การออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวา

นายเมธาวี แดงเพ็ง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2546
ISBN : 974-17-4131-6

DESIGN AND DEVELOPMENT OF A SOFTWARE TOOL FOR MEASURING REUSABILITY OF JAVA PROGRAMS

Mr. Maytawee Deangpheng

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computer Science

Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University

Academic year 2003
ISBN : 974-17-4131-6

หัวข้อวิทยานิพนธ์	การออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวา
โดย	นายเมธาวี แดงเพ็ง
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี
อาจารย์ที่ปรึกษา(ร่วม)	อาจารย์ นครทิพย์ พร้อมพูล

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.ดิเรก ลาวัณย์ศิริ)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ วันพร ปั้นเก่า)

..... อาจารย์ที่ปรึกษา
(ผู้ช่วยศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี)

..... อาจารย์ที่ปรึกษา (ร่วม)
(อาจารย์ นครทิพย์ พร้อมพูล)

..... กรรมการ
(รองศาสตราจารย์ ดร.สาธิต วงศ์ประทีป)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์)

เมธาวี แดงเพ็ง : การออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวา. (DESIGN AND DEVELOPMENT OF A SOFTWARE TOOL FOR MEASURING REUSABILITY OF JAVA PROGRAMS) อาจารย์ที่ปรึกษา: ผู้ช่วยศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี, อาจารย์ที่ปรึกษาร่วม: นครทิพย์ พร้อมพูล, 100 หน้า, ISBN 974-17-4131-6.

การวิจัยนี้มีวัตถุประสงค์เพื่อที่จะออกแบบและพัฒนาเครื่องมือ MTOOP รุ่นที่ 3 สำหรับการนำกลับมาใช้ใหม่ที่มีโปรแกรมต้นฉบับเป็นภาษาจาวา โดยการวิจัยนี้เป็นการทำวิจัยต่อจากงานวิจัย “การออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์สำหรับโปรแกรมเชิงวัตถุ” และโครงงาน “การออกแบบ และพัฒนาเครื่องมือวัดปัจจัยของความซับซ้อนของโปรแกรมเชิงวัตถุภาษาจาวา” ซึ่งได้มีการพัฒนาเครื่องมือ MTOOP (Metric Tool for Object-Oriented Programs) รุ่นที่ 1 และ 2 ตามลำดับ โดยสูตรการคำนวณและเกณฑ์สำหรับการวัดที่ใช้ในงานวิจัยนี้ได้มาจากผลงานวิจัยต่างๆ ที่เป็นที่เผยแพร่และเป็นที่ยอมรับในวงกว้างของการออกแบบและพัฒนาโปรแกรมเชิงวัตถุในปัจจุบัน

ค่าของตัววัดที่นำมาใช้ในงานวิจัยนี้ ช่วยให้ผู้ออกแบบและพัฒนาซอฟต์แวร์ทราบถึงคุณสมบัติในส่วนของความซับซ้อนภายในวิธีดำเนินการ ขนาดของโปรแกรม อัตราส่วนของข้อความอธิบาย การถ่ายทอดคุณสมบัติ การเข้าคู่กัน การห่อหุ้ม การพ้องรูป และการนำกลับมาใช้ใหม่ของซอฟต์แวร์จากค่าที่ได้จากการวัด และจากผลการทดสอบปรากฏว่าเครื่องมือ MTOOP รุ่นที่ 3 สามารถทำการหาค่าของมาตรวัดที่ได้กล่าวมาได้อย่างถูกต้อง ดังนั้นผู้ออกแบบและพัฒนาซอฟต์แวร์สามารถใช้เครื่องมือ MTOOP รุ่นที่ 3 นี้ เพื่อประเมินแนวโน้มสำหรับการนำโปรแกรมต้นฉบับภาษาจาวากลับมาใช้งานใหม่ จากค่าของปัจจัยต่างๆ ที่ได้จากการวัด

ภาควิชา วิศวกรรมคอมพิวเตอร์	ลายมือชื่อนิสิต……..
สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์	ลายมือชื่ออาจารย์ที่ปรึกษา……
ปีการศึกษา 2546	ลายมือชื่ออาจารย์ที่ปรึกษา (ร่วม)…….

4371471021 : MAJOR COMPUTER SCIENCE

KEY WORD: QUALITY, MEASUREMENT, REUSABILITY

MAYTAWEE DEANGPHENG: DESIGN AND DEVELOPMENT OF A SOFTWARE TOOL FOR MEASURING REUSABILITY OF JAVA PROGRAMS. THESIS ADVISOR: ASSISTANT PROFESSOR DR. PORNSIRI MUENCHAISRI, THESIS CO-ADVISOR: NAKORNTHIP PROMPOON, 100 pp, ISBN 974-17-4131-6.

The purpose of this study is to design and develop a software tool, MTOOP version 3.0, for measuring reusability of java programs. This thesis improves from the thesis, “The design and implementation of a measurement tool for object-oriented programs” and from the work, “The design and implementation of a software complexity-factor measurement tool for objected-oriented programs in java.” The two works design and develop MTOOP version 1.0 (Metric Tool for Object-Oriented Programs) and MTOOP version 2.0 respectively. Metrics and criteria of evaluations in the thesis are obtained from related research, which are widely and currently accepted by object-oriented design and development community.

The value of metrics in the thesis is useful for software designers and developers. They can use the tool to measure method cyclomatic complexity, method size, comment percentage, inheritance, coupling, encapsulation, polymorphism and reusability. According to the testing results, MTOOP version 3.0 can provide these metric values correctly. The software designers and developers can use the MTOOP version 3.0 to assess the reusability of the java programs from the reuse factor measurement.

Department Computer Engineering	Student’s signature……..
Field of study Computer Science	Advisor’s signature…….....
Academic year 2003	Co-advisor’s signature…….

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้เป็นผลสรุปจากการที่ผู้วิจัยได้รับโอกาสอันดี จากการที่ได้เข้ามาศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ในระหว่างการศึกษาผู้วิจัยได้รับความรู้ทั้งทางตรง และทางอ้อมจากคณาจารย์ภาควิชาวิศวกรรมคอมพิวเตอร์มากมาย โดยเฉพาะอย่างยิ่ง ผู้ช่วยศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี ซึ่งเป็นอาจารย์ที่ปรึกษา และอาจารย์ นครทิพย์ พร้อมพูล อาจารย์ที่ปรึกษาร่วม ที่ได้กรุณาให้ความรู้ คำปรึกษา และความช่วยเหลือต่างๆ ในการทำวิทยานิพนธ์จนสำเร็จลุล่วงลงด้วยดี และขอขอบคุณ ผู้ช่วยศาสตราจารย์ วันพร ปั้นเก่า รองศาสตราจารย์ ดร.สาธิต วงศ์ประทีป และผู้ช่วยศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์ กรรมการวิทยานิพนธ์ที่กรุณาเสียสละเวลาให้คำแนะนำ และตรวจสอบวิทยานิพนธ์ฉบับนี้

ขอขอบคุณพี่ๆ และเพื่อนๆ ที่ให้กำลังใจและข้อเสนอแนะต่างๆ ด้วยดี และขอขอบคุณท่านอื่นๆ ที่มีส่วนช่วยในการทำวิทยานิพนธ์ที่ไม่ได้กล่าวนามมา ณ โอกาสนี้ด้วย

ท้ายนี้ผู้วิจัยใคร่ขอกราบขอบพระคุณ บิดา มารดา ผู้ให้กำเนิดที่คอยอบรมสั่งสอนชี้แนะตลอดจนพี่สาวและน้องสาวของผู้วิจัยที่ให้การสนับสนุนทางด้านการศึกษา และคอยให้กำลังใจเสมอมาจนสำเร็จการศึกษา

เมธาวี แดงเพ็ง

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

หน้า

สารบัญตาราง(ต่อ)

หน้า

สารบัญรูปภาพ

หน้า

สารบัญรูปภาพ(ต่อ)

หน้า

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การพัฒนาโปรแกรมเชิงวัตถุ (Object-oriented programming) มีบทบาทที่สำคัญต่อการเปลี่ยนแปลงแนวทางในการพัฒนาซอฟต์แวร์ คุณลักษณะต่างๆ ของการพัฒนาโปรแกรมเชิงวัตถุ ไม่ว่าจะเป็นการสืบทอดคุณสมบัติ (Inheritance) การห่อหุ้ม (Encapsulation) และการพ้องรูป (Polymorphism) ล้วนเป็นปัจจัยส่งเสริมให้เกิดการนำองค์ประกอบ (Component) ของซอฟต์แวร์ในส่วนต่างๆ กลับมาใช้ใหม่ (Reuse) [1] ผู้พัฒนาสามารถทำการพัฒนาซอฟต์แวร์ใหม่ได้จากการรวบรวมองค์ประกอบย่อยๆ ที่ถูกออกแบบมาอย่างดีและเป็นมาตรฐานโดยทำการพัฒนาโปรแกรมเพียงเล็กน้อยเท่านั้นทำให้สามารถลดระยะเวลาที่ใช้ในการพัฒนาได้มาก

วิธีการนำกลับมาใช้ใหม่ (Reusability) หมายถึงระดับความสามารถในการนำองค์ประกอบที่เคยสร้างขึ้นแล้วกลับมาใช้ประโยชน์ใหม่ ความสามารถในการนำกลับมาใช้ใหม่ถือได้ว่าเป็นหัวใจสำคัญของซอฟต์แวร์เชิงวัตถุ เนื่องจากการนำกลับมาใช้ใหม่เป็นปัจจัยในการเพิ่มคุณภาพและเพิ่มผลผลิตของซอฟต์แวร์เชิงวัตถุ เพราะในการพัฒนาโครงงานขนาดใหญ่นั้นย่อมจะมีความคล้ายคลึงกันของปัญหาในหลายๆ ส่วน ดังนั้นการนำองค์ประกอบที่เคยสร้างไว้มาใช้ใหม่ก็จะช่วยลดเวลา (Time) ต้นทุน (Cost) และความพยายาม (Effort) ทั้งในส่วนของการพัฒนา (Implementation) การทดสอบ (Testing) และการบำรุงรักษา (Maintenance) ได้อย่างมาก แต่อุปสรรคของการนำกลับมาใช้ใหม่ในการพัฒนาซอฟต์แวร์เชิงวัตถุก็มีปัญหาต่างๆ มากมาย [2] เช่น ปัญหาที่เกิดจากการนำส่วนประกอบกลับมาใช้ใหม่จะต้องมีการวิเคราะห์และออกแบบอย่างละเอียด และต้องทำการทดสอบอย่างถี่ถ้วนเป็นต้น แม้ว่าเครื่องมือและกลไกของภาษาเชิงวัตถุจะเอื้อต่อการนำซอฟต์แวร์กลับมาใช้งานใหม่ แต่ปัจจุบันก็ยังไม่บรรลุวัตถุประสงค์หลักเพราะวัตถุประสงค์หลักของการนำซอฟต์แวร์กลับมาใช้ใหม่ คือจะต้องสร้างส่วนของซอฟต์แวร์ให้เป็นชิ้นส่วนมาตรฐานที่สามารถประกอบเป็นซอฟต์แวร์ประยุกต์เพื่อแก้ปัญหาใดๆ ก็ได้ [3]

การพัฒนาซอฟต์แวร์เชิงวัตถุให้มีคุณภาพ จำเป็นต้องอาศัยการวัดคุณภาพของซอฟต์แวร์ (Software Quality Measurement) เพราะถ้าไม่มีการวัดก็จะทำให้ไม่สามารถตรวจพบปัญหาที่เกิดขึ้นในกระบวนการสร้างซอฟต์แวร์ได้ การวัดจะทำหน้าที่เป็นการเตือนเกี่ยวกับปัญหาที่น่าจะเกิดขึ้น ทำให้ผู้พัฒนาสามารถป้องกันได้ล่วงหน้าและเป็นการรับประกันคุณภาพของซอฟต์แวร์อีก

ด้วย ดังนั้นการวัดการนำกลับมาใช้ใหม่ (Reusability Measurement) จึงเป็นการวัดเพื่อแสดงให้เห็นความเป็นไปได้ที่จะนำองค์ประกอบที่เคยสร้างขึ้นแล้วกลับมาใช้ประโยชน์ใหม่นั่นเอง

มาตรวัดที่นำมาใช้ในงานวิจัยนี้ได้แก่ การวัดแบบดั้งเดิมซึ่งประกอบด้วย การวัดความซับซ้อน การวัดขนาด และการหาอัตราส่วนของข้อความ การวัดเชิงวัตถุได้แก่ *MOOD* (Metrics for Object Oriented Design) R_{sf} (Reuse Size and Frequency) *RR* (Reuse Ratio) และ *RDCS* (Relative Degree of Code Saving) ซึ่งเป็นวิธีการวัดแบบหนึ่งของการวัดเชิงวัตถุเพราะสามารถนำไปใช้กับโปรแกรมเชิงวัตถุใดๆ ได้ นอกจากนี้ในปัจจุบันยังไม่มีเครื่องมือวัดตัวใดที่นำ *MOOD*, *RR*, R_{sf} และ *RDCS* มาใช้วัดในส่วนของการนำกลับมาใช้ใหม่ของซอฟต์แวร์ภาษาจาวา หรือถ้าทำการวัดก็จะเป็นการวัดเฉพาะในส่วนของการสืบทอดคุณสมบัติเท่านั้นแต่ไม่ได้พิจารณาคุณสมบัติอื่นๆ ของโปรแกรมเชิงวัตถุเช่นการเข้าคู่กัน การห่อหุ้ม การพ้องรูป และการนำกลับมาใช้ใหม่ [4],[5],[6] ดังนั้นวิทยานิพนธ์นี้จึงมีความมุ่งหมายที่จะออกแบบและพัฒนาเครื่องมือ เพื่อใช้ในการวัดค่าของการนำองค์ประกอบกลับมาใช้ใหม่จากซอฟต์แวร์ที่ใช้ภาษาจาวาในการพัฒนา

1.2 วัตถุประสงค์

เพื่อออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวา

1.3 ขอบเขตการวิจัย

การดำเนินงานในการวิจัยจะอยู่ภายใต้ขอบเขต ดังต่อไปนี้

1.3.1 การวัดในส่วนของการนำกลับมาใช้ใหม่ จะทำการวัดในส่วนของความซับซ้อนภายในวิธีดำเนินการ(method) ขนาดของโปรแกรม อัตราส่วนของข้อความอธิบาย การถ่ายทอดคุณสมบัติ การเข้าคู่กัน การห่อหุ้ม การพ้องรูป และการนำกลับมาใช้ใหม่ เท่านั้น

1.3.2 โปรแกรมต้นฉบับที่นำมาหาค่าต้องสามารถผ่านการแปลความหมาย(compile) โดยไม่มีข้อผิดพลาด

1.3.3 ขนาดของระบบงานที่ใช้ในการทดสอบต้องมีขนาดไม่น้อยกว่า 1000 บรรทัดและทำการทดสอบกับระบบงานอย่างน้อย 3 ระบบ

1.3.4 สามารถเก็บข้อมูลค่าวัดต่างๆ ลงฐานข้อมูลของไมโครซอฟท์แอคเซสได้

1.4 ขั้นตอนการดำเนินการ

วิทยานิพนธ์นี้มีขั้นตอนและวิธีการดำเนินงานดังนี้

1.4.1 ศึกษามาตรวัดต่างๆ ที่ใช้ในการประเมินค่าสำหรับการนำซอฟต์แวร์กลับมาใช้ใหม่

1.4.2 ศึกษาหลักไวยากรณ์ของโปรแกรมภาษาจาวา เพื่อหาวิธีการในการวัดค่าการนำกลับมาใช้ใหม่ของซอฟต์แวร์

1.4.3 ออกแบบส่วนของโปรแกรมที่ใช้มาตรวัด เพื่อประเมินค่าการนำกลับมาใช้ใหม่ของซอฟต์แวร์

1.4.4 พัฒนาต้นแบบที่ใช้การประเมินค่าการนำกลับมาใช้ใหม่ของซอฟต์แวร์ ที่ได้ออกแบบไว้

1.4.5 ทดสอบมาตรวัดที่ใช้ และต้นแบบที่ได้ออกแบบไว้

1.4.6 วิเคราะห์ผลที่ได้จากการทดสอบและสรุปผล

1.4.7 สรุปผลการวิจัยและเสนอแนะผลของการวิจัย

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ที่คาดว่าจะได้รับจากวิทยานิพนธ์มีดังนี้

1.5.1 ช่วยให้ทราบคุณสมบัติในส่วนของความซับซ้อนภายในวิธีดำเนินการ ขนาดของโปรแกรม อัตราส่วนของข้อความอธิบาย การถ่ายทอดคุณสมบัติ การเข้าคู่กัน การห่อหุ้ม การพ้องรูป และการนำกลับมาใช้ใหม่ ของซอฟต์แวร์จากค่าที่ได้จากการวัด

1.5.2 สามารถนำองค์ประกอบ ที่ได้จากการประเมินคุณภาพของซอฟต์แวร์ที่มีอยู่เดิมมาใช้กับโครงงานใหม่ได้

1.5.3 ลดเวลา ค่าใช้จ่าย และความพยายามในการสร้าง และดูแลโครงงานใหม่ที่สัมพันธ์กับโครงงานเดิม

1.5.4 นำค่าวัดต่างๆ ที่ได้มาทำการเปรียบเทียบกับเกณฑ์ที่กำหนดไว้ เพื่อช่วยในการตัดสินใจในการนำองค์ประกอบกลับมาใช้กับโครงงานใหม่ได้

1.5.5 เป็นแนวทางสำหรับการพัฒนาเครื่องมือที่ใช้วัดซอฟต์แวร์เชิงวัตถุอื่นๆ ในส่วนของการนำองค์ประกอบกลับมาใช้ใหม่

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ประเภทของการนำกลับมาใช้ใหม่

การนำกลับมาใช้ใหม่มีอยู่ 2 ประเภทคือ การนำกลับมาใช้ใหม่โดยบังเอิญ (Accidental reuse) และการนำกลับมาใช้ใหม่โดยเจตนา (Deliberate reuse) ถ้านักพัฒนาพบว่าส่วนประกอบของผลิตภัณฑ์เก่าที่ได้พัฒนาไว้ สามารถนำมาใช้กับผลิตภัณฑ์ใหม่ที่กำลังพัฒนาอยู่ได้จะเรียกว่าเป็นวิธีการของการนำกลับมาใช้ใหม่โดยบังเอิญ ในทางกลับกันการใช้ประโยชน์จากส่วนประกอบที่ถูกออกแบบไว้ ให้สามารถนำกลับมาใช้ใหม่ได้ในอนาคตจะเรียกว่าเป็นการนำกลับมาใช้ใหม่โดยเจตนา ซึ่งการนำกลับมาใช้ใหม่โดยเจตนาดีกว่าการนำกลับมาใช้ใหม่โดยบังเอิญคือส่วนประกอบที่ถูกสร้างมานั้นจะนำไปใช้ใหม่ได้ง่ายและปลอดภัยกว่าเนื่องจากมีเอกสารประกอบที่ดี และมีการทดสอบอย่างถี่ถ้วน นอกจากนั้นยังมีรูปแบบไปในทิศทางเดียวกับระบบอื่นๆ ทำให้สามารถดูแลรักษาระบบได้ง่าย

ตารางที่ 2.1 แสดงให้เห็นการเปลี่ยนแปลงของระบบว่ามีความสัมพันธ์อย่างไรต่อการนำกลับมาใช้ใหม่ในแต่ละประเภทดังต่อไปนี้

1) **การย้ายระบบ** (Rehosting) หมายถึงการย้ายระบบที่มีอยู่แล้วไปอยู่บนระบบปฏิบัติการหรือฮาร์ดแวร์ใหม่ โดยให้คงหน้าที่เดิมที่ระบบได้ทำไว้ซึ่งอาจหมายถึงการเขียนบางส่วนของโปรแกรมใหม่ กรรมวิธีนี้มักใช้เวลาและความพยายามมาก
2) **การเปลี่ยนแปลงเป้าหมาย** (Retargeting) หมายถึงการเปลี่ยนแปลงเป้าหมายของระบบในด้านความต้องการของผู้ใช้ หรือตัวผู้ใช้เองซึ่งทำให้ต้องมีการวิเคราะห์และออกแบบใหม่
3) **การใช้ส่วนประกอบที่ยังใช้ได้** (Salvaging) หมายถึงการพยายามใช้งานส่วนประกอบที่ยังใช้งานได้
4) **การย้าย** (Porting) หมายถึงการนำรหัสโปรแกรมไปแปลความหมาย (Recompile) ใหม่บนระบบปฏิบัติการหรือฮาร์ดแวร์ใหม่

5) **การปรับปรุง** (Tailoring) หมายถึงการดัดแปลงให้เข้ากับความต้องการใหม่ๆ มักจะเกิดขึ้น ในองค์กรที่มีการจัดทำโครงงานให้มีความสามารถในการนำกลับมาใช้ใหม่ที่ดี โดยการวิเคราะห์และออกแบบอย่างระมัดระวังและละเอียดถี่ถ้วน

6) **การนำส่วนประกอบกลับมาใช้ใหม่** (Assembling) หมายถึงการนำส่วนประกอบที่สามารถนำกลับมาใช้ใหม่มาประกอบกันเพื่อสร้างระบบใหม่ขึ้น

ตารางที่ 2.1 แสดงการเปลี่ยนแปลงระบบซึ่งมีความสัมพันธ์ต่อการนำกลับมาใช้ใหม่ในแต่ละประเภท [2]

สิ่งที่เปลี่ยนแปลง / รูปแบบของการนำกลับมาใช้ใหม่	**โดยบังเอิญ**	**โดยเจตนา**
ฮาร์ดแวร์ หรือ ซอฟต์แวร์ระบบ	การย้ายระบบ	การย้าย
ผู้ใช้งาน งานที่ได้รับมอบหมายหรือการติดตั้ง	การเปลี่ยนแปลงเป้าหมาย	การปรับปรุง
หน้าที่การทำงาน หรือ คุณสมบัติในการใช้งาน	การใช้ส่วนประกอบที่ยังใช้ได้	การนำส่วนประกอบกลับมาใช้ใหม่

จากตารางที่ 2.1 จะเห็นได้ว่าถึงแม้การนำกลับมาใช้ใหม่โดยบังเอิญจะเป็นรูปแบบหนึ่งของการนำกลับมาใช้ใหม่ แต่จะมีประโยชน์ในระยะของการพัฒนาระบบเท่านั้นและยังมีประโยชน์น้อยกว่าการนำกลับมาใช้ใหม่โดยเจตนาอีกด้วย

2.1.2 **การวัดทางซอฟต์แวร์** (Software Measurement)

วัตถุประสงค์ของการวัดเชิงวัตถุ (Object–Oriented Measurement) ในทางวิศวกรรมซอฟต์แวร์สามารถสรุปได้ดังนี้ [8]

1) เพื่อที่จะเข้าใจถึงคุณภาพของระบบงานที่สร้างได้ดีขึ้น
2) เพื่อใช้ในการประเมินประสิทธิภาพของขั้นตอนในแต่ละส่วนของการทำงาน
3) เพื่อทำการปรับปรุงคุณภาพของงานในแต่ละระดับขั้นของโครงงานที่สร้าง

จะเห็นได้ว่า คุณภาพของซอฟต์แวร์เป็นสิ่งที่มีความสำคัญต่อการพัฒนาระบบงานเป็นอย่างยิ่ง ซึ่งคุณภาพของซอฟต์แวร์ในแต่ละด้านควรจะใช้เกณฑ์และมาตรวัดที่เหมาะสมและ

สมควรที่จะมีเครื่องมือ(Tool) ที่จะนำไปใช้ในการวัดคุณภาพของซอฟต์แวร์เพื่อความสะดวกรวดเร็วและมีความถูกต้อง

ในการวัดคุณภาพของโครงงานนั้นผู้พัฒนาโครงงานสามารถเลือกพารามิเตอร์ต่างๆ ได้มากมายในการวัดเช่นจำนวนบรรทัดที่นำกลับมาใช้ใหม่ ซึ่งเป็นวิธีการที่ใช้กับการพัฒนาแบบเก่า แต่ก็มีพารามิเตอร์บางอย่างที่ถูกสร้างขึ้นมาใหม่เพื่อใช้กับการวัดเชิงวัตถุ ซึ่งในงานวิจัยนี้จะอ้างถึงการวัดที่ใช้การวัด 3 ชนิดจากวิธีการดั้งเดิมและเพิ่มเติมอีก 10 ชนิดที่ใช้กับการพัฒนาเชิงวัตถุรวมเป็น 13 ชนิดดังตารางที่ 2.2 เหตุผลที่นำวิธีการแบบดั้งเดิมมารวมด้วยคือวิธีการเหล่านี้ได้รับการยอมรับและใช้ในการวัดคุณภาพของโครงงานในวงกว้างนั่นเอง [6]

ตารางที่ 2.2 มาตรวัดที่นำมาใช้ในการพัฒนาเครื่องมือ MTOOP (Metric Tool for Object-Oriented Programs)

SOURCE	METRIC	OBJECT-ORIENTED CONSTRUCT
Traditional Measurement	Cyclomatic complexity (CC)	Method
	Lines of Code (LOC)	Method
	Comment percentage (CP)	Method
Object-Oriented Measurement	Method Inheritance Factor (MIF)	Inheritance
	Attribute Inheritance Factor (AIF)	Inheritance
	Method Hiding Factor (MHF)	Encapsulation , information hiding
	Attribute Hiding Factor (AHF)	Encapsulation , information hiding
	Coupling Factor (COF)	Coupling
	Polymorphism Factor (PF)	Polymorphism
	Reuse Factor (RF)	Reuse
	Relative Degree of Code Saving (RDCS)	Inheritance Reuse
	Reuse Size and Frequency (R_{sf})	Size Reuse
	Reuse Ratio (RR)	Size Reuse

การวัดในทางวิศวกรรมซอฟต์แวร์ (Software Engineering) นั้นสามารถแบ่งได้เป็น 2 ประเภท คือ [9]

1) การวัดทางตรง (Direct Measurement)

คือการวัดเพื่อบ่งชี้ถึงผลผลิตที่ได้จากการวัด[8] โดยผลที่ได้จากการวัดทำให้ผู้พัฒนาทราบลักษณะในด้านโครงสร้างของซอฟต์แวร์เช่น จำนวนบรรทัด จำนวนคลาส จำนวนตัวแปร เป็นต้น

2) การวัดทางอ้อม (Indirect Measurement)

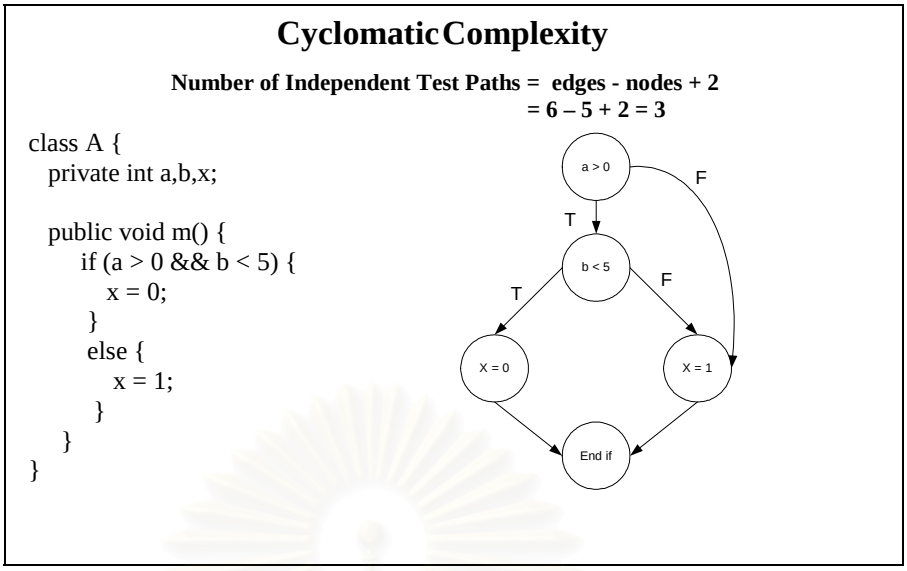
คือการวัดเพื่อบ่งชี้ถึงคุณภาพของซอฟต์แวร์[8] (Software Quality) ผลที่ได้จากการวัดทางอ้อมเช่น ความถูกต้อง (Correctness) ประสิทธิภาพ (Efficiency) และความยืดหยุ่น (Flexibility) เป็นต้น การที่จะทำการวัดเพื่อให้ทราบถึงคุณภาพของซอฟต์แวร์ในด้านใดด้านหนึ่งนั้นจะต้องกำหนดเกณฑ์ (Criteria) ในการวัดและนำมาตรวัดจากการวัดทางตรงมาใช้เพื่อให้ได้ผลของการวัดทางอ้อม

2.1.3 **การวัดแบบดั้งเดิม** (Traditional Measurement)

เป็นวิธีการวัดที่มีมาตั้งแต่การเขียนโปรแกรมแบบโครงสร้าง (Structured Programming) และยังเป็นที่นิยมและได้รับการยอมรับ เพื่อใช้ในการวัดคุณภาพของโครงงานกันอย่างแพร่หลายมาจนถึงปัจจุบัน ซึ่งวิธีการวัดแบบดั้งเดิมได้แก่

2.1.3.1 **การวัดความซับซ้อน** (Cyclomatic Complexity Measurement : CC) [9],[10]

การวัดความซับซ้อนภายในของแต่ละวิธีดำเนินการ สามารถหาค่าได้โดยนำจำนวนของเส้นเชื่อม (Edge) ซึ่งเป็นวิถีระหว่างโหนดทั้งหมดลบด้วยจำนวนของสมาชิกของข้อมูล (Node) ทั้งหมดแล้วนำไปบวก 2 ซึ่งในงานวิจัย “Applying and Interpreting Object-Oriented Metrics” ของ Linda H. Rosenberg ได้ระบุว่าถ้าค่าที่ได้จากการวัดมีค่าประมาณ 2-5 ต่อวิธีดำเนินการที่ทำการวัด แสดงว่าวิธีดำเนินการนั้นมีความซับซ้อนน้อย สามารถทำความเข้าใจได้ง่ายแต่การวัดความซับซ้อนไม่สามารถนำไปใช้วัดความซับซ้อนของคลาสได้ เพราะในแต่ละคลาสอาจมีการถ่ายทอดคุณสมบัติของคลาสลงมา ดังนั้นการวัดความซับซ้อนจึงทำได้เพียงการวัดความซับซ้อน ภายในวิธีดำเนินการของแต่ละคลาสได้เท่านั้น[9] ตัวอย่างของการหาค่าของการวัดความซับซ้อนแสดงได้ดังรูปที่ 2.1



รูปที่ 2.1 Cyclomatic Complexity

และสามารถแสดงสูตรการคำนวณได้ดังนี้

$$CC = \sum Edge - \sum Node + 2 \qquad \cdots\cdots\cdots(1)$$

โดยที่ :

$\sum Edge$ คือ ผลรวมของเส้นเชื่อมซึ่งเป็นวิถีระหว่างโหนดทั้งหมดของวิธีดำเนินการที่พิจารณา

$\sum Node$ คือ ผลรวมของสมาชิกของโหนดทั้งหมดของวิธีดำเนินการที่พิจารณา

2.1.3.2 **การวัดขนาด** (Size Measurement) [9],[10]

ขนาดของวิธีดำเนินการ มีผลกระทบกับความยากง่ายในการทำความเข้าใจการทำงานภายในวิธีดำเนินการ โดยที่การวัดขนาดของวิธีดำเนินการนั้นสามารถวัดได้จากหลายๆ วิธีที่แตกต่างกัน ซึ่งวิธีการเหล่านี้จะรวมถึงการนับจำนวนบรรทัดทั้งหมด จำนวนของข้อความสั่ง (Statement) จำนวนของบรรทัดว่าง และจำนวนของบรรทัดที่เป็นข้อความอธิบายที่มีอยู่ภายในแต่ละวิธีดำเนินการ เช่น Lines Of Code (LOC) จะเป็นการนับจำนวนบรรทัดทั้งหมดที่มีอยู่ภายในวิธีดำเนินการ Non Comment Non Blank (NCNB) จะเป็นการนับ

จำนวนบรรทัดภายในวิธีดำเนินการที่ไม่รวมบรรทัดที่เป็นข้อความอธิบาย และบรรทัดว่างเป็นต้น ซึ่งในงานวิจัย “Applying and Interpreting Object-Oriented Metrics” ของ Linda H. Rosenberg ได้ระบุว่าถ้าค่า LOC ที่ได้จากการวัดมีค่าประมาณ 80-100 บรรทัดต่อวิธีดำเนินการที่ทำการวัด แสดงว่าวิธีดำเนินการนั้นๆ ผู้พัฒนาสามารถทำความเข้าใจและบำรุงรักษาได้ดีขึ้น สามารถแสดงสูตรการคำนวณได้ดังนี้

$$LOC = MethodDeclaration() + \sum LocalVariableDeclaration() + \sum Statement() \quad \cdots\cdots\cdots (2)$$

โดยที่ :

$MethodDeclaration$ คือ จำนวนบรรทัดที่เป็นการประกาศใช้วิธีดำเนินการ

$\sum LocalVariableDeclaration$ คือ ผลรวมของจำนวนบรรทัดที่เป็นการประกาศค่าตัวแปรทั้งหมด ของวิธีดำเนินการที่พิจารณา

$\sum Statement$ คือ ผลรวมของจำนวนบรรทัดของข้อความสั่งภายในวิธีดำเนินการทั้งหมดที่พิจารณา

หมายเหตุ :

การนับจำนวนบรรทัดจะไม่นับรวมบรรทัดว่าง (Empty Statement) บรรทัดที่มีเครื่องหมายบล็อก (Block) และบรรทัดที่เป็นข้อความอธิบาย การประกาศตัวแปรหลายตัวอยู่ในบรรทัดเดียวกัน จะนับจำนวนบรรทัดเท่ากับจำนวนตัวแปรที่ประกาศ

2.1.3.3 **การหาอัตราส่วนของข้อความ** (Component Percentage Measurement : CP) [9],[10]

การหาอัตราส่วนของข้อความภายในแต่ละวิธีดำเนินการ สามารถหาได้โดยการนับจำนวนบรรทัดที่มีข้อความอธิบายทั้งหมด ถูกนำไปหารด้วยจำนวนบรรทัดทั้งหมดที่ลบด้วยจำนวนของบรรทัดว่างภายในวิธีดำเนินการนั้นๆ ซึ่งในงานวิจัย “Applying and Interpreting Object-Oriented Metrics” ของ Linda

H. Rosenberg ได้ระบุว่าถ้าค่า CP ที่ได้จากการวัดมีค่าประมาณ 20-30% ต่อวิธีดำเนินการที่ทำการวัดจะทำให้ผู้พัฒนาสามารถทำความเข้าใจ และบำรุงรักษาวิธีดำเนินการนั้นๆ ได้ดีขึ้น สามารถแสดงสูตรการคำนวณได้ดังนี้

$$CP = \frac{CommentCount}{LOC} \qquad \cdots\cdots\cdots(3)$$

โดยที่ :

CommentCount คือ จำนวนบรรทัดที่มีข้อความอธิบายทั้งหมด ของวิธีดำเนินการที่พิจารณา

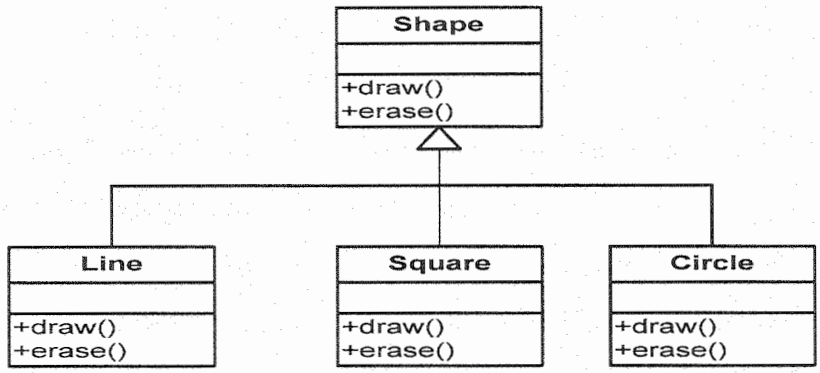
LOC คือ จำนวนบรรทัดทั้งหมดที่ลบด้วยจำนวนของบรรทัดว่างภายในวิธีดำเนินการนั้นๆ ของวิธีดำเนินการที่พิจารณา

2.1.4 การวัดเชิงวัตถุ

วัตถุประสงค์ของการวัดเชิงวัตถุก็เพื่อที่จะเข้าใจถึง คุณลักษณะและคุณภาพของโครงงาน สามารถประเมินประสิทธิภาพในแต่ละขั้นตอนของการทำงานได้ และการวัดเชิงวัตถุยังเป็นการวัดทางอ้อมอีกประเภทหนึ่ง โดยจะต้องกำหนดเกณฑ์ที่ใช้ในการวัดให้แน่นอนซึ่งวิธีการวัดได้แก่

2.1.4.1 **การถ่ายทอดคุณสมบัติ**[1],[3]

การถ่ายทอดคุณสมบัติเป็นเทคนิคในการนำคลาสกลับมาใช้งานใหม่ ซึ่งเป็นวิธีที่นิยมใช้และมีประสิทธิภาพมากวิธีหนึ่ง การถ่ายทอดคุณสมบัติจะใช้กับคลาสที่อยู่ในกลุ่มเดียวกันเช่นคลาสรูปทรงเรขาคณิต Shape จะมีการถ่ายทอดคุณสมบัติไปยังคลาสเส้นตรง Line สี่เหลี่ยมจัตุรัส Square และวงกลม Circle ซึ่งจะเห็นได้ว่าคลาสทั้ง 4 จัดอยู่ในกลุ่มเดียวกัน ดังแสดงในรูปที่ 2.2



รูปที่ 2.2 แสดงการถ่ายทอดคุณสมบัติของคลาส Shape ไปยังคลาส Line Square และ Circle

จากรูปที่ 2.2 เป็นไดอะแกรมคลาส ซึ่งแสดงการถ่ายทอดคุณสมบัติโดยลูกศรจะชี้ไปที่คลาสแม่ที่เรียกว่า "Superclass" หรือ "Based class" ส่วนคลาสที่รับคุณสมบัติเรียกว่าคลาสลูก "Subclass" หรือ "Derived class" โดยปกติแล้วคลาสลูกจะได้รับการถ่ายทอดคุณสมบัติซึ่งก็คือตัวแปรหรือวิธีดำเนินการจากคลาสแม่ทุกๆ ตัวแปรและวิธีดำเนินการ แต่ในบางกรณีเช่นการประกาศตัวแปรหรือวิธีดำเนินการในคลาสแม่ที่ไม่เอื้อต่อการถ่ายทอด เช่นตัวแปรที่กำหนดการเรียกใช้งานได้ภายในแต่ละคลาส ก็จะไม่สามารถถ่ายทอดคุณสมบัติไปยังคลาสลูกได้ ดังนั้นจึงอาจสรุปได้เป็น 2 แนวทางในการใช้คุณสมบัติของการถ่ายทอดดังนี้ [3]

1) คลาสลูกจะไม่ได้รับการถ่ายทอดวิธีดำเนินการที่มีชื่อเหมือนคลาสแม่ โดยที่วิธีดำเนินการนั้นไม่มีคำขยายใดๆ อยู่หน้าซึ่งเรียกว่าคอนสตรัคเตอร์ (Constructor) จากคลาสแม่

2) ตัวแปรหรือวิธีดำเนินการที่กำหนดการเรียกใช้งานได้ภายในแต่ละคลาสจะไม่ได้รับการถ่ายทอดคุณสมบัติ

คลาสแม่จะมีการถ่ายทอดคุณสมบัติให้คลาสลูก ดังนั้นคลาสลูกจึงไม่จำเป็นต้องเขียนคุณสมบัติที่คลาสแม่ถ่ายทอดมาให้ใหม่ แต่ถ้าวิธีดำเนินการมีการระบุซ้ำในคลาสลูกอีกแสดงว่ามีการปรับปรุงดัดแปลงวิธีดำเนินการดังกล่าวในคลาสลูก

สำหรับมาตรวัดที่ใช้ในการวัดการถ่ายทอดคุณสมบัติ ได้แก่

1) Method Inheritance Factor (MIF) เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการหาผลรวมของวิธีดำเนินการ ที่มีการถ่ายทอดคุณสมบัติของทุกๆ

คลาสที่พิจารณาภายในโครงงานที่ทำการวัด ซึ่งในงานวิจัย “Toward the Design Quality Evaluation of Object-Oriented Software Systems” ของ Fernando Brito Abreu ,Miguel Goulao และ Rita Esteves ได้ระบุว่าค่าวัดที่เหมาะสมของ MIF เท่ากับ 73.5% และไม่ควรต่ำกว่า 66.2% เพราะจะทำให้ผู้พัฒนาต้องใช้เวลาในการทำความเข้าใจ และบำรุงรักษาโครงงานนั้นๆ มากขึ้น เนื่องจากมีการสร้างวิธีดำเนินการขึ้นใหม่ภายในโครงงานที่ทำการวัดเป็นจำนวนมาก ซึ่งจะส่งผลให้มีการนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย และไม่ควรสูงกว่า 80.8% เพราะการที่ค่า MIF สูงกว่าเกณฑ์ที่กำหนดนั้นอาจจะมีสาเหตุมาจาก การออกแบบที่ผิดพลาดของโครงงานที่ทำการวัดก็ได้ [7] ซึ่งอาจจะส่งผลเสียถ้านำโครงงานนั้นกลับมาใช้ใหม่ และในกรณีค่า MIF เท่ากับ 0% แสดงว่าไม่มีการถ่ายทอดคุณสมบัติการทำงานภายในโครงงานที่ทำการวัด สามารถแสดงสูตรการคำนวณได้ดังนี้ [4],[6],[7]

$$MIF = \frac{\sum_{i=1}^{TC} M_i(C_i)}{\sum_{i=1}^{TC} M_a(C_i)} \qquad \cdots\cdots\cdots(4)$$

โดยที่ : $M_a(C_i) = M_d(C_i) + M_i(C_i)$

TC	คือ จำนวนคลาสทั้งหมดของโครงงานที่พิจารณา
$M_a(C_i)$	คือ จำนวนของวิธีดำเนินการทั้งหมดที่มีการใช้ในคลาส C_i
$M_d(C_i)$	คือ จำนวนของวิธีดำเนินการที่ถูกกำหนดขึ้นทั้งหมดในคลาส C_i
$M_i(C_i)$	คือ จำนวนของวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติ (ที่ไม่มีการ Override) ในคลาส C_i
$\sum_{i=1}^{TC} M_i(C_i)$	คือ ผลรวมของวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติทั้งหมด (ที่ไม่มีการ Override) ในคลาส C_i
$\sum_{i=1}^{TC} M_a(C_i)$	คือ ผลรวมของวิธีดำเนินการทั้งหมดที่มีการใช้ในคลาส C_i

2) Attribute Inheritance Factor (AIF) เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการหาผลรวมของลักษณะประจำ ที่มีการถ่ายทอดคุณสมบัติของทุกๆ คลาสที่พิจารณาภายในโครงงานที่ทำการวัด ซึ่งในงานวิจัย “Toward the Design Quality Evaluation of Object-Oriented Software Systems” ของ Fernando Brito Abreu ,Miguel Goulao และ Rita Esteves ได้ระบุว่าค่าวัดที่เหมาะสมของ AIF เท่ากับ 56.2% และไม่ควรต่ำกว่า 52.4% เพราะจะทำให้ผู้พัฒนาต้องใช้เวลาในการทำความเข้าใจ และบำรุงรักษาโครงงานนั้นๆ มากขึ้น เนื่องจากโครงงานที่ทำการวัดมีการใช้งานของลักษณะประจำจากคลาสที่มีการถ่ายทอดคุณสมบัติมาให้น้อย และมีการกำหนดลักษณะประจำขึ้นมาใหม่เป็นจำนวนมาก ส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย และไม่ควรสูงกว่า 60% เพราะการที่ค่า AIF สูงกว่าเกณฑ์ที่กำหนดมากๆนั้น อาจเกิดจากการออกแบบที่ผิดพลาดภายในโครงงานที่ทำการวัดได้ และในกรณีค่า AIF เท่ากับ 0% แสดงว่าไม่มีการถ่ายทอดคุณสมบัติของลักษณะประจำ ภายในโครงงานที่ทำการวัดสามารถแสดงสูตรการคำนวณได้ดังนี้ [4],[6],[7]

$$AIF = \frac{\sum_{i=1}^{TC} A_i(C_i)}{\sum_{i=1}^{TC} A_a(C_i)} \qquad \cdots\cdots\cdots(5)$$

โดยที่ : $A_a(C_i) = A_d(C_i) + A_i(C_i)$

TC	คือ จำนวนคลาสทั้งหมดของโครงงานที่พิจารณา
$A_a(C_i)$	คือ จำนวนของลักษณะประจำทั้งหมดที่มีการใช้ในคลาส C_i
$A_d(C_i)$	คือ จำนวนของลักษณะประจำที่ถูกกำหนดขึ้นทั้งหมดในคลาส C_i
$A_i(C_i)$	คือ จำนวนของลักษณะประจำที่มีการถ่ายทอดคุณสมบัติ (ที่ไม่มีการ Override) ในคลาส C_i
$\sum_{i=1}^{TC} A_i(C_i)$	คือ ผลรวมของลักษณะประจำที่มีการถ่ายทอดคุณสมบัติทั้งหมด (ที่ไม่มีการ Override) ในคลาส C_i

$\sum_{i=1}^{TC} A_a(C_i)$ คือ ผลรวมของลักษณะประจำทั้งหมดที่มีการใช้ในคลาส C_i

2.1.4.2 การห่อหุ้มและการซ่อนข้อมูล [1],[3]

การซ่อนข้อมูลทำให้เกิดส่วนของการติดต่อสำหรับติดต่อกับวัตถุ โดยให้เห็นโครงสร้างภายในให้น้อยที่สุด ส่วนการห่อหุ้มหมายถึงการที่ผู้ใช้งานจะไม่สามารถมองเห็นสิ่งที่อยู่ภายในวัตถุ แต่สามารถใช้วัตถุนั้นได้โดยเรียกวิธีดำเนินการของวัตถุนั้นๆ

การห่อหุ้มหรือการซ่อนข้อมูล เป็นเป้าหมายของการออกแบบในซอฟต์แวร์เชิงวัตถุทำให้วัตถุหนึ่ง ๆ ไม่สามารถแก้ไขข้อมูลของวัตถุอื่น ๆ ได้โดยตรง

การเรียกใช้คลาส วิธีดำเนินการ และตัวแปร แบ่งออกเป็น 3 ลักษณะด้วยกัน [3] คือ

1) Public ใช้กำหนดคลาส วิธีดำเนินการ หรือตัวแปร เพื่อให้ส่วนของโปรแกรมอื่นๆ สามารถเรียกใช้ได้

2) Private ใช้กำหนดคลาส วิธีดำเนินการ หรือตัวแปร ถ้าคลาสเป็น Private class จะต้องเป็นคลาสภายในคลาส (inner class) ส่วนต่างๆ ของคลาสภายนอก (outer class) เท่านั้น ที่สามารถใช้คลาสภายในได้ สำหรับวิธีดำเนินการหรือตัวแปรถ้าเป็น Private จะใช้ได้เฉพาะภายในคลาสเท่านั้น คลาสอื่นไม่สามารถเรียกใช้วิธีดำเนินการหรือตัวแปรที่เป็น Private ของคลาสอื่นได้

3) Protected เป็นการกำหนดให้ข้อมูลหรือวิธีปฏิบัติการ (operation) ของคลาสนั้นและคลาสลูกเข้าถึงได้จากภายในคลาสเดียวกันและคลาสอื่น โดยถ้าเป็นคลาสเดียวกันจะสามารถทำได้ทั้งการอ่านและเขียน แต่ถ้าเป็นคลาสอื่นจะทำได้เพียงการอ่านอย่างเดียว

สำหรับมาตรวัดที่ใช้ในการวัดการห่อหุ้มและการซ่อนข้อมูล ได้แก่

1) Method Hiding Factor (MHF) เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการหาผลรวมของวิธีดำเนินการทั้งหมด ที่เป็นการซ่อนข้อมูลของทุกๆ คลาสที่พิจารณาภายในโครงงานที่ทำการวัด ซึ่งในงานวิจัย “Toward the Design Quality Evaluation of Object-Oriented Software Systems”

ของ Fernando Brito Abreu ,Miguel Goulao และ Rita Esteves ได้ระบุว่าค่าวัดที่เหมาะสมของ MHF เท่ากับ 19.6% และไม่ควรต่ำกว่า 10.4% เพราะการที่ค่า MHF ต่ำกว่าเกณฑ์ที่กำหนดนั้นไม่ได้สรุปว่าการนำกลับมาใช้ใหม่ในส่วนของการห่อหุ้มและการซ่อนข้อมูลนั้น เหมาะแก่การนำกลับมาใช้ใหม่สูงขึ้นแต่ในทางตรงกันข้าม อาจเกิดจากการที่ผู้พัฒนาโครงงานออกแบบโครงงานผิดพลาดได้ และไม่ควรสูงกว่า 28.7% เพราะถ้าค่าวัดสูงกว่าเกณฑ์ที่กำหนด แสดงว่าโครงงานที่ทำการวัดนั้นมีจำนวนของวิธีดำเนินการที่มีการกำหนดการทำงาน ในลักษณะการเรียกใช้งานได้เฉพาะภายในแต่ละคลาสที่ทำการวัดเท่านั้น ซึ่งส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย ในกรณีค่า MHF เท่ากับ 0% แสดงว่าโครงงานที่ทำการวัดนั้น มีการกำหนดวิธีดำเนินการในลักษณะการทำงานแบบการใช้งานในลักษณะสาธารณะทั้งโครงงาน และถ้าหากค่าของ MHF เท่ากับ 100% แสดงว่าโครงงานที่ทำการวัดนั้นมีการกำหนดวิธีดำเนินการในลักษณะการเรียกใช้งานได้เฉพาะภายในแต่ละคลาสทั้งโครงงานเช่นกัน สามารถแสดงสูตรการคำนวณได้ดังนี้ [4],[6],[7]

$$MHF = \frac{\sum_{i=1}^{TC} M_h(C_i)}{\sum_{i=1}^{TC} M_d(C_i)} \qquad \cdots\cdots\cdots(6)$$

โดยที่ : $M_d(C_i) = M_v(C_i) + M_h(C_i)$

TC	คือ จำนวนคลาสทั้งหมดของโครงงานที่พิจารณา
$M_v(C_i)$	คือ จำนวนของวิธีดำเนินการที่กำหนดการใช้งานในลักษณะสาธารณะทั้งหมดที่สามารถมองเห็นได้จากส่วนของการติดต่อภายในคลาส C_i
$M_d(C_i)$	คือ จำนวนของวิธีดำเนินการทั้งหมดในคลาส C_i
$M_h(C_i)$	คือ จำนวนของวิธีดำเนินการที่ไม่สามารถมองเห็นได้จากส่วนของการพัฒนาโครงงานในคลาส C_i
$\sum_{i=1}^{TC} M_h(C_i)$	คือ ผลรวมของวิธีดำเนินการที่ไม่สามารถมองเห็นได้ทั้งหมด จากส่วนของการพัฒนาโครงงานในคลาส C_i
$\sum_{i=1}^{TC} M_d(C_i)$	คือ ผลรวมของวิธีดำเนินการทั้งหมดในคลาส C_i

2) Attribute Hiding Factor (AHF) เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการหาผลรวมของลักษณะประจำทั้งหมด ที่เป็นการซ่อนข้อมูลของทุกๆ คลาสที่พิจารณาภายในโครงงานที่ทำการวัด ซึ่งในงานวิจัย “Toward the Design Quality Evaluation of Object-Oriented Software Systems” ของ Fernando Brito Abreu ,Miguel Goulao และ Rita Esteves ได้ระบุว่าค่าวัดที่เหมาะสมของ AHF เท่ากับ 79.7% และไม่ควรต่ำกว่า 70.2% เพราะจะทำให้ผู้พัฒนาต้องใช้เวลาในการทำความเข้าใจ และบำรุงรักษาโครงงานนั้นๆ มากขึ้น เนื่องจากโครงงานที่ทำการวัดมีการกำหนดการใช้งานของลักษณะประจำให้คลาสอื่นๆ สามารถเรียกใช้งานได้สูงคือมีการกำหนดลักษณะประจำ ให้มีการใช้งานในลักษณะสาธารณะเป็นส่วนใหญ่ ทำให้โครงงานที่ทำการวัดมีความซับซ้อนเพิ่มขึ้น ส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลง และไม่ควรสูงกว่า 89.3% เพราะถ้าค่าที่วัดได้สูงกว่าเกณฑ์มากๆ แสดงว่าโครงงานที่ทำการวัดนั้นมีการกำหนดค่าของลักษณะประจำ ในลักษณะการเรียกใช้ภายในแต่ละคลาสเท่านั้นซึ่งไม่อาจสรุปได้ว่า การนำกลับมาใช้ใหม่ในส่วนของการห่อหุ้มและการซ่อนข้อมูลของลักษณะประจำ เหมาะแก่การนำกลับมาใช้ใหม่สูงขึ้น แต่ในทางตรงกันข้าม อาจเกิดจากการที่ผู้พัฒนาโครงงานออกแบบโครงงานผิดพลาดได้ ในกรณีค่า AHF เท่ากับ 0% แสดงว่าโครงงานที่ทำการวัดนั้น มีการกำหนดลักษณะประจำในลักษณะสาธารณะทั้งโครงงานและถ้าหากค่าของ AHF เท่ากับ 100% แสดงว่าโครงงานที่ทำการวัดนั้นมีการกำหนดลักษณะประจำในลักษณะการทำงานแบบ การเรียกใช้ภายในแต่ละคลาสทั้งโครงงานเช่นกัน และถ้าค่า AHF ที่ได้จากการวัดมีค่าเท่ากับ 50% แสดงว่ามีการกำหนดค่าของลักษณะประจำ ในลักษณะการใช้งานแบบสาธารณะและการเรียกใช้ภายในแต่ละคลาสเท่าๆ กันสามารถแสดงสูตรการคำนวณได้ดังนี้ [4],[6],[7]

$$AHF = \frac{\sum_{i=1}^{TC} A_h(C_i)}{\sum_{i=1}^{TC} A_d(C_i)} \qquad \cdots\cdots\cdots(7)$$

โดยที่ : $A_d(C_i) = A_v(C_i) + A_h(C_i)$

TC คือ จำนวนคลาสทั้งหมดของโครงงานที่พิจารณา

$A_v(C_i)$ คือ จำนวนของลักษณะประจำที่กำหนดการใช้งานในลักษณะสาธารณะที่สามารถมองเห็นได้จากส่วนของการติดต่อภายในคลาส C_i

$A_d(C_i)$ คือ จำนวนของลักษณะประจำทั้งหมดภายในคลาส C_i

$A_h(C_i)$ คือ จำนวนของลักษณะประจำที่ไม่สามารถมองเห็นได้จากส่วนของการพัฒนาโครงงานภายในคลาส C_i

$\sum_{i=1}^{TC} A_h(C_i)$ คือ ผลรวมของลักษณะประจำที่ไม่สามารถมองเห็นได้ทั้งหมด จากส่วนของการพัฒนาโครงงานภายในคลาส C_i

$\sum_{i=1}^{TC} A_d(C_i)$ คือ ผลรวมของลักษณะประจำทั้งหมดภายในคลาส C_i

2.1.4.3 **การพ้องรูป** [1],[3]

Poly จะหมายถึงหลากหลาย ส่วน Morph หมายถึงรูปแบบ ในซอฟต์แวร์เชิงวัตถุกำหนดไว้ว่าวัตถุจะมีได้หลากหลายรูปแบบ ดังนั้นการพ้องรูปนั้นก็หมายถึงการกระทำแบบเดียวกันอาจมีรูปแบบที่แตกต่างกันไปในแต่ละคลาสได้

สำหรับการเขียนโปรแกรมเชิงวัตถุการพ้องรูปแบบดั้งเดิม (pure polymorphism) จะหมายถึงวิธีดำเนินการหนึ่งๆ ที่ทำงานได้หลายอย่าง แต่สำหรับการเขียนโปรแกรมเชิงวัตถุในปัจจุบันแล้วการพ้องรูปมีหลายลักษณะแต่ที่เห็นเด่นชัดมีอยู่ 2 รูปแบบคือ การพ้องรูปที่มีชื่อวิธีดำเนินการเดียวกัน แต่ใช้พารามิเตอร์เป็นตัวจำแนกความแตกต่างภายในวิธีดำเนินการอาจทำงานต่างกันหรือที่เรียกว่าเมทธอดโอเวอร์โหลดดิ้ง (method overloading) แบบที่สองเป็นการพ้องรูปที่มีลักษณะวิธีดำเนินการของคลาสแม่และคลาสลูกที่มีชื่อเหมือนกัน หรือที่เรียกว่าเมทธอดโอเวอร์ไรดิ้ง(method overriding) ซึ่งการพ้องรูปแบบนี้จะอยู่ก้ำกึ่งระหว่างการพ้องรูปแบบดั้งเดิมและการพ้องรูปแบบโอเวอร์โหลดดิ้ง ดังนั้นอาจสรุปได้ว่าการพ้องรูปสำหรับการเขียนโปรแกรมเชิงวัตถุแล้วก็คือวิธีดำเนินการเดียวที่รับข่าวสารเดียวกัน แต่เมื่ออยู่ต่างวัตถุอาจแสดงพฤติกรรมต่างกัน เมทธอดโอเวอร์โหลดดิ้งและเมทธอดโอเวอร์ไรดิ้งก็เป็นรูปแบบหนึ่งของการพ้องรูป

สำหรับมาตรวัดที่ใช้ในการวัดการพ้องรูป ได้แก่

Polymorphism Factor (PF) เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการวัดความสัมพันธ์ระหว่างวัตถุที่มาจากต่างคลาส โดยมีการตอบสนองต่อการกระทำบนพื้นฐานเดียวกันด้วยรูปแบบที่แตกต่างกันไป ซึ่งในงานวิจัย “Toward the Design Quality Evaluation of Object-Oriented Software Systems” ของ Fernando Brito Abreu ,Miguel Goulao และ Rita Esteves ได้ระบุว่าค่าวัดที่เหมาะสมของ PF เท่ากับ 6.5% และไม่ควรต่ำกว่า 3.5% เพราะค่า PF ที่ต่ำกว่าเกณฑ์มากๆ จะทำให้ไม่สามารถสรุปผลการวัดในส่วนของการนำกลับมาใช้ใหม่ของการพ้องรูปได้[7] และไม่ควรสูงกว่า 9.6% เพราะจะทำให้ผู้พัฒนาต้องใช้เวลาในการทำความเข้าใจและบำรุงรักษาโครงงานนั้นๆ มากขึ้น เนื่องจากโครงงานที่ทำการวัด จะมีจำนวนของวิธีดำเนินการในลักษณะของเมทธอดโอเวอร์โหลดดิ้งและเมทธอดโอเวอร์ไรดิ้งอยู่จำนวนมากทำให้โครงงานที่ทำการวัดนั้น มีความซับซ้อนสูงส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย ในกรณีที่ค่า PF เท่ากับ 100% แสดงว่าวิธีดำเนินการที่ใช้ภายในโครงงานมีลักษณะของการพ้องรูปทุกวิธีดำเนินการภายในโครงงานที่ทำการวัด ซึ่งส่วนหนึ่งอาจมีสาเหตุมาจากการออกแบบที่ผิดพลาดของผู้พัฒนาโครงงาน และในกรณีค่า PF เท่ากับ 0% แสดงว่าโครงงานที่ทำการวัดนั้น ไม่มีการทำงานในลักษณะของการพ้องรูปอยู่เลย สามารถแสดงสูตรการคำนวณได้ดังนี้ [4],[6],[7]

$$PF = \frac{\sum_{i=1}^{TC} M_o(C_i)}{\sum_{i=1}^{TC} \left[M_n(C_i) \times DC(C_i) \right]} \qquad \cdots\cdots\cdots(8)$$

โดยที่ : $M_n(C_i) = M_d(C_i) - M_o(C_i)$

TC คือ จำนวนคลาสทั้งหมดของระบบที่พิจารณา

$DC(C_i)$ คือ จำนวนของคลาสทั้งหมดที่มีการสืบทอดมาจากคลาส C_i

$M_d(C_i)$ คือ จำนวนของวิธีดำเนินการที่ถูกกำหนดขึ้นทั้งหมดภายในคลาส C_i

$M_n(C_i)$ คือ จำนวนของวิธีดำเนินการที่ถูกสร้างขึ้นใหม่ภายในคลาส C_i

$M_o(C_i)$ คือ จำนวนของวิธีดำเนินการที่ถูก Override ภายในคลาส C_i

$\sum_{i=1}^{TC} M_o(C_i)$ คือ ผลรวมของวิธีดำเนินการที่ถูก Override ทั้งหมดภายในคลาส C_i

$\sum_{i=1}^{TC} [M_n(C_i) \times DC(C_i)]$ คือ ผลรวมของวิธีดำเนินการที่ถูกสร้างขึ้นใหม่ภายในคลาส C_i คูณด้วยจำนวนของคลาสที่มีการสืบทอดมาจากคลาส C_i ทั้งหมด

2.1.4.4 **การเข้าคู่** (Coupling) [1],[3]

เมื่อมีวัตถุหรือคลาสหนึ่งใช้วัตถุหรือคลาสอื่นๆ เราจะเรียกการทำเช่นนี้ว่าการเข้าคู่กัน แหล่งของการเข้าคู่กันส่วนใหญ่จะเป็นการเข้าคู่กันระหว่างคลาสแม่กับคลาสลูก [3] การเข้าคู่กันยังรวมไปถึงเมื่อวิธีดำเนินการหรือฟิลด์ในคลาสอื่นๆ ถูกเรียกใช้หรือเมื่อวัตถุของคลาสอื่นๆ มีการส่งคำสั่งร้องขอโดยผ่านการอ้างถึงในวิธีดำเนินการ ตัวอย่างเช่นวัตถุ “X” มีการเข้าคู่กันกับวัตถุ “Y” ถ้า “X” ส่งคำสั่งร้องขอไปที่ “Y” เป็นต้น

สำหรับมาตรวัดที่ใช้ในการวัดการเข้าคู่ ได้แก่

Coupling Factor (COF) เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการวัดการเข้าคู่กันระหว่างคลาส แต่จะไม่รวมถึงการเข้าคู่กันในลักษณะที่เป็นการถ่ายทอดคุณสมบัติ ซึ่งในงานวิจัย “Toward the Design Quality Evaluation of Object-Oriented Software Systems” ของ Fernando Brito Abreu ,Miguel Goulao และ Rita Esteves ได้ระบุว่าค่าวัดที่เหมาะสมของ COF เท่ากับ 10.8% และไม่ควรต่ำกว่า 3.9% เพราะค่า COF ที่ต่ำกว่าเกณฑ์มากๆ จะทำให้ไม่สามารถสรุปผลการวัด ในส่วนของการนำกลับมาใช้ใหม่ของการเข้าคู่กันได้[7] และไม่ควรสูงกว่า 17.7% เพราะจะทำให้ผู้พัฒนาต้องใช้เวลาในการทำความเข้าใจและบำรุงรักษาโครงงานนั้นๆ มากขึ้น เนื่องจากโครงงานที่ทำการวัดมีการเรียกใช้วิธีดำเนินการหรือมีการเข้าคู่กันระหว่างคลาสสูง ทำให้โครงงานที่ทำการวัดมีความซับซ้อนส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย ในกรณีที่ค่า COF เท่ากับ 0% แสดงว่าโครงงานที่ทำการวัดไม่มีการเรียกใช้การทำงานจากคลาสอื่นๆ และในกรณีที่ค่าของ COF เท่ากับ 100% แสดงว่าโครงงานที่ทำ

การวัดนั้นอาจมีการออกแบบที่ผิดพลาดเกิดขึ้น สามารถแสดงสูตรการคำนวณได้ดังนี้ [4],[6],[7]

$$COF = \frac{\sum_{i=1}^{TC}\left[\sum_{j=1}^{TC} is\_client(C_i, C_j)\right]}{TC^2 - TC} \qquad \cdots\cdots\cdots(9)$$

โดยที่ : $is\_client(C_i, C_j) = \begin{cases} 1 & iff \quad C_i \Rightarrow C_j \wedge C_i \neq C_j \\ 0 & otherwise \end{cases}$

TC คือ จำนวนคลาสทั้งหมดของโครงงานที่พิจารณา

$TC^2 - TC$ คือ จำนวนของคลาสที่มีการเข้าคู่กันทั้งหมดของโครงงานที่พิจารณา

$C_i \Rightarrow C_j$ คือ การแสดงความสัมพันธ์ระหว่าง client class (C_i) และ supplier class (C_j) โดยที่เมื่อใดก็ตามที่ C_i มีการอ้างอิงถึงวิธีดำเนินการ หรือลักษณะประจำของ C_j จะเรียก C_i ว่าเป็น client class ของคลาส C_j

$\sum_{i=1}^{TC}\left[\sum_{j=1}^{TC} is\_client(C_i, C_j)\right]$ คือ ผลรวมของค่าที่ได้จากความสัมพันธ์ระหว่าง client class (C_i) และ supplier class (C_j) ทั้งหมด

2.1.4.5 **การนำกลับมาใช้ใหม่** (Reuse) [3]

การนำกลับมาใช้ใหม่ หมายถึงระดับความสามารถในการนำองค์ประกอบที่เคยสร้างขึ้นแล้วกลับมาใช้ประโยชน์ใหม่

สำหรับวิธีการนำคลาสกลับมาใช้งานใหม่สามารถแบ่งออกได้เป็น 4 วิธีคือ

1) นำโปรแกรมที่เขียนเก็บไว้มาใช้ใหม่
2) เขียนใหม่จากข้อมูลที่มีอยู่
3) นำคลาสมาใช้ในลักษณะคอมโพสิตชัน(Composition) คือการนำองค์ประกอบเล็กๆ มาประกอบกัน
4) นำคลาสกลับมาใช้ใหม่ภายใต้การถ่ายทอดคุณสมบัติ

สำหรับมาตรวัดที่ใช้ในการวัดการนำกลับมาใช้ใหม่ ได้แก่

1) Reuse Factor (RF) [1] เป็นมาตรวัดหนึ่งของ MOOD โดยจะทำการวัดการนำกลับมาใช้ใหม่ ซึ่งจะพิจารณาจากการนำส่วนประกอบจากไลบรารีที่มีอยู่มาใช้ใหม่ หรือมีการนำกลับมาใช้ใหม่ในลักษณะของการถ่ายทอดคุณสมบัติ ซึ่งในงานวิจัย “Object-Oriented Software Engineering : Measuring and Controlling the Development Process” ของ Fernando Brito Abreu และ Rogerio Carapuca ได้ระบุว่าถ้าค่า RF ที่วัดได้มีค่ามากกว่า 43% ต่อโครงงานที่ทำการวัด จะทำให้ผู้พัฒนาสามารถทำความเข้าใจและบำรุงรักษาโครงงานนั้นๆ ได้ดีขึ้น เนื่องจากมีการเรียกใช้การทำงานจากไลบรารีหรือจากการถ่ายทอดคุณสมบัติ จึงทำให้ช่วยลดปริมาณของวิธีดำเนินการภายในคลาสที่ทำการวัด แต่จะต้องใช้เวลาในการทดสอบโครงงานมากทำให้เวลาในการพัฒนาสูง เนื่องจากคลาสส่วนใหญ่ภายในโครงงานที่ทำการวัด มีการเรียกใช้วิธีดำเนินการจากไลบรารี หรือมีการเรียกใช้วิธีดำเนินการที่เกิดจากการถ่ายทอดคุณสมบัติสูง จึงทำให้เกิดความซับซ้อนภายในโครงงานแต่จะส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้น ในกรณีที่ค่า RF เท่ากับ 0% แสดงว่าทุกๆ คลาสภายในโครงงานที่ทำการวัด ไม่มีการนำส่วนประกอบจากไลบรารีมาใช้งานและไม่มีการถ่ายทอดคุณสมบัติการทำงาน ภายในโครงงานที่ทำการวัด และในกรณีค่า RF เท่ากับ 100% แสดงว่าทุกๆ คลาสภายในโครงงานที่ทำการวัดมีการนำส่วนประกอบจากไลบรารีมาใช้ หรือทุกๆ คลาสภายในโครงงานที่ทำการวัดมีการถ่ายทอดคุณสมบัติ สามารถแสดงสูตรการคำนวณได้ดังนี้

$$RF = \frac{\sum_{i=1}^{TC} in\_library(C_i)}{TC} + \frac{MIF * \sum_{i=1}^{TC}\left[1 - in\_library(C_i)\right]}{TC} \quad \cdots\cdots\cdots(10)$$

โดยที่ : $in\_library(C_i) = \begin{cases} 1 & iff \quad C_i \in L \\ 0 & otherwise \end{cases}$

L คือ เซ็ตของคลาสที่อยู่ในไลบรารีของคลาสที่นำกลับมาใช้ใหม่

TC คือ จำนวนคลาสทั้งหมดของโครงงานที่พิจารณา

MIF คือ ผลรวมของวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติของจำนวนคลาสทั้งหมดของโครงงานที่พิจารณาซึ่งค่าที่หาได้มาจากสมการที่ (4)

$\sum_{i=1}^{TC} in\_library$ คือ ผลรวมของคลาสทั้งหมดที่อยู่ในไลบรารี่ของคลาสที่นำกลับมาใหม่

2) Relative Degree of Code Saving[11] เป็นการวัดระดับความสัมพันธ์ของวิธีดำเนินการที่มีการนำกลับมาใช้ใหม่ ในลักษณะของการลดการเขียนรหัสโปรแกรมเนื่องจากมีการถ่ายทอดคุณสมบัติ ซึ่งในงานวิจัย “Inheritance Tree Shapes and Reuse” ของ Byung-Kyoo Kang และ James M. Bieman ได้ระบุว่าถ้าค่า RDCS ที่ได้จากการวัดมีค่าเท่ากับ 0% ต่อโครงงานที่ทำการวัด แสดงว่าโครงงานที่ทำการวัดนั้นไม่มีการถ่ายทอดคุณสมบัติเกิดขึ้นภายในโครงงาน และในกรณีค่า RDCS ที่ทำการวัดมีค่าเข้าใกล้ 100% มากๆ แสดงว่าโครงงานที่ทำการวัดนั้นมีระดับความสัมพันธ์ของวิธีดำเนินการภายในโครงงานสูง ส่งผลให้สามารถลดการเขียนรหัสโปรแกรมได้มาก เนื่องจากมีการถ่ายทอดคุณสมบัติสูงภายในโครงงาน และในกรณีค่า RDCS มีค่ามากกว่า 100% แสดงว่าจำนวนของวิธีดำเนินการที่มาจากการถ่ายทอดคุณสมบัติ มีจำนวนมากกว่าวิธีดำเนินการภายในโครงงานที่ทำการวัด ซึ่งจะทำให้มีการนำกลับมาใช้ใหม่ของโครงงานสูงขึ้นด้วย สามารถแสดงสูตรการคำนวณได้ดังนี้

$$RDCS = \frac{CS}{T_M} \quad \cdots\cdots\cdots(11)$$

$$CS = Cap_T(t) - T_M \quad \cdots\cdots\cdots(12)$$

$$Cap_T(t) = \sum_{i=1}^{n} Cap(C_i) \quad \cdots\cdots\cdots(13)$$

$$Cap(C) = the\ number\ of\ methods\ in\ C + the\ number\ of\ all\ inherited\ methods \quad \cdots\cdots\cdots(14)$$

$Cap(C)$ คือ จำนวนวิธีดำเนินการที่มีการเรียกใช้งานทั้งหมด บวกด้วยจำนวนวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติทั้งหมด ของคลาส C ที่พิจารณา

$Cap_T(t)$ คือ จำนวนวิธีดำเนินการที่มีการเรียกใช้งานทั้งหมด บวกด้วยจำนวนวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติทั้งหมด ของต้นไม้การถ่ายทอดคุณสมบัติ (t) ของโครงงานที่พิจารณา

T_M คือ ผลรวมของวิธีดำเนินการที่เปลี่ยนแปลงหลังจากมีการถ่ายทอดคุณสมบัติทั้งหมด

CS คือ ค่าของวิธีดำเนินการที่ได้จากการนำกลับมาใช้ใหม่

$RDCS$ คือ ระดับความสัมพันธ์ของวิธีดำเนินการจากการนำกลับมาใช้ใหม่

3) Reuse Size and Frequency [12] เป็นการวัดขนาดของรหัสโปรแกรมและความถี่ของรหัสโปรแกรมที่มีการนำกลับมาใช้ใหม่ ซึ่งในงานวิจัย “Analytical and Empirical Evaluation of Software Reuse Metrics” ของ Prem Devanbu, Sakke Karstu, Walcelio Melo และ William Thomas ได้ระบุว่าค่า R_{sf} ที่ได้จากการวัดจะต้องมีค่าอยู่ในช่วงระหว่าง 0-100% ต่อโครงงานที่ทำการวัด และค่าที่เหมาะสมของ R_{sf} จะต้องมีค่ามากกว่า 66% ขึ้นไป ในกรณี R_{sf} มีค่าเท่ากับ 0% หมายความว่าโครงงานที่ทำการวัดนั้นไม่มีการนำรหัสโปรแกรมกลับมาใช้ใหม่เลย และในกรณี R_{sf} มีค่าเท่ากับ 100% หมายความว่าโครงงานที่ทำการวัดมีการนำรหัสโปรแกรมกลับมาใช้ใหม่ทั้งโครงงานที่ทำการวัด ซึ่งอาจจะแสดงว่ามีความผิดพลาดเกิดขึ้นภายในโครงงานได้ สามารถแสดงสูตรการคำนวณได้ดังนี้

$$R_{sf}(S) = \frac{Size_{sf} - Size_{act}}{Size_{sf}} \qquad \cdots\cdots\cdots(15)$$

$$Size_{act}(S) = \sum_{i=1}^{TC} Size(C_i) \qquad \cdots\cdots\cdots(16)$$

$$Size_{sf}(S) = \sum_{i=1}^{TC} Size(C_i) * calls(C_i) \qquad \cdots\cdots\cdots(17)$$

$Size_{act}(S)$ คือ ขนาดของรหัสโปรแกรมที่ใช้จริงทั้งหมดของโครงงาน S ที่ พิจารณา

$Size_{sf}(S)$ คือ ขนาดของรหัสโปรแกรมที่ไม่มีการนำกลับมาใช้ใหม่ทั้งหมดของโครงงาน S ที่พิจารณา

$calls(C_i)$ คือ จำนวนครั้งที่มีการเรียกใช้งานของคลาสที่พิจารณา

$Size(C_i)$ คือ ขนาดของรหัสโปรแกรมของคลาสที่พิจารณา

$\sum_{i=1}^{TC} Size(C_i)$ คือ ผลรวมของขนาดรหัสโปรแกรมทั้งหมดของโครงงาน S ที่พิจารณา

$\sum_{i=1}^{TC} Size(C_i) * calls(C_i)$ คือ ผลรวมของขนาดรหัสโปรแกรมทั้งหมดคูณด้วยจำนวนครั้งที่มีการเรียกใช้งานของคลาสทั้งหมด ภายในโครงงาน S ที่พิจารณา

4) Reuse Ratio [12] เป็นการหาอัตราส่วนของการนำองค์ประกอบของโครงงานกลับมาใช้ใหม่ ซึ่งในงานวิจัย “Analytical and Empirical Evaluation of Software Reuse Metrics”ของ Prem Devanbu, Sakke Karstu, Walcelio Melo และ William Thomas ได้ระบุว่าค่า RR ที่เหมาะสมเท่ากับ 31-64% ต่อโครงงานที่ทำการวัด ซึ่งจะทำให้ผู้พัฒนาใช้เวลาในการพัฒนาไม่มาก เนื่องจากมีผลกระทบกับคลาสอื่นๆ น้อยและมีผลทำให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้นด้วย ในกรณีที่ RR มีค่าต่ำกว่า 31% หมายความว่าโครงงานที่ทำการวัด มีอัตราการเปลี่ยนแปลงขององค์ประกอบคือมีการเพิ่มเติม การปรับปรุง หรือมีการลบวิธีดำเนินการเนื่องจากมีการทำงานแบบเฉพาะเจาะจงภายในโครงงานสูง ซึ่งจะทำให้การนำกลับมาใช้ใหม่น้อยลงด้วย และค่าของ RR ไม่ควรเกินกว่า 64% เพราะค่า RR ที่สูงกว่าเกณฑ์มากๆ จะทำให้ไม่สามารถสรุปผลการวัดในส่วนของการนำกลับมาใช้ใหม่ ของการหาอัตราส่วนของการนำองค์ประกอบได้[12] และในกรณีที่ RR มีค่าเท่ากับ 100% หมายความว่า

องค์ประกอบภายในโครงงาน มีการเปลี่ยนแปลงเพียงเล็กน้อยเท่านั้น สามารถแสดงสูตรการคำนวณได้ดังนี้

$$RR(S) = \frac{\sum_{C_i \in S} IR(i) * Size(C_i)}{\sum_{C_i \in S} Size(C_i)} \qquad \cdots\cdots\cdots(18)$$

โดยที่ : $IR_i = \begin{cases} 1 & iff \quad Change_i < 0.25 \\ 0 & otherwise \end{cases}$

$Size\ (C_i)$ คือ ขนาดของรหัสโปรแกรมของคลาสที่พิจารณา

$Change_i$ คือ อัตราการเปลี่ยนแปลงขององค์ประกอบที่เกิดจากการเขียนเพิ่ม การปรับปรุง และการลบวิธีดำเนินการ

$\sum_{C_i \in S} Size\ (C_i)$ คือ ผลรวมของรหัสโปรแกรมของคลาสภายในโครงงาน S ที่พิจารณา

$\sum_{C_i \in S} IR_{(i)} * Size\ (C_i)$ คือ ผลรวมของอัตราการเปลี่ยนแปลงขององค์ประกอบ คูณด้วยขนาดของรหัสโปรแกรมของคลาส ภายในโครงงาน S ที่พิจารณา

และสามารถดูการตีความค่าที่ได้จากการวัดในภาคผนวก ข ซึ่งเป็นข้อเสนอแนะสำหรับค่าที่ได้จากการวัดโดยใช้มาตรวัดแบบต่างๆ ที่แสดงให้เห็นว่าซอฟต์แวร์หรือระบบงานที่ทำการวัดนั้นมีคุณภาพเหมาะสำหรับการนำกลับมาใช้ใหม่

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 Inheritance Tree Shapes and Reuse โดย Byung-Kyoo Kang และ James M. Bieman [11]

งานวิจัยนี้ Byung-Kyoo Kang และ James M. Bieman ได้ทำการศึกษาเกี่ยวกับความสัมพันธ์ของรูปร่างของต้นไม้การสืบทอดคุณสมบัติ (Inheritance tree) กับความสามารถในการรองรับการนำกลับมาใช้ใหม่ของการสืบทอดคุณสมบัติลักษณะต่างๆ และได้กำหนดมาตรวัดซึ่งเป็นการวัดระดับการนำกลับมาใช้ใหม่ที่เกิดขึ้นภายในของระบบที่พิจารณา โดยมีการกำหนด

ค่าที่ใช้ในการวัดได้แก่ Capacity ซึ่งเป็นค่าที่บอกถึงจำนวนวิธีดำเนินการจากคลาสแม่มีให้แก่คลาสลูกจากการสืบทอดคุณสมบัติค่า Capacity นี้สามารถใช้เป็นตัวบอกระดับการนำกลับมาใช้ใหม่ของการสืบทอดคุณสมบัติในระดับระบบงาน (System Level) ได้ ส่วนการวัดระดับปริมาณการนำกลับมาใช้ใหม่จะวัดได้เป็นค่า Code Saving และระดับความสัมพันธ์ของวิธีดำเนินการจากการนำกลับมาใช้ใหม่ (Relative Degree of Code Saving) ซึ่งเป็นปริมาณที่วัดจากจำนวนหน้าที่การทำงาน

จากงานวิจัยนี้จึงได้นำมาตรวัดในส่วนของ Relative Degree of Code Saving ซึ่งเป็นการวัดจำนวนของวิธีดำเนินการที่มีการนำกลับมาใช้ใหม่ ในลักษณะของการถ่ายทอดคุณสมบัติมาใช้ในการออกแบบ และพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวาด้วย

2.2.2 Object-Oriented Software Engineering : Measuring and Controlling the Development Process โดย Fernando Brito e Abreu และ Rogerio Carapuca [1]

งานวิจัยนี้ Fernando Brito e Abreu และ Rogerio Carapuca ทำการหาวิธีการวัดที่เหมาะสมเพื่อทำการวัดโครงงานที่ออกแบบโดยวิธีการออกแบบเชิงวัตถุ ไม่ว่าจะเป็นการสืบทอดคุณสมบัติ การห่อหุ้ม และการพ้องรูป

ซึ่งได้กำหนดเกณฑ์ที่ใช้ในการเลือกมาตรวัดดังต่อไปนี้

1) มาตรวัดที่นำมาใช้ควรจะต้องมีการกำหนดแบบแผนในการวัดไว้แล้ว
2) ขนาดของมาตรวัดที่นำมาใช้ควรจะเป็นอิสระจากโครงงานที่ทำการวัด
3) มาตรวัดที่นำมาใช้สามารถแสดงให้เห็นวิธีการวัดได้อย่างชัดเจนในส่วนของระบบย่อย
4) มาตรวัดที่นำมาใช้ควรจะสามารถนำมาใช้วัดได้ในส่วนต้นๆ ของวงจรชีวิต (Life-Cycle) ของการพัฒนาโครงงาน คือสามารถทำการวัดค่าได้ตั้งแต่ขั้นตอนการวิเคราะห์และออกแบบซอฟต์แวร์ แทนที่จะทำการวัดหลังจากที่มีการเขียนโค้ดเสร็จแล้ว
5) มาตรวัดที่นำมาใช้ควรจะสามารถลดงานที่มีลักษณะการทำงานแบบวนซ้ำ (Repetitive)
6) มาตรวัดที่นำมาใช้ควรจะมีวิธีการที่ง่ายในการคำนวณ
7) มาตรวัดที่นำมาใช้ในการวัดนั้น ไม่ควรขึ้นอยู่กับภาษาใดภาษาหนึ่งจะต้องสามารถนำไปใช้วัดได้กับโครงงานที่พัฒนาขึ้นจากภาษาใดๆ ก็ได้

ซึ่งจากเกณฑ์ที่ใช้ในการเลือกมาตรวัด Fernando Brito e Abreu และ Rogerio Carapuca ได้นำกฎเกณฑ์เหล่านั้นมาหาวิธีที่เหมาะสมสำหรับใช้ในการวัดการออกแบบเชิงวัตถุ

โดยมาตรวัดที่เรียกว่า MOOD ซึ่งเป็นชุดของมาตรวัดที่สามารถหาค่าวัดในส่วนของการวัดการถ่ายทอดคุณสมบัติซึ่งได้แก่ MIF และ AIF การวัดการห่อหุ้มและการซ่อนข้อมูลซึ่งได้แก่ MHF และ AHF การวัดการพ้องรูปซึ่งได้แก่ PF การวัดการเข้าคู่กันซึ่งได้แก่ CF และการวัดการนำกลับมาใช้ใหม่ซึ่งได้แก่ RF ถูกนำมาใช้ในการวิจัยในครั้งนี้เพราะ MOOD เป็นมาตรวัดที่มีลักษณะตรงกับเกณฑ์ข้อ 1 ถึง 7 ที่ได้วางไว้

Fernando Brito e Abreu และ Rogerio Carapuca ได้นำ MOOD มาทำการวัดระบบงานที่พัฒนาโดยใช้ C++ และจะใช้ MOOD เพื่อทำการวัดระบบงานที่พัฒนาโดยใช้โปรแกรมภาษา Eiffel ในอนาคตอีกด้วยเพราะ MOOD (ยกเว้น Reuse Factor) สามารถนำมาใช้กับระบบเชิงวัตถุทั่วไปได้ในการวัดความซับซ้อน และหาค่าเพื่อทำการวิเคราะห์สิ่งที่ขาดหายไปจากความซับซ้อนสำหรับระบบงานเหล่านั้น

จากงานวิจัยนี้จึงได้นำหลักเกณฑ์ที่ใช้ในการเลือกมาตรวัดพร้อมทั้งนำ MOOD มาใช้ในการออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวานี้ด้วย

2.2.3 Analytical and Empirical Evaluation of Software Reuse Metrics โดย Prem Devanbu, Sakke Karstu, Walcelio Melo และ William Thomas [12]

งานวิจัยนี้ Prem Devanbu, Sakke Karstu, Walcelio Melo และ William Thomas ได้ทำการวิจัยในส่วนของผลตอบแทนทางอ้อมที่จะได้รับซึ่งได้แก่ เวลา เงิน และคุณภาพที่ได้รับจากการนำองค์ประกอบของซอฟต์แวร์กลับมาใช้ใหม่จากโปรแกรมที่พัฒนามาจาก C++ โดยจะใช้มาตร-วัด 5 มาตรวัด ซึ่งได้แก่ RSI (Reuse Source Instruction) เป็นการหาอัตราส่วนของจำนวนบรรทัดของคำสั่งที่มีการนำกลับมาใช้ใหม่ Reuse Level Mode เป็นมาตรวัดที่ใช้ค่า RL (Reuse Level) ในการหาระดับของการนำกลับมาใช้ใหม่ และค่า RF (Reuse Frequency) ในการหาความถี่ของการนำองค์ประกอบกลับมาใช้ใหม่ RR (Reuse Ratio) เป็นการหาอัตราส่วนของการนำองค์ประกอบของระบบงานกลับมาใช้ใหม่ และ R_{sf} (Reuse Size and Frequency) เป็นการวัดขนาดของรหัสโปรแกรมและความถี่ของรหัสโปรแกรมที่มีการนำกลับมาใช้ใหม่ ผลลัพธ์ที่ได้จะเป็นความสัมพันธ์ในรูปของสมการทางคณิตศาสตร์ จากนั้นจึงทำการสรุปผลและทำการวิเคราะห์ผลทั้งหมดที่ได้จากการทดสอบ เพื่อประเมินว่าสามารถนำส่วนประกอบนั้นกลับมาใช้ใหม่ได้หรือไม่

จากงานวิจัยนี้จึงได้นำมาตรวัดในส่วนของ RR และ R_{sf} มาใช้ในการออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวาด้วย เพื่อให้การวัดการนำกลับมาใช้ใหม่มีความน่าเชื่อถือยิ่งขึ้น

2.2.4 การออกแบบ และพัฒนาเครื่องมือวัดซอฟต์แวร์สำหรับโปรแกรมเชิงวัตถุ โดย สมหวัง แซ่ตั้ง [13]

งานวิจัยนี้ สมหวัง แซ่ตั้ง ได้ทำการออกแบบและพัฒนาเครื่องมือ MTOOP (Metric Tool for Object-Oriented Programs) ซึ่งเป็นเครื่องมือที่พัฒนามาจากโปรแกรมภาษาจาวาบนระบบปฏิบัติการวินโดวส์ เพื่อนำมาใช้ในการวัดโปรแกรมเชิงวัตถุที่พัฒนาด้วยซอฟต์แวร์ภาษาจาวา โดยจะทำการอ่านโปรแกรมต้นฉบับแล้วแปลงเป็นต้นไม้ที่เรียกว่าเอเอสที (Abstract Syntax Tree : AST) จากนั้นจึงทำการคำนวณหาค่าวัดโดยใช้มาตรวัดที่ได้จากงานวิจัยของ McCabe ซึ่งได้ออกแบบค่าวัดความซับซ้อนภายในของแต่ละวิธีดำเนินการ และงานวิจัยของ Chidamber and Kemerer ซึ่งได้ออกแบบการหาค่าวัดจำนวนวิธีดำเนินการต่อคลาส ระดับความลึกของแผนภูมิแสดงการสืบทอดคุณสมบัติ จำนวนของคลาสลูก ขนาดความสัมพันธ์ระหว่างวัตถุ ระดับการตอบสนองต่อคลาส และระดับของการขาดความสัมพันธ์ภายในคลาส หลังจากนั้นนำค่าที่ได้จากการวัดมาแสดงพร้อมทำการบันทึกค่าที่ได้ลงฐานข้อมูล งานวิจัยนี้จะแสดงเพียงค่าต่างๆ ที่ได้จากการวัดเท่านั้น แต่ไม่ได้แสดงให้เห็นความสัมพันธ์ของค่าต่างๆ ที่ได้จากการวัดนอกจากนี้ยังไม่ได้กำหนดเกณฑ์เพื่อใช้ในการวิเคราะห์ จึงเป็นการยากที่ผู้ใช้จะทำการวิเคราะห์เพื่อนำค่าที่ได้ไปใช้ประโยชน์กับโครงงานใหม่ต่อไป

2.2.5 การออกแบบ และพัฒนาเครื่องมือวัดปัจจัยของความซับซ้อนของโปรแกรมเชิงวัตถุภาษาจาวา โดย วัฒนชัย รอดกำเนิด [14]

งานวิจัยนี้ วัฒนชัย รอดกำเนิด ได้ทำการพัฒนาเครื่องมือ MTOOP รุ่นที่ 2 ซึ่งเป็นการทำการวิจัยต่อจากงานวิจัย “การออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์สำหรับโปรแกรมเชิงวัตถุ” ของ สมหวัง แซ่ตั้ง โดยได้ทำการเพิ่มการหาค่าวัดปัจจัยความซับซ้อนซึ่งได้แก่ ขนาดของคลาสที่ประเมินจากจำนวนของวิธีดำเนินการและจากค่าถ่วงน้ำหนักของตัวแปรอินสแตนซ์ ขนาดของวิธีดำเนินการ ความซับซ้อนของแต่ละวิธีดำเนินการ ค่าวัดที่ได้จากการพ้องรูป และค่าความรับผิดชอบของคลาส ผู้พัฒนาสามารถใช้เครื่องมือนี้ประเมินแนวโน้มของค่าความซับซ้อนของโปรแกรมภาษาจาวาจากค่าปัจจัยต่างๆ ที่วัดได้ ซึ่งจะช่วยให้ผู้พัฒนาสามารถลดความซับซ้อนของโปรแกรมได้ตรงสาเหตุมากขึ้นตามปัจจัยที่วัดได้

บทที่ 3

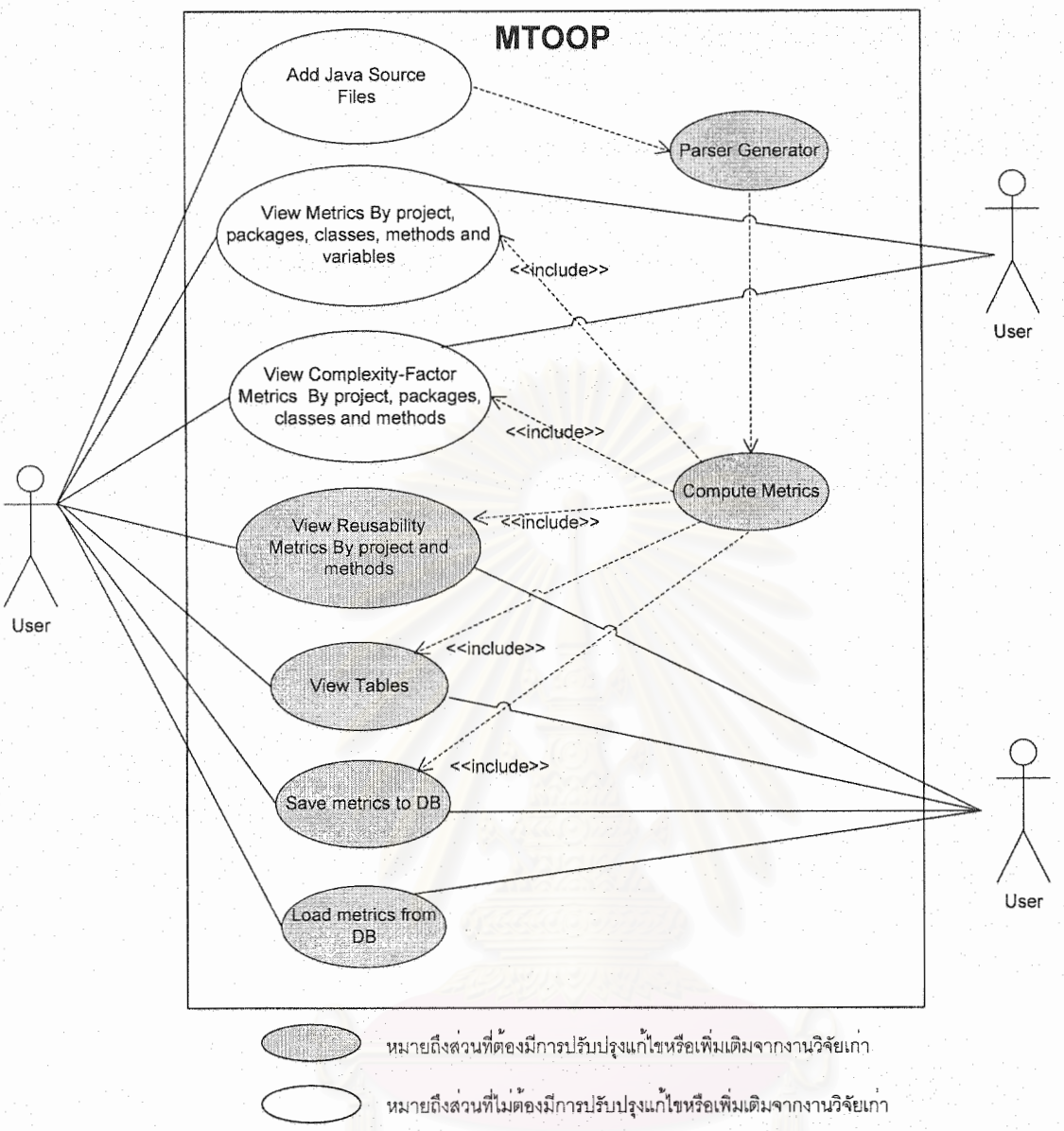
การวิเคราะห์และออกแบบเครื่องมือ MTOOP รุ่นที่ 3

ผู้วิจัยแสดงการวิเคราะห์และออกแบบเครื่องมือ MTOOP รุ่นที่ 3 ในการหาค่าวัดสำหรับการนำกลับมาใช้ใหม่ ซึ่งเป็นการทำวิจัยต่อจากงานวิจัย “การออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์สำหรับโปรแกรมเชิงวัตถุ” ของ สมหวัง แซ่ตั้ง และงานวิจัย “การออกแบบและพัฒนาเครื่องมือวัดปัจจัยของความซับซ้อนของโปรแกรมเชิงวัตถุภาษาจาวา” ของ วัฒนชัย รอดกำเนิด สาเหตุที่ผู้วิจัยนำเครื่องมือ MTOOP มาทำการพัฒนาต่อเนื่องจากเครื่องมือ MTOOP เป็นเครื่องมือที่ถูกพัฒนาขึ้นเพื่อใช้ในการหาค่าวัดสำหรับการวัดโปรแกรมเชิงวัตถุภาษาจาวา และมีการพัฒนาเพิ่มในส่วนของการวัดความซับซ้อน ซึ่งมีความสัมพันธ์กับการหาค่าวัดสำหรับวัดการนำกลับมาใช้ใหม่ ซึ่งผู้วิจัยได้ทำการวิจัยในเรื่องดังกล่าวจึงทำให้ผู้วิจัยเลือกเครื่องมือ MTOOP มาใช้ในการวิเคราะห์และออกแบบเครื่องมือวัดสำหรับการนำกลับมาใช้ใหม่

ผู้วิจัยได้ทำการแก้ไขเพิ่มเติมเครื่องมือ MTOOP รุ่นที่ 3 นี้ในส่วนของการสร้างโหนดของต้นไม้เอเอสทีเพื่อนำมาหาค่าวัดสำหรับการนำกลับมาใช้ใหม่ การคำนวณและการแสดงผลค่าที่ได้จากการวัดสำหรับการนำกลับมาใช้ใหม่ การบันทึกค่าที่ได้จากการวัดสำหรับการนำกลับมาใช้ใหม่ลงฐานข้อมูล และการเรียกข้อมูลค่าวัดสำหรับการนำกลับมาใช้ใหม่จากฐานข้อมูลมาแสดงผลบนเครื่องมือวัด โดยใช้ภาษายูเอ็มแอล (Unified Modeling Language – UML) ซึ่งเป็นภาษาที่ใช้อธิบายโมเดลต่างๆ ในการนำเสนอภาพรวมของระบบได้แก่การใช้ยูสเคสไดอะแกรม (Use Case Diagram) อธิบายแผนภาพแสดงการเกิดปฏิสัมพันธ์ระหว่างระบบงานกับสิ่งที่อยู่นอกระบบ การใช้คลาสไดอะแกรม (Class Diagram) อธิบายแผนภาพแสดงองค์ประกอบของระบบงาน การใช้ซีเควนซ์ไดอะแกรม (Sequence Diagram) อธิบายแผนภาพแสดงการเกิดปฏิสัมพันธ์ระหว่างออบเจ็กต์ในระบบงาน การใช้แอคทิวิตี้ไดอะแกรม (Activity Diagram) อธิบายแผนภาพแสดงกิจกรรมที่เกิดขึ้นในระบบงาน และการใช้สเตทชาร์ตไดอะแกรม (State Chart Diagram) อธิบายแผนภาพแสดงการเปลี่ยนแปลงสถานภาพของระบบงาน ดังรายละเอียดต่อไปนี้

3.1 แผนภาพยูสเคสแสดงการเกิดปฏิสัมพันธ์ระหว่างระบบงานกับสิ่งที่อยู่นอกระบบ

เป็นการพิจารณาถึงรูปแบบการใช้ระบบงานที่สามารถเกิดขึ้นได้ โดยจะอธิบายเป็นลำดับของเหตุการณ์ที่เกิดขึ้น และแสดงให้เห็นภาพว่าผู้ใช้จะนำระบบไปใช้อะไรบ้าง



รูปที่ 3.1 แผนภาพแสดงการเกิดปฏิสัมพันธ์ระหว่างระบบงานกับสิ่งที่อยู่นอกระบบงาน

จากการวิเคราะห์ต้นแบบของการเกิดปฏิสัมพันธ์ระหว่างระบบงานกับผู้ใช้งาน สามารถเขียนแผนภาพได้ดังรูปที่ 3.1 โดยส่วนที่แรเงาหมายถึงส่วนที่ต้องมีการปรับปรุงแก้ไขหรือเพิ่มเติมจากงานวิจัย "การออกแบบและพัฒนาเครื่องมือวัดปัจจัยของความซับซ้อนของโปรแกรมเชิงวัตถุภาษาจาวา" ของ วัฒนชัย รอดกำเนิด ซึ่งมีรายละเอียดดังนี้

ผู้ใช้งานสามารถใช้และติดต่อกับเครื่องมือนี้ได้ 7 วิธี ดังรูปที่ 3.1 คือ

1) ผู้ใช้งานสามารถกำหนดโปรแกรมต้นฉบับภาษาจาวาที่ต้องการหาค่าตัววัด
2) ผู้ใช้งานสามารถดูค่าวัดโดยแยกตามโครงงาน แพ็คเกจ คลาส วิธีดำเนินการ และตัวแปรได้

4) ผู้ใช้งานสามารถดูค่าวัดเพื่อประมาณการณ์ถึงความเหมาะสม ที่จะนำโปรแกรมต้นฉบับภาษาจาวากลับมาใช้ใหม่โดยแยกตามโครงงาน และวิธีดำเนินการ
5) ผู้ใช้งานสามารถดูค่าวัดในรูปแบบของตารางได้
6) ผู้ใช้งานสามารถเก็บบันทึกข้อมูลค่าที่ได้จากการวัดลงฐานข้อมูลได้
7) ผู้ใช้งานสามารถดูข้อมูลค่าตัววัดที่มีอยู่ในฐานข้อมูลได้

และสามารถอธิบายความสัมพันธ์ของยูสเคสที่เกิดขึ้นภายในระบบงานได้ดังต่อไปนี้

3.1.1 **ยูสเคส** Add Java Source Files

จากตารางที่ 3.1 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานต้องการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่เครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 3.1 แสดงขั้นตอนการทำงานของยูสเคส Add Java Source Filed

Use Case Name	Add Java Source Files
Entry condition	1. ผู้ใช้งานเลือกประเภทของการนำรหัสโปรแกรมเข้าสู่ระบบ ว่าต้องการนำรหัสโปรแกรมเพิ่มในโครงงานเดิมที่ทำการวัด หรือเริ่มต้นสำหรับทำการวัดโครงงานใหม่
Flow of events	2. ผู้ใช้งานเลือกรหัสโปรแกรมภาษาจาวาจากแหล่งที่เก็บ 3. เครื่องมือ MTOOP รุ่นที่ 3 นำรหัสโปรแกรมที่ผู้ใช้งานกำหนดเข้าสู่ระบบงาน
Exit condition	4. ระบบทำการเพิ่มรหัสโปรแกรมตามที่ผู้ใช้งานต้องการจนกว่าผู้ใช้งานจะทำการเลือกทำรายการเมนูอื่นๆ

3.1.2 **ยูสเคส** Parser Generator

จากตารางที่ 3.2 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อเครื่องมือ MTOOP เริ่มทำการสร้างโหนดของต้นไม้เอเอสที

ตารางที่ 3.2 แสดงขั้นตอนการทำงานของยูสเคส Parser Generator

Use Case Name	Parser Generator

Entry condition	1. รหัสโปรแกรมภาษาจาวาที่ต้องการวัด
Flow of events	2. ระบบนำรหัสโปรแกรมจากยูสเคส Add Java Source Files มาทำการสร้างโหนดของต้นไม้เอเอสที ที่เก็บข้อมูลหรือคุณสมบัติต่างๆ เช่น คลาส วิธีดำเนินการ เป็นต้น
Exit condition	3. รหัสโปรแกรมภาษาจาวาถูกตรวจสอบหมดแล้ว

3.1.3 **ยูสเคส** Compute Metrics

จากตารางที่ 3.3 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อเครื่องมือ MTOOP รุ่นที่ 3 เริ่มนำโหนดของต้นไม้เอเอสทีมาทำการคำนวณเพื่อหาค่าวัดสำหรับการนำกลับมาใช้ใหม่

ตารางที่ 3.3 แสดงขั้นตอนการทำงานของยูสเคส Compute Metrics

Use Case Name	Compute Metrics
Entry condition	1. โหนดของต้นไม้เอเอสที ที่เก็บข้อมูลหรือคุณสมบัติต่างๆ
Flow of events	2. ระบบนำโหนดของต้นไม้เอเอสทีจากยูสเคส Parser Generator มาทำการหาค่าสำหรับการนำกลับมาใช้ใหม่ ตามที่ได้กำหนดไว้ในเครื่องมือ MTOOP ซึ่งค่าวัดที่คำนวณได้มีดังนี้ 2.1 Cyclomatic Complexity (CC) 2.2 Lines Of Code (LOC) 2.3 Comment Percentage (CP) 2.4 Method Inheritance Factor (MIF) 2.5 Attribute Inheritance Factor (AIF) 2.6 Method Hiding Factor (MHF) 2.7 Attribute Hiding Factor (AHF) 2.8 Coupling Factor (COF) 2.9 Polymorphism Factor (PF) 2.10 Reuse Factor (RF) 2.11 Relative Degree of Code Saving (RDCS)

ตารางที่ 3.3 (ต่อ)แสดงขั้นตอนการทำงานของยูสเคส Compute Metrics

Flow of events	2.12 Reuse Size and Frequency (R_{sf})

	2.13 Reuse Ratio (RR)
Exit condition	3. โหนดของต้นไม้เอเอสที ถูกนำไปทำการหาค่าวัดครบทุกโหนด

3.1.4 ยูสเคส View Metrics By project, packages, classes, methods and variables

จากตารางที่ 3.4 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ View Metrics จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 3.4 แสดงขั้นตอนการทำงานของยูสเคส View Metrics By project, packages, classes, methods and variables

Use Case Name	View Metrics By project, packages, classes, methods and variables
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ View Metrics
Flow of events	2. ระบบนำค่าที่ได้จากยูสเคส Compute Metrics โดยแยกตามโครงงาน แพ็คเกจ คลาส วิธีดำเนินการ และตัวแปรมาแสดงผล
Exit condition	3. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

3.1.5 ยูสเคส View Complexity-Factor Metrics By project, packages, classes and methods

จากตารางที่ 3.5 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ View Complexity-Factor จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 3.5 แสดงขั้นตอนการทำงานของยูสเคส View Complexity-Factor Metrics By project, packages, classes and methods

Use Case Name	View Complexity-Factor Metrics By project, packages,

	classes and methods
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ View Complexity-Factor
Flow of events	2. ระบบนำค่าปัจจัยของความซับซ้อนที่ได้จากยูสเคส Compute Metrics โดยแยกตาม โครงงาน แพ็คเกจ คลาส และวิธีดำเนินการมาแสดงผล
Exit condition	3. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

3.1.6 **ยูสเคส** View Reusability Metrics By project and methods

จากตารางที่ 3.6 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ View Reusability จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 3.6 แสดงขั้นตอนการทำงานของยูสเคส View Reusability Metrics By project and methods

Use Case Name	View Reusability Metrics By project and methods
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ View Reusability
Flow of events	2. ระบบนำค่าวัดจากยูสเคส Compute Metrics มาแสดงผล และทำการประเมินค่าที่ได้จากการวัดในส่วนของการนำโปรแกรมต้นฉบับภาษาจาวากลับมาใช้ใหม่ โดยแยกตามโครงงาน และวิธีดำเนินการ
Exit condition	3. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

3.1.7 **ยูสเคส** View Tables

จากตารางที่ 3.7 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ View Tables จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 3.7 แสดงขั้นตอนการทำงานของยูสเคส View Tables

Use Case Name	View Tables
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ View Tables

Flow of events	2. ระบบนำค่าวัดที่คำนวณได้จากยูสเคส Compute Metrics มาแสดงผลในรูปแบบของตาราง
Exit condition	3. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

3.1.8 **ยูสเคส** Save metrics to DB

จากตารางที่ 3.8 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ Save DB จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 3.8 แสดงขั้นตอนการทำงานของยูสเคส Save metrics to DB

Use Case Name	Save metrics to DB
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ Save DB
Flow of events	2. ผู้ใช้งานตั้งชื่อโครงงานเพื่อจัดเก็บลงฐานข้อมูล 3. เมื่อระบบสามารถติดต่อกับฐานข้อมูลได้แล้วระบบจะทำการจัดเก็บค่าที่ได้จากการวัดจากยูสเคส Compute Metrics ลงฐานข้อมูล
Exit condition	4. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

3.1.9 **ยูสเคส** Load metrics from DB

จากตารางที่ 3.9 เป็นยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ Load DB จากเครื่องมือ MTOOP รุ่นที่ 3

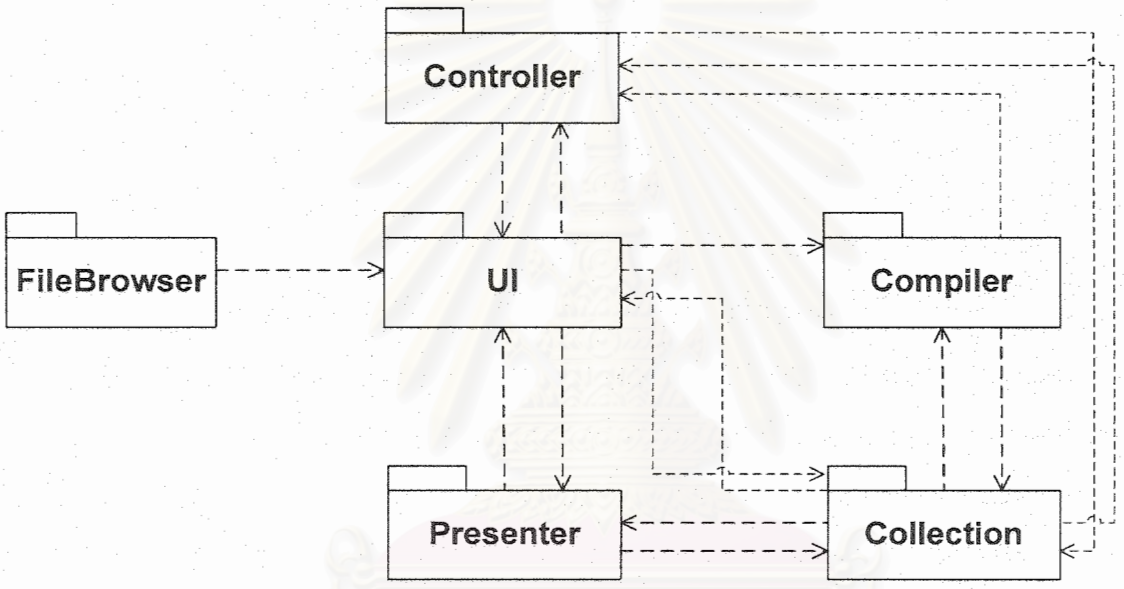
ตารางที่ 3.9 แสดงขั้นตอนการทำงานของยูสเคส Load metrics from DB

Use Case Name	Load metrics from DB
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ Load DB
Flow of events	2. ผู้ใช้งานเลือกชื่อโครงงานที่ต้องการนำรายละเอียดมาแสดงผลจากฐานข้อมูล 3. ระบบนำค่าที่ได้มาแสดงผล
Exit condition	4. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

3.2 **แผนภาพแพ็คเกจแสดงองค์ประกอบของระบบงาน**

3.2 แผนภาพแพ็คเกจแสดงองค์ประกอบของระบบงาน

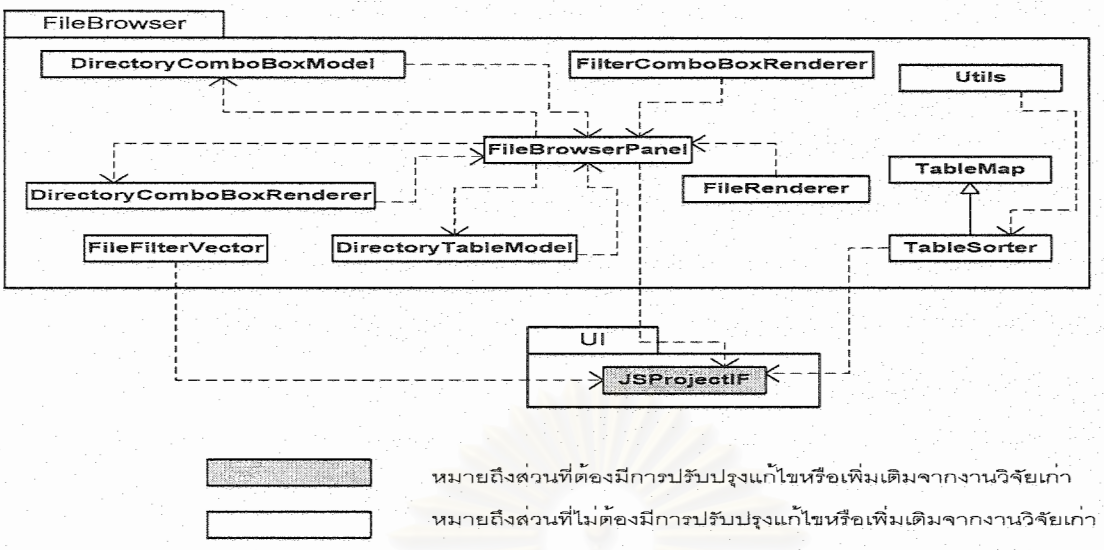
แผนภาพแพ็คเกจแสดงองค์ประกอบของระบบงานสามารถแบ่งออกเป็น 6 แพ็คเกจหลัก คือ แพ็คเกจส่วนของการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่ระบบ (FileBrowser) แพ็คเกจส่วนของการควบคุม (Controller) แพ็คเกจตัวแปลภาษา (Compiler) แพ็คเกจการคำนวณค่าตัววัด (Collection) แพ็คเกจส่วนของการติดต่อผู้ใช้งาน (UI) และแพ็คเกจการจัดรูปแบบการแสดงผลค่าตัววัด (Presenter) ซึ่งสามารถแสดงความสัมพันธ์ระหว่างแพ็คเกจต่างๆ ของระบบได้ดังรูปที่ 3.2



รูปที่ 3.2 แผนภาพของแพ็คเกจแสดงองค์ประกอบของระบบงาน

3.2.1 แพ็คเกจส่วนของการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่ระบบ

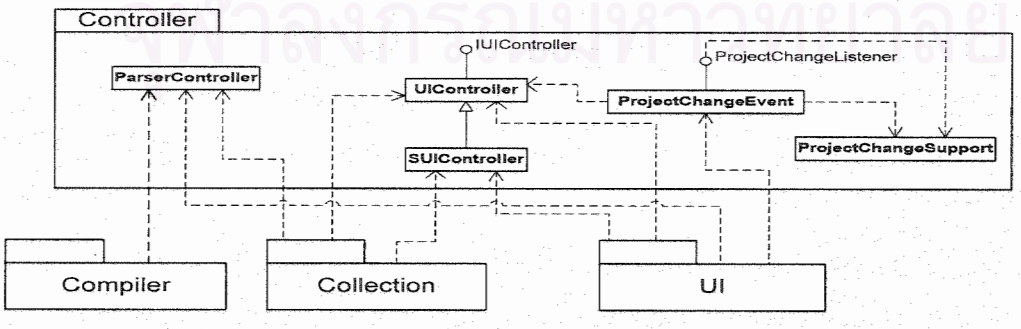
เป็นชุดของคลาสที่ทำหน้าที่ในการระบุตำแหน่งของโปรแกรมต้นฉบับ เพื่อนำมาหาค่าตัววัดซึ่งในแพ็คเกจนี้จะประกอบด้วยคลาสต่างๆ โดยที่แต่ละคลาสจะทำหน้าที่ประสานงานกัน เพื่อให้ระบบสามารถนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่ระบบงาน และทำการจัดเก็บโปรแกรมต้นฉบับลงในหน่วยความจำสำรองเพื่อใช้ในการหาค่าตัววัดต่อไป สามารถแสดงได้ดังรูปที่ 3.3



รูปที่ 3.3 แผนภาพคลาสในแพ็คเกจส่วนของการอ่านโปรแกรมต้นฉบับเข้าสู่ระบบ

3.2.2 แพ็คเกจส่วนของการควบคุม

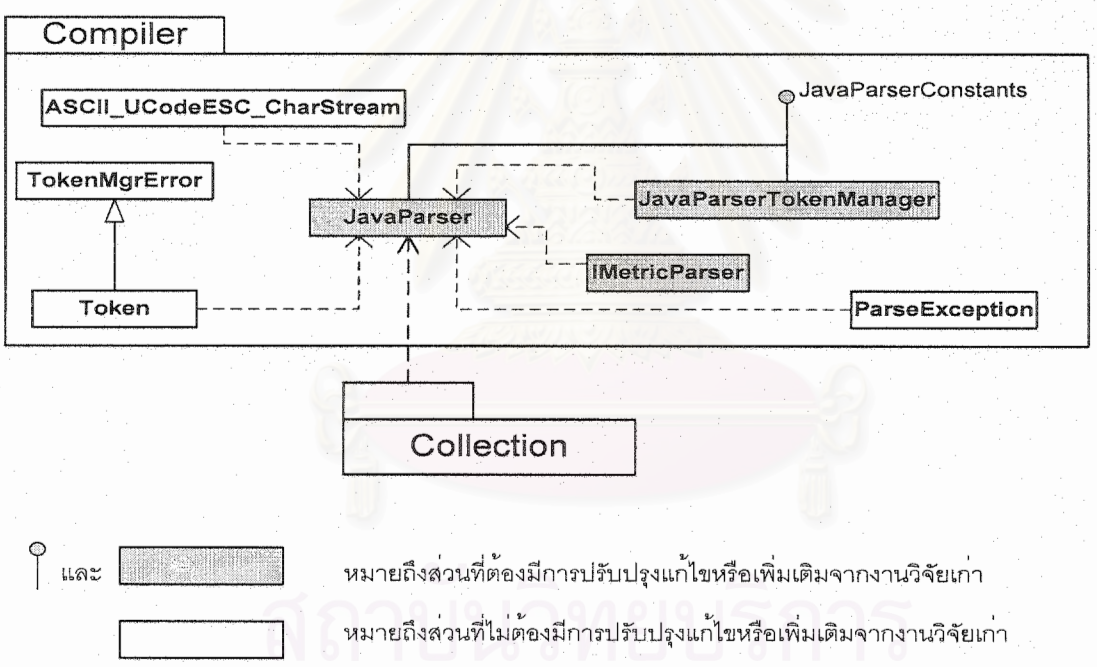
เป็นชุดของคลาสที่ทำหน้าที่ควบคุมการทำงานให้กับ แพ็คเกจส่วนของการติดต่อผู้ใช้งาน และแพ็คเกจส่วนของการคำนวณค่าตัววัด โดยทำการอินเตอร์เฟสผ่านทางคลาส IUIController และ ProjectChangeListener โดยในแพ็คเกจส่วนของการควบคุมนี้จะประกอบไปด้วย คลาสสำหรับทำการควบคุมในส่วนของการจัดการพาร์เซอร์ (ParserControl) การจัดการในการอ่านโปรแกรมต้นฉบับ (ProjectChangeEvent) การจัดการในส่วนของการเพิ่มโปรแกรมต้นฉบับ (ProjectChangeSupport) และการจัดการในส่วนของการติดต่อกับผู้ใช้งาน (SUIController) ซึ่งคลาสนี้จะถูกถ่ายทอดคุณสมบัติมาจากคลาสส่วนของการควบคุมการติดต่อกับผู้ใช้งาน (UIController) สามารถแสดงได้ดังรูปที่ 3.4



รูปที่ 3.4 แผนภาพคลาสในแพ็คเกจส่วนของการควบคุม

3.2.3 แพ็คเกจตัวแปลภาษา

เป็นชุดของคลาสที่ทำหน้าที่ในการนำโปรแกรมต้นฉบับ มาผ่านขบวนการสร้างตัวแปลภาษาเพื่อสร้างซินแท็กซ์ทรี และนำโครงสร้างที่ได้ไปทำการวิเคราะห์เพื่อหาค่าตัววัด โดยทำการอินเตอร์เฟสผ่านทางคลาส JavaParserConstants โดยในแพ็คเกจส่วนของตัวแปลภาษานี้จะประกอบไปด้วย คลาสสำหรับทำการสร้างสายอักขระของข้อมูล (Token) การตรวจสอบรหัสแอสกีของข้อมูล (ASCII_UCodeESC_CharStream) การแสดงข้อความในกรณีแปลความหมายผิดพลาด (ParseException) การตรวจสอบการกระจายคำ (Token) และการจัดการพาร์เซอร์ (JavaParserTokenManager) ซึ่งคลาสเหล่านี้จะถูกเรียกใช้โดยคลาสส่วนของการสร้างตัวพาร์เซอร์ (JavaParser) สามารถแสดงได้ดังรูปที่ 3.5

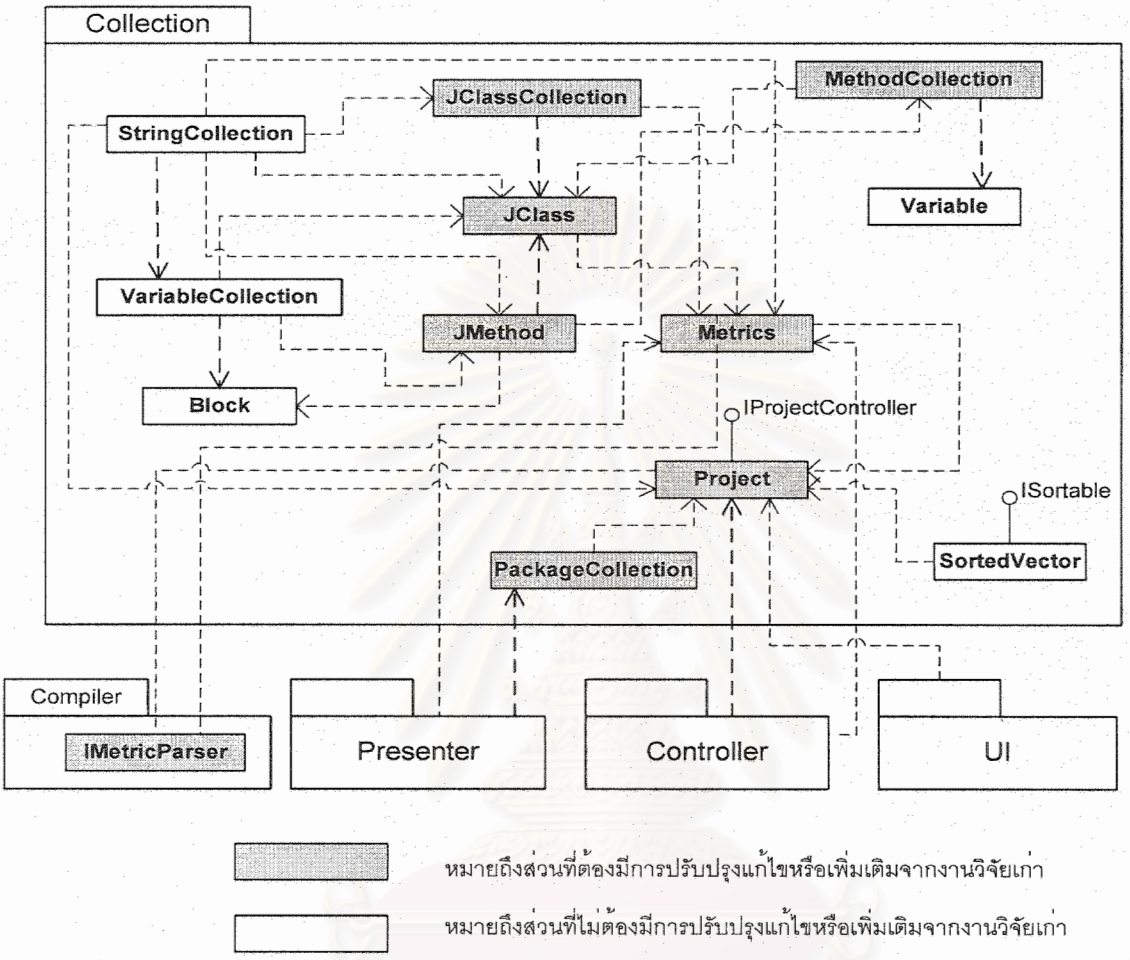


รูปที่ 3.5 แผนภาพคลาสในแพ็คเกจตัวแปลภาษา

3.2.4 แพ็คเกจการคำนวณค่าตัววัด

เป็นชุดของคลาสหน่วยรู้จำ โดยจะมีคลาสตัววัดทำหน้าที่ท่องไปบนซินแท็กซ์ทรี เพื่อเก็บข้อมูลจากโหนดต่างๆ และทำการคำนวณหาค่าตัววัดจากการท่องไปบนโหนดเหล่านั้น โดยทำการอินเตอร์เฟสผ่านทางคลาส IMetricParser และในแพ็คเกจส่วนของ

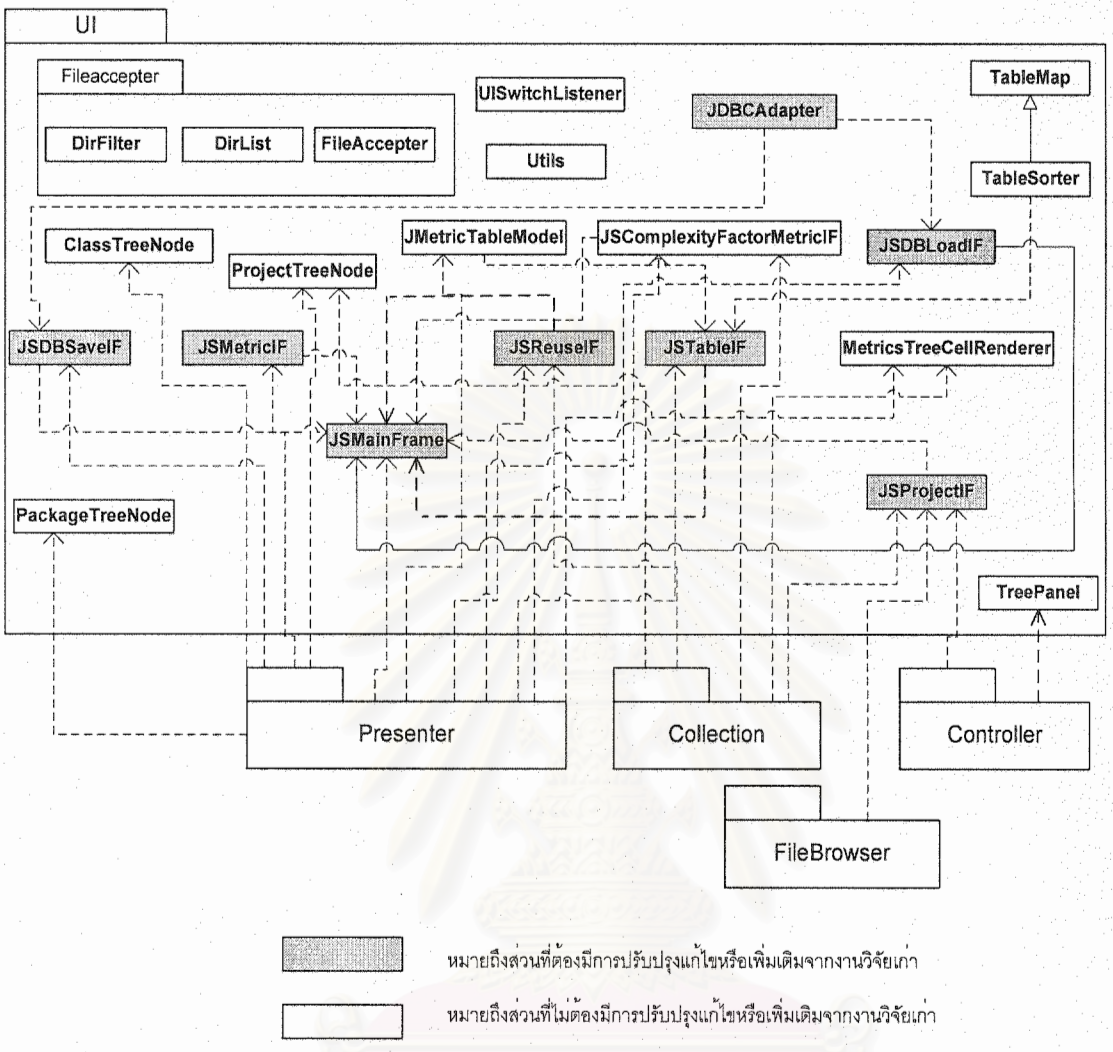
การคำนวณค่าตัววัดนี้จะประกอบไปด้วยคลาสต่างๆ ที่ทำหน้าที่คำนวณค่าวัดในระดับโครงงาน แพ็คเกจ คลาส วิธีดำเนินงาน และตัวแปรสามารถแสดงได้ดังรูปที่ 3.6



รูปที่ 3.6 แผนภาพคลาสในแพ็คเกจการคำนวณค่าตัววัด

3.2.5 แพ็คเกจส่วนของการติดต่อผู้ใช้งาน

เป็นชุดของคลาสที่ทำหน้าที่ติดต่อกับผู้ใช้งาน ในการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่ระบบเพื่อทำการหาค่าตัววัด การบันทึกผลจากการวัดลงฐานข้อมูลและการเรียกค่าตัววัดที่จัดเก็บในฐานข้อมูลมาแสดงโดยมีการทำงานผ่านทางคลาส JSDBSaveIF และ JSDBLoadIF ตามลำดับ และในการนำผลที่ได้จากการวัดมาแสดงในรูปแบบต่างๆ ตามที่ผู้ใช้งานกำหนดโดยมีการทำงานผ่านทางคลาส JSComplexityFactorMetricIF คลาส JSProjectIF คลาส JSMetricIF คลาส JSTableIF และ คลาส JSReuseIF สามารถแสดงได้ดังรูปที่ 3.7

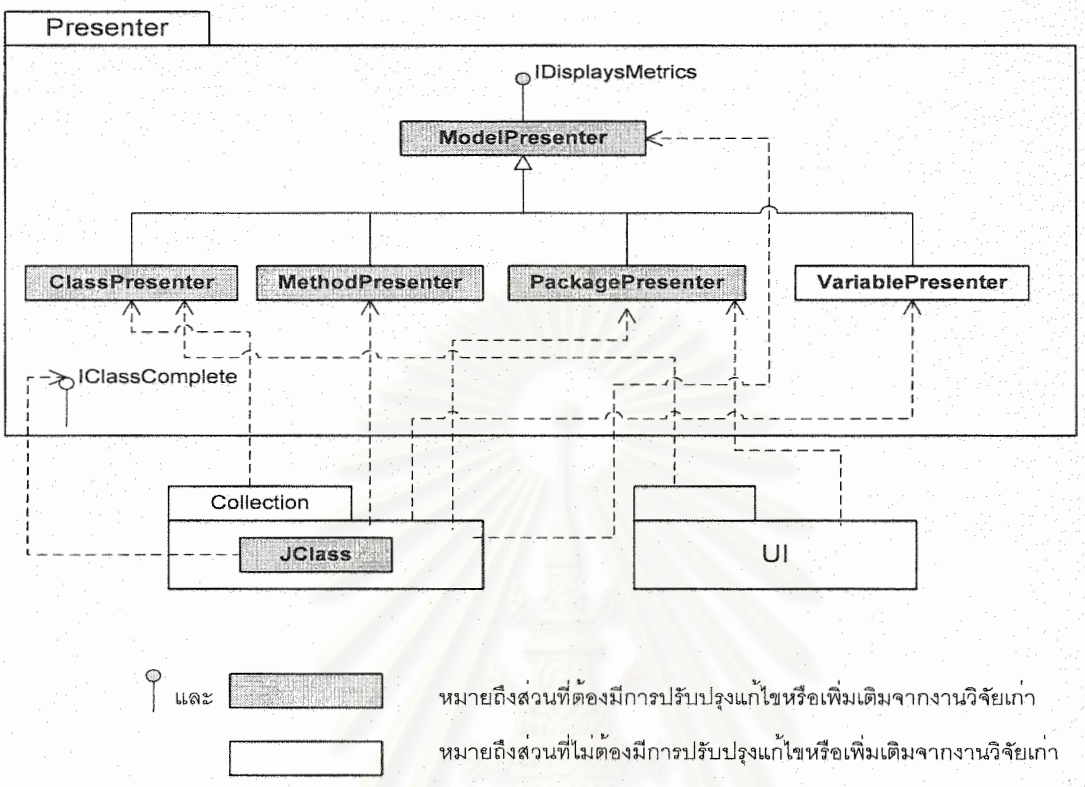


รูปที่ 3.7 แผนภาพคลาสในแพ็คเกจส่วนของการติดต่อผู้ใช้งาน

3.2.6 แพ็คเกจการจัดรูปแบบการแสดงผลค่าตัววัด

เป็นชุดของคลาสที่ทำหน้าที่นำค่าตัววัดจากแพ็คเกจการคำนวณค่าตัววัด มาจัดรูปแบบตามค่าตัววัดที่ต้องการ เพื่อส่งต่อให้แพ็คเกจส่วนของการติดต่อผู้ใช้งานนำไปแสดงผลค่าตัววัดตามที่ได้จัดรูปแบบไว้ โดยทำการอินเตอร์เฟสผ่านทางคลาส IDisplaysMetrics ในแพ็คเกจส่วนของการจัดรูปแบบการแสดงผลค่าตัววัดนี้จะประกอบไปด้วย คลาสสำหรับทำการจัดรูปแบบการแสดงผลค่าตัววัดในระดับแพ็คเกจ (PackagePresenter) ในระดับคลาส (ClassPresenter) ในระดับวิธีดำเนินการ (MethodPresenter) และในระดับตัวแปร (VariablePresenter) ซึ่งคลาสเหล่านี้จะถูก

ถ่ายทอดคุณสมบัติมาจากคลาสส่วนของการควบคุมโมเดล (ModelPresenter) สามารถแสดงได้ดังรูปที่ 3.8



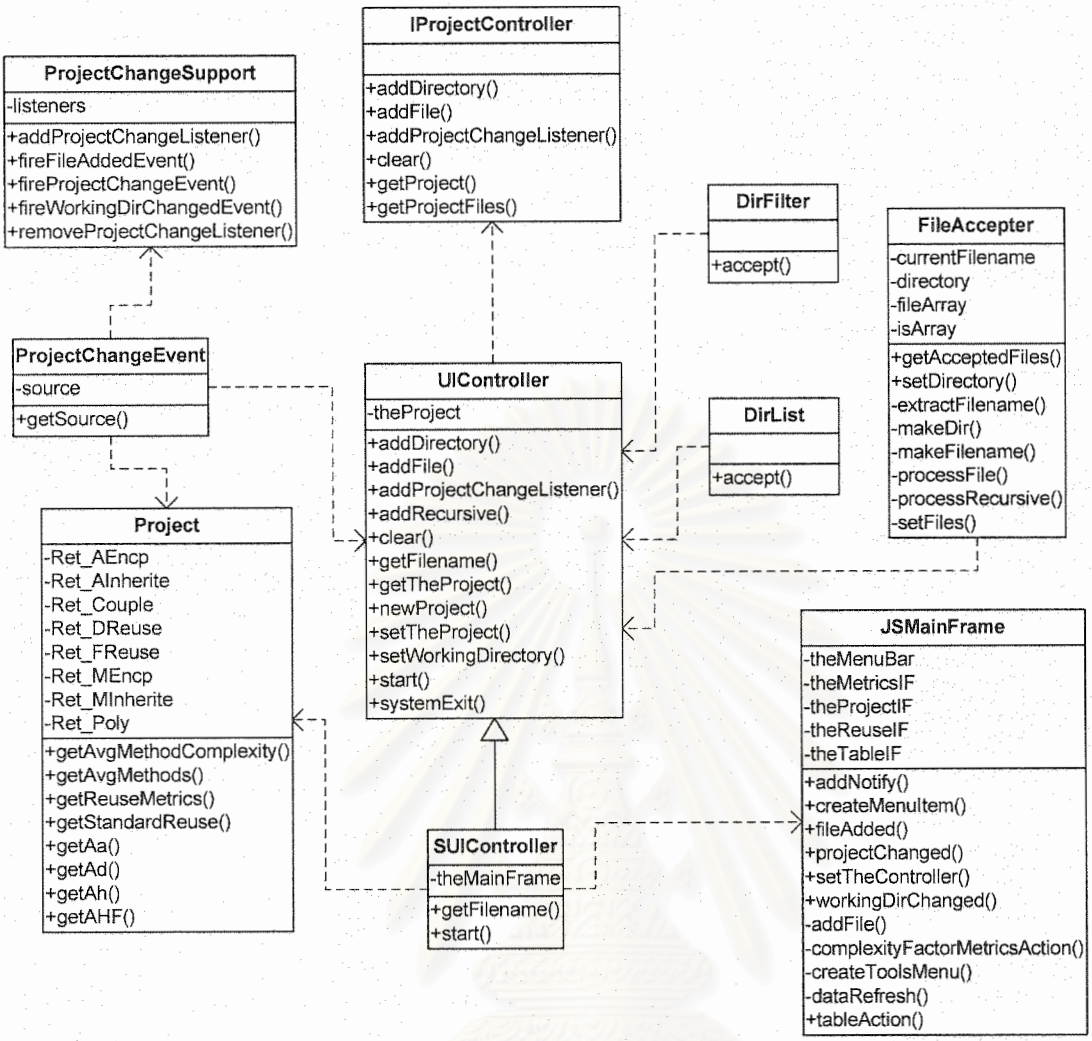
รูปที่ 3.8 แผนภาพแสดงแพ็คเกจจัดรูปแบบการแสดงผลค่าตัววัด

3.3 แผนภาพคลาสไดอะแกรมแสดงความสัมพันธ์ขององค์ประกอบภายในระบบงาน

สามารถนำคลาสไดอะแกรม มาอธิบายความสัมพันธ์ขององค์ประกอบในการสร้างเครื่องมือ MTOOP รุ่นที่ 3 ได้ดังนี้

3.3.1 แผนภาพคลาสแสดงหน้าจอการติดต่อกับเครื่องมือ MTOOP รุ่นที่ 3

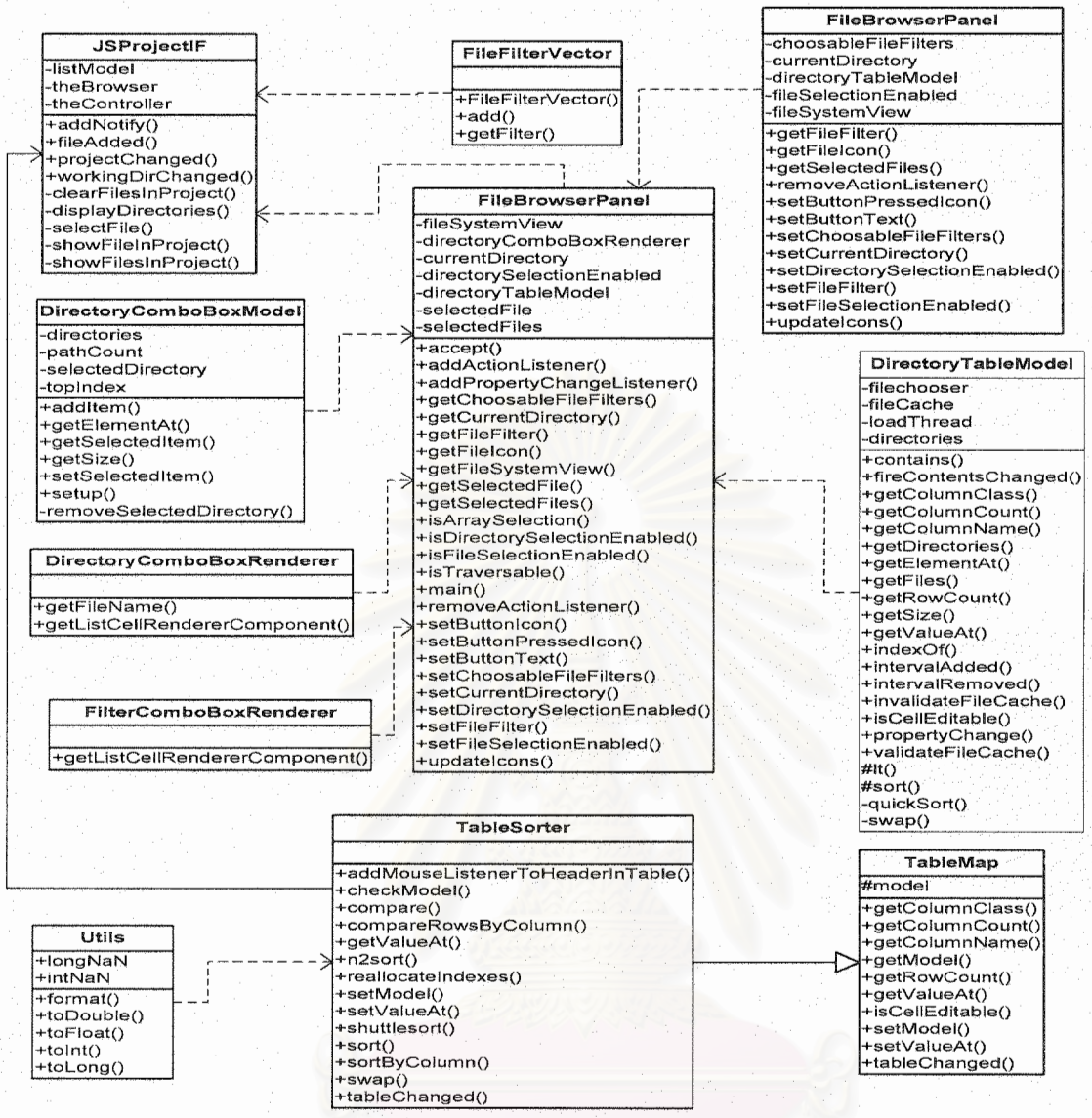
เป็นการแสดงชุดของคลาสที่มีความสัมพันธ์ ในการแสดงหน้าจอการติดต่อกับเครื่องมือ MTOOP รุ่นที่ 3 โดยเมื่อเริ่มต้นการทำงานคลาส UIController จะมีการเรียกการใช้งานไปยังคลาส ProjectChangeSupport ที่มีการเรียกใช้งานจากคลาส ProjectChangeEvent เพื่อทำหน้าที่ในการระบุโครงงานที่ต้องการนำเข้าสู่เครื่องมือ ส่วนคลาส DirFilter คลาส DirList และคลาส FileAccepter จะทำหน้าที่ในการแสดงแฟ้มข้อมูลที่ต้องการบนเครื่องมือ MTOOP รุ่นที่ 3 สามารถแสดงได้ดังรูปที่ 3.9



รูปที่ 3.9 แผนภาพคลาสแสดงหน้าจอการติดต่อกับเครื่องมือ MTOOP รุ่นที่ 3

3.3.2 แผนภาพคลาสแสดงการนำรหัสโปรแกรมภาษาจาวาเข้าสู่ระบบ

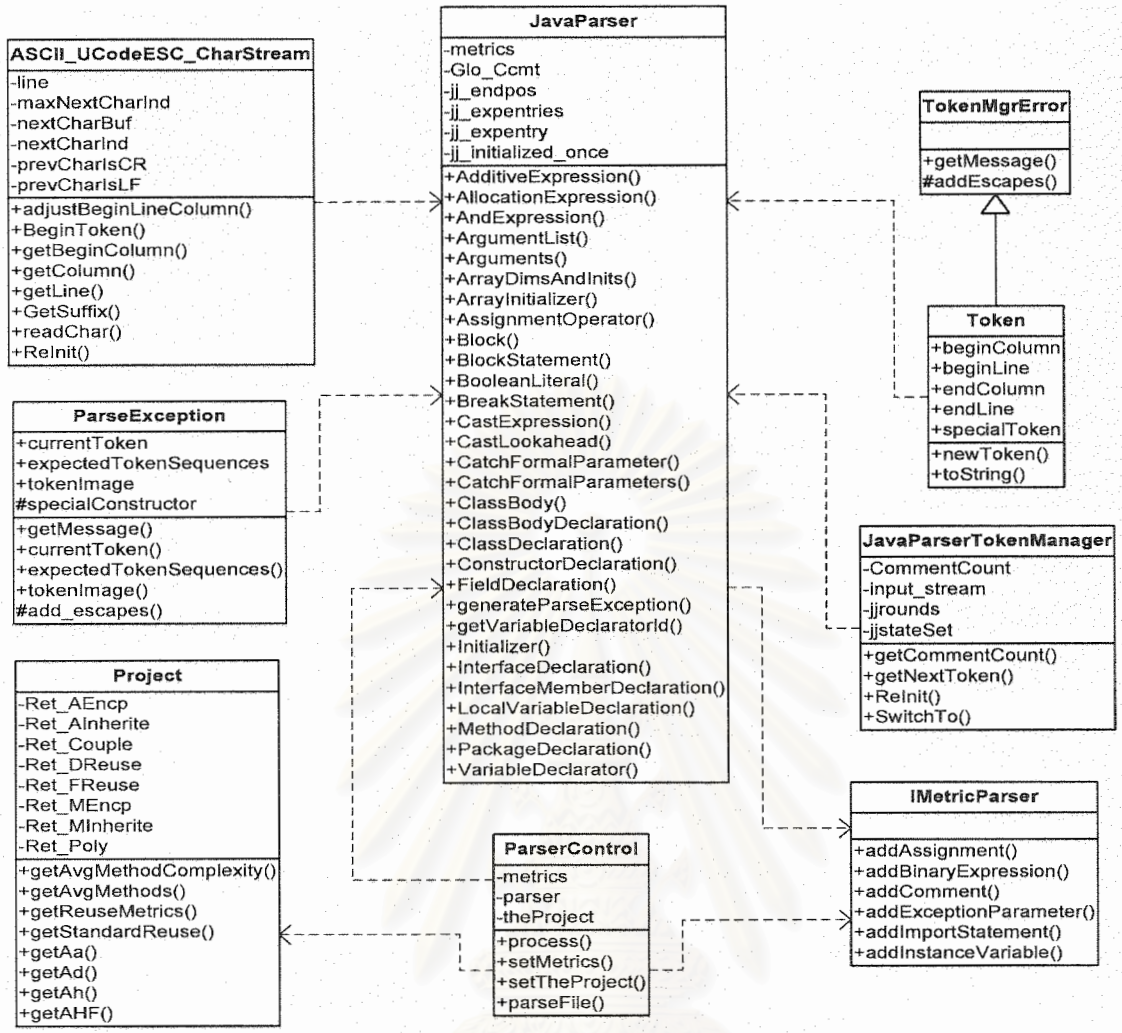
เป็นการแสดงชุดของคลาส ที่มีความสัมพันธ์ในการนำรหัสโปรแกรมภาษาจาวาเข้าสู่เครื่องมือ MTOOP รุ่นที่ 3 โดยเมื่อเริ่มต้นการทำงานคลาส JSProjectIF จะมีการเรียกการใช้งานไปยังคลาส FileFilterVector ในการระบุตำแหน่งแฟ้มข้อมูล คลาส TableSorter ซึ่งเป็นคลาสที่มีการถ่ายทอดคุณสมบัติมาจากคลาส TableMap จะมีการเรียกใช้งานจากคลาส Utils ในการจัดลำดับแฟ้มข้อมูล และคลาส FileBrowserPanel ในการนำแฟ้มข้อมูลเข้าสู่เครื่องมือ MTOOP รุ่นที่ 3 แต่คลาส FileBrowserPanel จะทำงานได้ก็ต่อเมื่อมีการเรียกใช้งานไปยังคลาส DirectoryComboBoxRenderer คลาส DirectoryTableModel คลาส FileBrowserPanel คลาส DirectoryComboBoxModel และคลาส getListCellRendererComponent สามารถแสดงได้ดังรูปที่ 3.10



รูปที่ 3.10 แผนภาพคลาสไดอะแกรมแสดงการนำรหัสโปรแกรมภาษาจาวาเข้าสู่ระบบ

3.3.3 แผนภาพคลาสแสดงการทำพาร์เซอร์

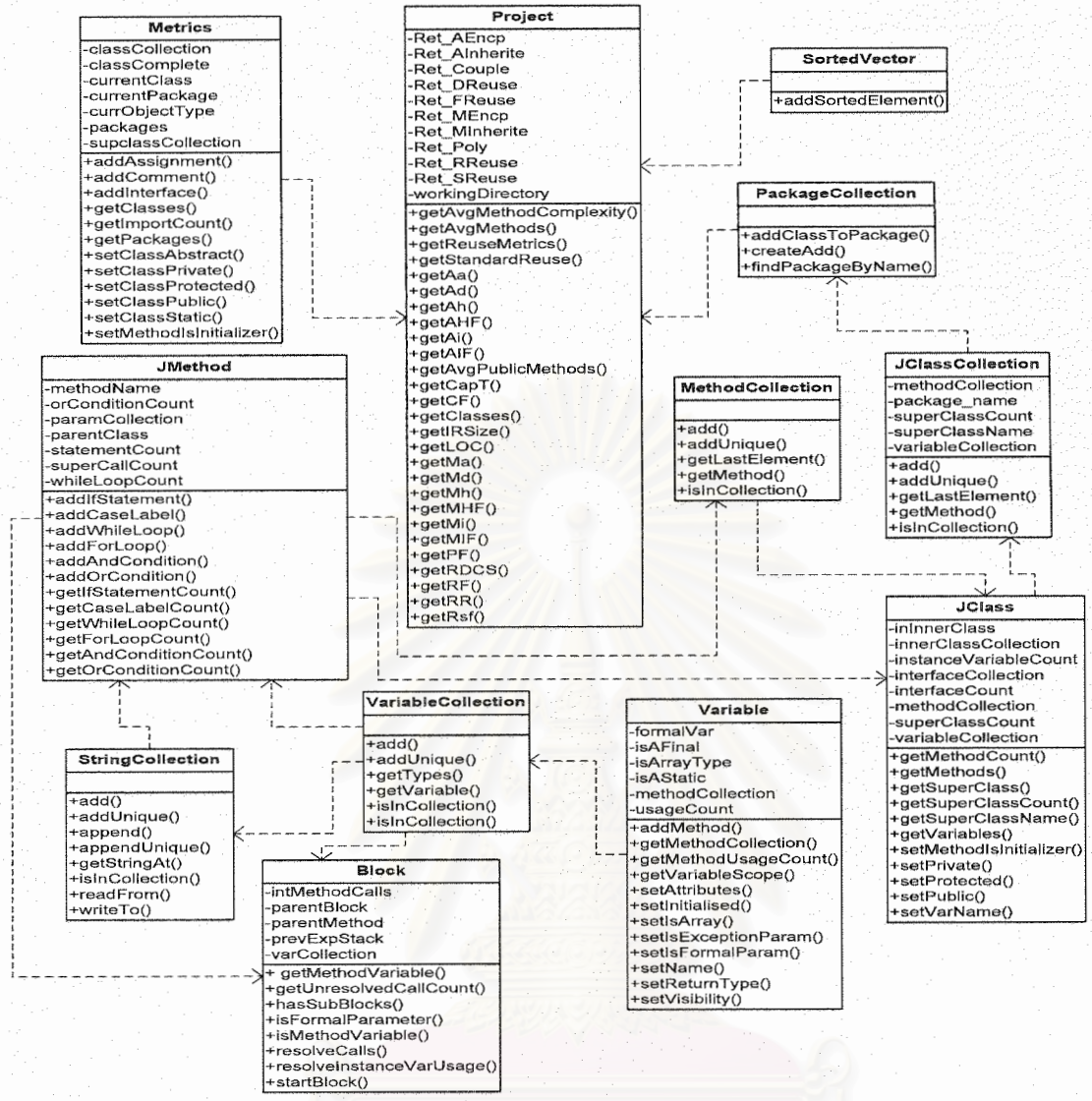
เป็นการแสดงชุดของคลาสที่มีความสัมพันธ์ในการทำพาร์เซอร์ โดยเมื่อเริ่มต้นการทำงานคลาส JavaParser จะมีการเรียกใช้งานจากคลาส Token ซึ่งเป็นคลาสที่มีการถ่ายทอดคุณสมบัติมาจากคลาส TokenMgrError เพื่อทำการสร้างสายอักขระของข้อมูล มีการเรียกใช้งานจากคลาส ASCII_UCodeESC_CharStream เพื่อทำการตรวจสอบรหัสแอสกีของข้อมูล มีการเรียกใช้งานจากคลาส ParseException เพื่อแสดงข้อความในกรณีแปลความหมายผิดพลาด และมีการเรียกใช้งานจากคลาส JavaParserTokenManager เพื่อทำการจัดการพาร์เซอร์ สามารถแสดงได้ดังรูปที่ 3.11



รูปที่ 3.11 แผนภาพคลาสไดอะแกรมแสดงการทำพาร์เซอร์

3.3.4 แผนภาพคลาสแสดงการคำนวณค่าตัววัด

เป็นการแสดงชุดของคลาสที่มีความสัมพันธ์ในการคำนวณค่าตัววัด โดยเมื่อเริ่มต้นการทำงาน คลาส Project จะเรียกการทำงานไปยังคลาส SortedVector เพื่อทำการเรียงลำดับแฟ้มข้อมูล และทำการเรียกคลาส Metrics และคลาส PackageCollection เพื่อทำการคำนวณค่าตัววัด โดยคลาส PackageCollection จะเรียกการคำนวณไปในระดับคลาสซึ่งได้แก่คลาส JClassCollection ค่าคำนวณที่ได้ในระดับคลาสได้มาจากการที่คลาส JClassCollection เรียกใช้การทำงานในคลาส JClass เพื่อให้คำนวณค่าตัววัด ส่วนในระดับของวิธีดำเนินการนั้นจะมีคลาส MethodCollection ควบคุมในการหาค่าวัดจากการทำงานร่วมกันของคลาส JMethod คลาส StringCollection คลาส VariableCollection และคลาส Variable สามารถแสดงได้ดังรูปที่ 3.12

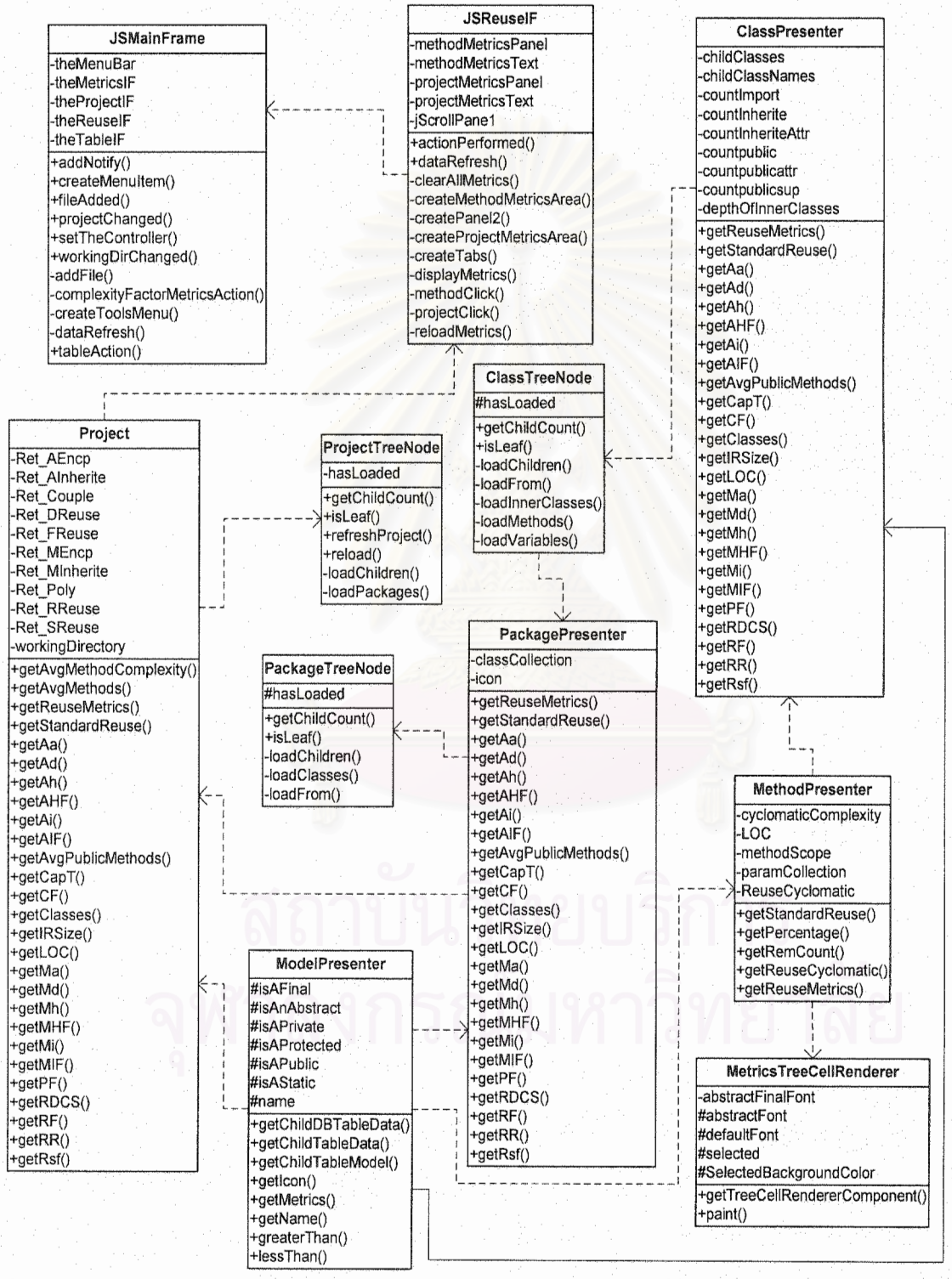


รูปที่ 3.12 แผนภาพคลาสไดอะแกรมแสดงการคำนวณค่าตัววัด

3.3.5 แผนภาพคลาสแสดงผลการวัดจากการเลือกเมนู View Reusability

เป็นการแสดงชุดของคลาสที่มีความสัมพันธ์จากการเลือกเมนู View Reusability โดยเมื่อเริ่มต้นการทำงานคลาส JSMainFrame จะเรียกการทำงานไปยังคลาส JSReuseIF ซึ่งเป็นคลาสที่ทำหน้าที่ในการนำค่าวัดที่คำนวณได้สำหรับการนำกลับมาใช้ใหม่มาแสดงผลในระดับของโครงงาน จากผลการทำงานของคลาส ProjectTreeNode ที่ได้มาจากการทำงานของคลาส Project โดยจะมีการเรียกการทำงานในคลาส PackageTreeNode ที่ได้มาจากการทำงานของคลาส PackagePresenter เพื่อรวบรวมค่าวัดในระดับแพ็คเกจ และคลาส PackagePresenter จะมีการเรียกการทำงานในคลาส ClassTreeNode ที่ได้มาจากการทำงานของคลาส ClassPresenter เพื่อรวบรวมค่าวัด

ในระดับคลาส และคลาส ClassPresenter จะมีการเรียกการทำงานในคลาส MethodPresenter เพื่อรวบรวมค่าวัดในระดับวิธีดำเนินการที่ได้จากการทำงานของคลาส MetricsTreeCellRenderer ต่อไป สามารถแสดงได้ดังรูปที่ 3.13



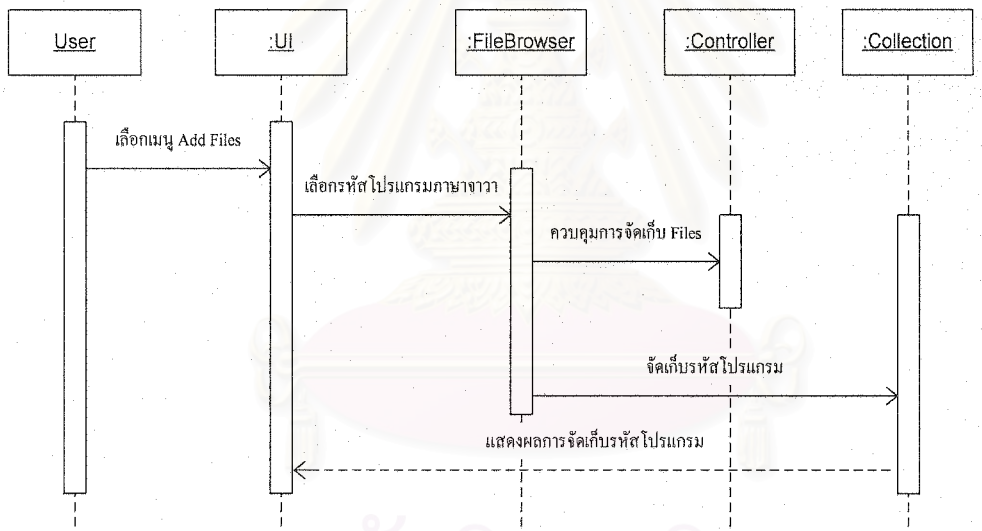
รูปที่ 3.13 แผนภาพคลาสไดอะแกรมแสดงผลการวัดจากการเลือกเมนู View Reusability

3.4 การเกิดปฏิสัมพันธ์ระหว่างวัตถุในระบบงาน

สามารถนำซีเควนซ์ไดอะแกรม ซึ่งเป็นไดอะแกรมที่จะเน้นไปที่ช่วงเวลาของการเกิดปฏิสัมพันธ์มาอธิบายการสร้างเครื่องมือ MTOOP รุ่นที่ 3 ได้ดังนี้

3.4.1 แผนภาพแสดงการเพิ่มรหัสโปรแกรมภาษาจาวาเข้าสู่ระบบ

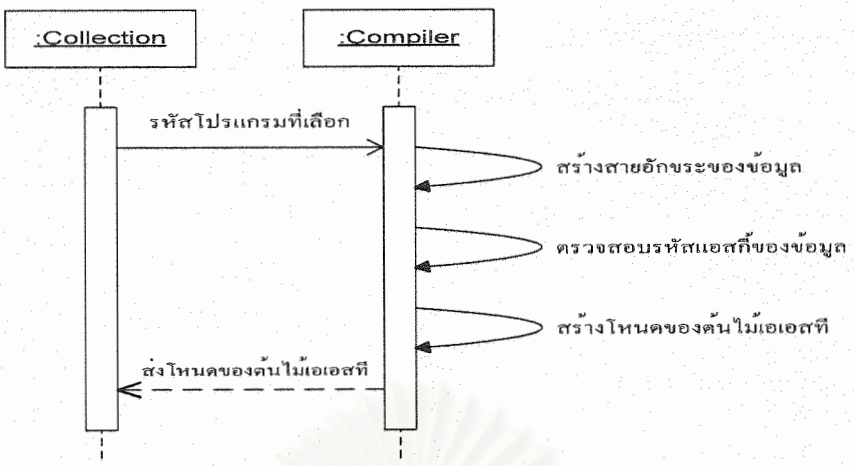
จากรูปที่ 3.14 เป็นแผนภาพที่แสดงการเพิ่มรหัสโปรแกรมภาษาจาวาเข้าสู่ระบบ โดยเริ่มจากผู้ใช้งานทำการเลือกเมนู Add Files จากวัตถุ UI หลังจากนั้นทำการเลือกรหัสโปรแกรมภาษาจาวาโดยใช้วัตถุ FileBrowser ในการนำรหัสโปรแกรมเข้าสู่ระบบโดยมีวัตถุ Controller ทำหน้าที่ในการควบคุมการจัดเก็บและจะใช้วัตถุ Collection ในการรวบรวมรหัสโปรแกรมเพื่อนำมาใช้ในการหาค่าตัววัด และผลของการจัดเก็บรหัสโปรแกรมจะถูกส่งกลับไปยังวัตถุ UI



รูปที่ 3.14 แผนภาพซีเควนซ์ไดอะแกรมแสดงการเพิ่มรหัสโปรแกรมภาษาจาวาเข้าสู่ระบบ

3.4.2 แผนภาพแสดงการทำพาร์เซอร์

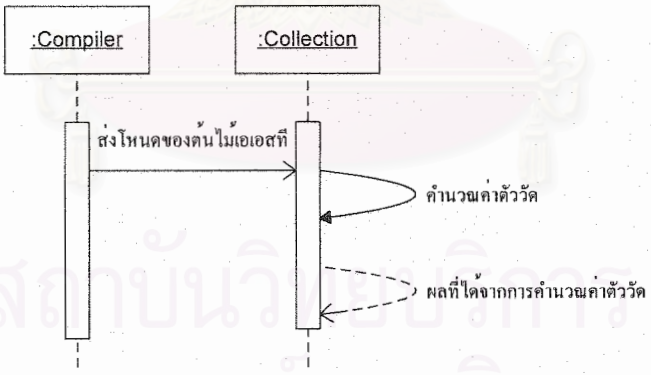
จากรูปที่ 3.15 เป็นแผนภาพที่แสดงการทำพาร์เซอร์โดยระบบจะนำรหัสโปรแกรมที่ได้จากการทำงานในวัตถุ Collection มาทำการสร้างสายอักขระของข้อมูลและทำการตรวจสอบความถูกต้องของข้อมูลเพื่อทำการสร้างต้นไม้เอเอสที โดยใช้วัตถุ Compiler ในการสร้าง และผลที่ได้จากการสร้างต้นไม้เอเอสทีจะส่งคืนกลับไปยังวัตถุ Collection ต่อไป



รูปที่ 3.15 แผนภาพซีเควนซ์ไดอะแกรมแสดงการทำพาร์เซอร์

3.4.3 แผนภาพแสดงการคำนวณค่าตัววัด

จากรูปที่ 3.16 เป็นแผนภาพที่แสดงการคำนวณค่าตัววัดโดยระบบจะนำโหนดของต้นไม้เอเอสทีจากวัตถุ Compiler มาทำการหาค่าวัดต่างๆ ที่มีการกำหนดในเครื่องมือ MTOOP รุ่นที่ 3 โดยใช้วัตถุ Collection ในการคำนวณค่าวัดต่างๆ และผลที่ได้จากการคำนวณค่าตัววัดจะส่งคืนกลับไปยังวัตถุ Collection ต่อไป

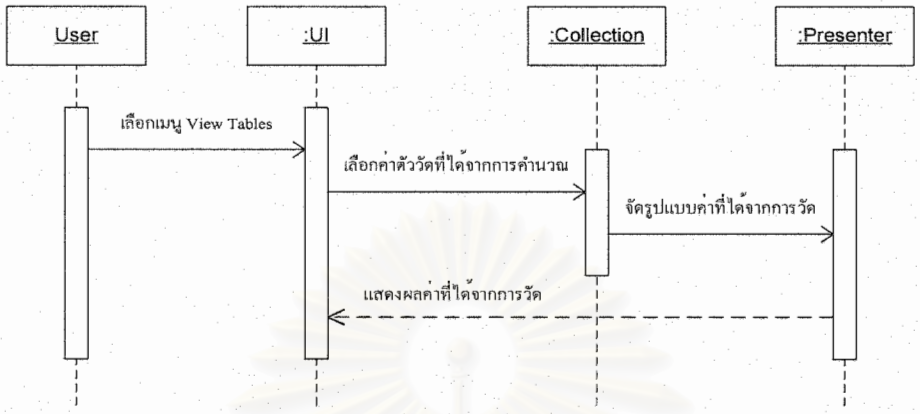


รูปที่ 3.16 แผนภาพซีเควนซ์ไดอะแกรมแสดงการคำนวณค่าวัด

3.4.4 แผนภาพแสดงผลการวัดจากการเลือกเมนู View Tables

จากรูปที่ 3.17 เป็นแผนภาพที่แสดงผลการวัดจากการเลือกเมนู View Tables โดยเริ่มจากผู้ใช้งานทำการเลือกเมนู View Tables จากวัตถุ UI หลังจากนั้นระบบจะทำการเลือกค่าตัววัดที่ได้จากการคำนวณจากวัตถุ Collection ซึ่งค่าที่ได้มีความสัมพันธ์กับ

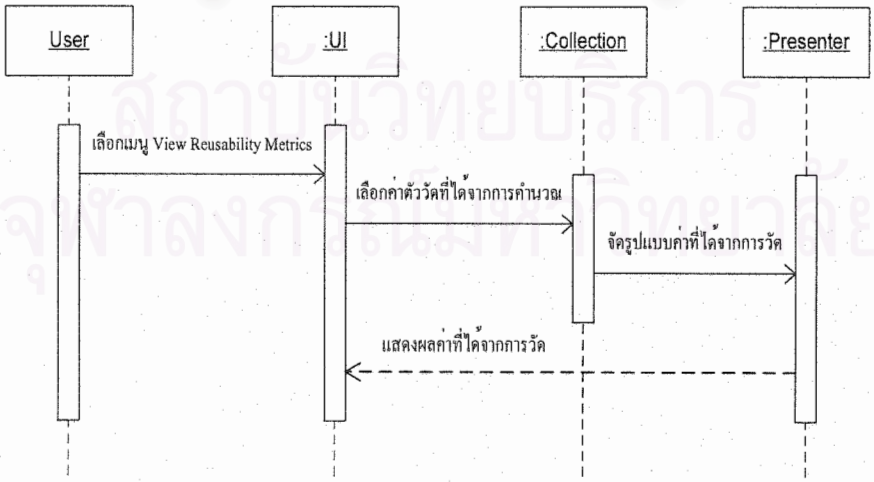
เมนูที่ผู้ใช้ทำการเลือกและจะถูกนำมาจัดรูปแบบในการแสดงผลโดยวัตถุ Presenter และผลของรูปแบบที่จัดได้จะถูกส่งกลับไปยังวัตถุ UI เพื่อนำไปแสดงผล



รูปที่ 3.17 แผนภาพซีเควนซ์ไดอะแกรมแสดงผลการวัดจากการเลือกเมนู View Tables

3.4.5 แผนภาพแสดงผลการวัดจากการเลือกเมนู View Reusability

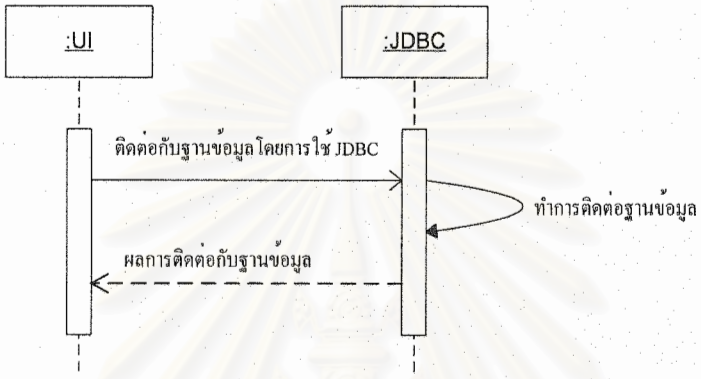
จากรูปที่ 3.18 เป็นแผนภาพที่แสดงผลการวัดจากการเลือกเมนู View Reusability โดยเริ่มจากผู้ใช้งานทำการเลือกเมนู View Reusability จากวัตถุ UI หลังจากนั้นระบบจะทำการเลือกค่าตัววัดที่ได้จากการคำนวณจากวัตถุ Collection ซึ่งค่าที่ได้มีความสัมพันธ์กับเมนูที่ผู้ใช้ทำการเลือก และจะถูกนำมาจัดรูปแบบในการแสดงผลโดยวัตถุ Presenter และผลของรูปแบบที่จัดได้จะถูกส่งกลับไปยังวัตถุ UI เพื่อนำไปแสดงผล



รูปที่ 3.18 แผนภาพซีเควนซ์ไดอะแกรมแสดงผลการวัดจากการเลือกเมนู View Reusability

3.4.6 แผนภาพแสดงการติดต่อกับฐานข้อมูล

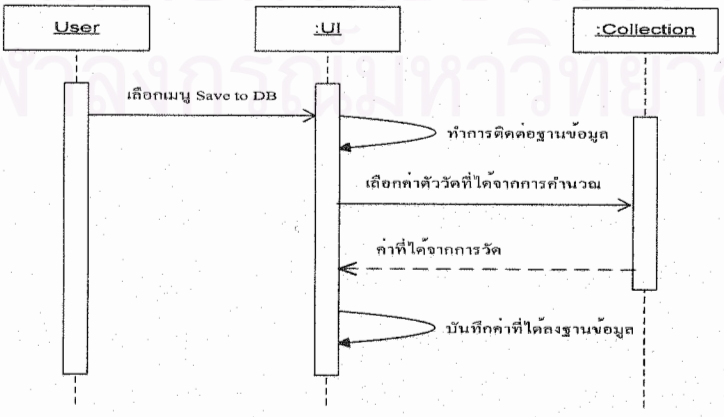
จากรูปที่ 3.19 เป็นแผนภาพที่แสดงผลการติดต่อกับฐานข้อมูลโดยวัตถุ UI จะเริ่มทำการติดต่อกับฐานข้อมูลโดยใช้จาวาดาต้าเบสคอนเน็กติวิตี (Java Database Connectivity – JDBC) ในการติดต่อและผลที่ได้จากการติดต่อจะถูกส่งค่ากลับ ไปยังวัตถุ UI



รูปที่ 3.19 แผนภาพซีเควนซ์ไดอะแกรมแสดงการติดต่อกับฐานข้อมูล

3.4.7 แผนภาพแสดงผลจากการเลือกเมนู Save to DB

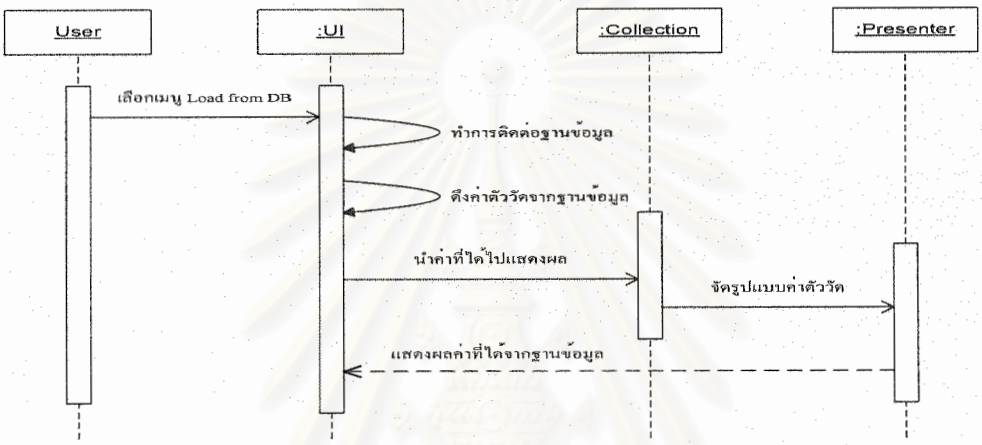
จากรูปที่ 3.20 เป็นแผนภาพที่แสดงจากการเลือกเมนู Save to DB โดยเริ่มจากผู้ใช้งานทำการเลือกเมนู Save to DB จากวัตถุ UI หลังจากนั้นระบบจะทำการเชื่อมการติดต่อไปยังฐานข้อมูลและนำค่าตัววัดที่คำนวณได้จากวัตถุ Collection จัดเก็บลงในฐานข้อมูลโดยใช้วัตถุ UI ในการจัดเก็บ



รูปที่ 3.20 แผนภาพซีเควนซ์ไดอะแกรมแสดงผลจากการเลือกเมนู Save to DB

3.4.8 แผนภาพแสดงผลจากการเลือกเมนู Load from DB

จากรูปที่ 3.21 เป็นแผนภาพที่แสดงจากการเลือกเมนู Load from DB โดยเริ่มจากผู้ใช้งานทำการเลือกเมนู Load from DB จากวัตถุ UI หลังจากนั้นระบบจะทำการเชื่อมการติดต่อไปยังฐานข้อมูล และนำค่าที่จัดเก็บในฐานข้อมูลมาแสดงผลโดยใช้วัตถุ Collection ในการรวมผลค่าตัววัดและใช้วัตถุ Presenter ในการจัดรูปแบบการแสดงผล และผลของรูปแบบที่จัดได้จะถูกส่งกลับไปยังวัตถุ UI เพื่อนำไปแสดงผล

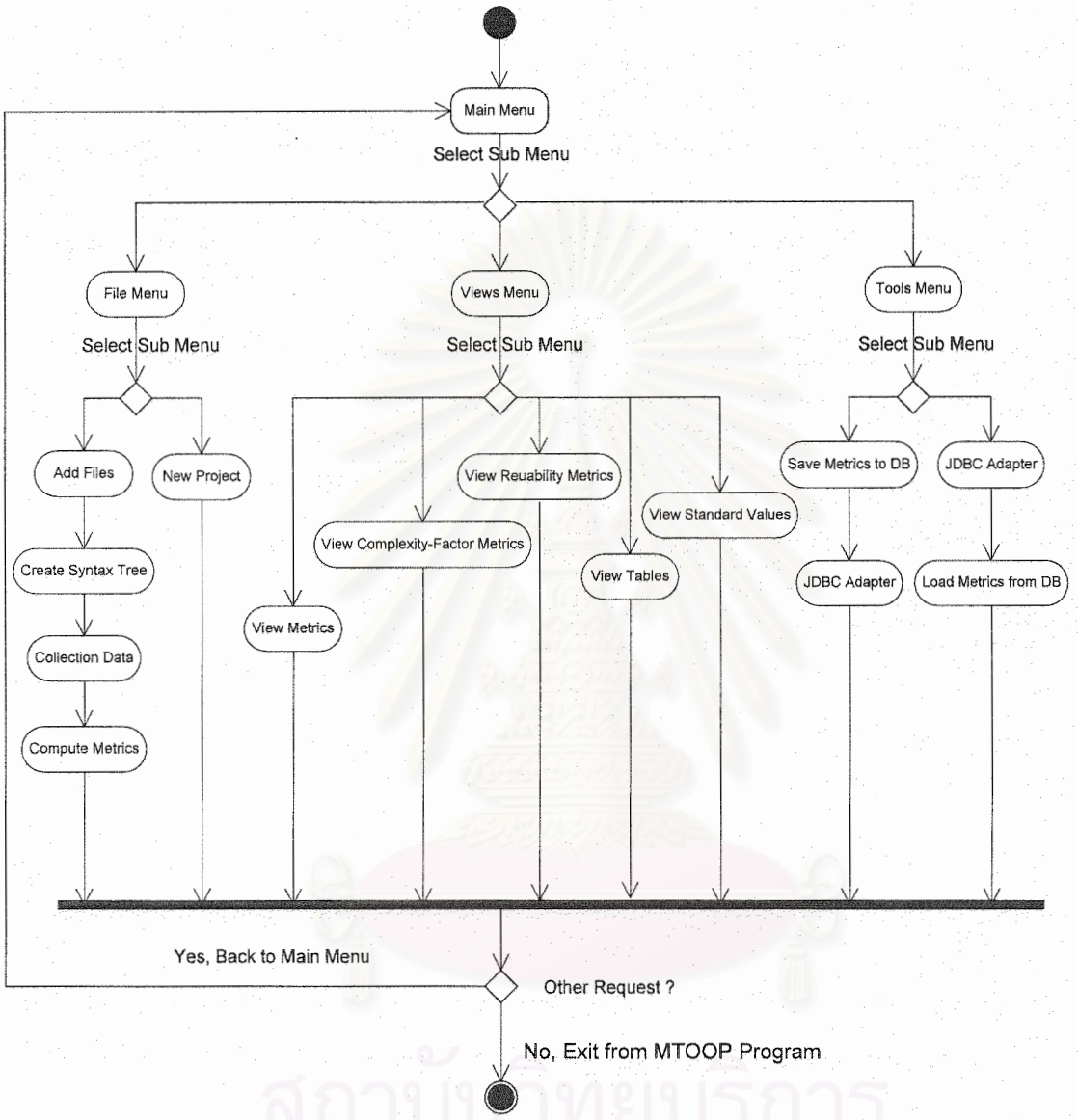


รูปที่ 3.21 แผนภาพซีเควนซ์ไดอะแกรมแสดงผลจากการเลือกเมนู Load from DB

3.5 แผนภาพแสดงการเกิดกิจกรรมในระบบงาน

จากรูปที่ 3.22 เป็นแผนภาพที่แสดงให้เห็นถึงกิจกรรมการทำงานต่างๆ ที่เกิดขึ้นภายในเครื่องมือ MTOOP รุ่นที่ 3 เมื่อผู้ใช้งานเข้าสู่ระบบผู้ใช้งานจะต้องทำการเลือกโปรแกรมต้นฉบับภาษาจาวาเข้ามาสู่เครื่องมือวัด หลังจากนั้นระบบจะนำโปรแกรมต้นฉบับมาทำการวิเคราะห์และสร้างโครงสร้างต้นไม้ขึ้นเพื่อจัดเก็บข้อมูลคุณสมบัติต่างๆ ของโปรแกรมต้นฉบับซึ่งแยกตามโครงงาน แพ็คเกจ คลาส วิธีดำเนินการ และตัวแปรอินสแตนซ์ จากนั้นก็จะทำการคำนวณหาค่าวัดต่างๆ ตามที่ได้แยกประเภทไว้ ซึ่งผู้ใช้งานสามารถทำการเพิ่มโปรแกรมต้นฉบับเข้ามาสู่โครงงานเดียวกันหรือสามารถนำโครงงานใหม่เข้าสู่ระบบเพื่อทำการวัดก็ได้ เมื่อผู้ใช้งานนำโปรแกรมต้นฉบับเข้าสู่ระบบเป็นที่เรียบร้อยผู้ใช้งานสามารถที่จะดูค่าตัววัดในรูปแบบต่างๆ ได้ 5 รูปแบบด้วยกันคือการดูค่าวัดทั่วไป การดูค่าวัดเพื่อให้ทราบถึงความซับซ้อนของปัจจัยที่เกิดขึ้น การดูค่าวัดเพื่อให้ทราบถึงความเหมาะสมของการนำโปรแกรมต้นฉบับกลับมาใช้ใหม่ การดูค่าวัดในรูปแบบของตาราง และการดูค่าวัดมาตรฐานสำหรับการนำกลับมาใช้ใหม่ นอกจากนี้ผู้ใช้งาน

สามารถที่จะทำการจัดเก็บข้อมูลค่าตัววัดต่างๆ ลงฐานข้อมูลและสามารถนำค่าที่จัดเก็บในฐานข้อมูลมาแสดงต่อผู้ใช้งานได้เมื่อมีการเรียกใช้

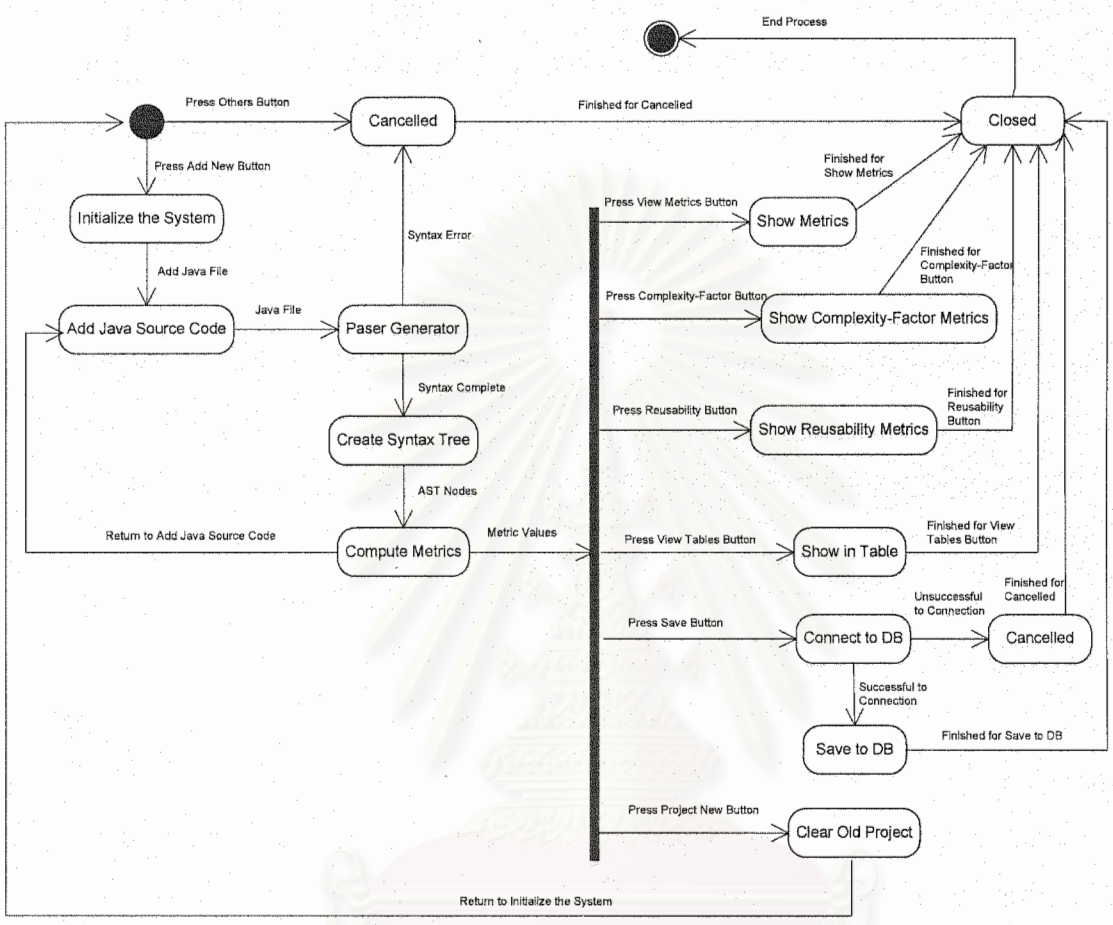


รูปที่ 3.22 แผนภาพแสดงกิจกรรมของเครื่องมือ MTOOP รุ่นที่ 3

3.6 แผนภาพสเตทชาร์ตไดอะแกรมแสดงการเปลี่ยนแปลงสถานภาพของระบบงาน

เป็นไดอะแกรมที่ใช้แสดงการเปลี่ยนสถานภาพของวัตถุ ตั้งแต่เริ่มต้นจนสิ้นสุดการเปลี่ยนแปลงในรอบหนึ่งๆ เราสามารถนำไดอะแกรมดังกล่าวมาอธิบายการเกิดปฏิสัมพันธ์ของวัตถุในการสร้างเครื่องมือ MTOOP รุ่นที่ 3 ได้ดังนี้

3.6.1 แผนภาพสเตทชาร์ตไดอะแกรมแสดงการเปลี่ยนสถานะ ในส่วนของการนำโปรแกรมต้นฉบับเข้าสู่เครื่องมือวัด การจัดเก็บค่าตัววัด และการแสดงผลค่าวัดที่ได้จากการคำนวณ



รูปที่ 3.23 แผนภาพแสดงการเปลี่ยนแปลงสถานภาพของระบบงานในส่วนของการนำโปรแกรมต้นฉบับเข้าสู่เครื่องมือวัด การจัดเก็บ และการแสดงผลค่าวัดที่ได้จากการคำนวณ

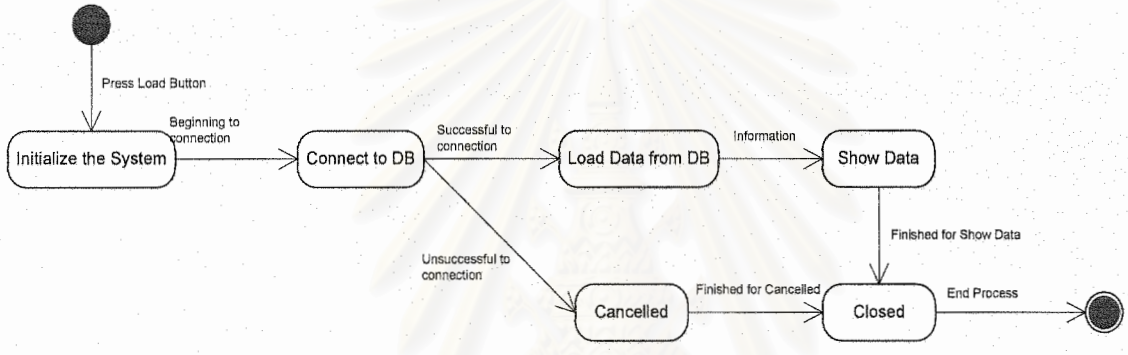
จากรูปที่ 3.23 เป็นแผนภาพที่แสดงให้เห็นการเปลี่ยนแปลงสถานภาพของระบบงานในส่วนของการนำโปรแกรมต้นฉบับเข้าสู่เครื่องมือวัด การจัดเก็บค่าตัววัด และการแสดงผลค่าตัววัดที่ได้จากการคำนวณเมื่อผู้ใช้งานทำการกดปุ่ม Add Files ซึ่งเป็นเมนูบนเครื่องมือ MTOOP รุ่นที่ 3 ระบบจะเริ่มเข้าสู่สถานะเริ่มต้นของการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่เครื่องมือวัด หลังจากนำรหัสโปรแกรมเข้าสู่ระบบฯ จะเปลี่ยนสถานะไปสู่สถานะ Parser Generator ซึ่งเป็นสถานะในการตรวจสอบความถูกต้องของรหัสโปรแกรม ในกรณีที่รหัสโปรแกรมไม่ถูกต้องระบบจะเปลี่ยนสถานะไปสู่สถานะ Cancelled และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการนำโปรแกรมต้นฉบับภาษาจา

วาเข้าสู่เครื่องมือวัด ในกรณีที่รหัสโปรแกรมถูกต้องตามโครงสร้างของภาษาระบบจะทำการเปลี่ยนสถานะไปสู่สถานะ Create Syntax Tree ซึ่งเป็นสถานะในการสร้างโหนดของต้นไม้เอเอสที เมื่อระบบสร้างโหนดของต้นไม้เอเอสทีเรียบร้อยระบบจะเปลี่ยนสถานะไปสู่สถานะ Compute Metrics ซึ่งเป็นสถานะในการคำนวณค่าวัดต่างๆ ที่มีอยู่ในเครื่องมือ MTOOP รุ่นที่ 3 ในกรณีที่ผู้ใช้งานต้องการนำรหัสโปรแกรมเพิ่มในโครงงานเดิมระบบจะกลับไปสู่สถานะ Add Source Code เพื่อเริ่มต้นสถานะการนำรหัสโปรแกรมเข้าสู่ระบบอีกครั้ง ในกรณีที่ผู้ใช้งานต้องการวัดค่าสำหรับโครงงานใหม่ระบบจะเปลี่ยนสถานะไปสู่สถานะ Clear Old Project และกลับไปสู่สถานะเริ่มต้นเพื่อรอรับสถานะการกดปุ่ม Add Files อีกครั้ง ในกรณีผู้ใช้งานเลือกเมนู View Metrics ระบบจะเปลี่ยนสถานะจาก Compute Metrics ไปสู่สถานะ Show Metrics เพื่อนำค่าที่ได้จากการวัดมาแสดงผลบนหน้าจอของเครื่องมือ MTOOP รุ่นที่ 3 โดยแยกตาม โครงงาน แพ็คเกจ คลาส วิธีดำเนินการ และตัวแปร และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการแสดงผลข้อมูล ในกรณีผู้ใช้งานเลือกเมนู View Complexity-Factor ระบบจะเปลี่ยนสถานะจาก Compute Metrics ไปสู่สถานะ Show Complexity-Factor Metrics เพื่อนำค่าปัจจัยของความซับซ้อนที่ได้จากการหาค่าวัดมาแสดงผลบนหน้าจอของเครื่องมือ MTOOP รุ่นที่ 3 โดยแยกตามโครงงาน แพ็คเกจ คลาส และวิธีดำเนินการ และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการแสดงผลข้อมูล ในกรณีผู้ใช้งานเลือกเมนู View Reusability ระบบจะเปลี่ยนสถานะจาก Compute Metrics ไปสู่สถานะ Show Reusability Metrics เพื่อนำค่าที่คำนวณได้มาทำการประมาณ เพื่อดูความเหมาะสมที่จะนำโปรแกรมต้นฉบับภาษาจาวากลับมาใช้ใหม่ โดยนำผลที่ได้มาแสดงบนหน้าจอของเครื่องมือ MTOOP รุ่นที่ 3 โดยแยกตาม โครงงาน และวิธีดำเนินการ และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการแสดงผลข้อมูล และในกรณีผู้ใช้งานเลือกเมนู View Tables ระบบจะเปลี่ยนสถานะจาก Compute Metrics ไปสู่สถานะ Show in Table เพื่อนำค่าที่ได้จากการวัดมาแสดงผลในรูปแบบของตารางบนหน้าจอของเครื่องมือ MTOOP รุ่นที่ 3 โดยแยกตาม โครงงาน แพ็คเกจ และวิธีดำเนินการ และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการแสดงผลข้อมูล

ในกรณีผู้ใช้งานเลือกเมนู Save DB ระบบจะเปลี่ยนสถานะจาก Compute Metrics ไปสู่สถานะ Connect to DB เพื่อทำการเริ่มต้นการติดต่อกับฐานข้อมูลในกรณีที่ไม่สามารถติดต่อฐานข้อมูลได้ระบบจะทำการเปลี่ยนสถานะจากสถานะ Connect to DB ไปสู่สถานะ Cancelled และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการบันทึกข้อมูล ในกรณีที่ระบบสามารถติดต่อฐานข้อมูลได้ระบบจะทำการเปลี่ยนสถานะ

จากสถานะ Connect to DB ไปสู่สถานะ Save to DB ซึ่งเป็นสถานะในการบันทึกข้อมูลต่างๆ ที่ได้จากการคำนวณค่าตัววัดเพื่อบันทึกลงฐานข้อมูลและจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการบันทึกข้อมูล และในกรณีที่ผู้ใช้งานเลือกเมนู View Metrics, View Complexity-Factor, View Reusability, View Tables และ Save DB ก่อนที่จะทำการเลือกเมนู Add Files ระบบจะเข้าสู่สถานะ Cancelled และสถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานทันที

3.6.2 แผนภาพสเตทชาร์ตไดอะแกรมแสดงการเปลี่ยนสถานะ ในส่วนของการโหลดข้อมูลจากฐานข้อมูล



รูปที่ 3.24 แผนภาพแสดงการเปลี่ยนแปลงสถานภาพของระบบงานในส่วนของการโหลดข้อมูลจากฐานข้อมูล

จากรูปที่ 3.24 เป็นแผนภาพที่แสดงให้เห็นการเปลี่ยนแปลงสถานภาพของระบบงานในส่วนของการโหลดข้อมูลจากฐานข้อมูล โดยเมื่อผู้ใช้งานทำการกดปุ่ม Load DB ซึ่งเป็นเมนูบนเครื่องมือ MTOOP รุ่นที่ 3 ระบบจะเริ่มเข้าสู่สถานะเริ่มต้นของการโหลดข้อมูลหลังจากนั้นระบบจะทำการเริ่มต้นติดต่อกับฐานข้อมูล ในกรณีที่ไม่สามารถติดต่อฐานข้อมูลได้ระบบจะทำการเปลี่ยนสถานะจากสถานะ Connect to DB ไปสู่สถานะ Cancelled และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการโหลดข้อมูล ในกรณีที่ระบบสามารถติดต่อฐานข้อมูลได้ระบบจะทำการเปลี่ยนสถานะจากสถานะ Connect to DB ไปสู่สถานะ Load Data from DB ซึ่งเป็นสถานะในการนำข้อมูลต่างๆ ที่เกี่ยวข้องจากฐานข้อมูลหลังจากนั้นระบบจะเปลี่ยนสถานะไปสู่สถานะ Show Data ซึ่งเป็นการนำข้อมูลที่ได้มาแสดงผลบนหน้าจอของเครื่องมือ MTOOP รุ่นที่ 3 และจะเข้าสู่สถานะ Closed ซึ่งเป็นการสิ้นสุดการทำงานในส่วนของการโหลดข้อมูล

บทที่ 4

การพัฒนาเครื่องมือ MTOOP รุ่นที่ 3

ในบทนี้เป็นการอธิบายรายละเอียดเกี่ยวกับเครื่องมือที่ใช้ในการพัฒนา การสร้างพาร์เซอร์ การประยุกต์โปรแกรมจาวาซีซี และการคำนวณค่าตัววัดในการพัฒนาเครื่องมือ MTOOP รุ่นที่ 3 มีรายละเอียดและกระบวนการพัฒนาดังต่อไปนี้คือ

4.1 เครื่องมือที่ใช้ในการพัฒนา MTOOP รุ่นที่ 3

4.1.1 ฮาร์ดแวร์ที่ใช้มีรายละเอียดดังนี้

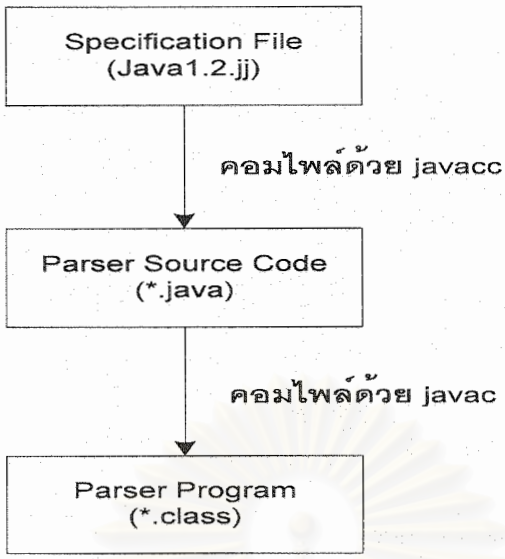
- ไมโครคอมพิวเตอร์ หน่วยประมวลผลกลางชนิดเพนเทียม III ความเร็ว 500 เมกะเฮิรตซ์
- หน่วยความจำหลัก 256 เมกะไบต์
- ความจุจานบันทึกแบบแข็ง (hard disk) ขนาด 5 กิกะไบต์
- ความละเอียดของจอภาพขนาด 1,024 x 768 แสดงสี 16 ล้านสี

4.1.2 ซอฟต์แวร์ที่ใช้มีรายละเอียดดังนี้

- ไมโครซอฟต์วินโดว์ 2000 (Microsoft window 2000) ใช้เป็นระบบปฏิบัติการ
- ไมโครซอฟต์แอกเซส รุ่น 2002 (Microsoft Access 2002) ใช้เป็นระบบจัดการฐานข้อมูล
- โปรแกรมเจบิวเดอร์ รุ่น 5.0 เอนเตอร์ไพส์ (JBuilder 5.0 Enterpise) ใช้เป็นเครื่องมือในการพัฒนาโปรแกรม

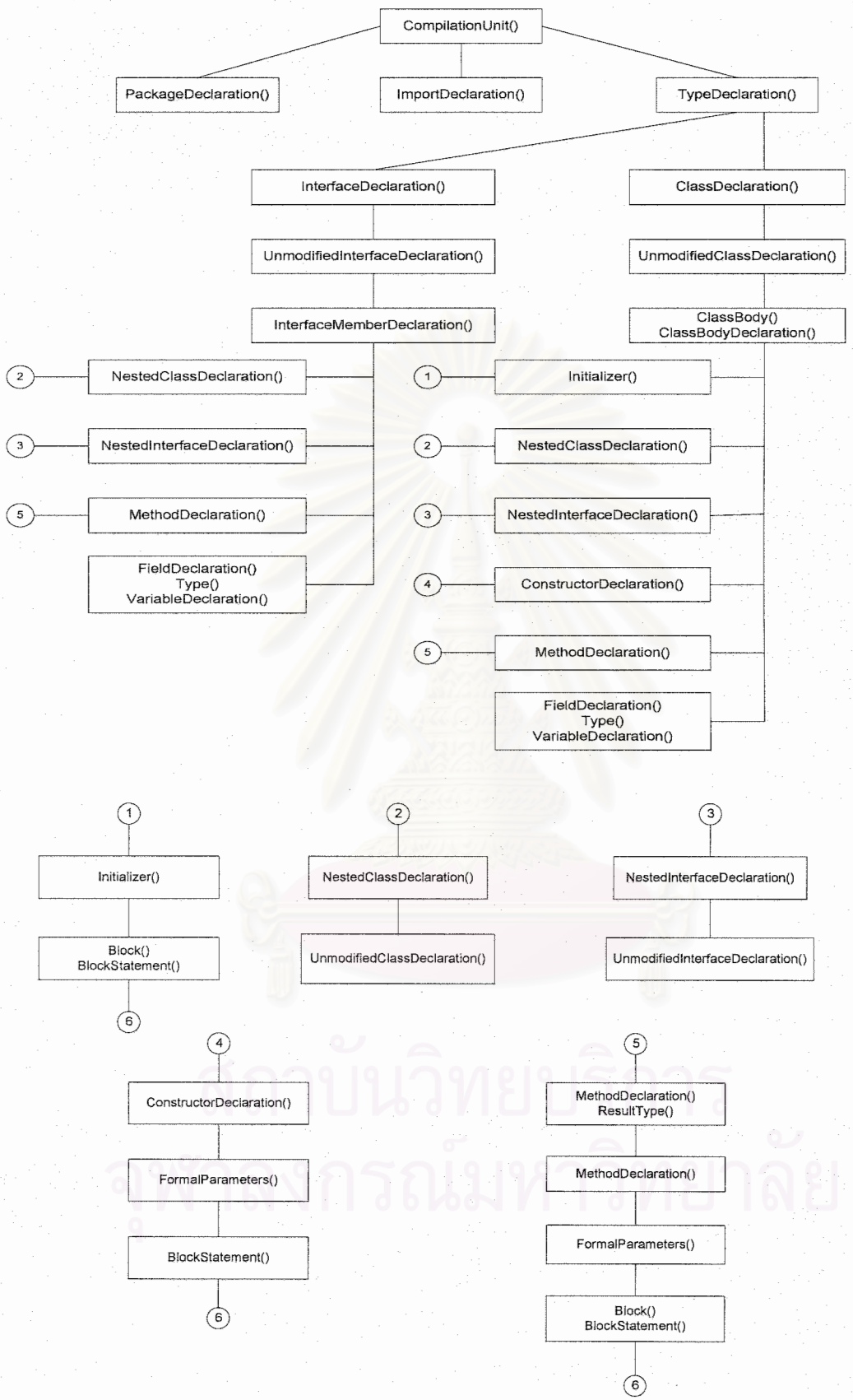
4.2 การสร้างพาร์เซอร์ของเครื่องมือ MTOOP รุ่นที่ 3

ขั้นตอนนี้เป็นการนำกระบวนการที่ได้ออกแบบไว้ สำหรับการสร้างพาร์เซอร์ที่กล่าวไว้ในบทที่ 3 มาทำการสร้างพาร์เซอร์ของเครื่องมือ MTOOP รุ่นที่ 3 โดยใช้โปรแกรมจาวาซีซี (JavaCC Program) ในการสร้างซึ่งโปรแกรมจาวาซีซีจะทำหน้าที่เป็นตัวพาร์เซอร์เจนเนอเรเตอร์ (Parser Generator) ที่ใช้ในการอ่านไฟล์ข้อกำหนดแล้วเปลี่ยนให้ไปอยู่ในรูปของโปรแกรมภาษาจาวาที่สามารถตรวจสอบความถูกต้องตามไวยากรณ์ที่กำหนดไว้ในไฟล์ข้อกำหนดได้

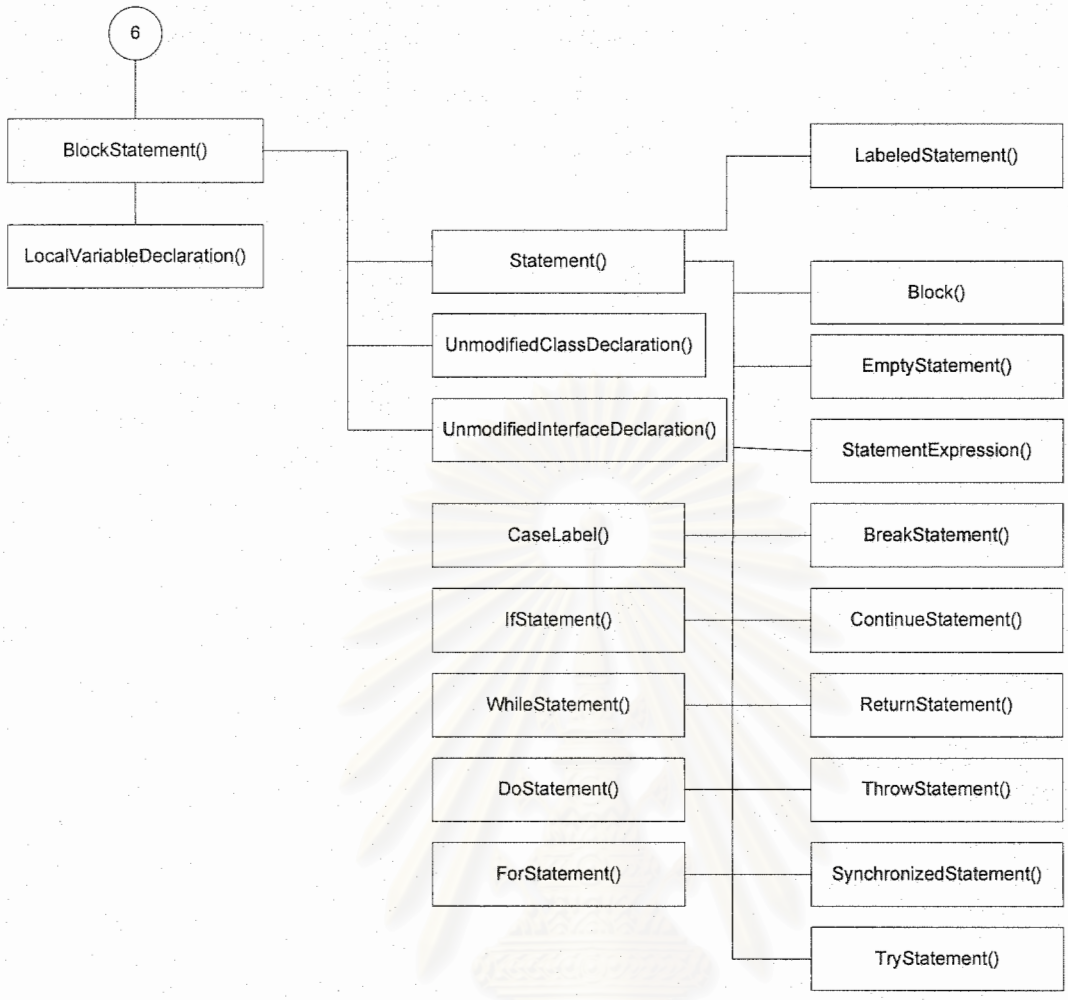


รูปที่ 4.1 แผนภาพแสดงขั้นตอนการใช้โปรแกรมจาวาซีซีเพื่อทำการสร้างต้นไม้เอเอสทีของเครื่องมือ MTOOP รุ่นที่ 3

รูปที่ 4.1 เป็นแผนภาพที่แสดงขั้นตอนการใช้โปรแกรมจาวาซีซีเพื่อทำการสร้างต้นไม้เอเอสทีของเครื่องมือ MTOOP รุ่นที่ 3 โดยการนำรหัสโปรแกรมภาษาจาวาของเครื่องมือ MTOOP รุ่นที่ 3 มาผ่านขั้นตอนของการทำเลกซิคัลอนาไลเซอร์ (เรียกอีกอย่างว่าการทำสแกนเนอร์) โดยจะทำการอ่านอักขระแต่ละตัวเข้ามาแล้วทำการจัดกลุ่มอักขระเหล่านั้นออกเป็นโทเคน (Tokens) หลังจากนั้นจะนำส่วนของโทเคนที่ได้ไปทำในขั้นตอนของการทำซินแท็กซ์อนาไลเซอร์ (เรียกอีกอย่างว่าการทำพาร์เซอร์) ซึ่งในขั้นตอนนี้จะทำการตรวจสอบโทเคนที่รับเข้ามาว่ามีการเรียงถูกต้องตามหลักไวยากรณ์หรือไม่โดยจะได้เป็นโครงสร้างต้นไม้ที่เรียกว่าเอเอสที ในแต่ละโหนดของต้นไม้เอเอสทีจะเก็บข้อมูลหรือคุณสมบัติต่างๆ ที่อยู่ภายในโหนดเช่น คลาส วิธีดำเนินการ หรือคำสั่งเงื่อนไขต่างๆ เป็นต้น จากนั้นจึงนำโปรแกรมฉบับต้นภาษาจาวาที่ผ่านขั้นตอนการทำพาร์เซอร์มาคอมไพล์ด้วยโปรแกรมจาวาซีซึ่งเป็นคอมไพเลอร์จากคลาสไลบรารีของจาวาดีเวลลอปเมนต์ทูลคิด (Java Development Toolkit - JDK) ผลที่ได้จากการคอมไพล์ก็คือจาวาคลาส (Java Class - *.class) ที่สามารถนำไปใช้งานได้ และสามารถสรุปเป็นแผนภาพต้นไม้เอเอสทีเพื่อแสดงโหนดต่างๆ ที่ได้จากการสร้างซินแท็กซ์ทรีของคลาสจาวาพาร์เซอร์ (JavaParser) ของเครื่องมือ MTOOP รุ่นที่ 3 สามารถแสดงได้ดังรูปที่ 4.2



รูปที่ 4.2 แผนภาพต้นไม้แสดงโหนดต่างๆ ที่มาจากการสร้างซินแท็กซ์ทรีของคลาสจาวาพาร์เซอร์ในการพัฒนาเครื่องมือ MTOOP รุ่นที่ 3



รูปที่ 4.2 (ต่อ) แผนภาพต้นไม้แสดงโหนดต่างๆ ที่มาจากการสร้างซินแท็กซ์ทรีของคลาสจาวาพาร์เซอร์ในการพัฒนาเครื่องมือ MTOOP รุ่นที่ 3

4.3 การประยุกต์โปรแกรมจาวาซีซีมาใช้ในการพัฒนาเครื่องมือ MTOOP รุ่นที่ 3

ขั้นตอนนี้เป็นการนำโปรแกรมจาวาซีซี ซึ่งเป็นโปรแกรมที่ใช้ในการอ่านไฟล์ข้อกำหนดมาใช้ในการพัฒนาเครื่องมือ MTOOP รุ่นที่ 3 โดยการนำรหัสโปรแกรมภาษาจาวามาทำตามขั้นตอนในหัวข้อที่ 4.2 แต่สิ่งที่ได้จากการทำตามขั้นตอนดังกล่าวคือโปรแกรมที่สามารถตรวจสอบโปรแกรมต้นฉบับภาษาจาวาว่ามีความถูกต้องตามหลักไวยากรณ์หรือไม่เท่านั้น แต่ไม่สามารถเก็บค่าที่ได้จากการกระจายคำได้ ในกรณีที่ต้องการจัดเก็บข้อมูลต่างๆ ของโปรแกรมต้นฉบับระหว่างการกระจายคำในไวยากรณ์ จึงจำเป็นต้องทำการแก้ไขเพิ่มเติมรหัสโปรแกรมภาษาจาวาลงในโปรแกรมต้นฉบับของพาร์เซอร์ยกตัวอย่างเช่น ผลจากการคอมไพล์ไฟล์ข้อกำหนดภาษาจาวาที่ยังไม่ได้มีการแก้ไข ดังรูปที่ 4.3

```
static final public void PackageDeclaration()
        Throws ParseException {
        jj_consume_token(PACKAGE);
        Name();
        jj_consume_token(SEMICOLON); }
  static final public void Name() throws ParseException {
        jj_consume_token(IDENTIFIER);
        label_19:
        while (true) {
            if (jj_2_14(2)) {
                ;
            } else {
                break label_19; }
            jj_consume_token(DOT);
            jj_consume_token(IDENTIFIER); } }
```

รูปที่ 4.3 แสดงตัวอย่างหัสโปรแกรมภาษาจาวาก่อนทำการแก้ไข

ในกรณีที่ต้องการจัดเก็บค่าของแพ็คเกจไว้ในตัวแปรชื่อ packageName ซึ่งสมมติว่าได้มีการประกาศชนิดของตัวแปรไว้แล้ว สามารถแสดงการแทรกรหัสโปรแกรมเพื่อที่จะเก็บชื่อของแพ็คเกจได้ดังรูปที่ 4.4 และจะทำการแก้ไขรหัสโปรแกรมในลักษณะนี้จนได้ข้อมูลครบตามต้องการแล้วจึงนำโปรแกรมต้นฉบับนี้ไปคอมไพล์ด้วยโปรแกรมจาวาซีต่อไป


```
static final public void PackageDeclaration()
        Throws ParseException {
        jj_consume_token(PACKAGE);
        packageName = Name();
        jj_consume_token(SEMICOLON);
        metrics.addPackage(packageName); }
```

รูปที่ 4.4 แสดงตัวอย่างรหัสโปรแกรมภาษาจาวาหลังจากทำการแก้ไข

```
static final public void Name() throws ParseException {
        token t = jj_consume_token(IDENTIFIER);
        string strName = t.image;
        label_19:
        while (true) { if (jj_2_14(2)) {
                        ;
                } else {
                        break label_19; }
                jj_consume_token(DOT);
                t = jj_consume_token(IDENTIFIER);
                strName = strName + "." + t.image; }
        return strName; }
```

รูปที่ 4.4 (ต่อ) แสดงตัวอย่างรหัสโปรแกรมภาษาจาวาหลังจากทำการแก้ไข

4.4 การคำนวณค่าตัววัด

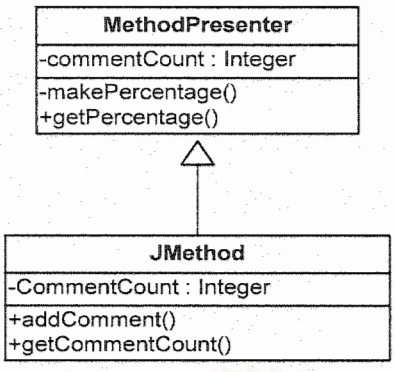
ขั้นตอนนี้เป็นการนำกระบวนการสำหรับการคำนวณค่าตัววัดที่กล่าวไว้ในบทที่ 3 มาทำการพัฒนาในเครื่องมือ MTOOP รุ่นที่ 3 ดังนี้

4.4.1 การคำนวณค่าตัววัดในส่วนของวิธีดำเนินการ

สามารถอธิบายคลาสและวิธีดำเนินการที่เกี่ยวข้องในการหาค่าตัววัดได้ดังต่อไปนี้

4.4.1.1 การหาอัตราส่วนของข้อความ

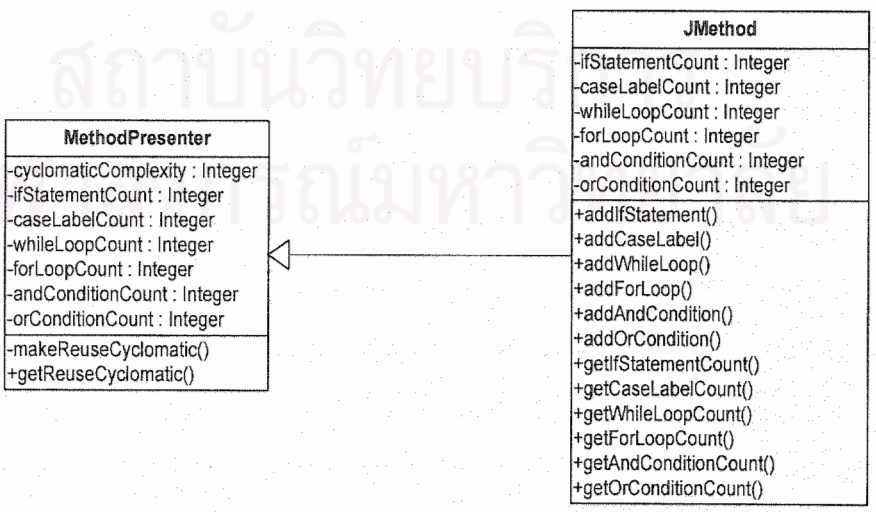
จากรูปที่ 4.5 เป็นการแสดงคลาสและวิธีดำเนินการที่ใช้ในการคำนวณการหาอัตราส่วนของข้อความ โดยคลาส MethodPresenter จะเรียกใช้วิธีดำเนินการ addComment() และ getCommentCount() จากคลาส JMethod เพื่อทำการหาค่าตัววัดหลังจากนั้นจะเรียกใช้วิธีดำเนินการ makePercentage() ในการหาอัตราส่วนของข้อความ ส่วน getPercentage() จะทำหน้าที่ในการนำผลที่ได้มาแสดง



รูปที่ 4.5 แสดงคลาสและวิธีดำเนินการในการคำนวณหาอัตราส่วนของข้อความ

4.4.1.2 การวัดความซับซ้อน

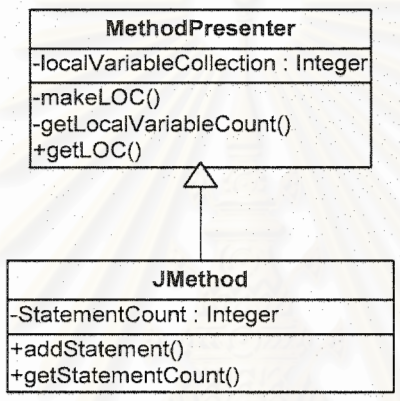
จากรูปที่ 4.6 เป็นการแสดงคลาสและวิธีดำเนินการที่ใช้ในการคำนวณการวัดค่าความซับซ้อนภายในวิธีดำเนินการ โดยคลาส MethodPresenter จะเรียกใช้วิธีดำเนินการ addIfStatement() addCaseLabel() addWhileLoop() addForLoop() addAndCondition() addOrCondition() getIfStatementCount() getCaseLabelCount() getWhileLoopCount() getForLoopCount() getAndConditionCount() และ วิธีดำเนินการ getOrConditionCount() จากคลาส JMethod เพื่อทำการหาค่าตัววัดหลังจากนั้นจะเรียกใช้วิธีดำเนินการ makeReuseCyclomatic() ในการหาค่าความซับซ้อนภายในวิธีดำเนินการ ส่วน getReuseCyclomatic() จะทำหน้าที่ในการนำผลที่ได้มาแสดง



รูปที่ 4.6 แสดงคลาสและวิธีดำเนินการในการคำนวณหาการวัดความซับซ้อน

4.4.1.3 การนับจำนวนบรรทัดของวิธีดำเนินการ

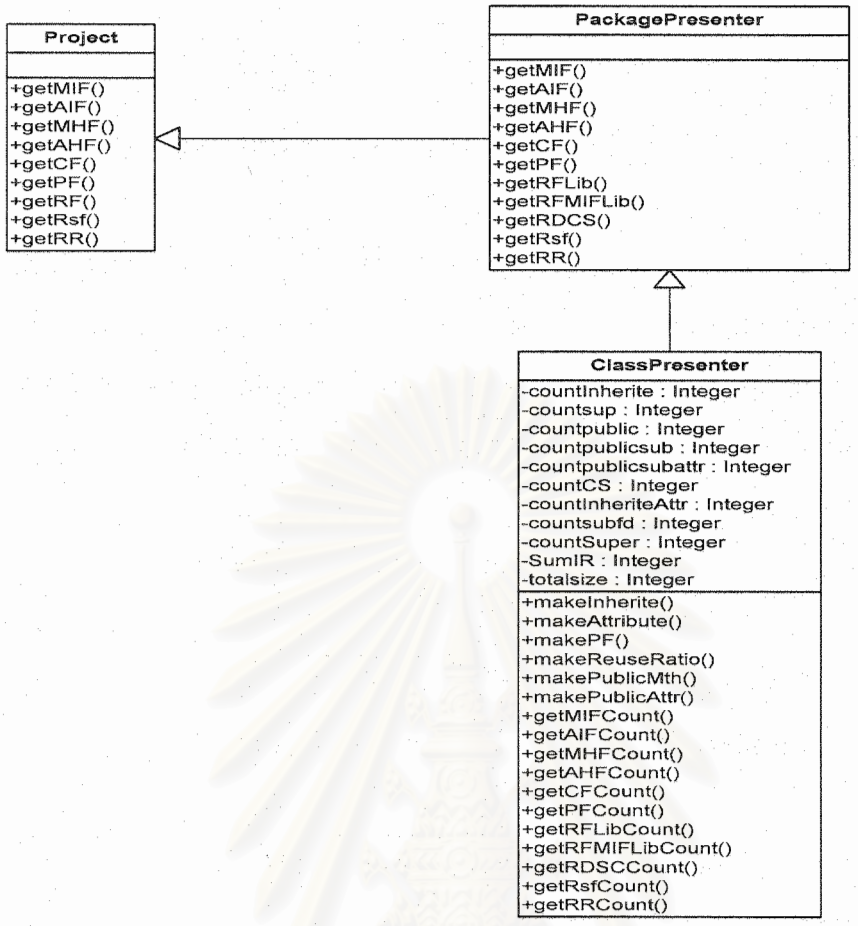
จากรูปที่ 4.7 เป็นการแสดงคลาสและวิธีดำเนินการที่ใช้ในการคำนวณการนับจำนวนบรรทัดของวิธีดำเนินการ โดยคลาส MethodPresenter จะเรียกใช้วิธีดำเนินการ addStatement() และ getStatementCount() จากคลาส JMethod เพื่อทำการหาค่าตัววัดหลังจากนั้น getLocalVariableCount() จะทำหน้าที่ในการหาค่าตัวแปรที่ใช้ในวิธีดำเนินการ แล้วส่งผลที่ได้ให้ makeLOC() เพื่อทำหน้าที่ในการหาจำนวนบรรทัดของวิธีดำเนินการต่อไป ส่วน getLOC() จะทำหน้าที่ในการนำผลที่ได้มาแสดง



รูปที่ 4.7 แสดงคลาสและวิธีดำเนินการในการคำนวณหาจำนวนบรรทัดของวิธีดำเนินการ

4.4.2 การคำนวณค่าตัววัดในส่วนของโครงงาน

จากรูปที่ 4.8 เป็นการแสดงคลาสและวิธีดำเนินการที่เกี่ยวข้องในการหาค่าตัววัดในส่วนของโครงงาน โดยแต่ละวิธีดำเนินการภายในคลาส ClassPresenter จะทำหน้าที่ในการคำนวณหาค่าตัววัดสำหรับค่าวัดต่างๆ ซึ่งวิธีดำเนินการที่เกี่ยวข้องได้แก่ makeInherite() makeAttribute() makePF() makeReuseRatio() makePublicMth() makePublicAttr() getMIFCount() getAIFCount() getMHFCount() getAHFCount() getCFCount() getPFCount() getRFLibCount() getRFMIFLibCount() getRDCSCount() getRSFCount() และ getRRCount() หลังจากนั้นคลาส PackagePresenter จะทำการรวมค่าที่วัดได้ในระดับแพ็คเกจ ส่วนวิธีดำเนินการ getMIF() getAIF() getMHF() getAHF() getCF() getPF() getRF() getRsf() และ getRR() ในคลาส Project จะทำการคำนวณค่าตัววัดเพื่อนำมาแสดงในส่วนของโครงงาน



รูปที่ 4.8 แสดงคลาสและวิธีดำเนินการในการคำนวณหาค่าตัววัดในส่วนของโครงงาน

บทที่ 5

การทดสอบเครื่องมือ MTOOP รุ่นที่ 3

ผู้วิจัยได้ทำการทดสอบการทำงานของเครื่องมือ MTOOP รุ่นที่ 3 โดยทำการทดสอบในส่วนของหน้าที่การทำงานของเครื่องมือวัดว่ามีความถูกต้องตามที่ได้ออกแบบไว้หรือไม่ โดยการนำโครงงาน 4 โครงงานที่มีขนาดไม่น้อยกว่า 1000 บรรทัดมาทำการหาค่าตัววัดในส่วนของการนำกลับมาใช้ใหม่ซึ่งสามารถแสดงรายละเอียดของโครงงานที่ทำการวัดดังตารางที่ 5.1 และ 5.2

ตารางที่ 5.1 แสดงรายละเอียดของโครงงานที่ทำการวัด

ชื่อโครงงาน	คำอธิบาย
Chromosomes [16]	เป็นโครงงานที่ถูกพัฒนาขึ้น เพื่อใช้ในการศึกษาทางชีววิทยาเกี่ยวกับลักษณะการจับคู่ของโครโมโซมตามจำนวนโครโมโซมที่ได้กำหนดไว้
Plankton [17]	เป็นโครงงานที่พัฒนาโดยสถาบัน CAIDA : Cooperative Association for Internet Data Analysis และสถาบัน NLANR : National Laboratory for Applied Network Research โดยมีหน้าที่ในการหาเส้นทางที่ดีที่สุดในการสำเนาข้อมูล เพื่อลดเวลาและลดการใช้ช่องทางในการถ่ายโอนข้อมูลในการสำเนาเอกสารบนอินเตอร์เน็ต
JMetric [18]	เป็นผลงานวิจัยของ School of Information Technology ที่ Swinburne University of Technology โดยมีวัตถุประสงค์ในการนำมาตรวัดเชิงวัตถุมาใช้ในการออกแบบเครื่องมือวัด เพื่อนำค่าที่ได้จากการวัดมาทำการวิเคราะห์ต่อไป
JUnit [19]	เป็นโครงงานที่พัฒนาโดย Erich Gamma และ Kent Beck เพื่อใช้ในการทดสอบระบบงานต่างๆ ในส่วนของการทำ unit testing และ integration testing

ตารางที่ 5.2 แสดงจำนวนของคลาส วิธีดำเนินการ และบรรทัด ของโครงงานที่ทำการวัด

ชื่อโครงงาน / จำนวน	Chromosomes	Plankton	JMetric	JUnit
คลาส	39	23	60	23
วิธีดำเนินการ	195	301	591	124
บรรทัด	2296	4736	26258	1850

5.1 การทดสอบหน้าที่การทำงานของเครื่องมือ MTOOP รุ่นที่ 3

ในการทดสอบหน้าที่การทำงานของเครื่องมือ MTOOP รุ่นที่ 3 ผู้วิจัยจะแสดงการทดสอบยูสเคสที่ได้แสดงไว้ในบทที่ 3 เฉพาะที่มีการแก้ไขเพิ่มเติมเท่านั้นซึ่งสามารถแสดงการทดสอบดังต่อไปนี้

5.1.1 การทดสอบสำหรับยูสเคส Parser Generator

จากตารางที่ 5.3 เป็นการทดสอบยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานต้องการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่เครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 5.3 แสดงขั้นตอนการทดสอบการทำงานของยูสเคส Add Java Source Filed

Test Case Name	Parser Generator
Entry condition	1. “DoubleChromosome.java”
Flow of events	2. ระบบสามารถเรียกคลาส JavaParser เพื่อเริ่มต้นทำงานในส่วนของการสร้างตัวพาร์เซอร์ 3. คลาส JavaParser สามารถเรียก Token() สำหรับทำการสร้างสายอักขระของข้อมูลได้ 4. เมื่อได้สายอักขระของข้อมูลแล้ว คลาส JavaParser สามารถเรียก ASCII_UCodeESC_CharStream() เพื่อทำการตรวจสอบรหัสแอสกีของข้อมูลได้ 5. หลังจากนั้นสามารถเรียก JavaParserTokenManager() เพื่อทำการจัดการพาร์เซอร์ได้ 6. ได้โหนดของต้นไม้เอเอสทีของรหัสโปรแกรมที่นำมาตรวจสอบ
Exit condition	7. “DoubleChromosome.java” ถูกตรวจสอบหมดแล้ว

5.1.2 **การทดสอบสำหรับยูสเคส** Compute Metrics

จากตารางที่ 5.4 เป็นการทดสอบยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อเครื่องมือ MTOOP รุ่นที่ 3 เริ่มนำโหนดของต้นไม้เอเอสทีมาทำการคำนวณเพื่อหาค่าวัดสำหรับการนำกลับมาใช้ใหม่

ตารางที่ 5.4 แสดงขั้นตอนการทดสอบการทำงานของยูสเคส Compute Metrics

Test Case Name	Compute Metrics
Entry condition	1. โหนดต้นไม้เอเอสที ของ “DoubleChromosome.java”

Flow of events	2. ระบบนำโหนดต้นไม้เอเอสที ของ “DoubleChromosome.java” มาทำการหาค่าสำหรับการนำกลับมาใช้ใหม่ ตามที่ได้กำหนดไว้ในเครื่องมือ MTOOP ซึ่งค่าวัดที่คำนวณได้มีดังนี้
	2.1 Cyclomatic Complexity (CC)
	public void copyChromosome(Chromosome c) { // copy this to c
	DoubleChromosome dc = (DoubleChromosome) c;
	dc.fitness = this.fitness; // these three
	dc.rfitness = this.rfitness; // really belong
	dc.cfitness = this.cfitness; // in superclass
	for (int i = 0; i < chromosomeLength; i++) {
	dc.gene[i] = this.gene[i];
	}
	}
	จากวิธีดำเนินการข้างต้นสามารถวัดค่า CC ได้ 2
	2.2 Lines Of Code (LOC) จากวิธีดำเนินการใน 2.1 สามารถวัดค่า LOC ได้ 8
	2.3 Comment Percentage (CP) จากวิธีดำเนินการใน 2.1 สามารถวัดค่า CP ได้ 4
	2.4 Method Inheritance Factor (MIF) วัดได้ 55.0 %
	2.5 Attribute Inheritance Factor (AIF) วัดได้ 66.67 %
	2.6 Method Hiding Factor (MHF) วัดได้ 70.37

ตารางที่ 5.4 (ต่อ)แสดงขั้นตอนการทดสอบการทำงานของยูสเคส Compute Metrics

Flow of events	2.7 Attribute Hiding Factor (AHF) วัดได้ 100 % 2.8 Coupling Factor (COF) วัดได้ 11.47% 2.9 Polymorphism Factor (PF) วัดได้ 41.55 % 2.10 Reuse Factor (RF) วัดได้ 100 % 2.11 Relative Degree of Code Saving (RDCS) วัดได้ 36% 2.12 Reuse Size and Frequency (R_{sf}) วัดได้ 43 % 2.13 Reuse Ratio (RR) วัดได้ 100 %
Exit condition	3. โหนดต้นไม้เอเอสที ของ “DoubleChromosome.java” ถูกนำไปทำการหาค่าวัดครบทุกโหนดแล้ว

5.1.3 **การทดสอบสำหรับยูสเคส** View Reusability Metrics By project and methods

จากตารางที่ 5.5 เป็นการทดสอบยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ View Reusability จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 5.5 แสดงขั้นตอนการทดสอบการทำงานของยูสเคส View Reusability Metrics By project and methods

Test Case Name	View Reusability Metrics By project and methods
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ View Reusability
Flow of events	2. ระบบนำค่าวัดที่ได้จากการคำนวณของ “DoubleChromosome.java” มาแสดงผลบนเครื่องมือ MTOOP โดยแยกตามโครงงาน และ วิธีดำเนินการ และทำการประเมินค่าที่ได้จากการวัดสำหรับการนำโครงงานนั้นกลับมาใช้ใหม่
Exit condition	3. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

5.1.4 **การทดสอบสำหรับยูสเคส** View Tables

จากตารางที่ 5.6 เป็นการทดสอบยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ View Tables จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 5.6 แสดงขั้นตอนการทดสอบการทำงานของยูสเคส View Tables

Test Case Name	View Tables
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ View Tables
Flow of events	2. ระบบนำค่าวัดที่คำนวณของ “DoubleChromosome.java” มาแสดงผลในรูปแบบของตาราง
Exit condition	3. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

5.1.5 **การทดสอบสำหรับยูสเคส** Save metrics to DB

จากตารางที่ 5.7 เป็นการทดสอบยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ Save DB จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 5.7 แสดงขั้นตอนการทดสอบการทำงานของยูสเคส Save metrics to DB

Test Case Name	Save metrics to DB
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ Save DB
Flow of events	2. ผู้ใช้งานตั้งชื่อโครงงาน “DoubleChromosome” เพื่อจัดเก็บลงฐานข้อมูลได้ 3. ผู้ใช้งานจัดเก็บค่าวัดต่างๆ ที่ได้จากคำนวณลงฐานข้อมูลได้
Exit condition	4. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

5.1.6 **การทดสอบสำหรับยูสเคส** Load metrics from DB

จากตารางที่ 5.8 เป็นการทดสอบยูสเคสที่แสดงลำดับของเหตุการณ์ที่เกิดขึ้นเมื่อผู้ใช้งานเลือกเมนูในหัวข้อ Load DB จากเครื่องมือ MTOOP รุ่นที่ 3

ตารางที่ 5.8 แสดงขั้นตอนการทดสอบการทำงานของยูสเคส Load metrics from DB

Test Case Name	Load metrics from DB
Entry condition	1. ผู้ใช้งานเลือกเมนูในหัวข้อ Load DB
Flow of events	2. ผู้ใช้งานเลือกชื่อโครงงาน “Chromosome” เพื่อนำรายละเอียดมาแสดงผล 3. ระบบนำค่าจากฐานข้อมูลมาแสดงผลบนเครื่องมือ MTOOP
Exit condition	4. ผู้ใช้งานเลือกทำรายการในหัวข้อเมนูอื่น

5.2 ตัวอย่างการทดสอบมาตรวัดที่นำมาใช้สำหรับการนำกลับมาใช้ใหม่บนเครื่องมือ MTOOP รุ่นที่ 3

เป็นการแสดงการทดสอบการทำงานของเครื่องมือ MTOOP รุ่นที่ 3 กับการทดสอบการหาค่าวัดด้วยมือโดยการนำรหัสโปรแกรมจากโครงงาน JMetric ได้แก่คลาส SUIController (รูปที่ 5.1) และคลาส UIController (รูปที่ 5.2) ซึ่งเป็นคลาสที่มีการถ่ายทอดคุณสมบัติมาจากคลาส SUIController มาทำการทดสอบเพื่อดูผลการทำงานของมาตรวัดในส่วนของโครงงาน สาเหตุที่ผู้วิจัยเลือกคลาสทั้ง 2 มาทำการทดสอบเนื่องจากคลาสทั้ง 2 มีจำนวนบรรทัดที่ไม่มากจนเกินไปสามารถตรวจสอบความถูกต้องด้วยมือได้ และจากความสัมพันธ์ของคลาสทั้ง 2 ผู้วิจัยสามารถนำไปใช้ในการทดสอบมาตรวัดสำหรับการนำกลับมาใช้ใหม่ที่มีอยู่บนเครื่องมือ MTOOP รุ่นที่ 3 ได้ครบทุกมาตรวัด

```
package controller;
import collection.Project;
import ui.JSMainFrame;
import java.awt.*;
import java.io.File;
import javax.swing.JFrame;
public class SUIController extends UIController {
   private JSMainFrame theMainFrame;
   public SUIController(Project tp) {
      super(tp);
      theMainFrame = new JSMainFrame(this); }
   public void start() {
      theMainFrame.setVisible(true); }
      public String getFilename(String heading, int fileDialogMode, String setFile) {
      try {
         return getFilename(heading, fileDialogMode, setFile, theMainFrame); }
    catch(Exception _ex) {  return null; } }
```

รูปที่ 5.1 แสดงรหัสโปรแกรมของคลาส SUIController จากโครงงาน JMetric

```
public String getFilename(String heading, int fileDialogMode, String setFile, JFrame frm)
throws Exception {
        FileDialog fileDialog = new FileDialog(frm);
        fileDialog.setMode(fileDialogMode);
        fileDialog.setTitle(heading);
        fileDialog.setDirectory(getTheProject().getWorkingDirectory().getPath());
        fileDialog.setFile(setFile);
        fileDialog.setVisible(true);
        String dir = fileDialog.getDirectory();
       if(dir == null)
           dir = getTheProject().getWorkingDirectory().getPath();
        if(!dir.endsWith(File.separator))
           dir = dir + File.separator;
        String filename = dir + fileDialog.getFile();
        if(fileDialog.getFile() == null)
           throw new Exception();
        else
           return filename; } }
```

รูปที่ 5.1 (ต่อ) แสดงรหัสโปรแกรมของคลาส SUIController จากโครงงาน JMetric

```
package controller;
import collection.*;
import java.awt.FileDialog;
import java.io.BufferedReader;
import java.io.File;
public abstract class UIController
   implements IUIController {
   private IProjectController theProject;
```

รูปที่ 5.2 แสดงรหัสโปรแกรมของคลาส UIController จากโครงงาน JMetric

```
public abstract String getFilename(String s, int i, String s1) throws Exception;
public abstract void start();
 public UIController(IProjectController the) {
        theProject = the; }
  public void clear() {
        theProject.clear(); }
  public void setTheProject(IProjectController the) {
        theProject = the; }
   public void setWorkingDirectory(String to) {
        theProject.setWorkingDirectory(to); }
    public void addDirectory(String dir) {
        theProject.addDirectory(dir); }
    public void addFile() {
        String filename = null;
        try {
            filename = getFilename("Add Java File", 0, "*.java"); }
        catch(Exception _ex) { }
        if(filename != null)
            addFile(filename); }
    public void addFile(String dir, String file) {
        theProject.addFile(dir, file); }
    public void addFile(String filename) {
        theProject.addFile(filename); }
    public Project getTheProject() {
        return theProject.getProject(); }
    public void systemExit() {
        System.exit(0);
    }
```

รูปที่ 5.2 (ต่อ) แสดงรหัสโปรแกรมของคลาส UIController จากโครงงาน JMetric

```
    public void newProject() {
        theProject.makeNew(); }
    public void addRecursive(String filename) {
        getTheProject().addRecursive(filename);  }
    public void addRecursive(String directory, String filename) {
        getTheProject().addRecursive(directory, filename); }
    public void addRecursive() {
        addRecursive("*.java"); }
    public void addProjectChangeListener(ProjectChangeListener listen) {
        theProject.addProjectChangeListener(listen); }
}
```

รูปที่ 5.2 (ต่อ) แสดงรหัสโปรแกรมของคลาส UIController จากโครงงาน JMetric

จากรหัสโปรแกรมในรูปที่ 5.1 และรูปที่ 5.2 สามารถแสดงผลการทดสอบการคำนวณหาค่าของมาตรวัดในระดับโครงงานที่ใช้บนเครื่องมือ MTOOP รุ่นที่ 3 ได้ดังตารางที่ 5.9 และสามารถดูวิธีการคำนวณสำหรับมาตรวัดต่างๆ ได้ในภาคผนวก ค

ตารางที่ 5.9 แสดงผลค่าที่คำนวณได้จากการทดสอบของเครื่องมือ MTOOP รุ่นที่ 3 กับการคำนวณค่าวัดด้วยมือสำหรับคลาส SUIController และคลาส UIController จากโครงงาน JMetric

มาตรวัด	คำอธิบาย	ค่าที่คำนวณได้ของเครื่องมือ MTOOP รุ่นที่ 3	ค่าที่คำนวณได้ (ด้วยมือ)
MIF	เป็นการหาผลรวม ของวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจาก คลาส UIController	40	40
AIF	เป็นการหาผลรวม ของลักษณะประจำที่มีการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจาก คลาส UIController	33.33	33.33

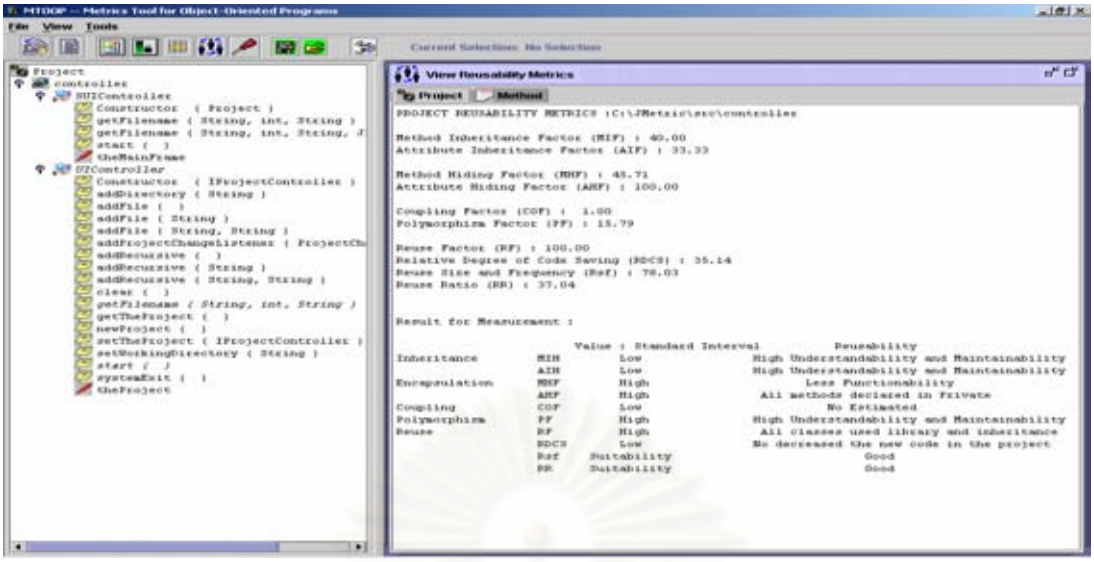
ตารางที่ 5.9 (ต่อ) แสดงผลค่าที่คำนวณได้จากการทดสอบของเครื่องมือ MTOOP รุ่นที่ 3 กับการคำนวณค่าวัดด้วยมือสำหรับคลาส SUIController และคลาส UIController จากโครงงาน JMetric

มาตรวัด	คำอธิบาย	ค่าที่คำนวณได้ของเครื่องมือ MTOOP รุ่นที่ 3	ค่าที่คำนวณได้ (ด้วยมือ)
MHF	เป็นการหาผลรวมของวิธีดำเนินการทั้งหมด ที่เป็นการซ่อนข้อมูลของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController	45.71	45.71
AHF	เป็นการหาผลรวมของลักษณะประจำทั้งหมดที่เป็นการซ่อนข้อมูลของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController	100	100
PF	เป็นการหาผลรวมของ การวัดความสัมพันธ์ระหว่างวัตถุที่มาจากต่างคลาส โดยมีการตอบสนองต่อการกระทำบนพื้นฐานเดียวกัน ด้วยรูปแบบที่แตกต่างกันของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจาก คลาส UIController	15.79	15.79
COF	เป็นการหาผลรวม ของการวัดการเข้าคู่กันระหว่างคลาส ของคลาส SUIController และคลาส UIController แต่จะไม่รวมถึงการเข้าคู่กันในลักษณะที่เป็นการถ่ายทอดคุณสมบัติ	1	1

ตารางที่ 5.9 (ต่อ) แสดงผลค่าที่คำนวณได้จากการทดสอบของเครื่องมือ MTOOP รุ่นที่ 3 กับการคำนวณค่าวัดด้วยมือสำหรับคลาส SUIController และคลาส UIController จากโครงงาน JMetric

มาตรวัด	คำอธิบาย	ค่าที่คำนวณได้ของเครื่องมือ MTOOP รุ่นที่ 3	ค่าที่คำนวณได้ (ด้วยมือ)
RF	เป็นการหาผลรวม ของการวัดการนำกลับมาใช้ใหม่ ซึ่งจะพิจารณาจากการนำส่วนประกอบจากไลบรารี่ที่มีอยู่มาใช้ใหม่ หรือมีการนำกลับมาใช้ใหม่ในลักษณะของการถ่ายทอดคุณสมบัติ ของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController	100	100
RDCS	เป็นการหาผลรวม ของการวัดระดับความสัมพันธ์ของวิธีดำเนินการที่มีการนำกลับมาใช้ใหม่ ในลักษณะของการลดการเขียนรหัสโปรแกรม เนื่องจากมีการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController	35.14	35.14
R_{sf}	เป็นการหาผลรวม ของการวัดขนาดของรหัสโปรแกรมและความถี่ของรหัสโปรแกรมที่มีการนำกลับมาใช้ใหม่ของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController	78.03	78.03
RR	เป็นการหาผลรวม ของการหาอัตราส่วนของการนำองค์ประกอบของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจาก คลาส UIController กลับมาใช้ใหม่	37.04	37.04

เมื่อนำผลที่ได้จากการคำนวณไปเทียบกับผลที่ได้จากเครื่องมือ MTOOP รุ่นที่ 3 จะได้ผลการคำนวณตรงกัน ดังตารางที่ 5.9 และรูปที่ 5.3



รูปที่ 5.3 แสดงผลที่ได้จากการทำงานของเครื่องมือ MTOOP รุ่นที่ 3 สำหรับคลาส SUIController และคลาส UIController จากโครงงาน JMetric

5.3 สรุปผลการทดสอบ

ตารางที่ 5.10 เป็นการแสดงผลของการทดสอบเครื่องมือ MTOOP รุ่นที่ 3 กับโครงงานต่างๆ ทั้ง 4 โครงงาน (รายละเอียดดังตารางที่ 5.1) และสามารถสรุปได้ว่าเครื่องมือนี้สามารถหาค่าวัดในส่วนของการนำกลับมาใช้ใหม่ได้ และสามารถนำค่าวัดที่คำนวณได้มาทำการเปรียบเทียบกับเกณฑ์ที่ได้กำหนดไว้ (ภาคผนวก ข) เพื่อประเมินค่าความเหมาะสมสำหรับการนำรหัสโปรแกรมภาษาจาวาจากโครงงานที่ทำการวัดกลับมาใช้ใหม่

ตารางที่ 5.10 แสดงผลการทดสอบเครื่องมือ MTOOP รุ่นที่ 3 กับโครงงานทั้ง 4 โครงงานที่นำมาทดสอบ

เกณฑ์ที่กำหนด	66.2% - 80.8%	52.4% - 60%	10.4% - 28.7%	70.2% - 89.3%	3.9% - 17.7%	3.5% - 9.6%	> 43%	เข้าใกล้ 100%	> 66%	31% - 64%
มาตรวัด / โครงงาน	MIF	AIF	MHF	AHF	COF	PF	RF	RDCS	R_{sf}	RR
Chromosomes	67.72%	47.47%	78.66%	96.20%	11.47%	41.55%	100%	21%	80%	6.27%
Plankton	55.41%	40.76%	61.28%	77.99%	58.50%	8.09%	100%	38%	91%	9%
JMetric	67.09%	46.31%	73.42%	88.93%	15.73%	7.52%	100%	56%	81%	6%
JUnit	69.53%	69.44%	78.77%	96.15%	19.57%	32.12%	100%	37%	73%	28%

จากตารางที่ 5.10 เมื่อพิจารณาค่าที่ได้จากการวัดสำหรับโครงงาน Chromosomes จะเห็นได้ว่าค่า MIF ที่ได้จากการวัดอยู่ในเกณฑ์ที่กำหนดเนื่องจากจำนวนของวิธีดำเนินการที่สร้างขึ้นใหม่มีจำนวนไม่มากส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้น และเมื่อพิจารณาค่า AIF จะเห็นได้ว่าค่าที่ได้จากการวัดต่ำกว่าเกณฑ์ที่กำหนด เนื่องจากมีการใช้งานของลักษณะประจำจากคลาสที่มีการถ่ายทอดคุณสมบัติมาให้น้อย และมีการกำหนดลักษณะประจำขึ้นมาใหม่เป็นจำนวนมากส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย

เมื่อพิจารณาค่า MHF จะเห็นได้ว่ามีค่าสูงกว่าเกณฑ์ที่กำหนดเนื่องจากมีจำนวนของวิธีดำเนินการที่มีการกำหนดการทำงาน ในลักษณะการเรียกใช้งานได้เฉพาะภายในแต่ละคลาสที่ทำการวัดเท่านั้น หรืออาจกล่าวได้ว่ามีการกำหนดวิธีดำเนินงานในลักษณะการทำงานแบบ private เป็นส่วนใหญ่ซึ่งส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลง และเมื่อพิจารณาค่า AHF จะเห็นได้ว่ามีค่าสูงกว่าเกณฑ์ที่กำหนดเนื่องจากมีการกำหนดค่าของลักษณะประจำในลักษณะการใช้งานแบบ private คือเป็นการเรียกใช้ภายในแต่ละคลาสเท่านั้นซึ่งไม่อาจสรุปได้ว่าการนำกลับมาใช้ใหม่ในส่วนของการห่อหุ้มและการซ่อนข้อมูลของลักษณะประจำ เหมาะแก่การนำกลับมาใช้ใหม่สูงขึ้น แต่ในทางตรงข้ามอาจเกิดจากการที่ผู้พัฒนาโครงงานออกแบบโครงงานผิดพลาดได้

เมื่อพิจารณาค่า COF จะเห็นได้ว่าค่าที่ได้จากการวัดอยู่ในเกณฑ์ที่กำหนดเนื่องจากมีการเรียกใช้วิธีดำเนินการหรือมีการเข้าคู่กันระหว่างคลาสไม่มากทำให้มีความซับซ้อนน้อยส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้น และเมื่อพิจารณาค่า PF จะเห็นได้ว่าค่าที่ได้จากการวัดสูงกว่าเกณฑ์ที่กำหนด เนื่องจากมีจำนวนของวิธีดำเนินการในลักษณะของเมทธอดโอเวอร์โหลดดิ้งและเมทธอดโอเวอร์ไรดิ้งอยู่จำนวนมาก ทำให้โครงงานมีความซับซ้อนสูงส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย

เมื่อพิจารณาค่า RF จะเห็นได้ว่าโครงงานที่ทำการวัดมีค่าเท่ากับ 100% เนื่องจากมีการเรียกใช้การทำงานจากไลบรารีหรือจากการถ่ายทอดคุณสมบัติ จึงทำให้ช่วยลดปริมาณของวิธีดำเนินการภายในคลาสที่ทำการวัด แต่จะต้องใช้เวลาในการทดสอบโครงงานมากทำให้เสียเวลาในการพัฒนาสูง เนื่องจากคลาสส่วนใหญ่ภายในโครงงานที่ทำการวัดมีการเรียกใช้วิธีดำเนินการจากไลบรารี หรือมีการเรียกใช้วิธีดำเนินการที่เกิดจากการถ่ายทอดคุณสมบัติสูงจึงทำให้เกิดความซับซ้อนภายในโครงงาน แต่จะส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้น เมื่อพิจารณาค่า RDCS จะเห็นได้ว่ามีค่าต่ำกว่าเกณฑ์ที่กำหนดเนื่องจากมีระดับความสัมพันธ์ของวิธีดำเนินการที่มีการนำกลับมาใช้ใหม่น้อย ส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย เมื่อพิจารณาค่า R_{sf} จะเห็นได้ว่ามีค่าอยู่ในเกณฑ์ที่กำหนดเนื่องจากมีการนำรหัสโปรแกรมกลับมาใช้ใหม่และมีความถี่ในการเรียกใช้งานสูง ส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้น

และเมื่อพิจารณาค่า RR จะเห็นได้ว่ามีค่าต่ำกว่าเกณฑ์ที่กำหนดเนื่องจากมีอัตราการเปลี่ยนแปลงขององค์ประกอบคือมีการเพิ่มเติม การปรับปรุง หรือการลบวิธีดำเนินการ เนื่องจากมีการทำงานแบบเฉพาะเจาะจงภายในโครงงานสูงซึ่งจะทำให้การนำกลับมาใช้ใหม่น้อยลงด้วย

เมื่อพิจารณาค่าที่ได้จากการวัดสำหรับโครงงาน Plankton จะเห็นได้ว่าค่า MIF ที่ได้จากการวัดต่ำกว่าเกณฑ์ที่กำหนดเนื่องจากมีการสร้างวิธีดำเนินการขึ้นใหม่เป็นจำนวนมาก ซึ่งส่งผลให้มีการนำกลับมาใช้ใหม่ของโครงงานน้อยลง และเมื่อพิจารณาค่า COF จะเห็นได้ว่ามีค่าสูงกว่าเกณฑ์ที่กำหนด เนื่องจากมีการเรียกใช้วิธีดำเนินการหรือมีการเข้าคู่กันระหว่างคลาสสูงทำให้โครงงานมีความซับซ้อนส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานน้อยลงด้วย

เมื่อพิจารณาค่าที่ได้จากการวัดสำหรับโครงงาน JMetric จะเห็นได้ว่าค่า AHF ที่ได้จากการวัดอยู่ในเกณฑ์ที่กำหนด เนื่องจากมีการกำหนดค่าของลักษณะประจำในลักษณะการใช้งานแบบ private คือเป็นการเรียกใช้ภายในแต่ละคลาสเท่านั้นจึงมีผลกระทบต่อการทำงานของคลาสอื่นน้อยส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้น และเมื่อพิจารณาค่า PF จะเห็นได้ว่ามีค่าอยู่ในเกณฑ์ที่กำหนด เนื่องจากมีจำนวนของวิธีดำเนินการในลักษณะของเมทธอดโอเวอร์โหลดดิ้งและเมทธอดโอเวอร์ไรดิ้งอยู่น้อย ทำให้โครงงานที่ทำการวัดมีความซับซ้อนไม่สูงส่งผลให้การนำกลับมาใช้ใหม่ของโครงงานสูงขึ้นด้วย

และเมื่อพิจารณาค่าที่ได้จากการวัดสำหรับโครงงาน JUnit จะเห็นได้ว่าค่า AIF ที่ได้จากการวัดสูงว่าเกณฑ์ที่กำหนดซึ่งอาจจะมีสาเหตุมาจากการออกแบบที่ผิดพลาดได้

บทที่ 6

บทสรุปและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงบทสรุปและข้อเสนอแนะเกี่ยวกับงานวิจัย ที่ได้จากการออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่สำหรับซอฟต์แวร์ภาษาจาวาดังรายละเอียดต่อไปนี้

6.1 บทสรุป

งานวิจัยนี้เป็นการออกแบบและพัฒนาเครื่องมือวัดการนำกลับมาใช้ใหม่ ที่มีโปรแกรมต้นฉบับเป็นภาษาจาวา ซึ่งสูตรการคำนวณและเกณฑ์สำหรับการวัดที่ใช้ในงานวิจัยนี้ได้มาจากผลงานวิจัยต่างๆ ที่เป็นที่เผยแพร่และเป็นที่ยอมรับในวงกว้างของการออกแบบและพัฒนาโปรแกรมเชิงวัตถุในปัจจุบัน

ค่าของตัววัดที่ใช้ในงานวิจัยนี้ ช่วยให้ผู้ออกแบบและพัฒนาซอฟต์แวร์ทราบถึงคุณสมบัติในส่วนของความซับซ้อนภายในวิธีดำเนินการ ขนาดของโปรแกรม อัตราส่วนของข้อความอธิบาย การถ่ายทอดคุณสมบัติ การเข้าคู่กัน การห่อหุ้ม การพ้องรูป และการนำกลับมาใช้ใหม่ของซอฟต์แวร์จากค่าที่ได้จากการวัด ซึ่งผู้ออกแบบและพัฒนาซอฟต์แวร์สามารถใช้เครื่องมือ MTOOP รุ่นที่ 3 นี้เพื่อประเมินแนวโน้มสำหรับการนำโปรแกรมต้นฉบับภาษาจาวากลับมาใช้งานใหม่ จากค่าของปัจจัยต่างๆ ที่ได้จากการวัด

ส่วนในการวิเคราะห์และออกแบบระบบในงานวิจัยนี้ ผู้วิจัยได้ใช้ยูเอ็มแอลซึ่งเป็นโมเดลที่จะทำให้สามารถระบุถึงโครงสร้างและพฤติกรรมของระบบงานที่พัฒนาได้ มาเป็นเครื่องมือในการวิเคราะห์และออกแบบเพื่อใช้ในการพัฒนาเครื่องมือ MTOOP รุ่นที่ 3 นี้โดยได้ทำการแบ่งชุดของคลาสออกเป็น 6 แพ็คเกจด้วยกันคือ แพ็คเกจส่วนของการนำโปรแกรมต้นฉบับภาษาจาวาเข้าสู่ระบบ แพ็คเกจส่วนของการควบคุม แพ็คเกจตัวแปลภาษา แพ็คเกจการคำนวณค่าตัววัด แพ็คเกจส่วนของการติดต่อผู้ใช้งาน และแพ็คเกจการจัดรูปแบบการแสดงผลค่าตัววัด โดยที่ผู้วิจัยได้พัฒนาเครื่องมือตามที่ได้ออกแบบ และทำการทดสอบความสามารถของเครื่องมือโดยการนำระบบงานต่างๆ 4 ระบบงาน (รายละเอียดดังตารางที่ 5.1) มาทำการทดสอบหาค่าตัววัดในส่วนของการนำกลับมาใช้ใหม่ซึ่งประกอบด้วย การวัดความซับซ้อน การวัดขนาด การหาอัตราส่วนของข้อความ การวัดการถ่ายทอดคุณสมบัติซึ่งได้แก่ MIF และ AIF การวัดการห่อหุ้มและการซ่อนข้อมูลซึ่งได้แก่ MHF และ AHF การวัดการพ้องรูปซึ่งได้แก่ PF การวัดการเข้าคู่กันซึ่งได้แก่ CF และการวัดการนำ

กลับมาใช้ใหม่ซึ่งได้แก่ RF R_{sf} RR และ $RDCS$ ซึ่งก็สามารถทำการหาค่าวัดได้ ตรงตามที่ได้ออกแบบไว้

ผู้วิจัยสามารถสรุปประโยชน์ของเครื่องวัด ได้ดังต่อไปนี้

6.1.1 สามารถใช้เครื่องมือวัดเพื่อดูค่าที่ได้จากการวัดในส่วนของการนำกลับมาใช้ใหม่แบบอัตโนมัติได้

6.1.2 สามารถนำค่าที่ได้จากการวัดมาช่วยในการตัดสินใจ ในการนำส่วนประกอบสำหรับโปรแกรมภาษาจาวากลับมาใช้ใหม่ในโครงงานต่อไป

6.1.3 สามารถนำค่าที่ได้จากการวัด มาทำการปรับปรุงส่วนประกอบสำหรับโปรแกรมภาษา จาวาให้ดีขึ้น เพื่อที่จะสามารถนำส่วนประกอบนั้นกลับมาใช้ใหม่ได้อีกครั้งในโครงงานต่อๆ ไป

6.2 ข้อเสนอแนะ

ผู้วิจัยมีข้อเสนอแนะเกี่ยวกับงานวิจัย ดังต่อไปนี้

6.2.1 เนื่องจากในงานวิจัยนี้ ผู้วิจัยทำการวัดค่าจากโปรแกรมต้นฉบับภาษาจาวาเท่านั้น ดังนั้นผู้ที่สนใจอาจนำมาตรวัดที่ใช้ในงานวิจัยนี้ไปใช้ทำการวัดโปรแกรมที่เขียนในแบบโปรแกรมภาษาเชิงวัตถุอื่นๆ ได้

6.2.2 ในอนาคตอาจจะมีการออกแบบมาตรวัดสำหรับการวัดโปรแกรมเชิงวัตถุ ในส่วนของการวัดการนำกลับมาใช้ใหม่เพิ่มมากขึ้น ดังนั้นผู้ที่สนใจอาจทำการเพิ่มมาตรวัดต่างๆ ที่ได้มาตรฐานและเป็นที่นิยมใช้โดยกว้างขวางลงในงานวิจัย เพื่อที่จะทำให้ผลที่ได้จากการวัดมีความน่าเชื่อถือยิ่งขึ้น

รายการอ้างอิง

1. Fernando Brito Abreu, and Rogerio Carapuca. 4th International Conference on Software Quality. Object-Oriented Software Engineering: Measuring and Controlling the Development Process, 1994.
2. ปวีณ ศรีทอง.สารเนคเทค พฤศจิกายน-ธันวาคม. การวัด Reusability ในการพัฒนาซอฟต์แวร์เชิงรุก, 2543.
3. ธวัชชัย งามสันติวงศ์. การเขียนโปรแกรมเชิงวัตถุ. พิมพ์ปรับปรุงครั้งที่ 1. กรุงเทพมหานคร: สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ, 2545.
4. Rachel Harrison, Steve J. Counsell, and Reuben V. Nithi. IEEE Transactions on Software Engineering Vol 24 No. 6. An Evaluation of the MOOD Set of Object-Oriented Software Metrics, 1998.
5. F.Alonso, J.L. Fuerts, C. Montes, and R.J. Navajo. IEEE Transactions on Software Engineering. A Quality Model : How to Improve the Object-Oriented Software Process, 1998.
6. Fernando Brito Abreu, and Walcelio Melo. IEEE Transactions on Software Engineering. Evaluating the Impact of Object-Oriented Design on Software Quality, 1996.
7. Fernando Brito Abreu, Miguel Goulao, and Rita Esteves. Conference on Software Quality Austin Texas 23 to 26 October. Toward the Design Quality Evaluation of Object-Oriented Software Systems, 1995.
8. Roger S. Pressman. Software Engineering A Practitioner's Approach. Fifth Edition. Singrapore: McGraw-Hill International Edition, 2001.
9. Linda H. Rosenberg. NASA. Applying and Interpreting Object-Oriented Metrics, 1998.
10. Lawrence E. Hyatt, and Linda H. Rosenberg. NASA. Software Quality Metrics for Object- Oriented System Environments,1995.
11. Byung-Kyoo Kang, and James M. Bieman. IEEE-CS Fourth Int. Software Metrics Symposium (Metrics'97). Inheritance Tree Shapes and Reuse, pp. 47-52. 1995.

12. Prem Devanbu, Sakke Karstu, Walcelio Melo, and William Thomas. Proceeding 18th International IEEE Conference on Software Engineering. Analytical and Empirical Evaluation of Software Reuse Metrics, 1996.
13. สมหวัง แซ่ตั้ง. การออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์สำหรับโปรแกรมเชิงวัตถุ. วิทยานิพนธ์ปริญญามหาบัณฑิต, ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2543.
14. วัฒนชัย รอดกำเนิด. การออกแบบและพัฒนาเครื่องมือวัดปัจจัยของความซับซ้อนของโปรแกรมเชิงวัตถุภาษาจาวา. วิทยานิพนธ์ปริญญามหาบัณฑิต, ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2544.
15. Alain Abran, and Marcela Maya. Journal of Software Maintenance : Research and Practice. Measurement of Functional Reuse, 1998.
16. Planet Source Code, Chromosomes [Online]. 1997. Available from: http://www.planet-source-code.com/vb/scripts/BrowseAllCategories.asp?lngWID=2 [2004, January 15]
17. Cooperative Association for Internet Data Analysis, Plankton [Online]. 2003. Available from: http://www.caida.org/tools/visualization/plankton/source [2004, January 15]
18. Swinburn University, JMetric [Online]. 2000. Available from: http://www.it.swin.edu.au/projects/jmetric/products/jmetric [2004, January 15]
19. Object Mentor, JUnit [Online]. 2001. Available from: http://www.junit.org/news/article/index.htm#JUnitHowTo [2004, January 15]

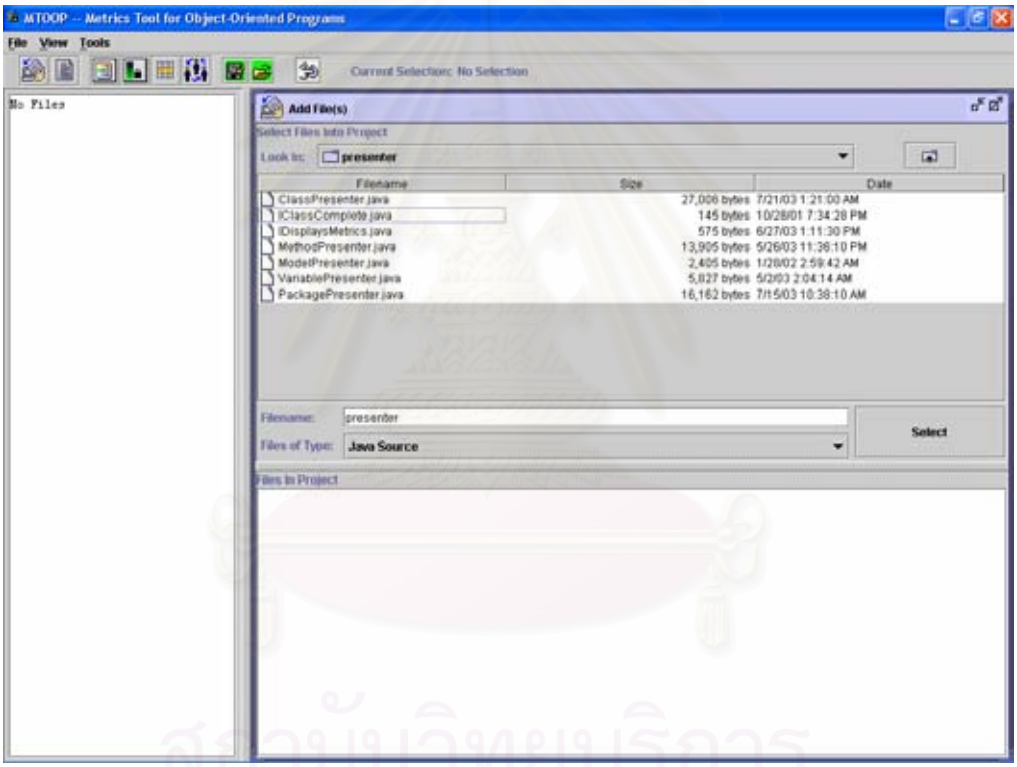
ภาคผนวก

ภาคผนวก ก

การใช้งานเครื่องมือ MTOOP รุ่นที่ 3

ในที่นี้จะอธิบายถึงการใช้งานเครื่องมือ MTOOP รุ่นที่ 3 ซึ่งแบ่งออกได้เป็น 3 ส่วนด้วยกัน คือ การอ่านโปรแกรมต้นฉบับเข้าสู่ระบบ การดูค่าตัววัด และการเก็บและเรียกดูค่าตัววัดจากฐานข้อมูลดังรายละเอียดต่อไปนี้

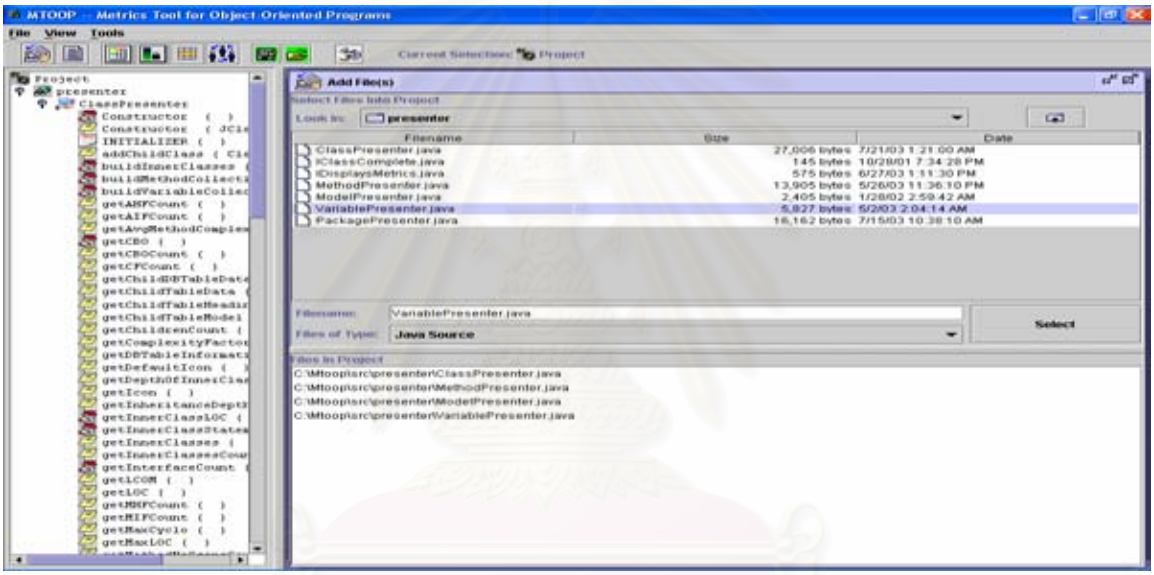
การอ่านโปรแกรมต้นฉบับเข้าสู่ระบบ



รูปที่ ก.1 แสดงหน้าจอเมื่อผู้ใช้งานเริ่มเข้าสู่ระบบ

จากรูปที่ ก.1 เป็นการแสดงหน้าจอเริ่มต้นเมื่อผู้ใช้งานเข้าสู่ระบบ ซึ่งจากหน้าจอนี้ผู้ใช้งานสามารถที่จะทำการเลือกโปรแกรมต้นฉบับเข้าสู่ระบบได้ โดยทำการเลือก  (File / Add Files) เพื่อเลือกโปรแกรมต้นฉบับเข้าสู่ระบบหลังจากนั้นระบบจะทำการแสดงชื่อของโครงงาน แพ็คเกจ คลาส วิธีดำเนินการ และตัวแปรอินสแตนท์ เพื่อให้ผู้ใช้งานสามารถเลือกดูค่าตัววัดในรูปแบบต่างๆ ได้ ดังแสดงในรูปที่ ก.2 และในกรณีที่ผู้ใช้งานต้องการที่จะทำการวัดโปรแกรมต้นฉบับอื่นซึ่ง

อยู่ในโครงงานเดียวกันผู้ใช้งานสามารถทำได้โดยเลือก  (File / Add Files) เพื่อนำโปรแกรมต้นฉบับเข้าสู่ระบบเพื่อทำการวัดค่าอีกครั้ง หรือทำซ้ำขั้นตอนนี้จนกว่าจะนำโปรแกรมต้นฉบับทั้งหมดเข้าสู่ระบบเป็นที่เรียบร้อย และในกรณีที่ผู้ใช้งานต้องการที่จะทำการวัดค่าโครงงานใหม่ ผู้ใช้งานสามารถทำได้โดยทำการเลือก  (File / New Project) เพื่อเป็นการเริ่มต้นการวัดค่าของโครงงานใหม่หลังจากนั้นผู้ใช้งานทำการเลือก  (File / Add Files) เพื่อทำการนำโปรแกรมต้นฉบับเข้าสู่ระบบเพื่อทำการวัดค่านั่นเอง และในกรณีที่ผู้ใช้งานต้องการออกจากระบบการใช้งานผู้ใช้งานสามารถทำได้โดยการเลือก  (File / Exit) ก็จะเป็นการสิ้นสุดการทำงานของเครื่องมือ MTOOP รุ่นที่ 3



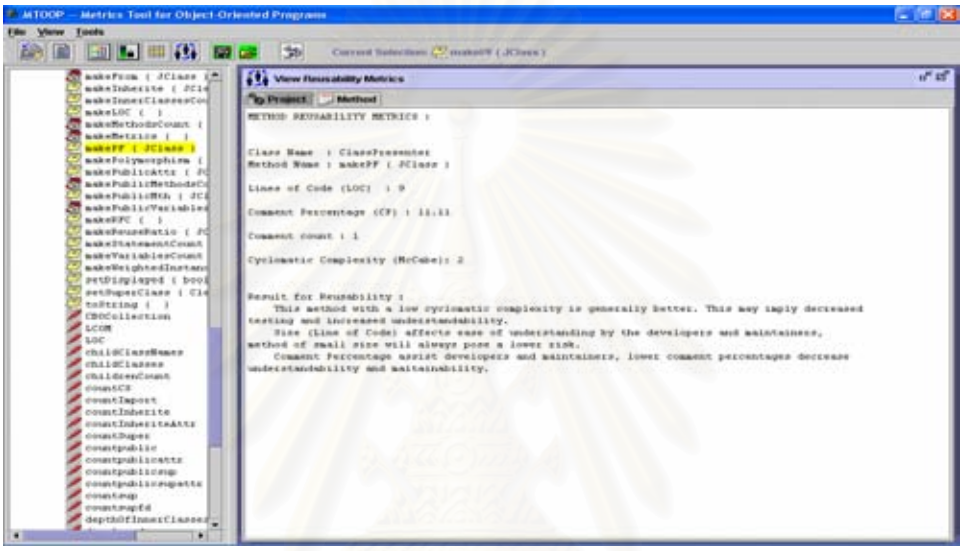
รูปที่ ก.2 แสดงรายละเอียดของหน้าจอหลังจากผู้ใช้งานทำการเลือกโปรแกรมต้นฉบับเข้าสู่ระบบ

การดูค่าตัววัด

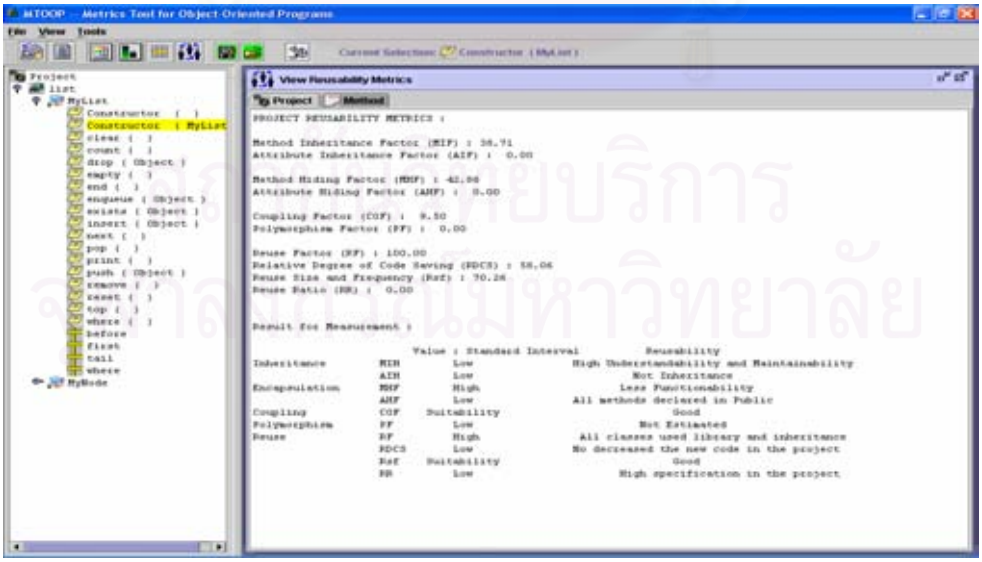
ในที่นี้จะกล่าวถึงเฉพาะการดูค่าตัววัด เพื่อใช้ในการตัดสินใจสำหรับการนำโปรแกรมต้นฉบับภาษาจาวากลับมาใช้ใหม่เท่านั้น

เมื่อผู้ใช้งานต้องการที่จะดูค่าตัววัดสำหรับการนำกลับมาใช้ใหม่ ผู้ใช้งานสามารถทำได้โดยเลือก  (View / Reusability Metrics) ระบบจะนำค่าที่ได้จากการวัดมาแสดงผลโดยแบ่งการแสดงค่าตัววัดออกเป็น 2 ส่วนด้วยกันคือ การดูค่าตัววัดในส่วนของวิธีดำเนินการ และการดูค่าตัววัดในส่วนของโครงงาน ซึ่งในการดูค่าตัววัดในส่วนของวิธีดำเนินการนั้นผู้ใช้งานจะต้องทำการเลือกวิธีดำเนินการที่ต้องการวัดเสียก่อน หลังจากนั้นจึงทำการเลือกดูค่าตัววัดในหัวข้อของ

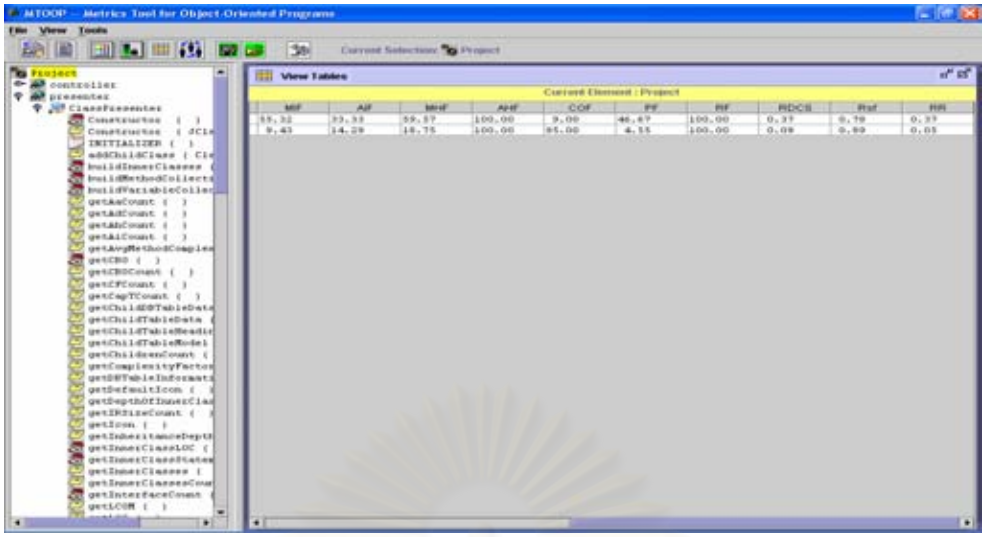
วิธีดำเนินการ ระบบจะทำการนำค่าที่ได้จากการวัดมาแสดงผลบนเครื่องมือวัดดังแสดงในรูปที่ ก.3 ในกรณีที่ผู้ใช้งานต้องการดูค่าตัววัดในส่วนของโครงงาน ก็สามารถทำได้โดยเลือกดูในหัวข้อของโครงงาน ระบบงานก็จะทำการนำค่าที่ได้จากการวัดในส่วนนี้มาแสดงผลบนเครื่องมือวัดดังแสดงในรูปที่ ก.4 และในกรณีที่ผู้ใช้งานต้องการดูค่าตัววัดในรูปแบบของตารางสามารถทำได้โดยผู้ใช้งานทำการเลือก  (View / View Tables) ระบบจะนำค่าวัดที่ได้จากการคำนวณมาแสดงผลในรูปแบบของตารางดังแสดงในรูปที่ ก.5



รูปที่ ก.3 แสดงรายละเอียดการดูค่าตัววัดสำหรับการนำกลับมาใช้ใหม่ ในส่วนของวิธีดำเนินการ



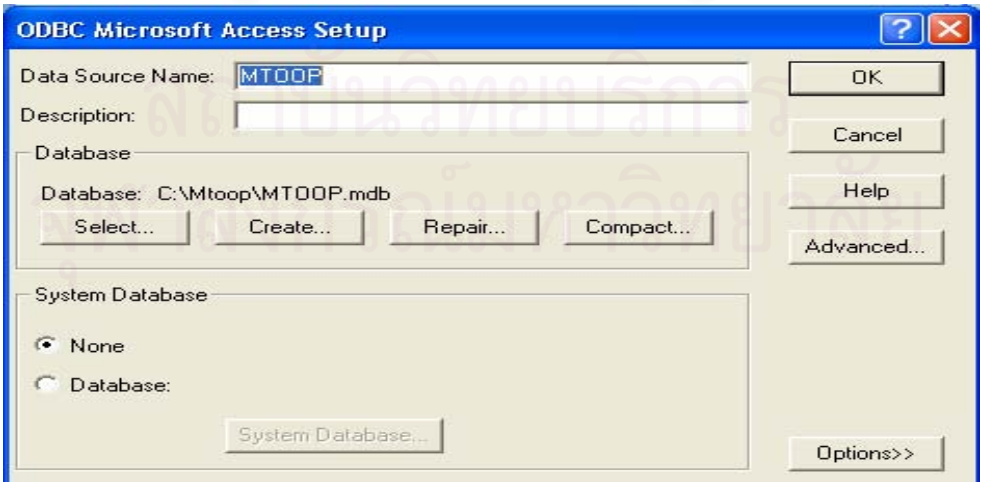
รูปที่ ก.4 แสดงรายละเอียดการดูค่าตัววัดสำหรับการนำกลับมาใช้ใหม่ ในส่วนของโครงงาน



รูปที่ ก.5 แสดงรายละเอียดการดูค่าตัววัดในรูปแบบของตาราง

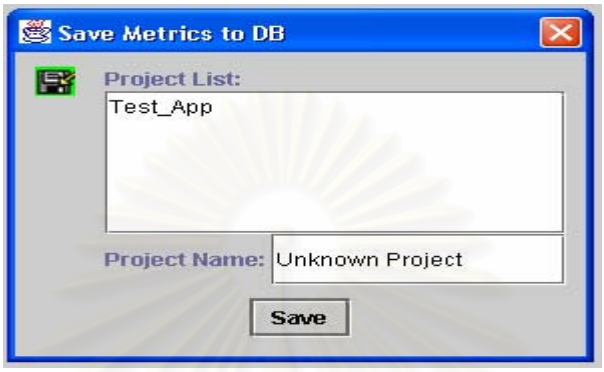
การเก็บและการเรียกดูค่าตัววัดจากฐานข้อมูล

การเก็บค่าตัววัดของโครงงานลงฐานข้อมูล ผู้ใช้งานจะต้องทำการบันทึกข้อมูลลงฐานข้อมูลผ่านเจดีบีซี (JDBC : Java Database Connectivity) ซึ่งจะต้องมีการกำหนดช่องทางการติดต่อเพื่อใช้ในการรับส่งข้อมูลด้วยการใช้โอดีบีซี (ODBC : Open Database Connectivity) โดยใช้ไดรเวอร์ของไมโครซอฟต์แอคเซส (Microsoft Access Driver) และกำหนดดาต้า-ซอร์สเนม (Data Source Name) ชื่อ “MTOOP” และไฟล์ของฐานข้อมูลชื่อ “mtoop.mdb” ในเครื่องมือโอดีบีซีนี้ผู้ใช้งานสามารถทำการเก็บบันทึกค่าหรือร้องขอข้อมูลได้ดังแสดงในรูปที่ ก.6



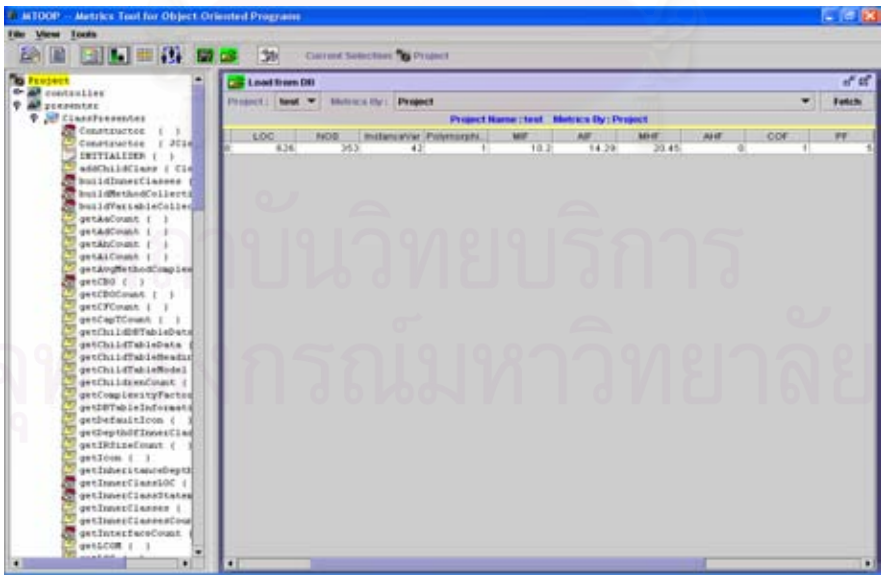
รูปที่ ก.6 การกำหนดค่าดาต้าซอร์สเนมและไฟล์ฐานข้อมูลในเครื่องมือจัดการโอดีบีซี

ผู้ใช้งานสามารถทำการบันทึกข้อมูลลงฐานข้อมูลได้โดยผู้ใช้งานทำการเลือก  (Tools / Save to DB) ระบบจะแสดงหน้าจอดังรูปที่ ก.7 เพื่อให้ผู้ใช้งานตั้งชื่อของโครงงานที่ต้องการจัดเก็บในฐานข้อมูลหลังจากนั้นผู้ใช้งานทำการกดปุ่ม “SAVE” ระบบจะทำการบันทึกรายละเอียดของข้อมูลลงในฐานข้อมูลตามชื่อโครงงานที่ผู้ใช้งานได้การกำหนดไว้



รูปที่ ก.7 แสดงหน้าจอเพื่อใช้ในการจัดเก็บค่าตัววัดลงฐานข้อมูล

และในกรณีที่ผู้ใช้งานต้องการที่จะดูค่าตัววัดจากฐานข้อมูล ผู้ใช้งานสามารถทำได้โดยการเลือก  (Tools / Load from DB) หลังจากนั้นกดปุ่ม “Fetch” ระบบก็จะนำค่าที่จัดเก็บในฐานข้อมูลมาแสดง ดังรูปที่ ก.8



รูปที่ ก.8 แสดงหน้าจอการดูค่าตัววัดที่จัดเก็บในฐานข้อมูล

ภาคผนวก ข

การตีความค่าที่ได้จากการวัด

ตารางที่ ข.1 การตีความค่าที่ได้จากการวัด [1],[4],[7],[8],[10],[11],[12]

Source	METRIC	ค่าที่ควรเป็น
Traditional Measurement	Cyclomatic Complexity (CC)	2-5 ต่อวิธีดำเนินการที่ทำการวัด
	Lines Of Code (LOC)	50-100 lines ต่อวิธีดำเนินการที่ทำการวัด
	Comment Percentage (CP)	20-30 % ต่อวิธีดำเนินการที่ทำการวัด
Object-Oriented Measurement	Method Inheritance Factor (MIF)	66.2-80.8 % ของระบบงานที่พิจารณา (ค่าที่เหมาะสม = 73.5%)
	Attribute Inheritance Factor (AIF)	52.4-60.0 % ของระบบงานที่พิจารณา (ค่าที่เหมาะสม = 56.2%)
	Method Hiding Factor (MHF)	10.4-28.7 % ของระบบงานที่พิจารณา (ค่าที่เหมาะสม = 19.6%)
	Attribute Hiding Factor (AHF)	70.2-89.3 % ของระบบงานที่พิจารณา (ค่าที่เหมาะสม = 79.7%)
	Coupling Factor (COF)	3.9-17.7 % ของระบบงานที่พิจารณา (ค่าที่เหมาะสม = 10.8%)
	Polymorphism Factor (PF)	3.5-9.6 % ของระบบงานที่พิจารณา (ค่าที่เหมาะสม = 6.5%)
	Reuse Factor (RF)	> 43 % ของระบบงานที่พิจารณา
	Relative Degree of Code Saving (RDCS)	เข้าใกล้ 100 % ของระบบงานที่พิจารณา
	Reuse Size and Frequency (R_{sf})	> 66 % ของระบบงานที่พิจารณา
	Reuse Ratio (RR)	31.0-64.0 % ของระบบงานที่พิจารณา

จากตารางที่ ข.1 เป็นข้อเสนอแนะสำหรับค่าที่ได้จากการวัดโดยใช้มาตรวัดแบบต่างๆ ที่แสดงให้เห็นว่า ซอฟต์แวร์หรือระบบงานที่ทำการวัดนั้นมีคุณภาพเหมาะสำหรับการนำกลับมาใช้ใหม่ และจากมาตรวัดที่ใช้วัดในส่วนของการสืบทอดคุณสมบัตินั้นควรจะมี Trade-off กับมาตรวัดตัวอื่นๆ คือถ้าค่าที่ได้จากการวัดมีค่าสูง จะทำให้มีความซับซ้อนมากในการพัฒนาและดูแลแต่จะมีความสามารถในการนำกลับมาใช้ใหม่สูง และถ้าหากมีจำนวนของคลาสที่มากแล้วจะทำให้ระบบนั้นจะต้องมีการทดสอบมากขึ้นแต่การนำกลับมาใช้ใหม่ก็จะมีประสิทธิภาพที่ดีขึ้นด้วย [1],[10]

ภาคผนวก ค

การทดสอบมาตรวัดในระดับโครงงาน

เป็นการนำรหัสโปรแกรมจากโครงงาน JMetric ซึ่งได้แก่คลาส SUIController.java (รูปที่ 5.9) และคลาส UIController.java (รูปที่ 5.10) ซึ่งเป็นคลาสที่มีการถ่ายทอดคุณสมบัติมาจากคลาส SUIController.java มาทำการทดสอบเพื่อดูผลการทำงานของมาตรวัดในส่วนของโครงงาน ได้ผลการทดสอบดังนี้

ค.1 การทดสอบค่า MIF

เป็นการทดสอบมาตรวัดในส่วนของ การหาผลรวมของวิธีดำเนินการที่มีการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ MIF ได้ดังนี้

จากสูตร $$MIF = \frac{\sum_{i=1}^{TC} M_i(C_i)}{\sum_{i=1}^{TC} M_a(C_i)}$$

โดยที่ : $M_a(C_i) = M_d(C_i) + M_i(C_i)$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

M_i = 0

M_a = 17

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

M_i = 17 – 3

= 14

M_a = 4 + 14

= 18

ดังนั้นค่า MIF ที่คำนวณได้

$$MIF = \frac{(0+14)}{(17+18)}$$

= 0.40

ค.2 การทดสอบค่า AIF

เป็นการทดสอบมาตรวัดในส่วนของ การหาผลรวมของลักษณะประจำที่มีการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ AIF ได้ดังนี้

จากสูตร $$AIF = \frac{\sum_{i=1}^{TC} A_i(C_i)}{\sum_{i=1}^{TC} A_a(C_i)}$$

โดยที่ : $A_a(C_i) = A_d(C_i) + A_i(C_i)$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

A_i = 0

A_a = 1

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

A_i = 1 – 0

= 1

A_a = 1 + 1

= 2

ดังนั้นค่า AIF ที่คำนวณได้

$$AIF = \frac{(0+1)}{(1+2)}$$

= 0.3333

ค.3 การทดสอบค่า MHF

เป็นการทดสอบมาตรวัดในส่วนของ การหาผลรวมของวิธีดำเนินการทั้งหมดที่เป็นการซ่อนข้อมูลของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ MHF ได้ดังนี้

จากสูตร $$MHF = \frac{\sum_{i=1}^{TC} M_h(C_i)}{\sum_{i=1}^{TC} M_d(C_i)}$$

โดยที่ : $M_d(C_i) = M_v(C_i) + M_h(C_i)$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

M_d = 16

M_h = 0

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

M_d = 16 + 3

= 19

M_h = 16

ดังนั้นค่า MHF ที่คำนวณได้

$$MHF = \frac{(0+16)}{(16+19)}$$

= 0.4571

ค.4 การทดสอบค่า AHF

เป็นการทดสอบมาตรวัดในส่วนของ การหาผลรวมของลักษณะประจำทั้งหมดที่เป็นการซ่อนข้อมูลของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ AHF ได้ดังนี้

จากสูตร $$AHF = \frac{\sum_{i=1}^{TC} A_h(C_i)}{\sum_{i=1}^{TC} A_d(C_i)}$$

โดยที่ : $A_d(C_i) = A_v(C_i) + A_h(C_i)$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

A_d = 0

A_h = 0

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

A_d = 0 + 1

= 1

A_h = 1

ดังนั้นค่า AHF ที่คำนวณได้

$$AHF = \frac{(0+1)}{(0+1)}$$

= 1

ค.5 การทดสอบค่า PF

เป็นการทดสอบมาตรวัดในส่วนของ การวัดความสัมพันธ์ระหว่างวัตถุที่มาจากต่างคลาส โดยมีการตอบสนองต่อการกระทำบนพื้นฐานเดียวกันด้วยรูปแบบที่แตกต่างกันของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ PF ได้ดังนี้

จากสูตร $$PF = \frac{\sum_{i=1}^{TC} M_o(C_i)}{\sum_{i=1}^{TC}[M_n(C_i) \times DC(C_i)]}$$

โดยที่ : $M_n(C_i) = M_d(C_i) - M_o(C_i)$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

M_o = 0

$M_n(C_i) \times DC(C_i)$ = 17 x 1

= 17

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

M_o = 3

$M_n(C_i) \times DC(C_i)$ = 1 x 2

= 2

ดังนั้นค่า PF ที่คำนวณได้

$$PF = \frac{(0+3)}{(17+2)}$$

= 0.1579

ค.6 การทดสอบค่า COF

เป็นการทดสอบมาตรวัดในส่วนของ การวัดการเข้าคู่กันระหว่างคลาส ของคลาส SUIController และคลาส UIController แต่จะไม่รวมถึงการเข้าคู่กันในลักษณะที่เป็นการถ่ายทอดคุณสมบัติ ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ COF ได้ดังนี้

จากสูตร $$COF = \frac{\sum_{i=1}^{TC}\left[\sum_{j=1}^{TC} is\_client(C_i, C_j)\right]}{TC^2 - TC}$$

โดยที่ : $$is\_client(C_i, C_j) = \begin{cases} 1 & iff \quad C_i \Rightarrow C_j \wedge C_i \neq C_j \\ 0 & otherwise \end{cases}$$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

is_client = 1

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

is_client = 1

ดังนั้นค่า COF ที่คำนวณได้

$$COF = \frac{(1+1)}{(2^2-2)}$$
$$= 1$$

ค.7 การทดสอบค่า RF

เป็นการทดสอบมาตรวัดในส่วนของ การวัดการนำกลับมาใช้ใหม่ซึ่งจะพิจารณาจากการนำส่วนประกอบจากไลบรารี่ที่มีอยู่มาใช้ใหม่ หรือมีการนำกลับมาใช้ใหม่ในลักษณะของการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ RF ได้ดังนี้

จากสูตร

$$RF = \frac{\sum_{i=1}^{TC} in\_library(C_i)}{TC} + \frac{MIF * \sum_{i=1}^{TC}[1 - in\_library(C_i)]}{TC}$$

โดยที่ : $in\_library(C_i) = \begin{cases} 1 & iff \quad C_i \in L \\ 0 & otherwise \end{cases}$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

$in\_library$ = 1

$1 - in\_library$ = 1 – 1

= 0

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

$in\_library$ = 1

$1 - in\_library$ = 1 – 1

= 0

ค่า MIF ที่ได้จากการคำนวณใน 5.2.1 = 0.4

ดังนั้นค่า RF ที่คำนวณได้

$$RF = \frac{(1+1)}{2} + \frac{0.4 \times 0}{2}$$
$$= 1$$

ค.8 การทดสอบค่า RDCS

เป็นการทดสอบมาตรวัดในส่วนของ การวัดระดับความสัมพันธ์ของวิธีดำเนินการที่มีการนำกลับมาใช้ใหม่ ในลักษณะของการลดการเขียนรหัสโปรแกรมเนื่องจากมีการถ่ายทอดคุณสมบัติของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ RDCS ได้ดังนี้

จากสูตร $RDCS = \frac{CS}{T_M}$

$$CS = Cap_T(t) - T_M$$

$$Cap_T(t) = \sum_{i=1}^{n} Cap(C_i)$$

$$Cap(C) = the\ number\ of\ methods\ in\ C + the\ number\ of\ all\ inherited\ methods$$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

$Cap(C)$ = 17 + 0

T_M = 17

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

$Cap(C)$ = (4 – 3) + 16 + 16

= 33

T_M = 4 + 16

= 20

ดังนั้นค่า RDCS ที่คำนวณได้

CS = (17 + 33) – (17 + 20)

= 13

$$RDCS = \frac{13}{(17+20)}$$

= 0.3514

ค.9 การทดสอบค่า R_{sf}

เป็นการทดสอบมาตรวัดในส่วนของ การวัดขนาดของรหัสโปรแกรมและความถี่ของรหัสโปรแกรมที่มีการนำกลับมาใช้ใหม่ของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ R_{sf} ได้ดังนี้

จากสูตร $$R_{sf}(S) = \frac{Size_{sf} - Size_{act}}{Size_{sf}}$$

โดยที่ : $Size_{act}(S) = \sum_{i=1}^{TC} Size(C_i)$

$$Size_{sf}(S) = \sum_{i=1}^{TC} Size(C_i) * calls(C_i)$$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

$Size_{sf}$ = (2 x 0) + (2 x 0) + (2 x 0) + (2 x 1) + (2 x 0) + (2 x 1) + (2 x 1) + (4 x 0)
+ (2 x 1) + (2 x 1) + (2 x 1) + (2 x 1) + (2 x 1) + (2 x 1) + (2 x 1) + (2 x 0)
+ (2 x 1)

= 22

$Size_{act}$ = 36

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

$Size_{sf}$ = (2 x 0) + (2 x 0) + (16 x 15) + (2 x 1)

= 242

$Size_{act}$ = 22

ดังนั้นค่า R_{sf} ที่คำนวณได้

$$R_{sf} = \frac{(242+22)-(36+22)}{(242+22)}$$
$$= 0.7803$$

ค.10 การทดสอบค่า RR

เป็นการทดสอบมาตรวัดในส่วนของ การหาอัตราส่วนของการนำองค์ประกอบของคลาส SUIController ที่มีการถ่ายทอดคุณสมบัติมาจากคลาส UIController กลับมาใช้ใหม่ ซึ่งสามารถแสดงการหาค่าวัดในส่วนของ RR ได้ดังนี้

จากสูตร $$RR(S) = \frac{\sum_{C_i \in S} IR(i) * Size(C_i)}{\sum_{C_i \in S} Size(C_i)}$$

โดยที่ : $$IR_i = \begin{cases} 1 & iff \quad Change_i < 0.25 \\ 0 & otherwise \end{cases}$$

แสดงค่าที่ได้จากคลาส UIController ดังนี้

$$Size = 34$$
$$IR * Size = 0$$

แสดงค่าที่ได้จากคลาส SUIController ดังนี้

$$Size = 20$$
$$IR * Size = 20$$

ดังนั้นค่า RR ที่คำนวณได้

$$RR = \frac{(0+20)}{(34+20)}$$
$$= 0.3704$$

ประวัติผู้เขียนวิทยานิพนธ์

เกิดเมื่อวันที่ 19 เมษายน พ.ศ. 2515 ที่จังหวัดนราธิวาส สำเร็จการศึกษาหลักสูตรวิทยาศาสตร์บัณฑิต (วท.บ) สาขาวิทยาศาสตร์คอมพิวเตอร์ จากมหาวิทยาลัยรามคำแหงเมื่อปีการศึกษา 2536 และสำเร็จการศึกษาหลักสูตรศิลปศาสตร์บัณฑิต (ศป.บ) สาขาการเมืองการปกครอง จากมหาวิทยาลัยรามคำแหง เมื่อปีการศึกษา 2545 ปัจจุบัน (พ.ศ. 2547) ทำงานที่บริษัทโปรเฟสชันแนล เวสต์ เทคโนโลยี (1999) จำกัด (มหาชน) และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต (วท.ม) สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปี พ.ศ. 2543