บทที่ 5

การวัดผลเปรียบเทียบโมเดล

งานวิจัยนี้ได้ทำการเปรียบเทียบผลของการออกแบบระหว่างเอโอเอ็มเดิมและเอโอเอ็มที่แปลงเป็นยูนิตโมเดลเพื่อเปรียบเทียบคุณลักษณะต่างๆของซอฟต์แวร์ทั้งสองโมเดล คุณลักษณะงานวิจัยนี้สนใจนำมาเปรียบเทียบนั้นคือ ความสามารถในการบำรุงรักษา (Maintainability) ซึ่งความสามารถในการบำรุงรักษานั้นมีคุณลักษณะย่อยอย่างหนึ่ง คือ ความสามารถในการทำความเข้าใจ (Understandability) และเนื่องจากความสามารถในการทำความเข้าใจนั้นมีความสัมพันธ์กับค่าความซับซ้อน (Complexity) ของแผนภาพต่างๆ ดังนั้นหากค่าความซับซ้อนนั้นมีค่ามาก ย่อมทำให้นักพัฒนาต้องใช้เวลาในการทำความเข้าใจมากขึ้นและส่งผลต่อต้นทุนในการบำรุงรักษาที่มากขึ้นไปด้วย

ด้วยเหตุผลดังกล่าว งานวิจัยนี้จึงทำการวัดค่าความซับซ้อนของเอโอเอ็มและเอโอเอ็มแบบยูนิตโมเดลด้วยมาตรวัดดังต่อไปนี้

5.1. กำหนดมาตรวัดค่าความซับซ้อน

มาตรวัดค่าความซับซ้อนของโครงสร้างแผนภาพคลาสถูกนำเสนอโดยงานวิจัย [11] ประกอบด้วย

1. Number of Classes (NC)
2. Number of Attributes (NA)
3. Number of Methods (NM)
4. Number of Associations (NAssoc)
5. Number of Aggregation (NAgg)
6. Number of Dependencies (NDep)
7. Number of Generalizations (NGen)
8. Number of Generalisations Hierarchies (NGenH)
9. Maximum DIT
10. Maximum HAgg (MaxHAgg).

ในงานวิจัยของ [11] พบว่ามาตรวัดเหล่านี้มีผลต่อค่าความสามารถในการทำความเข้าใจและการบำรุงรักษา ดังนั้นมาตรวัดทั้ง 10 ตัวนี้จะถูกนำมาใช้วัดค่าความซับซ้อนของโครงสร้างแผนภาพคลาสทั้งเอโอเอ็มเดิมและเอโอเอ็มแบบยูนิต

เนื่องจากการแปลงจากเอโอเอ็มเป็นยูนิตโมเดลทำให้ได้แผนภาพที่อยู่ในระดับคอมโพเนนต์ ซึ่งเป็นแผนภาพที่อยู่ระดับสูงกว่าแผนภาพคลาส นั่นหมายความว่า ความซับซ้อนในการพัฒนาระบบโดยรวมจะเพิ่มขึ้น เพราะนักพัฒนาต้องทำความเข้าใจแผนภาพอีกหนึ่งระดับ

จาก [12] แผนภาพระดับแพ็กเกจนั้นเทียบได้กับโมดูลหรือคอมโพเนนต์ ดังนั้น เพื่อวัดความซับซ้อนโดยรวมจึงต้องทำการวัดที่ระดับแผนภาพแพ็กเกจหรือแผนภาพคอมโพเนนต์ด้วย

อย่างไรก็ตาม ในงานวิจัยนี้จะมุ่งประเด็นที่กลุ่มคลาสของเอโอเอ็มเท่านั้น ไม่ได้นำคลาสอื่นๆที่ไม่เกี่ยวข้องกับเอโอเอ็มมาพิจารณา และโดยปกติการจัดแพ็กเกจของเอนทิตีคลาสทั่วไป ก็มักจะจัดให้อยู่ในกลุ่มแพ็กเกจเดียวกัน ดังนั้นการเปรียบเทียบแพ็กเกจหนึ่งแพ็กเกจ กับ ยูนิตหลายยูนิตที่ได้จากการขั้นตอนการแปลงเอโอเอ็มของงานวิจัย ไม่สามารถเปรียบเทียบได้โดยตรง ด้วยเหตุนี้การวัดผลค่าความซับซ้อนในงานวิจัยนี้จึงมุ่งไปยังความซับซ้อนของแผนภาพคลาสเป็นหลัก

5.2. ผลการวัดค่าความซับซ้อน

ตารางที่ 9 แสดงผลการวัดค่าความซับซ้อนของแผนภาพคลาสที่ได้จากการพัฒนาระบบทดสอบ ซึ่งในตารางจะแสดงค่าเปรียบเทียบระหว่างระบบที่ออกแบบและพัฒนาโดยใช้เอโอเอ็มแบบเดิม และเอโอเอ็มแบบยูนิต

ตารางที่ 9 ผลการวัดค่าความซับซ้อนของแผนภาพคลาสทั้ง 3 ระบบ

มาตรวัด	ระบบที่ 1: CFGen		ระบบที่ 2: ทะเบียนสินค้า		ระบบที่ 3: วัดประสิทธิภาพโฆษณา	
	AOM	AOM – UM	AOM	AOM – UM	AOM	AOM – UM
NC	22	18	14	14	14	14
NA	59	56	48	48	47	39
NM	134	125	104	104	102	87
NAssoc	14	13	13	13	13	10
NAgg	0	0	1	1	1	1
NDep	0	0	0	0	1	1
NGen	10	7	4	4	3	5
NGenH	3	2	2	2	1	2
Max DIT	1	2	1	1	1	1
Max HAgg	0	0	1	1	1	1

5.3. เปรียบเทียบผลการวัดค่าความซับซ้อนของโมเดลทั้งสอง

เมื่อเปรียบเทียบผลการวัดค่าความซับซ้อนของแผนภาพคลาสทั้งสองโมเดล พบว่า

ในระบบที่ 1 พบว่ามาตรวัดจำนวนคลาส (NC), จำนวนแอตทริบิวต์ (NA), จำนวนเมท็อด (NM) และ จำนวนการสืบทอด (NGen) มีค่าลดลง ซึ่งเป็นผลของการแปลงคลาสที่สืบทอดจากเอนทิตีไทป์ให้อยู่ในรูปของอินสแตนท์ของเอนทิตีไทป์ ส่วนกรณีของการเพิ่มของค่า จำนวนระดับชั้นความลึกของโครงสร้างแบบการสืบทอดที่ค่ามากที่สุด (Max DIT) สืบเนื่องมาจากการทำรีแฟคทอริ่งคลาส Entity ออกมาเพื่อประโยชน์ในการนำกลับมาใช้ใหม่ของยูนิตเอนทิตี จึงทำให้ระดับชั้นความลึกของการสืบทอดเพิ่มขึ้นอีกหนึ่งชั้น

ดังนั้น ผลของการแปลงให้อยู่ในรูปของยูนิตโมเดลสำหรับระบบที่ 1 นี้ ทำให้ค่ามาตรวัดโดยส่วนใหญ่ลดลง นั่นหมายความว่าวิธีการแปลงจากเอโอเอ็มเดิมเป็นยูนิตโมเดลที่นำเสนอนั้นจะช่วยลดความซับซ้อนของแผนภาพคลาสได้

ในระบบที่ 2 พบว่ามาตรวัดมีค่าเท่ากันไม่เปลี่ยนแปลง ทั้งนี้เนื่องจากแผนภาพของเอโอเอ็มเดิมนั้น คลาสถูกออกแบบให้เป็นแบบเจนเนอริก (Generic) คลาสของโดเมน แอตทริบิวต์ เมท็อด ความสัมพันธ์ที่อาจเกิดขึ้นใหม่นั้น โมเดลสามารถรองรับความเปลี่ยนแปลงเหล่านี้ได้ โมเดลไม่มีการประยุกต์ใช้แพทเทิร์นซ้ำกันหลายครั้ง ซึ่งตรงกับแนวทางที่ใช้ในการแปลงให้อยู่ในรูปของยูนิตโมเดล ทำให้แผนภาพคลาสที่ได้หลังการแปลงให้อยู่ในรูปของยูนิตโมเดลจึงยังคงได้คลาสและความสัมพันธ์เหมือนเดิม

ดังนั้นผลของการแปลงให้อยู่ในรูปแบบของยูนิตโมเดลสำหรับระบบที่ 2 นี้ ทำให้ค่าของมาตรวัดมีค่าเท่ากัน ซึ่งหมายความว่าการแปลงให้อยู่ในรูปแบบยูนิตโมเดลไม่ได้ช่วยลดค่าความซับซ้อนของแผนภาพในระบบนี้

ในระบบที่ 3 พบว่ามาตรวัด จำนวนแอตทริบิวต์ (NA), จำนวนเมท็อด (NM) จำนวนความสัมพันธ์แบบแอสโซซิเอชั่น (NAssoc) มีค่าลดลง เนื่องจากการออกแบบมีการใช้แพทเทิร์นซ้ำกันสองครั้งคือ ไทป์อ็อบเจกต์แพทเทิร์น เมื่อแปลงด้วยวิธีการที่นำเสนอแล้ว คลาสที่ซ้ำกันจะถูกรวมกันเป็นเพียงคลาสเดียว ทำให้แอตทริบิวต์ เมท็อด และเส้นความสัมพันธ์ลดลง แต่สำหรับค่าของจำนวนของความสัมพันธ์แบบการสืบทอด (NGen) และ จำนวนโครงสร้างแบบการสืบทอด (NGenH) นั้นเพิ่มขึ้นเนื่องจากการรีแฟคทอริงคลาส Entity ออกมาเพื่อการนำกลับมาใช้ใหม่และให้เอนทิตีคลาสของโดเมนคือ คลาส Media และคลาส Metric ทำการสืบทอดมาจากคลาส Entity ก็เพื่อรักษาคลาสที่เป็นตัวแทนของโดเมนนั้นๆไว้ เพื่อให้ง่ายต่อการทำความเข้าใจ

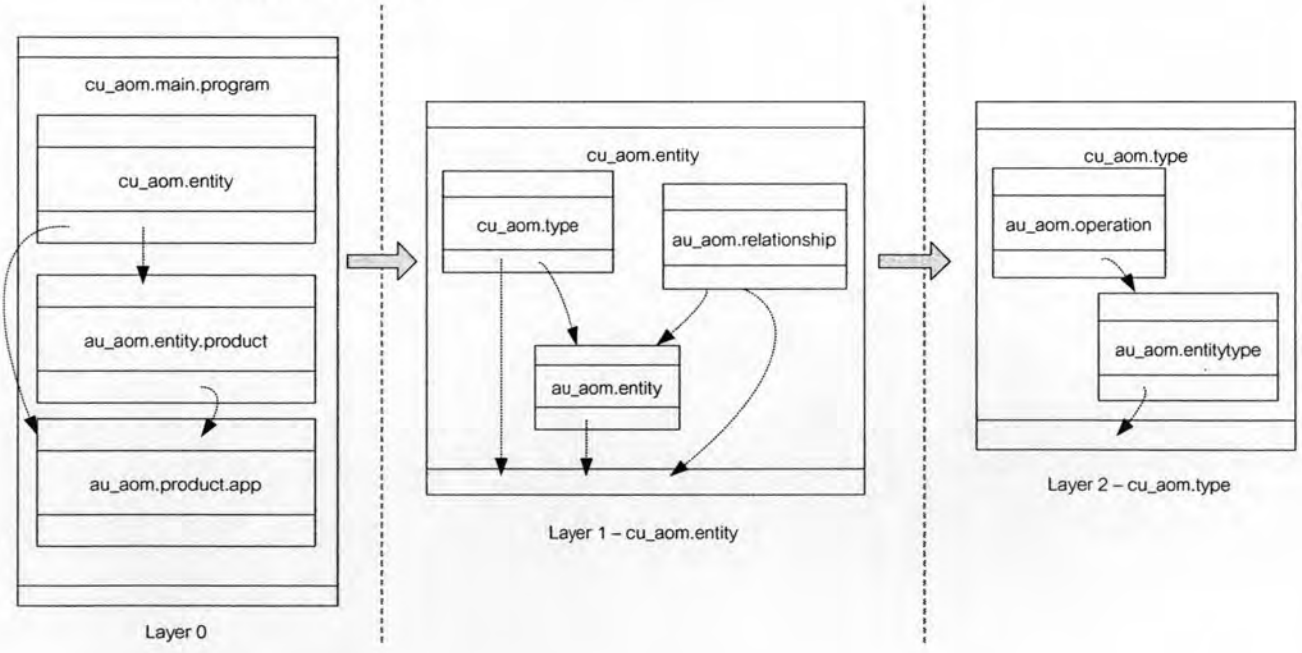
ดังนั้นผลของการแปลงให้อยู่ในรูปแบบของยูนิตโมเดลสำหรับระบบที่ 3 นี้ ทำให้ค่ามาตรวัดโดยส่วนใหญ่ลดลง นั่นหมายความว่าวิธีการแปลงจากเอโอเอ็มเดิมเป็นยูนิตโมเดลที่นำเสนอนั้นจะช่วยลดความซับซ้อนของแผนภาพคลาสได้

อย่างไรก็ตาม การเปรียบเทียบผลดังกล่าวเป็นการเปรียบเทียบเฉพาะความซับซ้อนของแผนภาพคลาสเพียงอย่างเดียว หากพิจารณาที่แผนภาพของยูนิตโมเดลที่ได้ แสดงให้เห็นว่าเป็นแผนภาพที่อยู่ในระดับสูงกว่าแผนภาพคลาสและสามารถแบ่งแยกเป็นระดับชั้นได้ ซึ่งรายละเอียดของการวิเคราะห์ผลจะกล่าวถึงในหัวข้อต่อไป

5.4. วิเคราะห์ผลลัพธ์ของการแปลงให้อยู่ในรูปแบบของยูนิตโมเดล

ผลที่ได้จากการแปลงเอโอเอ็มซึ่งอยู่ในรูปแบบดีไซน์แพทเทิร์นต่างๆเป็นยูนิตโมเดลนั้นพบว่า

- แอพพลิเคชันแบบเอโอเอ็มที่พัฒนาด้วยยูนิตโมเดล ทำให้โครงสร้างของแผนภาพคลาสถูกจัดกลุ่มให้อยู่ในรูปแบบของคอมโพเนนต์ ซึ่งเป็นแผนภาพที่อยู่ในระดับสูง (High Level) สถาปัตยกรรมของแอพพลิเคชันสามารถแยกเป็นระดับชั้นได้ (Layer Architecture) ดังรูปที่ 5.1 เริ่มจากทางด้านซ้ายจะเห็นได้ว่ายูนิตของโปรแกรม cu_aom.main.program ถูกประกอบกันขึ้นมาจากยูนิต cu_aom.entity, au_aom.entity.product และ au_aom.product.app เมื่อมองลึกลงไปในยูนิต cu_aom.entity ก็จะพบว่า ภายในประกอบด้วยยูนิตย่อยๆ ได้แก่ cu_aom.type, au_aom.relationship และ au_aom.entity เป็นระดับชั้นลึกลงไป ซึ่งจะช่วยให้นักพัฒนาสามารถเริ่มทำความเข้าใจจากแผนภาพที่อยู่ในระดับสูงได้ก่อนจะลงลึกไปในรายละเอียดของแต่ละระดับชั้น ย่อมส่งผลให้สามารถทำความเข้าใจระบบได้ง่ายขึ้น



รูปที่ 5.1 ยูนิตโมเดลที่ประกอบขึ้นมาจากยูนิตย่อยๆและแบ่งเป็นระดับชั้นได้

- ในกรณีที่ระบบมีขนาดใหญ่ มีจำนวนคลาสของดีไซน์แพทเทิร์นมาก คลาสที่ถูกออกแบบไว้อาจถูกเรียกใช้ผิดวัตถุประสงค์หรือไม่ตรงตามข้อกำหนดของดีไซน์ เช่น ไคลเอนต์จะต้องเรียกใช้ เมท็อดโดยอ้างถึงอินสแตนท์ผ่านทางแฟกทอรีคลาสเท่านั้น ซึ่งดีไซน์แพทเทิร์นนั้นไม่มีข้อกำหนดใดที่จะป้องกันข้อผิดพลาดที่อาจไม่ตั้งใจ ทำให้คลาสถูกเรียกใช้โดยตรง ส่งผลกระทบต่อการทำงานของโปรแกรมที่อาจจะผิดพลาด รวมทั้งยากต่อการบำรุงรักษา

  ดังนั้น แม้ว่าหลังจากการใช้ยูนิตโมเดลเข้ามาช่วยจะทำให้จำนวนคลาสของเอโอเอ็มเดิมกับเอโอเอ็มแบบยูนิตไม่แตกต่างกัน แต่หากสามารถแยกงานออกเป็นแต่ละส่วนอย่างชัดเจนด้วยยูนิต คลาสที่มีจำนวนมากจะถูกซ่อนรายละเอียดและป้องกันไม่ให้สามารถเรียกใช้ผิดไปจากข้อกำหนดที่ควรจะเป็นได้
- หากพิจารณาถึงคุณสมบัติด้านอื่นๆของยูนิตโมเดลเข้ามาประกอบ เช่น คุณสมบัติในเรื่องของการเชื่อมต่อภายนอกยูนิตทำให้ความสัมพันธ์ระหว่างยูนิตที่เกิดขึ้นมีความยืดหยุ่นต่อการถูกแทนที่ด้วยยูนิตใหม่ได้โดยที่ไม่ต้องแก้ไขโค้ดภายในยูนิต ทำการใช้บำรุงรักษาทำได้ง่ายขึ้น
- เนื่องจากดีไซน์แพทเทิร์นต่างๆที่ใช้ในการออกแบบเอโอเอ็มนั้น เป็นการประยุกต์ใช้แพทเทิร์น นั่นหมายความว่า เมื่อออกแบบเอโอเอ็มแอพพลิเคชัน รูปแบบการประยุกต์ใช้งานจะปรากฏซ้ำอยู่เสมอ ทำให้ผลที่ได้จากการแปลงเอโอเอ็มเป็นยูนิตโมเดลนั้น ปรากฏยูนิตที่สามารถนำกลับมาใช้ในการพัฒนาแอพพลิเคชันอื่นๆแบบเอโอเอ็มได้ ซึ่งยูนิตที่ได้นั้นก็คือ ยูนิต au_aom.entitytype, au_aom.operation, au_aom.relationship, au_aom.entity และ cu_aom.entity ยูนิตเหล่านี้จะถูกจัดเตรียมไว้พร้อมใช้งานกับเครื่องมือที่ใช้ในการสร้างยูนิตซึ่งจะกล่าวถึงรายละเอียดในบทต่อไป