หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 3

การวิเคราะห์และออกแบบระบบ

ในบทนี้จะกล่าวถึงรายละเอียดการวิเคราะห์และออกแบบเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคฟังก์ชันพอยต์สำหรับโปรแกรมเชิงวัตถุ โดยเครื่องมือที่พัฒนาขึ้นนี้มีชื่อว่า Chula OOFP Counting เครื่องมือวัดนี้ทำงานในรูปแบบไคลเอนต์เซอร์ฟเวอร์โดยใช้โปรโตคอลไฮเปอร์เท็กซ์ทรานสเฟอร์โปรโตคอล (Hypertext Transfer Protocol: HTTP) สำหรับการติดต่อสื่อสาร ขั้นตอนการวิเคราะห์และออกแบบระบบ จะนำเสนอผ่านทางแผนภาพภาษายูเอ็มแอล ซึ่งในงานวิจัยได้เลือกใช้แผนภาพ ประกอบด้วย โมเดลการใช้งาน แผนภาพคลาส แผนภาพแสดงลำดับการทำงาน และแผนภาพแสดงกิจกรรม โดยแผนภาพเหล่านี้สามารถครอบคลุมการวิเคราะห์และออกแบบระบบทั้งหมดของเครื่องมือวัดที่พัฒนาในงานวิจัยได้

3.1 สถาปัตยกรรมไคลเอนต์เซอร์ฟเวอร์ที่ใช้ในระบบ

ในงานวิจัยนี้ได้พัฒนาเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคการวัดแบบฟังก์ชันพอยต์เชิงวัตถุ เครื่องมือนี้ประกอบด้วย 2 ส่วนหลัก ได้แก่ ส่วนไคลเอนต์ซึ่งเป็นส่วนโปรแกรมที่ทำงานติดต่อกับผู้ใช้โดยตรงเพื่อรองรับคำสั่งต่างๆ อาทิเช่น คำสั่งสร้างภาษาเอโอแอล (Abstract Object Language: AOL) คำสั่งนับจำนวนฟังก์ชันพอยต์ เป็นต้น และส่วนที่สองได้แก่ ส่วนเซอร์ฟเวอร์ซึ่งเป็นเครื่องให้บริการเว็บที่บรรจุชุดคำสั่งภาษาเพิร์ล ที่ทำหน้าที่แปลงไฟล์ข้อมูลภาษาจาวาให้เป็นภาษาเอโอแอล และส่งผลที่ได้กลับสู่ส่วนโปรแกรมไคลเอนต์ โปรโตคอลเอชทีทีพี เป็นโปรโตคอลที่ใช้ในการติดต่อระหว่างไคลเอนต์และเซอร์ฟเวอร์ การทำงานของเครื่องมือวัดนี้สามารถทำงานได้ทั้งเครือข่ายอินเตอร์เน็ตและอินทราเน็ต รูปที่ 3.1 แสดงรูปแบบการทำงานของโปรแกรม Chula OOFP Counting เครื่องมือวัดที่พัฒนาขึ้นในงานวิจัยได้เลือกใช้สถาปัตยกรรมไคลเอนต์เซอร์ฟเวอร์ เนื่องจากในปัจจุบันพัฒนาการของภาษาจาวามีอยู่ตลอดเวลาเพราะได้รับการยอมรับจากนักพัฒนาทั่วโลกถึงข้อดีต่างๆของตัวภาษา เช่น สามารถทำงานได้บนหลายแพลตฟอร์ม โดยที่ไม่ต้องแก้ไขโค้ดต้นฉบับ (Platform Independence) เป็นภาษาที่เน้นการใช้กลไกการจัดการความผิดปกติ (Exception Handling) ทำให้สามารถจัดการกับความผิดปกติบางอย่างที่อาจเกิดขึ้นในขณะโปรแกรมทำงาน โดยโปรแกรมไม่ล้มเหลวหรือหยุดลง เป็นต้น ดังนั้นภาษาจาวาจึงมีการปรับปรุง เปลี่ยนแปลงอย่างต่อเนื่อง

ผู้วิจัยพัฒนาโมดูลสำหรับการแปลงไฟล์ต้นฉบับภาษาจาวาให้เป็นข้อมูลภาษาเอโอแอลเพื่อใช้ในการนับจำนวนฟังก์ชันพอยต์ ดังนั้นเพื่อให้โมดูลดังกล่าวทันสมัยอยู่ตลอดเวลาตามพัฒนาการของภาษาจาวา จึงเห็นควรให้โมดูลดังกล่าวทำงานอยู่บนเครื่องให้บริการกลาง ทำให้การปรับปรุง เปลี่ยนแปลง แก้ไขโมดูลทำได้ทันที โดยไม่มีผลกระทบใดๆต่อการทำงานของโปรแกรมไคลเอนต์ของเครื่องมือวัด ทำให้ง่ายต่อการดูแลรักษาระบบ

รูปที่ 3.1 รูปแบบการทำงานของโปรแกรม Chula OOFP Counting

3.2 ภาพรวมในการพัฒนาเครื่องมือวัด

การพัฒนาเครื่องมือวัดในงานวิจัยนี้ ได้ใช้แผนภาพคลาสซึ่งเป็นรูปแบบหนึ่งของข้อกำหนดความต้องการด้านซอฟต์แวร์ (Software Requirement Specification) เป็นข้อมูลเข้าของระบบ กระบวนการคำนวณหาจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากแผนภาพคลาสจะทำโดยอัตโนมัติผ่านทางเครื่องมือวัดที่พัฒนาขึ้นซึ่งมีชื่อว่า Chula OOFP Counting ขั้นตอนกิจกรรมหลักของระบบนี้ดังรูปที่ 3.2 ซึ่งมีรายละเอียดดังนี้

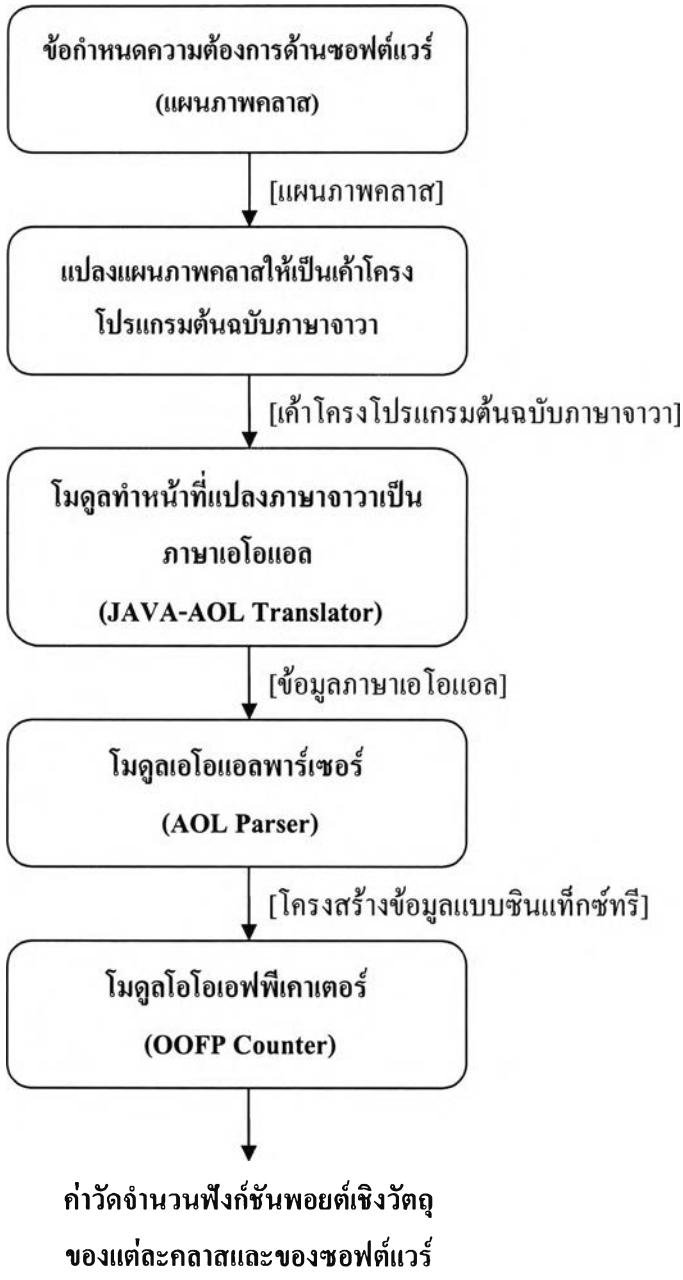
1) นักพัฒนาสร้างแผนภาพคลาสของซอฟต์แวร์ที่ต้องการขึ้นมา ส่วนใหญ่จะทำในระหว่างขั้นตอนการวิเคราะห์และออกแบบระบบโดยใช้เครื่องมือทางวิศวกรรมซอฟต์แวร์ (Computer Aids Software Engineering: CASE) เช่น โปรแกรมโออีดับบริว (OEW) หรือโปรแกรมเรชันแนลโรส (Rational Rose) เป็นต้น ในงานวิจัยได้เลือกใช้โปรแกรมโออีดับบริวเป็นเครื่องมือหลักในการสร้างแผนภาพคลาส

2) ทำการแปลงแผนภาพคลาสที่สร้างขึ้นให้เป็นโปรแกรมต้นฉบับภาษาจาวา เครื่องมือช่วยในการออกแบบซอฟต์แวร์เชิงวัตถุในปัจจุบันมีความสามารถในการสร้างเค้าโครงของโปรแกรมต้นฉบับภาษาจาวาที่เป็นไปตามแผนภาพคลาสที่ได้ออกแบบไว้

3) ทำการแปลงโปรแกรมต้นฉบับภาษาจาวาให้อยู่ในรูปแบบของภาษาเอโอแอล ภาษานี้ถูกใช้เป็นภาษากลางที่เชื่อมต่อระหว่างแผนภาพคลาสและกระบวนการนับจำนวนฟังก์ชันพอยต์เชิงวัตถุ ภาษาเอโอแอลเป็นภาษาที่ใช้อธิบายการออกแบบซอฟต์แวร์เชิงวัตถุ (General-Purpose Design Description Language) คุณสมบัติที่สำคัญของภาษาเอโอแอลคือ สามารถแสดงแนวคิดการออกแบบซอฟต์แวร์เชิงวัตถุได้ในรูปแบบถ้อยคำ (Textual) และยังคงอยู่บนพื้นฐานของภาษายูเอ็มแอลซึ่งเป็นภาษามาตรฐานที่ใช้ในการออกแบบซอฟต์แวร์เชิงวัตถุ (ตัวอย่างการใช้ภาษา เอโอแอลอยู่ในภาคผนวก ก) ขั้นตอนการแปลงข้อมูลภาษาจาวาของเครื่องมือช่วยในการออกแบบเป็นภาษาเอโอแอลจะทำงานโดยโมดูลการแปลง จาวา-เอโอแอล (JAVA-AOL Translator)

4) ภาษาเอโอแอลที่ได้รับจะถูกนำมาวิเคราะห์โดยโมดูลเอโอแอลพาร์เซอร์ (AOL Parser) ซึ่งผลการทำงานจะได้โครงสร้างข้อมูลแบบซินแท็กซ์ทรี (Abstract Syntax Tree: AST) ของแผนภาพคลาส

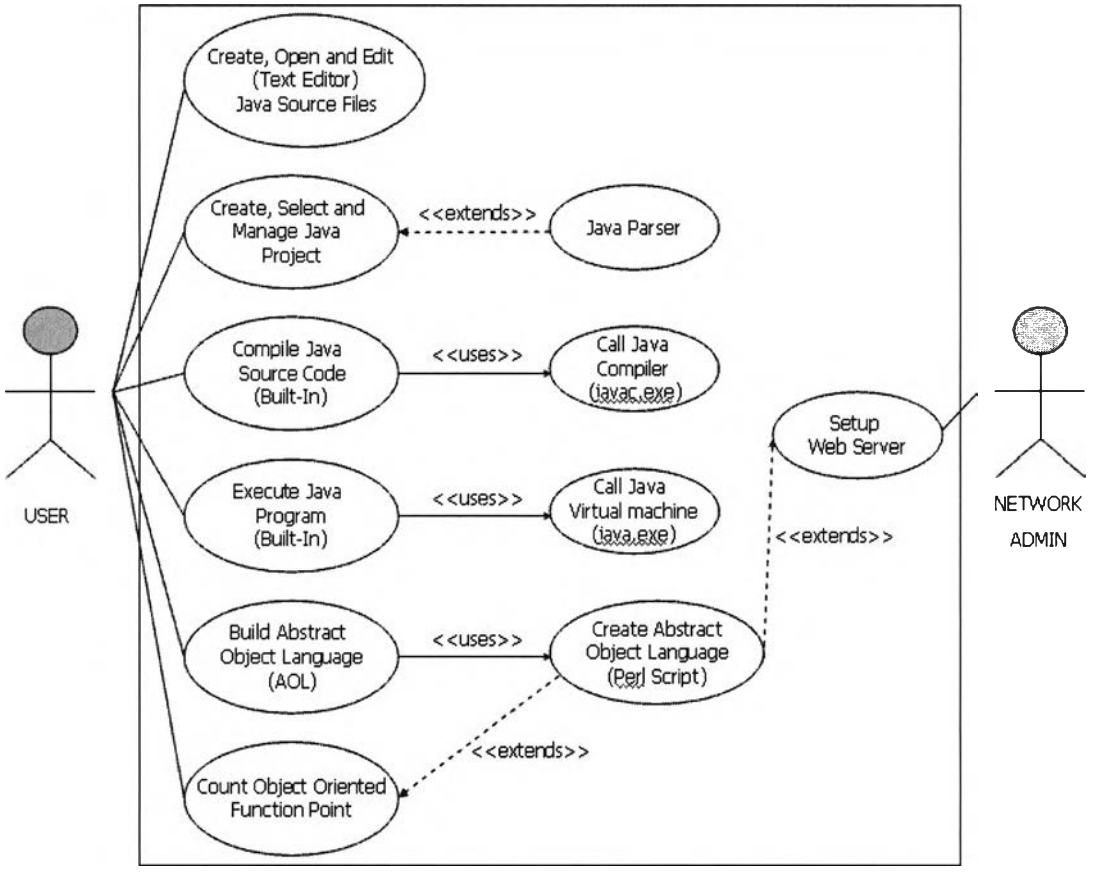
5) โมดูลโอโอเอฟพีเคาเตอร์ (OOFP Counter) ทำการนับจำนวนของฟังก์ชันพอยต์ ตามหลักเกณฑ์การนับของฟังก์ชันพอยต์เชิงวัตถุ (OOFP Counting Rule) ค่าวัดที่ได้จากการนับได้แก่ ฟังก์ชันพอยต์เชิงวัตถุของแต่ละคลาสและฟังก์ชันพอยต์เชิงวัตถุทั้งหมดของแผนภาพคลาสที่นำมาพิจารณา



รูปที่ 3.2 แผนภาพแสดงกิจกรรมการคำนวณหาจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากแผนภาพคลาส

3.3 โมเดลการใช้งาน (Use-Case Diagram) ของเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคฟังก์ชันพอยต์เชิงวัตถุ

เป้าหมายของการสร้างโมเดลการใช้งานก็เพื่ออธิบายเรื่องราวของขอบเขตของปัญหา ทั้งหมดว่ามีส่วนประกอบอะไรบ้างและเกี่ยวพันกันจนกลายเป็นระบบได้อย่างไร การสร้างโมเดลการใช้งานจะช่วยให้ผู้พัฒนาระบบสามารถแยกแยะได้ว่าจะมีกิจกรรมอะไรที่น่าจะเกิดขึ้นในระบบบ้าง ยิ่งไปกว่านั้นโมเดลการใช้งานถือว่าเป็นแผนภาพพื้นฐาน ซึ่งมีขีดความสามารถในการอธิบายสิ่งต่างๆด้วยรูปภาพที่ไม่ซับซ้อน และถือเป็นรากฐานในการเริ่มต้นวิเคราะห์ระบบ ดังนั้นโมเดลการใช้งานที่สมบูรณ์ถูกต้องย่อมช่วยให้การวิเคราะห์ระบบมีความสมบูรณ์และถูกต้องเช่นเดียวกัน จากการวิเคราะห์ทั้งหมดทำให้ได้โมเดลการใช้งาน สำหรับเครื่องมือวัดในงานวิจัย ดังรูปที่ 3.3



รูปที่ 3.3 โมเดลการใช้งานของเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคฟังก์ชันพอยต์เชิงวัตถุ

จากรูปที่ 3.3 ผลของการวิเคราะห์สามารถสรุปเพื่อใช้ในการสร้างโมเดลการใช้งานได้ดังนี้

1) แอคเตอร์ (Actor) ของระบบมีด้วยกัน 2 แอคเตอร์ คือ ผู้ใช้ (User) และผู้ดูแลระบบเครือข่าย (Network Administrator)

2) ยูสเคส หลักที่สำคัญประกอบด้วย

2.1 การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์ต้นฉบับภาษาจาวา (Create, Open and Edit Java Source Files)

ผลการวิเคราะห์พบว่า ผู้ใช้โปรแกรมต้องทำการสร้าง เปิด และแก้ไขไฟล์ต้นฉบับภาษาจาวาด้วยตนเอง ดังนั้นผู้ใช้จึงเป็นแอคเตอร์ของยูสเคสนี้

2.2 การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์ (Create, Select and Manage Java Project)

สิ่งที่ทำให้การเปิดโปรเจ็คไฟล์ไม่สมบูรณ์คือ ไวยากรณ์ของภาษาจาวาในไฟล์ต้นฉบับไม่ถูกต้องครบถ้วน โมดูลจาวาพาร์เซอร์ ทำหน้าที่ตรวจสอบความผิดพลาดของไวยากรณ์ ภาษาจาวาในไฟล์ต้นฉบับของโปรเจ็คไฟล์ ดังนั้นจึงมียูสเคสสำหรับโมดูลตรวจสอบไวยากรณ์ภาษาจาวา ที่จะมาเชื่อมขยาย (Extends) กับยูสเคส การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์ ซึ่งผู้ใช้โปรแกรมต้องทำการเปิดโปรเจ็คไฟล์ด้วยตนเอง ดังนั้นผู้ใช้จึงเป็นแอคเตอร์ของยูสเคสนี้

2.3 การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา (Compile Java Source Code)

การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา ใช้ โปรแกรมคอมไพเลอร์ภาษาจาวาในการคอมไพล์ ดังนั้นยูสเคส การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา ต้องเรียกใช้ยูสเคสการใช้งานโปรแกรมคอมไพเลอร์ภาษาจาวา ซึ่งผู้ใช้โปรแกรมต้องเป็นผู้สั่งคอมไพล์โปรแกรมต้นฉบับภาษาจาวาด้วยตนเอง ดังนั้นผู้ใช้จึงเป็นแอคเตอร์ของยูสเคสนี้

2.4 การประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา (Execute Java Program)

การประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวาต้องใช้โปรแกรมจาวาเวอร์ชวลแมชชีน (Java Virtual Machine) ในการประมวลผล ดังนั้นยูสเคสการประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา ต้องเรียกใช้ยูสเคสการใช้งานโปรแกรมจาวาเวอร์ชวลแมชชีน ซึ่งผู้ใช้โปรแกรมต้องเป็นผู้สั่งประมวลผลโปรแกรมจาวาด้วยตนเอง ดังนั้นผู้ใช้จึงเป็นแอคเตอร์ของยูสเคสนี้

2.5 การสร้างข้อมูลภาษาเอโอแอลจากไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์ (Build Abstract Object Language)

การสร้างข้อมูลภาษาเอโอแอลต้องใช้ชุดคำสั่งภาษาเพิร์ล (Perl) ซึ่งทำงานบนเครื่องให้บริการเว็บสำหรับการประมวลผล เพื่อวิเคราะห์ให้ได้ข้อมูลภาษาเอโอแอลจากไฟล์ต้นฉบับภาษาจาวา ดังนั้นยูสเคส การสร้างข้อมูลภาษาเอโอแอล ต้องเรียกใช้ยูสเคส การสร้างข้อมูลภาษาเอโอแอลด้วยชุดคำสั่งภาษาเพิร์ลอีกต่อหนึ่ง ซึ่งผู้ใช้โปรแกรมต้องเป็นผู้สั่งสร้างข้อมูลภาษาเอโอแอลด้วยตนเอง ดังนั้นผู้ใช้จึงเป็นแอคเตอร์ของยูสเคสนี้

2.6 การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล (Count Object Oriented Function Point)

การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจะกระทำบนข้อมูลภาษาเอโอแอล ซึ่งได้จากผลการทำงานของชุดคำสั่งภาษาเพิร์ลในเครื่องให้บริการเว็บ ดังนั้นยูสเคส การสร้างข้อมูลภาษาเอโอแอล ด้วยชุดคำสั่งภาษาเพิร์ลต้องเชื่อมขยายกับยูสเคส การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล ซึ่งผู้ใช้โปรแกรมต้องเป็นผู้สั่งนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุด้วยตนเอง ดังนั้นผู้ใช้จึงเป็นแอคเตอร์ของยูสเคสนี้

2.7 การติดตั้งเครื่องให้บริการเว็บ (Setup WebServer)

เครื่องให้บริการเว็บต้องติดตั้งโปรแกรมเว็บเซอร์ฟเวอร์ เช่น โปรแกรมอินเทอร์เน็ตอินฟอร์เมชันเซอร์เวอร์ (Internet Information Server)โปรแกรมอาปาเช (Apache) หรือโปรแกรมออมนิ (OmniHTTPd) ซึ่งในงานวิจัยนี้เลือกใช้โปรแกรมออมนิเป็นโปรแกรมให้บริการเว็บ นอกจากนี้ในเครื่องให้บริการเว็บต้องติดตั้งโปรแกรมทำหน้าที่แปลชุดคำสั่งภาษาเพิร์ล (Perl Interpreter) และติดตั้งไฟล์ชุดคำสั่งภาษาเพิร์ลที่ทำหน้าที่แปลงไฟล์ต้นฉบับภาษาจาวาเป็นข้อมูลภาษาเอโอแอลด้วย ดังนั้นยูสเคสของไฟล์ชุดคำสั่งภาษาเพิร์ล ต้องเชื่อมขยายกับยูสเคสการติดตั้งเครื่องให้บริการเว็บ ซึ่งการวิเคราะห์พบว่า ผู้ดูแลระบบเครือข่าย (Network Administrator) มีหน้าที่การติดตั้งโปรแกรมเหล่านี้บนเครื่องให้บริการเว็บ ดังนั้นผู้ดูแลระบบเหลือข่ายจึงเป็น แอคเตอร์ของยูสเคสนี้

3.4 แผนภาพคลาส (Class Diagram)

จากผลของการวิเคราะห์โมเดลการใช้งานสามารถนำมาพิจารณาวัตถุหรือคลาสได้ดังตารางที่ 3.1 และรูปที่ 3.4 แสดงแผนภาพคลาสหลักของเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคฟังก์ชันพอยต์เชิงวัตถุ

ตารางที่ 3.1 แสดงผลการวิเคราะห์โมเดลการใช้งานของระบบ

USE CASE	CLASS
การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์ต้นฉบับภาษาจาวา (Create, Open and Edit Java Source Files)	- คลาสแอพลิเคชัน(OOFPApp) - คลาสหน้าต่างหลัก (JMainFrame) - คลาสหน้าต่างย่อย (JMDIChildWnd) - คลาสหน้าต่างแสดงไฟล์ข้อมูล (JEditView) - คลาสเอกสาร (JStyledDocument)
การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์ (Create, Select and Manage Java Project)	- คลาสหน้าต่างหลัก (JMainFrame) - คลาสเก็บข้อมูลของโปรเจ็คไฟล์ (JProjectClass) - คลาสหน้าต่างย่อยกำหนดค่าคอนฟิกูเรชัน (JPreferencesDialog)

ตารางที่ 3.1 แสดงผลการวิเคราะห์โมเดลการใช้งานของระบบ (ต่อ)

USE CASE	CLASS
	- คลาสหน้าต่างย่อยเลือกโปรเจ็คไฟล์ (JSelectProjectDlg) - คลาสหน้าต่างย่อยเพิ่มชื่อโปรเจ็คไฟล์ (JAddProjectDialog) - คลาสแสดงลิสต์ของชื่อไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์ (JFilePage) - คลาสแสดงลิสต์ชื่อคลาส เมธอด และแอตริบิว (JClassPage)
การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา (Compile Java Source Code)	- คลาสหน้าต่างหลัก (JMainFrame) - คลาสหน้าต่างย่อยกำหนดค่าคอนฟิกูเรชัน (JPreferencesDialog) - คลาสเก็บข้อมูลของโปรแกรมอัดถะประโยชน์ (JUsrToolClass) - คลาสแสดงข้อความสถานะการทำงาน (JOutputList) - คลาสแอพลิเคชัน(OOFPApp) - คลาสอ่านและบันทึกข้อมูลค่าเริ่มต้นของระบบ (JINIProfile)
การประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา (Execute Java Program)	- คลาสหน้าต่างหลัก (JMainFrame) - คลาสหน้าต่างย่อยกำหนดค่าคอนฟิกูเรชัน (JPreferencesDialog) - คลาสเก็บข้อมูลของโปรแกรมอัตถะประโยชน์ (JUsrToolClass - คลาสแสดงข้อความสถานะการทำงาน (JOutputList) - คลาสแอพลิเคชัน(OOFPApp) - คลาสอ่านและบันทึกข้อมูลค่าเริ่มต้นของระบบ (JINIProfile)

ตารางที่ 3.1 แสดงผลการวิเคราะห์โมเดลการใช้งานของระบบ (ต่อ)

USE CASE	CLASS
การสร้างข้อมูลภาษาเอโอแอลจากไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์ (Build Abstract Object Language)	- คลาสหน้าต่างหลัก (JMainFrame) - คลาสเก็บข้อมูลของโปรเจ็คไฟล์ (JProjectClass) - คลาสการเชื่อมต่อเครื่องให้บริการเว็บ (JServConnect) - คลาสเธรดคำนวณเวลาที่ใช้ในการประมวลผล (JTimeConsumedThread) - คลาสแสดงข้อความสถานะการทำงาน (JOutputList)
การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล (Count Object Oriented Function Point)	- คลาสหน้าต่างหลัก (JMainFrame) - คลาสเก็บข้อมูลของโปรเจ็คไฟล์ (JProjectClass) - คลาสสร้างซินแท็กซ์ทรีจากภาษาเอโอแอล (AOLParser) - คลาสแสดงข้อความสถานะการทำงาน (JOutputList)

3.4.1 การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์ต้นฉบับภาษาจาวา (Create, Open and Edit Java Source Files)

เมื่อพิจารณาและวิเคราะห์ยูสเคสนี้ภายในขอบเขตของปัญหาแล้ว พบว่ามีคลาสที่เกี่ยวข้องดังแสดงในแผนภาพคลาสในรูปที่ 3.5 ซึ่งอธิบายได้ดังนี้

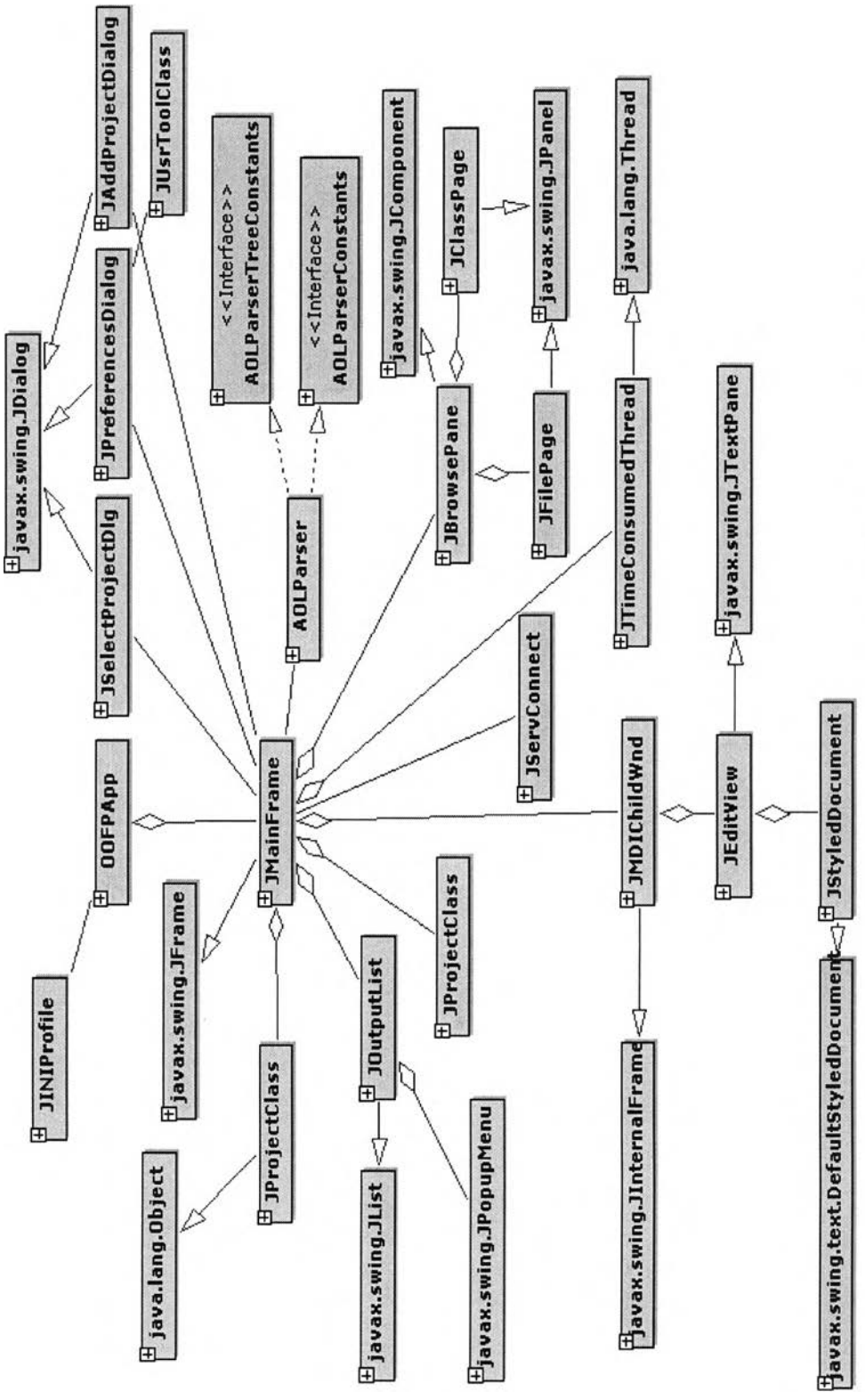
3.4.1.1 คลาสแอพลิเคชัน

แสดงด้วยคลาส OOFPApp โดยวัตถุของคลาสนี้เป็นวัตถุหลักและวัตถุแรกที่ถูกสร้างขึ้นเมื่อโปรแกรมเครื่องมือวัดของงานวิจัยถูกประมวลผลด้วยโปรแกรมจาวาเวอร์ชวลแมชชีน ฟังก์ชัน main() ซึ่งเป็นฟังก์ชันเริ่มต้นการทำงานของโปรแกรมจาวาปรากฏอยู่ในคลาสนี้ คลาส OOFPApp เก็บข้อมูลแอตริบิว (Attribute) ต่างๆ ของโปรแกรม เช่น ชื่อของโปรแกรมที่ปรากฏบนหน้าต่างหลัก ช่วงเวลาที่ต้องการบันทึกไฟล์ต้นฉบับอัตโนมัติ และโหมดเลือกการสร้างไฟล์ข้อมูลสำรอง เป็นต้น โดยที่แอตริบิวเหล่านี้คลาสอื่นๆสามารถเข้าถึงเพื่อนำไปใช้งานต่อไป

3.4.1.2 คลาสหน้าต่างหลัก

แสดงด้วยคลาส JMainFrame ซึ่งทำหน้าที่แสดงหน้าต่างหลักของโปรแกรมเครื่องมือวัด คลาสนี้สืบทอด (Inherit) มาจากคลาส JFrame ของแพ็กเกจสวิง (Swing) ซึ่งเป็นแพ็กเกจเพิ่มเติมมาตรฐานของภาษาจาวา คลาส JMainFrame สัมพันธ์กับคลาส OOFPApp ในลักษณะ Aggregation แบบ One to One โดยมี Cardinality เป็น 1..1

ซึ่งหมายความว่าวัตถุของคลาส OOFPApp สามารถมีวัตถุของคลาส JMainFrame ได้เพียง 1 วัตถุเท่านั้น เพื่อแสดงหน้าต่างหลักของโปรแกรม ภายในคลาส JMainFrame ได้บรรจุวัตถุส่วนประกอบย่อย ซึ่งสัมพันธ์ในลักษณะ Aggregation กับคลาส JMainFrame อาทิ วัตถุเมนู วัตถุแถบเครื่องมือ และวัตถุหน้าต่างย่อย เป็นต้น



รูปที่ 3.4 แสดงแผนภาพคลาสหลักของเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคฟังก์ชันพอยต์เชิงวัตถุ

3.4.1.3 คลาสหน้าต่างย่อย

แสดงด้วยคลาส JMDIChildWnd ซึ่งเป็นหน้าต่างย่อยที่บรรจุอยู่ในหน้าต่างหลัก (JMainFrame) คลาส JMDIChildWnd สืบทอดมาจากคลาส JInternalFrame ของแพ็กเกจสวิงนอกจากนี้ภายในคลาสยังบรรจุวัตถุส่วนประกอบย่อยที่สำคัญคือ วัตถุหน้าต่างแสดงไฟล์ข้อมูล (JEditView) คลาส JEditView สัมพันธ์กับคลาส JMainFrame ในลักษณะ Aggregation แบบ Many to One โดยมีค่า Cardinality เป็น 0..256 ซึ่งหมายความว่า วัตถุ JMainFrame สามารถมีวัตถุ JMDIChildWnd ได้ตั้งแต่ 0 ถึง 256 ตัว

3.4.1.4 คลาสหน้าต่างแสดงไฟล์ข้อมูล

แสดงด้วยคลาส JEditView ซึ่งทำหน้าที่แสดงไฟล์ข้อมูลพร้อมกับให้ผู้ใช้โปรแกรมสามารถเพิ่ม แก้ไข เปลี่ยนแปลงไฟล์ข้อมูลได้ ซึ่งคลาสนี้มีคุณสมบัติพื้นฐานเป็นอีดิเตอร์ โดนสืบทอดคุณสมบัติมาจากคลาส JTextPane ของแพ็กเกจสวิงในภาษาจาวา คลาส JEditView มีความสัมพันธ์กับคลาส JMDIChildWnd ในลักษณะ Aggregation แบบ One to One โดยมี Cardinality เป็น 1..1 กล่าวคือแต่ละวัตถุของ JMDIChildWnd สามารถมีวัตถุ JEditView ได้เพียง 1 วัตถุเท่านั้น

3.4.1.5 คลาสเอกสาร

แสดงด้วยคลาส JStyledDocument ซึ่งทำหน้าที่เก็บข้อมูลของไฟล์ที่ถูกเปิดใช้งาน เมื่อผู้ใช้โปรแกรมทำการเพิ่มเติม แก้ไข เปลี่ยนแปลงไฟล์ด้นฉบับที่แสดงอยู่บนหน้าจอ ข้อมูลของไฟล์ที่เก็บอยู่ในคลาสนี้จะเปลี่ยนแปลงตามที่ผู้ใช้แก้ไข คลาส JStyledDocument สืบทอดมาจากคลาส DefaultStyledDocument ของแพ็กเกจสวิงในภาษาจาวา นอกจากนี้คลาส JStyledDocument ยังมีความสัมพันธ์กับคลาส JEditView ในลักษณะ Aggregation แบบ One to One โดยมี Cardinality เป็น 1..1 กล่าวคือแต่ละวัตถุ JEditView สามารถมีวัตถุ JStyledDocument ได้เพียง 1 วัตถุเท่านั้น

3.4.2 การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์ (Create, Select and Manage Java Project)

เมื่อพิจารณาและวิเคราะห์ยูสเคสนี้ภายในขอบเขตของปัญหาแล้ว จะพบว่ามีคลาสที่เกี่ยวข้องดังแสดงในแผนภาพคลาสในรูปที่ 3.6 ซึ่งมีรายละเอียดดังนี้

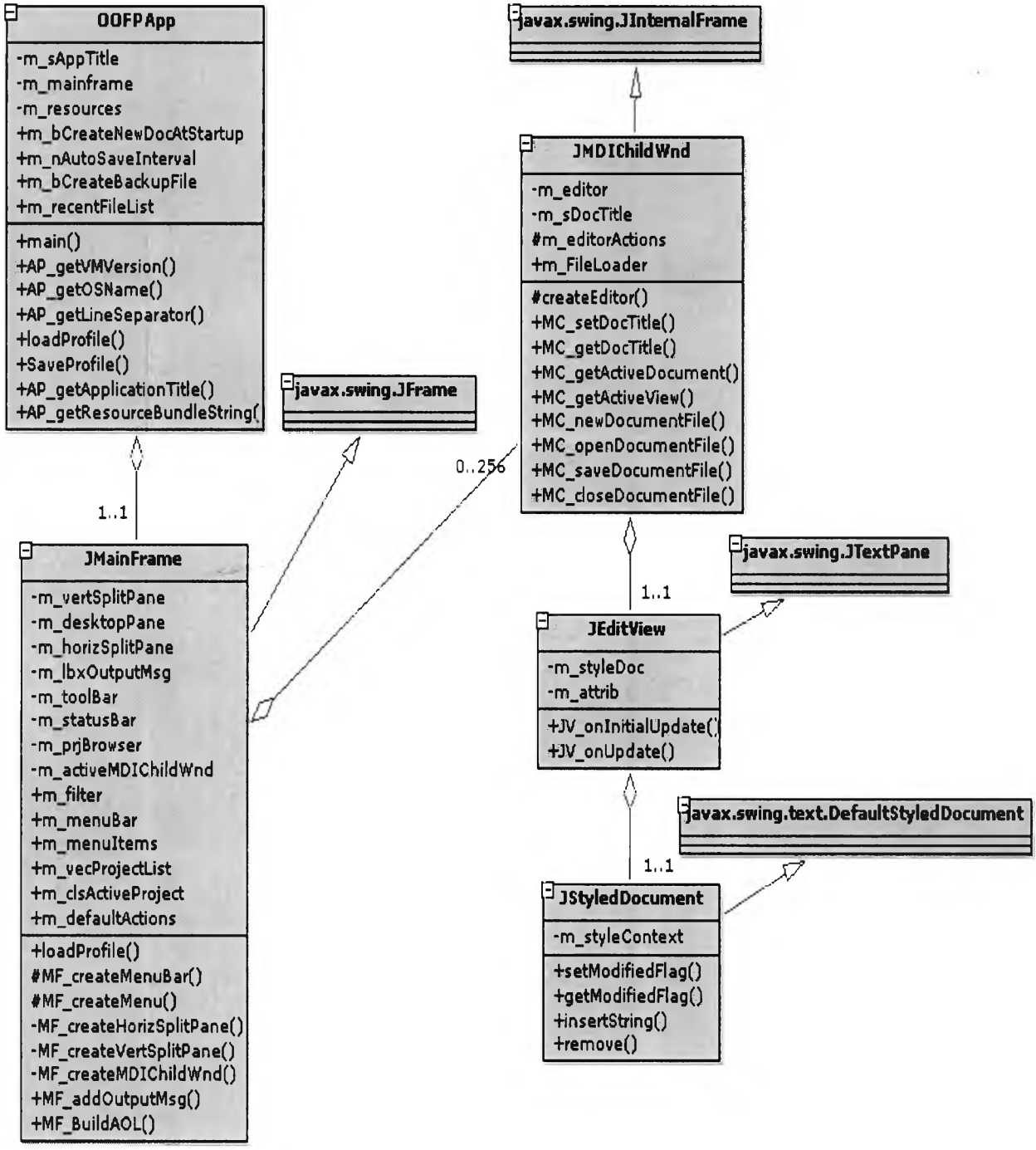
3.4.2.1 คลาสหน้าต่างหลัก

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.1.2

3.4.2.2 คลาสเก็บข้อมูลของโปรเจ็คไฟล์

แสดงด้วยคลาส JProjectClass คลาสนี้สืบทอดมาจากคลาส java.lang.Object ซึ่งเป็นคลาสแม่ของทุกๆ คลาสในภาษาจาวา แอตริบิว (attribute) ที่อยู่ในคลาสนี้ประกอบด้วย ชื่อของโปรเจ็คไฟล์และชื่อรวมถึงตำแหน่งที่อยู่ของทุกๆไฟล์ในโปรเจ็ค คลาส JProjectClass มีความสัมพันธ์กับคลาส JMainFrame ในลักษณะ

Aggregation แบบ Many to One โดยมีค่า Cardinality เป็น 0..n ซึ่งหมายความว่า คลาส JMainFrame สามารถสร้างวัตถุ JProjectClass ได้ไม่จำกัดจำนวน



รูปที่ 3.5 แผนภาพคลาสสำหรับยูสเคส การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์โปรแกรมภาษาจาวา

3.4.2.3 คลาสหน้าต่างย่อยกำหนดค่าคอนฟิกูเรชัน

แสดงด้วยคลาส JPreferencesDialog ซึ่งเป็นหน้าต่างย่อยสำหรับกำหนดค่าคอนฟิกกูเรชันต่างๆของเครื่องมือวัด ประกอบด้วย การกำหนดค่าทั่วไป การกำหนดฟอนต์ที่ใช้บนหน้าจอ การกำหนดฟอนต์ที่ใช้บนเครื่องพิมพ์ การกำหนดสีที่ใช้ในหน้าต่างย่อยต่างๆ การกำหนดค่าที่เกี่ยวกับไฟล์ข้อมูล การจัดการโปรเจ็คไฟล์ การจัดการโปรแกรมอัดฉะประโยชน์ (Utilities) ซึ่งรวมถึงโปรแกรมคอมไพเลอร์ภาษาจาวาและโปรแกรมจาวาเวอร์ชวลแมชชีนด้วย คลาส JPreferencesDialog สัมพันธ์กับคลาส JMainFrame ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..1 ทั้งในทิศทางความสัมพันธ์จากคลาส JMainFrame ไปยังคลาส JPreferencesDialog และในทิศทางกลับกันด้วย

3.4.2.4 คลาสหน้าต่างย่อยเลือกโปรเจ็คไฟล์

แสดงด้วยคลาส JSelectProjectDlg เป็นคลาสที่สืบทอดมาจากคลาส JDialog ของแพ็กเกจสวิงในภาษาจาวา คลาสนี้ทำหน้าที่แสดงลิสต์ของโปรเจ็คไฟล์ทั้งหมดที่ผู้ใช้กำหนดขึ้น คลาส JSelectProjectDlg สัมพันธ์กับคลาส JMainFrame ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..1 ทั้งในทิศทางความสัมพันธ์จากคลาส JMainFrame ไปยังคลาส JSelectProjectDlg และในทิศทางกลับกันด้วย

3.4.2.5 คลาสหน้าต่างย่อยเพิ่มชื่อโปรเจ็คไฟล์

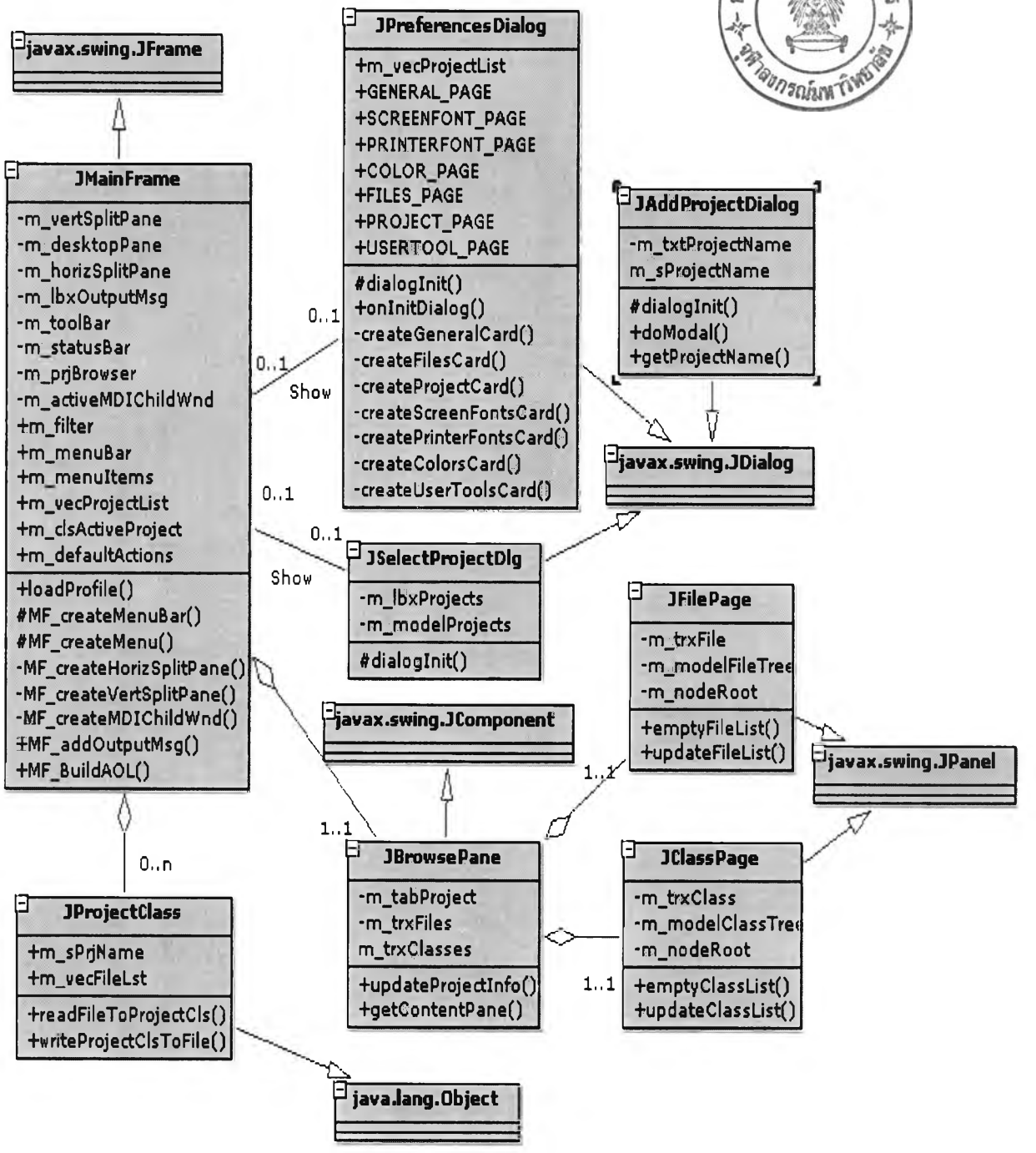
แสดงด้วยคลาส JAddProjectDialog เป็นคลาสที่สืบทอดมาจากคลาส JDialog ของแพ็กเกจสวิงในภาษาจาวา โดยทำหน้าที่รับชื่อโปรเจ็คไฟล์ใหม่ที่ผู้ใช้กำหนดขึ้น คลาสนี้จะถูกเรียกใช้โดยคลาส JPreferencesDialog ดังนั้นคลาส JAddProjectDialog สัมพันธ์กับคลาส JPreferencesDialog ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..1 ทั้งในทิศทางความสัมพันธ์จากคลาส JPreferencesDialog ไปยังคลาส JAddProjectDialog และในทิศทางกลับกันด้วย

3.4.2.6 คลาสแสดงลิสต์ของชื่อไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์

แสดงด้วยคลาส JFilePage เป็นคลาสที่สืบทอดมาจากคลาส JDialog ของแพ็กเกจสวิงในภาษาจาวา ทำหน้าที่แสดงลิสต์ของชื่อไฟล์ข้อมูลทั้งหมดในโปรเจ็คไฟล์ที่ถูกเปิดใช้ คลาสนี้สัมพันธ์กับคลาส JBrowsePane ในลักษณะ Aggregation โดยมีค่า Cardinality เป็น 1..1

3.4.2.7 คลาสแสดงลิสต์ชื่อคลาส เมธอด และแอตริบิว

แสดงด้วยคลาส JClassPage เป็นคลาสที่สืบทอดมาจากคลาส JDialog ของแพ็กเกจสวิง ทำหน้าที่แสดงลิสต์ของชื่อคลาส เมธอด และแอตริบิว ของไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์ที่ถูกเปิดใช้ คลาสนี้สัมพันธ์กับคลาส JBrowsePane ในลักษณะ Aggregation โดยมีค่า Cardinality เป็น 1..1



รูปที่ 3.6 แผนภาพคลาสสำหรับยูสเคส การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์

3.4.3 การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา (Compile Java Source Code) และการประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา (Execute Java Program)

เมื่อพิจารณาและวิเคราะห์ยูสเคสนี้ภายในขอบเขตของปัญหาแล้ว จะพบว่ามีคลาสที่เกี่ยวข้องดังแสดงในแผนภาพคลาสในรูปที่ 3.7 ซึ่งมีรายละเอียดดังนี้

3.4.3.1 คลาสหน้าต่างหลัก

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.1.2

3.4.3.2 คลาสหน้าต่างย่อยกำหนดค่าคอนฟิกูเรชัน

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.2.3

3.4.3.3 คลาสเก็บข้อมูลของโปรแกรมอัตถะประโยชน์

แสดงด้วยคลาส JUsrToolClass ทำหน้าที่เก็บข้อมูลของโปรแกรมอัตถะประโยชน์ (Utility Program) ต่างๆที่ผู้ใช้กำหนด ทำให้สามารถเรียกใช้งานโปรแกรมที่ผู้ใช้กำหนดได้โดยตรงจากภายในโปรแกรม Chula OOFP Counting ข้อมูลดังกล่าวประกอบไปด้วย

1) ข้อความที่ปรากฏบนเมนูหลักเพื่อเรียกใช้งาน
2) ชื่อและตำแหน่งที่อยู่ของโปรแกรมอัตถะประโยชน์ที่ต้องการ
3) ค่าพารามิเตอร์ต่างๆที่จำเป็นสำหรับการประมวลผลโปรแกรมอัตถะประโยชน์
4) ไดเร็กทอรี่เริ่มต้นของการประมวลผลโปรแกรมอัตถะประโยชน์

คลาสนี้สัมพันธ์กับคลาส JPreferencesDialog ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..n ในทิศทางความสัมพันธ์จากคลาส JPreferencesDialog ไปยังคลาส JUsrToolClass ซึ่งหมายความว่าวัตถุ JPreferencesDialog สามารถสร้างวัตถุ JUsrToolClass ได้ไม่จำกัดหรือไม่สร้างเลยก็ได้และในทางกลับกันวัตถุ JUsrToolClass จะถูกสร้างจากวัตถุ JPreferencesDialog เท่านั้น โดยมีค่า Cardinality เป็น 1..1

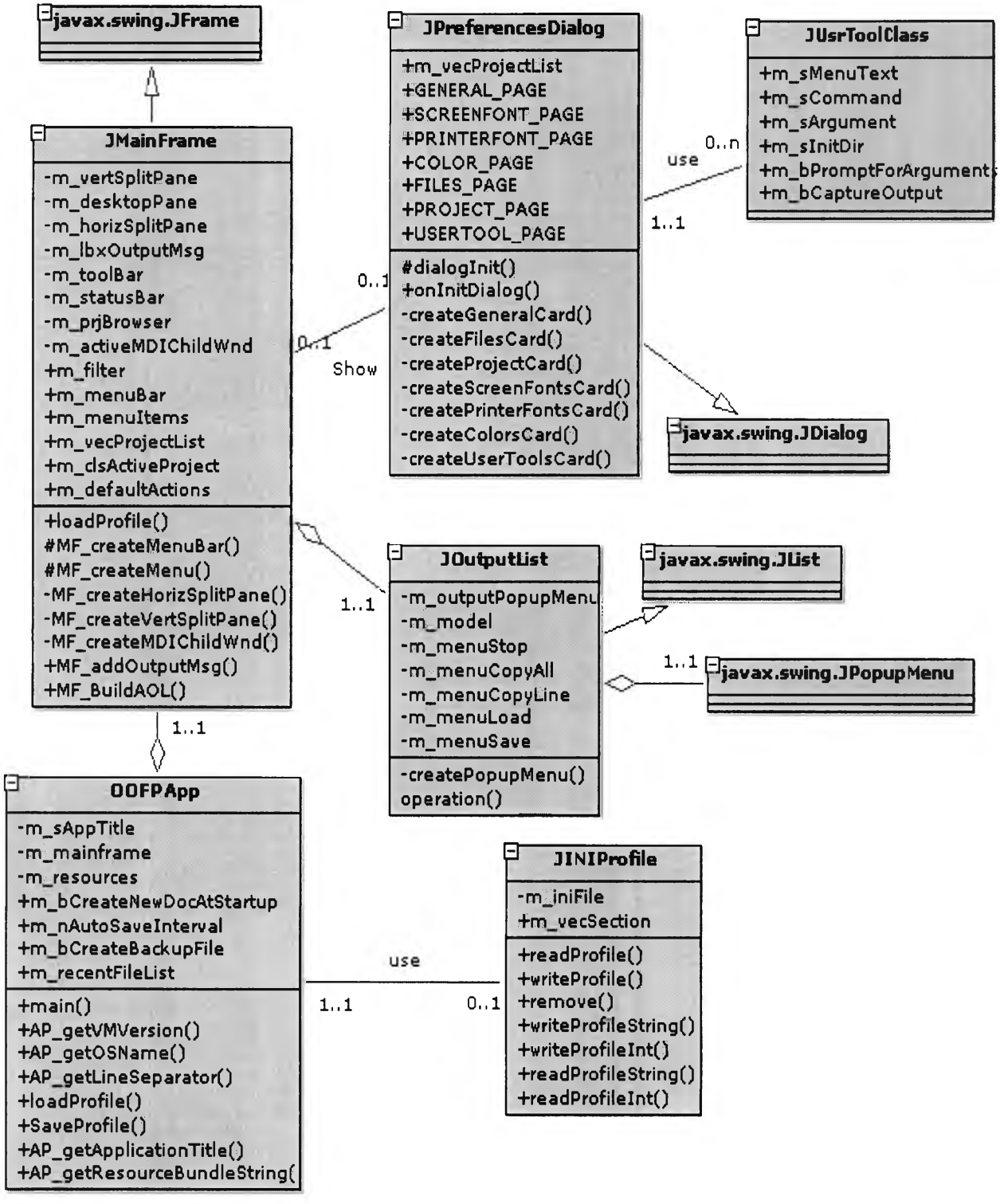
3.4.3.4 คลาสแสดงข้อความสถานะการทำงาน

แสดงด้วยคลาส JOutputList ทำหน้าที่แสดงข้อความหรือรายละเอียดต่างๆที่เป็นผลลัพธ์จากการประมวลผลโปรแกรม ได้แก่

1) ข้อความที่เกิดจากการประมวลผลโปรแกรมอัตถะประโยชน์ที่ผู้ใช้กำหนด
2) ข้อความที่เกิดจากการสร้างข้อมูลภาษาเอโอแอล (Abstract Object Language: AOL) จากไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์
3) ข้อความที่เกิดจากการนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล

คลาสนี้สัมพันธ์กับคลาส JMainFrame ในลักษณะ Aggregation โดยมีค่า Cardinality เป็น 1..1 นั่นคือวัตถุ JMainFrame มีวัตถุ JOutputList ได้เพียงวัตถุเดียวเท่านั้น

I20305229



รูปที่ 3.7 แผนภาพคลาสสำหรับยูสเคส การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา

3.4.3.5 คลาสแอพลิเคชัน

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.1.1

3.4.3.6 คลาสอ่านและบันทึกข้อมูลค่าเริ่มต้นของระบบ

แสดงด้วยคลาส JINIProfile ทำหน้าที่อ่านและบันทึกข้อมูลค่าเริ่มต้นต่างๆที่ผู้ใช้กำหนดไว้ก่อนหน้า โดยทำการอ่านและบันทึกลงในไฟล์ข้อมูลที่ชื่อ usertools.dat คลาสนี้สัมพันธ์กับคลาส OOFPApp ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..1 ในทิศทางความสัมพันธ์จากคลาส OOFPApp ไปยังคลาส JINIProfile และในความสัมพันธ์ในทิศทางตรงกันข้ามจะมีค่า Cardinality เป็น 1..1

3.4.4 การสร้างข้อมูลภาษาเอโอแอล (Abstract Object Language: AOL) จากไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์ (Build Abstract Object Language)

เมื่อพิจารณาและวิเคราะห์ยูสเคสนี้ภายในขอบเขตของปัญหาแล้ว จะพบว่ามีคลาสที่เกี่ยวข้องดังแสดงในแผนภาพคลาสในรูปที่ 3.8 ซึ่งอธิบายได้ดังนี้

3.4.4.1 คลาสหน้าต่างหลัก

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.1.2

3.4.4.2 คลาสเก็บข้อมูลของโปรเจ็คไฟล์

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.2.2

3.4.4.3 คลาสการเชื่อมต่อเครื่องให้บริการเว็บ

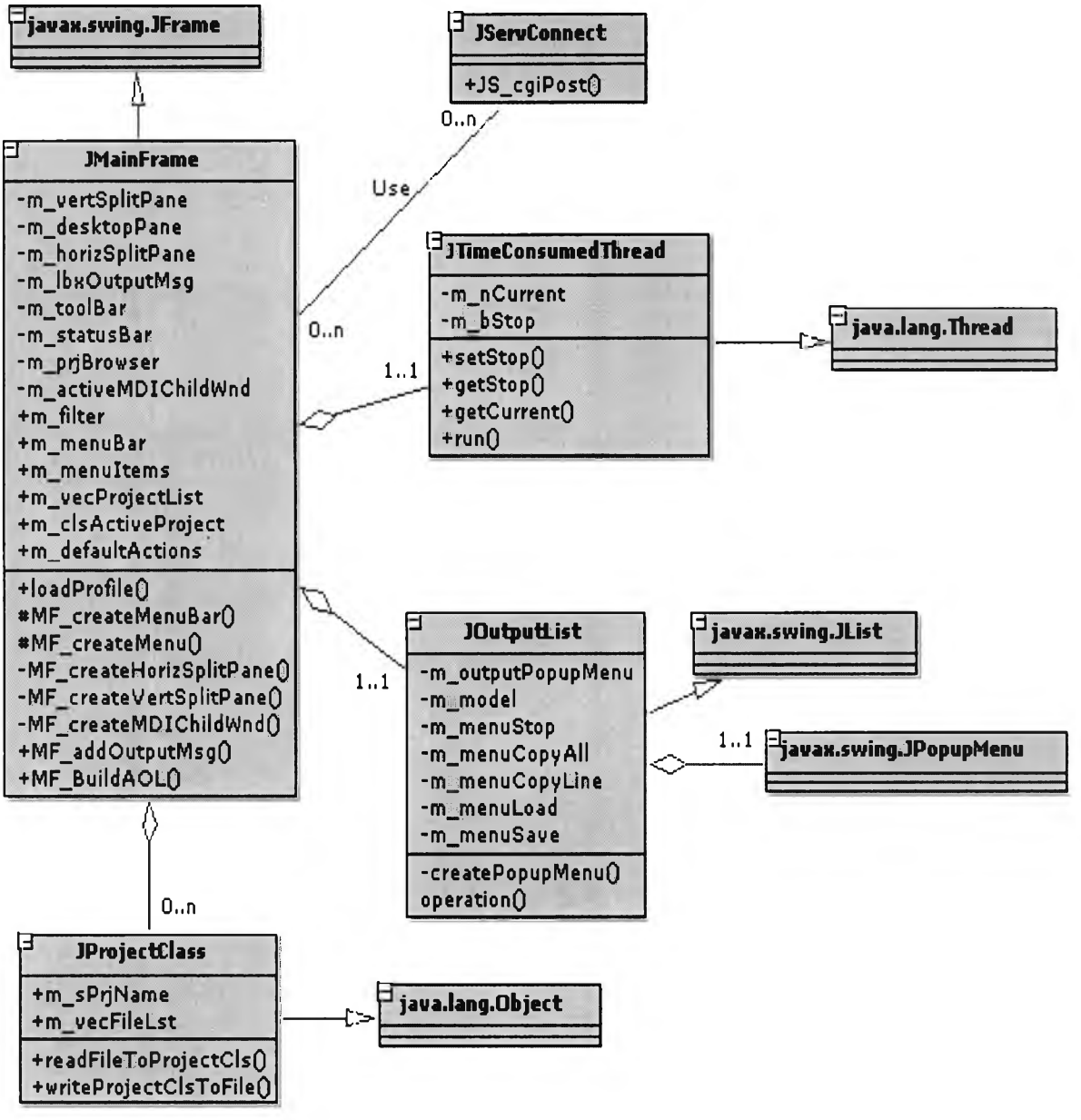
แสดงด้วยคลาส JServConnect ทำหน้าที่ติดต่อกับเครื่องให้บริการเว็บเมื่อมีคำสั่งร้องขอใช้บริการโดยส่งคำสั่งเอชทีทีพี รีเควส (HTTP Request) แบบโพสต์ (POST) โดยเป้าหมายการขอใช้บริการคือ เรียกใช้ชุดคำสั่งภาษาเพิร์ลในเครื่องให้บริการเว็บเพื่อวิเคราะห์และสร้างข้อมูลภาษาเอโอแอลจากไฟล์ต้นฉบับภาษาจาวาที่ถูกส่งไปจากเครื่องมือวัดในส่วนไคลเอนต์ พร้อมกันนี้คลาส JServConnect ยังทำหน้าที่รับข้อมูลตอบกลับซึ่งเป็นข้อมูลภาษาเอโอแอล ซึ่งเป็นผลการทำงานจากโมดูลแปลงภาษาจาวาเป็นภาษาเอโอแอลในเครื่องให้บริการเว็บ คลาส JServConnect สัมพันธ์กับคลาส JMainFrame ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..n ทั้งในทิศทางความสัมพันธ์จากคลาส JMainFrame ไปยังคลาส JServConnect และในทิศทางกลับกันด้วย

3.4.4.4 คลาสเธรดคำนวณเวลาที่ใช้ในการประมวลผล

แสดงด้วยคลาส JTimeConsumedThread ทำหน้าที่คำนวณเวลาในการประมวลผลโปรเซสหรือเธรดใดๆ คลาสนี้สืบทอดมาจากคลาส java.lang.Thread ซึ่งเป็นคลาสมาตรฐานในภาษาจาวา คลาส JTimeConsumedThread สัมพันธ์กับคลาส JMainFrame ในลักษณะ Aggregation โดยมีค่า Cardinality เป็น 1..1 นั่นคือคลาส JMainFrame สามารถมีวัตถุ JTimeConsumedThread ได้เพียงตัวเดียวเท่านั้น

3.4.4.5 คลาสแสดงข้อความสถานะการทำงาน

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.3.4



รูปที่ 3.8 แผนภาพคลาสสำหรับยูสเคส การสร้างข้อมูลภาษาเอโอแอลจากไฟล์ภาษาจาวาในโปรเจ็คไฟล์

3.4.5 การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล (Count Object Oriented Function Point)

การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากภาษาเอโอแอล คือการนำข้อมูลภาษาเอโอแอล มาผ่านกระบวนการการสร้างซินแท็กทรี (Syntax Tree) โดยผู้วิจัยได้ใช้เครื่องมือช่วยในการสร้างตัวแปลภาษาและซินแท็กทรี ที่ชื่อว่า จาวา คอมไพเลอร์ คอมไพเลอร์ (Java Compiler Compiler -- JavaCC) โดยผลที่ได้จากการทำงานของเครื่องมือนี้คือ กลุ่มของคลาสที่เข้าใจภาษาและไวยากรณ์ของภาษาเอโอแอล ซึ่งเป็นภาษาหลักสำหรับการวิเคราะห์คำนวณหาจำนวนของฟังก์ชันพอยต์เชิงวัตถุ เมื่อพิจารณาและวิเคราะห์ยูสเคส นี้ภายในขอบเขตของปัญหาแล้ว จะพบว่ามีคลาสที่เกี่ยวข้องดังแสดงในแผนภาพคลาสในรูปที่ 3.9 ซึ่งมีรายละเอียดดังนี้

3.4.5.1 คลาสหน้าต่างหลัก

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.1.2

3.4.5.2 คลาสเก็บข้อมูลของโปรเจ็คไฟล์

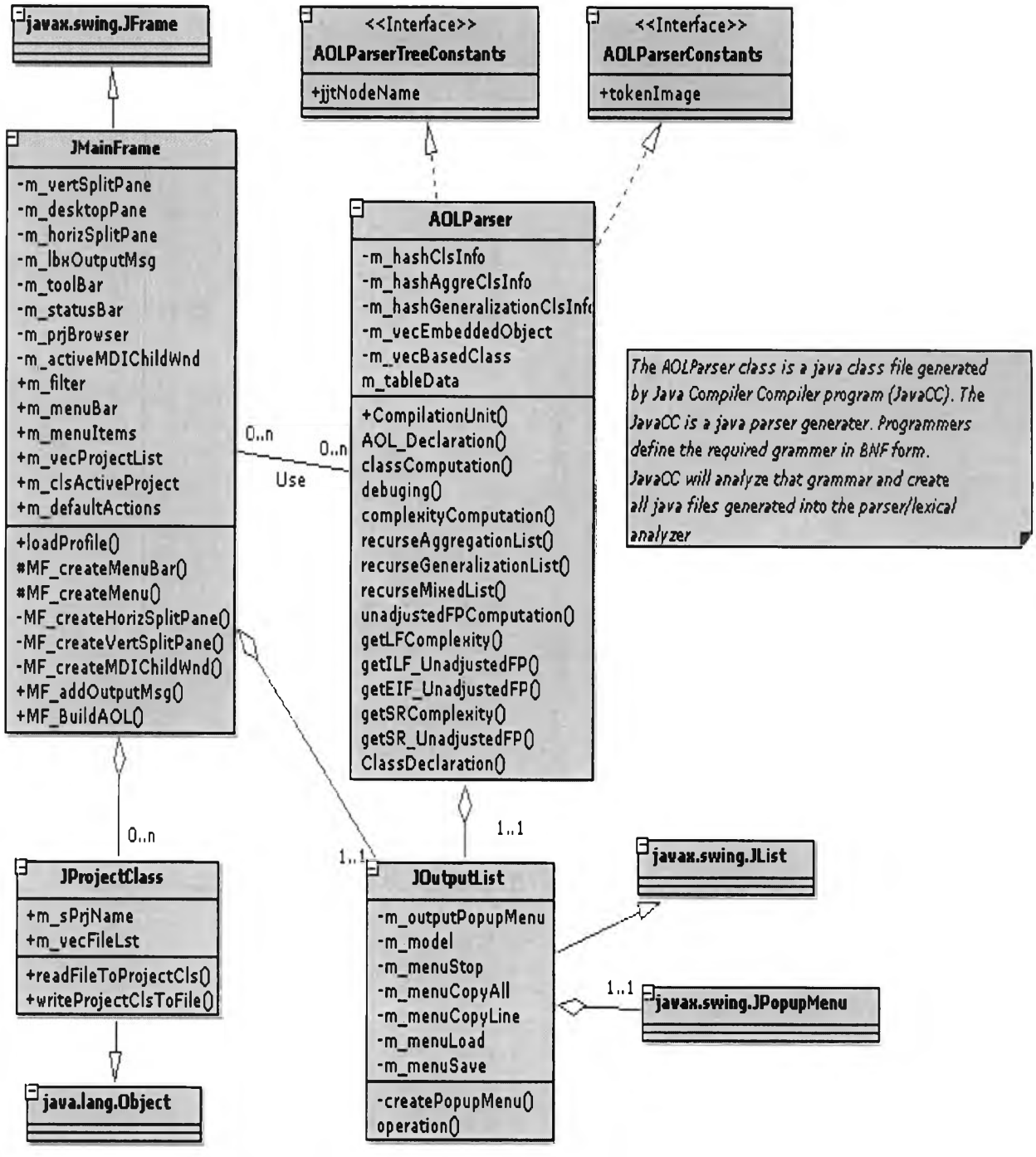
สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.2.2

3.4.5.3 คลาสสร้างซินแท็กซ์ทรีจากภาษาเอโอแอล

แสดงด้วยคลาส AOLParser เป็นคลาสหลักทำหน้าที่วิเคราะห์ไฟล์ต้นฉบับภาษาจาวาและทำการสร้างซินแท็กซ์ทรี โดยมีอินเตอร์เฟสคลาส AOLParserConstants ซึ่งประกาศค่าคงที่ต่างๆและโทเคนต่างๆ (โทเคนเหล่านี้คือ กลุ่มอักขระตามไวยากรณ์ของภาษาเอโอแอล) และมีคลาส Token และคลาส AOLParserTokenManager ที่มีหน้าที่อ่านอักขระของไฟล์ต้นฉบับและแบ่งให้เป็นโทเคน คลาส AOLParser สัมพันธ์กับคลาส JMainFrame ในลักษณะ Association โดยมีค่า Cardinality เป็น 0..n ทั้งในทิศทางความสัมพันธ์จากคลาส JMainFrame ไปยังคลาส AOLParser และในทิศทางกลับกันด้วย

3.4.5.4 คลาสแสดงข้อความสถานะการทำงาน

สามารถดูรายละเอียดได้ในหัวข้อที่ 3.4.3.4



รูปที่ 3.9 แผนภาพคลาสสำหรับยูสเคส การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล

3.5 แผนภาพแสดงลำดับการทำงาน (Sequence Diagram)

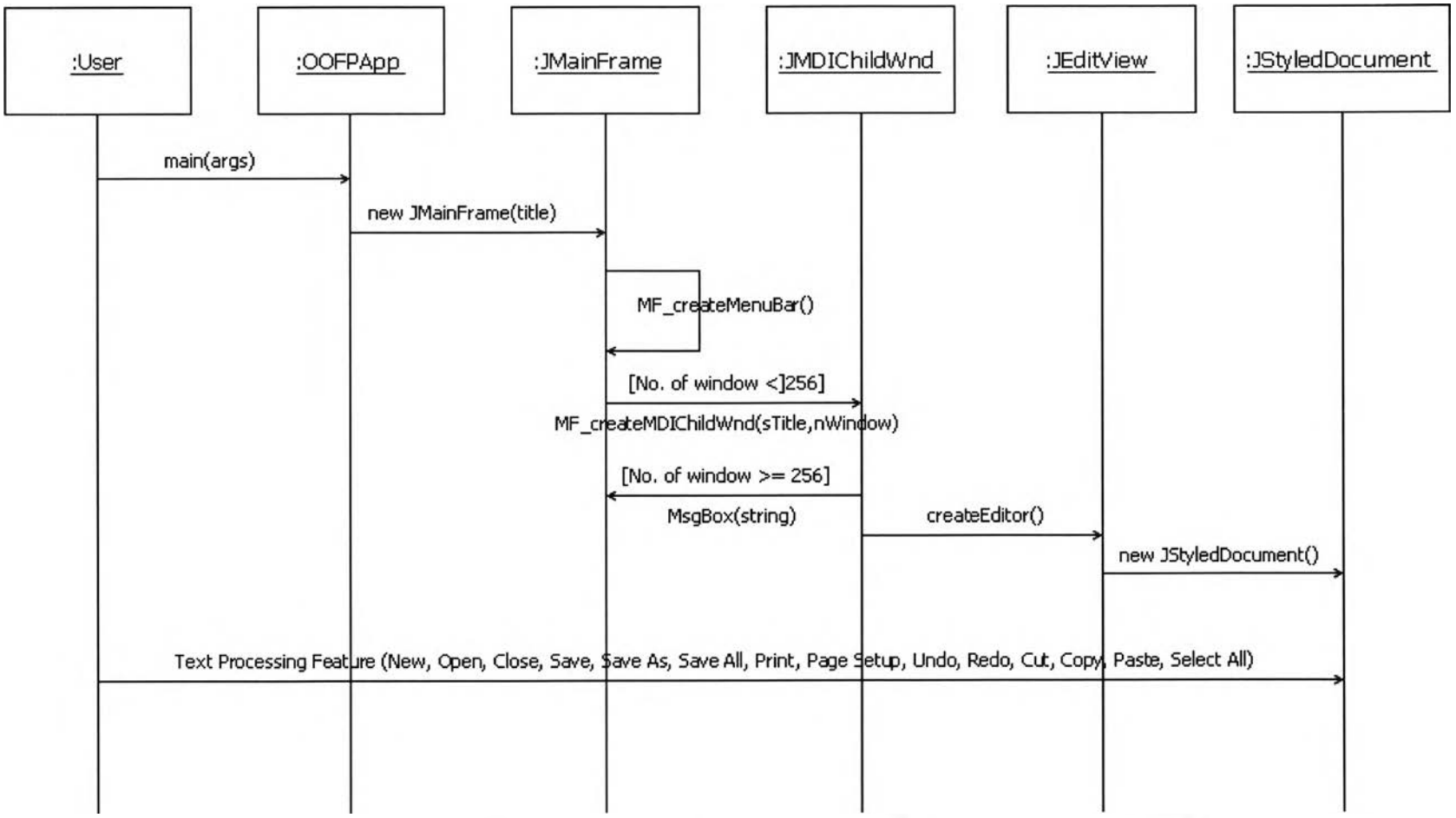
แผนภาพแสดงลำดับการทำงานเป็นหนึ่งในแผนภาพแบบจำลองเชิงกิจกรรม ซึ่งตามแนวคิดแล้วแบบจำลองเชิงกิจกรรม หมายถึงการจำลองกิจกรรมและลำดับของกิจกรรมที่เกิดขึ้นหรืออาจเกิดขึ้นในขอบเขตของปัญหาที่กำหนด โดยให้อยู่ในรูปที่เข้าใจได้ง่าย โดยการเขียนภาพตามลำดับเวลาและเหตุการณ์ ส่วนแผนภาพแสดงลำดับการทำงาน เป็นการจำลองกระบวนการที่ประกอบไปด้วยคลาสหรือวัตถุ เส้นที่ใช้ในแผนภาพจะแสดงลำดับเวลา และแสดงกิจกรรมที่เกิดขึ้นจากวัตถุหรือคลาส ในงานวิจัยนี้ได้นำแผนภาพการใช้งาน และแผนภาพคลาสที่ได้สร้างไว้ข้างต้นมาพิจารณาเพื่อวิเคราะห์สร้างแผนภาพแสดงลำดับการทำงาน ซึ่งมีรายละเอียดดังต่อไปนี้

3.5.1 การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์ต้นฉบับภาษาจาวา (Create, Open and Edit Java Source Files)

รูปที่ 3.10 เป็นแผนภาพแสดงลำดับการทำงาน ที่แสดงกิจกรรมในยูสเคส การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์ต้นฉบับภาษาจาวา กิจกรรมในยูสเคส นี้เริ่มจากที่ผู้ใช้ได้ทำการรันโปรแกรม Chula OOFP Counting โดยใช้คำสั่ง

java OOFPApp

คำสั่งข้างต้นเป็นการเรียกใช้โปรแกรม java.exe ซึ่งเป็นโปรแกรมจาวาเวอร์ชวลแมชชีน สำหรับรันโปรแกรมภาษาจาวาที่อยู่ในรูปไบต์โค้ด (Byte Code) หลังจากนั้นโปรแกรมเจวีเอ็มจะเรียกฟังก์ชัน main() ของวัตถุ OOFPApp ซึ่งมีหน้าที่สำคัญคือการสร้างหน้าต่างหลักคือวัตถุ JMainFrame โดยหน้าต่างหลักที่สร้างขึ้นนี้ประกอบด้วย วัตถุเมนู วัตถุแถบเครื่องมือ วัตถุแถบแสดงสถานะ วัตถุแสดงลิสต์ของไฟล์ในโปรเจ็ค และข้อมูลของทุกคลาส ได้แก่ แอตริบิวและเมธอดต่างๆ เมื่อผู้ใช้เลือกเมนูเพื่อสร้างหน้าต่างย่อยซึ่งทำให้วัตถุ JMDIChildWnd ถูกสร้าง แต่จะมีการตรวจสอบจำนวนของวัตถุ JMDIChildWnd ที่เปิดใช้งานอยู่ ถ้ามีมากกว่าหรือเท่ากับ 256 จะไม่สามารถสร้างวัตถุ JMDIChildWnd ได้ พร้อมกับแสดงข้อความเตือน ซึ่งทางแก้ไขก็คือวัตถุ JMDIChildWnd ที่เปิดใช้งานอยู่ต้องถูกทำลายหรือปิดไปเสียก่อน แต่ถ้าวัตถุ JMDIChildWnd ที่เปิดใช้งานอยู่มีจำนวนน้อยกว่าแล้ว วัตถุที่สร้างขึ้นใหม่จะสร้างหน้าต่างย่อยสำหรับงานพิมพ์ได้แก่วัตถุ JEditView และภายในวัตถุ JEditView จะสร้างวัตถุ JStyledDocument สำหรับเก็บข้อมูลที่แสดงผลอยู่บนวัตถุ JEditView แต่ละวัตถุ JMDIChildWnd จะมีวัตถุ JEditView และ JStyledDocument ได้เพียงอย่างละ 1 วัตถุเท่านั้น เมื่อวัตถุ JMDIChildWnd วัตถุ JEditView และ วัตถุ JStyledDocument ได้ถูกสร้างเสร็จสมบูรณ์แล้ว ผู้ใช้จะสามารถสร้างแก้ไข เปลี่ยนแปลงไฟล์ข้อมูลได้ รวมถึงไฟล์ข้อมูลต้นฉบับภาษาจาวาบนเครื่องมือ Chula OOFP Counting นี้



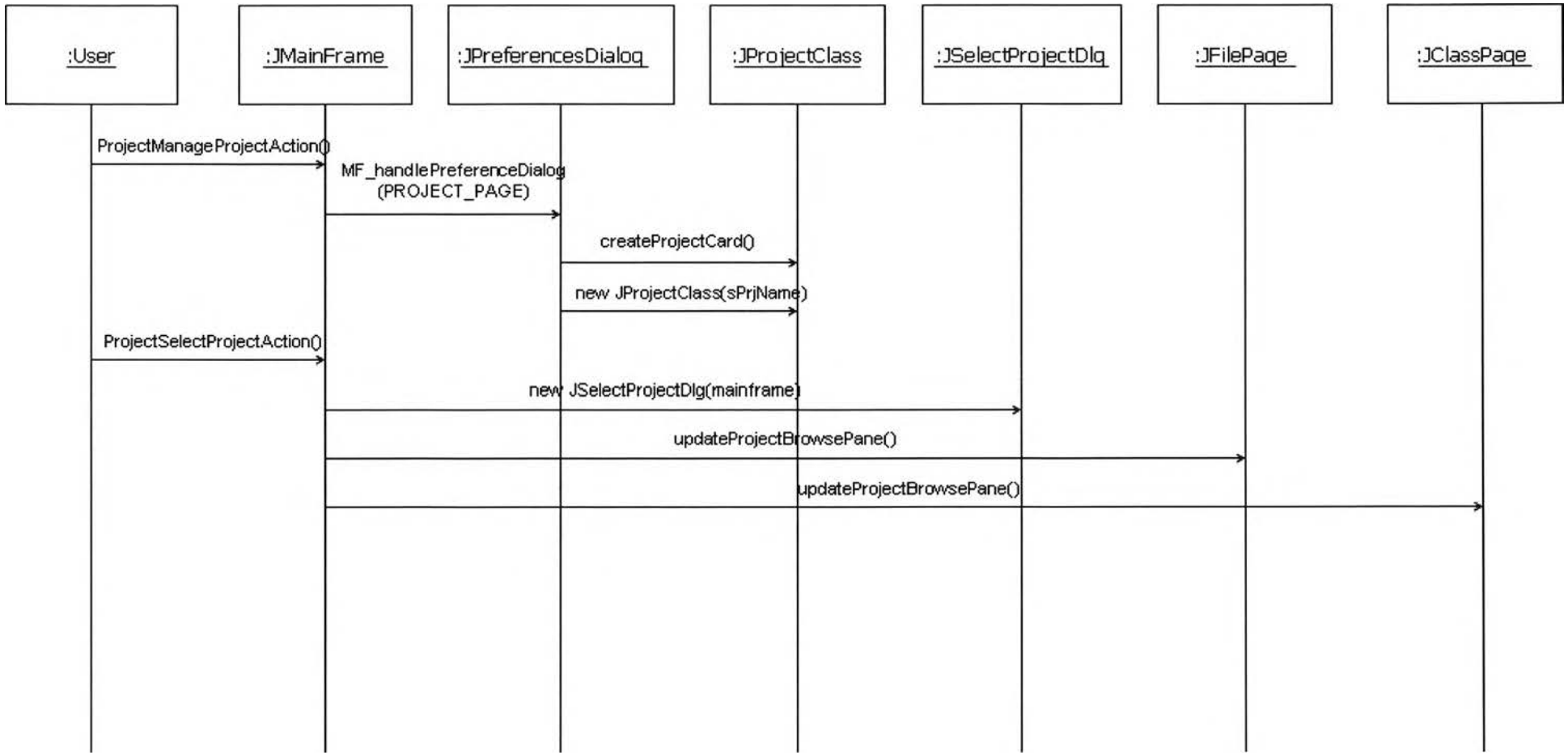
รูปที่ 3.10 แผนภาพแสดงลำดับการทำงานสำหรับยูสเคส การสร้าง การเปิด และการปรับปรุงแก้ไขไฟล์ต้นฉบับภาษาจาวา

3.5.2 การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์ (Create, Select and Manage Java Project)

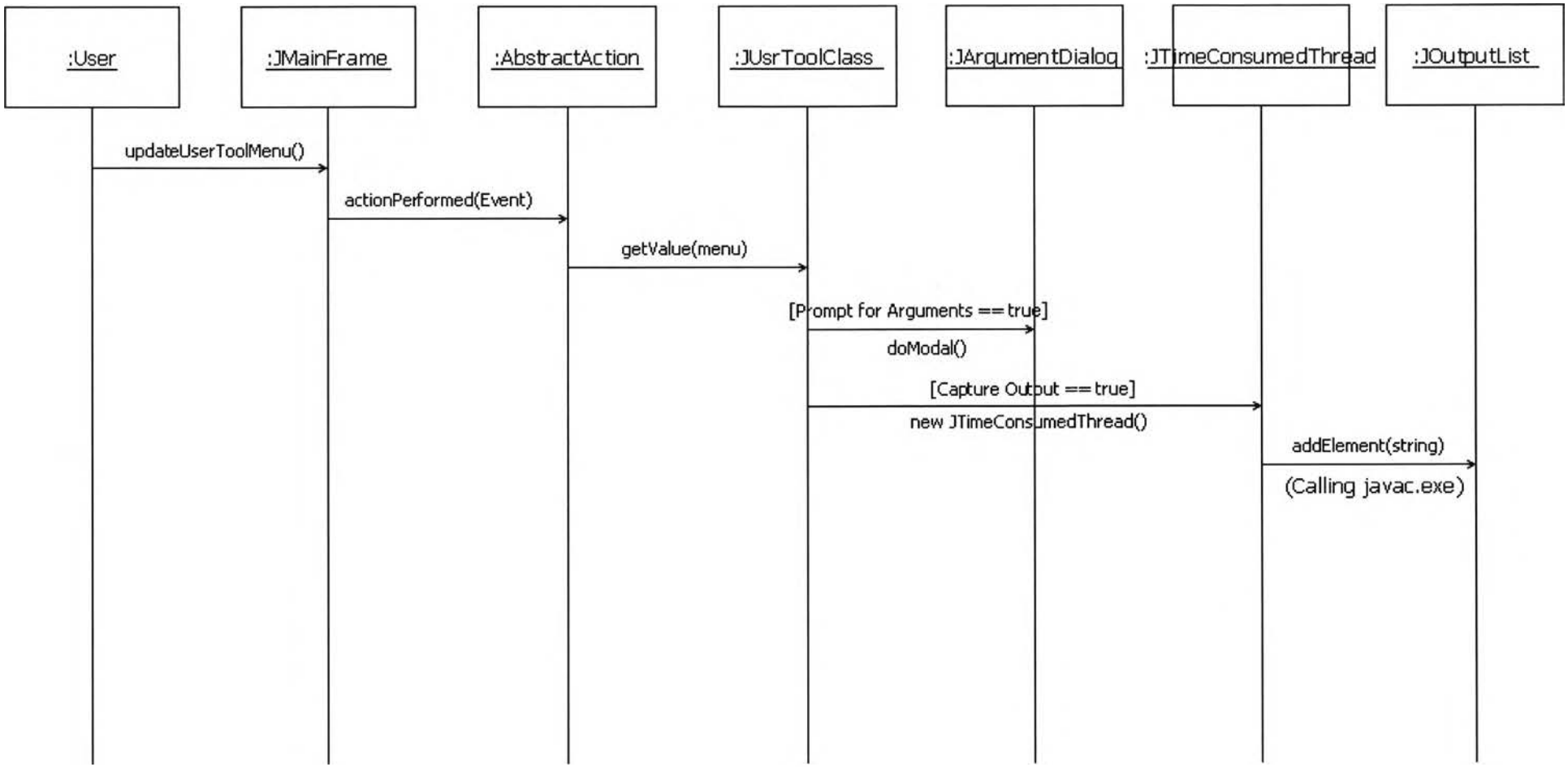
รูปที่ 3.11 เป็นแผนภาพแสดงลำดับการทำงาน ที่แสดงกิจกรรมในยูสเคส การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์ เมื่อผู้ใช้ได้ทำการประมวลผลโปรแกรม Chula OOFP Counting แล้วกิจกรรมในยูสเคส นี้จะเริ่มต้นจากการที่ผู้ใช้ได้เลือกเมนูสำหรับการจัดการโปรเจ็คไฟล์ (ProjectManageProjectAction()) บนหน้าต่างหลักของโปรแกรม (JMainFrame) จากนั้นจะปรากฏหน้าต่างย่อยสำหรับการกำหนดค่าคอนฟิกกูเรชันต่างๆของโปรแกรม (JPreferencesDialog) สำหรับหน้าต่างจัดการโปรเจ็คไฟล์ (createProjectCard()) ผู้ใช้สามารถสร้างโปรเจ็คไฟล์ใหม่ (new JProjectClass(sPrjName)) หรือจัดการโปรเจ็คไฟล์ที่มีอยู่แล้วไม่ว่าจะเป็นการเพิ่มและลบไฟล์ต้นฉบับภาษาจาวาจากโปรเจ็คไฟล์ที่มีอยู่ก็สามารถทำได้โดยผ่านหน้าต่างย่อยนี้ จากนั้นเมื่อผู้ใช้ต้องการเลือกโปรเจ็คไฟล์ที่ตนได้ทำการสร้างไว้แล้วจากขั้นตอนข้างต้น ผู้ใช้ต้องเลือกเมนูสำหรับเปิดโปรเจ็คไฟล์ที่ต้องการ (ProjectSelectProjectAction()) บนหน้าต่างหลักของโปรแกรม (JMainFrame) จากนั้นจะปรากฏหน้าต่างย่อยสำหรับการเลือกโปรเจ็คไฟล์ (JSelectProjectDlg) เมื่อผู้ใช้ได้เลือกโปรเจ็คไฟล์ที่ต้องการแล้ว ชื่อไฟล์ต้นฉบับภาษาจาวาทั้งหมดที่อยู่ในโปรเจ็คไฟล์จะปรากฏบนหน้าต่างแสดงลิสต์ของชื่อไฟล์ (JFilePage) โดยผ่านทางเมธอด updateProjectBrowsePane() และข้อมูลคลาสทั้งหมดที่มีในโปรเจ็คไฟล์ซึ่งประกอบไปด้วยแอตริบิวและเมธอด จะปรากฏบนหน้าต่างแสดงข้อมูลเกี่ยวกับคลาส (JClassPage) โดยผ่านทางเมธอด updateProjectBrowsePane() เช่นเดียวกัน

3.5.3 การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา (Compile Java Source Code)

รูปที่ 3.12 เป็นแผนภาพแสดงลำดับการทำงาน ที่แสดงกิจกรรมในยูสเคส การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา เมื่อผู้ใช้ได้ทำการประมวลผลโปรแกรม Chula OOFP Counting แล้วกิจกรรมในยูสเคส นี้จะเริ่มต้นจากผู้ใช้เลือกเมนูสำหรับการคอมไพล์ไฟล์ต้นฉบับภาษาจาวา (updateUserToolMenu()) บนหน้าต่างหลักของโปรแกรม (JMainFrame) จากนั้นเมธอด actionPerformed(Event) ของคลาส AbstractAction จะถูกเรียกใช้เพื่อตอบสนองการถูกเรียกใช้งาน ภายในเมธอด actionPerformed(Event) จะพิจารณาว่าเมนูที่ถูกเลือกเป็นเมนูสำหรับการคอมไพล์ไฟล์ต้นฉบับภาษาจาวาหรือไม่ เหตุที่ต้องตรวจสอบเนื่องจากเมนูดังกล่าวเป็นหนึ่งในเมนูที่ถูกเพิ่มเติมขึ้นมาสำหรับโปรแกรมอัตถะประโยชน์ใดๆที่กำหนดขึ้นโดยผู้ใช้โปรแกรม (สำหรับโปรแกรมคอมไพเลอร์ภาษาจาวาและโปรแกรมจาวาเวอร์ชวลแมชชีนจะถูกกำหนดขึ้นก่อนเพื่ออำนวยความสะดวกแก่ผู้ใช้ในกรณีที่ต้องการคอมไพล์และรันโปรแกรมจาวาบนเครื่องมือวัดนี้โดยตรง) เพราะฉะนั้นเมธอด actionPerformed(Event) จะเป็นช่องทางร่วมสำหรับเมนูของโปรแกรมอัตถะประโยชน์เหล่านี้ซึ่งรวมถึงคอมไพเลอร์ภาษาจาวาด้วย จากนั้นจะทำการตรวจสอบว่าได้มีการกำหนดให้แสดงหน้าต่างย่อยสำหรับรับพารามิเตอร์พิเศษ (JArgumentDialog) หรือไม่ (การกำหนดนี้จะกระทำผ่านทางหน้าต่างย่อยสำหรับการกำหนดค่าคอนฟิกต่างๆของโปรแกรม (JPreferencesDialog)) ถ้าใช่ ([Prompt for Arguments == true]) จะปรากฏหน้าต่างย่อยนี้ (doModal()) ขึ้นมา จากนั้นจะทำการตรวจสอบอีกว่าต้องการแสดงข้อความที่เป็นผลลัพธ์จากการรันโปรแกรมคอมไพเลอร์ภาษาจาวาหรือไม่ ถ้าใช่ ([Capture Output == true]) จะปรากฏข้อความดังกล่าวบนหน้าต่างแสดงรายละเอียดของผลการทำงาน (JOutputList) โดยผ่านทางเมธอด addElement(string) ซึ่งข้อความดังกล่าวเกิดขึ้นจากการประมวลผลโปรแกรม javac.exe กับไฟล์ต้นฉบับภาษาจาวานั่นเอง



รูปที่ 3.11 แผนภาพแสดงลำดับการทำงานสำหรับยูสเคส การสร้าง การเลือก และการจัดการโปรเจ็คไฟล์



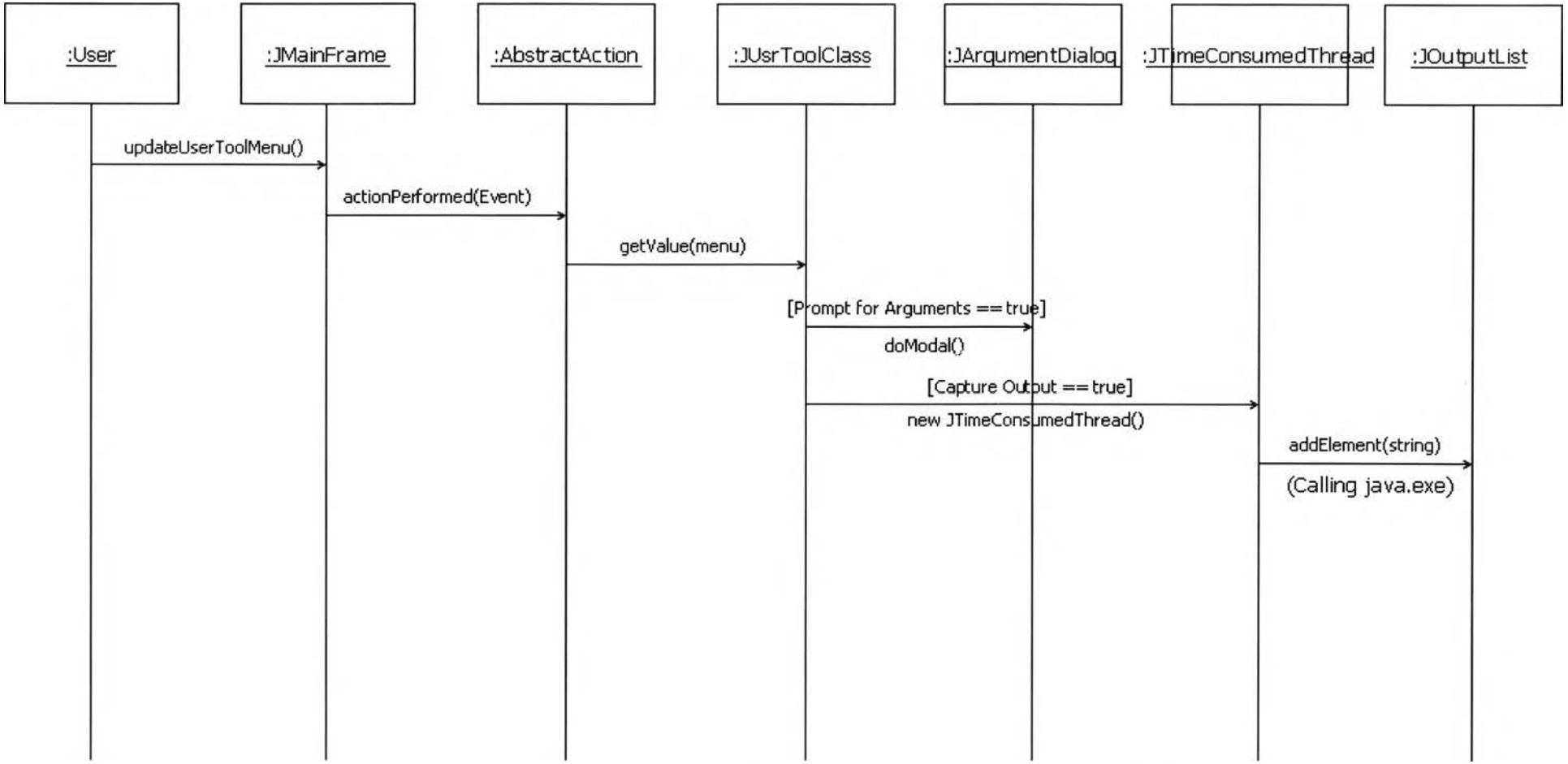
รูปที่ 3.12 แผนภาพแสดงลำดับการทำงานสำหรับยูสเคส การคอมไพล์ไฟล์ต้นฉบับภาษาจาวา

3.5.4 การประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา (Execute Java Program)

รูปที่ 3.13 เป็นแผนภาพแสดงลำดับการทำงาน ที่แสดงกิจกรรมในยูสเคส การประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา เมื่อผู้ใช้ประมวลผลโปรแกรม Chula OOFP Counting แล้วกิจกรรมในยูสเคส นี้จะเริ่มต้นจากผู้ใช้เลือกเมนูสำหรับการประมวลผลโปรแกรมจาวา (updateUserToolMenu()) บนหน้าต่างหลักของโปรแกรม (JMainFrame) จากนั้นเมธอด actionPerformed(Event) ของคลาส AbstractAction จะถูกเรียกใช้เพื่อตอบรับกับการถูกเลือกจากเมนูนี้ ภายในเมธอด actionPerformed(Event) จะพิจารณาว่าเมนูที่ถูกเลือกเป็นเมนูสำหรับการประมวลผลโปรแกรมจาวาหรือไม่ เหตุที่ต้องตรวจสอบเนื่องจากเมนูดังกล่าวเป็นหนึ่งในเมนูที่ถูกเพิ่มเติมขึ้นมาสำหรับโปรแกรมอัตถะประโยชน์ใดๆที่กำหนดขึ้นโดยผู้ใช้โปรแกรม (แต่สำหรับโปรแกรมคอมไพเลอร์ภาษาจาวาและโปรแกรมจาวาเวอร์ชวลแมชชีนจะถูกกำหนดขึ้นก่อนเพื่ออำนวยความสะดวกแก่ผู้ใช้ในกรณีที่ต้องการคอมไพล์และประมวลผลโปรแกรมจาวาบนเครื่องมือนี้โดยตรง) เพราะฉะนั้นเมธอด actionPerformed(Event) จะเป็นช่องทางร่วมสำหรับเมนูโปรแกรมอัตถะประโยชน์เหล่านี้ซึ่งรวมถึงโปรแกรมจาวาเวอร์ชวลแมชชีนด้วย จากนั้นจะทำการตรวจสอบว่าได้มีการกำหนดให้แสดงหน้าต่างย่อยสำหรับรับพารามิเตอร์พิเศษ (JArgumentDialog) หรือไม่ (การกำหนดนี้จะกระทำผ่านทางหน้าต่างย่อยสำหรับการกำหนดค่าคอนฟิกต่างๆของโปรแกรม (JPreferencesDialog)) ถ้าใช่ ([Prompt for Arguments == true]) จะปรากฏหน้าต่างย่อยนี้ (doModal()) ขึ้นมา จากนั้นจะทำการตรวจสอบอีกว่าต้องการแสดงข้อความที่เป็นผลลัพธ์จากการประมวลผลโปรแกรมจาวาเวอร์ชวลแมชชีนหรือไม่ ถ้าใช่ ([Capture Output == true]) จะปรากฏข้อความดังกล่าวบนหน้าต่างแสดงรายละเอียดของผลการทำงาน (JOutputList) โดยผ่านทางเมธอด addElement(string) ซึ่งข้อความดังกล่าวเกิดขึ้นจากการประมวลผลโปรแกรม java.exe กับไฟล์จาวาคลาสนั่นเอง

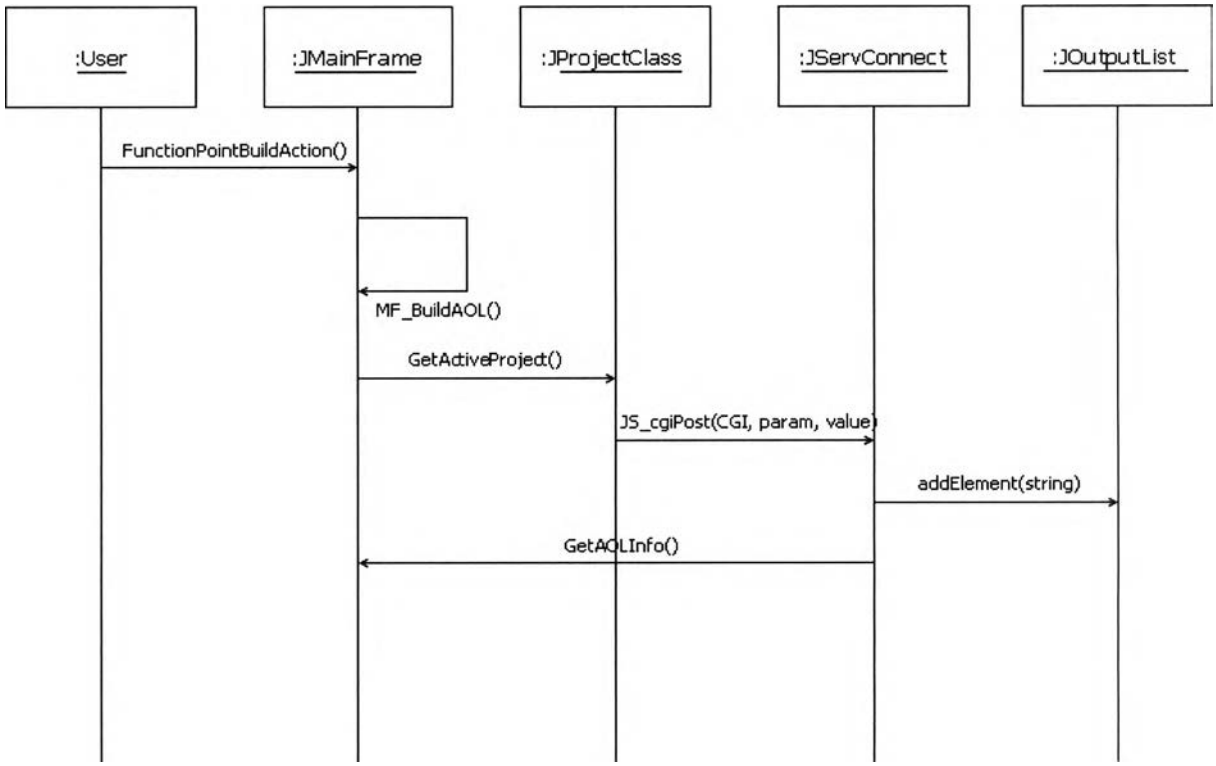
3.5.5 การสร้างข้อมูลภาษาเอโอแอล (Abstract Object Language: AOL) จากไฟล์ต้นฉบับภาษาจาวาในโปรเจ็คไฟล์ (Build Abstract Object Language)

รูปที่ 3.14 เป็นแผนภาพแสดงลำดับการทำงาน ที่แสดงกิจกรรมในยูสเคส การสร้างข้อมูลภาษาเอโอแอลจากไฟล์ต้นฉบับภาษาจาวา เมื่อผู้ใช้ได้ทำการประมวลผลโปรแกรม Chula OOFP Counting แล้วกิจกรรมในยูสเคส นี้จะเริ่มต้นจากผู้ใช้เลือกเมนูสำหรับการสร้างข้อมูลภาษาเอโอแอล (FunctionPointBuildAction()) บนหน้าต่างหลักของโปรแกรม (JMainFrame) เมธอด MF_BuildAOL() จะถูกเรียกใช้งานต่อเพื่อเตรียมการสร้างข้อมูลภาษา เอโอแอล โดยเริ่มจากการพิจารณาโปรเจ็คไฟล์ที่กำลังถูกเปิดใช้งานอยู่ (GetActiveProject()) สิ่งสำคัญที่ต้องเตรียมคือลิสต์ของชื่อไฟล์ต้นฉบับภาษาจาวาทั้งหมดที่มีในโปรเจ็ค จากนั้นจึงทำการสร้างการเชื่อมต่อเข้ากับเครื่องให้บริการเว็บ (JS_cgiPost(CGI, param, value)) โดยใช้วัตถุ JServConnect และจัดการส่งลิสต์ของรายชื่อไฟล์พร้อมกับโค้ดต้นฉบับภาษาจาวาไปสู่เครื่องให้บริการเว็บ จากนั้นเครื่องให้บริการเว็บจะส่งผ่านการทำงานไปสู่ไฟล์ชุดคำสั่งภาษาเพิร์ลอีกทอดหนึ่งซึ่งอยู่ในรูปของโมดูลประมวลผลฝั่งเครื่องให้บริการ (Common Gateway Interface – CGI) ซึ่งทำหน้าที่ในการวิเคราะห์ สร้าง และส่งกลับข้อมูลภาษาเอโอแอลที่ได้จากการวิเคราะห์ ในขณะเดียวกันการทำงานเหล่านี้จะแสดงข้อความบ่งบอกสถานะการทำงานอยู่ตลอดเวลาเพื่อแจ้งผู้ใช้ให้ทราบถึงความคืบหน้าของการดำเนินงานโดยจะปรากฏข้อความดังกล่าวบนหน้าต่างแสดงรายละเอียดของผลการทำงาน (JOutputList) โดยผ่านทางเมธอด addElement(string)



รูปที่ 3.13 แผนภาพแสดงลำดับการทำงานสำหรับยูสเคส การประมวลผลโปรแกรมที่พัฒนาด้วยภาษาจาวา

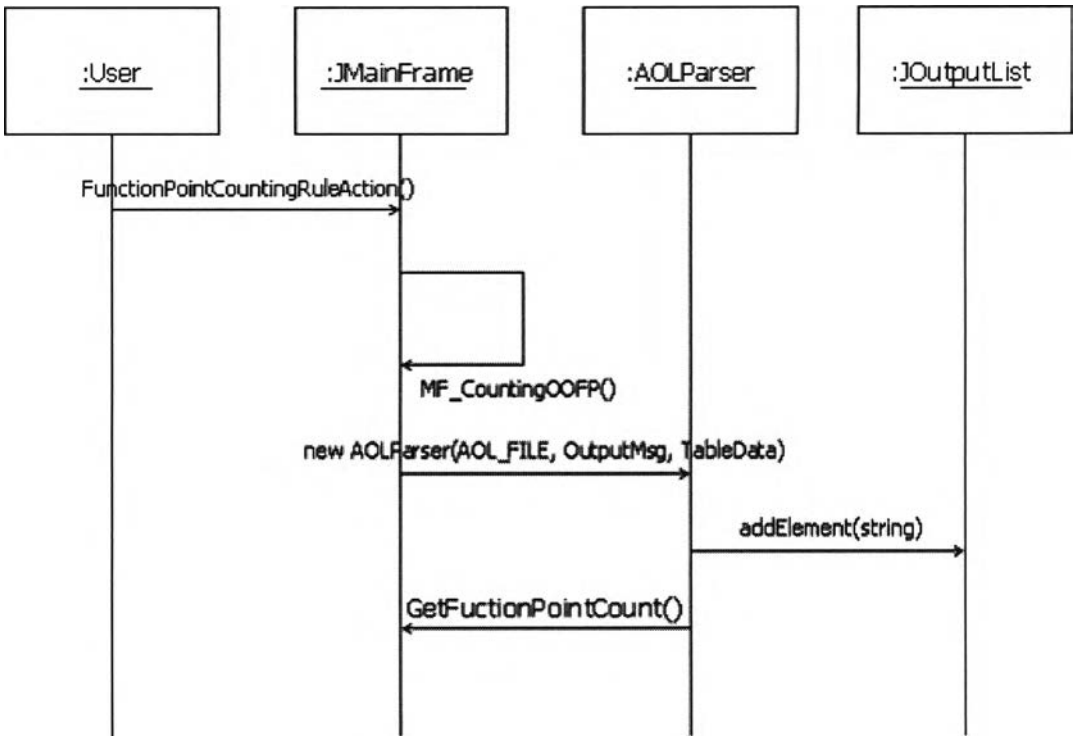
และท้ายสุดเมื่อการวิเคราะห์และสร้างข้อมูลภาษาเอโอแอลเสร็จสิ้น ชุดคำสั่งภาษาเพิร์ลจะทำการส่งข้อมูลภาษาเอโอแอลนี้กลับสู่โปรแกรมไคลเอนต์ Chula OOFP Counting เพื่อใช้ในการนับจำนวนของฟังก์ชันพอยต์ต่อไป



รูปที่ 3.14 แผนภาพแสดงลำดับการทำงานสำหรับยูสเคส การสร้างข้อมูลภาษาเอโอแอล

3.5.6 การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล (Count Object Oriented Function Point)

รูปที่ 3.15 เป็นแผนภาพแสดงลำดับการทำงาน ที่แสดงกิจกรรมในยูสเคส การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล เมื่อผู้ใช้ได้ทำการประมวลผลโปรแกรม Chula OOFP Counting แล้ว กิจกรรมในยูสเคส นี้จะเริ่มต้นจากผู้ใช้เลือกเมนูสำหรับการนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากภาษาเอโอแอล (FunctionPointCountingRuleAction()) บนหน้าต่างหลักของโปรแกรม (JMainFrame) เมธอด MF_CountingOOFP() จะถูกเรียกใช้งานต่อเพื่อเตรียมการการนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุ โดยเริ่มจากการสร้างวัตถุ AOLParser ขึ้นมา (new AOLParser(AOL_FILE, OutputMsg, TableData)) จากนั้นส่งไฟล์ข้อมูลภาษาเอโอแอลซึ่งได้จากการทำงานในหัวข้อ 3.5.5 ในระหว่างขั้นตอนการนับ จะปรากฏข้อความบ่งบอกสถานะการทำงานอยู่ตลอดเวลาเพื่อแจ้งผู้ใช้ให้ทราบถึงความคืบหน้าของการดำเนินงานโดยจะปรากฏข้อความดังกล่าวบนหน้าต่างแสดงรายละเอียดของผลการทำงาน (JOutputList) โดยผ่านทางเมธอด addElement(string) และท้ายสุดเมื่อการการนับเสร็จสิ้น จำนวนฟังก์ชันพอยต์เชิงวัตถุของแต่ละคลาสและของซอฟต์แวร์จะแสดงออกมาในหน้าต่างย่อยแสดงรายละเอียดของผลการทำงาน (JOutputList)



รูปที่ 3.15 แผนภาพแสดงลำดับการทำงานสำหรับยูสเคส การนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากข้อมูลภาษาเอโอแอล

3.6 แผนภาพแสดงกิจกรรม (Activity Diagram)

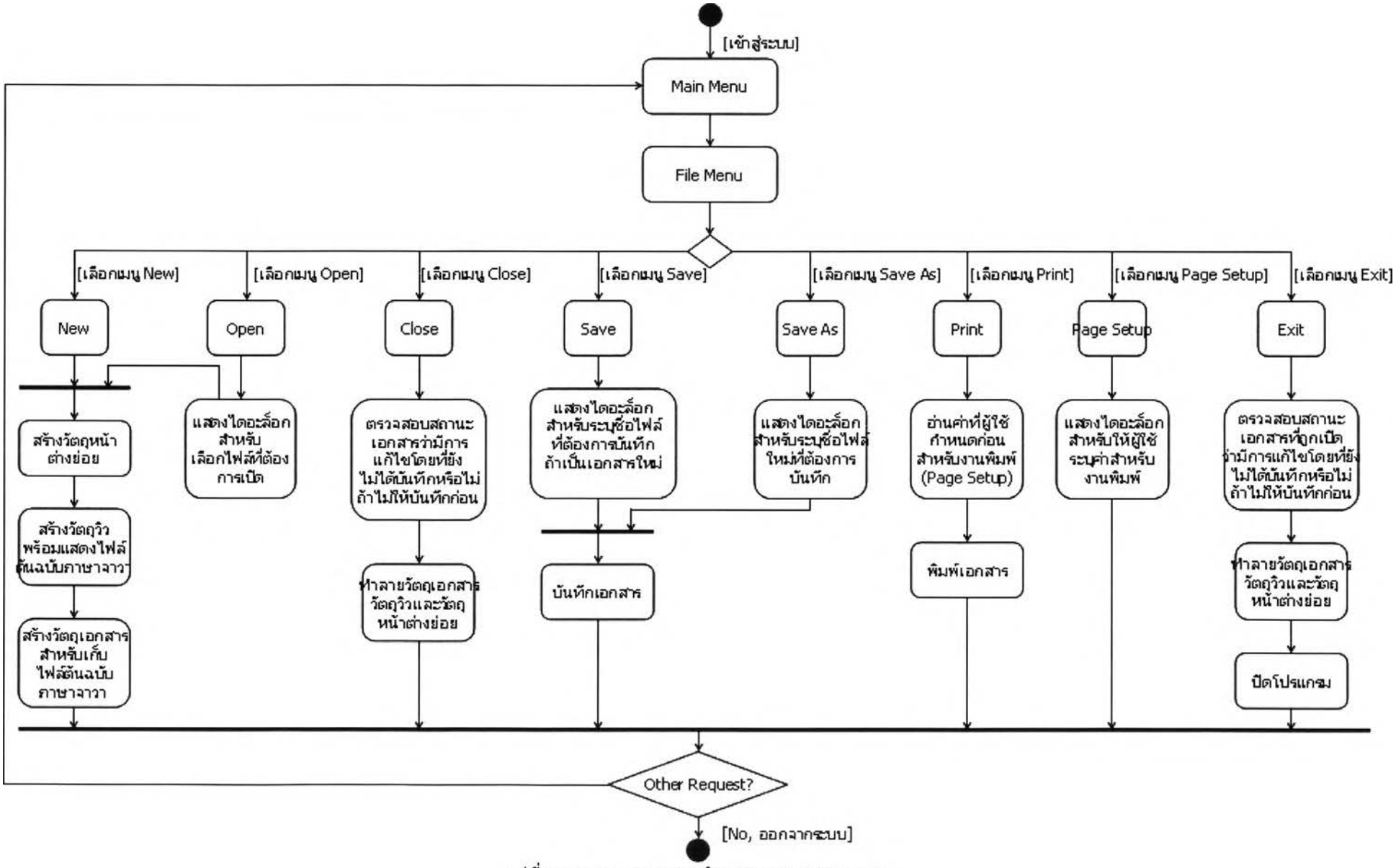
สำหรับแผนภาพแสดงกิจกรรม ผู้วิจัยได้ทำการพิจารณาแบ่งตามกลุ่มของเมนูการใช้งานเพื่อให้ง่ายต่อการวิเคราะห์และออกแบบระบบโดยโปรแกรม Chula OOFP Counting นี้ประกอบไปด้วยกลุ่มของเมนู File, Edit, Project, Function Point, Tools, Window และ Help ซึ่งทั้งหมดแบ่งตามหน้าที่การใช้งานเพื่อให้สะดวกและเข้าใจได้ง่ายต่อผู้ใช้

3.6.1 แผนภาพแสดงกิจกรรมสำหรับเมนู File

รูปที่ 3.16 แสดงแผนภาพกิจกรรมของเมนู File ซึ่งประกอบไปด้วย 8 เมนูย่อย มีรายละเอียดดังนี้

3.6.1.1 เมนู New

เมื่อผู้ใช้เลือกเมนู New แล้ว จะทำการสร้างวัตถุหน้าต่างย่อย (JMDIChildWnd) ซึ่งภายในวัตถุนี้มีวัตถุส่วนประกอบที่สำคัญคือวัตถุอิดิตวิว (JEditView) ซึ่งทำหน้าที่แสดงไฟล์และพร้อมรับการแก้ไขไฟล์ต้นฉบับภาษาจาวา นอกจากนี้ภายในวัตถุอิดิตวิวยังมีวัตถุที่สำคัญคือวัตถุเอกสาร (JStyledDocument) ซึ่งทำหน้าที่เก็บข้อมูลไฟล์ต้นฉบับภาษาจาวาซึ่งผู้ใช้กำลังเปิดใช้งานอยู่



รูปที่ 3.16 แผนภาพแสดงกิจกรรมสำหรับเมนู File

3.6.1.2 เมนู Open

เมื่อเมนู Open ถูกเรียกใช้งาน วัตถุหน้าต่างย่อยสำหรับการเลือกไฟล์ (JFileChooser) จะถูกสร้างขึ้นเพื่อให้ผู้ใช้เลือกไฟล์ต้นฉบับภาษาจาวาที่ต้องการ จากนั้นทำการสร้างวัตถุหน้าต่างย่อย ซึ่งภายในวัตถุนี้มีส่วนประกอบที่สำคัญคือวัตถุอิดิตวิวซึ่งทำหน้าที่แสดงไฟล์ที่ผู้ใช้เลือกและสร้างวัตถุเอกสารเพื่อเก็บข้อมูลไฟล์ที่ถูกเปิดขึ้น

3.6.1.3 เมนู Close

เมื่อเมนู Close ถูกเรียกใช้งาน จะมีการตรวจสอบสถานะของวัตถุเอกสารว่ามีการแก้ไขโดยที่ยังไม่ได้บันทึกหรือไม่ ถ้ายังไม่ได้บันทึกต้องทำการบันทึกเอกสารก่อน จากนั้นจึงทำลายวัตถุเอกสาร วัตถุอิดิตวิว และวัตถุหน้าต่างย่อย ตามลำดับ

3.6.1.4 เมนู Save

เมื่อเมนู Save ถูกเรียกใช้งาน จะมีการตรวจสอบสถานะของวัตถุเอกสารว่ามีการตั้งชื่อเอกสารที่ผู้ใช้กำลังทำงานอยู่หรือไม่ ถ้ายังไม่มีการตั้งชื่อเอกสาร วัตถุหน้าต่างย่อยสำหรับการกำหนดชื่อไฟล์ (JFileChooser) จะถูกสร้างขึ้นเพื่อให้ผู้ใช้กำหนดชื่อไฟล์ตามต้องการ จากนั้นจึงทำการบันทึกข้อมูลซึ่งถูกเก็บอยู่ในวัตถุเอกสารลงไฟล์

3.6.1.5 เมนู Save As

เมื่อเมนู Save As ถูกเรียกใช้งาน วัตถุหน้าต่างย่อยสำหรับการกำหนดชื่อไฟล์ (JFileChooser) จะถูกสร้างขึ้นเพื่อให้ผู้ใช้กำหนดชื่อไฟล์ใหม่ตามต้องการ จากนั้นจึงทำการบันทึกข้อมูลซึ่งถูกเก็บอยู่ในวัตถุเอกสารลงไฟล์ที่ผู้ใช้กำหนด

3.6.1.6 เมนู Print

เมื่อเมนู Print ถูกเรียกใช้งาน จะมีการอ่านค่าที่ผู้ใช้ได้กำหนดขึ้นก่อนสำหรับงานพิมพ์ ซึ่งค่าดังกล่าวได้มาจากวัตถุหน้าต่างย่อยสำหรับระบุค่างานพิมพ์ (PrinterJob) ที่ผู้ใช้กำหนด วัตถุฟอร์แมตงานพิมพ์ (Book) จะถูกสร้างขึ้น เพื่อเตรียมหน้าเอกสารก่อนพิมพ์ จากนั้นจะทำการอ่านข้อมูลจากวัตถุเอกสาร และสั่งพิมพ์ข้อมูลออกทางเครื่องพิมพ์

3.6.1.7 เมนู Page Setup

เมื่อเมนู Page Setup ถูกเรียกใช้ วัตถุหน้าต่างย่อยสำหรับระบุค่างานพิมพ์ จะถูกสร้างขึ้นเพื่อให้ผู้ใช้ได้กำหนดรูปแบบงานพิมพ์ ซึ่งการกำหนดค่างานพิมพ์นี้สัมพันธ์กับการใช้งานเมนู Print

3.6.1.8 เมนู Exit

เมื่อเมนู Exit ถูกเรียกใช้ จะมีการตรวจสอบสถานะของวัตถุเอกสารทั้งหมดที่ถูกสร้างขึ้น ว่ามีการแก้ไขโดยที่ยังไม่ได้ทำการบันทึกหรือไม่ ถ้ายังไม่ได้มีการบันทึกให้ทำการบันทึกวัตถุเอกสารนั้นเสียก่อน โดยขั้นตอนการบันทึกจะกระทำเช่นเดียวกับการทำงานของเมนู Save จากนั้นวัตถุเอกสาร วัตถุอิดิตวิว วัตถุหน้าต่างย่อยและวัตถุหน้าต่างหลัก (JMainFrame) จะถูกทำลายทั้งหมด เพื่อทำการปิดโปรแกรม

3.6.2 แผนภาพแสดงกิจกรรมสำหรับเมนู Edit

รูปที่ 3.17 แสดงแผนภาพกิจกรรมของเมนู Edit ซึ่งประกอบไปด้วย 6 เมนูย่อย มีรายละเอียดดังนี้

3.6.2.1 เมนู Undo

เมื่อเมนู Undo ถูกเรียกใช้ วัตถุ UndoManager ซึ่งเป็นส่วนประกอบในวัตถุหน้าต่างย่อยจะถูกสร้างขึ้นเพื่อจัดการงานเกี่ยวกับการ Undo ทั้งหมด วัตถุที่เกี่ยวข้องหลักสำหรับเมนู Undo คือวัตถุอิดิตวิวและวัตถุเอกสาร โดยหน้าที่หลักของเมนู Undo คือการกู้ข้อมูลเก่าบนวัตถุเอกสารก่อนการเปลี่ยนแปลง

3.6.2.2 เมนู Redo

เมื่อเมนู Redo ถูกเรียกใช้งาน วัตถุอิดิตวิวและวัตถุเอกสารจะทำงานร่วมกันเพื่อให้สามารถทำซ้ำข้อมูลล่าสุดบนวัตถุเอกสารได้โดยที่วัตถุอิดิตวิวแสดงข้อมูลอย่างถูกต้อง

3.6.2.3 เมนู Cut-to-clipboard

เมื่อเมนู Cut-to-clipboard ถูกเรียกใช้งาน วัตถุอิดิตวิวและวัตถุเอกสารจะทำงานร่วมกันเพื่อให้สามารถตัดข้อมูลที่ผู้ใช้เลือกบนวัตถุเอกสารขึ้นคลิ๊ปบอร์ดโดยที่วัตถุอิดิตวิวแสดงข้อมูลอย่างถูกต้อง

3.6.2.4 เมนู Copy-to-clipboard

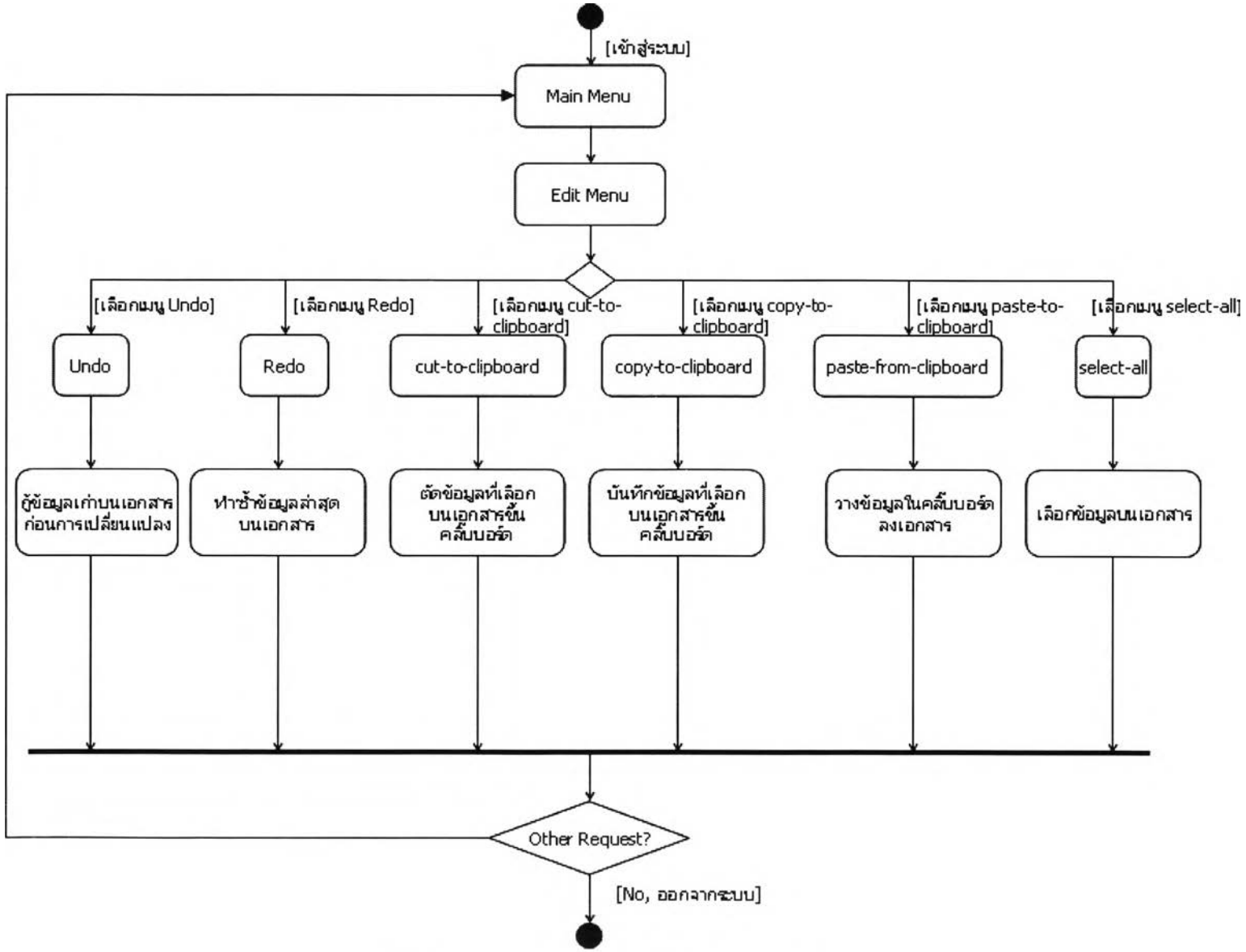
เมื่อเมนู Copy-to-clipboard ถูกเรียกใช้งาน วัตถุอิดิตวิวและวัตถุเอกสารจะทำงานร่วมกันเพื่อให้สามารถบันทึกข้อมูลที่ผู้ใช้เลือกบนวัตถุเอกสารขึ้นคลิ๊ปบอร์ดได้

3.6.2.5 เมนู Paste-from-clipboard

เมื่อเมนู Paste-from-clipboard ถูกเรียกใช้งาน วัตถุอิดิตวิวและวัตถุเอกสารจะทำงานร่วมกันเพื่อให้สามารถวางข้อมูลที่อยู่ในคลิ๊ปบอร์ดปรากฏบนวัตถุเอกสารได้

3.6.2.6 เมนู Select-all

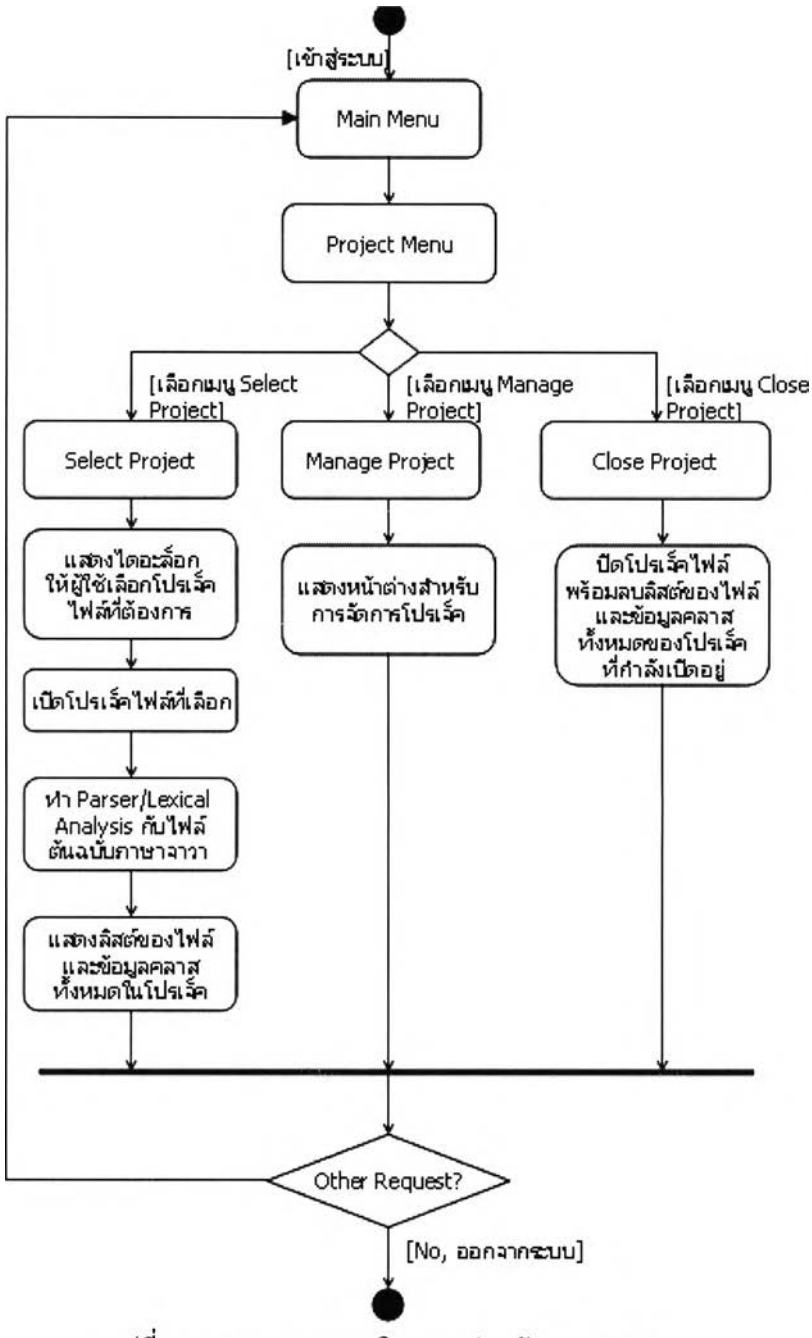
เมื่อเมนู Select-all ถูกเรียกใช้งาน วัตถุอิดิตวิวและวัตถุเอกสารจะทำงานร่วมกันเพื่อให้สามารถเลือกข้อมูลบนวัตถุเอกสารได้โดยปรากฏเงาแสดงข้อมูลที่ถูกเลือกอย่างเด่นชัดบนวัตถุอิดิตวิว



รูปที่ 3.17 แผนภาพแสดงกิจกรรมสำหรับเมนู Edit

3.6.3 แผนภาพแสดงกิจกรรมสำหรับเมนู Project

รูปที่ 3.18 แสดงแผนภาพกิจกรรมของเมนู Project ซึ่งประกอบไปด้วย 3 เมนูย่อย มีรายละเอียดดังนี้



รูปที่ 3.18 แผนภาพแสดงกิจกรรมสำหรับเมนู Project

3.6.3.1 เมนู Select Project

เมื่อเมนู Select Project ถูกเรียกใช้ วัตถุหน้าต่างย่อยสำหรับเลือกโปรเจ็ค (JSelectProjectDlg) ถูกสร้าง เพื่อให้ผู้ใช้ได้เลือกโปรเจ็คไฟล์ที่ต้องการ และทำการเปิดโปรเจ็คไฟล์ พร้อมกันนี้คลาส JavaParser จะถูกสร้าง เพื่อทำ Parser และ Lexical Analysis กับไฟล์ต้นฉบับภาษาจาวาที่ระบุในโปรเจ็คไฟล์ที่เลือก เพื่อเก็บข้อมูลคลาส แอตริบิวและเมธอดที่มีทั้งหมด ข้อมูลดังกล่าวนี้จะแสดงบนวัตถุ JFilePage และ วัตถุ JClassPage สำหรับลิสต์

ของไฟล์ภาษาจาวาและข้อมูลคลาสทั้งหมดในโปรเจ็คตามลำดับ ข้อมูลของโปรเจ็คซึ่งได้แก่ ชื่อโปรเจ็คไฟล์และลิสต์ของชื่อไฟล์ภาษาจาวาจะถูกเก็บในวัตถุ JProjectClass

3.6.3.2 เมนู Manage Project

เมื่อเมนู Manage Project ถูกเรียกใช้ วัตถุหน้าต่างย่อยแสดงหน้าต่างสำหรับการจัดการโปรเจ็คจะถูกสร้าง (JPreferencesDialog) เพื่อให้ผู้ใช้ได้ทำการเพิ่มหรือลบไฟล์ต้นฉบับภาษาจาวาในโปรเจ็ค การแก้ไขเหล่านี้จะส่งผลกระทบต่อวัตถุ JProjectClass

3.6.3.3 เมนู Close project

เมื่อเมนู Close project ถูกเรียกใช้ หมายถึงการปิดโปรเจ็คไฟล์พร้อมกับการลบลิสต์ของไฟล์ต้นฉบับภาษาจาวาและข้อมูลคลาสทั้งหมดของโปรเจ็คที่ต้องการปิด

3.6.4 แผนภาพแสดงกิจกรรมสำหรับเมนู Function Point

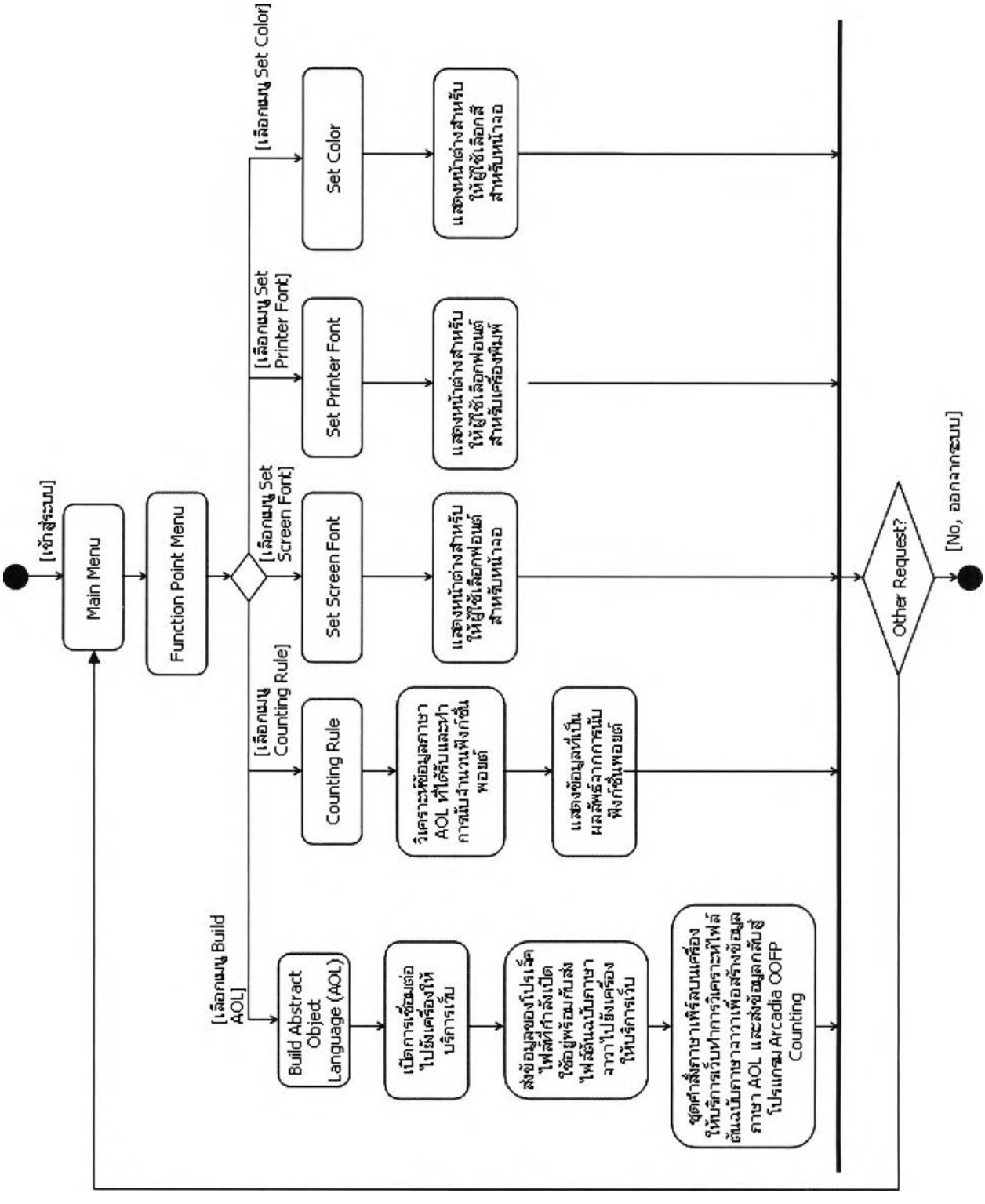
รูปที่ 3.19 แสดงแผนภาพกิจกรรมของเมนู Function Point ซึ่งประกอบไปด้วย 5 เมนูย่อย มีรายละเอียดดังนี้

3.6.4.1 เมนู Build Abstract Object Language (AOL)

เมื่อเมนู Build Abstract Object Language (AOL) ถูกเรียกใช้ วัตถุ JServConnect จะถูกสร้างและเปิดการเชื่อมต่อไปยังเครื่องให้บริการเว็บ จากนั้นจะส่งข้อมูลของวัตถุ JProjectClass ซึ่งได้แก่ชื่อโปรเจ็คและลิสต์ของชื่อไฟล์ต้นฉบับภาษาจาวาในโปรเจ็ค พร้อมกับส่งไฟล์ข้อมูลภาษาจาวาไปยังเครื่องให้บริการเว็บ จากนั้นชุดคำสั่งภาษาเพิร์ลบนเครื่องให้บริการเว็บทำการวิเคราะห์ไฟล์ภาษาจาวาเพื่อสร้างข้อมูลภาษา เอโอแอล และส่งข้อมูลภาษา เอโอแอล นั้นกลับสู่ไคลเอนต์ซึ่งได้แก่โปรแกรม Chula OOFP Counting

3.6.4.2 เมนู Counting Rule

เมื่อเมนู Counting Rule ถูกเรียกใช้ วัตถุ AOLParser จะถูกสร้างเพื่อทำการวิเคราะห์ข้อมูลภาษาเอโอแอลที่ได้รับจากขั้นตอนการเรียกใช้เมนู Build Abstract Object Language (AOL) และทำการนับจำนวนฟังก์ชันพอยต์ที่มีสำหรับโปรเจ็คที่เปิดใช้อยู่ จากนั้นทำการแสดงข้อมูลซึ่งเป็นผลลัพธ์จากการนับฟังก์ชันพอยต์โดยผ่านทางวัตถุ JOutputList



รูปที่ 3.19 แผนภาพแสดงกิจกรรมสำหรับเมนู Function Point

3.6.4.3 เมนู Set Screen Font

เมื่อเมนู Set Screen Font ถูกเรียกใช้ วัตถุหน้าต่างย่อยสำหรับเลือกฟอนต์ที่ใช้แสดงบนหน้าจอจะถูกสร้างและพร้อมให้ผู้ใช้เลือกรูปแบบฟอนต์ตามต้องการ วัตถุวิวจะแสดงข้อมูลภาษาจาวาตามรูปแบบฟอนต์ที่ผู้ใช้เลือก

3.6.4.4 เมนู Set Printer Font

เมื่อเมนู Set Printer Font ถูกเรียกใช้ วัตถุหน้าต่างย่อยสำหรับเลือกฟอนต์สำหรับงานพิมพ์จะถูกสร้างและพร้อมให้ผู้ใช้เลือกรูปแบบฟอนต์ตามต้องการ ซึ่งรูปแบบฟอนต์นี้จะถูกใช้เมื่อเมนู Print ถูกเรียกใช้งาน

3.6.4.5 เมนู Set Color

เมื่อเมนู Set Color ถูกเรียกใช้ วัตถุหน้าต่างย่อยสำหรับเลือกสีที่แสดงผลบนหน้าจอ จะถูกสร้างและพร้อมให้ผู้ใช้เลือกสี Foreground และ Background ตามต้องการ วัตถุวิวจะแสดงสีตามที่ผู้ใช้เลือก

3.6.5 แผนภาพแสดงกิจกรรมสำหรับเมนู Tools

รูปที่ 3.20 แสดงแผนภาพกิจกรรมของเมนู Tools ซึ่งประกอบไปด้วย 4 เมนูย่อย มีรายละเอียดดังนี้

3.6.5.1 เมนู Preference

เมื่อเมนู Preference ถูกเรียกใช้ วัตถุหน้าต่างย่อยแสดงหน้าต่างสำหรับให้ผู้ใช้กำหนดค่าคอนฟิกกูเรชั่นต่างๆของระบบ (JPreferencesDialog) จะถูกสร้างและพร้อมให้ผู้ใช้กำหนดค่าคอนฟิกต่างตามต้องการ

3.6.5.2 เมนู Configure User Tools

เมื่อเมนู Configure User Tools ถูกเรียกใช้ จะมีการสร้างวัตถุหน้าต่างย่อยซึ่งแสดงหน้าต่างให้ผู้ใช้เพิ่มหรือลบโปรแกรมอัตถะประโยชน์พิเศษที่ต้องการให้สามารถใช้งานจากโปรแกรม Chula OOFP Counting ได้โดยตรง โดยเมนูของวัตถุหน้าต่างหลัก (JMainFrame) จะสะท้อนให้เห็นถึงการเปลี่ยนแปลงที่ผู้ใช้กำหนด

3.6.5.3 เมนู Build Java Program

เมื่อเมนู Build Java Program ถูกเรียกใช้ วัตถุหน้าต่างหลักจะเรียกใช้งานโปรแกรม javac.exe เพื่อคอมไพล์ไฟล์ต้นฉบับภาษาจาวามีทั้งหมดในโปรเจ็คที่เปิดใช้อยู่

3.6.5.4 เมนู Execute Java Program

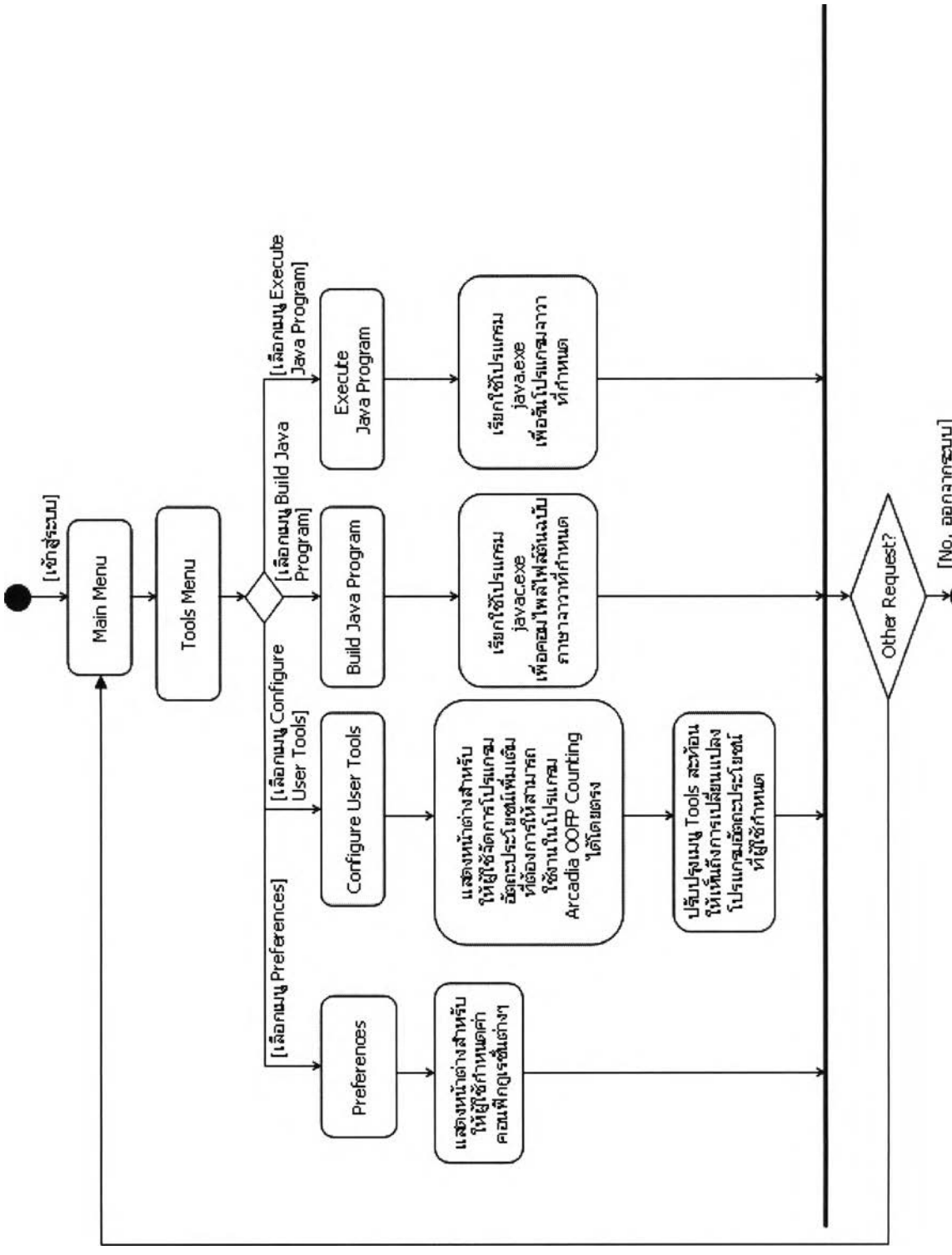
เมื่อเมนู Execute Java Program ถูกเรียกใช้ วัตถุหน้าต่างหลักจะเรียกใช้งานโปรแกรม java.exe เพื่อรันโปรแกรมจาวาที่กำหนด

3.6.6 แผนภาพแสดงกิจกรรมสำหรับเมนู Window

รูปที่ 3.21 แสดงแผนภาพกิจกรรมของเมนู Window ซึ่งประกอบไปด้วย 5 เมนูย่อย มีรายละเอียดดังนี้

3.6.6.1 เมนู Cascade

เมื่อเมนู Cascade ถูกเรียกใช้ วัตถุหน้าต่างย่อยทั้งหมดจะถูกเรียงใหม่แบบ Cascade



รูปที่ 3.20 แผนภาพแสดงกิจกรรมสำหรับเมนู Tools

3.6.6.2 เมนู Tile

เมื่อเมนู Tile ถูกเรียกใช้ วัตถุหน้าต่างย่อยทั้งหมดจะถูกเรียงใหม่แบบ Tile

3.6.6.3 เมนู Close All

เมื่อเมนู Close All ถูกเรียกใช้ จะมีการตรวจสอบสถานะของวัตถุเอกสารว่ามีการแก้ไขโดยที่ยังไม่ได้บันทึกหรือไม่ ถ้ายังไม่มีการบันทึกข้อมูลในวัตถุเอกสารให้บันทึกลงไฟล์ก่อน จากนั้นวัตถุเอกสาร วัตถุวิว และวัตถุหน้าต่างย่อยทั้งหมดจะถูกทำลาย

3.6.6.4 เมนู Minimize All

เมื่อเมนู Minimize All ถูกเรียกใช้ จะทำให้วัตถุหน้าต่างย่อยทั้งหมดเปลี่ยนเป็นรูปไอคอน

3.6.6.5 เมนู Restore All

เมื่อเมนู Restore All ถูกเรียกใช้ ทำให้วัตถุหน้าต่างย่อยทั้งหมดกลับมาอยู่ในขนาดและตำแหน่งเดิมตามปกติ

3.6.7 แผนภาพแสดงกิจกรรมสำหรับเมนู Help

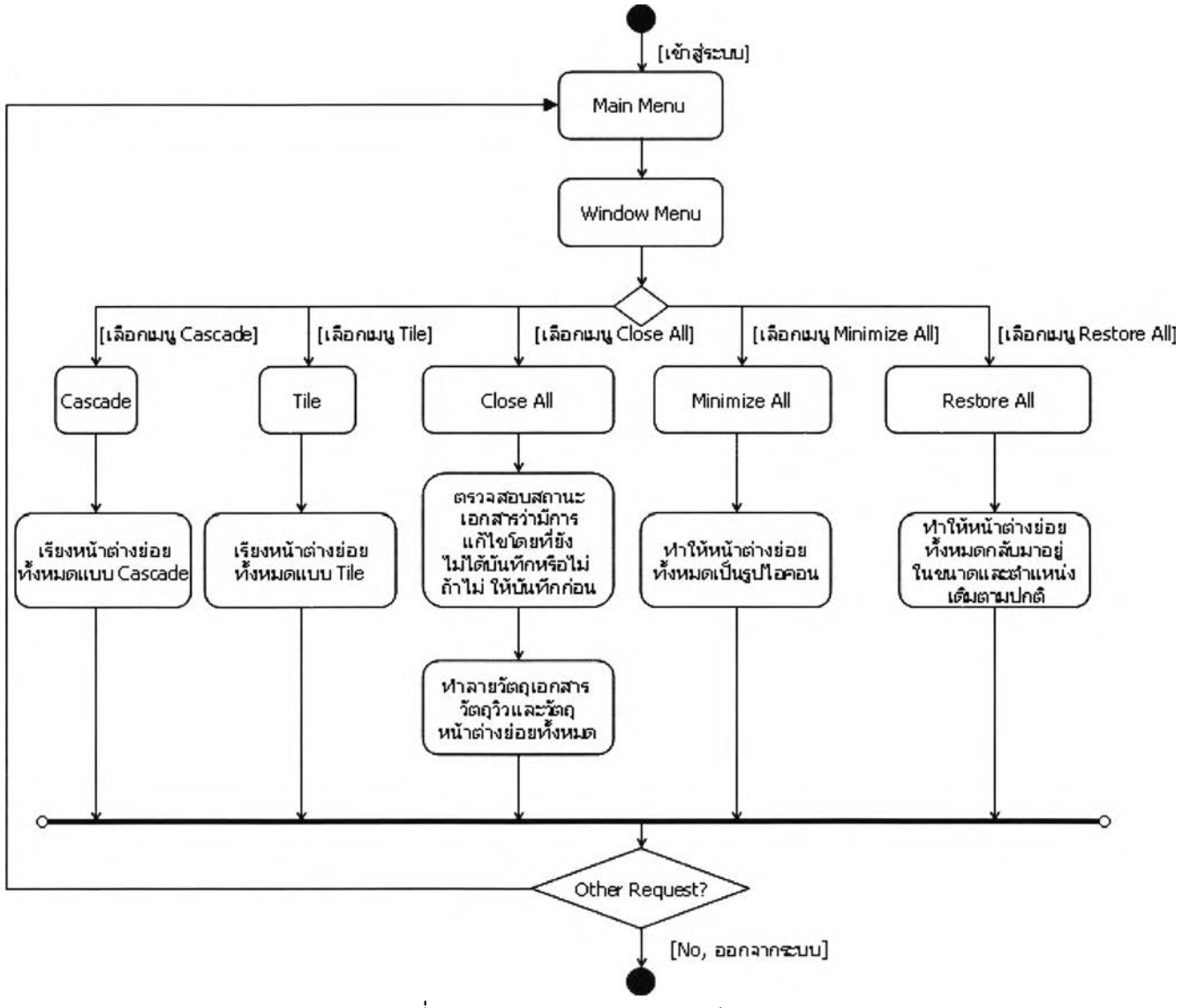
รูปที่ 3.22 แสดงแผนภาพกิจกรรมของเมนู Help ซึ่งประกอบไปด้วย 2 เมนูย่อย มีรายละเอียดดังนี้

3.6.7.1 เมนู Feedback

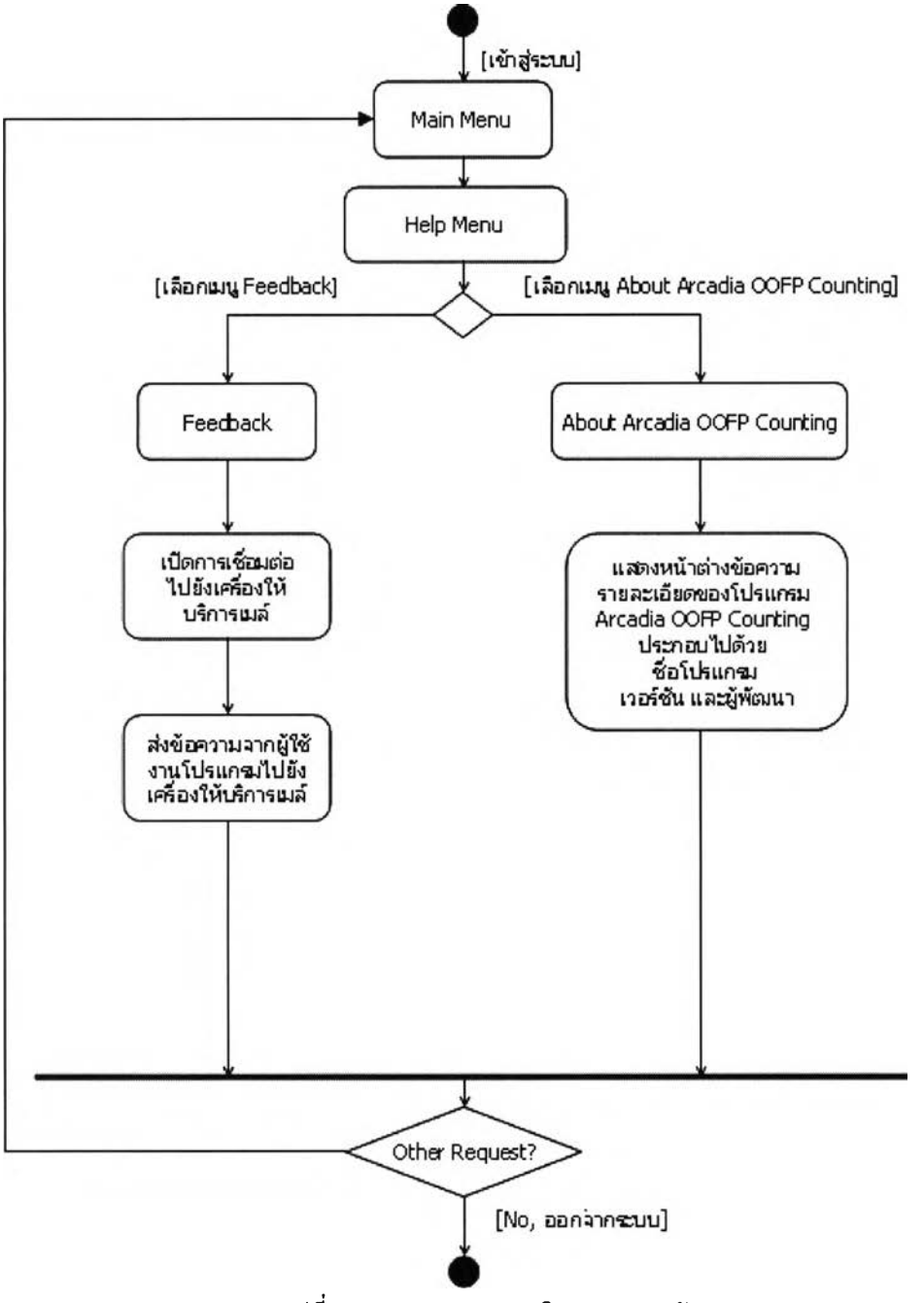
เมื่อเมนู Feedback ถูกเรียกใช้ วัตถุหน้าต่างย่อยสำหรับกรอกข้อความตอบรับ (JFeedbackWindow) จะถูกสร้างเพื่อให้ผู้ใช้กรอกข้อความที่ต้องการ และวัตถุ JSocketMonitor จะถูกสร้างเพื่อเปิดการเชื่อมต่อไปยังเครื่องให้บริการเมล์ จากนั้นจะส่งข้อความจากผู้ใช้ไปยังเครื่องให้บริการเมล์

3.6.7.2 เมนู About Chula OOFP Counting

เมื่อเมนู About Chula OOFP Counting ถูกเรียกใช้ วัตถุหน้าต่างย่อยแสดงข้อความรายละเอียดของโปรแกรม (JAboutDialog) จะถูกสร้างซึ่งข้อความรายละเอียดประกอบไปด้วย ชื่อโปรแกรม เวอร์ชัน และผู้พัฒนา



รูปที่ 3.21 แผนภาพแสดงกิจกรรมสำหรับเมนู Window



รูปที่ 3.22 แผนภาพแสดงกิจกรรมสำหรับเมนู Help