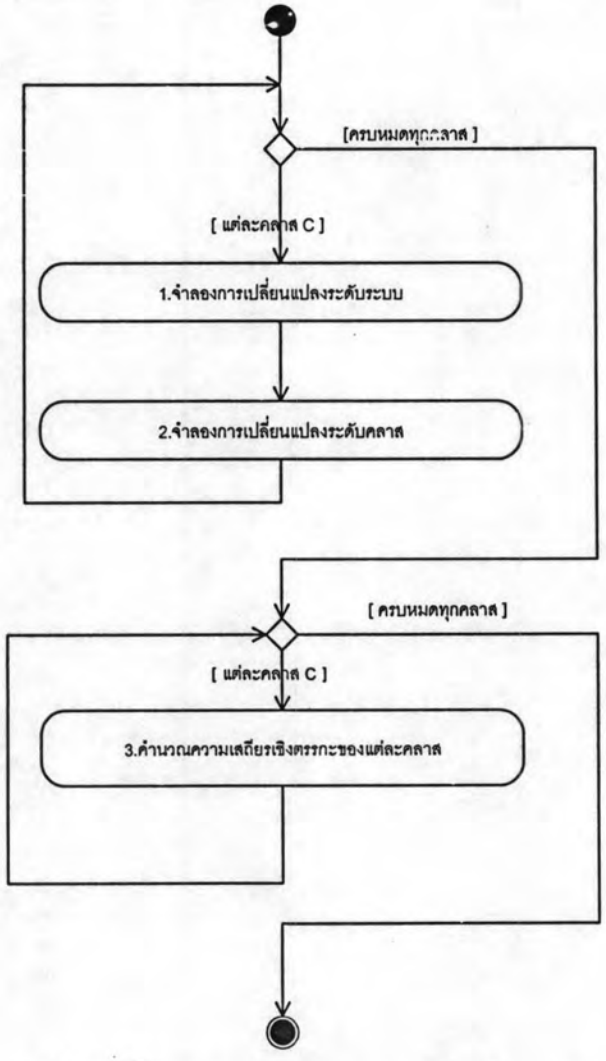
การเปลี่ยนแปลงในตารางที่ 3.1 เป็นการเปลี่ยนแปลงพื้นฐานทั่วไปและเป็นองค์ประกอบของการเปลี่ยนแปลงระดับสูงต่างๆ เช่นการทำรีแฟคเตอริง เป็นต้น ตัวอย่างของความสัมพันธ์ระหว่างการทำรีแฟคเตอริง [13] และการเปลี่ยนแปลงแสดงดังตารางที่ 3.1 เช่น

- การย้ายเมทธอด (Move Method): คือการลบเมทธอด และเพิ่มเมทธอดตามลำดับ

- การขยายคลาส (Extract Class) : คือการลบเมทธอด ลบแอททริบิวต์ เพิ่มคลาส และเพิ่มความสัมพันธ์ระหว่างคลาสตามลำดับ

โดยตารางแสดงความสัมพันธ์ระหว่างการทำรีแฟคเตอริงกับการเปลี่ยนแปลงพื้นฐาน แสดงในภาคผนวก ข

ขั้นตอนการคำนวณความเสถียรเชิงตรรกะของคลาสสามารถแสดงเป็นแผนภาพกิจกรรมได้ดังรูปที่ 3.2 โดยวิธีการในการคำนวณความเสถียรเชิงตรรกะของงานวิจัยนี้จะทำการปรับปรุงวิธีการหาค่าความเสถียรเชิงตรรกะจากงานวิจัยของ Mahmoud O. Elish และ David Rine [7] ซึ่งเป็นวิธีการหาค่าความเสถียรเชิงตรรกะที่สอดคล้องกับแนวทางของงานวิจัยนี้เนื่องจากเป็นวิธีการในการหาค่าความเสถียรเชิงตรรกะของโปรแกรมเชิงวัตถุจากซอร์สโคด และมีวิธีการคำนวณความเสถียรเชิงตรรกะใกล้เคียงกับที่ต้องการ



รูปที่ 3.2 ขั้นตอนการคำนวณความเสถียรเชิงตรรกะ

จากรูปที่ 3.2 การคำนวณความเสถียรเชิงตรรกะของโปรแกรมหนึ่งประกอบไปด้วย 4 ขั้นตอนด้วยกัน เริ่มจากทำการจำลองการเปลี่ยนแปลงระดับของระบบแก่คลาสในขั้นตอนที่ 1 แล้วจึงทำการจำลองการเปลี่ยนแปลงแก่ทุกๆ เมทธอดของคลาสในขั้นตอนที่ 2 และทำการจำลองการเปลี่ยนแปลงแก่ทุกๆ แอททริบิวต์ของคลาสในขั้นตอนที่ 3 และกลับไปทำขั้นตอนที่ 1 ใหม่จนกว่าจะครบทุกคลาส และในระหว่างขั้นตอนที่ 1 ถึง 3 จะมีการเก็บค่าจำนวนการเปลี่ยนแปลงที่เป็นไปได้ทั้งหมดของโปรแกรม และจำนวนครั้งที่แต่ละคลาสจะได้รับผลกระทบจากการเปลี่ยนแปลงนั้นๆ เพื่อนำไปคำนวณค่าความเสถียรเชิงตรรกะของแต่ละคลาสในขั้นตอนที่ 4 ต่อไป

การคำนวณความเสถียรเชิงตรรกะเริ่มด้วยการจำลองการเปลี่ยนแปลงตามตารางที่ 3.1 ให้กับทุกคลาสในโปรแกรมทีละคลาส โดยเริ่มจากการเปลี่ยนแปลงในระดับของระบบก่อน จากนั้นจึงจำลองการเปลี่ยนแปลงระดับของคลาสให้แก่ทุกเมทธอดและแอททริบิวต์ในคลาส และในการจำลองการเปลี่ยนแปลงแต่ละครั้งจะทำการตรวจสอบและบันทึกข้อมูลดังนี้

1. เพิ่มค่าจำนวนการเปลี่ยนแปลงทั้งหมดไปอีกหนึ่งเมื่อการเปลี่ยนแปลงนั้นสามารถทำการจำลองได้
2. ตรวจสอบคลาสที่อาจได้รับผลกระทบและเพิ่มค่าจำนวนครั้งที่ได้รับผลกระทบของคลาสนั้นๆ โดยจะนับเพียงแค่ 1 ครั้งต่อ 1 การเปลี่ยนแปลงเท่านั้น

เมื่อเสร็จสิ้นการจำลองการเปลี่ยนแปลงและเก็บข้อมูลทุกคลาสเสร็จสิ้น จึงทำการคำนวณหาค่าความเสถียรเชิงตรรกะของแต่ละคลาสตามสมการที่ 3 และตัวอย่างการคำนวณความเสถียรเชิงตรรกะแสดงได้ดังตารางที่ 3.2 และ 3.3

ตารางที่ 3.2 ตัวอย่างซอร์สโคด

```
public class C1{
  public int a1;
  private C2 a2;
  public int mC1(){
    C3 c = new C3();
    a1= c.mC3();
    return a1; }
}
```

```
public class C2{
  public int b1;
  Public void mC2(){
    C3 c = new C3();
    b1 = c.mC3() ;
  }
}
```

```
public class C3 extends C1{
  public int c1;
  public int mC3(){
    c1 = 10;
    return c1; }
}
```

ตารางที่ 3.3 ตัวอย่างการคำนวณความเสถียรเชิงตรรกะ

การเปลี่ยนแปลง	คลาส C1	คลาส C2	คลาส C3
ลบคลาส C1	x		x
ลบคลาส C2	x	x	
ลบคลาส C3	x	x	x
เพิ่มแอสโซซิเอชันของ C1	x		x
ลบแอททริบิวต์ a1	x		x
เปลี่ยนขอบเขต a1 เป็น private			x
เปลี่ยนขอบเขต a1 เป็น protected			
เปลี่ยนชื่อ method mC3()	x	x	x
เปลี่ยนขอบเขต mC3() เป็น private	x	x	
เปลี่ยนขอบเขต mC3() เป็น protected	x	x	
รวม	8	5	6
ความเสถียรเชิงตรรกะ	1-(8/10) =0.2	1-(5/10) =0.5	1-(6/10) =0.4

ตารางที่ 3.2 คือตัวอย่างซอร์สโคด ซึ่งประกอบไปด้วยคลาส 3 คลาส คือ C1, C2 และ C3 โดย C2 เป็นคลาสลูกของ C1 และเมทธอด mC3() ถูกเรียกใช้โดย C1 และ C2 และตารางที่ 3.3 คือส่วนหนึ่งของการเปลี่ยนแปลงที่เป็นไปได้ทั้งหมด และคลาสที่ได้รับผลกระทบจากการเปลี่ยนแปลงนั้นๆ ที่นำมาเพื่อแสดงให้เห็นวิธีการคำนวณความเสถียรเชิงตรรกะเท่านั้น โดยให้การเปลี่ยนแปลงที่เป็นไปได้ทั้งหมดเท่ากับ 10 ครั้ง และคลาส C1, C2 และ C3 ได้รับผลกระทบเป็นจำนวน 8, 5 และ 6 ครั้งตามลำดับ ดังนั้นสามารถคำนวณความเสถียรเชิงตรรกะของแต่ละคลาสตามสมการที่ 3 ได้ดังนี้

ความเสถียรเชิงตรรกะของ C1 = 1 – (8/10) = 0.2

ความเสถียรเชิงตรรกะของ C2 = 1 – (5/10) = 0.5

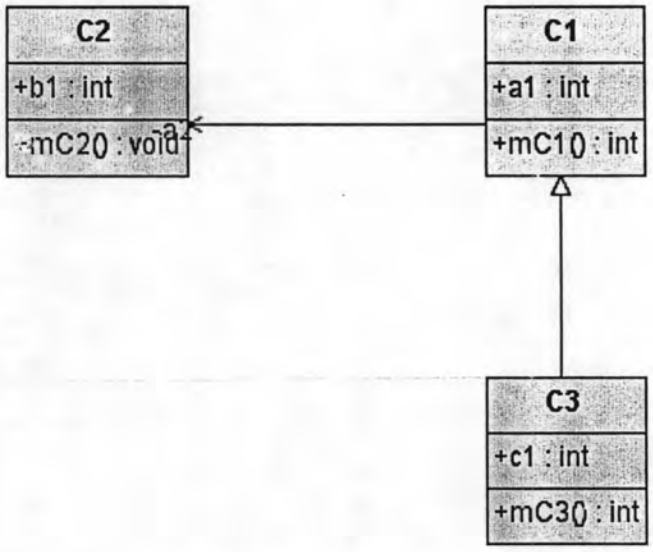
ความเสถียรเชิงตรรกะของ C3 = 1 – (6/10) = 0.4

ดังนั้นจะได้ว่าคลาส C1 C2 และ C3 มีความเสถียรเชิงตรรกะเป็น 0.2 0.5 และ 0.4 ตามลำดับ

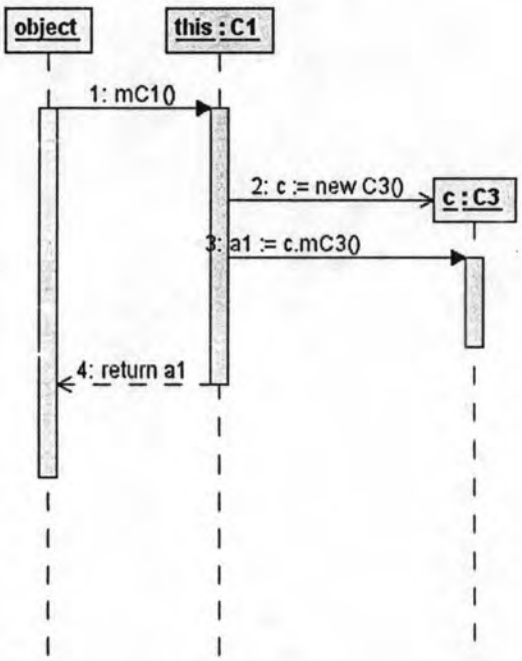
3.2 การแปลงซอร์สโคดเป็นแผนภาพคลาสและแผนภาพซีเควนซ์

การแปลงกลับแผนภาพในขั้นตอนนี้จะอาศัยโปรแกรม MagicDraw UML เวอร์ชัน 9.5 [14] ซึ่งมีคุณสมบัติในการแปลงกลับซอร์สโคดภาษาต่างๆ เป็นแผนภาพยูเอมแอลรวมไปถึงภาษาจาวาด้วย โดยในการแปลงกลับนี้โปรแกรมจะทำการค้นหาไฟล์ซอร์สโคดภาษาจาวาทั้งหมดที่อยู่ภายใต้โฟลเดอร์ที่ต้องการแล้วทำการแปลงกลับเป็นข้อมูลเข้าสู่โปรแกรม และผู้ใช้สามารถเลือกให้โปรแกรมทำการสร้างแผนภาพคลาสของโปรแกรมที่ถูกแปลงกลับได้โดยอัตโนมัติ จากนั้นผู้ใช้ต้องทำการเลือกและสั่งให้โปรแกรม MagicDraw ทำการสร้างแผนภาพซีเควนซ์จากเมทธอดของโปรแกรมที่ถูกแปลงกลับทีละเมทธอด โดยจะได้ 1 แผนภาพซีเควนซ์ต่อ1 เมทธอด เมื่อสร้างแผนภาพซีเควนซ์ของทุกเมทธอดจนครบแล้วสามารถทำการบันทึกข้อมูลแผนภาพให้อยู่ในรูปแบบของภาษาเอ็กซ์เอมแอลได้เพื่อนำไปใช้ต่อไป ตัวอย่างวิธีการแปลงกลับแสดงอยู่ในภาคผนวก ค

รูปที่ 3.3 แสดงตัวอย่างแผนภาพคลาสที่จากการแปลงกลับซอร์สโคดในตารางที่ 3.2 และรูปที่ 3.4 แสดงตัวอย่างแผนภาพซีเควนซ์ที่ได้จากการแปลงกลับเมทธอด mC1() ของคลาส C1 ในตารางที่ 3.2



รูปที่ 3.3 ตัวอย่างแผนภาพคลาสที่ได้จากการแปลงกลับ



รูปที่ 3.4 ตัวอย่างแผนภาพซีเควนซ์ที่ได้จากการแปลงกลับ

3.3 การวัดค่าตัววัดแผนภาพที่ได้จากการแปลงกลับ

หลังจากแปลงกลับแผนภาพคลาสและแผนภาพซีเควนซ์ของชุดโปรแกรมพัฒนาทุกโปรแกรม และบันทึกอยู่ในรูปแบบของเอ็กซ์เอมแอลเป็นที่เรียบร้อยแล้ว จะทำการวัดตัววัดแผนภาพที่ได้กำหนดไว้ด้วยโปรแกรม SDMetrics เวอร์ชัน 2.0 [15] ซึ่งสามารถอ่านข้อมูลแผนภาพจากไฟล์เอ็กซ์เอมแอลและทำการวัดค่าตัววัดแผนภาพต่างๆ ได้ โดยได้มีการเพิ่มเติมตัววัดแผนภาพเข้าไป และรายละเอียดของตัววัดแผนภาพที่ทำการเพิ่มเติมอยู่ในบทที่ 5 เมื่อทำการคำนวณตัววัดแผนภาพแล้วสามารถส่งออกข้อมูลให้อยู่ในรูปแบบไฟล์ซีเอสวี (CSV) เพื่อนำไปใช้ในการหาความสัมพันธ์กับความเสถียรเชิงตรรกะของคลาส เพื่อสร้างโมเดลประมาณค่าความเสถียรเชิงตรรกะจากแผนภาพในบทที่ 4

ตัววัดแผนภาพที่ใช้ในงานวิจัยนี้สามารถแบ่งออกได้เป็น 2 กลุ่มคือ ตัววัดแผนภาพคลาสและตัววัดแผนภาพซีเควนซ์ โดยตัววัดแผนภาพคลาสประกอบไปด้วย

1. ตัววัดจำนวนแอททริบิวต์ (NumAttr) คือ จำนวนของแอททริบิวต์ของคลาส
2. ตัววัดจำนวนโอเปอร์เรเตอร์ (NumOps) คือ จำนวนโอเปอร์เรเตอร์หรือเมทธอดของคลาส
3. ตัววัดจำนวนโอเปอร์เรเตอร์ที่มีขอบเขตการเข้าถึงเป็นแบบพับบลิค (NumPubOps) จำนวนโอเปอร์เรเตอร์หรือเมทธอดของคลาสที่มีขอบเขตการเข้าถึงเป็นพับบลิค

4. ตัววัดระดับความลึกคลาสภายใน (Nesting) คือ ระดับความลึกที่คลาสอยู่ในคลาสอื่นๆ
5. ตัววัดจำนวนอินเตอร์เฟสที่ใช้สร้างคลาส (IFImpl) คือ จำนวนของอินเตอร์เฟสที่คลาสที่สนใจนำมาใช้ในการสร้างคลาส
6. ตัววัดจำนวนคลาสลูก (NOC) คือ จำนวนของคลาสลูกของคลาสที่สนใจ
7. ตัววัดจำนวนคลาสแม่ (NOP) คือจำนวนคลาสแม่ของคลาสที่สนใจ
8. ตัววัดจำนวนคลาสสืบทอด (NumDesc) คือ จำนวนของคลาสลูกรวมไปถึงคลาสที่สืบทอดจากคลาสลูกและต่อไป
9. ตัววัดจำนวนคลาสบรรพบุรุษ (NumAnc) คือ จำนวนของคลาสแม่และคลาสต้นที่คลาสแม่สืบทอดมา
10. ตัววัดระดับความลึกของการสืบทอด (DIT) คือ ระดับของความลึกของคลาสที่สนใจในแผนผังต้นไม้การสืบทอด
11. ตัววัดความลึกจากคลาสสู่ปลายสุดของการสืบทอด (CLD) คือ ระดับของความลึกจากคลาสที่สนใจจนถึงระดับที่ลึกที่สุดในแผนผังต้นไม้การสืบทอด
12. ตัววัดจำนวนโอเปอร์เรเตอร์ที่รับสืบทอดมา (Opsinh) คือ จำนวนของโอเปอเรเตอร์หรือเมทธอดที่รับสืบทอดมาจากคลาสอื่น
13. ตัววัดจำนวนแอททริบิวต์ที่รับสืบทอดมา (Attrinh) คือจำนวนของแอททริบิวต์ที่รับสืบทอดมาจากคลาสอื่น
14. จำนวนครั้งที่คลาสมีความสัมพันธ์แอสโซซิเอชันในฐานะผู้เรียกใช้ (NumAss_User) คือ จำนวนความสัมพันธ์แอสโซซิเอชันของคลาสที่สนใจในแผนภาพคลาส เมื่อคลาสที่สนใจเป็นผู้ใช้งานคลาสอื่น
15. จำนวนครั้งที่คลาสมีความสัมพันธ์แอสโซซิเอชันในฐานะผู้ถูกเรียกใช้(NumAss_Provider) คือจำนวนความสัมพันธ์แอสโซซิเอชันของคลาสที่สนใจในแผนภาพคลาส เมื่อคลาสที่สนใจเป็นผู้ถูกใช้งานจากคลาสอื่น
16. จำนวนครั้งที่คลาสถูกใช้เป็นพารามิเตอร์ (EC_Par) คือ จำนวนครั้งที่คลาสที่สนใจถูกนำไปใช้เป็นพารามิเตอร์ในคลาสอื่นและคลาสตัวเอง
17. จำนวนพารามิเตอร์ในคลาสที่มีชนิดเป็นคลาสอื่นๆ (IC_Par) คือ จำนวนของพารามิเตอร์ในคลาสที่สนใจที่มีชนิดเป็นคลาสอื่นๆ

สำหรับตัววัดแผนภาพซีเควนซ์ประกอบไปด้วย

1. จำนวนคลาสสิฟายเออร์ที่มีคลาสนี้เป็นพื้นฐาน (ClassifInst) คือ จำนวนครั้งที่คลาสที่สนใจปรากฏในแผนภาพซีเควนซ์ หรือคือจำนวนออบเจคต์บนแผนภาพซีเควนซ์ของคลาสที่สนใจ
2. จำนวนครั้งการส่งแมสเสจไปยังคลาสอื่น (MsgSent) คือ จำนวนครั้งที่คลาสที่สนใจส่งแมสเสจไปยังคลาสอื่นๆ
3. จำนวนครั้งที่ได้รับแมสเสจจากคลาสอื่น (MsgRecv) คือ จำนวนครั้งที่คลาสที่สนใจได้รับแมสเสจจากคลาสอื่นๆ
4. จำนวนครั้งที่ส่งแมสเสจหาตัวเอง (MsgSelf) คือ จำนวนครั้งที่คลาสที่สนใจส่งแมสเสจไปยังตนเอง