บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันการออกแบบและพัฒนาโปรแกรมเชิงวัตถุได้รับการยอมรับมากขึ้น เนื่องจากข้อดีหลายประการจากแนวคิดเชิงวัตถุ ได้แก่ ความสามารถในการนำซอฟต์แวร์โมดูลกลับมาใช้ใหม่ (Reusability) ความสามารถในการสืบทอดคุณสมบัติของซอฟต์แวร์โมดูล (Inheritance) ความสามารถในการนำข้อมูล (Attribute) และพฤติกรรม (Behavior/Method) ที่จะกระทำต่อข้อมูลนั้นมาผูกกันกลายเป็นโครงสร้างข้อมูลเฉพาะ (Encapsulation) และความสามารถในการกำหนดความหลากหลายของรูปแบบพฤติกรรมของซอฟต์แวร์โมดูล (Polymorphism) ซึ่งข้อดีเหล่านี้เป็นสิ่งที่ช่วยผลักดันทำให้การออกแบบและพัฒนาโปรแกรมเชิงวัตถุได้รับความนิยมอย่างสูง

ในการพัฒนาซอฟต์แวร์เชิงวัตถุ แผนการดำเนินงานเป็นส่วนสำคัญในความสำเร็จของโครงงาน การวางแผนจัดสรรทรัพยากรที่มีอยู่จำกัดอย่างเหมาะสม มีส่วนสนับสนุนทำให้การดำเนินงานเป็นไปตามเป้าหมายที่วางไว้ โดยเฉพาะอย่างยิ่งถ้าผู้วางแผนทราบขนาดของซอฟต์แวร์รวมถึงขนาดของส่วนประกอบย่อยต่างๆตั้งแต่ขั้นตอนการวิเคราะห์และออกแบบ จะยิ่งทำให้การจัดสรรทรัพยากรเป็นไปด้วยความถูกต้องมากขึ้น ดังนั้นผู้วิจัยได้ตระหนักถึงการวัด (Measurement) ขนาดของซอฟต์แวร์ว่าเป็นส่วนสำคัญในความสำเร็จของโครงงาน

วิธีวัดซอฟต์แวร์ที่นิยมในปัจจุบันได้แก่การนับจำนวนบรรทัดของโค้ด (Line of code: LOC) ซึ่งเป็นวิธีการวัดที่ทำได้ง่ายที่สุด การนับจำนวนบรรทัดของโค้ดเป็นวิธีการวัดที่มีมาตั้งแต่การเขียนโปรแกรมแบบโครงสร้าง (Structured Programming) และยังคงเป็นที่นิยมอยู่ในปัจจุบัน ข้อด้อยของวิธีวัดลักษณะนี้คือภาษาคอมพิวเตอร์ที่ใช้ในการพัฒนาจะมีอิทธิพลอย่างสูงต่อการวัด ถึงแม้จะเป็นโปรแกรมที่ทำงานเหมือนกันแต่ถ้าใช้ภาษาคอมพิวเตอร์แตกต่างกันขนาดที่วัดได้ย่อมต่างกัน

ฟังก์ชันพอยต์เชิงวัตถุเป็นวิธีการวัดซอฟต์แวร์เชิงวัตถุที่ดัดแปลงมาจากฟังก์ชันพอยต์แบบดั้งเดิมที่ทำงานบนแนวคิดการออกแบบและพัฒนาซอฟต์แวร์แบบโครงสร้าง ข้อดีของฟังก์ชันพอยต์คือเป็นเทคนิคการวัดที่ไม่ขึ้นอยู่กับภาษาคอมพิวเตอร์ที่ใช้ในการพัฒนา ฟังก์ชันพอยต์ใช้ข้อกำหนดความต้องการของซอฟต์แวร์ (Software Requirement Specification) เป็นข้อมูลเข้าในการวิเคราะห์คำนวณขนาดของซอฟต์แวร์ ในปัจจุบันได้มีผู้ทำการวิจัยเกี่ยวกับฟังก์ชันพอยต์เชิงวัตถุไว้บางส่วน อาทิ งานวิจัยของ T. Fetcke, A. Abran และ T-H Nguyen [3] ได้พยายามสร้างหลักเกณฑ์การนับจำนวนฟังก์ชันพอยต์ โดยนำโมเดลการใช้งาน (Use-Case Model) ในภาษายูเอ็มแอลไปประยุกต์เข้ากับแนวคิดของฟังก์ชันพอยต์แบบดั้งเดิม และงานวิจัยของ G.Caldiera , G.Antoniol และ C. Logan [4] ได้นำเสนอวิธีการนับจำนวนของฟังก์ชันพอยต์เชิงวัตถุ โดยใช้แผนภาพคลาสเป็นเครื่องมือในการคำนวณ แต่ยังไม่มีการพัฒนาเครื่องมือวัดสำหรับการนับจำนวนฟังก์ชันพอยต์เชิงวัตถุที่ประยุกต์

แนวคิดและวิธีการนับที่สามารถใช้งานได้จริง และในงานวิจัยที่ได้จัดทำขึ้นนี้เป็นการพัฒนาต่อจากงานวิจัยของ G.Caldiera , G.Antoniol และ C. Logan เพื่อทำการออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์ซึ่งใช้เทคนิคการวัดแบบฟังก์ชันพอยต์เชิงวัตถุ การทำงานของเครื่องมือวัดเป็นแบบอัตโนมัติ โดยใช้แผนภาพคลาสซึ่งเป็นข้อกำหนดความต้องการของซอฟต์แวร์เป็นข้อมูลเข้า จำนวนฟังก์ชันพอยต์ของคลาสและของซอฟต์แวร์เป็นค่าวัดที่ได้จากเครื่องมือนี้ ข้อมูลเหล่านี้ทำให้ผู้พัฒนาสามารถทราบขนาดของซอฟต์แวร์เชิงวัตถุได้ในระหว่างขั้นตอนการวิเคราะห์และออกแบบระบบ ซึ่งช่วยให้นักพัฒนาหรือผู้บริหารโครงงานสามารถจัดสรรทรัพยากรที่มีอยู่อย่างเหมาะสมกับขนาดซอฟต์แวร์ที่วัดได้

1.2 วัตถุประสงค์

วิทยานิพนธ์นี้มีวัตถุประสงค์เพื่อออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์โดยใช้เทคนิคการวัดแบบฟังก์ชันพอยต์เชิงวัตถุ

1.3 ขอบเขตของงานวิจัย

วิทยานิพนธ์มีขอบเขตของงานวิจัยดังนี้

1.3.1 พัฒนาเครื่องมือมาตรวัดซอฟต์แวร์เชิงวัตถุแบบฟังก์ชันพอยต์ สำหรับภาษาโปรแกรมเชิงวัตถุ

1.3.2 โปรแกรมที่นำมาพิจารณาหาค่าวัดต้องเป็นโปรแกรมที่พัฒนาด้วยภาษาจาวา

1.3.3 โปรแกรมที่นำมาพิจารณาหาค่าวัดต้องมีไฟล์ข้อมูลของภาษา Abstract Object Language (AOL) ซึ่งแสดงโมเดลเชิงวัตถุแบบข้อความ (Textual) ประกอบการพิจารณา

1.3.4 จำนวนโปรแกรมที่ใช้ในการทดสอบกับเครื่องมือวัดมีอย่างน้อย 3 โปรแกรม

1.3.5 พัฒนาและใช้เครื่องมือบนระบบปฏิบัติการวินโดว์ตั้งแต่รุ่น 95 ขึ้นไป

1.4 ขั้นตอนและวิธีการดำเนินงาน

วิทยานิพนธ์นี้มีขั้นตอนและวิธีการดำเนินงานดังนี้

1.4.1 ศึกษาแนวคิดและทฤษฎีที่เกี่ยวกับการวัดขนาดซอฟต์แวร์เชิงวัตถุ

1.4.2 ศึกษาแนวคิดและทฤษฎีที่เกี่ยวกับการวัดขนาดซอฟต์แวร์แบบฟังก์ชันพอยต์

1.4.3 ศึกษาแนวคิดและทฤษฎีที่เกี่ยวกับการวัดขนาดซอฟต์แวร์แบบฟังก์ชันพอยต์เชิงวัตถุ

1.4.4 ศึกษาหลักไวยากรณ์ต่างๆของโปรแกรมภาษาจาวา เพื่อใช้ในการพัฒนาเครื่องมือวัด

1.4.5 ออกแบบและพัฒนาเครื่องมือวัดซอฟต์แวร์

1.4.6 ทดสอบเครื่องมือวัดซอฟต์แวร์

1.4.7 ประเมินผลเครื่องมือวัดที่พัฒนากับเครื่องมือวัดซอฟต์แวร์ที่ใช้ในงานวิจัยที่มีอยู่ขณะนี้

1.4.8 สรุปผลและเสนอแนะผลของการวิจัย

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ที่คาดว่าจะได้รับจากวิทยานิพนธ์และเครื่องมือวัดซอฟต์แวร์มีดังนี้

1.5.1 สามารถใช้เป็นเครื่องมือในการวัดขนาดของซอฟต์แวร์เชิงวัตถุ

เมื่อผู้ใช้สร้างแผนภาพคลาสด้วยเครื่องมือช่วยในการออกแบบซอฟต์แวร์เชิงวัตถุและใช้งานร่วมกับเครื่องมือวัดซอฟต์แวร์ที่พัฒนาขึ้นในงานวิจัย เพื่อคำนวณหาจำนวนของฟังก์ชันพอยต์เชิงวัตถุ ผู้ใช้สามารถทราบขนาดของซอฟต์แวร์ที่กำลังจะพัฒนาได้ตั้งแต่ขั้นตอนการวิเคราะห์และออกแบบระบบ ทำให้การวางแผนจัดสรรทรัพยากรที่มีอยู่จำกัดเป็นไปอย่างเหมาะสม

1.5.2 สามารถใช้เป็นเครื่องมือในการติดตามความก้าวหน้าของการพัฒนาซอฟต์แวร์

เครื่องมือวัดสามารถคำนวณหาจำนวนของฟังก์ชันพอยต์เชิงวัตถุจากโค้ดต้นฉบับ ดังนั้นในระหว่างการพัฒนาซอฟต์แวร์ผู้บริหารโครงงานหรือนักพัฒนาสามารถทราบจำนวนฟังก์ชันพอยต์ที่พัฒนาไปแล้วได้ โดยการนำโค้ดต้นฉบับที่ได้พัฒนาเสร็จแล้วเป็นข้อมูลเข้าสำหรับเครื่องมือวัด

1.5.3 สามารถใช้เป็นเครื่องมือในการเลือกปรับปรุงคลาสที่มีขนาดใหญ่

ถ้าผู้พัฒนาซอฟต์แวร์ได้มีการกำหนดขนาดของคลาส เช่นจำนวนฟังก์ชันพอยต์เชิงวัตถุของคลาสหนึ่งๆ จะต้องมีค่าไม่เกิน 40 ฟังก์ชันพอยต์ เป็นต้น หลังจากใช้เครื่องมือวัดเพื่อดูจำนวนของฟังก์ชันพอยต์แล้วพบว่ามีขนาดเกินกว่าที่ได้กำหนดไว้ ผู้พัฒนาก็สามารถพิจารณาปรับปรุงให้มีขนาดให้เล็กลงได้ซึ่งทำให้ลดความซับซ้อนลง เช่นแยกคลาสหรือโมดูลออกเป็นโมดูลย่อย หรือลดไฟล์ต้นฉบับที่ไม่มีความจำเป็นออกจากโปรแกรม เป็นต้น