การออกแบบและพัฒนาการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบ
อัตโนมัติโดยใช้โครงสร้างยูไอ

นายณัฐรัตน์ หาญวรวงศ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2556

DESIGN AND IMPLEMENTATION OF TEST CASE FOR AUTOMATED TESTING USING UI STRUCTURE

Mr. Nutharat Harnvorawong

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Software Engineering
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2013

หัวข้อวิทยานิพนธ์	การออกแบบและพัฒนาการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติโดยใช้โครงสร้างยูไอ
โดย	นายณัฐรัตน์ หาญวรวงศ์
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร. ธาราทิพย์ สุวรรณศาสตร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร. บัณฑิต เอื้ออาภรณ์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(รองศาสตราจารย์ ดร. วิวัฒน์ วัฒนาวุฒิ)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(รองศาสตราจารย์ ดร. ธาราทิพย์ สุวรรณศาสตร์)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร. อาทิตย์ ทองทักษ์)

.....กรรมการภายนอกมหาวิทยาลัย

(ผู้ช่วยศาสตราจารย์ ดร. ภัทรชัย ลลิตโรจน์วงศ์)

ณัฐรัตน์ หาญวรวงศ์ : การออกแบบและพัฒนาการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติโดยใช้โครงสร้างยูไอ. (DESIGN AND IMPLEMENTATION OF TEST CASE FOR AUTOMATED TESTING USING UI STRUCTURE) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร. ธาราทิพย์ สุวรรณศาสตร์, 86 หน้า.

นักทดสอบซอฟต์แวร์จะทำการทดสอบซอฟต์แวร์เวอร์ชันใหม่โดยใช้กรณีทดสอบที่มีอยู่แล้ว เพื่อทดสอบให้มั่นใจว่าการเปลี่ยนแปลงของซอฟต์แวร์ไม่ได้ส่งผลให้การทำงานต่างๆ ที่เคยมีอยู่มีความผิดพลาดหรือไม่เป็นไปอย่างที่เคยเป็นเมื่อซอฟต์แวร์มีการปรับปรุงหรือแก้ไข เรียกว่าการทดสอบเชิงถดถอย วิธีการหนึ่งที่สามารถช่วยลดค่าใช้จ่ายในกระบวนการทดสอบในระยะยาว คือ การทดสอบแบบอัตโนมัติ อย่างไรก็ตามการทดสอบเชิงถดถอยและการทดสอบแบบอัตโนมัติยังคงเป็นกระบวนการที่มีค่าใช้จ่ายสูงในช่วงพัฒนา ดังนั้นงานวิจัยชิ้นนี้จึงมีแนวความคิดสร้างกรอบการทำงานที่ช่วยให้การพัฒนากรณีทดสอบสำหรับการทดสอบแบบอัตโนมัติมีความสะดวกรวดเร็ว และสามารถวิเคราะห์กรณีทดสอบเพื่อหาส่วนของซอฟต์แวร์ที่ควรได้รับการทดสอบเพิ่มเติม เพื่อให้นักทดสอบสร้างกรณีทดสอบได้อย่างมีประสิทธิภาพ งานวิจัยนี้ประกอบไปด้วยเครื่องมือต่างๆ สำหรับช่วยให้นักทดสอบสามารถสร้างกรณีทดสอบสำหรับการทดสอบแบบอัตโนมัติได้อย่างง่ายดายและมีประสิทธิภาพ เครื่องมือสามารถสร้างโครงสร้างยูไอของหน้าจอของซอฟต์แวร์ที่ต้องการทดสอบ โครงสร้างยูไอคือข้อมูลยูสเซอร์คอนโทรลต่างๆ ที่ถูกจัดเก็บในลักษณะโครงสร้างในรูปแบบภาษาการเขียนโปรแกรม เครื่องมือสามารถสร้างกรณีทดสอบได้โดยอัตโนมัติจากข้อมูลของโครงสร้างยูไอ นอกจากนี้ยังมีเครื่องมือสำหรับช่วยแก้ไขกรณีทดสอบและวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ ซึ่งเครื่องมือดังกล่าวสามารถระบุยูสเซอร์คอนโทรลที่ไม่ถูกเรียกใช้งานได้ ช่วยให้นักทดสอบตระหนักถึงจุดที่ขาดการทดสอบภายในซอฟต์แวร์และเพิ่มการทดสอบได้อย่างเหมาะสม

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2556

5470937321 : MAJOR SOFTWARE ENGINEERING
KEYWORDS: AUTOMATED TESTING / GUI / AUTOMATION / SOFTWARE TESTING / NEO AUTOMATION FRAMEWORK

NUTHARAT HARNVORAWONG: DESIGN AND IMPLEMENTATION OF TEST CASE FOR AUTOMATED TESTING USING UI STRUCTURE. ADVISOR: ASSOC. PROF. TARATIP SUWANNASART, 86 pp.

Testers usually run a new version of software against existing test cases to validate that changes do not cause any unexpected results in legacy functionalities when the software is modified or enhanced. A solution that can reduce cost in long term is automated testing. However regression testing and automated testing are still resources consuming and high cost during development time. In this thesis we propose a framework to facilitate automated testing and it also can analyze test cases to reveal parts in software where lack of testing. This thesis provides tools that allow testers to create and develop automated test cases easily and efficiently. The tool can generate UI structure of a given form inside software under test. The UI structure is a list of usable UI controls in hierarchical data structure in a class format of programming language. Automated test cases can be automatically generated from the UI structure. There is a tool for simply modifying test case and analyzing usage of UI controls in the test cases. The analyzer tool can identify UI controls which are not used and so testers can be aware of area lacking of test in their test cases.

Department: Computer Engineering Student's Signature .....
Field of Study: Software Engineering Advisor's Signature .....
Academic Year: 2013

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สำเร็จลงได้ด้วยความช่วยเหลืออย่างดียิ่งจากรองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่เสียสละเวลาให้คำปรึกษาและแนะนำแนวทางในการทำวิจัย

ขอขอบพระคุณรองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ ประธานกรรมการสอบวิทยานิพนธ์ ผู้ช่วยศาสตราจารย์ ดร.อาทิตย์ ทองทักษ์ และผู้ช่วยศาสตราจารย์ ดร.ภัทรชัย ลลิตโรจน์วงศ์ กรรมการสอบวิทยานิพนธ์ ที่ได้กรุณาสละเวลาให้คำแนะนำและขัดเกลาวิทยานิพนธ์ฉบับนี้ให้มีคุณภาพ

ขอขอบพระคุณอาจารย์ทุกท่าน ที่ประสิทธิ์ประสาทความรู้ให้แก่ผู้วิจัย เพื่อนำมาใช้ในงานวิจัยและทำวิทยานิพนธ์ฉบับนี้

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ทุกคน สำหรับคำปรึกษา คำแนะนำ และกำลังใจต่างๆ ในการทำวิจัย ขอขอบคุณ คุณนิภาพร ธนประโยชน์ศักดิ์ ที่เป็นกำลังใจและช่วยสนับสนุนการทำงานวิจัย

สุดท้ายนี้ขอขอบพระคุณบิดา มารดา และน้องสาว สมาชิกในครอบครัว ที่ให้การสนับสนุนในทุกด้าน และคอยเป็นกำลังใจตลอดมา

สารบัญ

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญรูป

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การทดสอบซอฟต์แวร์ (Software Testing) เป็นหนึ่งในขั้นตอนหลักของกระบวนการพัฒนาซอฟต์แวร์ มีวัตถุประสงค์เพื่อตรวจสอบความถูกต้อง ให้เป็นไปตามข้อกำหนดหรือสิ่งที่คาดหวัง [1] การทดสอบที่ดีจะต้องตรวจพบความไม่ถูกต้องหรือไม่เหมาะสมในซอฟต์แวร์ได้ นักทดสอบจะต้องมีความเข้าใจในซอฟต์แวร์ที่จะทำการทดสอบเป็นอย่างดี มีการออกแบบกรณีทดสอบ (Test Case) อย่างเหมาะสม นอกจากนี้การทดสอบซอฟต์แวร์ภายหลังจากการปรับปรุงการทำงานเพิ่มเติมหรือแก้ไขข้อบกพร่อง ก็มีความจำเป็นเช่นเดียวกัน เพื่อให้มั่นใจว่าการทำงานในส่วนอื่นของซอฟต์แวร์ยังคงทำงานได้อย่างถูกต้อง ไม่มีผลกระทบจากการเปลี่ยนแปลงที่เกิดขึ้น การทดสอบเช่นนี้เรียกว่า การทดสอบเชิงถดถอย (Regression Testing) [2] อย่างไรก็ตามขั้นตอนการทดสอบซอฟต์แวร์นั้นมีค่าใช้จ่ายค่อนข้างสูงเนื่องจากต้องใช้ทรัพยากรต่างๆ เช่น นักทดสอบ เวลา เครื่องมือสำหรับการทดสอบ และสภาพแวดล้อมของระบบที่จะทำการทดสอบ เป็นต้น ดังนั้นยิ่งซอฟต์แวร์มีขนาดใหญ่เพียงใด ย่อมมีค่าใช้จ่ายในการทดสอบที่สูงตามไปด้วย

วิธีหนึ่งในการลดค่าใช้จ่ายและลดระยะเวลาในการทดสอบคือ การทดสอบแบบอัตโนมัติ (Automated Testing) ซึ่งเป็นการทดสอบโดยใช้เครื่องมือหรือซอฟต์แวร์อื่นทำการทดสอบซอฟต์แวร์โดยทำงานตามชุดคำสั่ง (Script) ที่ได้กำหนดไว้ การทดสอบด้วยวิธีนี้มีข้อดีคือ การทดสอบทำงานได้รวดเร็ว เที่ยงตรงและแม่นยำ อย่างไรก็ตามข้อเสียของการทดสอบแบบอัตโนมัติคือ จะต้องมีการเตรียมชุดคำสั่งเอาไว้ล่วงหน้า ซอฟต์แวร์บางประเภทอาจไม่เอื้ออำนวยต่อการทดสอบแบบอัตโนมัติ และการทดสอบบางประเภทยังจำเป็นต้องใช้การตัดสินใจจากนักทดสอบ เมื่อซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลง ชุดคำสั่งจะต้องถูกแก้ไขให้เหมาะสมตามไปด้วย เป็นต้น

เครื่องมือสำหรับการทดสอบแบบอัตโนมัติประเภทหนึ่งที่สามารถใช้งานได้ง่ายและรวดเร็วเรียกว่าการบันทึกและเล่นซ้ำ (Capture & Playback) [3] ขั้นตอนการใช้งานมีลักษณะเหมือนการบันทึกวีดีโอและนำมาเล่นซ้ำ นักทดสอบจะต้องสั่งให้เครื่องมือเริ่มบันทึกการทำงานก่อนแล้วจึงทำการทดสอบ เมื่อการทดสอบเสร็จสิ้นนักทดสอบจะสั่งให้เครื่องมือหยุดการบันทึก หลังจากนั้นเครื่องมือจะสร้างกรณีทดสอบขึ้นมาให้โดยอัตโนมัติจากกิจกรรมต่างๆ ของนักทดสอบ เช่น เลื่อนเม้าส์ พิมพ์ข้อความ กดปุ่ม เป็นต้น อย่างไรก็ตามรายละเอียดต่างๆ ของการทดสอบ เช่น ตำแหน่งพิกัดที่เม้าส์กดปุ่ม ข้อความที่ถูกพิมพ์ลงไปบนกล่องข้อความ ตำแหน่งของปุ่มที่ถูกกด ล้วนถูกบันทึกลงไปในกรณีทดสอบโดยตรง ปัญหาที่พบคือเมื่อนักทดสอบต้องการทดสอบกรณีทดสอบอื่นที่มีการทำงานคล้ายกันจะไม่สามารถนำกรณีทดสอบที่มีอยู่แล้วมาใช้งานได้ นักทดสอบจะต้องทำการบันทึกการกระทำต่างๆ ใหม่ทั้งหมด [4] และปัญหาอีกอย่างหนึ่งคือ เมื่อซอฟต์แวร์มีการเปลี่ยนแปลง [5] เช่น ตำแหน่งของยูสเซอร์คอนโทรล (User Control) ถูกย้ายจากตำแหน่งหนึ่งไปยังอีกตำแหน่งหนึ่งบนหน้าจอ กรณี

ทดสอบจะไม่สามารถทำงานได้อย่างถูกต้อง นักทดสอบจำเป็นต้องบันทึกการกระทำต่างๆ ใหม่อีกครั้งหนึ่ง

งานวิจัยนี้จะนำเสนอการออกแบบและพัฒนาการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติ โดยมีกรอบการทำงาน (Framework) ที่มีแนวคิดหลักคือวิธีการจัดเก็บรายชื่อยูสเซอร์คอนโทรลในลักษณะโครงสร้างในรูปแบบคลาส เรียกว่า โครงสร้างยูไอ (UI Structure) โดยจัดเก็บข้อมูลที่ใช้สำหรับอ้างอิงถึงยูสเซอร์คอนโทรลต่างๆ รวมถึงข้อมูลขนาดและตำแหน่งของยูสเซอร์คอนโทรลเอาไว้ ทำให้สามารถสร้างกรณีทดสอบแบบอัตโนมัติได้โดยอัตโนมัติและมีความใกล้เคียงกับการทำงานของผู้ใช้งานมากที่สุด การสั่งงานยูสเซอร์คอนโทรลต่างๆ ในระหว่างการทดสอบมีความถูกต้องและเชื่อถือได้ ประโยชน์อีกประการหนึ่งจากการจัดเก็บรายชื่อยูสเซอร์คอนโทรลในรูปแบบดังกล่าวคือ ลดความซ้ำซ้อนของข้อมูลเกี่ยวกับยูสเซอร์คอนโทรลในกรณีทดสอบต่างๆ โดยจัดเก็บอยู่ในแหล่งเดียว ทำให้สะดวกต่อการปรับปรุงแก้ไขในกรณีที่ซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลง อีกทั้งยังสามารถวิเคราะห์หาความครอบคลุมของการทดสอบจากการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบได้อีกด้วย

1.2 วัตถุประสงค์ของงานวิจัย

เพื่อออกแบบและพัฒนาการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติ โดยใช้รายละเอียดยูสเซอร์คอนโทรลของซอฟต์แวร์ที่อยู่ในลักษณะโครงสร้างในรูปแบบคลาส และตรวจวัดความครอบคลุมของการทดสอบจากกรณีทดสอบ

1.3 ขอบเขตของงานวิจัย

รายละเอียดขอบเขตของงานวิจัยมีดังนี้

1.3.1 ข้อมูลนำเข้า

1) ซอฟต์แวร์ที่ต้องการทดสอบที่ถูกพัฒนาด้วยกลุ่มภาษาโปรแกรมดอทเน็ต (.NET) เช่น ซีชาร์ป (C#) เป็นต้น และเป็นเอ็กเซ็กคิวเทเบิลไฟล์ประเภทวินโดวส์แอพพลิเคชันที่ทำงานบนระบบปฏิบัติการวินโดวส์
2) ยูสเซอร์คอนโทรลที่ใช้ในซอฟต์แวร์ที่ต้องการทดสอบเป็นยูสเซอร์คอนโทรลมาตรฐานประเภท WinControl เท่านั้น ไม่มียูสเซอร์คอนโทรลประเภทอื่นหรือยูสเซอร์คอนโทรลที่โปรแกรมเมอร์ (Programmer) สร้างขึ้นมาเอง (Custom Control) และไม่รองรับกรณีที่ยูสเซอร์คอนโทรลถูกสร้างแบบพลวัต (Dynamic) ณ ขณะรันไทม์

1.3.2 การทำงานของแต่ละส่วนภายในระบบ

1.3.2.1 ส่วนสร้างโครงสร้างยูไอ

1) ส่วนสร้างโครงสร้างยูไอสามารถสร้างโครงสร้างยูไอและจัดเก็บอยู่ในรูปแบบคลาสในภาษาซีชาร์ป โดยแยกเก็บแบ่งตามหน้าจอของซอฟต์แวร์ที่ต้องการทดสอบ หน้าจอละ 1 คลาส

2) ในกรณีที่ซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลง เช่น มียูสเซอร์คอนโทรลเพิ่มขึ้นหรือหายไปจากซอฟต์แวร์ เป็นต้น โครงสร้างยูไอสามารถรับการแก้ไขได้โดยอัตโนมัติโดยใช้ส่วนสร้างโครงสร้างยูไอ

1.3.2.2 ส่วนสร้างกรณีทดสอบ

1) ส่วนสร้างกรณีทดสอบจะสร้างกรณีทดสอบโดยเรียงลำดับขั้นตอนการทำงานจากยูสเซอร์คอนโทรลที่อยู่ด้านบนก่อน เรียงจากซ้ายไปขวา และเรียงลงไปตามลำดับ เท่านั้น

2) กรณีทดสอบที่ถูกสร้างขึ้นโดยอัตโนมัติ ถ้าหากการทำงานกับยูสเซอร์คอนโทรลในขั้นตอนใดต้องการข้อมูลนำเข้าค่าเริ่มต้นที่ระบุจะเป็นค่าว่าง

1.3.2.3 ส่วนแก้ไขกรณีทดสอบ

1) สามารถแก้ไขกรณีทดสอบที่ถูกสร้างจากส่วนสร้างกรณีทดสอบ หรือถูกแก้ไขจากส่วนแก้ไขกรณีทดสอบเท่านั้น ไม่สามารถแก้ไขกรณีทดสอบที่ผ่านการแก้ไขโดยตรงด้วยการใช้เครื่องมือแก้ไขข้อความ (Text Editor) อื่นๆ

2) ในกรณีที่ซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลง กรณีทดสอบที่มีอยู่แล้วจะยังคงเหมือนเดิม ซึ่งอาจใช้งานไม่ได้เนื่องจากการเปลี่ยนแปลงที่เกิดขึ้น นักทดสอบจะต้องทำการแก้ไขด้วยตนเอง อย่างไรก็ตามนักทดสอบสามารถสร้างกรณีทดสอบใหม่ให้สอดคล้องกับโครงสร้างยูไอที่เปลี่ยนแปลงไปได้โดยใช้ส่วนสร้างกรณีทดสอบ

3) ในกรณีที่ซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลง กรณีทดสอบที่มีอยู่แล้วอาจจะยังสามารถใช้งานได้ หรืออาจไม่สามารถใช้งานได้เนื่องจากการเปลี่ยนแปลงแปลงที่เกิดขึ้น งานวิจัยนี้ไม่ครอบคลุมถึงการวิเคราะห์ดังกล่าว

1.3.2.4 ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบสามารถวิเคราะห์โดยเปรียบเทียบโครงสร้างยูไอและการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ เพื่อหาค่าความครอบคลุมและนำไปแสดงในรายงานการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

1.3.3 การทดสอบ

1) ประเภทของยูสเซอร์คอนโทรลและการทำงานของยูสเซอร์คอนโทรล (Control Pattern) ต่างๆ ภายในกรณีทดสอบ จะสามารถทำได้ตามที่การทำงานของ UIA รองรับเท่านั้น [6]

2) ส่วนจัดการการทดสอบ (Test Manager) และผลการทดสอบ (Test Report) ไม่อยู่ในขอบเขตการพัฒนาของงานวิจัยนี้ ผู้วิจัยจะใช้เครื่องมือของ Microsoft ที่มีอยู่แล้วในชุดซอฟต์แวร์ Microsoft Visual Studio เวอร์ชัน 2010 เพื่อการวิจัย

3) ทำการทดสอบการทำงานของวิธีการที่นำเสนอ ด้วยซอฟต์แวร์ที่ต้องการทดสอบอย่างน้อย 2 แอพพลิเคชัน

1.4 ขั้นตอนและวิธีการดำเนินงานวิจัย

1) ศึกษาข้อมูลและทำความเข้าใจวิธีการทดสอบแบบอัตโนมัติ และเทคโนโลยีการเข้าถึงวัตถุ
2) วิเคราะห์และออกแบบวิธีการจัดเก็บรายชื่อยูสเซอร์คอนโทรล
3) ออกแบบและพัฒนาเครื่องมือสำหรับสร้างโครงสร้างยูไอ
4) ออกแบบและพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบ
5) ออกแบบและพัฒนาเครื่องมือสำหรับแก้ไขกรณีทดสอบ
6) ออกแบบและพัฒนาเครื่องมือสำหรับวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
7) ทดสอบและประเมินผลวิธีวิจัย
8) สรุปผลการทำวิจัยและจัดทำรายงานวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) ได้วิธีการสร้างกรณีทดสอบสำหรับการทดสอบแบบอัตโนมัติที่สามารถพัฒนากรณีทดสอบได้อย่างรวดเร็วและรองรับการเปลี่ยนแปลงของซอฟต์แวร์ที่ต้องการทดสอบ
2) ได้เครื่องมือสำหรับช่วยสร้างรายชื่อยูสเซอร์คอนโทรลโดยอัตโนมัติและสามารถปรับปรุงแก้ไขรายละเอียดเมื่อซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลงได้โดยอัตโนมัติ
3) ได้เครื่องมือสำหรับช่วยสร้างกรณีทดสอบโดยอัตโนมัติ
4) ได้เครื่องมือสำหรับช่วยแก้ไขกรณีทดสอบสำหรับการทดสอบแบบอัตโนมัติ
5) ได้เครื่องมือสำหรับตรวจสอบการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ เพื่อช่วยให้นักทดสอบเพิ่มกรณีทดสอบได้อย่างมีประสิทธิภาพ
6) ช่วยให้นักทดสอบลดระยะเวลาที่ใช้ในการสร้างกรณีทดสอบสำหรับการทดสอบแบบอัตโนมัติ
7) ช่วยให้นักทดสอบลดระยะเวลาที่ใช้ในการปรับปรุงแก้ไขกรณีทดสอบเมื่อซอฟต์แวร์ที่ต้องการทดสอบมีการเปลี่ยนแปลง

1.6 ลำดับขั้นตอนในการเสนองานวิจัย

วิทยานิพนธ์ฉบับนี้แบ่งเนื้อหาออกเป็น 6 บทต่อไปนี้ บทที่ 1 บทนำกล่าวถึงความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ของงานวิจัย ขอบเขตของงานวิจัย ขั้นตอนและวิธีการวิจัย ประโยชน์ที่คาดว่าจะได้รับ และบทความวิชาการที่ได้รับการตีพิมพ์ บทที่ 2 กล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง บทที่ 3 นำเสนอภาพรวมการทำงานของเครื่องมือ การวิเคราะห์และออกแบบเครื่องมือ บทที่ 4 กล่าวถึงสภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือ และโครงสร้างส่วนต่อประสานกับผู้ใช้เครื่องมือ บทที่ 5 กล่าวถึงสภาพแวดล้อมที่ใช้ในการทดสอบ ขั้นตอนการทดสอบเครื่องมือ ระบบที่ใช้ในการทดสอบ และผลการทดสอบ และบทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ

1.7 บทความวิชาการที่ได้รับการตีพิมพ์

งานวิจัยนี้ได้รับการตีพิมพ์เป็นบทความทางวิชาการเรื่อง “An Automated Testing Tool Using UI Structure” โดย Nutharat Harnvorawong และ Taratip Suwannasart ซึ่งได้รับคัดเลือกให้นำเสนอในงานประชุมวิชาการ International MultiConference of Engineers and Computer Scientists 2014 (IMECS 2014) และตีพิมพ์ในเอกสาร “Proceedings of ICSE’14” จัดโดย International Association of Engineers (IAENG) ระหว่างวันที่ 12-14 มีนาคม พ.ศ. 2557 ณ เขตการปกครองพิเศษฮ่องกง ประเทศจีน

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

ทฤษฎีที่เกี่ยวข้องสำหรับงานวิจัยนี้มีอยู่หลายทฤษฎี มีรายละเอียดดังนี้

2.1.1 ลักษณะของการทดสอบ

การทดสอบสามารถแบ่งออกเป็น 2 ลักษณะใหญ่ๆ [7] คือ

2.1.1.1 การทดสอบแบบธรรมดา (Manual Testing)

นักทดสอบจะทำตัวเสมือนเป็นผู้ใช้งานจริงของซอฟต์แวร์และทำการทดสอบซอฟต์แวร์ด้วยการใช้งานเช่นเดียวกับผู้ใช้งานซอฟต์แวร์เพื่อค้นหาข้อบกพร่องของซอฟต์แวร์

2.1.1.2 การทดสอบแบบอัตโนมัติ (Automated Testing)

นักทดสอบจะสร้างหรือจัดเตรียมชุดคำสั่งด้วยเครื่องมือสำหรับช่วยทดสอบต่างๆ และสั่งงานผ่านซอฟต์แวร์ช่วยทดสอบให้ดำเนินการทดสอบซอฟต์แวร์ที่ต้องการทดสอบตามที่ได้กำหนดไว้

การทดสอบซอฟต์แวร์แบบอัตโนมัตินิยมนำมาใช้สำหรับการทดสอบเชิงถดถอย เนื่องจากโดยทั่วไปนักทดสอบจะให้ความสนใจกับการทดสอบซอฟต์แวร์ในส่วนที่กำลังพัฒนาเป็นหลัก และจะให้ความสนใจกับส่วนของซอฟต์แวร์ที่มีอยู่เดิมน้อยลงไป ดังนั้นการทดสอบแบบอัตโนมัติจึงมีประโยชน์อย่างมากในการตรวจสอบความถูกต้องของการทำงานเดิมของซอฟต์แวร์

การทดสอบแบบอัตโนมัติเป็นวิธีที่ดีที่จะช่วยลดระยะเวลาการทดสอบและค่าใช้จ่าย อย่างไรก็ตามเครื่องมือสำหรับช่วยในการทดสอบและเทคนิคในการทดสอบยังคงเป็นปัญหาต่อการทดสอบซอฟต์แวร์ เนื่องจากเครื่องมือแต่ละตัวมีความเฉพาะตัว เหมาะสมกับซอฟต์แวร์เพียงบางประเภทและไม่สามารถรองรับการขยายขนาดของการทดสอบได้โดยง่าย

ประโยชน์ของการทดสอบแบบอัตโนมัติมีดังนี้

1) สามารถทำซ้ำได้ (Repeatable) เนื่องจากเป็นชุดคำสั่ง จึงสามารถสั่งงานให้ทำงานซ้ำได้มากตามที่ต้องการ
2) มีความน่าเชื่อถือ (Reliable) เนื่องจากเป็นการทำงานด้วยซอฟต์แวร์คอมพิวเตอร์ การทำงานแต่ละครั้งจึงไม่มีความแตกต่างกัน ทำงานเหมือนเดิมเสมอ
3) สามารถเขียนเป็นชุดคำสั่งโปรแกรมได้ (Programmable) ตัวอย่างเช่น การตรวจสอบสี ตำแหน่งของวัตถุบนหน้าจอ ไม่สามารถตรวจสอบได้โดยง่ายด้วยการทดสอบแบบธรรมดา แต่การเขียนโปรแกรมเพื่อไปอ่านค่าคุณสมบัติต่างๆ เหล่านั้นสามารถช่วยแก้ปัญหาเหล่านี้ได้
4) นำมาใช้ซ้ำได้ (Reusable) นักทดสอบสามารถนำชุดคำสั่งการทดสอบที่มีอยู่แล้วมาใช้ซ้ำได้เพื่อลดเวลาในการพัฒนากรณีทดสอบ

5) มีความรวดเร็ว การกระทำกิจกรรมต่างๆ กับวัตถุบนหน้าจอแสดงผลมีความรวดเร็ว

6) ลดค่าใช้จ่าย เนื่องจากการสร้างชุดคำสั่งทดสอบใช้เวลาครั้งเดียวและสามารถนำไปใช้งานได้มากเท่าที่ต้องการ ไม่ต้องเสียเวลาให้นักทดสอบมาทดสอบแบบธรรมดาด้วยคนซึ่งใช้เวลานาน

2.1.2 กรอบการทำงานของการทดสอบแบบอัตโนมัติ (Automated Testing Framework) [4]

กรอบการทำงานของการทดสอบแบบอัตโนมัติ คือกลุ่มของสมมติฐาน แนวคิด รวมถึงเครื่องมือสำหรับการทำงาน เพื่อสนับสนุนการทดสอบแบบอัตโนมัติ มีหน้าที่หลักคือ

1) กำหนดรูปแบบของวิธีการเขียนกรณีทดสอบและผลลัพธ์ที่คาดหวัง

2) รับผิดชอบการเชื่อมต่อหรือกระทำกิจกรรมใดๆ กับส่วนต่อประสานกับผู้ใช้ (User Interface) ของซอฟต์แวร์ที่ต้องการทดสอบ

3) ทำการทดสอบ

4) รายงานผลการทดสอบในรูปแบบต่างๆ อย่างเหมาะสม

ตัวอย่างของเครื่องมือที่มีใช้งานอยู่ในปัจจุบันมีดังนี้

1) Visual Studio Test Professional พัฒนาโดย Microsoft [8]

2) HP Unified Functional Testing พัฒนาโดย Hewlett-Packard [9]

3) Rational Functional Tester พัฒนาโดย IBM [10]

4) White พัฒนาโดย TestStack [11]

5) Sahi พัฒนาโดย Tyto Software Private [12]

2.1.3 เทคโนโลยีการเข้าถึงวัตถุ (Accessibility Technology) [13, 14]

เทคโนโลยีการเข้าถึงวัตถุเป็นชุดคำสั่งส่วนต่อประสานโปรแกรมประยุกต์ (Application Programming Interface - API) เพื่อเข้าถึงวัตถุที่แสดงบนหน้าจอ (User Interface Element) โปรแกรมเมอร์สามารถเขียนโปรแกรมเพื่อเข้าถึงข้อมูลต่างๆ และสามารถสั่งงานวัตถุเหล่านั้นได้ผ่านทางชุดคำสั่งนี้ เทคโนโลยีการเข้าถึงวัตถุที่เป็นที่นิยมในปัจจุบันมี 2 อย่าง ได้แก่

2.1.3.1 Microsoft Active Accessibility (MSAA) [15]

ชุดคำสั่งส่วนต่อประสานที่ได้รับการพัฒนาและนำเสนอโดยบริษัทไมโครซอฟท์ตั้งแต่ปี ค.ศ. 1997 จุดประสงค์หลักคือเพื่อใช้สำหรับผลิตภัณฑ์ที่เกี่ยวข้องกับเทคโนโลยีการช่วยเหลือ (Assistive Technology - AT) สามารถอ่านรายละเอียดต่างๆ ของวัตถุบนหน้าจอ และสามารถสั่งงานให้วัตถุเหล่านั้นทำงานบางอย่างได้เช่นเดียวกับการสั่งงานจากซอฟต์แวร์ของวัตถุนั้นเอง ซอฟต์แวร์ที่เกี่ยวข้องกับเทคโนโลยีการช่วยเหลือส่วนใหญ่จะถูกผลิตมาเพื่อช่วยเหลือผู้พิการทางด้านต่างๆ เช่น

ซอฟต์แวร์อ่านข้อความสำหรับคนตาบอด ซอฟต์แวร์แสดงสีให้มีความต่างสูงสำหรับผู้ที่มีปัญหาด้านสายตา เป็นต้น

MSAA ทำงานโดยมีพื้นฐานมาจากเทคโนโลยี Component Object Model (COM) ซึ่งเป็นกลไกที่ระบบปฏิบัติการใช้สื่อสารกับซอฟต์แวร์ต่างๆ ซอฟต์แวร์ที่สามารถเข้าถึงผ่านทางเทคโนโลยี MSAA มีเงื่อนไขคือจะต้องทำการอิมพลิเมนต์อินเตอร์เฟซ (Interface) ที่ชื่อว่า IAccessible มิเช่นนั้นจะไม่สามารถเข้าถึงข้อมูลของวัตถุหรือสั่งงานใดๆ ได้

2.1.3.2 Microsoft UI Automation (UIA) [16]

ชุดคำสั่งส่วนต่อประสานที่ได้รับการพัฒนาและนำเสนอโดยบริษัทไมโครซอฟท์ในปี ค.ศ. 2005 มีลักษณะคล้ายกับ MSAA แต่มีความสามารถสูงกว่า ใช้สำหรับผลิตภัณฑ์ที่เกี่ยวข้องกับเทคโนโลยีการช่วยเหลือ และใช้สำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติโดยเฉพาะ รองรับการทำงานกับซอฟต์แวร์หลายประเภท เช่น Win32 และ .NET Application เป็นต้น

2.1.4 แหล่งจัดเก็บข้อมูลวัตถุบนหน้าจอ (User Interface Element Repository)

วัตถุที่แสดงบนหน้าจอแต่ละตัวจะมีรายละเอียดต่างๆ เช่น ข้อความที่แสดง (Label หรือ Caption) ตำแหน่งที่แสดงผลบนหน้าจอ ขนาดของวัตถุ และออโตเมชันไอดี (Automation ID) เป็นต้น การเข้าถึงวัตถุเหล่านี้ผ่านทางเทคโนโลยีการเข้าถึงวัตถุนั้นจะต้องมีการค้นหาวัตถุที่ต้องการก่อน ซึ่งอาจค้นหาโดยใช้รายละเอียดต่างๆ เป็นเงื่อนไขในการค้นหา วิธีที่สะดวกและรวดเร็วที่สุดคือการค้นหาจากออโตเมชันไอดี ซึ่งเปรียบเสมือนหมายเลขประจำตัว วัตถุแต่ละตัวที่อยู่ภายใต้วัตถุพ่อแม่ (Parent) เดียวกันจะมีค่าออโตเมชันไอดีไม่ซ้ำกัน ดังนั้นการจัดเก็บรายชื่อของวัตถุบนหน้าจอต่างๆ ที่ใช้ในการทดสอบแบบอัตโนมัติจึงควรจะเก็บหมายเลขออโตเมชันไอดีนี้ เรียกว่าแหล่งจัดเก็บข้อมูลวัตถุบนหน้าจอ ซึ่งสามารถจัดเก็บในรูปแบบใดก็ได้ขึ้นอยู่กับการออกแบบของนักทดสอบ

2.1.5 ประเภทของยูสเซอร์คอนโทรล

ยูสเซอร์คอนโทรล คือวัตถุที่แสดงบนหน้าจอประเภทหนึ่งที่อนุญาตให้ผู้ใช้งานสามารถสื่อสารกับซอฟต์แวร์ได้ เช่น กล่องข้อความสำหรับระบุชื่อ-นามสกุล ปุ่มตกลง ปุ่มยกเลิก เป็นต้น ยูสเซอร์คอนโทรลที่ใช้งานในซอฟต์แวร์ต่างๆ นั้นมีอยู่มากมายขึ้นอยู่กับชนิดของซอฟต์แวร์ สามารถแบ่งออกได้เป็นหลายกลุ่มหลักๆ เช่น WinControl, WpfControl และ HtmlControl เป็นต้น ในที่นี้จะสนใจเฉพาะยูสเซอร์คอนโทรลที่เป็น WinControl เท่านั้น ซึ่งใช้สำหรับซอฟต์แวร์ที่ทำงานบนระบบปฏิบัติการวินโดวส์และพัฒนาด้วยกลุ่มภาษาโปรแกรมดอทเน็ต ส่วน WpfControl และ HtmlControl นั้นเป็นยูสเซอร์คอนโทรลที่ใช้ในซอฟต์แวร์ที่พัฒนาในรูปแบบดับบลิวพีเอฟ (WPF) และบนเว็บไซต์ตามลำดับ

ตัวอย่างรายชื่อของยูสเซอร์คอนโทรลที่สามารถใช้ในการทดสอบแบบอัตโนมัติมีดังนี้ [17]

- Button คือ ปุ่มใช้สำหรับกด เพื่อให้ทำงานบางอย่างตามที่ได้กำหนดไว้
- Check Box คือ กล่องสำหรับทำเครื่องหมายเพื่อเลือกหรือไม่เลือกรายการ

- Combo Box คือ กล่องที่บรรจุรายการต่างๆ เพื่อให้ผู้ใช้งานเลือกรายการที่ต้องการ
- Edit คือ ช่องแสดงข้อความ ผู้ใช้งานสามารถพิมพ์ข้อความลงไปเพื่อให้ซอฟต์แวร์นำไปประมวลผลหรือทำงานต่อได้
- Group คือ การจัดกลุ่มของยูสเซอร์คอนโทรลเข้าไว้ด้วยกัน โดยอาจแสดงเส้นกรอบล้อมรอบไว้ให้เห็นหรือไม่ก็ได้
- Hyperlink คือ ข้อความที่ถูกกำหนดตำแหน่งของการเชื่อมต่อบางอย่าง เช่น เว็บไซต์ อีเมล เป็นต้น เมื่อผู้ใช้งานคลิกที่ Hyperlink จะเป็นการเปิดเว็บไซต์หรือส่งอีเมลไปยังตำแหน่งที่กำหนดไว้
- List คือ กล่องสำหรับแสดงรายการต่างๆ ให้ผู้ใช้งานเลือก
- List Item คือ รายการต่างๆ ที่แสดงอยู่ในกล่องแสดงรายการ
- Pane คือ การจัดกลุ่มของยูสเซอร์คอนโทรลเข้าไว้ด้วยกัน เพื่อง่ายต่อการเขียนโปรแกรม
- Radio Button คือ ตัวเลือกสำหรับผู้ใช้งาน โดยผู้ใช้งานจะสามารถเลือกได้แค่เพียง 1 รายการเท่านั้น จากตัวเลือกที่อยู่ในกลุ่มเดียวกัน เช่น ตัวเลือกของเพศจะมี ชายและหญิง เป็นต้น
- Scroll Bar คือ ส่วนที่แสดงว่าส่วนแสดงผลในขณะนี้แสดงรายละเอียดช่วงใดอยู่ และใช้สำหรับเลื่อนการแสดงผลไปยังส่วนอื่นที่ต้องการ เช่น ข้อมูลมี 100 บรรทัด แต่หน้าจอแสดงผลได้ครั้งละ 20 บรรทัด ผู้ใช้งานสามารถใช้ Scroll Bar เพื่อเลื่อนดูข้อมูลส่วนอื่นๆ เป็นต้น
- Spinner คือ ตัวเลือกที่ผู้ใช้งานสามารถเลือกตัวเลขที่ต้องการ โดยจะมี 2 ปุ่มเพื่อเพิ่มค่าและลดค่า
- Tab คือ กลุ่มของ Tab Item ต่างๆ
- Tab Item คือ หน้าสำหรับแสดงผลต่างๆ มักจะนำมาใช้ในการทำงานที่มียูสเซอร์คอนโทรลเป็นจำนวนมาก โดยจะแบ่งยูสเซอร์คอนโทรลที่มีความเกี่ยวข้องกันแสดงอยู่ในหน้าเดียวกัน ผู้ใช้งานสามารถเลือกให้แสดงยูสเซอร์คอนโทรลอื่นๆ ได้โดยการเลือกหน้าอื่นๆ
- Table คือ ตารางสำหรับแสดงข้อมูลต่างๆ
- Text คือ กล่องแสดงข้อความ ผู้ใช้งานสามารถพิมพ์ข้อความลงไปเพื่อให้ซอฟต์แวร์นำไปประมวลผลหรือทำงานต่อได้
- Tree คือ การแสดงข้อมูลต่างๆ โดยอยู่ในรูปแบบโครงสร้างต้นไม้ สามารถแสดงหรือซ่อนข้อมูลรายการรองที่อยู่ภายใต้รายการหลักดังกล่าวได้
- Tree Item คือ ข้อมูลแต่ละรายการที่แสดงอยู่บน Tree

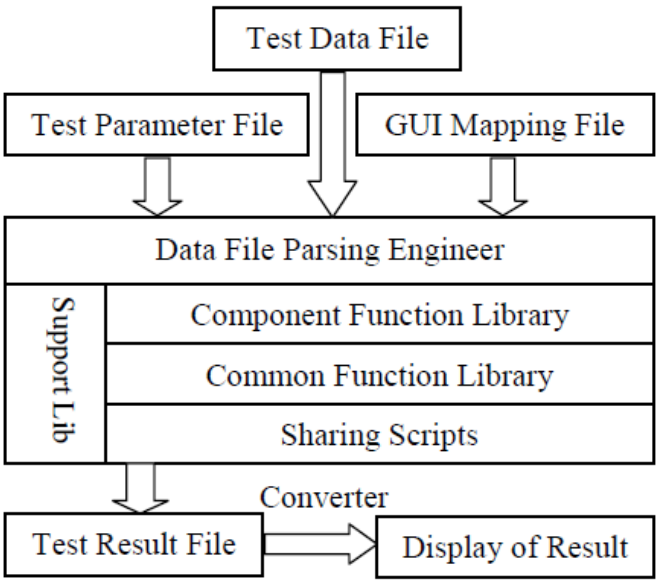
อย่างไรก็ตามยูสเซอร์คอนโทรลบางชนิดก็ไม่สามารถนำมาใช้สำหรับการทดสอบแบบอัตโนมัติในงานวิจัยได้ เช่น ยูสเซอร์คอนโทรลที่แสดงข้อมูลในลักษณะกราฟิก หรือยูสเซอร์คอนโทรล

ที่โปรแกรมเมอร์พัฒนาขึ้นมาใช้เอง เป็นต้น ซึ่งหากต้องการทำการทดสอบกับยูสเซอร์คอนโทรลเหล่านี้ จะต้องทำการปรับปรุงซอฟต์แวร์ในขั้นตอนการพัฒนาซอฟต์แวร์ให้รองรับต่อการทดสอบเพิ่มเติม ซึ่งอยู่นอกเหนือขอบเขตของงานวิจัยนี้

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 Design and Implementation of GUI Automated Testing Framework Based on XML [18]

งานวิจัยดังกล่าวนำเสนอกรอบการทำงานของการทดสอบแบบอัตโนมัติโดยจัดเก็บกรณีทดสอบและข้อมูลทดสอบแยกออกจากกัน ซึ่งจัดเก็บในรูปแบบไฟล์เอกซ์เอ็มแอล (XML) และจัดเก็บออโตเมชันไอดีไว้ในรูปแบบคู่ลำดับในไฟล์ข้อความ (Text File) เรียกว่า GUI Mapping File โครงสร้างของระบบแสดงได้ดังรูปที่ 2.1



รูปที่ 2.1 โครงสร้างการทำงานของระบบ [18]

จากรูปจะเห็นว่าข้อมูลนำเข้า (Input) มี 3 ส่วนได้แก่ Test Parameter File, Test Data File และ GUI Mapping File ซึ่งล้วนแต่เป็นไฟล์ข้อความ จึงจำเป็นต้องมีส่วนการทำงานที่อ่านข้อมูลนำเข้าเพื่อแปลงให้อยู่ในรูปแบบที่ระบบสามารถเข้าใจได้ เรียกว่า Data File Parsing Engineer หลังจากนั้นจึงเข้าสู่กระบวนการทำงานเพื่อทดสอบและแสดงผลการทดสอบออกมาในที่สุด

ตัวอย่างของไฟล์กรณีทดสอบแสดงได้ดังรูปที่ 2.2

```
<WorkFlow Name="CreateNotePad" Enable="Yes" LoopTime="1">
    <TestCase Name="CreateNotePad" Enable="Yes" LoopTime="1">
        <Action Name="TypeHotKey" Enable="Yes" LoopTime="1">
            <Type>Component</Type>
            <Name>DesktopWindow</Name>
            <ActionName>HotKey</ActionName>
            <ComponentType>Window</ComponentType>
            <Parameter>500</Parameter>
            <ExpectedResult></ExpectedResult>
        </Action>
        <Action Name="ClickOkBtn" Enable="Yes" LoopTime="1">
            <Type>Component</Type>
            <Name>OkButton</Name>
            <ActionName>Click</ActionName>
            <ComponentType>Button</ComponentType>
        </Action>
    </TestCase>
    <TestCase Name="SaveNotePad" Enable="Yes" LoopTime="1">
        <Action Name="SelectSaveMenu" Enable="Yes" LoopTime="1">
            <Type>Component</Type>
            <Name>NotepadWindow</Name>
            <ActionName>MenuSelect</ActionName>
            <ComponentType>Window</ComponentType>
            <Parameter>%SaveMenu%</Parameter>
            <ExpectedResult></ExpectedResult>
        </Action>
    </TestCase>
</WorkFlow>
```

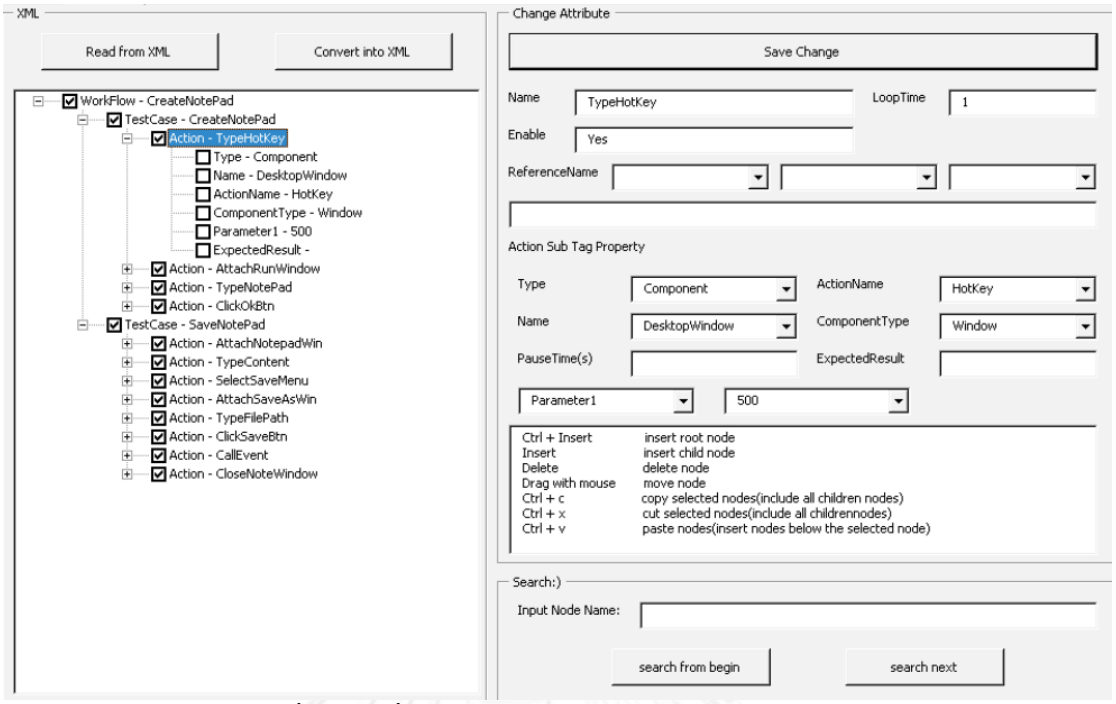
รูปที่ 2.2 ตัวอย่างไฟล์กรณีทดสอบ [18]

รายชื่ออยูสเซอร์คอนโทรลและออโตเมชันไอดีถูกจัดเก็บในไฟล์ GUI Mapping File โดยแยกตามหน้าจอของซอฟต์แวร์ ตัวอย่างไฟล์มีลักษณะดังแสดงในรูปที่ 2.3

Logical Name	Identify name
LoginWindow	“classname=window;index=1”
SaveButton	Caption='save(&S)'

รูปที่ 2.3 ตัวอย่างไฟล์ GUI Mapping File [18]

ถึงแม้ว่าไฟล์นำเข้าส่วนใหญ่จะอยู่ในรูปแบบไฟล์เอกซ์เอ็มแอลซึ่งมีเครื่องมือช่วยจัดการอยู่มากมาย แต่การจัดการกับไฟล์ที่มีจำนวนมากยังคงเป็นขั้นตอนที่ใช้เวลามาก งานวิจัยดังกล่าวจึงได้นำเสนอเครื่องมือช่วยเหลือในการจัดการไฟล์กรณีทดสอบโดยเฉพาะ แสดงได้ดังรูปที่ 2.4

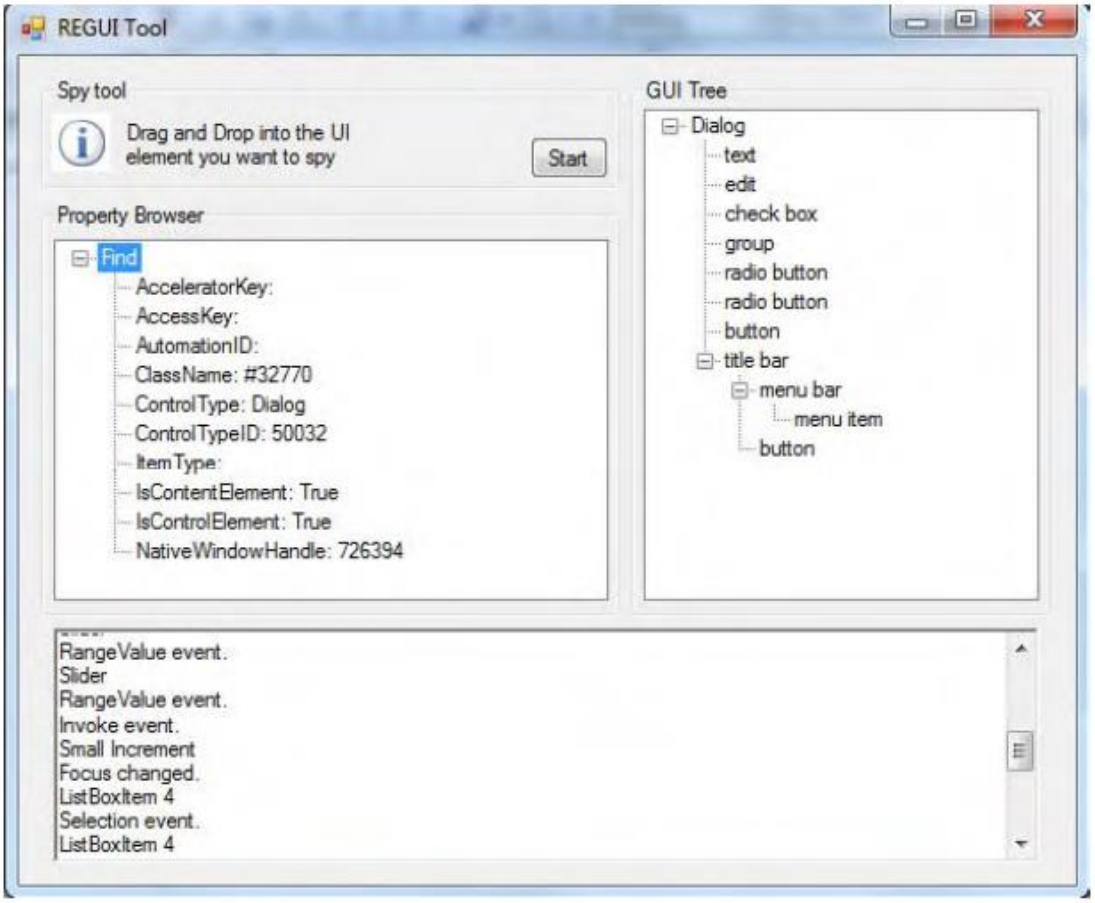


รูปที่ 2.4 เครื่องมือช่วยเหลือจัดการกรณีทดสอบ [18]

จากแนวความคิดของงานวิจัยชิ้นนี้ การแยกเก็บข้อมูลออโตเมชันไอดีของวัตถุออกมาจากไฟล์กรณีทดสอบ เป็นวิธีที่ดีที่ช่วยลดความซับซ้อนของยูสเซอร์คอนโทรลและเพิ่มความสะดวกในการแก้ไขข้อมูล การจัดเก็บไฟล์กรณีทดสอบต่างๆ ล้วนอยู่ในรูปแบบไฟล์เอกซ์เอ็มแอลซึ่งมีข้อดีคือมีความเป็นอิสระไม่ผูกติดกับเทคโนโลยีหรือประเภทของเครื่องมือทดสอบ แต่อย่างไรก็ตามการอ่านข้อมูลจากไฟล์เอกซ์เอ็มแอลจะต้องได้รับการประมวลผลก่อนเสมอเพื่อให้ซอฟต์แวร์สามารถเข้าใจได้ ทำให้มีค่าใช้จ่ายเบื้องต้นเกิดขึ้นอย่างหลีกเลี่ยงไม่ได้ และการจัดเก็บข้อมูลออโตเมชันไอดีในรูปแบบคู่ลำดับในไฟล์ข้อความธรรมดา ทำให้ยากแก่การตรวจสอบหรือนำไปใช้งาน เช่น การตรวจสอบลำดับชั้นความสัมพันธ์ระหว่างยูสเซอร์คอนโทรล การตรวจสอบการซ้ำซ้อนของข้อมูล เป็นต้น

2.2.2 Reverse Engineering of GUI Models for Testing [19]

งานวิจัยดังกล่าวนำเสนอวิธีการจัดเตรียมข้อมูลรายชื่อยูสเซอร์คอนโทรลสำหรับการทดสอบแบบอัตโนมัติ โดยใช้เครื่องมือวิเคราะห์ซอฟต์แวร์ ณ ขณะรันไทม์ (Runtime) และสร้างไฟล์ข้อมูลรายชื่อยูสเซอร์คอนโทรลให้โดยอัตโนมัติเรียกว่า GUI Mapping Information จัดเก็บอยู่ในรูปแบบไฟล์เอกซ์เอ็มแอล เครื่องมือดังกล่าวมีลักษณะดังรูปที่ 2.5



รูปที่ 2.5 เครื่องมือวิเคราะห์ซอฟต์แวร์ [19]

แนวความคิดในการจัดเก็บรายชื่อยูสเซอร์คอนโทรลให้อยู่ในไฟล์แยกต่างหากออกจากกรณีทดสอบเป็นแนวคิดที่ดี อย่างไรก็ตามการจัดเก็บรายละเอียดดังกล่าวถูกจัดเก็บแยกตามแต่ละหน้าจอของซอฟต์แวร์ที่ต้องการทดสอบในรูปแบบไฟล์เอกซ์เอ็มแอล ซึ่งส่งผลให้มีค่าใช้จ่ายในการประมวลผลก่อนที่จะนำมาใช้งาน ถ้าหากซอฟต์แวร์ที่ต้องการทดสอบมีขนาดใหญ่มีหลายหน้าจอ จำนวนไฟล์จะมีเป็นจำนวนมากและส่งผลให้จัดการมีความลำบากมากขึ้นตามไปด้วย

2.2.3 GUI Testing Made Easy [20]

งานวิจัยดังกล่าวนำเสนอการทดสอบแบบอัตโนมัติโดยใช้โอเพนซอร์ส (Open Source) ชื่อว่า FEST ซึ่งมีลักษณะของกรณีทดสอบอยู่ในรูปการเขียนโปรแกรม โดยแต่ละประเภทของยูสเซอร์คอนโทรลจะมีเมธอด (Method) ที่เหมาะสมสำหรับการทำงาน เรียกว่า Fluent Interface ยกตัวอย่างเช่น การกรอกข้อความในกล่องข้อความหนึ่ง และกดปุ่ม แล้วทำการตรวจสอบความถูกต้องที่ข้อความที่แสดงอยู่อีกที่หนึ่ง ถ้าหากใช้ Fluent Interface คำสั่งการทำงานจะเป็นดังรูปที่ 2.6

```
window.textBox("anagramField")
      .enterText("abstraction");
window.button("guessButton").click();
window.label("resultLabel")
      .requireText("Correct!");
```

รูปที่ 2.6 ตัวอย่างโค้ดโปรแกรมที่ใช้ Fluent Interface [20]

แต่หากไม่ได้ใช้ Fluent Interface คำสั่งการทำงานจะเป็นดังรูปที่ 2.7

```
JFrameOperator frmAnagrams =
  new JFrameOperator("Anagrams");
JLabelOperator lblGuess =
  new JLabelOperator(frmAnagrams,
                     "Your Guess:");
JTextField tf =
  (JTextField)lblGuess.getLabelFor();
JTextFieldOperator txtGuess =
  new JTextFieldOperator(tf);
txtGuess.typeText("abstraction");
new JButtonOperator(frmAnagrams,
                    "Guess").push();
JLabelOperator lblFeedback =
  new JLabelOperator(frmAnagrams, 2);
String expectedMessage = "Correct";
assertEquals("Evaluation was wrong:",
             expectedMessage,
             lblFeedback.getText());
```

รูปที่ 2.7 ตัวอย่างโค้ดโปรแกรมที่ไม่ได้ใช้ Fluent Interface [20]

จะเห็นได้ชัดว่าการสร้างกรณีทดสอบแบบที่เรียกคำสั่งผ่าน Fluent Interface มีความชัดเจน สามารถอ่านและทำความเข้าใจได้ง่ายกว่ากรณีทดสอบแบบที่ไม่ได้ใช้ Fluent Interface นักทดสอบไม่จำเป็นต้องเข้าใจชุดคำสั่งส่วนต่อประสานโปรแกรมประยุกต์ต่างๆ อย่างละเอียด เพียงเข้าใจคำสั่งของ Fluent Interface ซึ่งสั้นและชัดเจนก็เพียงพอ หลังจากนั้นจะเป็นหน้าที่ของไลบรารี (Library) FEST ที่จะทำการค้นหายูสเซอร์คอนโทรลที่ต้องการและสร้างตัวแปรขึ้นมาเพื่อกระทำกิจกรรมที่ต้องการโดยอัตโนมัติ

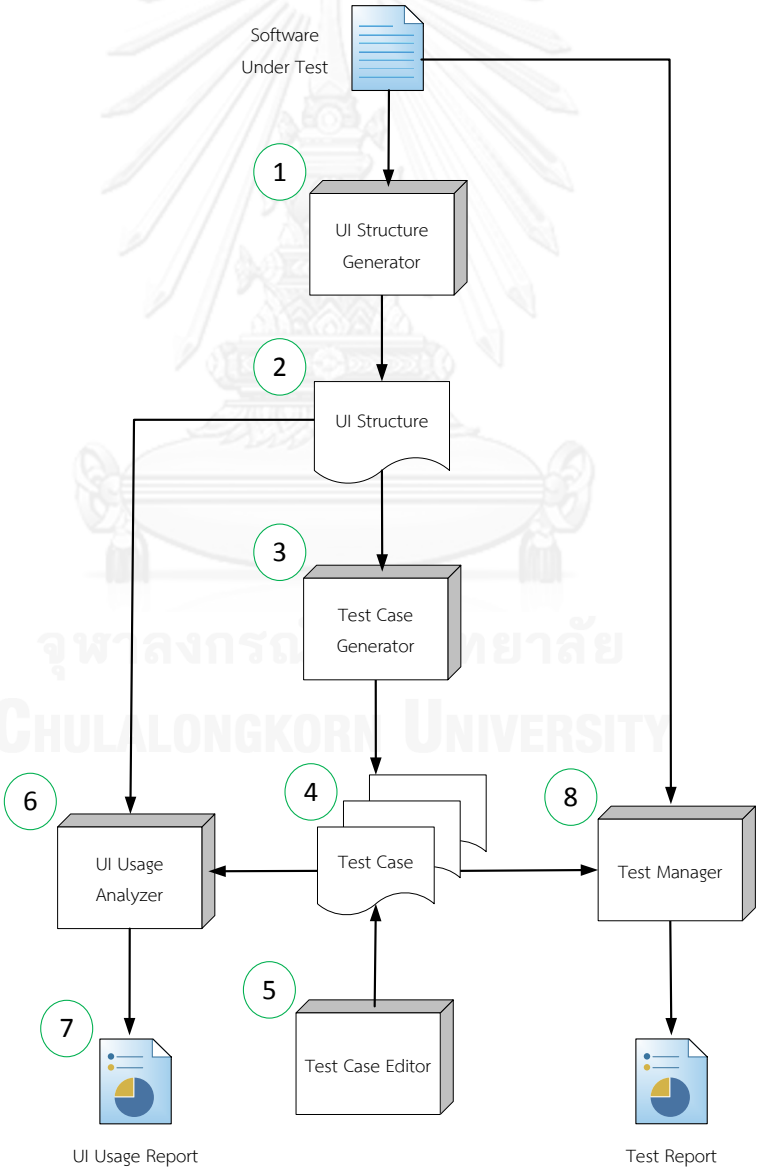
จากงานวิจัยดังกล่าวพบว่าการเขียนกรณีทดสอบถ้าอยู่ในรูปแบบคำสั่งที่สั้นและชัดเจน นักทดสอบสามารถพัฒนากรณีทดสอบได้อย่างรวดเร็ว สามารถควบคุมการไหลของการทำงาน (Control Flow) แบบซับซ้อนได้ง่าย เช่น If-Else, For Loop และ While Loop เป็นต้น และในกรณีที่มีปัญหา

ต้องการดีบัก (Debug) ก็สามารถทำได้อย่างง่ายดายเนื่องจากมีลักษณะเหมือนกับการเขียนโปรแกรมโดยทั่วไป

บทที่ 3
การวิเคราะห์และออกแบบเครื่องมือ

3.1 ภาพรวมการทำงานของเครื่องมือ

งานวิจัยนี้นำเสนอการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติ โดยใช้การจัดเก็บรายชื่อยูสเซอร์คอนโทรลในรูปแบบคลาส เรียกว่า โครงสร้างยูไอ และนำเสนอเครื่องมือช่วยเหลือ มีความสามารถในการทำงานหลัก 4 ส่วน ได้แก่ ส่วนสร้างโครงสร้างยูไอ ส่วนสร้างกรณีทดสอบ ส่วนแก้ไขกรณีทดสอบ และส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ โดยการทำงานต่างๆ ของเครื่องมือทั้งหมดนี้อยู่ภายใต้กรอบการทำงานที่เรียกว่า นีโอออโตเมชันเฟรมเวิร์ก (Neo Automation Framework หรือ NAF) ภาพรวมของระบบสามารถแสดงได้ดังรูปที่ 3.1

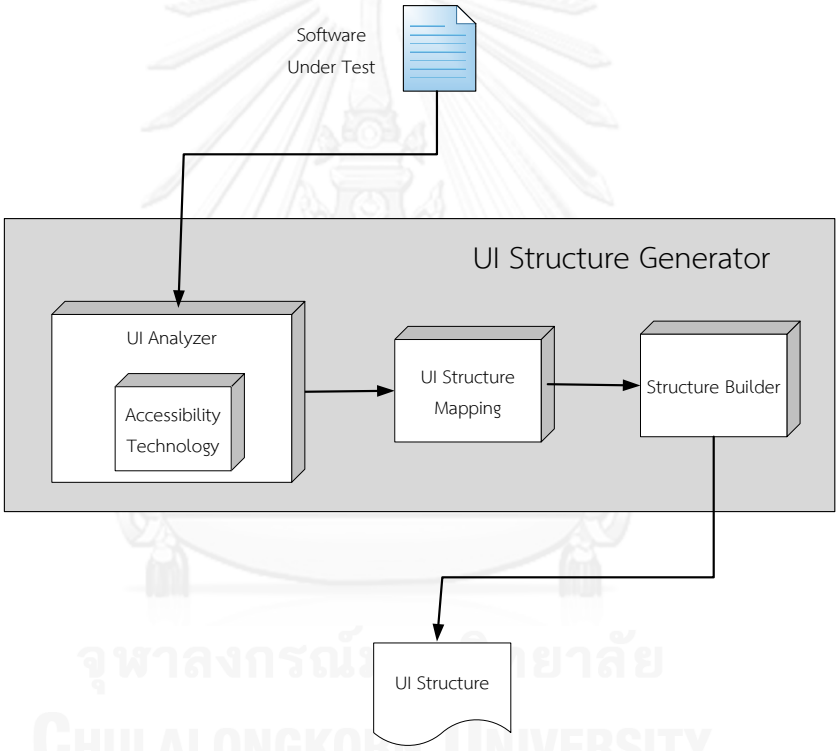


รูปที่ 3.1 ภาพรวมของระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติ

จากรูปที่ 3.1 ระบบมีส่วนประกอบสำคัญอยู่ 8 ส่วน ได้แก่

3.1.1 ส่วนสร้างโครงสร้างยูไอ (UI Structure Generator)

ส่วนสำหรับช่วยสร้างโครงสร้างยูไอ ทำการวิเคราะห์ซอฟต์แวร์ที่ต้องการทดสอบ (Software Under Test - SUT) ที่อยู่ในรูปแบบเอ็กเซ็กคิวเทเบิลไฟล์ (Executable File หรือ .exe) โดยตรง ไม่ต้องใช้รหัสต้นฉบับ (Source Code) ในการทดสอบ ถึงแม้ว่าในบางกรณีนักทดสอบสามารถเข้าถึงรหัสต้นฉบับของซอฟต์แวร์ที่ต้องการทดสอบได้ แต่การทดสอบแบบอัตโนมัติในงานวิจัยนี้จะทดสอบซอฟต์แวร์ที่ต้องการทดสอบโดยตรงเท่านั้นโดยไม่นำรหัสต้นฉบับมาพิจารณา ส่วนสร้างโครงสร้างยูไอมีส่วนประกอบในการทำงาน 3 ส่วนดังรูปที่ 3.2



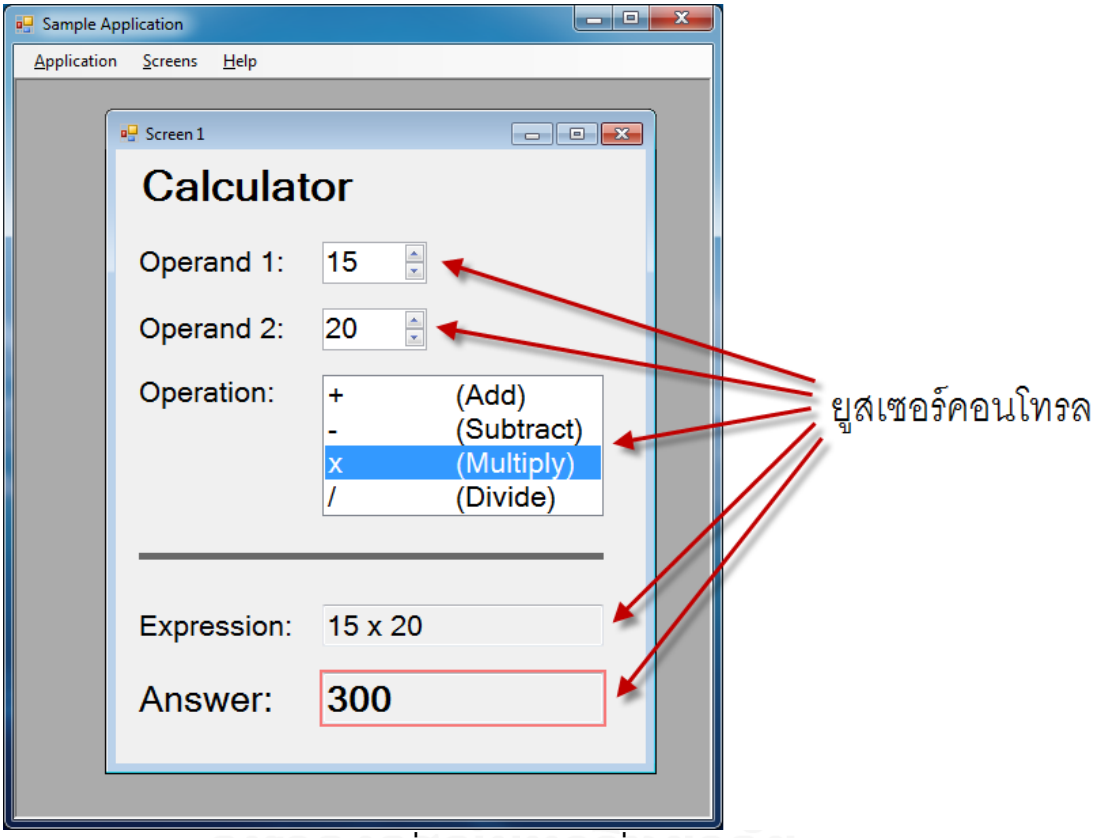
รูปที่ 3.2 ส่วนประกอบของ UI Structure Generator

3.1.1.1 ส่วนวิเคราะห์ยูไอ (UI Analyzer)

ทำหน้าที่ตรวจสอบซอฟต์แวร์ที่ต้องการทดสอบ ณ ขณะรันไทม์โดยเรียกใช้ชุดคำสั่งส่วนต่อประสานโปรแกรมประยุกต์ของเทคโนโลยีการเข้าถึงวัตถุ เช่น MSAA หรือ UIA ซึ่งเป็นเครื่องมือสำหรับการทำงานเกี่ยวกับวัตถุ (Object) ต่างๆ ที่เป็นส่วนต่อประสานกับผู้ใช้ พัฒนาโดยบริษัทไมโครซอฟท์ (Microsoft) เพื่อให้โปรแกรมเมอร์นำไปใช้วิเคราะห์ว่าในขณะนั้นซอฟต์แวร์มียูสเซอร์คอนโทรลอะไรแสดงอยู่และมีคุณสมบัติ (Property) เป็นอย่างไร โดยวิเคราะห์ลงไปเป็นลำดับชั้น (Hierarchy) ตามโครงสร้างของยูสเซอร์คอนโทรลที่ถูกออกแบบไว้โดยโปรแกรมเมอร์ เมื่อส่วน

วิเคราะห์ยูไอได้รับข้อมูลยูสเซอร์คอนโทรลจากเทคโนโลยีการเข้าถึงวัตถุแล้ว จะรวบรวมค่าออโตเมชันไอดี ขนาด และตำแหน่งของยูสเซอร์คอนโทรลแต่ละตัวไว้ เพื่อนำไปใช้ในขั้นตอนถัดไป

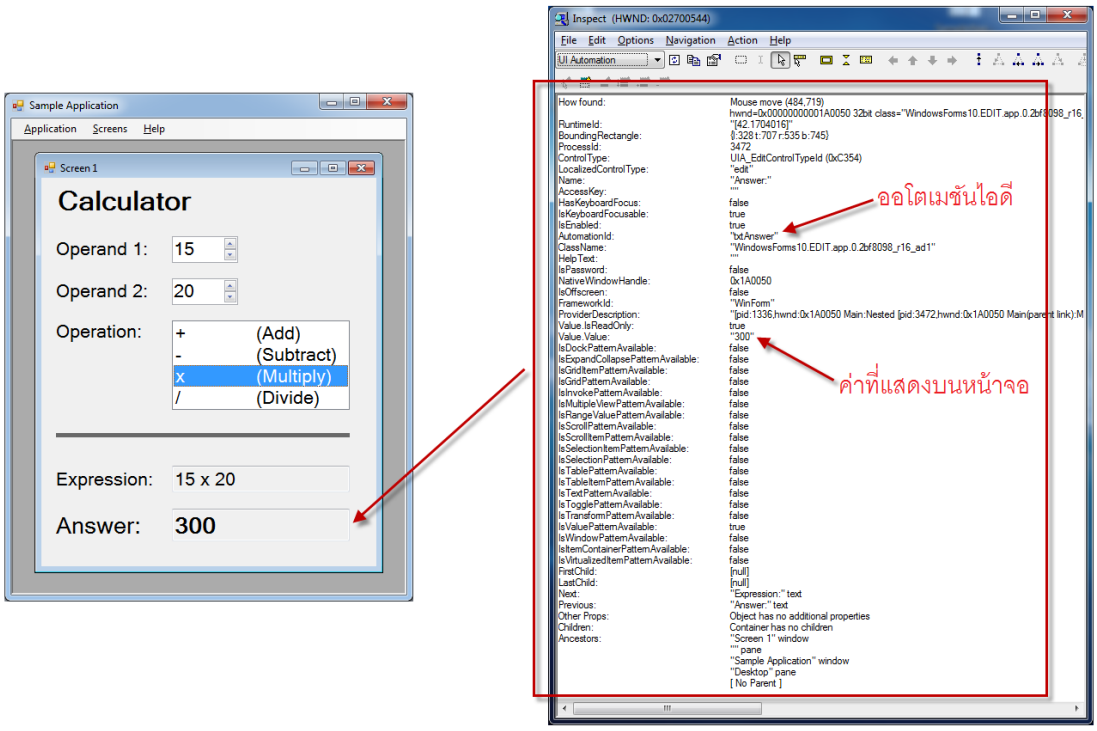
รูปที่ 3.3 แสดงตัวอย่างของซอฟต์แวร์ที่ต้องการทดสอบ มีหน้าจอชื่อ Screen 1 เป็นโปรแกรมเครื่องคิดเลข มียูสเซอร์คอนโทรลเพื่อให้ระบุจำนวน 2 จำนวน เลือกการกระทำ บวก ลบ คูณ หรือหาร และมียูสเซอร์คอนโทรลสำหรับแสดงคำตอบ



รูปที่ 3.3 โปรแกรมเครื่องคิดเลข

เมื่อใช้เครื่องมือที่ใช้เทคโนโลยีเข้าถึงวัตถุตรวจสอบซอฟต์แวร์ จะแสดงรายละเอียดต่างๆ เช่น ค่าออโตเมชันไอดี ซึ่งเป็นค่าเฉพาะตัว ไม่ซ้ำกัน ใช้สำหรับอ้างอิงในกรณีที่จะสั่งงานใดๆ กับยูสเซอร์คอนโทรลนั้น ตำแหน่งของยูสเซอร์คอนโทรล (Bounding Rectangle) และข้อความที่แสดงอยู่ (Text) เป็นต้น

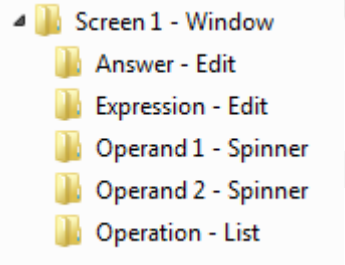
รูปที่ 3.4 แสดงให้เห็นคุณสมบัติของยูสเซอร์คอนโทรลสำหรับแสดงคำตอบที่อ่านค่าโดยเครื่องมือที่ใช้เทคโนโลยีเข้าถึงวัตถุ จากรูปจะเห็นว่ายูสเซอร์คอนโทรลมีคุณสมบัติต่างๆ เป็นจำนวนมาก ซึ่งคุณสมบัติที่สำคัญและน่าสนใจ ได้แก่ ค่าออโตเมชันไอดีคือ txtAnswer และค่าที่แสดงบนหน้าจอคือ 300 เป็นต้น



รูปที่ 3.4 ตัวอย่างคุณสมบัติของยูสเซอร์คอนโทรล

3.1.1.2 ส่วนความสัมพันธ์โครงสร้างยูไอ (UI Structure Mapping)

ส่วนนี้ทำหน้าที่บันทึกข้อมูลยูสเซอร์คอนโทรลที่ถูกจัดวางอยู่ในซอฟต์แวร์และค่าออโตเมชันไอดีที่ตรวจสอบมาได้จากส่วนวิเคราะห์ยูไอ โดยจัดเก็บอยู่ในรูปแบบโครงสร้างต้นไม้ภายในหน่วยความจำ แสดงได้ดังรูปที่ 3.5



รูปที่ 3.5 โครงสร้างของยูสเซอร์คอนโทรล

3.1.1.3 ส่วนสร้างโครงสร้าง (Structure Builder)

เมื่อส่วนวิเคราะห์ยูไอตรวจสอบซอฟต์แวร์เรียบร้อยแล้ว จะได้โครงสร้างของยูสเซอร์คอนโทรลของซอฟต์แวร์ทั้งหมดอยู่ในรูปแบบลำดับชั้น และจะถูกบันทึกลงไปยังไฟล์ในรูปแบบที่กำหนด ในที่นี้จะเป็นรูปแบบคลาสในภาษาซีชาร์ป

นอกจากนี้ส่วนสร้างโครงสร้างยูไอยังสามารถช่วยแก้ไขโครงสร้างยูไอที่ถูกสร้างไปแล้วให้โดยอัตโนมัติในกรณีที่โปรแกรมเมอร์มีการปรับปรุงแก้ไขซอฟต์แวร์และการจัดวาง (Layout) ของยูสเซอร์คอนโทรลบนหน้าจอมีการเปลี่ยนแปลงไป

3.1.2 โครงสร้างยูไอ (UI Structure)

ข้อมูลยูสเซอร์คอนโทรลของซอฟต์แวร์ที่ต้องการทดสอบ เก็บอยู่ในรูปแบบคลาสในภาษาซีชาร์ป โดยจัดเก็บแยกตามหน้าจอของซอฟต์แวร์ แต่ละหน้าจอจะมี 1 คลาส ยกตัวอย่างเช่น ในรูปที่ 3.3 มีโครงสร้างยูไอดังรูปที่ 3.6

```
1  using NeoAutomationFramework.Attributes;
2  using NeoAutomationFramework.Controls;
3
4  namespace UIStructure {
5
6      [UIStructureClass(true)]
7      public partial class Screen1 : NeoWindow {
8
9          private const string _automationID = "Main.*.Screen1";
10
11         public Screen1() : base(_automationID) {
12
13         }
14
15         [NeoControl("679_634_207_38")]
16         public NeoEdit TxtAnswer {
17             get { return new NeoEdit("Main.*.Screen1.txtAnswer"); }
18         }
19
20         [NeoControl("679_584_207_31")]
21         public NeoEdit TxtExpression {
22             get { return new NeoEdit("Main.*.Screen1.txtExpression"); }
23         }
24
25         [NeoControl("679_366_77_31")]
26         public NeoSpinner NumOperand2 {
27             get { return new NeoSpinner("Main.*.Screen1.numOperand2"); }
28         }
29
30         [NeoControl("679_415_207_104")]
31         public NeoList LstOperation {
32             get { return new NeoList("Main.*.Screen1.lstOperation"); }
33         }
34
35         [NeoControl("679_317_77_31")]
36         public NeoSpinner NumOperand1 {
37             get { return new NeoSpinner("Main.*.Screen1.numOperand1"); }
38         }
39     }
40 }
41
```

รูปที่ 3.6 ตัวอย่างโครงสร้างยูไอ

จากรูปที่ 3.6 จะเห็นว่าเป็นคลาสสำหรับ Screen 1 ซึ่งมีค่าออโตเมชันไอดีเป็น Screen1 แต่เนื่องจากหน้าจอดังกล่าวอยู่ภายใต้ยูสเซอร์คอนโทรลอื่นอีกจึงมีค่าออโตเมชันไอดีของยูสเซอร์

คอนโทรลดังกล่าวนำหน้าอยู่และถูกคั่นด้วยเครื่องหมายมหัพภาค (Period) และมีสมาชิกภายในคลาสอยู่ 5 ตัว ได้แก่ ยูสเซอร์คอนโทรลสำหรับ Operand 1, Operand 2, Operation, Expression และ Answer สมาชิกแต่ละตัวจะมีค่าออโตเมชันไอดีระบุอยู่ เพื่อสำหรับอ้างอิงในการเรียกใช้งานในกรณีทดสอบ ถ้าหากซอฟต์แวร์ที่ต้องการทดสอบมีหลายหน้าจอ จะมีคลาสจำนวนหลายคลาสตามจำนวนหน้าจอ เพื่อให้มีความเป็นระเบียบและสะดวกต่อการสร้างกรณีทดสอบในภายหลังต่อไป

นักทดสอบสามารถสร้างโปรเจกต์ด้วยเครื่องมือไมโครซอฟท์วิชวลสตูดิโอสำหรับคลาสของโครงสร้างยูไอและทำการคอมไพล์ (Compile) เพื่อให้อยู่ในรูปแบบไฟล์แอสเซมบลี (Assembly) ที่มีนามสกุล DLL เพื่อสำหรับใช้งานในขั้นตอนต่อไป

3.1.3 ส่วนสร้างกรณีทดสอบ (Test Case Generator)

ส่วนสร้างกรณีทดสอบจะอ่านข้อมูลจากโครงสร้างยูไอที่ได้จากขั้นตอนก่อนหน้าที่อยู่ในรูปแบบไฟล์แอสเซมบลี ที่มีนามสกุล DLL และนำไปสร้างกรณีทดสอบให้โดยอัตโนมัติ โดยจะสร้างกรณีทดสอบให้กระทำกับยูสเซอร์คอนโทรลทั้งหมดภายในโครงสร้างยูไอ โดยจะสร้างขั้นตอนภายในกรณีทดสอบสำหรับยูสเซอร์คอนโทรลที่แสดงอยู่ทางด้านบนของหน้าจอก่อน เรียงจากซ้ายไปขวาและเรียงลงไปตามลำดับ ทั้งนี้กรณีทดสอบที่ถูกสร้างขึ้นมานั้นอาจไม่สอดคล้องหรือเหมาะสมกับการใช้งานจริง นักทดสอบสามารถปรับแต่งแก้ไขกรณีทดสอบได้ตามความเหมาะสมโดยใช้ส่วนแก้ไขกรณีทดสอบ (Test Case Editor) เพื่อให้สอดคล้องกับการใช้งานจริงของผู้ใช้งานหรือที่เรียกว่า ยูสเซอร์ซิเนริโอ (User Scenario)

ขั้นตอนต่างๆ ภายในกรณีทดสอบที่ถูกสร้างขึ้นโดยอัตโนมัติที่กระทำกับแต่ละยูสเซอร์คอนโทรลนั้นจะเป็นการกระทำมาตรฐานของแต่ละประเภทของยูสเซอร์คอนโทรลที่ถูกกำหนดภายในนีโอออโตเมชันเฟรมเวิร์ก เช่น การกระทำมาตรฐานของปุ่มคือ การกดปุ่ม การกระทำมาตรฐานของเช็คบ็อกซ์คือ การเลือกรายการ เป็นต้น

3.1.4 กรณีทดสอบ (Test Case)

คือกรณีทดสอบที่ถูกสร้างขึ้นโดยส่วนสร้างกรณีทดสอบ จัดเก็บอยู่ในรูปแบบของคลาสในภาษาซีชาร์ป มีลักษณะคล้ายกับการทดสอบระดับหน่วย (Unit Test) ของโปรแกรมเมอร์ คือเป็นการเขียนชุดคำสั่งทดสอบด้วยภาษาการเขียนโปรแกรม (Programming Language) รายละเอียดในกรณีทดสอบจะประกอบไปด้วยชุดคำสั่งสำหรับสั่งงานต่างๆ บนซอฟต์แวร์ที่ต้องการทดสอบ ตัวอย่างของกรณีทดสอบแสดงได้ดังรูปที่ 3.7

```
using System.Drawing;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using NeoAutomationFramework.Common;

namespace UIStructure {

    [TestClass]
    public class Screen1Tests {

        [TestMethod]
        public void TestScreen1() {

            // ##### Initialization Commands #####

            // [Initialization] ObjectName=screen1, UIStructureClass=UIStructure.Screen1
            UIStructure.Screen1 screen1 = new UIStructure.Screen1();

            // ##### Test Steps #####

            // [TestStep] Description=, Name=NumOperand1, Type=NeoAutomationFramework.Controls.NeoSpinner, X=679, Y=317
            screen1.NumOperand1.SetValue("0");

            // [TestStep] Description=, Name=NumOperand2, Type=NeoAutomationFramework.Controls.NeoSpinner, X=679, Y=366
            screen1.NumOperand2.SetValue("0");

            // [TestStep] Description=, Name=LstOperation, Type=NeoAutomationFramework.Controls.NeoList, X=679, Y=415
            screen1.LstOperation.GetSelection();

            // [TestStep] Description=, Name=TxtExpression, Type=NeoAutomationFramework.Controls.NeoEdit, X=679, Y=584
            screen1.TxtExpression.GetValue();

            // [TestStep] Description=, Name=TxtAnswer, Type=NeoAutomationFramework.Controls.NeoEdit, X=679, Y=634
            screen1.TxtAnswer.GetValue();

        }
    }
}

```

รูปที่ 3.7 ตัวอย่างกรณีทดสอบ

จากรูปที่ 3.7 คลาสชื่อ TestScreen1 และเมธอดทดสอบ (Test Method) ชื่อ TestScreen1 จะทำการสร้างตัวแปรเพื่อใช้อ้างถึงหน้าจอ Screen 1 และทำการกำหนดค่าให้กับ Operand 1 และ Operand 2 เป็น 0 และ 0 ตามลำดับ หลังจากนั้นจะทำการอ่านค่า Operation ที่ถูกเลือกและอ่านค่าที่อยู่ภายในยูสเซอร์คอนโทรล Expression และ Answer ตามลำดับ

ในกรณีที่นักทดสอบต้องการปรับปรุงหรือแก้ไขกรณีทดสอบที่มีอยู่แล้ว นักทดสอบสามารถเปิดกรณีทดสอบดังกล่าวขึ้นมาแก้ไขได้ โดยใช้ส่วนแก้ไขกรณีทดสอบ

3.1.5 ส่วนแก้ไขกรณีทดสอบ (Test Case Editor)

ส่วนแก้ไขกรณีทดสอบจะมีหน้าจอให้นักทดสอบสามารถเลือกเปิดกรณีทดสอบที่มีอยู่แล้วขึ้นมาเพื่อเพิ่ม ลบ หรือแก้ไขลำดับของการทำงานของแต่ละขั้นตอนได้โดยง่าย เพื่อให้มีความเหมาะสม เป็นไปตามความต้องการในการทดสอบ

3.1.6 ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ (UI Usage Analyzer)

ส่วนนี้ช่วยวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ โดยตรวจสอบจากการเรียกใช้งานโครงสร้างยูไอว่ามียูสเซอร์คอนโทรลใดบ้างที่ถูกเรียกใช้ในกรณีทดสอบ และยูสเซอร์คอนโทรลใดบ้างที่ไม่ถูกเรียกใช้ในกรณีทดสอบเลย เพื่อช่วยให้นักทดสอบตระหนักถึงจุดที่ยังขาดการทดสอบในซอฟต์แวร์

3.1.7 รายงานการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ (UI Usage Report)

รายงานแสดงผลการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ โดยแสดงรายละเอียดจำนวนของยูสเซอร์คอนโทรลที่ถูกเรียกใช้และไม่ถูกเรียกใช้ในกรณีทดสอบ และแสดงค่าเป็นเปอร์เซ็นต์ของการใช้งานยูสเซอร์คอนโทรลในกรณีทดสอบ สามารถแสดงได้ดังรูปที่ 3.8

	Screen 1
Number of User Controls	5
Used in Test Case	4
Unused	1
UI Usage	80%

รูปที่ 3.8 ตัวอย่างข้อมูลในรายงานการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

3.1.8 ส่วนจัดการการทดสอบ (Test Manager)

ส่วนจัดการการทดสอบทำหน้าที่ควบคุมและจัดการการทดสอบ ซึ่งมีหน้าที่นำกรณีทดสอบไปทำงาน โดยสามารถกำหนดระยะเวลาที่จะใช้ในการทดสอบ กำหนดเครื่องคอมพิวเตอร์สำหรับการทดสอบ (Test Agent) เพื่อกระจายการทำงานไปยังเครื่องเหล่านั้น ตั้งค่าการทำงานบนเครื่องทดสอบ และตรวจสอบความคืบหน้าของการทดสอบได้ เมื่อการทดสอบเสร็จสิ้นลงจะแสดงผลการทดสอบ (Test Result) และรายละเอียดที่เกี่ยวข้องกับการทำงานให้นักทดสอบทราบ เช่น เวลาที่ทำการทดสอบ จำนวนกรณีทดสอบที่ผ่านและไม่ผ่าน เป็นต้น

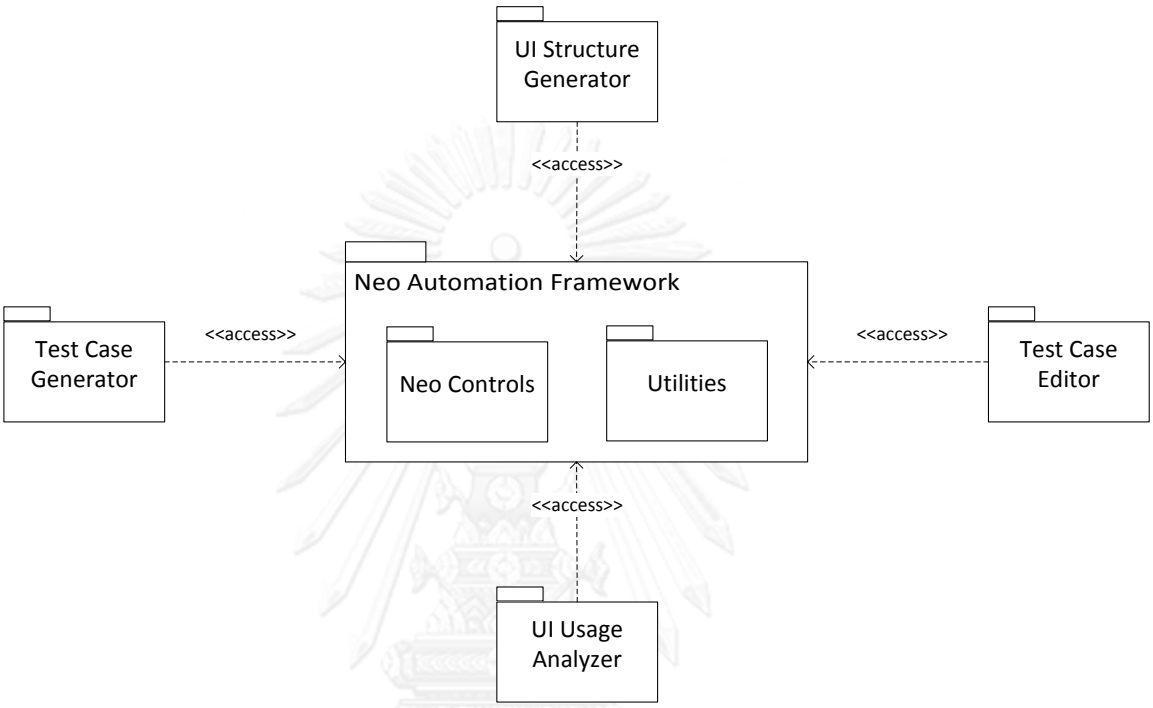
อย่างไรก็ตามส่วนจัดการการทดสอบและผลการทดสอบไม่อยู่ในขอบเขตการพัฒนาของงานวิจัยชิ้นนี้ ทางผู้วิจัยจะใช้เครื่องมือของ Microsoft ที่มีอยู่แล้วในชุดซอฟต์แวร์ Microsoft Visual Studio 2010 เพื่อการวิจัย เครื่องมือดังกล่าวเป็นเครื่องมือสำหรับโปรแกรมเมอร์ใช้พัฒนาซอฟต์แวร์ต่างๆ ไม่ว่าจะเป็นซอฟต์แวร์บนคอมพิวเตอร์ เว็บไซต์ หรือเว็บเซอร์วิส อีกทั้งยังสามารถทำการทดสอบซอฟต์แวร์โดยเรียกการทำงานของกรณีทดสอบที่อยู่ในลักษณะของโปรแกรม เช่น การทดสอบระดับหน่วย เป็นต้น ซึ่งคุณสมบัติดังกล่าวได้ถูกนำมาใช้เพื่อเป็นเครื่องมือในการทำการทดสอบซอฟต์แวร์แบบอัตโนมัติ ด้วยกรณีทดสอบที่ถูกสร้างขึ้นภายในงานวิจัยนี้

3.2 การวิเคราะห์และออกแบบเครื่องมือ

การวิเคราะห์และออกแบบเครื่องมือสามารถอธิบายได้ด้วยแผนภาพยูเอ็มแอล ซึ่งเป็นแผนภาพมาตรฐานในการอธิบายและนำเสนอแนวคิดของซอฟต์แวร์ โดยแผนภาพที่เลือกใช้เพื่อวิเคราะห์และออกแบบเครื่องมือได้แก่ แผนภาพยูสเคส (Use Case Diagram) และแผนภาพคลาส (Class Diagram) โดยมีรายละเอียดดังนี้

3.2.1 แผนภาพแพ็กเกจ

แผนภาพแพ็กเกจใช้แสดงความสัมพันธ์ระหว่างองค์ประกอบต่างๆ ภายในระบบ ซึ่งแผนภาพแพ็กเกจของระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติแสดงได้ดังรูปที่ 3.9



รูปที่ 3.9 แผนภาพแพ็กเกจของระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติ

ระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติประกอบไปด้วยส่วนประกอบต่างๆ 5 ส่วน ได้แก่

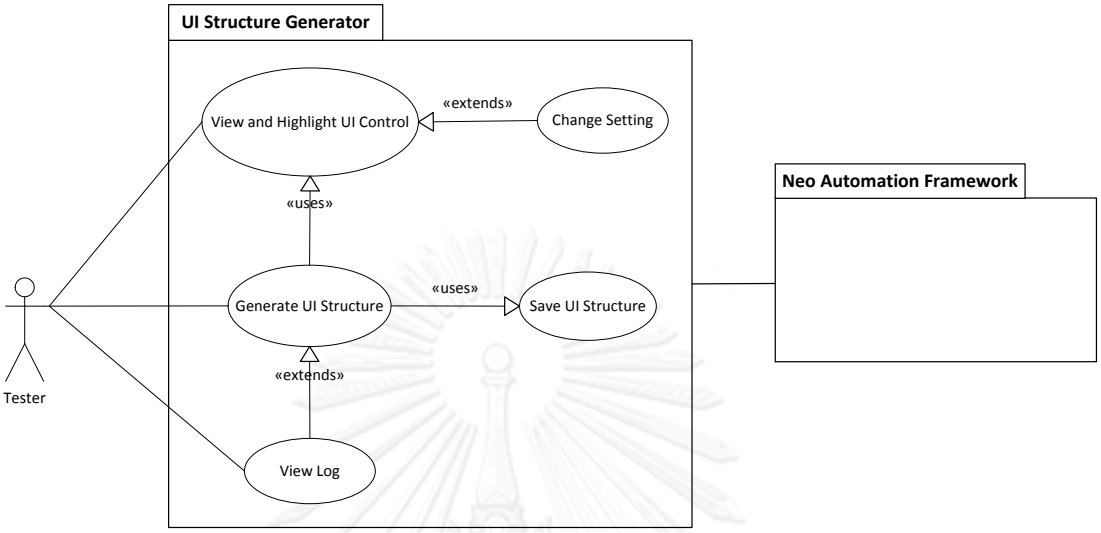
1) ส่วนสร้างโครงสร้างยูไอ (UI Structure Generator)
2) ส่วนสร้างกรณีทดสอบ (Test Case Generator)
3) ส่วนแก้ไขกรณีทดสอบ (Test Case Editor)
4) ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ (UI Usage Analyzer)
5) ส่วนนีโอออโตเมชันเฟรมเวิร์ก (Neo Automation Framework)

ส่วนประกอบหมายเลข 1-4 นั้นจะทำงานโดยเรียกใช้งานหรือส่งคำสั่งไปยังส่วนประกอบหมายเลข 5 ซึ่งเป็นส่วนสำคัญเปรียบเสมือนหัวใจของการทำงานทั้งหมด ประกอบไปด้วยส่วนย่อย คือ นีโอคอนโทรล (Neo Controls) และเครื่องมือทำงาน (Utilities)

3.2.2 แผนภาพยูสเคส

แผนภาพยูสเคสเป็นแผนภาพที่ใช้แสดงการทำงานของระบบในมุมมองของผู้ใช้งาน แผนภาพยูสเคสของระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติแบ่งออกเป็น 5 ส่วน ดังนี้

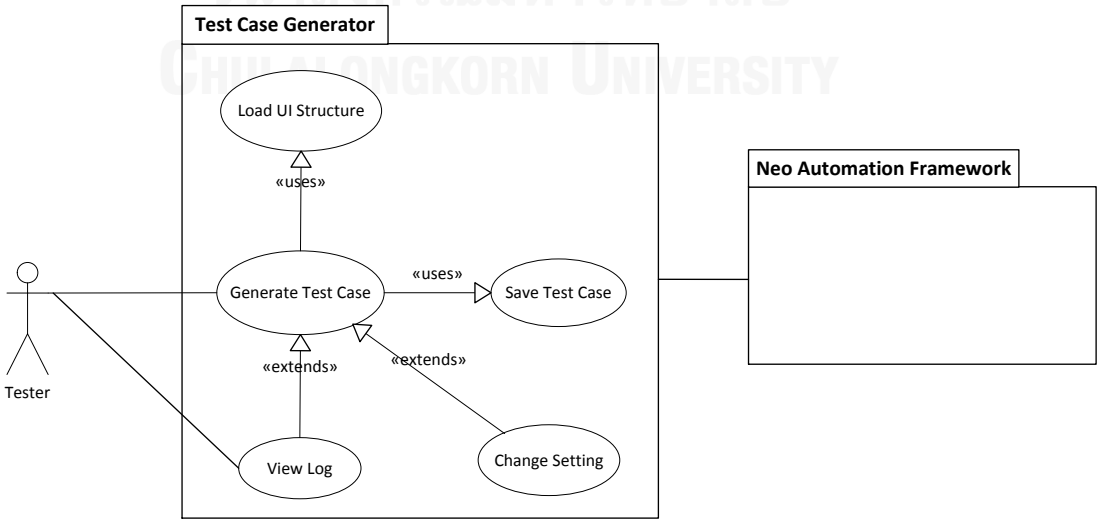
3.2.2.1 UI Structure Generator



รูปที่ 3.10 แผนภาพยูสเคสของส่วนสร้างโครงสร้างยูไอ

รูปที่ 3.10 แสดงให้เห็นแผนภาพยูสเคสของส่วนสร้างโครงสร้างยูไอ นักทดสอบเริ่มต้นการค้นหาและแสดงยูสเซอร์คอนโทรลของซอฟต์แวร์ที่ต้องการทดสอบ (ยูสเคส View and Highlight UI Control) ในขั้นตอนนี้นักทดสอบสามารถตั้งค่าสำหรับการแสดงยูสเซอร์คอนโทรลได้ (ยูสเคส Change Setting) และนำข้อมูลรายละเอียดยูสเซอร์คอนโทรลเหล่านั้นมาสร้างโครงสร้างยูไอ (ยูสเคส Generate UI Structure) ซึ่งจะอยู่ในรูปแบบคลาสในภาษาซีชาร์ป จากนั้นนักทดสอบสามารถบันทึกโครงสร้างยูไอดังกล่าวไปเป็นไฟล์เพื่อใช้ในลำดับถัดไป (ยูสเคส Save UI Structure) ทั้งนี้ในกระบวนการสร้างโครงสร้างยูไอนั้นนักทดสอบสามารถดูรายละเอียดการทำงานที่ผ่านมาได้ (ยูสเคส View Log)

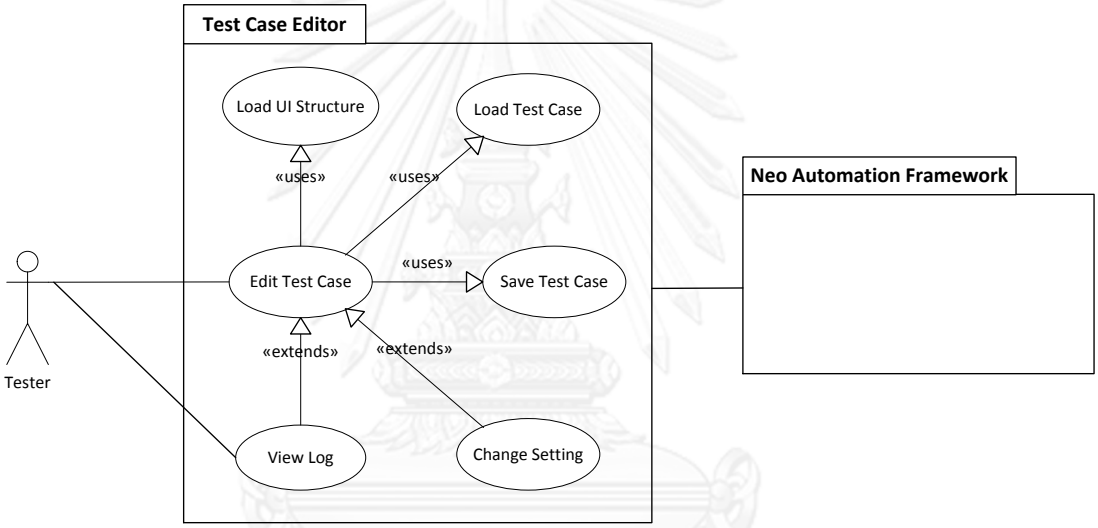
3.2.2.2 Test Case Generator



รูปที่ 3.11 แผนภาพยูสเคสของส่วนสร้างกรณีทดสอบ

รูปที่ 3.11 แสดงให้เห็นแผนภาพยูสเคสของส่วนสร้างกรณีทดสอบ นักทดสอบทำการนำเข้าข้อมูลโครงสร้างยูไอที่ต้องการสร้างกรณีทดสอบ (ยูสเคส Load UI Structure) และดำเนินการสร้างกรณีทดสอบแบบอัตโนมัติ (ยูสเคส Generate Test Case) ในขั้นตอนนี้นักทดสอบสามารถระบุชื่อของเนมสเปซ คลาส และเมธอดทดสอบที่ต้องการได้ อีกทั้งสามารถตั้งค่าการทำงานสำหรับกรณีทดสอบได้ (ยูสเคส Change Setting) ผลลัพธ์จะอยู่ในรูปแบบเมธอดทดสอบแบบการทดสอบระดับหน่วยในภาษาซีชาร์ป จากนั้นนักทดสอบสามารถบันทึกกรณีทดสอบไปเป็นไฟล์เพื่อใช้ในการทดสอบต่อไป (ยูสเคส Save Test Case) ทั้งนี้ในกระบวนการสร้างกรณีทดสอบนั้นนักทดสอบสามารถดูรายละเอียดการทำงานที่ผ่านมาได้ (ยูสเคส View Log)

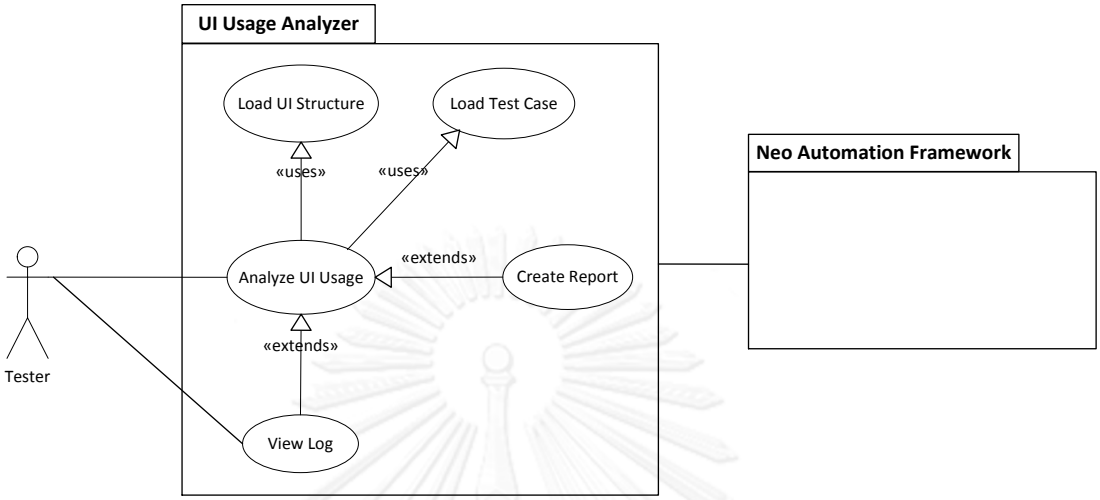
3.2.2.3 Test Case Editor



รูปที่ 3.12 แผนภาพยูสเคสของส่วนแก้ไขกรณีทดสอบ

รูปที่ 3.12 แสดงให้เห็นแผนภาพยูสเคสของส่วนแก้ไขกรณีทดสอบ นักทดสอบทำการนำเข้าข้อมูลโครงสร้างยูไอ (ยูสเคส Load UI Structure) และนำเข้าข้อมูลกรณีทดสอบที่ต้องการแก้ไข (ยูสเคส Load Test Case) แล้วจึงดำเนินการขั้นตอนแก้ไขกรณีทดสอบ (ยูสเคส Edit Test Case) ในขั้นตอนนี้นักทดสอบสามารถเพิ่ม ลบ แก้ไขการทำงานในแต่ละขั้นตอนภายในกรณีทดสอบได้ อีกทั้งสามารถตั้งค่าการทำงานสำหรับกรณีทดสอบได้ (ยูสเคส Change Setting) จากนั้นนักทดสอบสามารถบันทึกการแก้ไขของกรณีทดสอบดังกล่าว (ยูสเคส Save Test Case) ทั้งนี้ในกระบวนการแก้ไขกรณีทดสอบนั้นนักทดสอบสามารถดูรายละเอียดการทำงานที่ผ่านมาได้ (ยูสเคส View Log)

3.2.2.4 UI Usage Analyzer



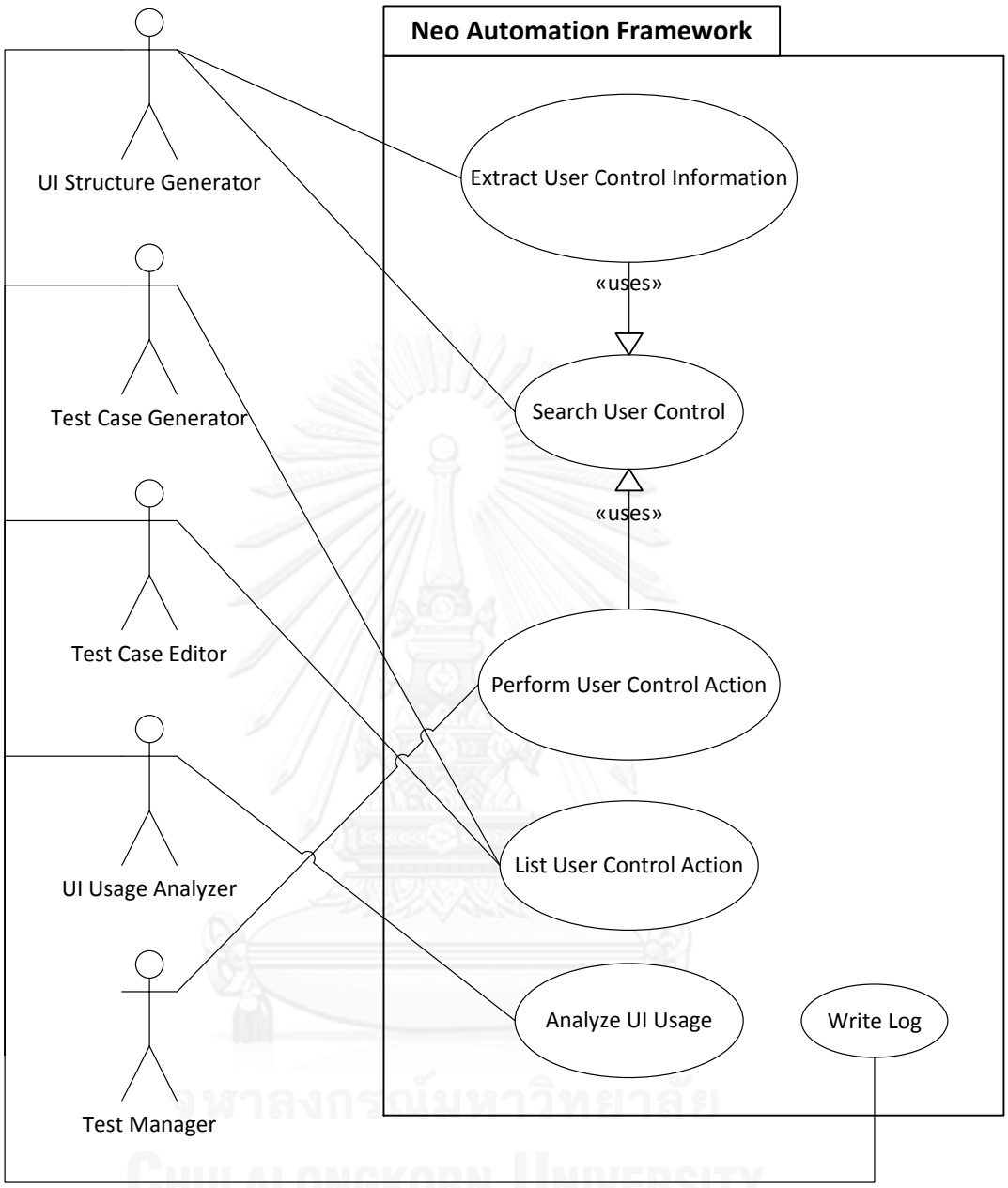
รูปที่ 3.13 แผนภาพยูสเคสของส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

รูปที่ 3.13 แสดงให้เห็นแผนภาพยูสเคสของส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ นักทดสอบทำการนำเข้าข้อมูลโครงสร้างยูไอ (ยูสเคส Load UI Structure) และนำเข้าข้อมูลกรณีทดสอบที่ต้องการวิเคราะห์ (ยูสเคส Load Test Case) แล้วจึงดำเนินการขั้นตอนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ (ยูสเคส Analyze UI Usage) ผลลัพธ์จะแสดงผ่านทางหน้าจอ นักทดสอบสามารถเลือกพิมพ์เป็นรายงานได้ (ยูสเคส Create Report) ทั้งนี้ในกระบวนการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบนั้นนักทดสอบสามารถดูรายละเอียดการทำงานที่ผ่านมาได้ (ยูสเคส View Log)

3.2.2.5 Neo Automation Framework

รูปที่ 3.14 แสดงให้เห็นแผนภาพยูสเคสของนีโอออโตเมชันเฟรมเวิร์ก ส่วนสร้างโครงสร้างยูไอสามารถค้นหายูสเซอร์คอนโทรล (ยูสเคส Search User Control) และสามารถวิเคราะห์ข้อมูลของยูสเซอร์คอนโทรลได้ (ยูสเคส Extract User Control Information) ส่วนสร้างกรณีทดสอบและส่วนแก้ไขกรณีทดสอบสามารถแสดงรายชื่อการกระทำของแต่ละประเภทของยูสเซอร์คอนโทรลได้ (ยูสเคส List User Control Action) ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบสามารถเรียกใช้งานการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลได้ (ยูสเคส Analyze UI Usage) ส่วนจัดการการทดสอบสามารถสั่งงานให้ยูสเซอร์คอนโทรลทำงานตามคำสั่งที่กำหนดไว้ในกรณีทดสอบได้ (ยูสเคส Perform User Control Action) และการบันทึกรายละเอียดการทำงาน (ยูสเคส Write Log) สามารถถูกเรียกใช้งานได้จากส่วนต่างๆ

รายละเอียดของแต่ละยูสเคสแสดงอยู่ในภาคผนวก ก.



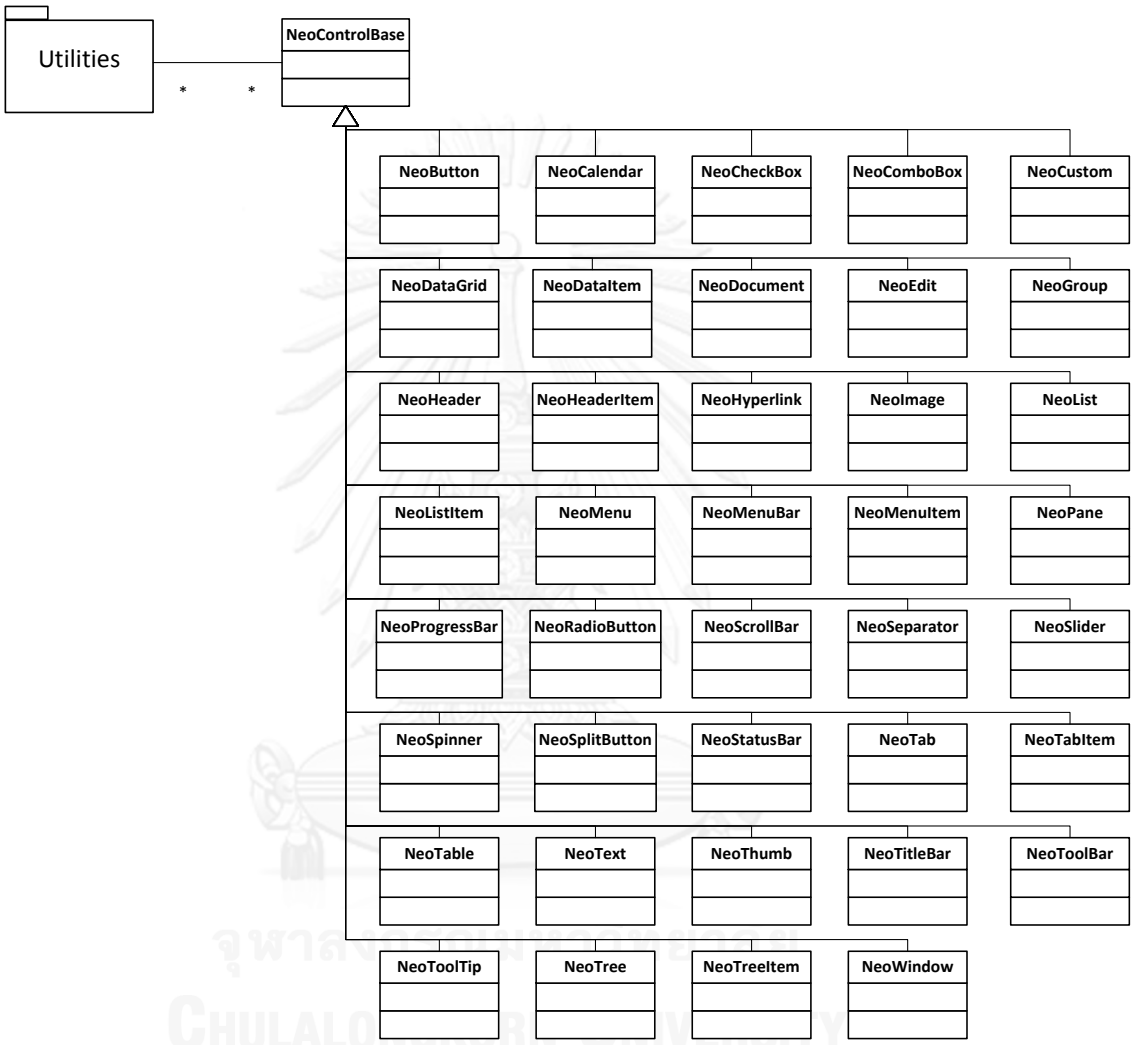
รูปที่ 3.14 แผนภาพยูสเคสของนีโอออโตเมชันเฟรมเวิร์ก

3.2.3 แผนภาพคลาส

แผนภาพคลาสใช้แสดงคุณสมบัติของคลาสและความสัมพันธ์ระหว่างคลาสต่างๆ ภายในระบบ เนื่องจากคลาสทั้งหมดภายในนีโอออโตเมชันเฟรมเวิร์กมีเป็นจำนวนมาก ผู้วิจัยจึงจะนำเสนอเฉพาะคลาสที่มีความสำคัญต่อการทำงานในแต่ละส่วนเท่านั้น แผนภาพคลาสของระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติมีดังนี้

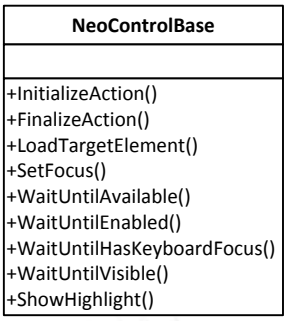
3.2.3.1 นีโอคอนโทรล

ยูสเซอร์คอนโทรลชนิดต่างๆ ที่ใช้ภายในนีโอออโตเมชันเฟรมเวิร์ก เรียกว่า นีโอคอนโทรล แผนภาพคลาสที่เกี่ยวข้องกับนีโอคอนโทรลแสดงได้ดังรูปที่ 3.15



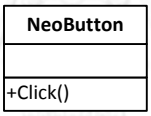
รูปที่ 3.15 แผนภาพคลาสที่เกี่ยวข้องกับนีโอคอนโทรล

1) คลาส NeoControlBase คือ คลาสแม่ของทุกนีโอคอนโทรล หรือยูสเซอร์คอนโทรลที่สามารถใช้งานได้ภายในระบบการทดสอบซอฟต์แวร์แบบอัตโนมัติของนีโอออโตเมชันเฟรมเวิร์ก มีฟังก์ชันการทำงานหลักที่จำเป็นต่อทุกยูสเซอร์คอนโทรล เช่น SetFocus() และ WaitUntilAvailable() เป็นต้น มีการติดต่อกับคลาสในส่วนอื่น เช่น แพ็กเกจ Utilities ภายในนีโอออโตเมชันเฟรมเวิร์ก รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.16



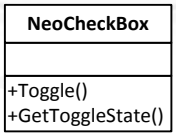
รูปที่ 3.16 คลาส NeoControlBase

2) คลาส NeoButton คือ คลาสที่ใช้สำหรับควบคุมการทำงานของยูสเซอร์คอนโทรลชนิด Button โดยมีฟังก์ชันการทำงานที่เกี่ยวข้องกับปุ่ม เช่น Click() เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.17



รูปที่ 3.17 คลาส NeoButton

3) คลาส NeoCheckBox คือ คลาสที่ใช้สำหรับควบคุมการทำงานของยูสเซอร์คอนโทรลชนิด Check Box โดยมีฟังก์ชันการทำงานที่เกี่ยวข้องกับเช็คบ็อกซ์ เช่น Toggle() เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.18

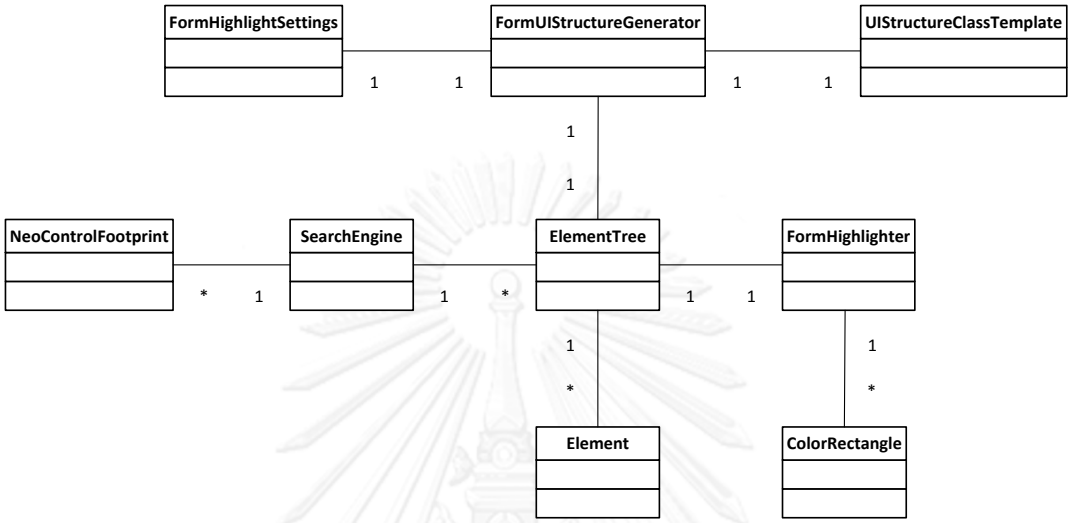


รูปที่ 3.18 คลาส NeoCheckBox

ทั้งนี้ยังมีคลาสต่างๆ สำหรับการทำงานของแต่ละยูสเซอร์คอนโทรลอีกเป็นจำนวนมากตามที่ได้แสดงในรูปที่ 3.15 ซึ่งแต่ละคลาสจะใช้สำหรับควบคุมการทำงานของยูสเซอร์คอนโทรลในชนิดต่างๆ โดยมีฟังก์ชันการทำงานที่เกี่ยวข้องกับยูสเซอร์คอนโทรลนั้นๆ ในที่นี้จะขออธิบายเพียง 3 คลาส ได้แก่ คลาส NeoControlBase คลาส NeoButton และคลาส NeoCheckBox ดังที่อธิบายไปข้างต้น

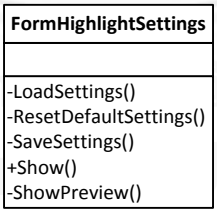
3.2.3.2 ส่วนสร้างโครงสร้างยูไอ

แผนภาพคลาสที่เกี่ยวข้องกับส่วนสร้างโครงสร้างยูไอแสดงได้ดังรูปที่ 3.19



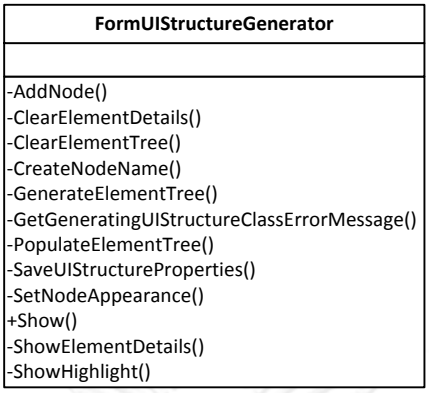
รูปที่ 3.19 แผนภาพคลาสที่เกี่ยวข้องกับส่วนสร้างโครงสร้างยูไอ

1) คลาส FormHighlightSettings คือ คลาสที่ใช้สำหรับแสดงหน้าต่างสำหรับตั้งค่าการแสดงสีไฮไลท์บนยูสเซอร์คอนโทรลที่ต้องการ เช่น สี และขนาดความหนาของเส้นไฮไลท์ เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.20



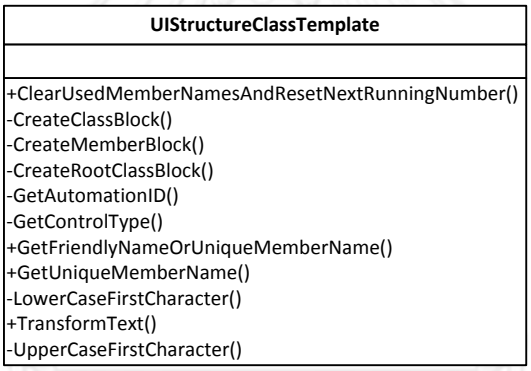
รูปที่ 3.20 คลาส FormHighlightSettings

2) คลาส FormUIStructureGenerator คือ คลาสที่ใช้สำหรับแสดงหน้าต่างสำหรับจัดการการสร้างโครงสร้างยูไอ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.21



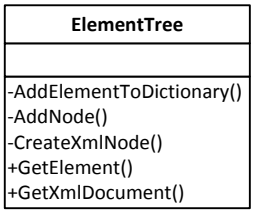
รูปที่ 3.21 คลาส FormUIStructureGenerator

3) คลาส UIStructureClassTemplate คือ คลาสแม่แบบที่ใช้สำหรับสร้างโครงสร้างยูไอ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.22



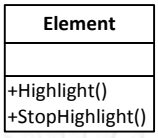
รูปที่ 3.22 คลาส UIStructureClassTemplate

4) คลาส ElementTree คือ คลาสที่ใช้สำหรับสร้างโครงสร้างของแต่ละยูสเซอร์คอนโทรล ซึ่งอ็อบเจ็กต์แต่ละอันนั้นเรียกว่าอิลิเมนต์ (Element) เมื่อจัดเตรียมโครงสร้างของอิลิเมนต์ต่างๆ เสร็จเรียบร้อยแล้วจะได้อยู่ในชนิดเอ็กซ์เอ็มแอลด็อคคิวเมนต์ (XmlDocument) เพื่อนำไปใช้ในการแสดงบนหน้าจอให้นักทดสอบใช้งานต่อไป รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.23



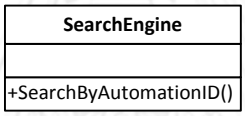
รูปที่ 3.23 คลาส ElementTree

5) คลาส Element คือ คลาสที่ใช้สำหรับแทนแต่ละอิลิเมนต์หรืออ็อบเจ็กต์บนหน้าจอ ซึ่งแต่ละยูสเซอร์คอนโทรลอาจประกอบไปด้วยหลายๆ อิลิเมนต์ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.24



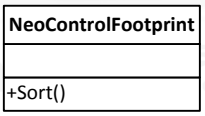
รูปที่ 3.24 คลาส Element

6) คลาส SearchEngine คือ คลาสที่ใช้สำหรับช่วยค้นหายูสเซอร์คอนโทรลที่ต้องการ โดยทำการค้นหาจากค่าประจำตัวของยูสเซอร์คอนโทรลหรือออโตเมชันไอดี รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.25



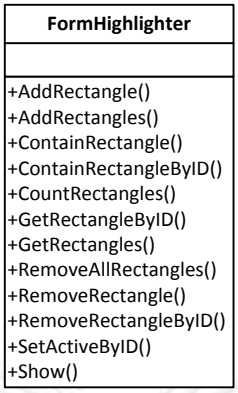
รูปที่ 3.25 คลาส SearchEngine

7) คลาส NeoControlFootprint คือ คลาสที่ใช้สำหรับเก็บข้อมูลของแต่ละยูสเซอร์คอนโทรลบนหน้าจอ โดยจะจัดเก็บตำแหน่งพิกัดและขนาด เพื่อนำไปใช้เป็นข้อมูลในการสร้างกรณีทดสอบในภายหลัง รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.26



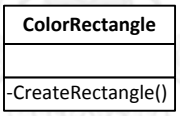
รูปที่ 3.26 คลาส NeoControlFootprint

8) คลาส FormHighlighter คือ คลาสที่ใช้สำหรับแสดงไฮไลท์ตัวยูสเซอร์คอนโทรลที่ต้องการ โดยมีลักษณะเป็นหน้าต่างที่มีพื้นหลังเป็นสีโปร่งแสง จะมีเพียงกรอบสี่เหลี่ยมที่มีขนาดเดียวกับยูสเซอร์คอนโทรลและแสดงอยู่ตรงตำแหน่งยูสเซอร์คอนโทรลที่ถูกระบุไว้เท่านั้นที่มีสี จึงมองเห็นเหมือนมีการไฮไลท์ที่ยูสเซอร์คอนโทรลที่ต้องการ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.27



รูปที่ 3.27 คลาส FormHighlighter

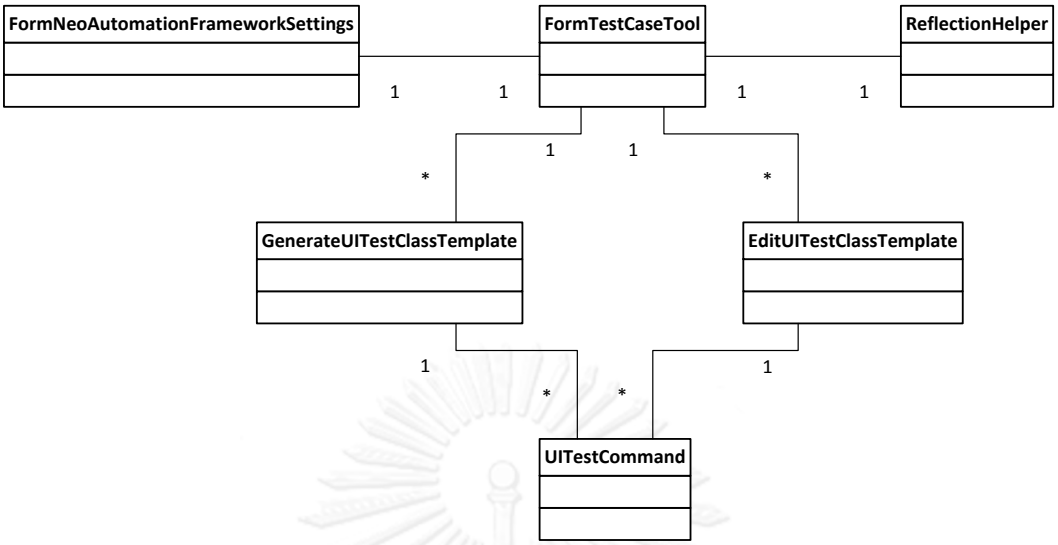
9) คลาส ColorRectangle คือ คลาสที่ใช้สำหรับเก็บข้อมูลที่ใช้สำหรับการแสดงไฮไลท์ยูสเซอร์คอนโทรลเป็นกรอบสี่เหลี่ยม ตัวอย่างข้อมูลที่เก็บ เช่น ความกว้าง ความสูง สี และความหนาของกรอบสี่เหลี่ยม เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.28



รูปที่ 3.28 คลาส ColorRectangle

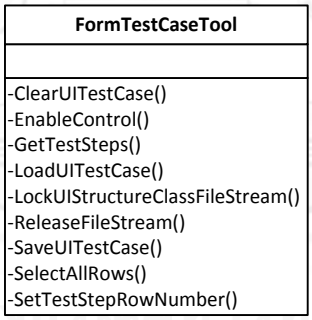
3.2.3.3 ส่วนสร้างกรณีทดสอบและแก้ไขกรณีทดสอบ

แผนภาพคลาสที่เกี่ยวข้องกับส่วนสร้างกรณีทดสอบและแก้ไขกรณีทดสอบแสดงได้ดังรูปที่ 3.29



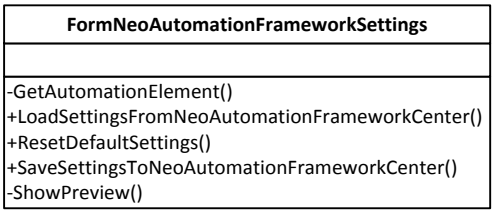
รูปที่ 3.29 แผนภาพคลาสที่เกี่ยวข้องกับส่วนสร้างกรณีทดสอบและแก้ไขกรณีทดสอบ

1) คลาส FormTestCaseTool คือคลาสที่ใช้สำหรับแสดงหน้าต่างสำหรับการจัดการเกี่ยวกับกรณีทดสอบ ซึ่งประกอบไปด้วยแท็บ (Tab) ย่อยสำหรับการสร้างกรณีทดสอบ และการแก้ไขกรณีทดสอบ อยู่ภายในหน้าต่างนี้ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.30



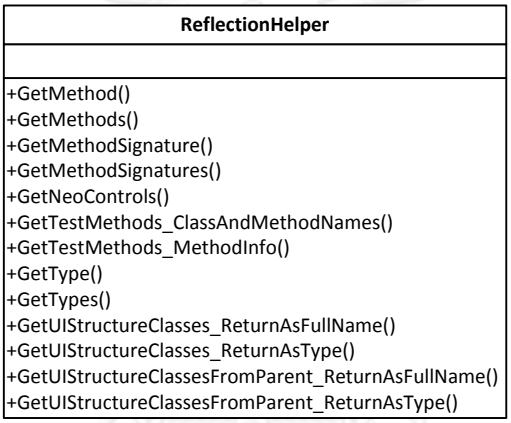
รูปที่ 3.30 คลาส FormTestCaseTool

2) คลาส FormNeoAutomationFrameworkSettings คือคลาสที่ใช้สำหรับแสดงหน้าต่างสำหรับตั้งค่าเกี่ยวกับการทำการทดสอบแบบอัตโนมัติ เพื่อบันทึกลงไปในแต่ละกรณีทดสอบที่จะถูกสร้างขึ้นหรือได้รับการแก้ไข รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.31



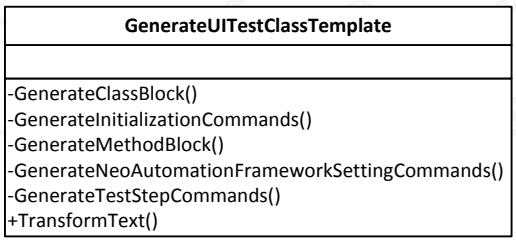
รูปที่ 3.31 คลาส FormNeoAutomationFrameworkSettings

3) คลาส ReflectionHelper คือคลาสที่ใช้สำหรับอ่านข้อมูลจากไฟล์แอสเซมบลีโดยตรง ว่ามีคลาสและเมธอดอะไรบ้าง รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.32



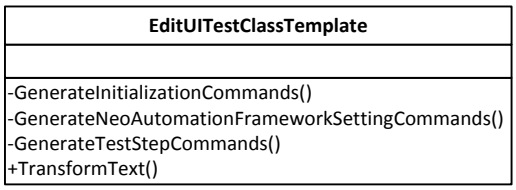
รูปที่ 3.32 คลาส ReflectionHelper

4) คลาส GenerateUITestClassTemplate คือคลาสที่ใช้สำหรับสร้างกรณีทดสอบในรูปแบบภาษาซีชาร์ป จากการสร้างโดยอัตโนมัติโดยใช้เครื่องมือสร้างกรณีทดสอบ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.33



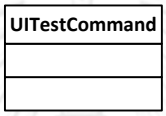
รูปที่ 3.33 คลาส GenerateUITestClassTemplate

5) คลาส EditUITestClassTemplate คือคลาสที่ใช้สำหรับสร้างกรณีทดสอบในรูปแบบภาษาซีชาร์ป จากการแก้ไขผ่านเครื่องมือแก้ไขกรณีทดสอบ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.34



รูปที่ 3.34 คลาส EditUITestClassTemplate

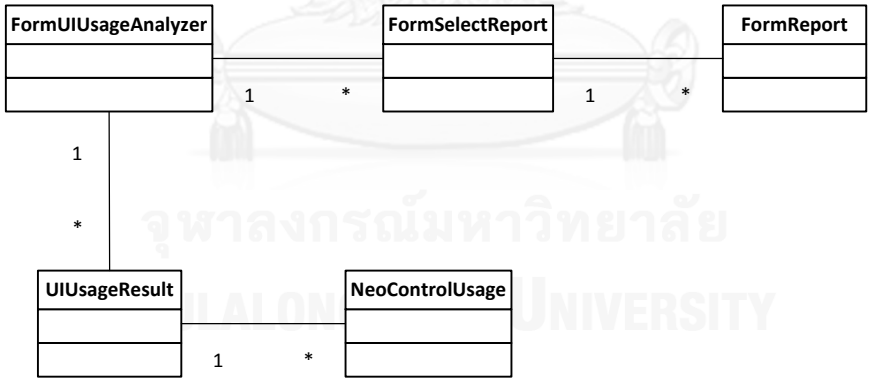
6) คลาส UITestCommand คือคลาสที่ใช้สำหรับเก็บข้อมูลแต่ละคำสั่งการทำงานภายในกรณีทดสอบ เช่น หมายเลขบรรทัด คำสั่ง ประเภทของคำสั่ง คำอธิบาย เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.35



รูปที่ 3.35 คลาส UITestCommand

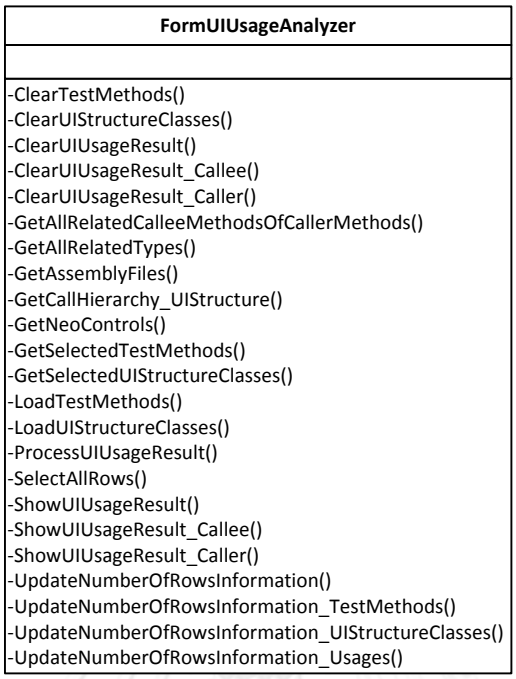
3.2.3.4 ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

แผนภาพคลาสที่เกี่ยวข้องกับส่วนส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบแสดงได้ดังรูปที่ 3.36



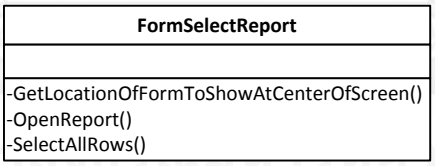
รูปที่ 3.36 แผนภาพคลาสที่เกี่ยวข้องกับส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

1) คลาส FormUIUsageAnalyzer คือคลาสที่ใช้สำหรับแสดงหน้าต่างสำหรับเลือกโครงสร้างยูไอและกรณีทดสอบที่ต้องการเพื่อนำไปวิเคราะห์หาการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.37



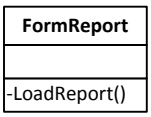
รูปที่ 3.37 คลาส FormUIUsageAnalyzer

2) คลาส FormSelectReport คือคลาสที่ใช้สำหรับแสดงหน้าต่างสำหรับเลือกโครงสร้างยูไอที่ต้องการเพื่อสร้างรายงาน รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.38



รูปที่ 3.38 คลาส FormSelectReport

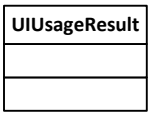
3) คลาส FormReport คือคลาสที่ใช้สำหรับแสดงหน้าต่างรายงานของโครงสร้างยูไอที่ถูกเลือก รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.39



รูปที่ 3.39 คลาส FormReport

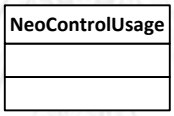
4) คลาส UIUsageResult คือคลาสที่ใช้สำหรับเก็บข้อมูลการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบของแต่ละโครงสร้างยูไอ ข้อมูลที่เก็บมีดังนี้ แอสเซมบลีของโครงสร้างยูไอ เนมสเป

ซของโครงสร้างยูไอ จำนวนยูสเซอร์คอนโทรลทั้งหมด จำนวนยูสเซอร์คอนโทรลที่ถูกเรียกใช้ จำนวนยูสเซอร์คอนโทรลที่ไม่ถูกเรียกใช้ เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.40



รูปที่ 3.40 คลาส UIUsageResult

5) คลาส NeoControlUsage คือคลาสที่ใช้สำหรับเก็บข้อมูลการเรียกใช้งานของแต่ละยูสเซอร์คอนโทรล ข้อมูลที่เก็บมีดังนี้ ชื่อยูสเซอร์คอนโทรล ประเภทของยูสเซอร์คอนโทรล จำนวนครั้งที่ถูกเรียกใช้งาน รายชื่อเมธอดที่มีการเรียกใช้งานยูสเซอร์คอนโทรล เป็นต้น รายละเอียดของคลาสแสดงได้ดังรูปที่ 3.41



รูปที่ 3.41 คลาส NeoControlUsage

บทที่ 4
การพัฒนาเครื่องมือ

4.1 สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือ

สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือมีรายละเอียดดังนี้

4.1.1 ฮาร์ดแวร์ (Hardware)

- คอมพิวเตอร์ขนาดสมุดบันทึก (Notebook Computer) หน่วยประมวลผลอินเทลคอร์ไอเซเว่น 2.67 กิกะเฮิร์ต (Intel Core i7 2.67 GHz)
- หน่วยความจำ (RAM) 8 กิกะไบต์ (8 GB)
- ฮาร์ดดิสก์ (Hard Disk) 500 กิกะไบต์ (500 GB)

4.1.2 ซอฟต์แวร์ (Software)

- ระบบปฏิบัติการไมโครซอฟท์วินโดวส์เซเว่น โปรเฟสชันนอล เซอร์วิสแพ็ค 1 (Microsoft Windows 7 Professional Service Pack 1)
- ไมโครซอฟท์วิชวลสตูดิโอ 2010 พรีเมียม (Microsoft Visual Studio 2010 Premium)
- ไมโครซอฟท์ดอทเน็ทเฟรมเวิร์ก เวอร์ชัน 4.0 (Microsoft .NET Framework 4.0)

4.2 โครงสร้างส่วนต่อประสานกับผู้ใช้เครื่องมือ

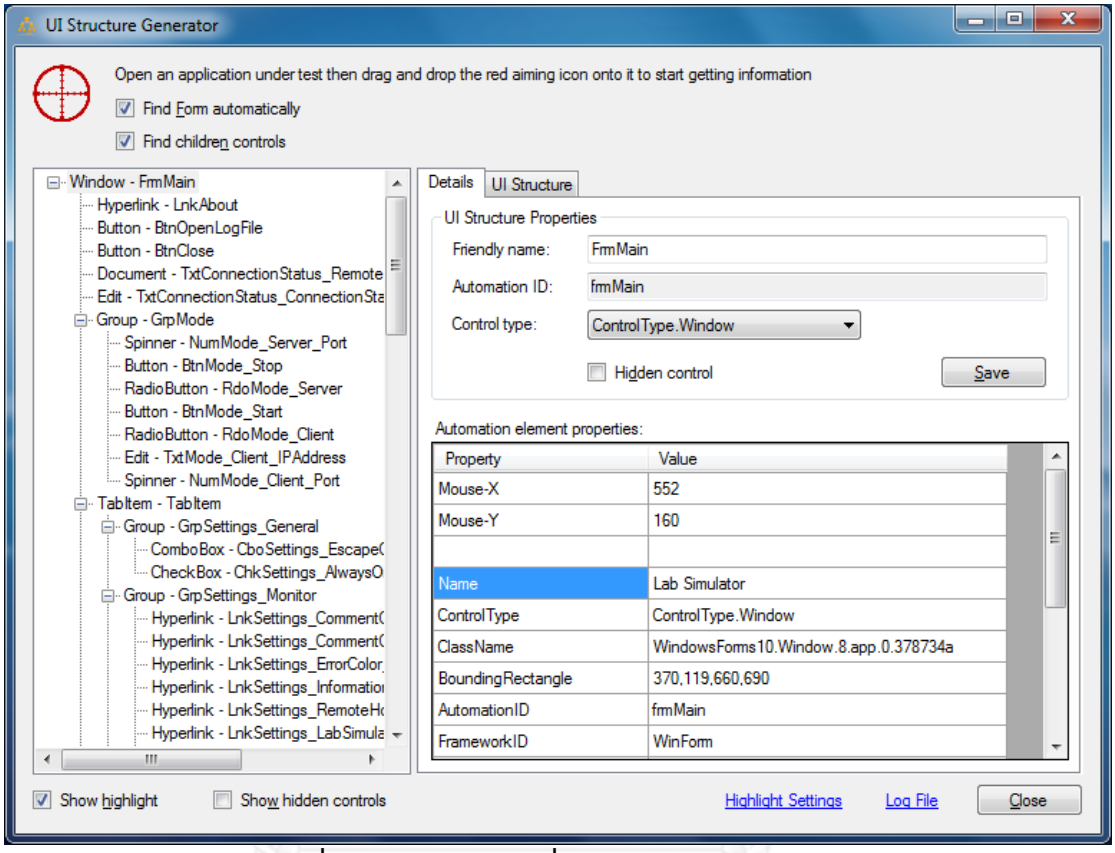
โครงสร้างส่วนต่อประสานกับผู้ใช้เครื่องมือ สามารถแสดงโดยแบ่งตามเครื่องมือได้ดังนี้

4.2.1 เครื่องมือสร้างโครงสร้างยูไอ

หน้าจอหลักของเครื่องมือสร้างโครงสร้างยูไอแสดงดังรูปที่ 4.1 นักทดสอบสามารถระบุหน้าจอของซอฟต์แวร์ที่ต้องการทดสอบเพื่อสร้างโครงสร้างยูไอ จากนั้นเครื่องมือจะทำการประมวลผลและจัดเตรียมข้อมูลสำหรับสร้างโครงสร้างยูไอ โดยแสดงอยู่ในลักษณะโครงสร้างต้นไม้ (Tree View) ยูสเซอร์คอนโทรลที่สามารถเป็นจุดเริ่มต้นของการสร้างโครงสร้างยูไอจะต้องมีชนิด Form ซึ่งในขั้นตอนนี้นักทดสอบสามารถเลือกให้เครื่องมือค้นหาคอนโทรลที่มีชนิด Form และค้นหายูสเซอร์คอนโทรลย่อยต่างๆ โดยอัตโนมัติโดยการเลือกเช็คบ็อกซ์ Find Form automatically และ Find children controls ตามลำดับ

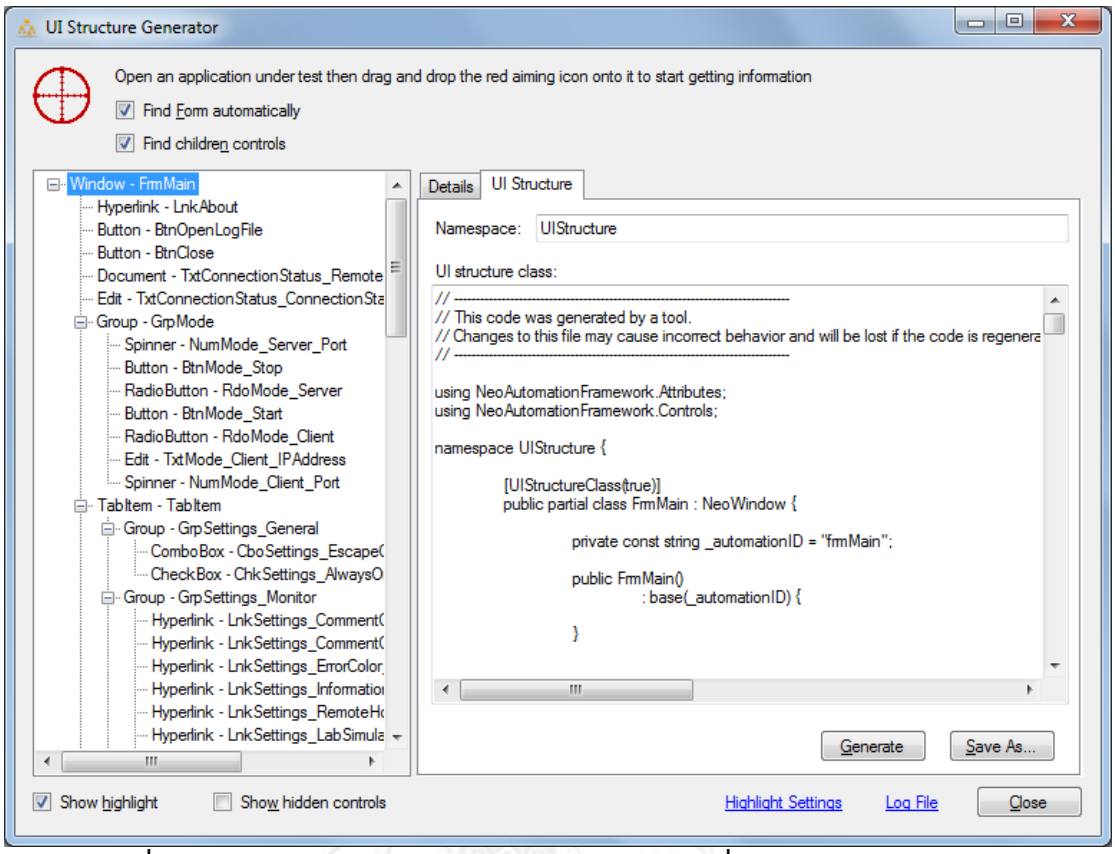
นักทดสอบสามารถเลือกยูสเซอร์คอนโทรลที่แสดงอยู่ใน Tree View ทางด้านซ้ายและดูรายละเอียดของยูสเซอร์นั้นทางด้านขวาในแท็บชื่อ Details ซึ่งจะแสดงรายละเอียดต่างๆ เกี่ยวกับยูสเซอร์คอนโทรลตัวนั้น เช่น ชนิดของยูสเซอร์คอนโทรล ขนาด ตำแหน่ง ออโตเมชันไอดี เป็นต้น นักทดสอบสามารถกำหนดชื่อของยูสเซอร์คอนโทรลที่จะถูกสร้างในโครงสร้างยูไอได้เพื่อความสะดวกใน

การใช้งาน เนื่องจากในบางกรณี ชื่อของยูสเซอร์คอนโทรลที่ตรวจสอบมาได้นั้นอาจไม่ชัดเจนหรือมีความกำกวม ทำให้ยากต่อการอ้างอิงเพื่อใช้งานในภายหลัง



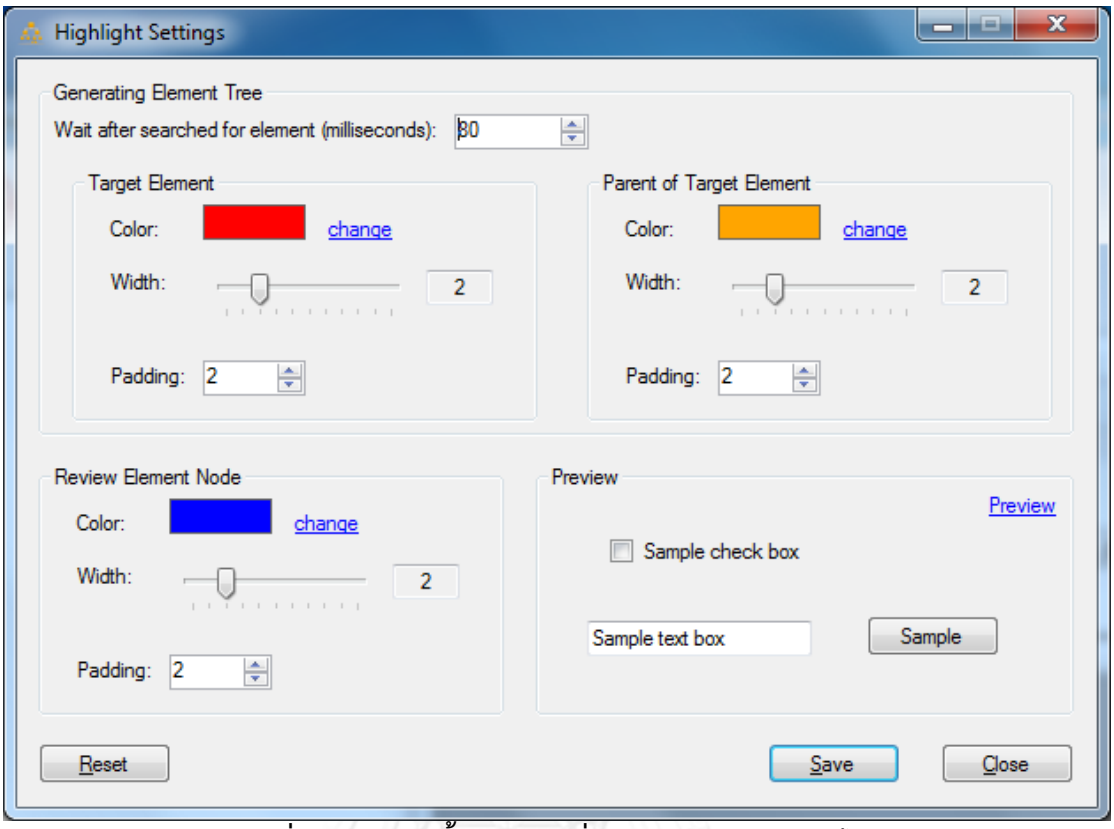
รูปที่ 4.1 หน้าจอของเครื่องมือสร้างโครงสร้างยูไอ

ภายในแท็บชื่อ UI Structure นักทดสอบสามารถกำหนดชื่อของเนมสเปซที่ต้องการและสร้างโครงสร้างยูไอได้โดยการกดปุ่ม Generate ซึ่งเครื่องมือจะทำการสร้างคลาสของโครงสร้างยูไอขึ้นมาและแสดงให้ดูบนหน้าจอ นักทดสอบสามารถบันทึกคลาสดังกล่าวไปยังไฟล์ที่ต้องการได้โดยการกดปุ่ม Save As ดังแสดงในรูปที่ 4.2



รูปที่ 4.2 ตัวอย่างโครงสร้างยูไอแสดงในหน้าจอของเครื่องมือสร้างโครงสร้างยูไอ

นักทดสอบสามารถตั้งค่าให้แก่เครื่องมือสร้างโครงสร้างยูไอได้โดยการกดปุ่มไฮเปอร์ลิงค์ด้านล่างชื่อ Highlight Settings ซึ่งจะมีหน้าจอตั้งค่าแสดงขึ้นมาดังรูปที่ 4.3 นักทดสอบสามารถตั้งค่าการหน่วงเวลาสำหรับแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลบนหน้าจอได้ สามารถตั้งค่าสี ความหน้าของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล ของยูสเซอร์คอนโทรลเป้าหมายได้ ซึ่งการตั้งค่าต่างๆ จะมีตัวอย่างแสดงให้เห็นบนหน้าจอด้วย นอกจากนี้นักทดสอบสามารถดูรายละเอียดการทำงานของการสร้างโครงสร้างยูไอได้โดยกดปุ่มไฮเปอร์ลิงค์ Log File



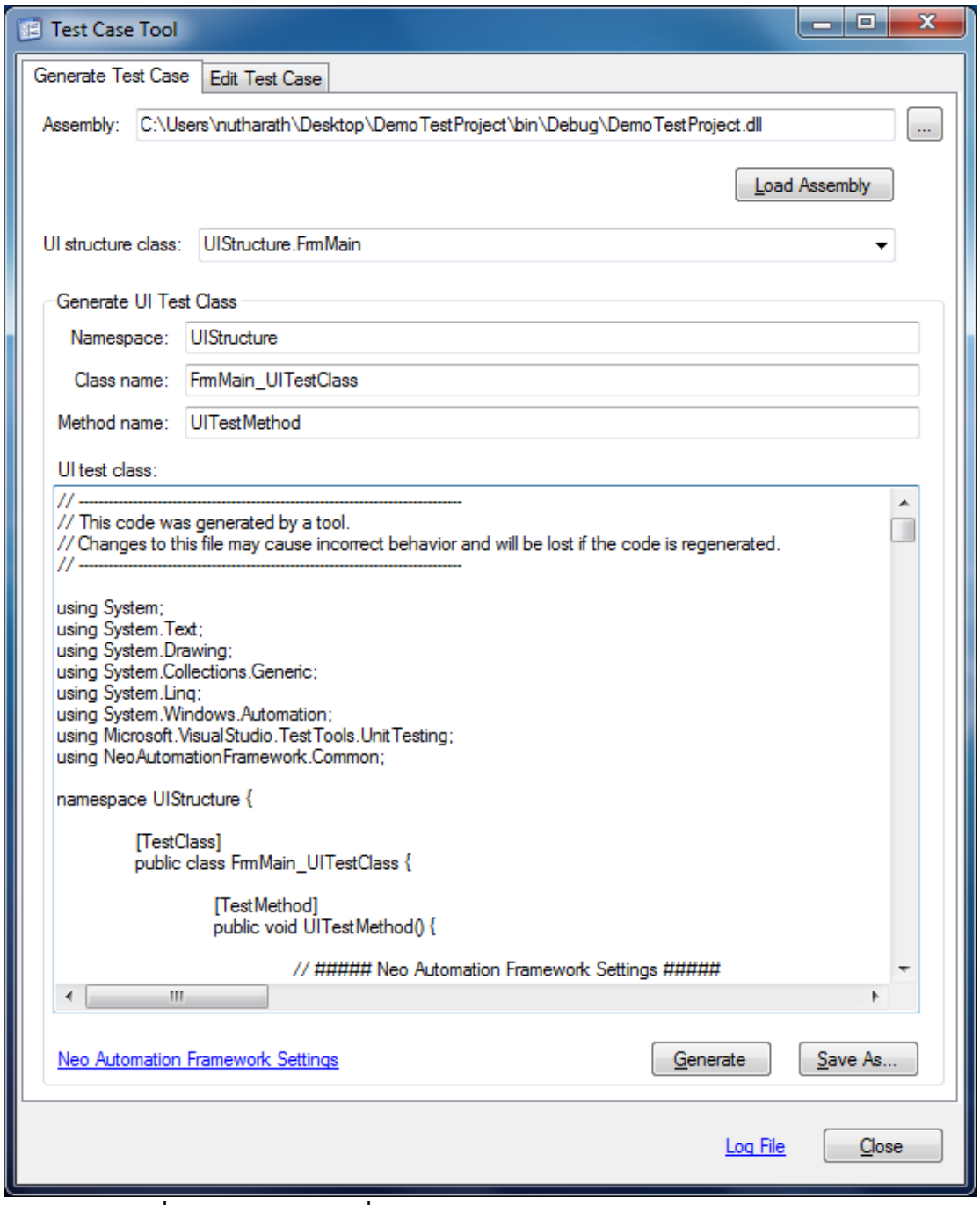
รูปที่ 4.3 หน้าจอตั้งค่าของเครื่องมือสร้างโครงสร้างยูไอ

4.2.2 เครื่องมือจัดการกรณีทดสอบ

หน้าจอหลักของเครื่องมือจัดการกรณีทดสอบประกอบไปด้วยส่วนประกอบ 2 ส่วน ได้แก่ ส่วนสร้างกรณีทดสอบ แสดงอยู่ในแท็บ Generate Test Case และส่วนแก้ไขกรณีทดสอบ แสดงอยู่ในแท็บ Edit Test Case

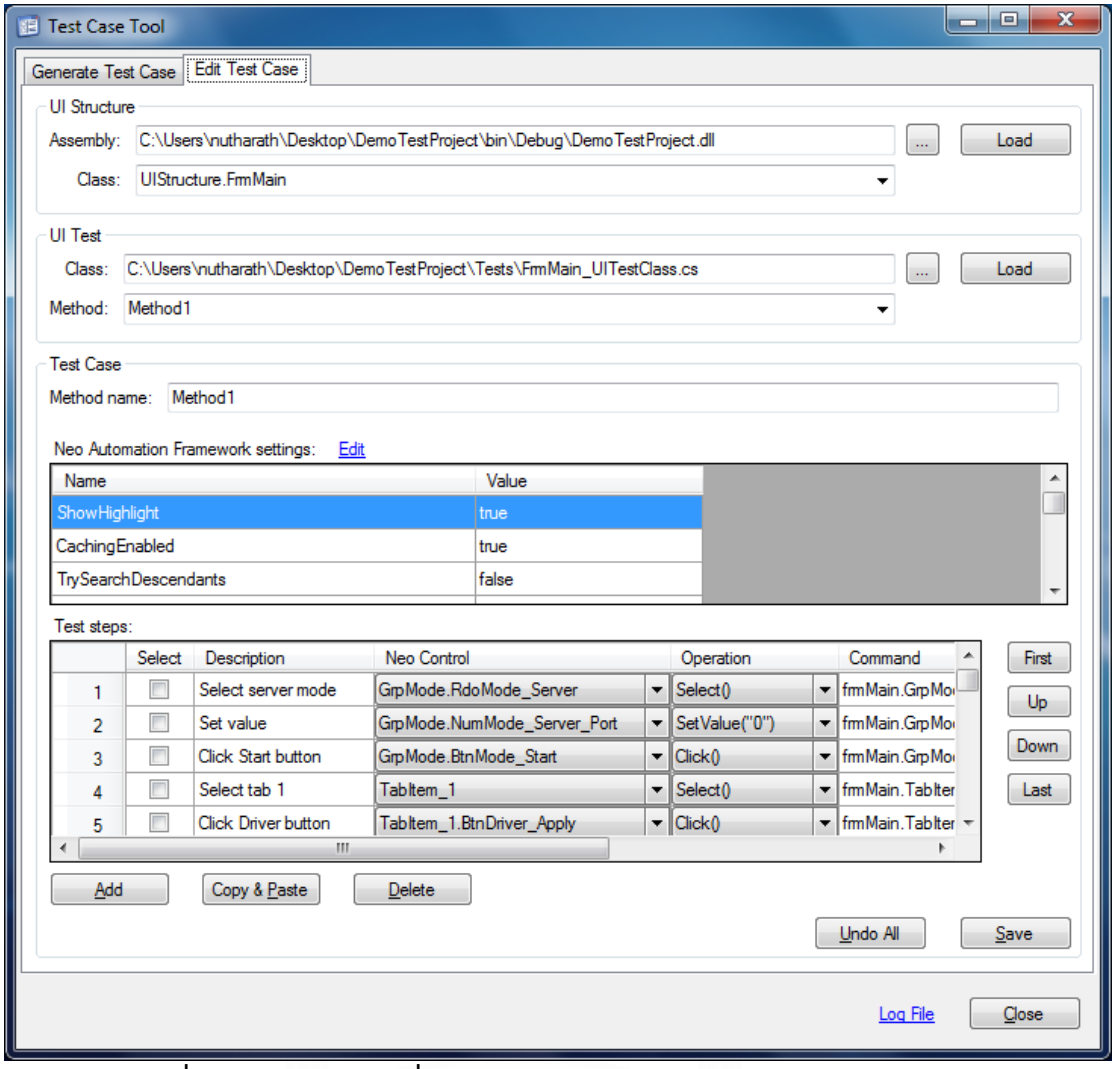
ในการสร้างกรณีทดสอบ นักทดสอบเลือกไฟล์แอสเซมบลีของโครงสร้างยูไอที่ต้องการ จากนั้นกดปุ่ม Load Assembly เครื่องมือจะทำการอ่านรายชื่อโครงสร้างยูไอและแสดงในคอมโบบ็อกซ์ นักทดสอบสามารถเลือกโครงสร้างยูไอที่ต้องการสร้างกรณีทดสอบและกำหนดชื่อเนมสเปซ คลาส และเมธอด ที่ต้องการสร้าง จากนั้นกดปุ่ม Generate เครื่องมือจะทำการสร้างคลาสสำหรับการทดสอบและแสดงบนหน้าจอ ซึ่งนักทดสอบสามารถบันทึกคลาสดังกล่าวไปยังไฟล์ที่ต้องการได้โดยการกดปุ่ม Save As ดังแสดงในรูปที่ 4.4

นักทดสอบสามารถตั้งค่าการทำงานสำหรับกรณีทดสอบได้โดยการกดปุ่มไฮเปอร์ลิงค์ด้านล่างชื่อ Neo Automation Framework Settings ซึ่งจะมีหน้าจอตั้งค่าแสดงขึ้นมาดังรูปที่ 4.6 นอกจากนี้นักทดสอบสามารถดูรายละเอียดการทำงานของการสร้างกรณีทดสอบได้โดยกดปุ่มไฮเปอร์ลิงค์ Log File



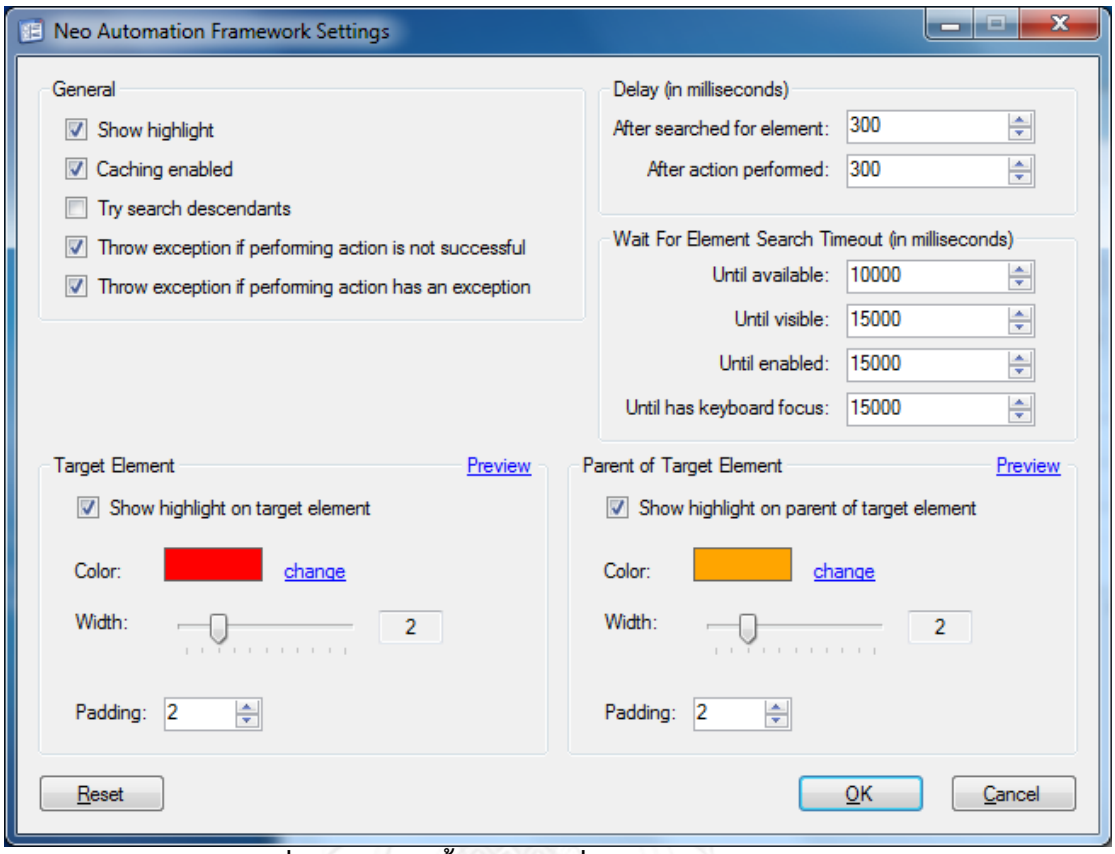
รูปที่ 4.4 หน้าจอของเครื่องมือจัดการกรณีทดสอบ ส่วนสร้างกรณีทดสอบ

นักทดสอบสามารถแก้ไขกรณีทดสอบได้โดยเลือกแท็บ Edit Test Case จากนั้นเลือกไฟล์แอสเซมบลีของโครงสร้างยูไอ และไฟล์คลาสของกรณีทดสอบที่ต้องการแก้ไข จากนั้นกดปุ่ม Load เครื่องมือจะทำการอ่านรายชื่อโครงสร้างยูไอและรายชื่อคลาสตามลำดับ เครื่องมือจะแสดงชื่อเมธอดและรายละเอียดการทดสอบภายในเมธอดในตาราง นักทดสอบสามารถแก้ไขการตั้งค่าของกรณีทดสอบ และลำดับการทำงานของกรณีทดสอบได้อย่างสะดวก เมื่อแก้ไขกรณีทดสอบเสร็จแล้ว สามารถบันทึกกลับไปยังไฟล์คลาสของกรณีทดสอบได้โดยการกดปุ่ม Save ดังแสดงในรูปที่ 4.5



รูปที่ 4.5 หน้าจอของเครื่องมือจัดการกรณีทดสอบ ส่วนแก้ไขกรณีทดสอบ

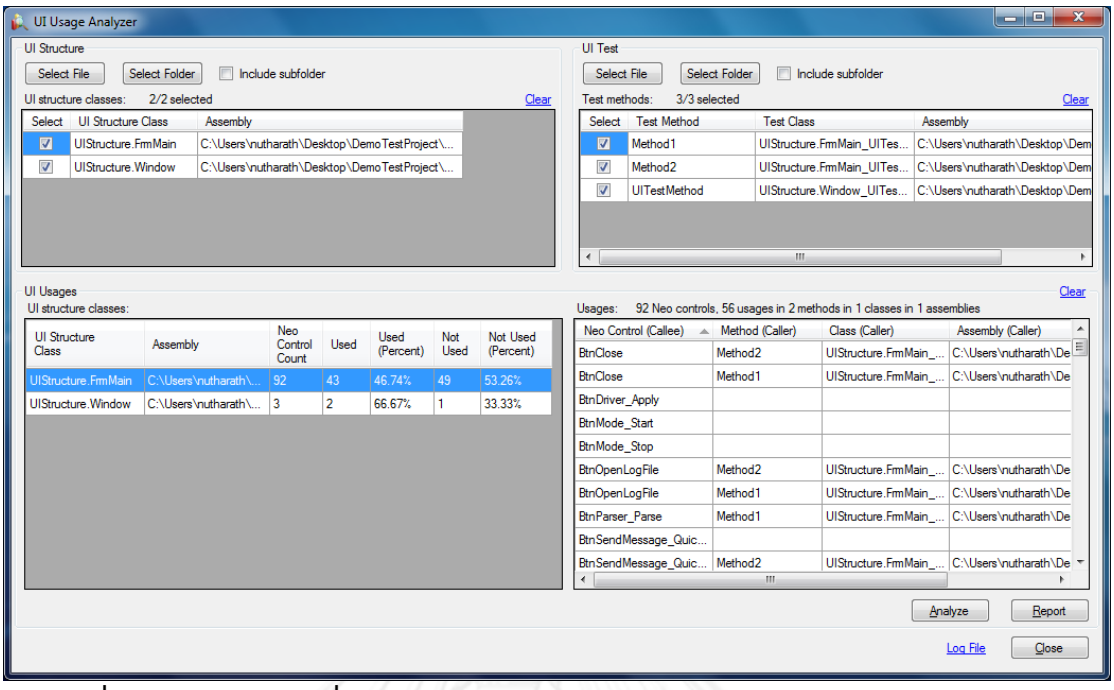
นักทดสอบสามารถตั้งค่าการทดสอบได้ โดยการกดปุ่มไฮเปอร์ลิงค์ Edit หน้าจอสำหรับตั้งค่าการทดสอบแสดงได้ดังรูปที่ 4.6 นักทดสอบสามารถตั้งค่าต่างๆ ได้ เช่น ต้องการแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลหรือไม่ วิธีการค้นหายูสเซอร์คอนโทรล การแสดงความผิดพลาดหากการกระทำกับยูสเซอร์คอนโทรลมีปัญหา การหน่วงเวลาสำหรับแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรล การหน่วงเวลาหลังจากกระทำกับยูสเซอร์คอนโทรล การหน่วงเวลาสำหรับค้นหายูสเซอร์คอนโทรลที่ต้องการ การตั้งค่าสี ความหนาของเส้นสี และระยะเว้นตรงขอบยูสเซอร์คอนโทรลของยูสเซอร์คอนโทรลเป้าหมาย เป็นต้น นอกจากนี้นักทดสอบสามารถดูรายละเอียดการทำงานของการแก้ไขกรณีทดสอบได้โดยกดปุ่มไฮเปอร์ลิงค์ Log File



รูปที่ 4.6 หน้าจอตั้งค่าของเครื่องมือจัดการกรณีทดสอบ

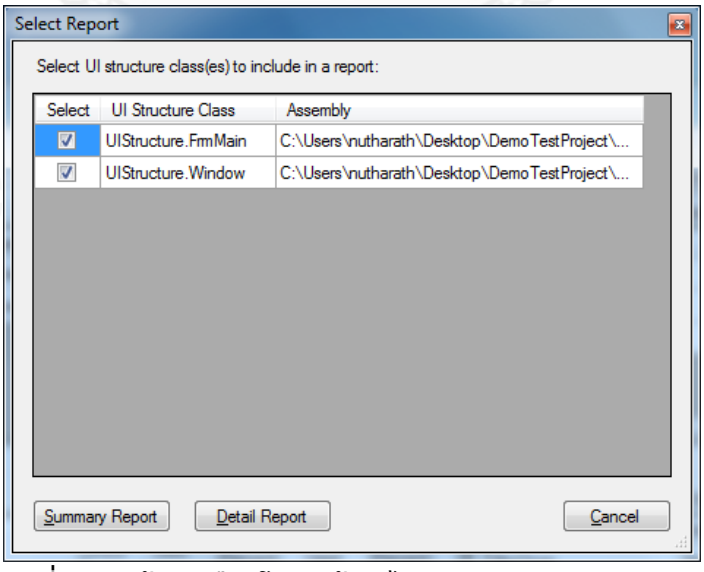
4.2.3 เครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

หน้าจอหลักของเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบประกอบไปด้วย 3 ส่วนประกอบหลัก ได้แก่ ส่วนกำหนดไฟล์แอสเซมบลีของโครงสร้างยูไอ ส่วนกำหนดไฟล์แอสเซมบลีของกรณีทดสอบ และส่วนแสดงผลการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในการทดสอบ ซึ่งประกอบไปด้วยข้อมูลการวิเคราะห์โดยรวมและข้อมูลแบบละเอียด ดังแสดงในรูปที่ 4.7



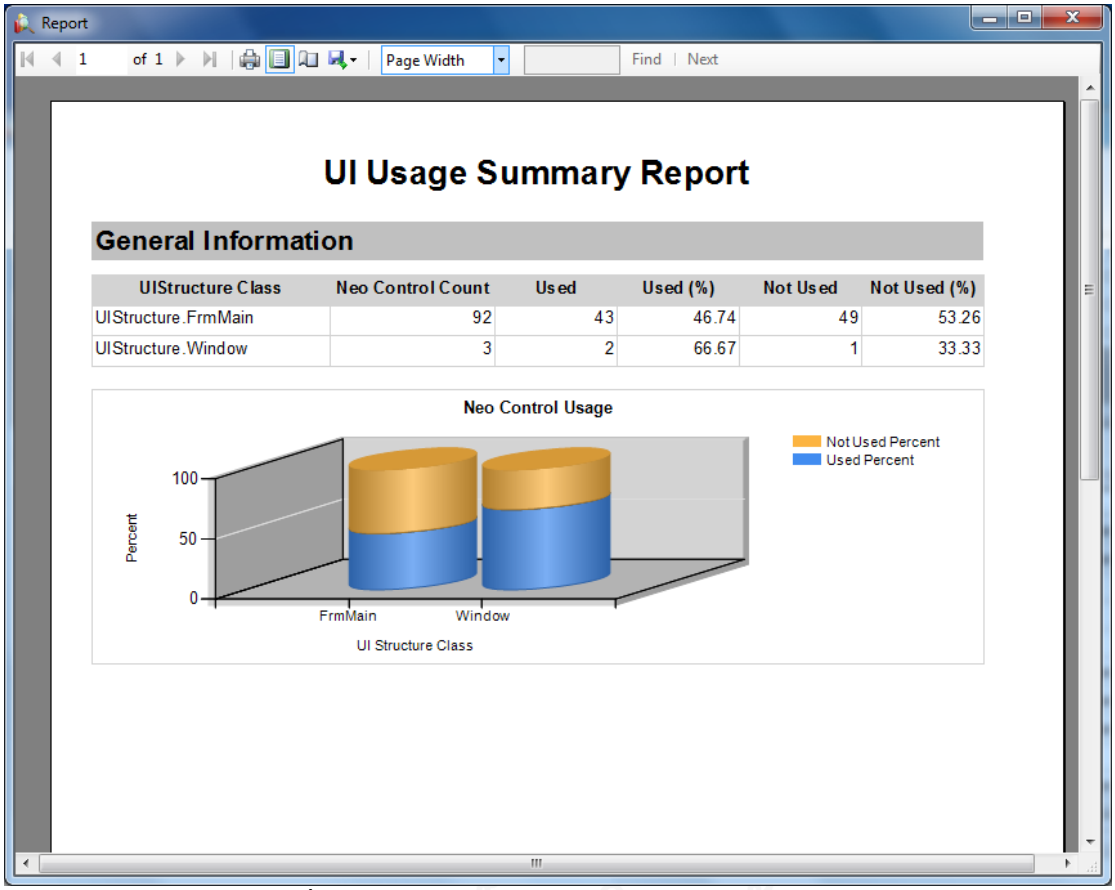
รูปที่ 4.7 หน้าจอของเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

เมื่อนักทดสอบได้ทำการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบแล้ว สามารถพิมพ์ผลลัพธ์ออกไปในรูปแบบรายงานการวิเคราะห์ได้โดยการกดปุ่ม Report เครื่องมือจะแสดงหน้าจอสำหรับเลือกโครงสร้างยูไอและประเภทของรายงานที่ต้องการ ดังแสดงในรูปที่ 4.8 นอกจากนี้นักทดสอบสามารถดูรายละเอียดการทำงานของการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบได้โดยกดปุ่มไฮเปอร์ลิงค์ Log File



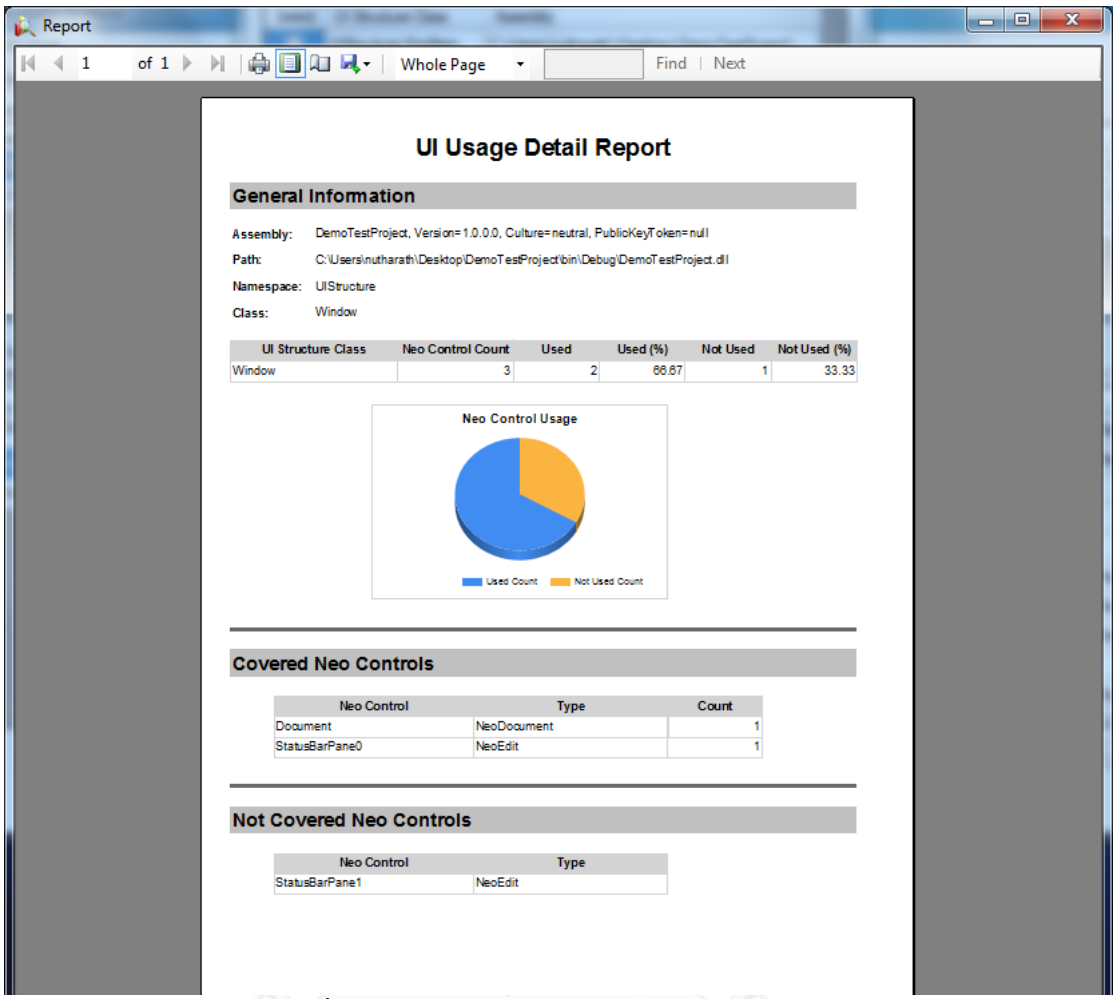
รูปที่ 4.8 หน้าจอเลือกโครงสร้างยูไอและประเภทของรายงาน

รายงานแบบ Summary Report ประกอบไปด้วย ชื่อคลาสของโครงสร้างยูไอ จำนวนยูสเซอร์คอนโทรล จำนวนยูสเซอร์คอนโทรลที่ถูกใช้งาน เปอร์เซ็นต์ของยูสเซอร์คอนโทรลที่ถูกใช้งาน จำนวนยูสเซอร์คอนโทรลที่ไม่ถูกใช้งาน เปอร์เซ็นต์ของยูสเซอร์คอนโทรลที่ไม่ถูกใช้งาน และแผนภูมิแท่ง แสดงได้ดังรูปที่ 4.9



รูปที่ 4.9 ตัวอย่างรายงานแบบ Summary Report

รายงานแบบ Detail Report ประกอบไปด้วย ข้อมูลเกี่ยวกับโครงสร้างยูไอ ผลการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในโครงสร้างยูไอ รายชื่อยูสเซอร์คอนโทรลที่ถูกใช้งานและไม่ถูกใช้งาน เป็นต้น แสดงได้ดังรูปที่ 4.10



รูปที่ 4.10 ตัวอย่างรายงานแบบ Detail Report

บทที่ 5
การทดสอบเครื่องมือ

5.1 สภาพแวดล้อมที่ใช้ในการทดสอบ

สภาพแวดล้อมที่ใช้ในการทดสอบเครื่องมือมีรายละเอียดดังนี้

5.1.1 ฮาร์ดแวร์

- คอมพิวเตอร์ขนาดสมุดบันทึก หน่วยประมวลผลอินเทลคอร์ไอเซเว่น 2.67 กิกะเฮิร์ต
- หน่วยความจำ 8 กิกะไบต์
- ฮาร์ดดิสก์ 500 กิกะไบต์

5.1.2 ซอฟต์แวร์

- ระบบปฏิบัติการไมโครซอฟท์วินโดวส์เซเว่น โปรเฟสชันนอล เซอร์วิสแพ็ค 1
- ไมโครซอฟท์วิชวลสตูดิโอ 2010 พรีเมียม
- ไมโครซอฟท์ดอทเน็ทเฟรมเวิร์ก เวอร์ชัน 4.0

5.2 ขั้นตอนการทดสอบเครื่องมือ

ขั้นตอนการทดสอบเครื่องมือนี้มีไว้เพื่อตรวจสอบความถูกต้องของการทำงานของเครื่องมือที่ได้ออกแบบและพัฒนาขึ้นมา การทดสอบนี้แบ่งออกเป็น 4 ส่วน คือ การทดสอบกระบวนการสร้างโครงสร้างยูไอ การทดสอบกรณีทดสอบที่ได้จากเครื่องมือจัดการกรณีทดสอบ การทดสอบความถูกต้องของเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ และการทดสอบการทำงานของกรณีทดสอบ

5.3 ระบบที่ใช้ในการทดสอบ

ระบบที่ใช้ในการทดสอบความถูกต้องของเครื่องมือนั้น ผู้วิจัยได้เลือกระบบที่มีการใช้งานจริงจำนวน 2 ระบบมาทำการทดสอบ โดยระบบแรกนั้นเป็นโปรแกรมที่มากับระบบปฏิบัติการวินโดวส์และเป็นที่รู้จักโดยทั่วไป ส่วนอีกระบบหนึ่งนั้นเป็นระบบสำหรับทำงานเฉพาะด้านเกี่ยวกับการสื่อสารผ่านทางเครือข่าย ตัวอย่างหน้าจอของระบบที่ใช้ในการทดสอบสามารถดูได้จากภาคผนวก ข.

5.3.1 ระบบที่ 1

เป็นโปรแกรมเครื่องคิดเลข ในหน้าจอประกอบไปด้วยยูสเซอร์คอนโทรลเป็นจำนวนมากเพื่อรับคำสั่งในการคำนวณ และแสดงผลลัพธ์ของการคำนวณบนหน้าจอ

5.3.2 ระบบที่ 2

เป็นโปรแกรมสื่อสารผ่านทางเครือข่าย โดยมีฟังก์ชันการทำงานเป็นจำนวนมาก เช่น สามารถทำการรอรับการเชื่อมต่อจากเครือข่ายหรือทำการติดต่อไปยังปลายทางอื่นภายในเครือข่ายได้ เป็นต้น

5.4 ผลการทดสอบ

หลังจากที่ได้ทำการทดสอบเครื่องมือทั้ง 3 เครื่องมือกับระบบสำหรับการทดสอบทั้ง 2 ระบบ ได้ผลการทดสอบดังนี้

5.4.1 ผลการทดสอบกระบวนการสร้างโครงสร้างยูไอ

ผลการทดสอบกระบวนการสร้างโครงสร้างยูไอกับโปรแกรมเครื่องคิดเลขและโปรแกรมสื่อสารผ่านทางเครือข่าย แสดงได้ดังตารางที่ 5.1

ตารางที่ 5.1 ตารางแสดงผลการทดสอบกระบวนการสร้างโครงสร้างยูไอ

การทดสอบ	ระบบที่ 1		ระบบที่ 2	
	คาดหวัง	ผลลัพธ์	คาดหวัง	ผลลัพธ์
จำนวนยูสเซอร์คอนโทรลในโครงสร้างยูไอ	28 คอนโทรล	28 คอนโทรล	92 คอนโทรล	92 คอนโทรล
การนำคลาสโครงสร้างยูไอไปใช้งาน	คอมไพล์ผ่าน	คอมไพล์ผ่าน	คอมไพล์ผ่าน	คอมไพล์ผ่าน

จากผลการทดสอบพบว่าเมื่อนำเอาระบบทดสอบมาทดสอบกับเครื่องมือเพื่อให้เครื่องมือสร้างโครงสร้างยูไอและนำโครงสร้างยูไอที่ได้ในรูปแบบคลาสภาษาซีชาร์ปไปใช้งาน ผลที่ได้คือจำนวนยูสเซอร์คอนโทรลที่ถูกสร้างภายในโครงสร้างยูไอตรงกับจำนวนที่คาดหวัง และคลาสโครงสร้างยูไอสามารถนำไปใช้งานในโปรเจกต์การทดสอบได้อย่างไม่มีปัญหา จึงสรุปได้ว่าเครื่องมือสร้างโครงสร้างยูไอสามารถทำงานได้อย่างถูกต้อง

5.4.2 ผลการทดสอบกรณีทดสอบที่ได้จากเครื่องมือจัดการกรณีทดสอบ

ผลการทดสอบกรณีทดสอบที่ได้จากเครื่องมือจัดการกรณีทดสอบกับโปรแกรมเครื่องคิดเลขและโปรแกรมสื่อสารผ่านทางเครือข่าย แสดงได้ดังตารางที่ 5.2

ตารางที่ 5.2 ตารางแสดงผลการทดสอบกรณีทดสอบที่ได้จากเครื่องมือจัดการกรณีทดสอบ

การทดสอบ	ระบบที่ 1		ระบบที่ 2	
	คาดหวัง	ผลลัพธ์	คาดหวัง	ผลลัพธ์
จำนวนขั้นตอนในกรณีทดสอบ	28 ขั้นตอน	28 ขั้นตอน	92 ขั้นตอน	92 ขั้นตอน
จำนวนขั้นตอนในกรณีทดสอบหลังมีการแก้ไข	7 ขั้นตอน	7 ขั้นตอน	15 ขั้นตอน	15 ขั้นตอน
การนำกรณีทดสอบไปใช้งาน	คอมไพล์ผ่าน	คอมไพล์ผ่าน	คอมไพล์ผ่าน	คอมไพล์ผ่าน

จากผลการทดสอบพบว่าเมื่อนำเอาโครงสร้างยูไอที่ได้จากขั้นตอนก่อนหน้ามาสร้างกรณีทดสอบโดยใช้เครื่องมือจัดการกรณีทดสอบ ผลที่ได้คือสามารถสร้างกรณีทดสอบที่มีขั้นตอนการทำงานครบถ้วนตรงตามที่คาดหวัง และเมื่อทำการแก้ไขกรณีทดสอบให้มีขั้นตอนการทำงานเป็นไปตามที่ต้องการสอดคล้องกับการใช้งานจริงของผู้ใช้งาน กรณีทดสอบที่ได้รับการแก้ไขก็มีขั้นตอนการทำงานถูกต้องตามที่แก้ไข และสามารถนำกรณีทดสอบที่ได้ไปใช้งานในโปรเจกต์การทดสอบได้อย่างไม่มีปัญหา จึงสรุปได้ว่าเครื่องมือจัดการกรณีทดสอบสามารถทำงานได้ถูกต้อง

5.4.3 ผลการทดสอบความถูกต้องของเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

ผลการทดสอบความถูกต้องของเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบกับโปรแกรมเครื่องคิดเลขและโปรแกรมสื่อสารผ่านทางเครือข่าย โดยการสร้างกรณีทดสอบให้มีการใช้งานยูสเซอร์คอนโทรลภายในโครงสร้างยูไอเพียงบางส่วน แสดงได้ดังตารางที่ 5.3

ตารางที่ 5.3 ตารางแสดงผลการทดสอบความถูกต้องของเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ

การทดสอบ	ระบบที่ 1		ระบบที่ 2	
	คาดหวัง	ผลลัพธ์	คาดหวัง	ผลลัพธ์
จำนวนยูสเซอร์คอนโทรลทั้งหมด	28 คอนโทรล	28 คอนโทรล	92 คอนโทรล	92 คอนโทรล
จำนวนยูสเซอร์คอนโทรลที่ถูกใช้งาน	7 คอนโทรล	7 คอนโทรล	15 คอนโทรล	15 คอนโทรล
เปอร์เซ็นต์ของยูสเซอร์คอนโทรลที่ถูกใช้งาน	25%	25%	16%	16%
จำนวนยูสเซอร์คอนโทรลที่ไม่ถูกใช้งาน	21 คอนโทรล	21 คอนโทรล	77 คอนโทรล	77 คอนโทรล
เปอร์เซ็นต์ของยูสเซอร์คอนโทรลที่ไม่ถูกใช้งาน	75%	75%	84%	84%

จากผลการทดสอบพบว่าการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบโดยนับการใช้งานยูสเซอร์คอนโทรลภายในโครงสร้างยูไอมีความถูกต้อง เครื่องมือสามารถแสดงข้อมูลในรูปแบบตัวเลข และกราฟ ได้อย่างถูกต้อง และสามารถจัดพิมพ์เป็นรายงานได้อย่างสมบูรณ์ จึงสรุปได้ว่าเครื่องมือวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบสามารถทำงานได้อย่างถูกต้อง

5.4.4 ผลการทดสอบการทำงานของกรณีทดสอบ

ผลการทดสอบการทำงานของกรณีทดสอบกับโปรแกรมเครื่องคิดเลขและโปรแกรมสื่อสารผ่านทางเครือข่าย แสดงได้ดังตารางที่ 5.4

ตารางที่ 5.4 ตารางแสดงผลการทดสอบการทำงานของกรณีทดสอบ

การทดสอบ	ระบบที่ 1		ระบบที่ 2	
	คาดหวัง	ผลลัพธ์	คาดหวัง	ผลลัพธ์
จำนวนขั้นตอนในกรณีทดสอบที่สามารถทำงานได้สมบูรณ์จากกรณีทดสอบที่ได้จากเครื่องมือสร้างกรณีทดสอบ	28 ขั้นตอน	0 ขั้นตอน	92 ขั้นตอน	0 ขั้นตอน
จำนวนขั้นตอนในกรณีทดสอบที่สามารถทำงานได้สมบูรณ์จากกรณีทดสอบที่ได้รับการแก้ไขให้สอดคล้องกับการทำงาน	7 ขั้นตอน	0 ขั้นตอน	12 ขั้นตอน	12 ขั้นตอน

จากผลการทดสอบพบว่าเมื่อนำเอากรณีทดสอบที่ได้จากเครื่องมือสร้างกรณีทดสอบของทั้ง 2 ระบบมาใช้งาน ผลการทดสอบจากกรณีทดสอบดังกล่าวมีผลลัพธ์เป็น ไม่ผ่าน โดยกรณีทดสอบล้มเหลวตั้งแต่การทำงานขั้นตอนแรกสุด ที่กระทำกับยูสเซอร์คอนโทรลที่อยู่ด้านบนสุดและซ้ายสุดที่แสดงบนหน้าจอ

ในการทดสอบถัดไปจึงได้ทำการแก้ไขกรณีทดสอบให้ขั้นตอนการกระทำกับยูสเซอร์คอนโทรลมีความสอดคล้องกับการทำงานจริงของแต่ละระบบ โดยจำนวนขั้นตอนในกรณีทดสอบของระบบที่ 1 มีจำนวน 7 ขั้นตอน และจำนวนขั้นตอนในกรณีทดสอบของระบบที่ 2 มีจำนวน 12 ขั้นตอน ผลการทดสอบพบว่า ในระบบที่ 1 กรณีทดสอบมีผลลัพธ์เป็น ไม่ผ่าน โดยกรณีทดสอบล้มเหลวตั้งแต่การทำงานขั้นตอนแรกสุด ในระบบที่ 2 กรณีทดสอบมีผลลัพธ์เป็น ผ่าน โดยสามารถกระทำกับยูสเซอร์คอนโทรลต่างๆ ได้ครบตามที่กำหนดในกรณีทดสอบ

จากผลการทดสอบด้วยกรณีทดสอบที่ได้รับการแก้ไขของระบบที่ 1 ซึ่งมีผลลัพธ์ไม่ตรงตามที่คาดหวัง ผู้วิจัยได้ตรวจสอบและวิเคราะห์ถึงสาเหตุแล้วพบว่า ค่าออโตเมชันไอดีของยูสเซอร์คอนโทรลภายในโปรแกรมเครื่องคิดเลขของระบบที่ 1 มีความไม่แน่นอน ค่าดังกล่าวสามารถเปลี่ยนแปลงไปได้ในการเปิดโปรแกรมแต่ละครั้ง ดังนั้นจึงเป็นไปได้ที่ค่าออโตเมชันไอดีที่ถูกบันทึกไว้ในไฟล์โครงสร้างยูไอจะไม่ตรงกับค่าออโตเมชันไอดีของโปรแกรมในขณะที่ทำการทดสอบ

จึงสรุปได้ว่ากรณีทดสอบสำหรับการทำงานควรจะได้รับการแก้ไขให้สอดคล้องกับการทำงานของซอฟต์แวร์ที่ต้องการทดสอบ จึงจะสามารถทำงานได้อย่างสมบูรณ์และให้ผลลัพธ์ของการทดสอบอย่างถูกต้อง เนื่องจากขั้นตอนแรกภายในกรณีทดสอบที่ได้จากเครื่องมือสร้างกรณีทดสอบอาจจะเป็นยูสเซอร์คอนโทรลที่ไม่อยู่ในสถานะพร้อมในการทำงาน เช่น ข้อมูลยังไม่รับการอ่านขึ้นมา (Load) และยูสเซอร์คอนโทรลยังไม่พร้อมรับคำสั่ง (Disabled) เป็นต้น และซอฟต์แวร์ที่ต้องการทดสอบควร

จะมีความพร้อมสำหรับการทดสอบแบบอัตโนมัติด้วย เช่น ยูสเซอร์คอนโทรลมีค่าออโตเมชันไอดีที่คงที่ในการเปิดโปรแกรมแต่ละครั้ง และค่าออโตเมชันไอดีมีความเหมาะสมสำหรับการทดสอบแบบอัตโนมัติ ไม่ขึ้นต้นด้วยตัวเลข เป็นต้น

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

จากการศึกษา ออกแบบ พัฒนาและทดสอบเครื่องมือสำหรับการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติโดยใช้โครงสร้างยูไอ สามารถสรุปผลการวิจัย ข้อจำกัดของเครื่องมือ และข้อเสนอแนะสำหรับเป็นแนวทางในการพัฒนาเพิ่มเติม ได้ดังนี้

6.1 สรุปผลการวิจัย

งานวิจัยนี้นำเสนอเครื่องมือสำหรับการสร้างกรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติโดยใช้โครงสร้างยูไอ เพื่อทดสอบกับซอฟต์แวร์ที่อยู่ในรูปแบบวินโดวส์แอพพลิเคชันที่ใช้ยูสเซอร์คอนโทรลแบบ WinControl ซึ่งเครื่องมือช่วยสร้างคลาสที่เก็บข้อมูลเกี่ยวกับยูสเซอร์คอนโทรลของซอฟต์แวร์ที่ทำการทดสอบให้อยู่ในลักษณะโครงสร้าง มีลำดับชั้นสอดคล้องกับลำดับชั้นของยูสเซอร์คอนโทรลที่ถูกออกแบบและจัดวางไว้ในซอฟต์แวร์ให้โดยอัตโนมัติ จากข้อมูลดังกล่าวเครื่องมืออีกตัวหนึ่งจะช่วยสร้างกรณีทดสอบที่มีการกระทำต่างๆ กับยูสเซอร์คอนโทรลต่างๆ ให้โดยอัตโนมัติ อย่างไรก็ตามนักทดสอบสามารถปรับแต่ง แก้ไขลำดับการทำงานของขั้นตอนในกรณีทดสอบได้อย่างง่ายดายโดยใช้เครื่องมือแก้ไขกรณีทดสอบ ซึ่งช่วยให้การพัฒนากรณีทดสอบสำหรับการทดสอบซอฟต์แวร์แบบอัตโนมัติมีความสะดวกและรวดเร็ว

นอกจากนี้ยังมีเครื่องมือช่วยวิเคราะห์หาการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบเปรียบเทียบกับข้อมูลยูสเซอร์คอนโทรลที่อยู่ในรูปแบบโครงสร้างยูไอ เพื่อช่วยให้นักทดสอบทราบถึงยูสเซอร์คอนโทรลภายในซอฟต์แวร์ที่ยังไม่ได้รับการทดสอบ ทำให้นักทดสอบสามารถเพิ่มกรณีทดสอบได้อย่างมีประสิทธิภาพ

6.2 ข้อจำกัดของเครื่องมือ

1) เครื่องมือสามารถทำงานกับซอฟต์แวร์ที่ต้องการทดสอบที่พัฒนาด้วยกลุ่มภาษาโปรแกรมดอทเน็ต เช่น ซีชาร์ป เป็นต้น และทำงานบนระบบปฏิบัติการวินโดวส์เท่านั้น
2) ยูสเซอร์คอนโทรลที่ใช้ภายในซอฟต์แวร์ที่ต้องการทดสอบที่เครื่องมือรองรับ คือ ยูสเซอร์คอนโทรลมาตรฐานในประเภท WinControl เท่านั้น หากต้องการสร้างกรณีทดสอบสำหรับการทดสอบแบบอัตโนมัติสำหรับยูสเซอร์คอนโทรลที่เป็นชนิด Custom จะต้องได้รับการร่วมมือจากโปรแกรมเมอร์ของซอฟต์แวร์ดังกล่าว โดยการปรับปรุงซอฟต์แวร์โดยเฉพาะการทำงานของยูสเซอร์คอนโทรลให้มีการดำเนินการสอดคล้องกับมาตรฐานการสร้างซอฟต์แวร์เพื่อการทดสอบแบบอัตโนมัติ
3) เครื่องมือไม่สามารถอ่านค่าข้อมูลที่อยู่ภายในยูสเซอร์คอนโทรลแบบพลวัตได้ เช่น รายการภายในคอมโบบ็อกซ์ เป็นต้น

4) เครื่องมือจัดการกรณีทดสอบจะสร้างคำสั่งของแต่ละขั้นตอนการทำงานในกรณีทดสอบโดยใช้การกระทำมาตรฐานของยูสเซอร์คอนโทรลเท่านั้น ในกรณีที่ต้องการให้ยูสเซอร์คอนโทรลใช้คำสั่งเป็นการกระทำอื่นๆ นักทดสอบจะต้องใส่คำสั่งด้วยตัวเอง

5) กรณีทดสอบที่สามารถแก้ไขได้ผ่านเครื่องมือ จะต้องอยู่ในรูปแบบที่เครื่องมือกำหนดไว้เท่านั้น ในกรณีที่นักทดสอบทำการแก้ไขกรณีทดสอบโดยตรงผ่านเครื่องมือแก้ไขอื่นๆ กรณีทดสอบนั้นอาจไม่สามารถแก้ไขได้ผ่านทางเครื่องมือแก้ไขกรณีทดสอบอีกต่อไป

6) กรณีทดสอบที่ถูกสร้างขึ้นโดยเครื่องมือสร้างและแก้ไขกรณีทดสอบจะไม่มีการใส่คำสั่งสำหรับการตรวจสอบ นักทดสอบจะต้องเพิ่มคำสั่งดังกล่าวตามความเหมาะสมด้วยตัวเอง

7) ชุดคำสั่งที่เครื่องมือจัดการกรณีทดสอบสร้างให้โดยอัตโนมัตินั้นจะมีการทำงานโดยสั่งงานยูสเซอร์คอนโทรลโดยผ่านทางไลบราลี่ของนีโอออโตเมชันเฟรมเวิร์ก ซึ่งสามารถใช้งานได้กับยูสเซอร์คอนโทรลมาตรฐานทั่วไป แต่มีบางกรณีที่ยูสเซอร์คอนโทรลในบางซอฟต์แวร์ไม่สามารถทำงานได้อย่างสมบูรณ์ คาดว่าเนื่องจากการพัฒนาที่มีความซับซ้อนของซอฟต์แวร์

6.3 ข้อเสนอแนะ

1) พัฒนาเครื่องมือให้รองรับยูสเซอร์คอนโทรลประเภทอื่นๆ เพิ่มเติม เช่น ดับบลิวพีเอฟ เป็นต้น

2) ปรับปรุงเครื่องมือแก้ไขกรณีทดสอบให้นักทดสอบสามารถเลือกคำสั่งการกระทำอื่นๆ ได้นอกเหนือจากคำสั่งการกระทำมาตรฐานของยูสเซอร์คอนโทรลแต่ละชนิด

3) พัฒนาเครื่องมือเปรียบเทียบซอฟต์แวร์ที่ต้องการทดสอบกับโครงสร้างยูไอที่มีอยู่เพื่อตรวจสอบว่าซอฟต์แวร์มีการเปลี่ยนแปลงไปหรือไม่ ถ้าหากมีการเปลี่ยนแปลงก็ให้ทำการปรับปรุงแก้ไขโครงสร้างยูไอโดยอัตโนมัติ

4) พัฒนาการสร้างกรณีทดสอบแบบอัตโนมัติให้มีจำนวนและลำดับของขั้นตอนการทำงานภายในกรณีทดสอบที่เหมาะสม สอดคล้องกับการใช้งานจริงของผู้ใช้งาน

รายการอ้างอิง

1. Whittaker, J.A., *What Is Software Testing? And Why Is It So Hard?* Software, IEEE, 2000. **17**(1): p. 10.
2. Microsoft. *Regression Testing*. Available from: http://msdn.microsoft.com/en-us/library/aa292167(v=vs.71).aspx.
3. Aho, P.V.T.R.C.o.F., Espoo, Finland ; Menz, N. ; Raty, T. ; Schieferdecker, I., *Automated Java GUI Modeling for Model-Based Testing Purposes*, in *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*. 2011, IEEE: Las Vegas, NV. p. 268 - 273.
4. Mascarenhas, R. *Developing and Implementing an Automation Framework*. 2008.
5. Gopalakrishnan, A., *Conquest: An Interface for Test Automation Design*. Florida Institute of Technology, 2012. p. 232.
6. Microsoft. *Control Types and Their Supported Control Patterns*. Available from: http://msdn.microsoft.com/en-us/library/windows/desktop/ee671193(v=vs.85).aspx.
7. Karhu, K.L.U.o.T., Lappeenranta ; Repo, T. ; Taipale, O. ; Smolander, K., *Empirical Observations on Software Testing Automation*, in *Software Testing Verification and Validation, 2009. ICST '09. International Conference on*. 2009, IEEE: Denver, CO. p. 201 - 209.
8. Microsoft. *Visual Studio Test Professional*. Available from: http://www.microsoft.com/visualstudio/eng#products/visual-studio-test-professional-2012+product-edition-testpro.
9. Hewlett-Packard. *HP Unified Functional Testing*. Available from: http://www8.hp.com/us/en/software-solutions/software.html?compURI=1172957.
10. IBM. *Rational Functional Tester*. Available from: http://www-01.ibm.com/software/awdtools/tester/functional/.
11. TestStack. *White*. Available from: http://teststack.github.com/White/.
12. Private, T.S. *Sahi*. Available from: http://sahi.co.in/.
13. Grechanik, M.A.T.L., Chicago, IL ; Xie, Qing ; Chen Fu, *Creating GUI Testing Tools Using Accessibility Technologies*, in *Software Testing, Verification and Validation Workshops, 2009. ICSTW '09. International Conference on*. 2009, IEEE: Denver, CO. p. 243 - 250.
14. Microsoft. *Microsoft Accessibility*. Available from: http://www.microsoft.com/enable/.
15. Microsoft. *MSAA*. Available from: http://msdn.microsoft.com/en-us/library/ms971310.aspx.

16. Microsoft. *Microsoft UI Automation*. Available from: http://msdn.microsoft.com/en-us/library/ms747327.aspx.
17. Microsoft. *UI Automation Control Types Overview (Windows)*. Available from: http://msdn.microsoft.com/en-us/library/windows/desktop/ee671197(v=vs.85).aspx.
18. Bin Mu ; Tongji Univ., S., China ; Mingkui Zhan ; Lanfang Hu, *Design and Implementation of GUI Automated Testing Framework Based on XML*, in *Software Engineering, 2009. WCSE '09. WRI World Congress on*. 2009, IEEE: Xiamen. p. 194 - 199.
19. Grilo, A.M.P.F.d.E., Dept. de Eng. Inf., Univ. do Porto, Porto, Portugal ; Paiva, A.C.R. ; Faria, J.P., *Reverse engineering of GUI models for testing*, in *Information Systems and Technologies (CISTI), 2010 5th Iberian Conference on*. 2010: Santiago de Compostela. p. 1 - 6.
20. Ruiz, A.P., Y.W., *GUI Testing Made Easy*, in *Practice and Research Techniques, 2008. TAIC PART '08. Testing: Academic & Industrial Conference*. 2008, IEEE: Windsor. p. 99 - 103.

ภาคผนวก

ภาคผนวก ก.
รายละเอียดยูสเคส

ตารางที่ ก-1 รายละเอียดยูสเคส View and Highlight UI Control

รหัสยูสเคส:	1
ชื่อยูสเคส:	View and Highlight UI Control
ผู้กระทำหลัก (Actor):	นักทดสอบ
รายละเอียดยูสเคส:	แสดงโครงสร้างและรายละเอียดของยูสเซอร์คอนโทรล
ความสัมพันธ์	Extends: Change Setting
เงื่อนไขก่อนหน้า:	ซอฟต์แวร์ที่ต้องการทดสอบเข้าสู่ระบบ
ขั้นตอนการทำงาน:	1. นักทดสอบระบุยูสเซอร์คอนโทรลบนซอฟต์แวร์เป้าหมายที่ต้องการทดสอบ 2. ระบบทำการค้นหายูสเซอร์คอนโทรลที่เป็นจุดเริ่มต้น (Root) ซึ่งเป็นประเภท Form 3. ระบบทำการวิเคราะห์และจัดเก็บข้อมูลยูสเซอร์คอนโทรลที่เกี่ยวข้องทั้งหมด 4. ระบบแสดงรายละเอียดยูสเซอร์คอนโทรลในรูปแบบโครงสร้างลำดับชั้น 5. นักทดสอบสามารถเรียกใช้งานยูสเคส Change Setting เพื่อตั้งค่าการแสดงยูสเซอร์คอนโทรล 6. นักทดสอบเลือกยูสเซอร์คอนโทรล 7. ระบบแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลที่เลือกบนหน้าจอ
เงื่อนไขภายหลัง:	ข้อมูลยูสเซอร์คอนโทรลถูกจัดเตรียมอยู่ในรูปแบบโครงสร้างลำดับชั้น

ตารางที่ ก-2 รายละเอียดยูสเคส Change Setting

รหัสยูสเคส:	2
ชื่อยูสเคส:	Change Setting
ผู้กระทำหลัก:	-

ตารางที่ ก-2 รายละเอียดยูสเคส Change Setting (ต่อ)

รายละเอียดยูสเคส:	ตั้งค่าการแสดงยูสเซอร์คอนโทรล
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. นักทดสอบตั้งค่าการหน่วงเวลาสำหรับแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลบนหน้าจอตามที่ต้องการ 2. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล ของยูสเซอร์คอนโทรลเป้าหมายตามที่ต้องการ 3. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล ของยูสเซอร์คอนโทรลที่รองรับยูสเซอร์คอนโทรลเป้าหมายตามที่ต้องการ 4. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล สำหรับยูสเซอร์คอนโทรลที่ถูกเลือกตามที่ต้องการ 5. กดปุ่มบันทึกการตั้งค่า
เงื่อนไขภายหลัง:	การแสดงยูสเซอร์คอนโทรลเป็นไปตามที่ตั้งค่า

ตารางที่ ก-3 รายละเอียดยูสเคส Generate UI Structure

รหัสยูสเคส:	3
ชื่อยูสเคส:	Generate UI Structure
ผู้กระทำหลัก:	นักทดสอบ
รายละเอียดยูสเคส:	จัดสร้างโครงสร้างยูไอ
ความสัมพันธ์	Uses: View and Highlight UI Control, Save UI Structure Extends: View Log
เงื่อนไขก่อนหน้า:	ข้อมูลยูสเซอร์คอนโทรลถูกจัดเตรียมอยู่ในรูปแบบโครงสร้างลำดับชั้น
ขั้นตอนการทำงาน:	1. เรียกใช้งานยูสเคส View and Highlight UI Control เพื่อแสดงโครงสร้างและรายละเอียดของยูสเซอร์คอนโทรล 2. นักทดสอบกำหนดชื่อเนมสเปซที่ต้องการ 3. นักทดสอบกดปุ่มสร้างโครงสร้างยูไอ

ตารางที่ ก-3 รายละเอียดยูสเคส Generate UI Structure (ต่อ)

	4. ระบบทำการสร้างโครงสร้างยูไอจากข้อมูลยูสเซอร์คอนโทรลแบบลำดับชั้นให้อยู่ในรูปแบบคลาส 5. ระบบแสดงผลลัพธ์บนหน้าจอ 6. เรียกใช้งานยูสเคส Save UI Structure เพื่อบันทึกโครงสร้างยูไอลงเป็นไฟล์ 7. นักทดสอบสามารถเรียกใช้งานยูสเคส View Log เพื่อดูรายละเอียดการทำงานของการสร้างโครงสร้างยูไอ
เงื่อนไขภายหลัง:	โครงสร้างยูไอถูกสร้างโดยอยู่ในรูปแบบคลาสในภาษาซีชาร์ป

ตารางที่ ก-4 รายละเอียดยูสเคส Save UI Structure

รหัสยูสเคส:	4
ชื่อยูสเคส:	Save UI Structure
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	บันทึกโครงสร้างยูไอลงเป็นไฟล์
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	โครงสร้างยูไอถูกจัดเตรียมในรูปแบบคลาสในภาษาซีชาร์ป
ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้ระบุตำแหน่งเพื่อบันทึกไฟล์ 2. นักทดสอบระบุตำแหน่งที่ต้องการบันทึกไฟล์และชื่อไฟล์ของคลาส 3. ระบบบันทึกข้อมูล
เงื่อนไขภายหลัง:	คลาสของโครงสร้างยูไอถูกจัดเก็บลงไฟล์

ตารางที่ ก-5 รายละเอียดยูสเคส View Log

รหัสยูสเคส:	5
ชื่อยูสเคส:	View Log
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	ดูรายละเอียดการทำงานของการสร้างโครงสร้างยูไอ

ตารางที่ ก-5 รายละเอียดยูสเคส View Log (ต่อ)

ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. นักทดสอบกดปุ่มแสดงรายละเอียด 2. ระบบแสดงข้อมูลการทำงานที่ผ่านมาพร้อมรายละเอียดและวันเวลาของการทำงาน
เงื่อนไขภายหลัง:	-

ตารางที่ ก-6 รายละเอียดยูสเคส Load UI Structure

รหัสยูสเคส:	6
ชื่อยูสเคส:	Load UI Structure
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	นำเข้าไฟล์โครงสร้างยูไอ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	โครงสร้างยูไอถูกคอมไพล์อยู่ในรูปแบบไฟล์แอสเซมบลีที่มีนามสกุลเป็น DLL
ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้เลือกไฟล์ 2. นักทดสอบเลือกไฟล์แอสเซมบลีของโครงสร้างยูไอที่ต้องการ 3. ระบบนำเข้าข้อมูล
เงื่อนไขภายหลัง:	ไฟล์แอสเซมบลีของโครงสร้างยูไอถูกนำเข้าในระบบ

ตารางที่ ก-7 รายละเอียดยูสเคส Generate Test Case

รหัสยูสเคส:	7
ชื่อยูสเคส:	Generate Test Case
ผู้กระทำหลัก:	นักทดสอบ
รายละเอียดยูสเคส:	สร้างกรณีทดสอบ
ความสัมพันธ์	Uses: Load UI Structure, Save Test Case

ตารางที่ ก-7 รายละเอียดยูสเคส Generate Test Case (ต่อ)

	Extends: Change Setting, View Log
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. เรียกใช้งานยูสเคส Load UI Structure เพื่อนำเข้าไฟล์โครงสร้างยูไอ 2. นักทดสอบเลือกโครงสร้างยูไอจากไฟล์ที่นำเข้า 3. นักทดสอบระบุชื่อเนมสเปซ คลาส และกรณีทดสอบ ที่ต้องการสร้าง 4. กดปุ่มสร้างกรณีทดสอบ 5. ระบบแสดงผลลัพธ์บนหน้าจอ 6. นักทดสอบสามารถเรียกใช้งานยูสเคส Change Setting เพื่อตั้งค่าการทำงานสำหรับกรณีทดสอบ 7. เรียกใช้งานยูสเคส Save Test Case เพื่อบันทึกกรณีทดสอบ 8. นักทดสอบสามารถเรียกใช้งานยูสเคส View Log เพื่อดูรายละเอียดการทำงานของการสร้างกรณีทดสอบ
เงื่อนไขภายหลัง:	กรณีทดสอบถูกสร้างขึ้นอยู่ในรูปแบบของการทดสอบระดับหน่วยในภาษาซีชาร์ป

ตารางที่ ก-8 รายละเอียดยูสเคส Change Setting

รหัสยูสเคส:	8
ชื่อยูสเคส:	Change Setting
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	ตั้งค่าการทำงานสำหรับกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. นักทดสอบตั้งค่าว่าต้องการให้แสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลหรือไม่ 2. นักทดสอบตั้งค่าเกี่ยวกับการค้นหายูสเซอร์คอนโทรลและการแสดงความผิดพลาดหากการกระทำกับยูสเซอร์คอนโทรลมีปัญหาตามที่

ตารางที่ ก-8 รายละเอียดยูสเคส Change Setting (ต่อ)

	ต้องการ 3. นักทดสอบตั้งค่าการหน่วงเวลาสำหรับแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลบนหน้าจอตามที่ต้องการ 4. นักทดสอบตั้งค่าการหน่วงเวลาหลังจากกระทำกับยูสเซอร์คอนโทรลตามที่ต้องการ 5. นักทดสอบตั้งค่าการหน่วงเวลาสำหรับค้นหายูสเซอร์คอนโทรลตามที่ต้องการ 6. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล ของยูสเซอร์คอนโทรลเป้าหมายตามที่ต้องการ 7. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล ของยูสเซอร์คอนโทรลที่รองรับยูสเซอร์คอนโทรลเป้าหมายตามที่ต้องการ 8. กดปุ่มตกลงเพื่อบันทึกการตั้งค่า
เงื่อนไขภายหลัง:	กรณีทดสอบได้รับการตั้งค่าเป็นไปตามที่เลือกไว้

ตารางที่ ก-9 รายละเอียดยูสเคส Save Test Case

รหัสยูสเคส:	9
ชื่อยูสเคส:	Save Test Case
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	บันทึกกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	กรณีทดสอบถูกสร้างขึ้นในรูปแบบการทดสอบระดับหน่วยในภาษาซีชาร์ป
ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้ระบุตำแหน่งเพื่อบันทึกไฟล์ 2. นักทดสอบระบุตำแหน่งที่ต้องการบันทึกไฟล์และชื่อไฟล์ของกรณีทดสอบ 3. ระบบบันทึกข้อมูล
เงื่อนไขภายหลัง:	กรณีทดสอบถูกจัดเก็บลงไฟล์

ตารางที่ ก-10 รายละเอียดยูสเคส View Log

รหัสยูสเคส:	10
ชื่อยูสเคส:	View Log
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	ดูรายละเอียดการทำงานของการสร้างกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. นักทดสอบกดปุ่มแสดงรายละเอียด 2. ระบบแสดงข้อมูลการทำงานที่ผ่านมาพร้อมรายละเอียดและวันเวลาของการทำงาน
เงื่อนไขภายหลัง:	-

ตารางที่ ก-11 รายละเอียดยูสเคส Load UI Structure

รหัสยูสเคส:	11
ชื่อยูสเคส:	Load UI Structure
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	นำเข้าไฟล์โครงสร้างยูไอ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	โครงสร้างยูไอถูกคอมไพล์อยู่ในรูปแบบไฟล์แอสเซมบลีที่มีนามสกุลเป็น DLL
ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้เลือกไฟล์ 2. นักทดสอบเลือกไฟล์แอสเซมบลีของโครงสร้างยูไอที่ต้องการ 3. ระบบนำเข้าข้อมูล
เงื่อนไขภายหลัง:	ไฟล์แอสเซมบลีของโครงสร้างยูไอถูกนำเข้าในระบบ

ตารางที่ ก-12 รายละเอียดยูสเคส Load Test Case (ต่อ)

รหัสยูสเคส:	12
ชื่อยูสเคส:	Load Test Case
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	นำเข้าไฟล์กรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้เลือกไฟล์ 2. นักทดสอบเลือกไฟล์กรณีทดสอบในรูปแบบคลาสภาษาซีชาร์ป 3. ระบบนำเข้าข้อมูล
เงื่อนไขภายหลัง:	ไฟล์กรณีทดสอบถูกนำเข้าในระบบ

ตารางที่ ก-13 รายละเอียดยูสเคส Edit Test Case

รหัสยูสเคส:	13
ชื่อยูสเคส:	Edit Test Case
ผู้กระทำหลัก:	นักทดสอบ
รายละเอียดยูสเคส:	แก้ไขกรณีทดสอบ
ความสัมพันธ์	Uses: Load UI Structure, Load Test Case, Save Test Case Extends: Change Setting, View Log
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. เรียกใช้งานยูสเคส Load UI Structure เพื่อนำเข้าไฟล์โครงสร้างยูไอ 2. นักทดสอบเลือกโครงสร้างยูไอจากไฟล์ที่นำเข้า 3. เรียกใช้งานยูสเคส Load Test Case เพื่อนำเข้าไฟล์กรณีทดสอบ 4. นักทดสอบเลือกกรณีทดสอบจากไฟล์ที่นำเข้า 5. แก้ไขรายละเอียดของกรณีทดสอบตามที่ต้องการ 6. นักทดสอบสามารถเรียกใช้งานยูสเคส Change Setting เพื่อตั้งค่า

ตารางที่ ก-13 รายละเอียดยูสเคส Edit Test Case (ต่อ)

	การทำงานสำหรับกรณีทดสอบ 7. กดปุ่มบันทึกข้อมูล 8. เรียกใช้งานยูสเคส Save Test Case เพื่อบันทึกการแก้ไขกรณีทดสอบ 9. ระบบบันทึกข้อมูลการแก้ไข 10. นักทดสอบสามารถเรียกใช้งานยูสเคส View Log เพื่อดูรายละเอียดการทำงานของการแก้ไขกรณีทดสอบ
เงื่อนไขภายหลัง:	กรณีทดสอบถูกแก้ไขและบันทึกลงไปยังไฟล์

ตารางที่ ก-14 รายละเอียดยูสเคส Change Setting

รหัสยูสเคส:	14
ชื่อยูสเคส:	Change Setting
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	ตั้งค่าการทำงานสำหรับกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. นักทดสอบตั้งค่าว่าต้องการให้แสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลหรือไม่ 2. นักทดสอบตั้งค่าเกี่ยวกับการค้นหายูสเซอร์คอนโทรลและการแสดงความผิดพลาดหากการกระทำกับยูสเซอร์คอนโทรลมีปัญหาตามที่ต้องการ 3. นักทดสอบตั้งค่าการหน่วงเวลาสำหรับแสดงสีเน้นให้เห็นตำแหน่งของยูสเซอร์คอนโทรลบนหน้าจอตามที่ต้องการ 4. นักทดสอบตั้งค่าการหน่วงเวลาหลังจากกระทำกับยูสเซอร์คอนโทรลตามที่ต้องการ 5. นักทดสอบตั้งค่าการหน่วงเวลาสำหรับค้นหายูสเซอร์คอนโทรลตามที่ต้องการ 6. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์

ตารางที่ ก-14 รายละเอียดยูสเคส Change Setting (ต่อ)

	คอนโทรล ของยูสเซอร์คอนโทรลเป้าหมายตามที่ต้องการ 7. นักทดสอบตั้งค่าสี ความหนาของเส้นสี ระยะเว้นตรงขอบยูสเซอร์คอนโทรล ของยูสเซอร์คอนโทรลที่รองรับยูสเซอร์คอนโทรลเป้าหมายตามที่ต้องการ 8. กดปุ่มตกลงเพื่อบันทึกการตั้งค่า
เงื่อนไขภายหลัง:	กรณีทดสอบได้รับการตั้งค่าเป็นไปตามที่เลือกไว้

ตารางที่ ก-15 รายละเอียดยูสเคส Save Test Case

รหัสยูสเคส:	15
ชื่อยูสเคส:	Save Test Case
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	บันทึกการแก้ไขกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	กรณีทดสอบได้รับการแก้ไขเรียบร้อยแล้ว
ขั้นตอนการทำงาน:	1. กดปุ่มเพื่อบันทึกข้อมูล 2. กดยืนยันการแก้ไขข้อมูล 3. ระบบบันทึกข้อมูล
เงื่อนไขภายหลัง:	กรณีทดสอบได้รับการแก้ไขและบันทึกลงไฟล์

ตารางที่ ก-16 รายละเอียดยูสเคส View Log

รหัสยูสเคส:	16
ชื่อยูสเคส:	View Log
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	ดูรายละเอียดการทำงานของการแก้ไขกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-

ตารางที่ ก-16 รายละเอียดยูสเคส View Log (ต่อ)

ขั้นตอนการทำงาน:	1. นักทดสอบกดปุ่มแสดงรายละเอียด 2. ระบบแสดงข้อมูลการทำงานที่ผ่านมาพร้อมรายละเอียดและวันเวลาของการทำงาน
เงื่อนไขภายหลัง:	-

ตารางที่ ก-17 รายละเอียดยูสเคส Load UI Structure

รหัสยูสเคส:	17
ชื่อยูสเคส:	Load UI Structure
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	นำเข้าไฟล์โครงสร้างยูไอ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	โครงสร้างยูไอถูกคอมไพล์อยู่ในรูปแบบไฟล์แอสเซมบลีที่มีนามสกุลเป็น DLL
ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้เลือกไฟล์ 2. นักทดสอบเลือกไฟล์แอสเซมบลีของโครงสร้างยูไอที่ต้องการ 3. ระบบนำเข้าข้อมูล
เงื่อนไขภายหลัง:	ไฟล์แอสเซมบลีของโครงสร้างยูไอถูกนำเข้าในระบบ

ตารางที่ ก-18 รายละเอียดยูสเคส Load Test Case

รหัสยูสเคส:	18
ชื่อยูสเคส:	Load Test Case
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	นำเข้าไฟล์กรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	กรณีทดสอบถูกคอมไพล์อยู่ในรูปแบบไฟล์แอสเซมบลีที่มีนามสกุลเป็น DLL

ตารางที่ ก-18 รายละเอียดยูสเคส Load Test Case (ต่อ)

ขั้นตอนการทำงาน:	1. ระบบแสดงหน้าจอให้เลือกไฟล์ 2. นักทดสอบเลือกไฟล์แอสเซมบลีของกรณีทดสอบที่ต้องการ 3. ระบบนำเข้าข้อมูล
เงื่อนไขภายหลัง:	ไฟล์แอสเซมบลีของกรณีทดสอบถูกนำเข้าในระบบ

ตารางที่ ก-19 รายละเอียดยูสเคส Analyze UI Usage

รหัสยูสเคส:	19
ชื่อยูสเคส:	Analyze UI Usage
ผู้กระทำหลัก:	นักทดสอบ
รายละเอียดยูสเคส:	วิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
ความสัมพันธ์	Uses: Load UI Structure, Load Test Case Extends: Create Report, View Log
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. เรียกใช้งานยูสเคส Load UI Structure เพื่อนำเข้าไฟล์โครงสร้างยูไอ 2. นักทดสอบเลือกโครงสร้างยูไอจากไฟล์ที่นำเข้า 3. เรียกใช้งานยูสเคส Load Test Case เพื่อนำเข้าไฟล์กรณีทดสอบ 4. นักทดสอบเลือกกรณีทดสอบจากไฟล์ที่นำเข้า 5. กดปุ่มวิเคราะห์ข้อมูล 6. ระบบแสดงผลลัพธ์บนหน้าจอ 7. นักทดสอบสามารถเรียกใช้งานยูสเคส Create Report เพื่อสร้างรายงานการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ 8. นักทดสอบสามารถเรียกใช้งานยูสเคส View Log เพื่อดูรายละเอียดการทำงานของการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
เงื่อนไขภายหลัง:	ข้อมูลการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบถูกแสดงบนหน้าจอ

ตารางที่ ก-20 รายละเอียดยูสเคส Create Report

รหัสยูสเคส:	20
ชื่อยูสเคส:	Create Report
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	สร้างรายงานการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	ข้อมูลการใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบถูกแสดงบนหน้าจอ
ขั้นตอนการทำงาน:	1. กดปุ่มสร้างรายงาน 2. นักทดสอบเลือกโครงสร้างยูไอที่ต้องการ 3. เลือกประเภทของรายงานที่ต้องการ 4. กดปุ่มแสดงรายงาน 5. รายงานถูกแสดงบนหน้าจอ
เงื่อนไขภายหลัง:	รายงานถูกแสดงบนหน้าจอ นักทดสอบสามารถสั่งพิมพ์ได้

ตารางที่ ก-21 รายละเอียดยูสเคส View Log

รหัสยูสเคส:	21
ชื่อยูสเคส:	View Log
ผู้กระทำหลัก:	-
รายละเอียดยูสเคส:	ดูรายละเอียดการทำงานของการวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. นักทดสอบกดปุ่มแสดงรายละเอียด 2. ระบบแสดงข้อมูลการทำงานที่ผ่านมาพร้อมรายละเอียดและวันเวลาของการทำงาน
เงื่อนไขภายหลัง:	-

ตารางที่ ก-22 รายละเอียดยูสเคส Extract User Control Information

รหัสยูสเคส:	22
ชื่อยูสเคส:	Extract User Control Information
ผู้กระทำหลัก:	ส่วนสร้างโครงสร้างยูไอ
รายละเอียดยูสเคส:	วิเคราะห์ข้อมูลของยูสเซอร์คอนโทรล
ความสัมพันธ์	Uses: Search User Control
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. รับค่าตำแหน่งพิกัดบนหน้าจอจากส่วนสร้างโครงสร้างยูไอ 2. เรียกใช้งานยูสเคส Search User Control เพื่อค้นหายูสเซอร์คอนโทรล 3. เมื่อพบยูสเซอร์คอนโทรล ทำการอ่านข้อมูลที่เกี่ยวข้องกับยูสเซอร์คอนโทรล 4. ส่งข้อมูลยูสเซอร์คอนโทรลกลับไปยังผู้เรียก
เงื่อนไขภายหลัง:	-

ตารางที่ ก-23 รายละเอียดยูสเคส Search User Control

รหัสยูสเคส:	23
ชื่อยูสเคส:	Search User Control
ผู้กระทำหลัก:	ส่วนสร้างโครงสร้างยูไอ
รายละเอียดยูสเคส:	ค้นหายูสเซอร์คอนโทรล
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. รับค่าออโตเมชันไอดีของยูสเซอร์คอนโทรลที่ต้องการ 2. ทำการค้นหายูสเซอร์คอนโทรลที่มีออโตเมชันไอดีตรงกับที่ต้องการ 3. เมื่อพบยูสเซอร์คอนโทรลที่ต้องการ คืนผลลัพธ์การค้นหากลับไปยังผู้เรียก
เงื่อนไขภายหลัง:	-

ตารางที่ ก-24 รายละเอียดยูสเคส Perform User Control Action

รหัสยูสเคส:	24
ชื่อยูสเคส:	Perform User Control Action
ผู้กระทำหลัก:	ส่วนจัดการการทดสอบ
รายละเอียดยูสเคส:	สั่งงานยูสเซอร์คอนโทรลให้ทำงานตามคำสั่ง
ความสัมพันธ์	Uses: Search User Control
เงื่อนไขก่อนหน้า:	กรณีทดสอบได้รับการสั่งงานโดยส่วนจัดการการทดสอบ
ขั้นตอนการทำงาน:	1. อ่านค่าออโตเมชันไอดีของยูสเซอร์คอนโทรลที่ถูกเรียกใช้ภายในกรณีทดสอบ 2. เรียกใช้งานยูสเคส Search User Control เพื่อค้นหายูสเซอร์คอนโทรล 3. ส่งคำสั่งที่กำหนดไว้ในกรณีทดสอบไปยังยูสเซอร์คอนโทรล
เงื่อนไขภายหลัง:	ยูสเซอร์คอนโทรลกระทำตามคำสั่งที่กำหนด

ตารางที่ ก-25 รายละเอียดยูสเคส List User Control Action

รหัสยูสเคส:	25
ชื่อยูสเคส:	List User Control Action
ผู้กระทำหลัก:	ส่วนสร้างกรณีทดสอบ, ส่วนแก้ไขกรณีทดสอบ
รายละเอียดยูสเคส:	แสดงรายชื่อการกระทำต่างๆ ที่สอดคล้องกับประเภทของยูสเซอร์คอนโทรล
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-
ขั้นตอนการทำงาน:	1. รับค่าประเภทของยูสเซอร์คอนโทรล 2. ค้นหาการกระทำที่สามารถดำเนินการได้กับยูสเซอร์คอนโทรลประเภทที่ต้องการ 3. คืนผลลัพธ์การค้นหากลับไปยังผู้เรียก
เงื่อนไขภายหลัง:	-

ตารางที่ ก-26 รายละเอียดยูสเคส Analyze UI Usage

รหัสยูสเคส:	26
ชื่อยูสเคส:	Analyze UI Usage
ผู้กระทำหลัก:	ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
รายละเอียดยูสเคส:	วิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	1. ข้อมูลโครงสร้างยูไอถูกนำเข้า 2. ข้อมูลกรณีทดสอบถูกนำเข้า
ขั้นตอนการทำงาน:	1. อ่านค่าข้อมูลโครงสร้างยูไอที่ต้องการวิเคราะห์ 2. นับจำนวนยูสเซอร์คอนโทรลภายในโครงสร้างยูไอ 3. อ่านค่าข้อมูลกรณีทดสอบที่ต้องการวิเคราะห์ 4. ค้นหาการเรียกใช้งานยูสเซอร์คอนโทรลของโครงสร้างยูไอที่ต้องการภายในกรณีทดสอบ 5. ประมวลผลการเรียกใช้งานยูสเซอร์คอนโทรล 6. สรุปผลการเรียกใช้งานยูสเซอร์คอนโทรลเป็นจำนวนและเปอร์เซ็นต์ 7. คืนผลลัพธ์การวิเคราะห์ข้อมูลกลับไปยังผู้เรียก
เงื่อนไขภายหลัง:	-

ตารางที่ ก-27 รายละเอียดยูสเคส Write Log

รหัสยูสเคส:	27
ชื่อยูสเคส:	Write Log
ผู้กระทำหลัก:	ส่วนสร้างโครงสร้างยูไอ, ส่วนสร้างกรณีทดสอบ, ส่วนแก้ไขกรณีทดสอบ, ส่วนวิเคราะห์การใช้งานยูสเซอร์คอนโทรลภายในกรณีทดสอบ
รายละเอียดยูสเคส:	บันทึกรายละเอียดการทำงาน
ความสัมพันธ์	-
เงื่อนไขก่อนหน้า:	-

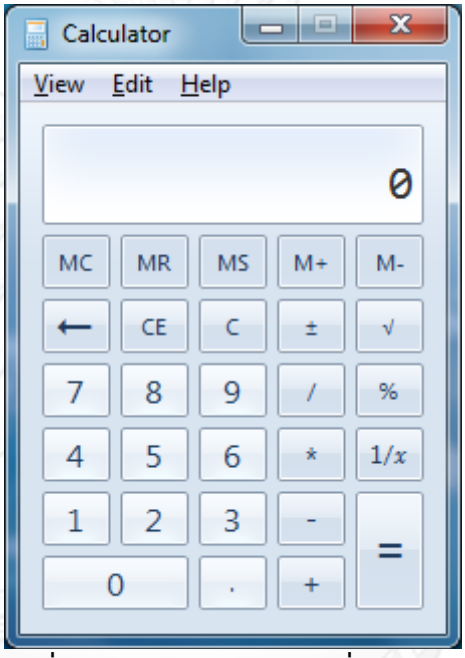
ตารางที่ ก-27 รายละเอียดยูสเคส Write Log (ต่อ)

ขั้นตอนการทำงาน:	1. รับค่าข้อมูลการทำงานที่ต้องการบันทึก 2. ค้นหาไฟล์รายละเอียดการทำงาน 3. บันทึกข้อมูลการทำงานลงไปในไฟล์รายละเอียดการทำงานที่ต้องการ
เงื่อนไขภายหลัง:	-

ภาคผนวก ข.
ระบบที่ใช้ในการทดสอบ

1. ระบบที่ 1

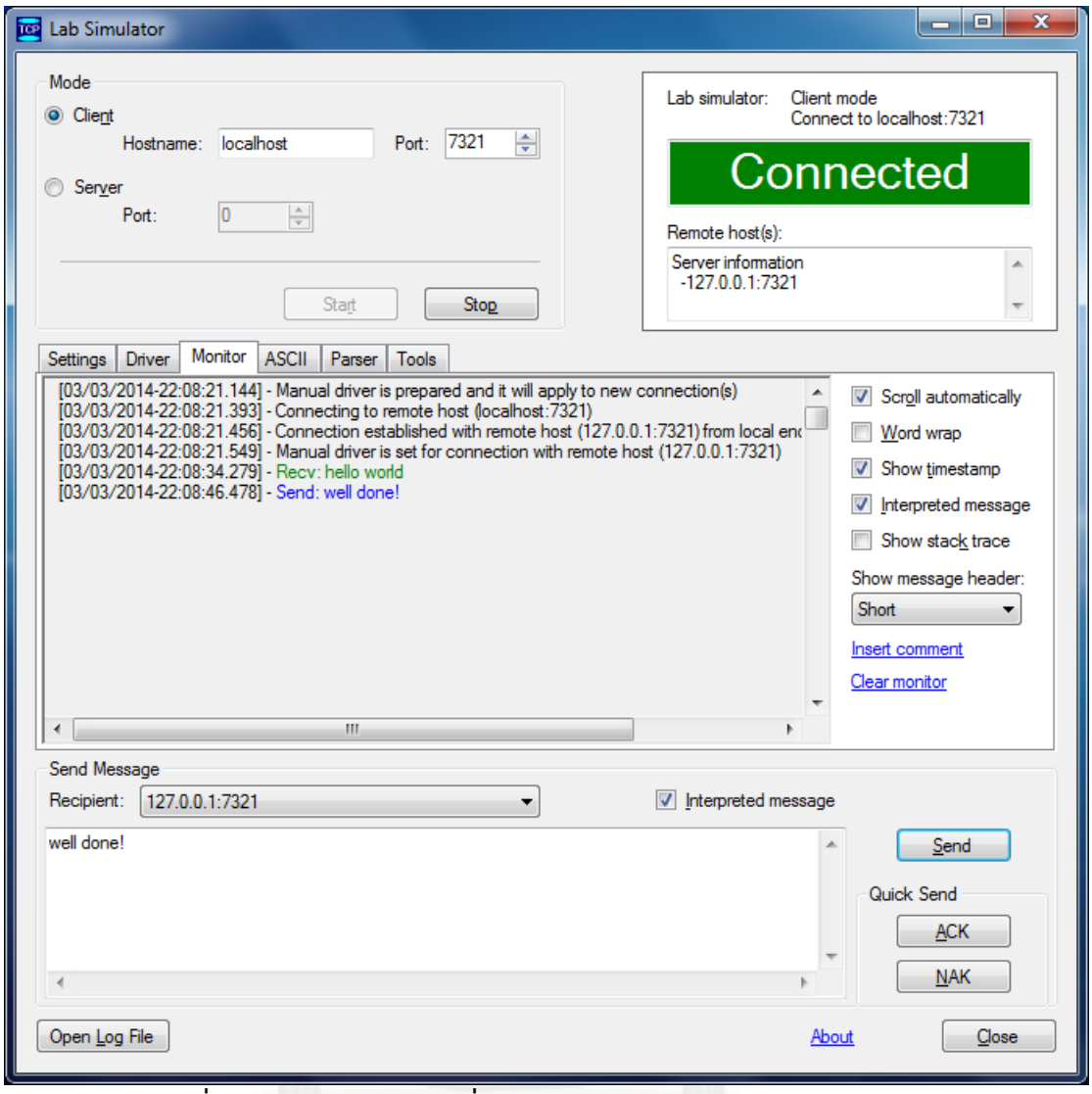
โปรแกรมเครื่องคิดเลข (Calculator) ที่มากับระบบปฏิบัติการไมโครซอฟท์วินโดวส์เซเว่น ในหน้าจอประกอบไปด้วยยูสเซอร์คอนโทรลเป็นจำนวนมากเพื่อรับคำสั่งในการคำนวณ และแสดงผลลัพธ์ของการคำนวณบนหน้าจอ ตัวอย่างหน้าจอแสดงดังรูปที่ ข-1



รูปที่ ข-1 หน้าจอโปรแกรมเครื่องคิดเลข

2. ระบบที่ 2

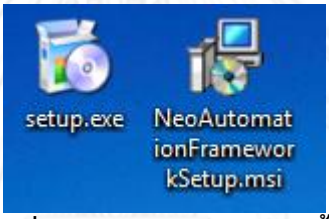
โปรแกรมสื่อสารผ่านทางเครือข่ายที่ถูกพัฒนาและใช้งานภายในบริษัทโอไรออน เฮลท์ (ประเทศไทย) จำกัด เรียกว่า Lab Simulator มีฟังก์ชันการทำงานเป็นจำนวนมาก เช่น สามารถทำการรอรับการเชื่อมต่อจากเครือข่ายหรือทำการติดต่อไปยังปลายทางอื่นภายในเครือข่ายได้ เป็นต้น ตัวอย่างหน้าจอแสดงดังรูปที่ ข-2



รูปที่ ข-2 หน้าจอโปรแกรมสื่อสารผ่านทางเครือข่าย Lab Simulator

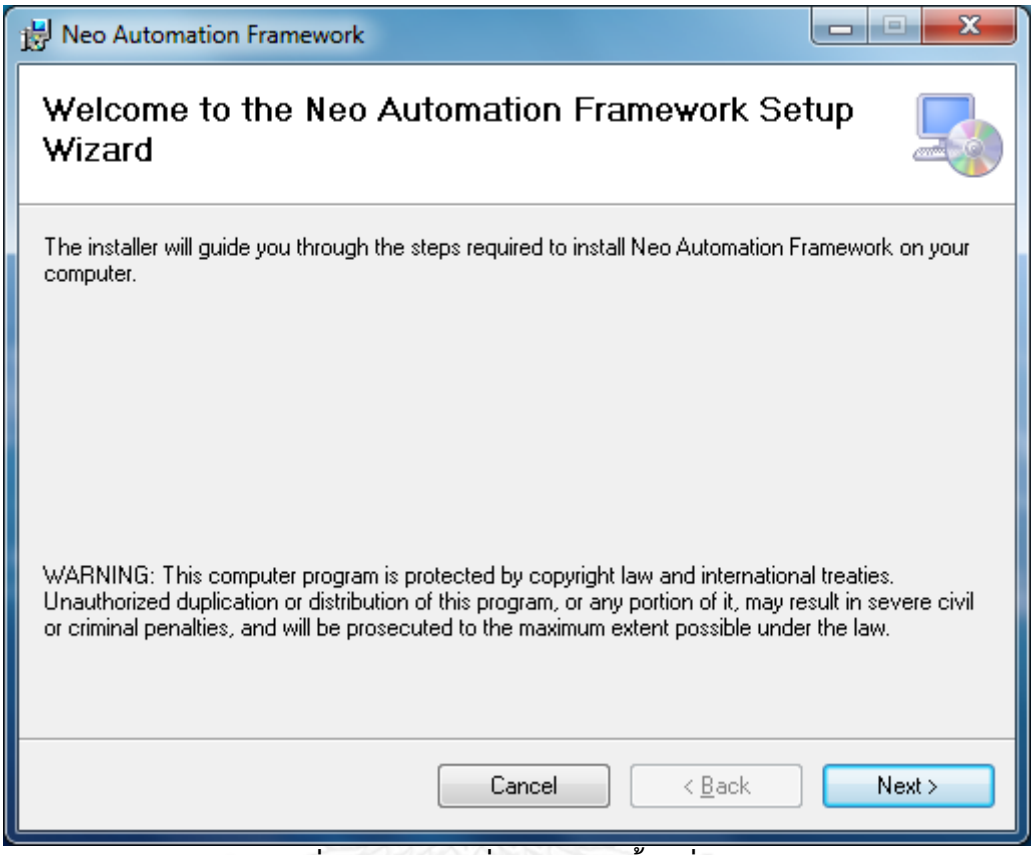
ภาคผนวก ค.
คู่มือการติดตั้งเครื่องมือ

1. ความต้องการของระบบ (System Requirement) มีดังนี้
   a. ดอทเน็ตเฟรมเวิร์ก (.NET Framework) เวอร์ชัน 4.0 ขึ้นไป
   b. เครื่องมือพัฒนาซอฟต์แวร์ภาษาตระกูลดอทเน็ตเฟรมเวิร์ก เช่น ไมโครซอฟท์วิชวลสตูดิโอ (Microsoft Visual Studio) เวอร์ชัน 2010 ขึ้นไป
2. ขั้นตอนการติดตั้งเครื่องมือ มีดังนี้
   a. ใส่แผ่นซีดีรอม (CD-Rom) ติดตั้งเครื่องมือ
   b. ไฟล์สำหรับการติดตั้งประกอบไปด้วย 2 ไฟล์ดังรูปที่ ค-1 เริ่มการติดตั้งเครื่องมือโดยดับเบิ้ลคลิก (Double Click) ที่ไฟล์ setup.exe



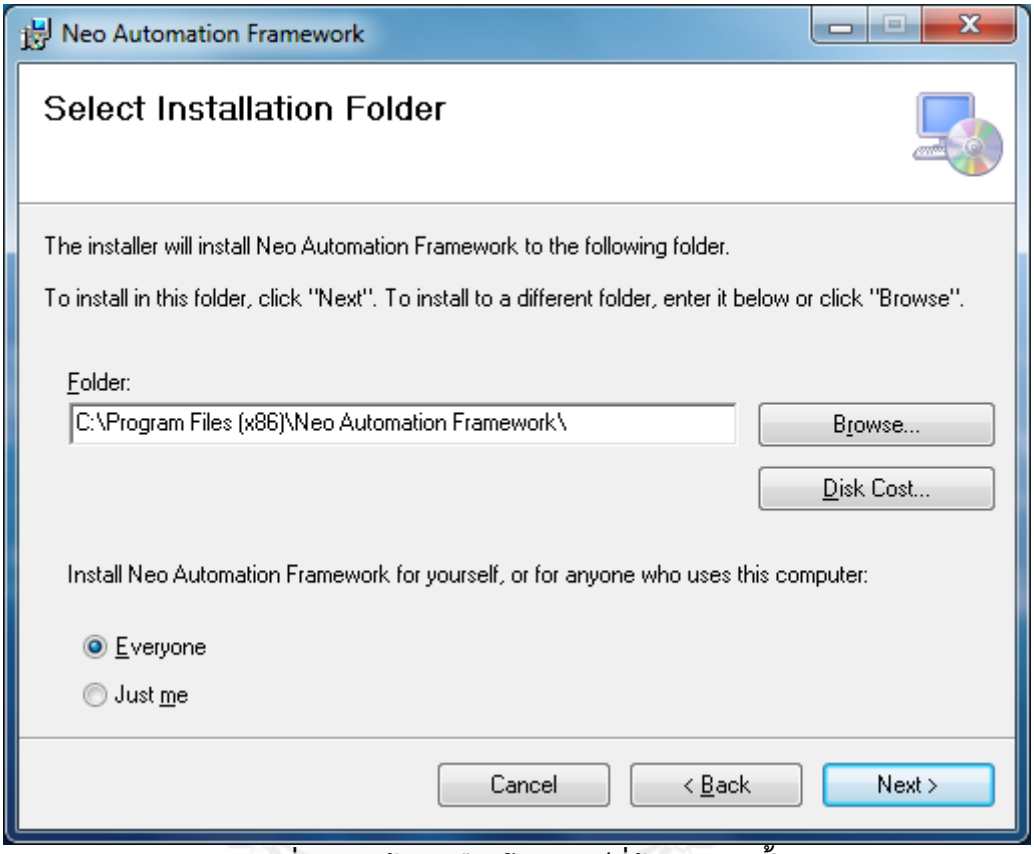
รูปที่ ค-1 ไฟล์สำหรับการติดตั้ง

c. หน้าจอเริ่มต้นการติดตั้งเครื่องมือจะแสดงขึ้นมา กดปุ่ม Next เพื่อดำเนินการต่อไป



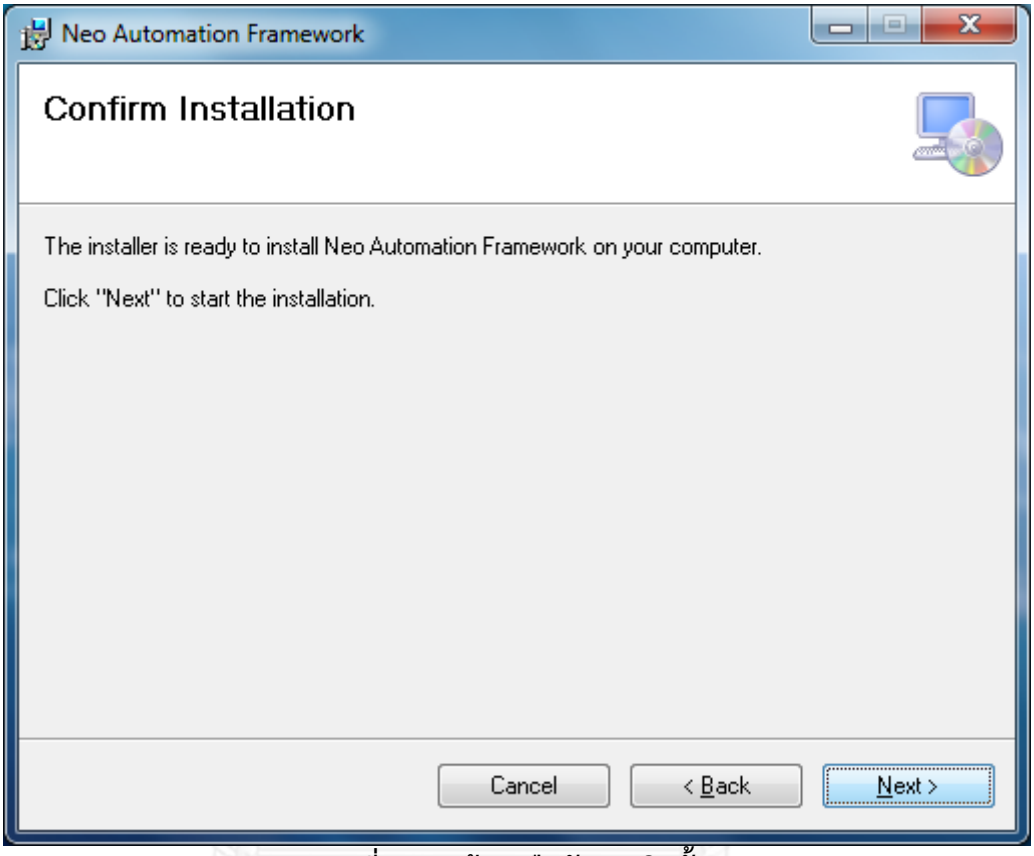
รูปที่ ค-2 หน้าจอเริ่มต้นการติดตั้งเครื่องมือ

d. เลือกโฟลเดอร์ (Folder) ที่ต้องการติดตั้งเครื่องมือ สามารถพิมพ์ตำแหน่งของโฟลเดอร์ลงไปโดยตรงในช่อง หรือกดปุ่ม Browse เพื่อค้นหาโฟลเดอร์ที่ต้องการ และกดปุ่ม Next เพื่อดำเนินการต่อไป



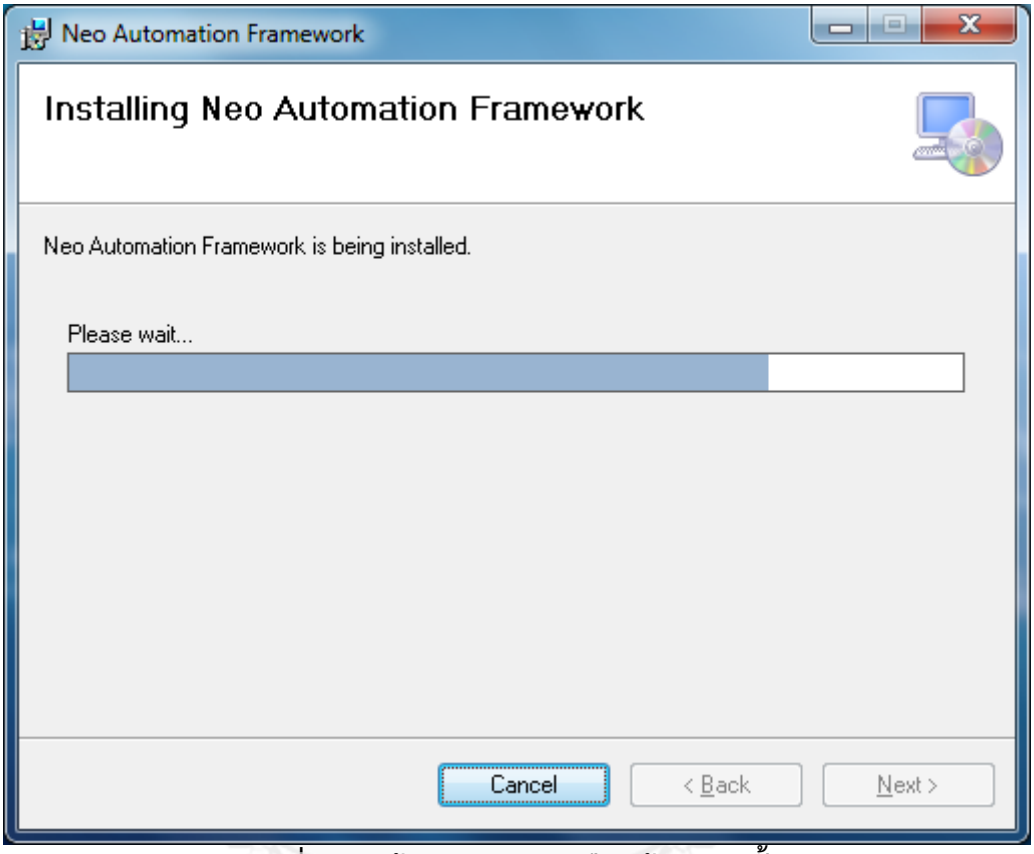
รูปที่ ค-3 หน้าจอเลือกโฟลเดอร์ที่ต้องการติดตั้ง

e. กดปุ่ม Next เพื่อยืนยันการติดตั้ง หรือกดปุ่ม Back เพื่อย้อนกลับไปแก้ไขโฟลเดอร์ที่ต้องการติดตั้ง



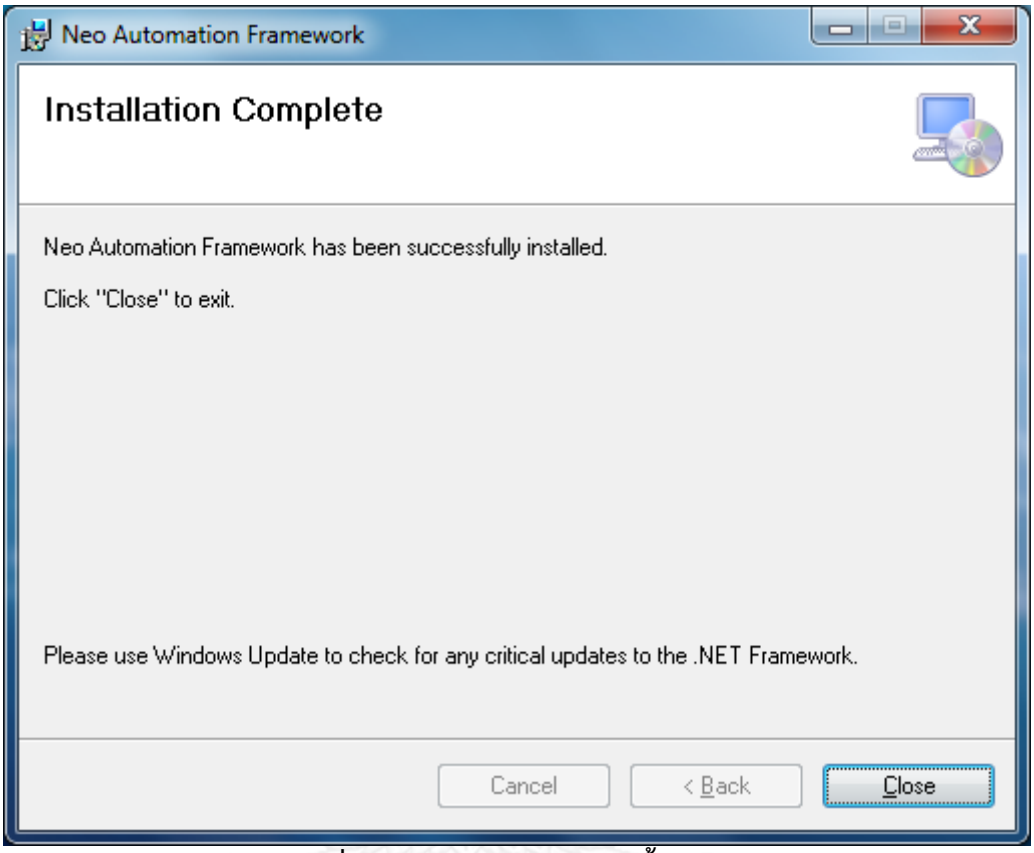
รูปที่ ค-4 หน้าจอยืนยันการติดตั้ง

f. ในระหว่างการติดตั้งเครื่องมือ จะมีแถบแสดงระดับความคืบหน้าของการติดตั้ง รอจนกว่าแถบระดับความคืบหน้าแสดงครบทั้งแถบและปุ่ม Next สามารถกดได้ และให้กดปุ่ม Next เพื่อดำเนินการต่อไป



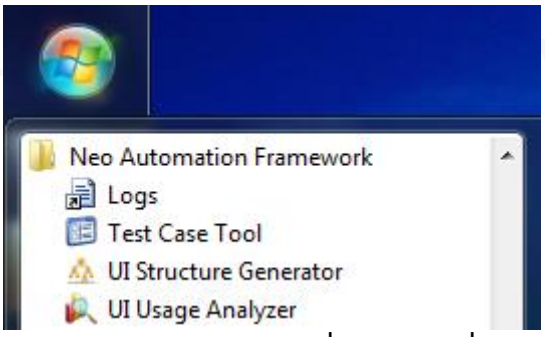
รูปที่ ค-5 หน้าจอแสดงความคืบหน้าการติดตั้ง

g. เมื่อการติดตั้งเสร็จสมบูรณ์ กดปุ่ม Close เพื่อปิดหน้าจอ



รูปที่ ค-6 หน้าจอแสดงการติดตั้งสำเร็จ

h. เมื่อเครื่องมือถูกติดตั้งเสร็จเรียบร้อยแล้ว จะมีชอร์ตคัท (Shortcut) สำหรับเรียกใช้เครื่องมืออยู่ใน Start Menu



รูปที่ ค-7 หน้าจอแสดงชอร์ตคัทเพื่อเรียกใช้เครื่องมือต่างๆ

ประวัติผู้เขียนวิทยานิพนธ์

นายณัฐรัตน์ หาญวรวงศ์ เกิดเมื่อวันที่ 3 ตุลาคม พ.ศ. 2526 ที่จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาระดับปริญญาวิทยาศาสตรบัณฑิต สาขาวิชาศาสตรคอมพิวเตอร์ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์ ในปีการศึกษา 2547 ในระหว่างการศึกษาได้เข้ารับการฝึกงานที่ศูนย์เทคโนโลยีอิเลิกทรอนิกส์และคอมพิวเตอร์แห่งชาติ (NECTEC)

เมื่อจบการศึกษาได้เริ่มทำงานเป็นนักพัฒนาซอฟต์แวร์ที่ บริษัท ซี.เอส.ไอ. (ประเทศไทย) จำกัด จากนั้นได้ย้ายไปเป็นนักทดสอบซอฟต์แวร์ที่ บริษัท ไมโครซอฟท์ (ประเทศไทย) จำกัด ในระหว่างนั้นได้เข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต (ภาคนอกเวลาราชการ) สาขาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ. 2554 และได้ย้ายไปทำงานที่ บริษัท โอไรออน เฮลท์ (ประเทศไทย) จำกัด ในตำแหน่งนักทดสอบซอฟต์แวร์จนถึงปัจจุบัน