วิทยานิพนธ์นี้นำเสนอวิธีการตรวจจับข้อบกพร่องของโมเดลการออกแบบ 6 ประเภท ได้แก่ Data Class, Feature Envy, Message Chains, Middle Man, God Class, และ Switch Statements โดยพิจารณาว่ามีข้อบกพร่องประเภทใดบ้างที่ส่งผลกระทบต่อความสามารถในการเปลี่ยนแปลงซอฟต์แวร์เชิงวัตถุ ในงานวิจัยมีการออกแบบกลยุทธ์การตรวจจับข้อบกพร่องด้วยมาตรวัดเชิงวัตถุสำหรับโมเดลการออกแบบที่แผนภาพคลาสและแผนภาพซีเควนซ์ นำเสนอวิธีการรีแฟคทอริงเพื่อแก้ไขข้อบกพร่องแต่ละประเภท พร้อมทั้งนำเสนอวิธีการหาช่วงของมาตรวัดที่บอกถึงข้อบกพร่องโดยประยุกต์ใช้อัลกอริทึมซีมูเลทเต็ดแอนนิลลิงเพื่อหาค่าช่วงที่เหมาะสมที่สุดโดยใช้กลุ่มตัวอย่าง 35 ตัวอย่างสำหรับข้อบกพร่องแต่ละประเภท รวมถึงได้ออกแบบและพัฒนาเครื่องมือสำหรับตรวจจับข้อบกพร่องของโมเดลการออกแบบและเครื่องมือที่ใช้ในการหาค่าช่วงที่เหมาะสม

ในงานวิจัยได้ทดสอบกลยุทธ์การตรวจจับข้อบกพร่องและประเมินผลกระทบของการเกิดข้อบกพร่องต่าง ๆ ต่อความสามารถในการเปลี่ยนแปลงซอฟต์แวร์ด้วยกลุ่มตัวอย่างสำหรับทดสอบ 5 ตัวอย่างต่อข้อบกพร่องของโมเดลการออกแบบหนึ่งประเภท โดยเปรียบเทียบค่ามาตรวัดความสามารถในการเปลี่ยนแปลงก่อนและหลังการตรวจจับข้อบกพร่องของโมเดลการออกแบบ และประยุกต์ใช้วิธีรีแฟคทอริงสำหรับแก้ไขข้อบกพร่องของโมเดลการออกแบบแต่ละประเภท ผลของการทดสอบระบุว่าการแก้ไขระบบที่มีข้อบกพร่องของโมเดลการออกแบบประเภท Data Class, Middle Man, และ God Class สามารถทำให้ความสามารถในการเปลี่ยนแปลงซอฟต์แวร์ดีขึ้น การแก้ไขระบบที่มีข้อบกพร่องประเภท Feature Envy และ Message Chains ไม่มีผลทำให้ความสามารถในการเปลี่ยนแปลงซอฟต์แวร์ดีขึ้น ส่วนการแก้ไขระบบที่มีข้อบกพร่องประเภท Switch Statements มีผลทำให้ความสามารถในการเปลี่ยนแปลงซอฟต์แวร์ลดลง

This thesis proposes detection strategies for six design defects including Data Class Feature Envy, Message Chains, Middle Man, God Class, and Switch Statement to verify whether the particular design defects affect object-oriented software modifiability. The strategies use object-oriented software design metrics for determining a fraction of class and sequence diagram which is affected by particular design defects. The approach also suggests refactoring techniques for modifying the class and the sequence diagram. In addition, an approach for finding the optimized threshold values for detecting particular design defects is also provided by applying simulated annealing algorithm to 35 design models for each design defect. An automated tool for design defects detection and for calculating the optimized threshold values is also implemented.

The thesis approach is evaluated by comparing modifiability metrics before and after applying the refactoring techniques to 5 design models for each design defect. The result shows that modifiability of software is enhanced after applying the refactoring to Data Class, Middle Man, and God Class design defects and not changed for Feature Envy and Middle Man design defects. For Switch Statements design defect, the modifiability metric values are dropped.