ระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์

A WEB-BASED WORKFLOW CONTROL SOFTWARE TESTING

โดย

ศิริประภา วงศักดา

SIRIPRAPA WONGSAKDA

อาจารย์ที่ปรึกษา

ผศ.ดร.พรฤดี เนติโสภากุล

รายงานนี้เป็นส่วนหนึ่งของวิชาการศึกษาอิสระ 2

หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 2 ปีการศึกษา 2554

A WEB-BASED WORKFLOW CONTROL SOFTWARE TESTING

SIRIPRAPA WONGSAKDA

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS OF THE COURSE

INDEPENDENT STUDY 2

MASTER OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2/ 2011

หัวข้อ	ระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์
นักศึกษา	นางสาวศิริประภา วงศักดา
รหัสนักศึกษา	53660562
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
แขนงวิชา	เทคโนโลยีระบบสารสนเทศ
ปีการศึกษา	2554
อาจารย์ที่ปรึกษา	ผศ.ดร.พรฤดี เนติโสภากุล

บทคัดย่อ

เทคโนโลยีด้าน Workflow Management System ถูกนำมาใช้เพื่อจัดการกระบวนการที่ต้องมีการประมวลผลหลายขั้นตอน หรือใช้ใน Application ด้านการจัดการกระบวนงาน หรือกิจกรรมทางธุรกิจขององค์กร เพื่อลดขั้นตอนที่ยุ่งยาก และเพิ่มความสามารถในการควบคุมกระบวนงานให้แม่นยำ

การพัฒนาระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์นี้มีวัตถุประสงค์ในการพัฒนาระบบโดยการวิเคราะห์ออกแบบ และพัฒนาระบบ เพื่อมาประยุกต์ให้เข้ากับกระบวนงานขององค์กร ซึ่งนำระบบมาช่วยในการจัดการกระบวนงานให้มีความเป็นขั้นตอน พนักงานสามารถดูงานที่ได้รับมอบหมาย ปฏิบัติงานตามขั้นตอนที่ได้กำหนดไว้ หัวหน้างานสามารถติดตามความก้าวหน้าของงานได้อย่างมีประสิทธิภาพ

Title	A WEB-BASED WORKFLOW CONTROL SOFTWARE TESTING
Student	Ms. Siriprapa Wongsakda
Student ID.	53660562
Degree	Master of Science
Program	Information Technology
Mojor	Information System Technology
Academic Year	2011
Advisor	Assistant Professor Ponrudee Netisopakul

ABSTRACT

Workflow Management System is used to process that requires several processing steps. The procedure used in Application Management or business activities of the organization. To reduce the complexity of the work and the ability to precisely control procedures.

A Web-Based Workflow Control Software Testing is intended for system analysis, design and development to adapt to the procedures of the organization. This system helps to manage the process with the process. Employees can view work as assigned. The steps defined. Supervisors can monitor the progress of the work effectively.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้เป็นอย่างดีด้วยความกรุณาและสนับสนุนจากท่านอาจารย์ที่ปรึกษา ผศ.ดร.พรฤดี เนติโสภากุล ที่กรุณาเสียสละเวลาอันมีค่าให้คำปรึกษาและคำแนะนำแก่ข้าพเจ้า ช่วยตรวจสอบแก้ไขข้อบกพร่อง ตลอดจนให้ความรู้และข้อคิดเห็นที่เป็นประโยชน์อย่างยิ่ง ข้าพเจ้ารู้สึกซาบซึ้งในความอนุเคราะห์จากท่านอาจารย์ และขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ หน่วยงาน Test Management บริษัท ทรู อินฟอร์เมชั่น เทคโนโลยี จำกัด ที่ช่วยสนับสนุนการทำโครงการ ช่วยให้คำแนะนำแนวทาง และข้อเสนอแนะที่เป็นประโยชน์ยิ่งต่อการทำโครงการ อีกทั้งให้ข้อมูลสำหรับการจัดทำโครงการด้วยดีเสมอมา

ขอขอบคุณเพื่อน ๆ พี่ ๆ น้อง ๆ ที่เป็นกำลังใจและช่วยเหลือในการหาข้อมูลเพิ่มเติม

สุดท้ายขอกราบขอบพระคุณบิดามารดา และบุคคลในครอบครัวที่ให้ความสนับสนุน ความรัก ความห่วงใย และคอยเป็นกำลังใจที่สำคัญที่สุดอย่างดีเสมอมา

ศิริประภา วงศักดา

สารบัญ

หน้า

สารบัญตาราง

ตารางที่ หน้า

สารบัญรูป

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันระบบการมอบหมายงาน และกระบวนการทำงานต่าง ๆ ของหน่วยงาน IT Test Management ของบริษัททรู อินฟอร์เมชั่น เทคโนโลยี จำกัด ยังกระทำด้วยมือ โดยในขั้นตอนต่างๆ ตั้งแต่ขั้นตอนการวางแผนและเตรียมข้อมูลสำหรับการทดสอบระบบต่าง ๆ ขั้นตอนการบันทึกผลการทดสอบ ตลอดจนขั้นตอนการรายงานผลการทดสอบให้กับผู้ที่เกี่ยวข้องทราบข้อมูล ถูกบันทึกไว้ในไฟล์ MS-Word และ MS-Excel ทำให้เกิดปัญหาในการสูญหายของเอกสารและบางครั้งเอกสารไม่ครบตามมาตรฐานที่หน่วยงานได้กำหนดไว้ ปัญหาการจัดสรรงานให้เหมาะสมกับพนักงานตามจำนวนภาระงาน และความยากลำบากในการติดตามความคืบหน้าของงาน ซึ่งไม่สามารถบอกได้ทันทีว่างานนั้นติดอยู่ที่ขั้นตอนใด หรืออยู่ที่พนักงานคนใด

จากปัญหาข้างต้นจึงเป็นสาเหตุให้มีการพัฒนาระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ขึ้น โดยนำหลักการ Workflow management system มาใช้ ซึ่งจะพัฒนาโดยการวิเคราะห์ออกแบบ และพัฒนาระบบ เพื่อมาประยุกต์ให้เข้ากับกระบวนงานขององค์กร ทั้งนี้มีวัตถุประสงค์เพื่อนำระบบมาช่วยในการจัดการกระบวนงานให้มีความเป็นขั้นตอน สามารถติดตามความก้าวหน้าของงานได้

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

จากปัญหาที่เกิดขึ้น ทำให้การควบคุมการทำงานแต่ละขั้นตอนทำได้ยาก เนื่องจากไม่มีระบบที่คอยติดตามผลว่าดำเนินงานแต่ละขั้นตอนมีความก้าวหน้าอยู่ที่ระดับใดแล้ว ดังนั้นจึงพัฒนาระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ เพื่อเป็นอีกแนวทางหนึ่งที่จะช่วยให้ติดตามผลการดำเนินงานในแต่ละขั้นตอนได้อย่างมีประสิทธิภาพ โดยมีวัตถุประสงค์ของการพัฒนาระบบ ดังนี้

1. เพื่อช่วยให้หัวหน้าหน่วยงานมอบหมายงานให้บุคลากร ได้อย่างสะดวกและมีความถูกต้องเพิ่มมากขึ้น

2. เพื่อช่วยให้บุคลากรทั้งหน่วยงานดำเนินงานตาม Workflow ที่กำหนดไว้

3. เพื่อให้บุคคลเกี่ยวข้องกับโครงการ เช่น ผู้บริหารจัดการโครงการ หัวหน้าหน่วยงาน สามารถทราบความก้าวหน้าของโครงการได้สะดวกและรวดเร็วยิ่งขึ้น

4. เพื่อดำเนินงานได้ตรงตามมาตรฐานของหน่วยงาน สามารถควบคุมและตรวจสอบได้

1.3 ทฤษฎีหรือแนวคิดที่ใช้ในการศึกษา

1. Java เป็นภาษาโปรแกรมแบบ Object Oriented Programming ของบริษัท Sun Microsystem ถูกออกแบบมาด้วยแนวคิดที่ว่า Write Once Run Everywhere ซึ่งหมายถึงเขียนเพียงครั้งเดียว แต่สามารถนำไปใช้งานต่อกับระบบอื่น ๆ ได้ เพราะ Java ทำงานอยู่บน Runtime ของ Java โดยเฉพาะ ดังนั้นตัว Java จึงไม่ขึ้นกับระบบปฏิบัติการใด แต่สามารถทำงานได้กับทุก ๆ ระบบ และที่สำคัญคือได้รับความนิยม และมีการนำไปใช้งานอย่างแพร่หลาย นับตั้งแต่มีการเผยแพร่ตัวโปรแกรมภาษานี้ อีกทั้งยังเป็น Open Source และมีประสิทธิภาพที่ดี เนื่องจากมีการพัฒนา และแก้ไขจากผู้พัฒนาทั่วโลกอยู่ตลอดเวลา โดยที่คุณสมบัติเด่น ๆ ของ Java มีดังนั้น

- พัฒนาง่าย, มีลักษณะการออกแบบเชิงวัตถุ, และเป็นภาษาที่มนุษย์ก็อ่านเข้าใจง่าย
- มีความทนทาน และมีความปลอดภัยสูง
- สามารถนำไปใช้งานได้ในหลาย ๆ ระบบ
- มีประสิทธิภาพการทำงานที่สูง
- มีความยืดหยุ่น ในการทำงานสูง

2. J2EE ย่อมาจากคำว่า Java 2 Enterprise Edition คือ กลุ่มของเทคโนโลยีภาษาจาวาที่จำเป็นสำหรับการพัฒนาโปรแกรมประยุกต์ด้วยภาษจาวาให้ใช้งานได้ในระดับองค์กร

1.4 ขอบเขตของการศึกษา

ระบบที่ทำการพัฒนานี้ จะเป็นเครื่องมือที่ช่วยให้การทำงานของบุคลากรในหน่วยงาน IT Test Management สายงานเทคโนโลยีสารสนเทศ สามารถดำเนินงานตาม Workflow ที่ได้ทำการออกแบบไว้ตามแผนงาน เพื่อบรรลุวัตถุประสงค์ของหน่วยงานและองค์กร ซึ่งการทำงานทั้งหมดจะเป็นการใช้ติดต่อกับผู้ใช้ผ่านทางเวบเบราเซอร์ และนำเสนอในรูปแบบของเวบเพจ ระบบจะครอบคลุมการทำงานดังต่อไปนี้

- Test Planning & Preparation
- Test Execution
- Test Result Reporting
- Test Closing

โดยระบบดังกล่าวที่พัฒนาขึ้นภายใต้การทำงานบนเครือข่ายอินทราเน็ตในบริษัทเท่านั้น

1.5 ขั้นตอนของการศึกษา

1.5.1 การวิเคราะห์ระบบ

- ศึกษาการทำงานภายในหน่วยงาน IT Test Management โดยศึกษาลักษณะของการดำเนินงานเกี่ยวกับขั้นตอนการทดสอบซอฟต์แวร์ รวมถึงเอกสารที่จำเป็นในการดำเนินงานนั้น
- สอบถามปัญหา และเก็บรวมรวมความต้องการของพนักงานในหน่วยงานเกี่ยวกับขั้นตอนการทดสอบซอฟต์แวร์
- ศึกษาข้อมูลของการนำเอาระบบสำนักงานอัตโนมัติมาใช้ในการสร้างระบบทั้งทางอินเทอร์เน็ต หนังสือ
- สรุปผลที่ได้ทำการศึกษามาทั้งหมด รวมถึงกำหนดทางเลือกเพื่อการตัดสินใจที่ดีที่สุด
- กำหนดขอบเขตของระบบที่ต้องการจะพัฒนาโดยใช้ข้อมูลจากการศึกษาและวิเคราะห์ความต้องการของผู้ใช้

1.5.2 การออกแบบระบบ

- วิเคราะห์ข้อมูลและสรุปที่มีอยู่ทั้งหมด แล้วทำการออกแบบระบบงานใหม่
- กำหนดเครื่องมือและทรัพยากรที่ใช้ในการพัฒนาระบบ
- ออกแบบการไหลของงานและข้อมูลในระบบ
- ออกแบบฐานข้อมูล
- ออกแบบโครงสร้างของโปรแกรมทั้งหมด คือ ส่วนนำเข้าข้อมูล การแสดงผลข้อมูล และส่วนติดต่อกับผู้ใช้
- การวิเคราะห์และออกแบบระบบงานใหม่จะต้องอยู่ภายในขอบเขตที่กำหนดไว้ โดยจะต้องออกแบบให้เหมาะสมและสอดคล้องกับกระบวนการทำงานภายในหน่วยงานจริง

1.5.3 การพัฒนาและติดตั้งระบบ

- ทำการพัฒนาระบบตามที่ได้วิเคราะห์และออกแบบไว้
- ติดตั้งระบบที่ทำการพัฒนาขึ้น

1.5.4 การทดสอบระบบ

ทำการทดสอบการนำไปใช้งานของตัวระบบว่าสามารถทำงานได้อย่างมีประสิทธิภาพและเป็นไปตามที่ออกแบบไว้หรือไม่ และปรับปรุงเพิ่มเติมในส่วนของระบบที่จะสามารถช่วยเพิ่มศักยภาพการทำงานของระบบให้ดีมากยิ่งขึ้น

1.5.5 การทำเอกสารประกอบระบบ

- ทำเอกสารประกอบการออกแบบระบบ
- ทำเอกสารประกอบการเขียนโปรแกรม

- ทำเอกสารแสดงผลการทดสอบระบบ
- ทำเอกสารประกอบการใช้งานระบบ

1.6 ประโยชน์ที่คาดว่าจะได้รับ

เมื่อพัฒนาระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ขึ้นแล้ว คาดว่าจะสามารถนำระบบนี้ไปใช้ในองค์กรได้ โดยระบบจะช่วยจัดการกระบวนงานต่างๆ ตั้งแต่การมอบหมายงานของหัวหน้าหน่วยงาน จนถึงขั้นตอนการส่งมอบงานที่แล้วเสร็จของพนักงาน การติดตามความก้าวหน้าของงานได้ผ่านเวบไซต์ และการวัดประสิทธิภาพในการทำงานของพนักงาน ซึ่งทำให้เกิดความคล่องตัวในการทำงาน ลดค่าใช้จ่ายในการติดต่อสื่อสารระหว่างพนักงานแต่ละแผนก

บทที่ 2

ทฤษฎีและเทคโนโลยี

2.1 การจัดการกระแสงาน

ระบบการจัดการกระแสงาน คือ ระบบการจัดการอัตโนมัติ สำหรับกระบวนการทางธุรกิจ (business process) เพื่อประหยัดเวลาที่ต้องเสียไปในการทำงาน ช่วยลดค่าใช้จ่ายในกระบวนการทำงานและทำให้การปฏิบัติภารกิจได้ราบรื่น

2.1.1 หน้าที่ของระบบกระแสงาน มีรายละเอียดดังต่อไปนี้

- การส่งเอกสารโดยอัตโนมัติ สามารถส่งข้อความ เอกสาร งาน ไปตามเส้นทางต่างๆ ได้อย่างอัตโนมัติ ตามที่ได้กำหนดไว้ระหว่างการเขียนแผนภูมิกระแสงาน (workflow diagram) ซึ่งผู้ใช้ระบบจะต้องวิเคราะห์ออกแบบกระบวนการทำงาน และกำหนดรายละเอียดต่างๆ ขององค์ประกอบของระบบการจัดการกระแสงาน

- การกำหนดค่าตัวแปร ระบบสามารถกำหนดค่าตัวแปรที่ใช้กำหนดกฎการทำกิจกรรม หรือกฎทางธุรกิจ

- การควบคุมกิจกรรม ระบบสามารถกำหนดควบคุมการทำกิจกรรมว่าได้ถูกกระทำ และรับการยืนยันการสำเร็จของงาน โดยอาจให้มีข้อยกเว้นได้สำหรับกรณีฉุกเฉิน

- การนำเข้าและส่งออก ระบบควรมีความสามารถในการนำแฟ้มเข้า (import) และส่งแฟ้มออก (export) เพื่อเปิดโอกาสให้มีการติดต่อกันกับระบบงานข้างเคียง

- การบันทึกงานแต่ละขั้นตอน ระบบจะช่วยเก็บรายละเอียดของเวลาแต่ละขั้นตอนไว้ทั้งหมด เพื่อนำไปประเมิน พัฒนา และปรับปรุงกระแสงานต่อไป

2.1.2 มาตรฐานของระบบการจัดการกระแสงาน

ในปีค.ศ. 1993 มีการก่อตั้ง Workflow Management Coalition (WfMC) หรือเรียกว่า ดับบลิวเอฟเอ็มซี ซึ่งเกิดจากการรวมตัวของผู้ผลิตซอฟต์แวร์การจัดการกระแสงานหลายรายและนักวิชาการ เพื่อช่วยกันกำหนดมาตรฐานสำหรับการพัฒนาซอฟต์แวร์การจัดการกระแสงาน

โมเดลอ้างอิงกระแสงาน(workflow reference model) การจัดแบ่งระบบในระดับภาพรวม โดยพิจารณาตามหน้าที่การทำงานในระบบการจัดการกระแสงานแบ่งได้ดังนี้

- บิวด์ ไทม์ ฟังก์ชัน (build time function) หมายถึง ซอฟต์แวร์ส่วนที่ทำหน้าที่เกี่ยวกับการสร้าง หรือเขียนแผนภูมิกระแสงาน กำหนดแบบจำลองของกระแสงาน กำหนดวิธีการทำงานในแต่ละขั้นตอน การบันทึกรายละเอียดของนิยามกระบวนการทำงาน

- รัน ไทม์ คอนโทรล ฟังก์ชัน (run time control function) หมายถึง ซอฟต์แวร์ส่วนที่ทำหน้าที่ควบคุมดูแลกระแสงานในขณะปฏิบัติงานอยู่ในแต่ละเหตุการณ์ ซึ่งจะเป็นไปตามกฎเกณฑ์ เงื่อนไขที่กำหนดไว้

- รัน ไทม์ อินเตอร์แอคชัน (run time interaction) หมายถึง ซอฟต์แวร์ส่วนที่ทำหน้าที่ติดต่อโต้ตอบกับผู้ใช้ระบบงาน และติดต่อกับระบบงานประยุกต์ที่ต้องถูกเรียกขึ้นมาทำหน้าที่ในกิจกรรมหนึ่ง

2.1.3 ข้อควรพิจารณาในการประยุกต์การจัดการกระแสงาน

- องค์การควรให้ความสำคัญกับการเตรียมความพร้อมขององค์การ โดยให้การอบรม ชี้แจงถึงประโยชน์ของระบบการจัดการกระแสงานและการปรับรื้อระบบขององค์การ เพื่อลดกระแสการต่อต้านจากผู้ปฏิบัติงาน

- นักวิเคราะห์ระบบ และผู้ใช้งานปลายทางร่วมกันกำหนดรายละเอียดของกระบวนการทำงานแบบใหม่ โดยอาศัยกราฟิกใช้ในการติดต่อกับระบบ ระบบจัดการกระแสงานที่ไม่ผ่านการวิเคราะห์ออกแบบที่ดี จะไม่ได้ประโยชน์สูงสุด

- การเชื่อมต่อเครือข่ายคอมพิวเตอร์เข้าด้วยกันจึงเป็นเรื่องที่จำเป็นอย่างยิ่งในระบบ การจัดการกระแสงาน องค์การควรจัดเตรียมให้มีเครือข่ายที่มั่นคง เพื่อให้การทำงานของระบบ การจัดการกระแสงานดำเนินต่อไปได้อย่างอัตโนมัติ

- ซอฟต์แวร์ระบบการจัดการกระแสงานต้องสามารถเชื่อมต่อระบบการจัดการกระแสงานระบบที่สองได้ การติดตั้งระบบและจัดเตรียมการใช้งานไม่ควรยุ่งยากมากนัก สามารถดูแลบำรุงระบบให้สามารถทำงานได้ตามที่ตั้งเป้าหมายไว้ และสามารถใช้งานได้กับฮาร์ดแวร์ขนาดต่างๆ กัน

2.2 ความรู้เบื้องต้นของการทดสอบซอฟต์แวร์

การทดสอบซอฟแวร์ (Software Testing) เป็นกิจกรรมที่จัดทำขึ้นเพื่อประเมินและปรับปรุงคุณภาพของซอฟแวร์โดยการตรวจหาข้อผิดพลาดและปัญหาที่เกิดขึ้น แล้วทำการแก้ไขข้อผิดพลาดหรือปัญหาดังกล่าวให้ถูกต้อง วัตถุประสงค์ของการทดสอบซอฟแวร์ เพื่อพิสูจน์ว่าซอฟแวร์ทำงานได้ครบทุกฟังก์ชันตามข้อกำหนดความต้องการ และตรวจสอบว่าแต่ละฟังก์ชันสามารถประมวลผลข้อมูลได้อย่างถูกต้อง

การที่จะตรวจสอบหาข้อผิดพลาดของซอฟแวร์ให้ได้มากที่สุดนั้น ขึ้นอยู่กับกรณีทดสอบ (Test Case) ที่ทีมงานออกแบบไว้ ซึ่งไม่ใช่เพียงแค่การรันซอฟแวร์แล้วดูว่ามีข้อผิดพลาดหรือไม่ แต่ทีมงานยังต้องพยายามค้นหาข้อผิดพลาดที่ไม่แสดงให้เห็น ด้วยการทดสอบซอฟแวร์ให้เกิดข้อผิดพลาดให้ได้มากที่สุด จะทำให้สามารถแก้ไขได้ทันท่วงทีก่อนการใช้งานจริงซึ่งนับว่าเป็นการป้องกันปัญหาได้อีกทางหนึ่ง และการที่จะทดสอบให้ซอฟต์แวร์เกิดข้อผิดพลาดได้นั้น จะต้องอาศัยข้อมูลและกรณีทดสอบที่ดีผนวกกับประสบการณ์ของทีมงานทดสอบด้วย จึงได้มีการคิดค้นวิธีในการทดสอบซอฟต์แวร์ขึ้นมา เพื่อช่วยให้ทีมงานค้นหาข้อผิดพลาดให้ได้มากที่สุด

2.2.1 ระดับการทดสอบซอฟต์แวร์

ในแต่ละรอบของการเขียนโปรแกรม สิ่งที่ได้คือ แต่ละส่วนประกอบของซอฟต์แวร์ การทดสอบซอฟต์แวร์จึงสามารถดำเนินการได้เป็นระดับ ตามส่วนประกอบที่ได้ในแต่ละรอบ ซึ่งรวมไปถึงระยะการบำรุงรักษาด้วย เนื่องจากการที่ซอฟต์แวร์จะมีคุณภาพได้นั้น จะต้องเกิดจากแต่ละองค์ประกอบที่ล้วนแต่มีคุณภาพด้วยกันทั้งสิ้น และการที่ทีมงานจะรอทดสอบซอฟต์แวร์เฉพาะช่วงหลังจากการประกอบรวมซอฟต์แวร์เสร็จสิ้นแล้วนั้น เป็นวิธีการที่ทำให้เสียเวลาอย่างมาก และอาจจะไม่สามารถค้นหาข้อผิดพลาดได้อย่างครบถ้วน ทีมงานสามารถทดสอบทุกโปรแกรมย่อยได้ทันทีที่สร้างเสร็จ โดยการทดสอบเป็นอิสระต่อกันในแต่ละโปรแกรมย่อย เพื่อค้นหาข้อผิดพลาดและแก้ไขจนโปรแกรมย่อยนั้นทำงานได้จริงและถูกต้องและทีมงานสามารถทดสอบซ้ำอีกครั้งเมื่อนำโปรแกรมย่อยมารวมกันในแต่ละครั้ง จะทำให้โอกาสในการค้นหาข้อผิดพลาดมีมากขึ้น ซอฟต์แวร์จะมีคุณภาพมากขึ้นด้วย จึงสามารถจำแนกระดับการทดสอบซอฟต์แวร์ออกเป็น 3 ระดับสำคัญ ได้แก่

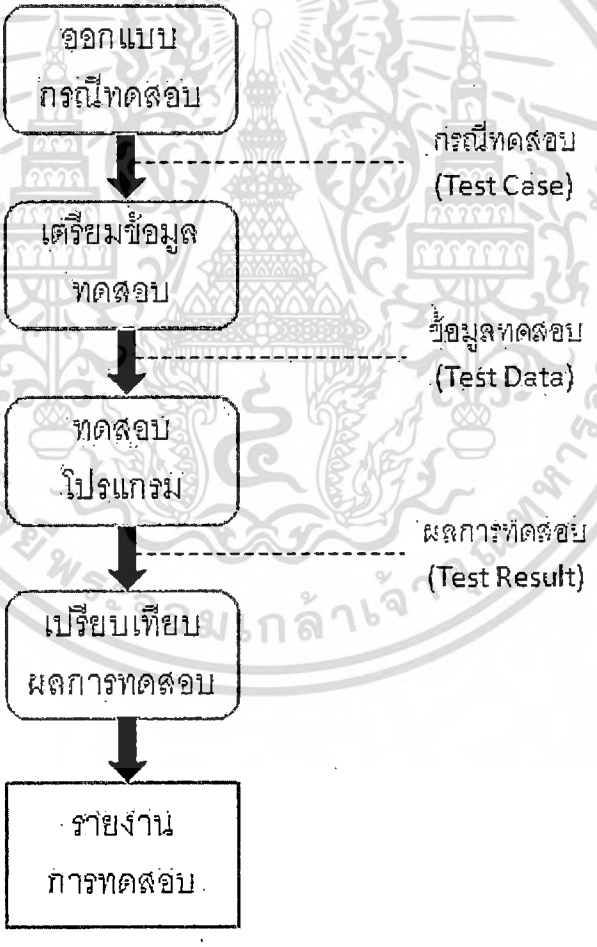
2.2.1.1 การทดสอบระดับหน่วย (Unit Testing) เป็นการทดสอบหน่วยย่อยที่สุดของซอฟต์แวร์ เพื่อประเมินการทำงานในด้านต่าง ๆ ของหน่วยย่อย ไม่ว่าจะเป็นโครงสร้างข้อมูล โครงสร้างควบคุมการทำงาน ตลอดจนการรองรับการทำงานเมื่อเกิดความผิดพลาด

2.2.1.2 การทดสอบระดับรวมหน่วย (Integration Testing) เป็นการทดสอบการทำงานของกลุ่มโปรแกรมหรือส่วนประกอบย่อยที่ถูกประสานเข้าด้วยกันเพื่อทำงานหน้าที่ใดหน้าที่หนึ่งร่วมกัน เพื่อค้นหาข้อผิดพลาดที่อาจเกิดขึ้นได้ ถึงแม้ว่าแต่ละโปรแกรมย่อยจะผ่านการทดสอบมาแล้วก็ตาม

2.2.1.3 การทดสอบระบบ (System Testing) เมื่อการทดสอบซอฟต์แวร์จนครบทุกส่วนประกอบย่อย และทดสอบซอฟต์แวร์ระดับรวมแล้ว จะต้องนำซอฟต์แวร์ดังกล่าวมาทดสอบรวมเข้ากับองค์ประกอบอื่นของระบบ ได้แก่ อุปกรณ์ บุคลากร และข้อมูล

2.2.2 แนวทางการทดสอบซอฟต์แวร์

แนวทางที่เหมาะสมของการทดสอบซอฟต์แวร์ คือ การทดสอบตามรอบของการสร้างซอฟต์แวร์ที่ในแต่ละรอบจะได้ชิ้นงานเพิ่มขึ้นจนกลายเป็นซอฟต์แวร์ที่สมบูรณ์ โดยเริ่มจากทดสอบทีละโมดูล ในรอบต่อไปจะได้โมดูลใหม่ ซึ่งต้องทำการทดสอบก่อน จึงจะสามารถนำไปประสานรวมกับโมดูลก่อนหน้าได้ แล้วจึงทดสอบการทำงานของโมดูลอีกครั้ง เรียกแนวทางดังกล่าวว่า "Incremental Testing Approach" ซึ่งก็คือรูปแบบหนึ่งของกระบวนการผลิตซอฟต์แวร์ จึงกล่าวได้ว่าการทดสอบซอฟต์แวร์นั้นดำเนินการไปตามแต่ละระยะของการผลิตซอฟต์แวร์ สำหรับขั้นตอนการทดสอบซอฟต์แวร์ในแต่ละระดับ มีดังนี้



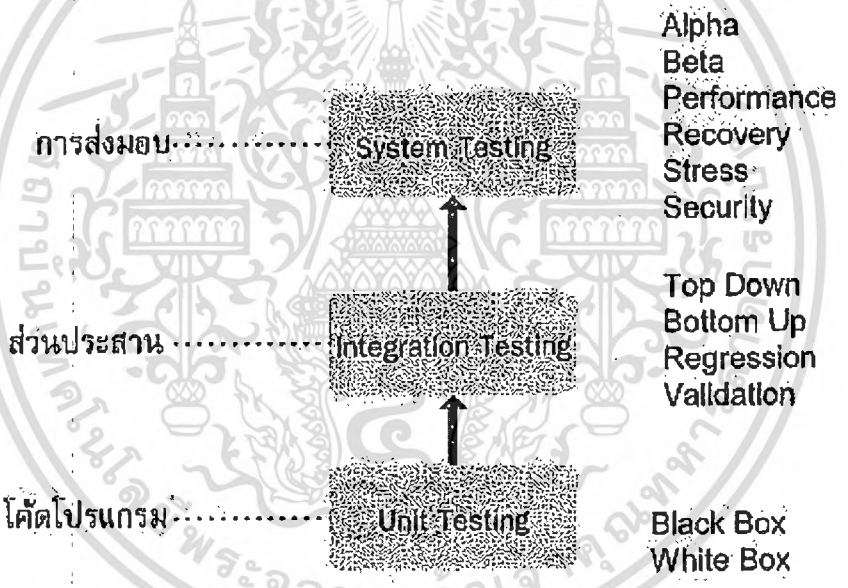
รูปที่ 2.1 แสดงขั้นตอนการทดสอบซอฟต์แวร์

การทดสอบซอฟต์แวร์เริ่มต้นโดยทีมงานต้องออกแบบ "กรณีทดสอบ (Test Case)" ซึ่งหมายถึงข้อกำหนดของข้อมูลนำเข้าและผลลัพธ์ที่ต้องการจากการทำงานของโปรแกรมในสถานการณ์

สมมติต่าง ๆ จากนั้น ทีมงานจะนำกรณีทดสอบที่ออกแบบไว้มากำหนดชุดข้อมูลทดสอบ (Test Data) ให้สอดคล้องกัน เพื่อนำไปใช้ในการประมวลผลของโปรแกรม ขั้นต่อมา เริ่มทำการทดสอบโดยการรันโปรแกรม แล้วนำผลลัพธ์ที่ได้ไปเปรียบเทียบกับกรณีทดสอบที่ได้ออกแบบไว้ และสรุปเป็นรายงานผลการทดสอบ สิ่งสำคัญของขั้นตอนการทดสอบคือ "กรณีทดสอบ" จะต้องมีประสิทธิภาพมากพอที่จะทำให้ทีมงานค้นพบข้อผิดพลาด

2.2.3 วิธีการทดสอบซอฟต์แวร์

วิธีการทดสอบซอฟต์แวร์ในแต่ละระดับจะมีจุดมุ่งเน้นที่แตกต่างกัน โดยการทดสอบระดับหน่อยส่วนใหญ่จะมุ่งเน้นที่โค้ดโปรแกรม และเมื่อทดสอบในระดับรวมหน่วยก็จะมุ่งเน้นที่ส่วนประสานการทำงานระหว่างโมดูลว่าเกิดปัญหาขึ้นหรือไม่ หรือทำงานได้ถูกต้องหรือไม่ สุดท้ายคือการทดสอบระบบ เป็นการทดสอบเพื่อการส่งมอบระบบให้กับลูกค้า อย่างไรก็ตาม ทั้ง 3 ระดับต่างมีความสัมพันธ์กัน สำหรับวิธีที่ทีมงานสามารถเลือกใช้ทดสอบซอฟต์แวร์แต่ละระดับ แสดงดังรูป



รูปที่ 2.2 แสดงวิธีที่ใช้ในการทดสอบซอฟต์แวร์แต่ละระดับ

2.2.4 กรณีทดสอบและการวางแผนการทดสอบ

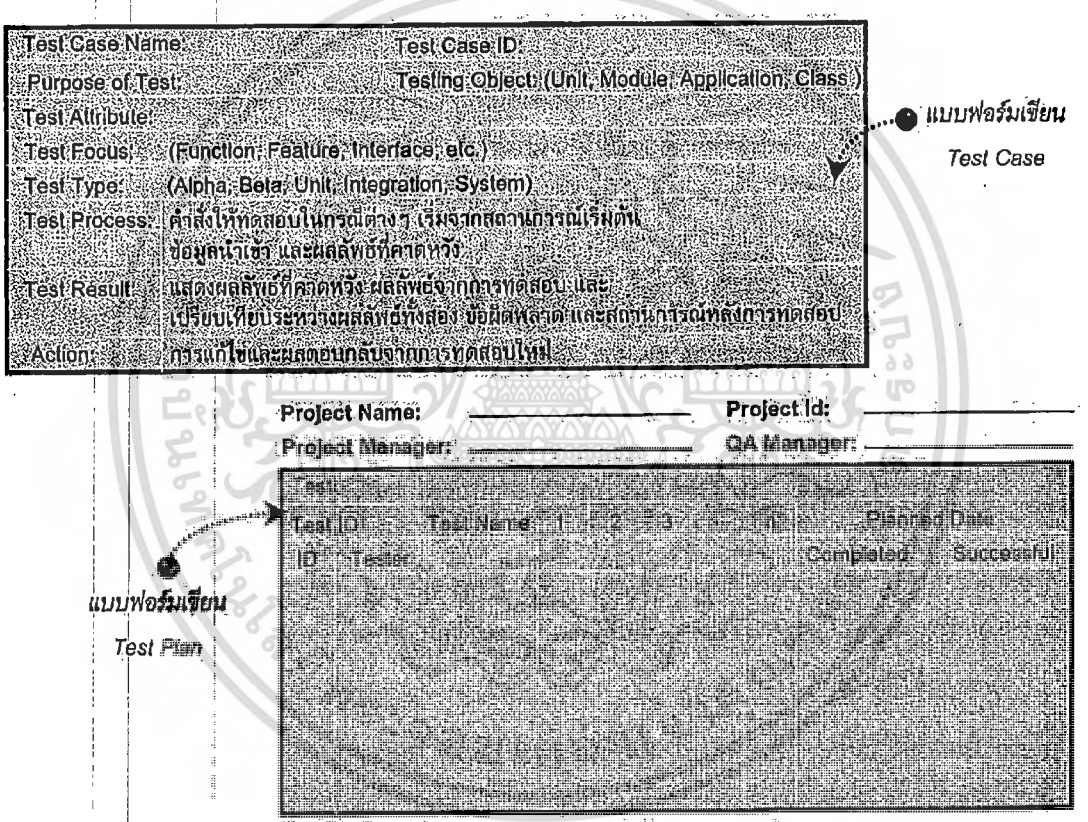
สิ่งที่สำคัญที่สุดคือ "กรณีทดสอบ (Test Case)" ที่ทีมงานมีหน้าที่ออกแบบกรณีทดสอบให้เหมาะสมกับวิธีการทดสอบซอฟต์แวร์ที่เลือกใช้ และต้องเป็นกรณีทดสอบที่สามารถเปิดเผยความผิดพลาดหรือข้อบกพร่องของซอฟต์แวร์ให้ได้มากที่สุด

การออกแบบกรณีทดสอบ คือ การกำหนดชุดข้อมูลนำเข้า (Input) และผลลัพธ์ที่คาดหวัง (Output) โดยมีเป้าหมายเพื่อค้นพบข้อผิดพลาดและข้อบกพร่องของซอฟต์แวร์ให้ได้มากที่สุด และแสดงให้เห็นว่ามีส่วนใดบ้างที่ตรงตามความต้องการของผู้ใช้ การออกแบบกรณีทดสอบ เป็นเพียง

กิจกรรมหนึ่งในกระบวนการทดสอบซอฟต์แวร์ทั้งหมด ดังนั้น ทีมงานจำเป็นต้องมีการวางแผนการทำงานในทุก ๆ ขั้นตอนของการทดสอบ

การวางแผนการทดสอบ (Test Planning) เป็นการกำหนดรายละเอียดการทำงานในแต่ละขั้นตอนของกระบวนการทดสอบ สิ่งที่ได้จากการวางแผนคือ "เอกสารแผนการทดสอบ (Test Plan) ที่จะช่วยให้ทีมงานออกแบบและจัดการทดสอบได้อย่างเป็นระเบียบและมีแบบแผน

แผนการทดสอบ (Test Plan) คือ ชุดเอกสารที่ประกอบไปด้วยชุดข้อมูลนำเข้าของแต่ละเส้นทางการทำงานของทุกโปรแกรม และผลลัพธ์ของการทดสอบของแต่ละเส้นทางนั้น ชุดข้อมูลจะถูกจัดเรียงกันไว้ตามลำดับเส้นทาง และชนิดของการทดสอบ นอกจากนี้ ในแผนการทดสอบอาจประกอบไปด้วยข้อมลอื่น ๆ ที่จำเป็นด้วย



รูปที่ 2.3 แสดงตัวอย่างแบบฟอร์มเอกสารสำหรับ Test Case และ Test Plan

2.3 พื้นฐานการทดสอบซอฟต์แวร์

- ความสามารถในการทดสอบได้ (Testability) การที่โปรแกรมคอมพิวเตอร์สามารถถูกทดสอบได้โดยง่าย ลักษณะต่อไปนี้นำไปสู่ซอฟต์แวร์ที่ทดสอบได้

- ความสามารถในการทำงาน (Operability) "ยิ่งทำงานได้ดีเท่าไร ก็ยิ่งง่ายต่อการทดสอบเท่านั้น" ถ้าระบบได้รับการออกแบบและสร้างมาโดยยึดคุณภาพ จุดบกพร่องที่เป็นอุปสรรคต่อการทดสอบจะมีค่อนข้างน้อย ซึ่งทำให้การทดสอบมีความก้าวหน้าไปโดยไม่มีอุปสรรค

- ความสามารถในการสังเกตเห็น (Observability) "สิ่งที่เห็นก็คือสิ่งที่ทดสอบได้" ข้อมูลเข้าซึ่งเป็นส่วนหนึ่งของการทดสอบจะผลิตตัวเอาท์พุตต่าง ๆ สถานะของระบบและตัวแปรสามารถมองเห็นได้ หรือสอบถามค่าได้ระหว่างการลงมือทำงาน เอาท์พุตที่ไม่ถูกต้องจะง่ายต่อการค้นพบ ความผิดพลาดภายในจะถูกค้นพบและรายงานโดยอัตโนมัติ ซอร์สโค้ดจะเข้าถึงได้ง่าย

- ความสามารถในการควบคุม (Controllability) "ยิ่งเราสามารถควบคุมซอฟต์แวร์ได้ดีเท่าไร เรายิ่งสามารถทดสอบโดยอัตโนมัติและเหมาะสมได้มากเท่านั้น" สถานะและตัวแปรของซอฟต์แวร์และฮาร์ดแวร์ ควรสามารถควบคุมโดยตรงจากนักวิศวกรซอฟต์แวร์ การทดสอบต้องสามารถระบุเจาะจงได้ ทำโดยอัตโนมัติได้ และทำซ้ำได้โดยสะดวก

- ความสามารถในการแบ่งเป็นส่วนย่อย (Decomposability) "การควบคุมขอบเขตของการทดสอบ ทำให้สามารถแยกแยะปัญหาและทำการทดสอบได้อย่างรวดเร็ว" เนื่องจากระบบซอฟต์แวร์สร้างจากโมดูลอิสระที่สามารถทดสอบอย่างอิสระได้

- ความเรียบง่าย (Simplicity) "ยิ่งมีสิ่งที่ต้องทดสอบน้อยเท่าไร ยิ่งทดสอบได้เร็วเท่านั้น" โปรแกรมควรจะแสดงความเรียบง่ายด้านหน้าที่ ด้านโครงสร้างและด้านโค้ด คือมีชุดของลักษณะหน้าที่น้อยเท่าที่จำเป็นจะสนองตอบต่อความต้องการ มีสถาปัตยกรรมแบบโมดูลที่จำกัดการแพร่กระจายของความล้มเหลว และมีมาตรฐานของโปรแกรมที่ง่ายต่อการตรวจสอบและบำรุงรักษา

- ความคงตัว (Stability) "การเปลี่ยนแปลงยิ่งน้อย ความขัดแย้งในการทดสอบก็ยิ่งน้อยตาม" การเปลี่ยนแปลงในซอฟต์แวร์ควรจะเกิดไม่บ่อยนัก จึงต้องมีการควบคุมเมื่อมีการเปลี่ยนแปลงเกิดขึ้น และไม่ทำให้ชุดทดสอบเดิมใช้งานไม่ได้ ซอฟต์แวร์ควรจะคืนสภาพเดิมได้ดีจากความล้มเหลว

- ความสามารถในการเข้าใจ (Understandability) "ยิ่งรู้มากเท่าไร เราก็ยิ่งทดสอบอย่างฉลาดได้มากเท่านั้น" ความเข้าใจในการออกแบบสถาปัตยกรรม และการขึ้นแก่กันระหว่างองค์ประกอบภายในและภายนอกที่ใช้ร่วมกัน เอกสารทางเทคนิคที่เข้าถึงได้ง่าย จัดเรียงมีระเบียบ มีรายละเอียดตรงตามความจริง การเปลี่ยนแปลงการออกแบบที่สามารถสื่อสารกับผู้ทดสอบ

- ลักษณะการทดสอบ (Test Characteristics)

1. ตัวทดสอบที่ดีมีความน่าจะเป็นสูงในการหาความผิดพลาดพบ การที่นักทดสอบจะบรรลุเป้าหมายนากรออกแบบตัวทดสอบที่ดีได้ นักทดสอบจำเป็นต้องมีความเข้าใจในตัวซอฟต์แวร์เป็นอย่างดี และพยายามสร้างสถานการณ์ที่ทำให้ซอฟต์แวร์ล้มเหลว ตัวอย่างเช่น ในการทดสอบตัว

ติดต่อผู้ใช้เชิงกราฟฟิก ความผิดพลาดแบบหนึ่ง คือ การที่ซอฟต์แวร์ทำงานผิดพลาดกับการรู้ค่าตำแหน่งเมาส์ ดังนั้น ชุดการทดสอบอาจจะออกแบบให้ทำงานกับเมาส์ในลักษณะที่พยายามหาข้อผิดพลาดในการรู้จักตำแหน่งเมาส์

2. ตัวทดสอบที่ดีไม่ซ้ำซ้อน ทรัพยากรและเวลาในการทดสอบมีจำกัด จึงไม่มีประโยชน์ที่จะใช้ตัวทดสอบที่มีเป้าหมายอย่างเดียวกับตัวทดสอบอื่น ๆ นั่นคือ ทุก ๆ ตัวทดสอบควรมีวัตถุประสงค์แตกต่างกัน

3. ตัวทดสอบที่ดีควรเป็นตัวที่คิดมาดีที่สุดในกลุ่ม ในบรรดาตัวทดสอบที่มีวัตถุประสงค์คล้าย ๆ กัน ผู้ทดสอบควรคัดสรรชุดทดสอบย่อยเพื่อประหยัดเวลาและทรัพยากร ตัวทดสอบที่ถูกคัดสรรควรมีความเป็นไปได้สูงที่จะเปิดเผยความผิดพลาดทั้งหมดในกลุ่มนี้

4. ตัวทดสอบที่ดีไม่ควรเป็นทั้งธรรมดาเกินไปหรือซับซ้อนเกินไป การประสมประสานชุดของการทดสอบหลาย ๆ ชุดเข้าด้วยกันเป็นหนึ่งกรณีทดสอบ อาจก่อให้เกิดผลข้างเคียงที่บดบังความผิดพลาดได้ โดยทั่วไปแล้ว ตัวทดสอบแต่ละตัวควรถูกใช้งานแยกกัน

บทที่ 3

การวิเคราะห์และออกแบบระบบ

3.1 วิเคราะห์ระบบงานปัจจุบัน

จากการศึกษาระบบงานปัจจุบัน หน่วยงาน IT Test Management คือ หน่วยงานที่ดำเนินการทำการทดสอบซอฟต์แวร์ร่วมกับผู้ใช้งานในขั้นตอนสุดท้ายของการพัฒนาซอฟต์แวร์ User Acceptance Testing (UAT) ก่อนที่จะประกาศเปิดใช้งานจริงบนระบบ Production เพื่อให้ผู้ใช้มั่นใจได้ว่าระบบที่จะนำไปใช้จริงนั้นมีประสิทธิภาพและมีความถูกต้อง ตรงกับความต้องการ

ขั้นตอนการทำงานของระบบงานปัจจุบัน โดยเริ่มต้นจากวงจรชีวิตของการพัฒนาซอฟต์แวร์ (System Development Life Cycle : SDLC) ซึ่งประกอบไปด้วยขั้นตอนต่างๆ หลายๆ ส่วนมาประกอบกัน แต่ละโครงการจะมีรายละเอียดปลีกย่อยแตกต่างกันไปตามขนาด หรือความซับซ้อนของโครงการ วงจรการพัฒนาระบบ หรือ SDLC ประกอบไปด้วย

1. การกำหนดปัญหา(Problem Definition)

2. การวิเคราะห์ปัญหา(Analysis)

3. การออกแบบ(Design)

4. การพัฒนาระบบงาน(Development)

5. การทดสอบ(Testing)

6. การติดตั้ง (Deployment)

7. การบำรุงรักษา(Maintenance)

สำหรับหน่วยงาน IT Test Management จะมีหน้าที่รับผิดชอบในส่วนของการทดสอบ (Testing) ซึ่งมุ่งไปไปที่ User Acceptance Testing ในแผนภาพขั้นตอนการทำงานของระบบที่แสดงดังรูปที่ 3.2 สามารถอธิบายขั้นตอนการทำงานของผู้ที่เกี่ยวข้องกับระบบงานปัจจุบันได้ดังนี้

1. ผู้บริหารจัดการโครงการ (Project Manager) ที่มีหน้าที่ในการดำเนินการเริ่มต้นโครงการและจัด The Kick-Off Meeting เริ่มทำงานในโครงการอย่างเป็นทางการ เพื่อสื่อสารกับผู้เกี่ยวข้อง

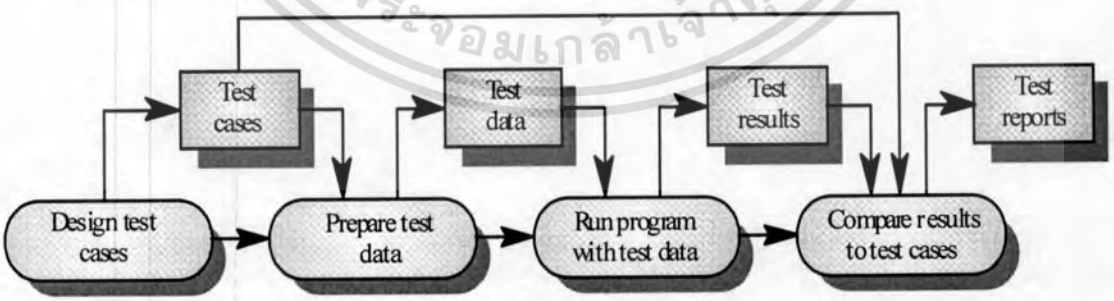
ว่าโครงการคืออะไร เกี่ยวข้องกับหน่วยงานใดบ้าง และเกี่ยวกับอะไร จากนั้นจะดำเนินงานตามแผนในช่วง Project Executing จนถึงกระบวนการปิดโครงการ

2. นักวิเคราะห์ระบบงาน (Business Analyst) เป็นผู้รวบรวมความต้องการของซอฟต์แวร์นำมาวิเคราะห์และออกแบบระบบ จัดประชุม Walkthrough requirement เพื่อระบุถึงขอบเขต , ข้อกำหนด / เงื่อนไข , ข้อจำกัดของระบบ รวมถึงการกำหนดเป้าหมายหลักที่ต้องการให้บุคลากรที่เกี่ยวข้อทราบ

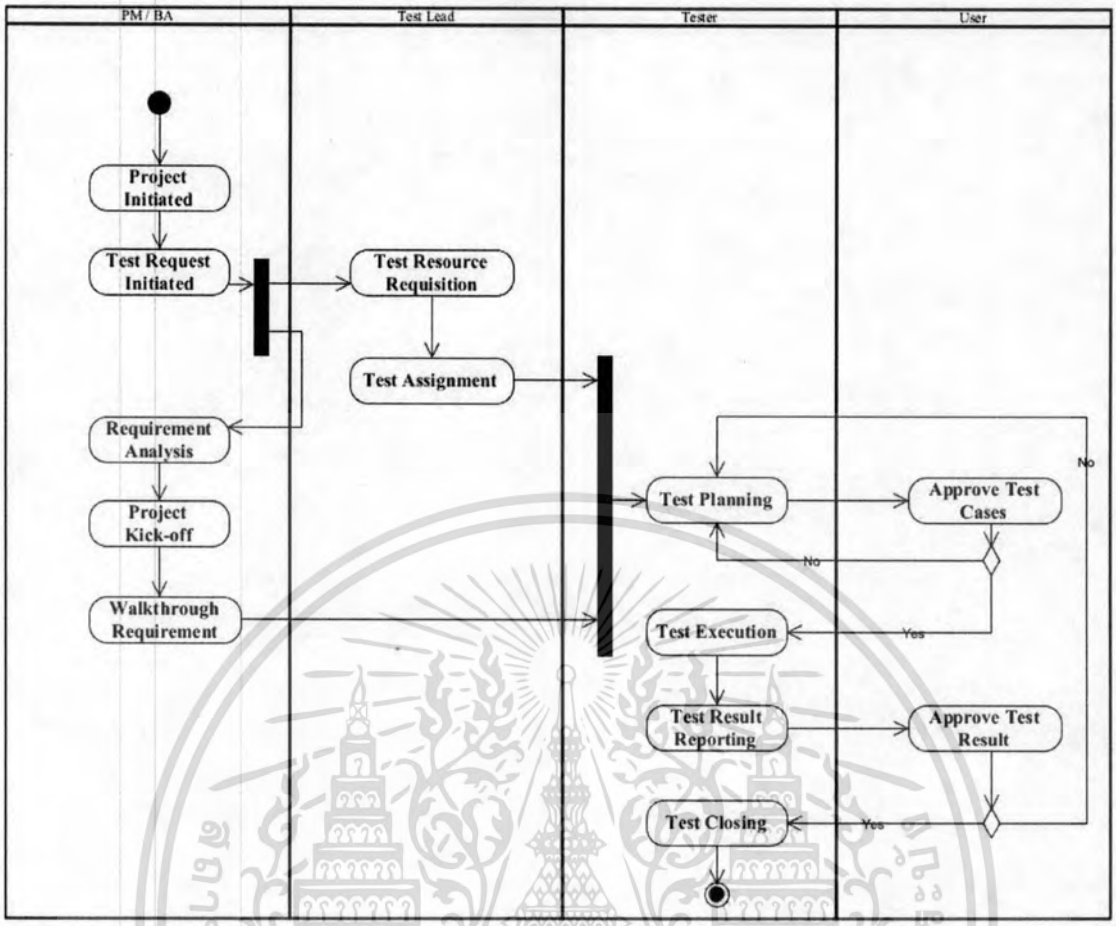
3. ผู้จัดการหน่วยงาน IT Test Management มีหน้าที่บริหารจัดการส่วนต่าง ๆ ในหน่วยงาน ตั้งแต่คัดกรองโครงการที่ส่งมาทำการทดสอบ มอบหมายงานให้กับพนักงาน คอยติดตามความก้าวหน้าของงานรวมทั้งดูผลการทำงานของพนักงาน ซึ่งต้องกระจายงานไปยังพนักงานในหน่วยงานโดยการมอบหมายงานจะดูจากทักษะ ความถนัด ความรู้และความสามารถ ภาระงานของพนักงานแต่ละคน

3. นักทดสอบซอฟต์แวร์ (Software Tester) เป็นผู้ทดสอบซอฟต์แวร์ ให้เป็นไปตามความต้องการ และข้อกำหนดตามแบบ. มีหน้าที่ปฏิบัติงานที่ได้รับมอบหมาย และดำเนินงานตามกระบวนการต่าง ๆ ที่หน่วยงานกำหนดไว้ในแต่ละขั้นตอน คือ Test Planning & preparation , Test Execution , Test Result Reporting และ Test Closing

4. ผู้ใช้งานระบบ (User) เป็นผู้ใช้งานระบบและร่วมทดสอบซอฟต์แวร์ที่พัฒนา เพื่อเป็นการทวนสอบให้มั่นใจว่าซอฟต์แวร์นั้น ๆ ตรงกับความต้องการที่ได้ระบุไว้ใน Requirement Specification Document ซึ่งผู้ใช้งานระบบต้องอนุมัติ Test Case ก่อนเริ่มทำการทดสอบ และอนุมัติผลการทดสอบหลังจากที่ได้ทดสอบครบทุก Test Case



รูปที่ 3.1 แสดงแผนภาพการทำงานของนักทดสอบซอฟต์แวร์



รูปที่ 3.2 แสดงแผนภาพขั้นตอนการทำงานของผู้ที่เกี่ยวข้องในระบบงานปัจจุบัน

สำหรับระบบการทำงานปัจจุบัน ของหน่วยงาน Test Management ในส่วนงานของผู้ทดสอบระบบจะทำการสร้างและบันทึกกรณีทดสอบลงในเอกสาร Excel โดยมีรูปแบบดังรูปที่ 3.3

Test Case/Test Script Report							
Project Name/UCR#				Prepared By			
Test Level				Created Date			
Test Environment				Reference Document			
Test Period				Approved By			
Case#	Test Case	Test Step/Scenario	Expected Result	Source of Requirement	Actual Result	Test By	Remark
Ivan							

รูปที่ 3.3 เอกสารสำหรับสร้างและบันทึกกรณีทดสอบสำหรับโครงการต่าง ๆ

3.2 ปัญหาที่พบได้จากการวิเคราะห์ระบบงานปัจจุบัน

1. ความยากลำบากในการติดตามความก้าวหน้าของแต่ละโครงการ เนื่องจากปัจจุบันจะทำการส่งเอกสาร Test Result Reporting ทาง E-Mail ให้ผู้ที่เกี่ยวข้องให้โครงการทราบ ทำให้ผู้ที่ต้องการทราบถึงความก้าวหน้าต้องรอรับการรายงานผลการทำสอบจากทาง E-Mail เท่านั้น

2. การจัดเก็บข้อมูลของระบบงานปัจจุบันไม่สามารถนำมาใช้ประโยชน์ได้อย่างรวดเร็ว เนื่องจากไม่ได้เป็นฐานข้อมูล ซึ่งจะนำมาวิเคราะห์และใช้ประโยชน์โดยรวมได้

3. ไม่สามารถตรวจสอบได้ว่าโครงการนั้นจะได้รับการดำเนินการทดสอบจนสิ้นสุดเสร็จทันเวลาหรือไม่

4. หัวหน้าหน่วยงานไม่มีระบบที่ช่วยจัดเก็บข้อมูลการจัดสรรโครงการให้กับพนักงาน

5. พนักงานที่เป็นนักทดสอบซอฟต์แวร์จัดทำเอกสารที่ใช้ในแต่ละขั้นตอนของกระบวนของทดสอบซอฟต์แวร์ไม่ครบถ้วน

6. ต้องสิ้นเปลืองเวลาในการค้นหาเอกสารหากต้องการจะนำมาตรวจสอบภายหลังจากสิ้นสุดโครงการไปแล้ว

3.3 ขอบเขตระบบงานใหม่

จากที่ได้ศึกษาระบบงานที่ใช้อยู่ในปัจจุบัน ได้รวบรวมปัญหาที่เกิดขึ้นจากการทำงาน และได้ทำการวิเคราะห์ระบบงานโดยละเอียดแล้ว จึงออกแบบระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์โดยการออกแบบจะเน้นที่ความต้องการของผู้ใช้เป็นหลัก โดยเริ่มต้นจากการศึกษาความต้องการและขอบเขตของระบบงานใหม่ คุณสมบัติของระบบงาน ส่วนประกอบของระบบงาน โดยจะแสดงรายละเอียดการทำงานของระบบงานใหม่ด้วย Use Case Diagram เพื่อแสดงการทำงานของผู้ใช้ระบบ (User) และความสัมพันธ์กับระบบย่อย (Sub systems) ภายในระบบใหญ่ จึงสามารถวิเคราะห์และออกแบบฟังก์ชันการทำงานที่ประกอบไปด้วย

ส่วนที่ 1: Security System

1. การเข้าสู่ระบบ

2. การออกจากระบบ

ส่วนที่ 2: Management System

1. การจอง resource ล่วงหน้าเพื่อทดโครงการนั้น ๆ

2. การยกเลิกการจอง resource

3. การสร้างกรณีทดสอบสำหรับโครงการที่ส่งมาทดสอบ

4. การแก้ไขกรณีทดสอบสำหรับโครงการที่ส่งมาทดสอบ

5. การยกเลิกกรณีทดสอบสำหรับโครงการที่ส่งมาทดสอบ

6. การรายงานสรุปผลการทดสอบ

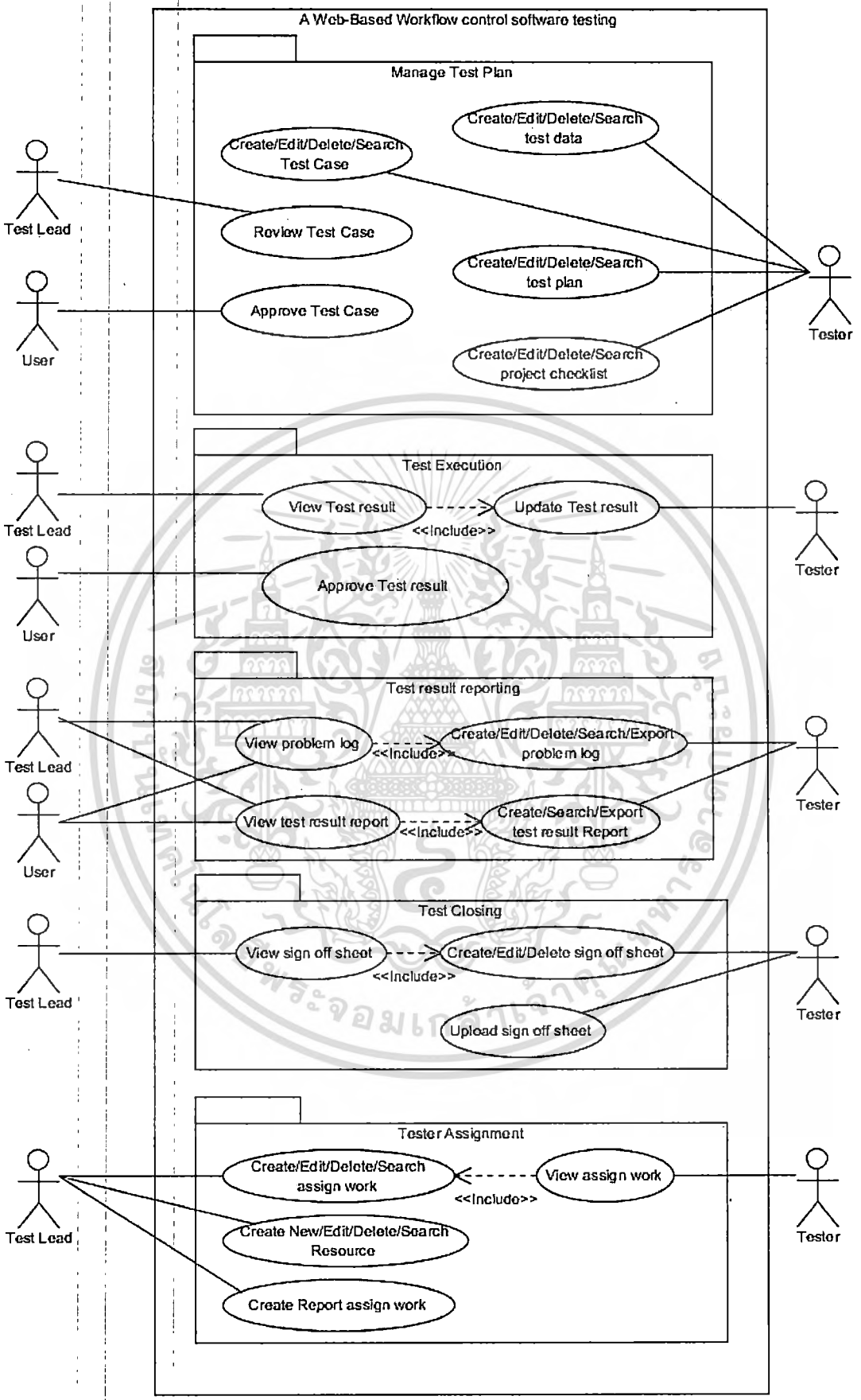
ส่วนที่ 3: Data Management System

1. การเพิ่มชื่อพนักงาน

2. การแก้ไขข้อมูลพนักงาน

3.3.1 ยูสเคสไดอะแกรม

ยูสเคสไดอะแกรม คือ แผนภาพที่แสดงฟังก์ชันการทำงานของระบบที่จะพัฒนาขึ้น เป็นการอธิบายภาพรวมของระบบ โดยแสดงด้วยยูสเคสไดอะแกรมดังรูปที่ 3.4 และแสดงรายละเอียดของยูสเคสไดอะแกรม ของระบบระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ ดังตารางที่ 3.1 – ตารางที่ 3.20



รูปที่ 3.4 Use Case Diagram ของระบบระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์

จากรูปแสดง Use Case Diagram ของระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ โดยมีActor ในระบบ แบ่งออกตามหน้าที่ความรับผิดชอบดังนี้

- User : มีหน้าที่ในอนุมัติ Test Case , อนุมัติผลการทดสอบ , ดูรายงานผลการทดสอบ , ตรวจดูปัญหาที่เกิดขึ้นจากการทดสอบ
- Tester มีหน้าที่จัดเตรียม Test Plan , Test Case , Test Data , Project Checklist , บันทึกผลการทดสอบ, บันทึกปัญหาที่เกิดขึ้นจากการทดสอบ , รายงานผลการทดสอบ , จัดทำ UAT Sign-off sheet และรับทราบแผนปฏิบัติงานจากการ Assign Work
- Test Lead มีหน้าที่ในการ Review Test Case , ดูรายงานผลการทดสอบ ตรวจดูปัญหาที่เกิดขึ้นจากการทดสอบ และบริหารจัดการ Test Resources

ตารางที่ 3.1 Use Case Description ของส่วน Create/Edit/Delete/Search test plan

<table>
<tr><td>Use-case name</td><td colspan="2">1. Create/Edit/Delete/Search test plan</td></tr>
<tr><td>Scenario</td><td colspan="2">การสร้าง แก้ไข ลบ ค้นหา Test Plan</td></tr>
<tr><td>Brief Description</td><td colspan="2">สามารถสร้าง แก้ไข ลบ ค้นหา Test Plan ผ่านระบบ</td></tr>
<tr><td>Actor</td><td colspan="2">Tester</td></tr>
<tr><td>Related Use-cases</td><td colspan="2">-</td></tr>
<tr><td>Preconditions</td><td colspan="2">เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Planning</td></tr>
<tr><td>Postconditions</td><td colspan="2">ดูรายละเอียดต่าง ๆ ของ Test Plan ของแต่ละโครงการได้</td></tr>
<tr><td rowspan="2">Flow of Events</td><td>Actor</td><td>System</td></tr>
<tr><td>1.Tester เข้าสู่ระบบ
2. เลือกหน้า Test Planning > Test Plan
3. ทำการ Create/Edit/Delete/Search test plan</td><td>1. ตรวจสอบรหัสผู้ใช้
2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search test plan
3. ระบบทำการบันทึกข้อมูล</td></tr>
<tr><td>Exception Conditions</td><td colspan="2">-</td></tr>
</table>

ตารางที่ 3.2 Use Case Description ของส่วน Create/Edit/Delete/Search Test Case

<table>
<tr><td>Use-case name</td><td colspan="2">2. Create/Edit/Delete/Search Test Case</td></tr>
<tr><td>Scenario</td><td colspan="2">การสร้าง แก้ไข ลบ ค้นหา Test Case</td></tr>
<tr><td>Brief Description</td><td colspan="2">สามารถสร้าง แก้ไข ลบ ค้นหา Test Case ผ่านระบบ</td></tr>
<tr><td>Actor</td><td colspan="2">Tester</td></tr>
<tr><td>Related Use-cases</td><td colspan="2">-</td></tr>
<tr><td>Preconditions</td><td colspan="2">เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Planning</td></tr>
<tr><td>Postconditions</td><td colspan="2">ดูรายละเอียดต่าง ๆ ของ Test Case ของแต่ละโครงการได้</td></tr>
<tr><td rowspan="2">Flow of Events</td><td>Actor</td><td>System</td></tr>
<tr><td>1.Tester เข้าสู่ระบบ
2. เลือกหน้า Test Planning>Test Case
3. ทำการ Create/Edit/Delete/Search Test case</td><td>1. ตรวจสอบรหัสผู้ใช้
2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search Test case
3. ระบบทำการบันทึกข้อมูล</td></tr>
<tr><td>Exception Conditions</td><td colspan="2">-</td></tr>
</table>

ตารางที่ 3.3 Use Case Description ของส่วน Create/Edit/Delete/Search test data

<table>
<tr><td>Use-case name</td><td colspan="2">3. Create/Edit/Delete/Search test data</td></tr>
<tr><td>Scenario</td><td colspan="2">การสร้าง แก้ไข ลบ ค้นหา Test data</td></tr>
<tr><td>Brief Description</td><td colspan="2">สามารถสร้าง แก้ไข ลบ ค้นหา Test data ผ่านระบบ</td></tr>
<tr><td>Actor</td><td colspan="2">Tester</td></tr>
<tr><td>Related Use-cases</td><td colspan="2">-</td></tr>
<tr><td>Preconditions</td><td colspan="2">เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Planning</td></tr>
<tr><td>Postconditions</td><td colspan="2">ดูรายละเอียดต่าง ๆ ของ Test Case ของแต่ละโครงการได้</td></tr>
<tr><td>Flow of Events</td><td>Actor</td><td>System</td></tr>
<tr><td></td><td>1.Tester เข้าสู่ระบบ
2. เลือกหน้า Test Planning>Test Data
3. ทำการ Create/Edit/Delete/Search Test data</td><td>1. ตรวจสอบรหัสผู้ใช้
2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search Test data
3. ระบบทำการบันทึกข้อมูล</td></tr>
<tr><td>Exception Conditions</td><td colspan="2">-</td></tr>
</table>

ตารางที่ 3.4 Use Case Description ของส่วน Create/Edit/Delete/Search project checklist

<table>
<tr><td>Use-case name</td><td colspan="2">4. Create/Edit/Delete/Search project checklist</td></tr>
<tr><td>Scenario</td><td colspan="2">การสร้าง แก้ไข ลบ ค้นหา Project checklist</td></tr>
<tr><td>Brief Description</td><td colspan="2">สามารถสร้าง แก้ไข ลบ ค้นหา Project checklist ผ่านระบบ</td></tr>
<tr><td>Actor</td><td colspan="2">Tester</td></tr>
<tr><td>Related Use-cases</td><td colspan="2">-</td></tr>
<tr><td>Preconditions</td><td colspan="2">เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Planning</td></tr>
<tr><td>Postconditions</td><td colspan="2">ดูรายละเอียดต่าง ๆ ของ Project Checklist ของแต่ละโครงการได้</td></tr>
<tr><td rowspan="2">Flow of Events</td><td>Actor</td><td>System</td></tr>
<tr><td>1.Tester เข้าสู่ระบบ
2. เลือกหน้า Test Planning>UAT Project Checklist
3. ทำการ Create/Edit/Delete/Search Project check list</td><td>1. ตรวจสอบรหัสผู้ใช้
2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search Project checklist
3. ระบบทำการบันทึกข้อมูล</td></tr>
<tr><td>Exception Conditions</td><td colspan="2">-</td></tr>
</table>

ตารางที่ 3.5 Use Case Description ของส่วน Review Test Case

Use-case name	5. Review Test Case	
Scenario	การ Review Test Case	
Brief Description	สามารถ Review Test Case ผ่านระบบ และ update ข้อมูลในระบบว่าผ่านการ Review เสร็จสมบูรณ์แล้ว	
Actor	Test Lead	
Related Use-cases	-	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Planning	
Postconditions	ตรวจดูรายละเอียดต่าง ๆ ของ Test Case ของแต่ละโครงการได้	
Flow of Events	Actor	System
	1.Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test Planning>Test Case 3. ทำการ Search Test case เพื่อมา review 4. ปรับปรุงสถานะ Test Case ว่าผ่านการ Review แล้ว	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search Test Case 3. แสดง Test Case 4. ระบบบันทึกข้อมูล
Exception Conditions	-	

ตารางที่ 3.6 Use Case Description ของส่วน Approve Test Case

Use-case name	6. Approve Test Case	
Scenario	การอนุมัติให้สามารถใช้ Test Case ในการทำการทดสอบ	
Brief Description	สามารถอนุมัติให้นำ Test Case ไปใช้ในการทำการทดสอบ	
Actor	User	
Related Use-cases	-	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Planning	
Postconditions	อนุมัติ Test Case ของแต่ละโครงการได้	
Flow of Events	Actor	System
	1.User เข้าสู่ระบบ 2. เลือกหน้า Test Planning>Test Case 3. ทำการ Search Test case เพื่อมา Approve 4. ทำการ Approve Test case	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Approve Test Case 3. แสดง Test Case 4. ระบบบันทึกข้อมูล
Exception Conditions	-	

ตารางที่ 3.7 Use Case Description ของส่วน Update Test result

<table>
<tr><td>Use-case name</td><td colspan="2">7. Update Test result</td></tr>
<tr><td>Scenario</td><td colspan="2">การบันทึกผลการทดสอบใน Test Case</td></tr>
<tr><td>Brief Description</td><td colspan="2">สามารถบันทึกผลใน Test Case ที่นำไปใช้ทดสอบ</td></tr>
<tr><td>Actor</td><td colspan="2">Tester</td></tr>
<tr><td>Related Use-cases</td><td colspan="2">View Test result</td></tr>
<tr><td>Preconditions</td><td colspan="2">เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Executed</td></tr>
<tr><td>Postconditions</td><td colspan="2">บันทึกผลการทดสอบของแต่ละโครงการได้</td></tr>
<tr><td rowspan="2">Flow of Events</td><td>Actor</td><td>System</td></tr>
<tr><td>1. Tester เข้าสู่ระบบ
2. เลือกหน้า Test Planning>Test Executed
3. ทำการบันทึกผลการทดสอบ</td><td>1. ตรวจสอบรหัสผู้ใช้
2. แสดงแถบเครื่องมือให้ Update ข้อมูล
3. แสดงผลการบันทึกข้อมูล</td></tr>
<tr><td>Exception Conditions</td><td colspan="2">-</td></tr>
</table>

ตารางที่ 3.8 Use Case Description ของส่วน View Test result

Use-case name	8. View Test result	
Scenario	การดูผลการทดสอบใน Test Case	
Brief Description	สามารถดูผลการทดสอบใน Test Case ที่นำไปใช้ทดสอบ	
Actor	Test Lead	
Related Use-cases	Update Test result	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Executed	
Postconditions	ดูผลการทดสอบของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test Planning>Test Executed 3. ค้นหา Test Case 4. กด View Test result	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search Test Case 3. แสดงผลการค้นหา Test Case 4. แสดง Test result
Exception Conditions	-	

ตารางที่ 3.9 Use Case Description ของส่วน Approve Test result

Use-case name	9. Approve Test result	
Scenario	การอนุมัติผลการทดสอบ	
Brief Description	สามารถอนุมัติผลการทดสอบในระบบ	
Actor	User	
Related Use-cases	-	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Executed	
Postconditions	อนุมัติผลการทดสอบของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. User เข้าสู่ระบบ	1. ตรวจสอบรหัสผู้ใช้
	2. เลือกหน้า Test Planning>Test Executed	2. แสดงแถบเครื่องมือให้ Approve Test result
	3. ค้นหา Test Case	3. แสดงผลการค้นหา Test Case
	4. กด Approve Test result	4. แสดงผลการ Approve Test result
Exception Conditions	-	

ตารางที่ 3.10 Use Case Description ของส่วน Create/Edit/Delete/Search/Export problem log

Use-case name	10. Create/Edit/Delete/Search/Export problem log	
Scenario	การสร้าง แก้ไข ลบ ค้นหา และนำข้อมูลออก สำหรับ Problem log	
Brief Description	สามารถสร้าง แก้ไข ลบ และนำข้อมูลออกในส่วนของ Problem log ผ่านระบบ	
Actor	Tester	
Related Use-cases	View problem log	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test result reporting	
Postconditions	อนุมัติผลการทดสอบของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. Tester เข้าสู่ระบบ 2. เลือกหน้า Test result reporting>Problem log 3. ทำการ Create/Edit/Delete /Search/Export problem log	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search/Export problem log 3. บันทึกข้อมูลลงระบบ
Exception Conditions	-	

ตารางที่ 3.11 Use Case Description ของส่วน Create/Search/Export Test result report

Use-case name	11. Create/Search/Export Test result report	
Scenario	การสร้าง ค้นหา และนำข้อมูลออก สำหรับ Test Result Report	
Brief Description	สามารถสร้าง แก้ไข ลบ และนำข้อมูลออกในส่วนของ Problem log ผ่านระบบ	
Actor	Tester	
Related Use-cases	View test result report	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test result reporting	
Postconditions	อนุมัติผลการทดสอบของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. Tester เข้าสู่ระบบ 2. เลือกหน้า Test result reporting>Problem log 3. ทำการ Create/Edit/Delete /Search/Export problem log	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search/Export problem log 3. บันทึกข้อมูลลงระบบ
Exception Conditions	-	

ตารางที่ 3.12 Use Case Description ของส่วน View problem log

Use-case name	12. View problem log	
Scenario	การดูข้อมูล Problem log	
Brief Description	สามารถดูข้อมูลในส่วนของ Problem log ผ่านระบบ	
Actor	Test Lead	
Related Use-cases	Create/Edit/Delete/Search/Export problem log	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test result reporting	
Postconditions	เข้าดูข้อมูล Problem log ของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test result reporting>Problem log 3. ทำการค้นหา problem log 4. กด View	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ View problem log 3. แสดงผลการค้นหา Problem Log 4. แสดงรายละเอียดข้อมูล Problem Loge
Exception Conditions	-	

ตารางที่ 3.13 Use Case Description ของส่วน View test result report

Use-case name	13. View test result report	
Scenario	การดูข้อมูล Test result report	
Brief Description	สามารถดูข้อมูลในส่วนของ Test result report ผ่านระบบ	
Actor	Test Lead	
Related Use-cases	Create/Search/Export test result Report	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test result reporting	
Postconditions	เข้าดูข้อมูล Test result report ของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test result reporting> Test result report 3. ทำการค้นหา Test result report 4. กด View	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ View Test result report 3. แสดงผลการค้นหา Test result report 4. แสดงรายละเอียดข้อมูล Test result report
Exception Conditions	-	

ตารางที่ 3.14 Use Case Description ของส่วน Create/Edit/Delete sign off sheet

Use-case name	14. Create/Edit/Delete sign off sheet	
Scenario	การสร้าง แก้ไข และลบ UAT Sign-Off sheet	
Brief Description	สามารถสร้าง แก้ไข และลบ UAT Sign-Off sheet ผ่านระบบ	
Actor	Tester	
Related Use-cases	View sign off sheet	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Closing	
Postconditions	อนุมัติผลการทดสอบของแต่ละโครงการได้	
Flow of Events	Actor	System
	1. Tester เข้าสู่ระบบ 2. เลือกหน้า Test Closing >Sign-Off sheet 3. ทำการ Create/Edit/Delete sign off sheet	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create/Edit/Delete sign off sheet 3. บันทึกข้อมูลลงระบบ
Exception Conditions	-	

ตารางที่ 3.15 Use Case Description ของส่วน Upload sign off sheet

Use-case name	15. Upload sign off sheet	
Scenario	การอัพโหลด UAT Sign-Off sheet	
Brief Description	หลังจากที่ User ลงนามที่เอกสารแล้ว ทำการสแกนเป็นไฟล์ PDF และนำเข้าเก็บในระบบ	
Actor	Tester	
Related Use-cases	-	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Closing	
Postconditions	สามารถอัพโหลด UAT Sign-Off sheet เข้าระบบได้	
Flow of Events	Actor	System
	1. Tester เข้าสู่ระบบ 2. เลือกหน้า Test Closing >Sign-Off sheet 3. ทำการ Upload sign-off sheet	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Upload sign off sheet 3. บันทึกข้อมูลลงระบบ
Exception Conditions	-	

ตารางที่ 3.16 Use Case Description ของส่วน View sign off sheet

Use-case name	16. View sign off sheet	
Scenario	การดูข้อมูล UAT Sign-Off sheet	
Brief Description	สามารถเข้าระบบเพื่อดูข้อมูล UAT Sign-Off sheet ที่บันทึกอยู่ในระบบ	
Actor	Test Lead	
Related Use-cases	Create/Edit/Delete sign off sheet	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Closing	
Postconditions	สามารถดูรายละเอียดข้อมูล UAT Sign-Off sheet ในระบบได้	
Flow of Events	Actor	System
	1. Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test Closing >Sign-Off sheet 3. ทำการค้นหา sign-off sheet 4. กด View sign-off sheet	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ View Sign-Off sheet 3. แสดงผลการค้นหา Sign-Off sheet 4. แสดงรายละเอียดข้อมูล Sign-Off sheet
Exception Conditions	-	

ตารางที่ 3.17 Use Case Description ของส่วน Create New/Edit/Delete/Search Resource

Use-case name	17. Create New/Edit/Delete/Search Resource	
Scenario	การสร้าง แก้ไข ลบ ค้นหา Test Resource	
Brief Description	สามารถสร้าง แก้ไข ลบ ค้นหา Test Resource ผ่านระบบ	
Actor	Test Lead	
Related Use-cases	-	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Resource	
Postconditions	สามารถการสร้าง แก้ไข ลบ ค้นหา Test Resource ผ่านระบบได้	
Flow of Events	Actor	System
	1.Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test Resource >Resource 3. ทำการ Create/Edit/Delete/Search Test Resource	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create/Edit/Delete/Search Test resource 3. ระบบทำการบันทึกข้อมูล
Exception Conditions	-	

ตารางที่ 3.18 Use Case Description ของส่วน Create/Edit/Delete/Search assign work

Use-case name	18. Create/Edit/Delete/Search assign work	
Scenario	การสร้าง แก้ไข ลบ ค้นหา Assign work	
Brief Description	สามารถสร้าง แก้ไข ลบ ค้นหา Assign work ผ่านระบบ	
Actor	Test Lead	
Related Use-cases	View assign work	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Resource	
Postconditions	สามารถการสร้าง แก้ไข ลบ ค้นหา Assign work ผ่านระบบได้	
Flow of Events	Actor	System
	1.Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test Resource >Assign Work 3. ทำการ Create/Edit/Delete/ Search Assign Work	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Assign Work 3. ระบบทำการบันทึกข้อมูล
Exception Conditions	-	

ตารางที่ 3.19 Use Case Description ของส่วน Create Report assign work

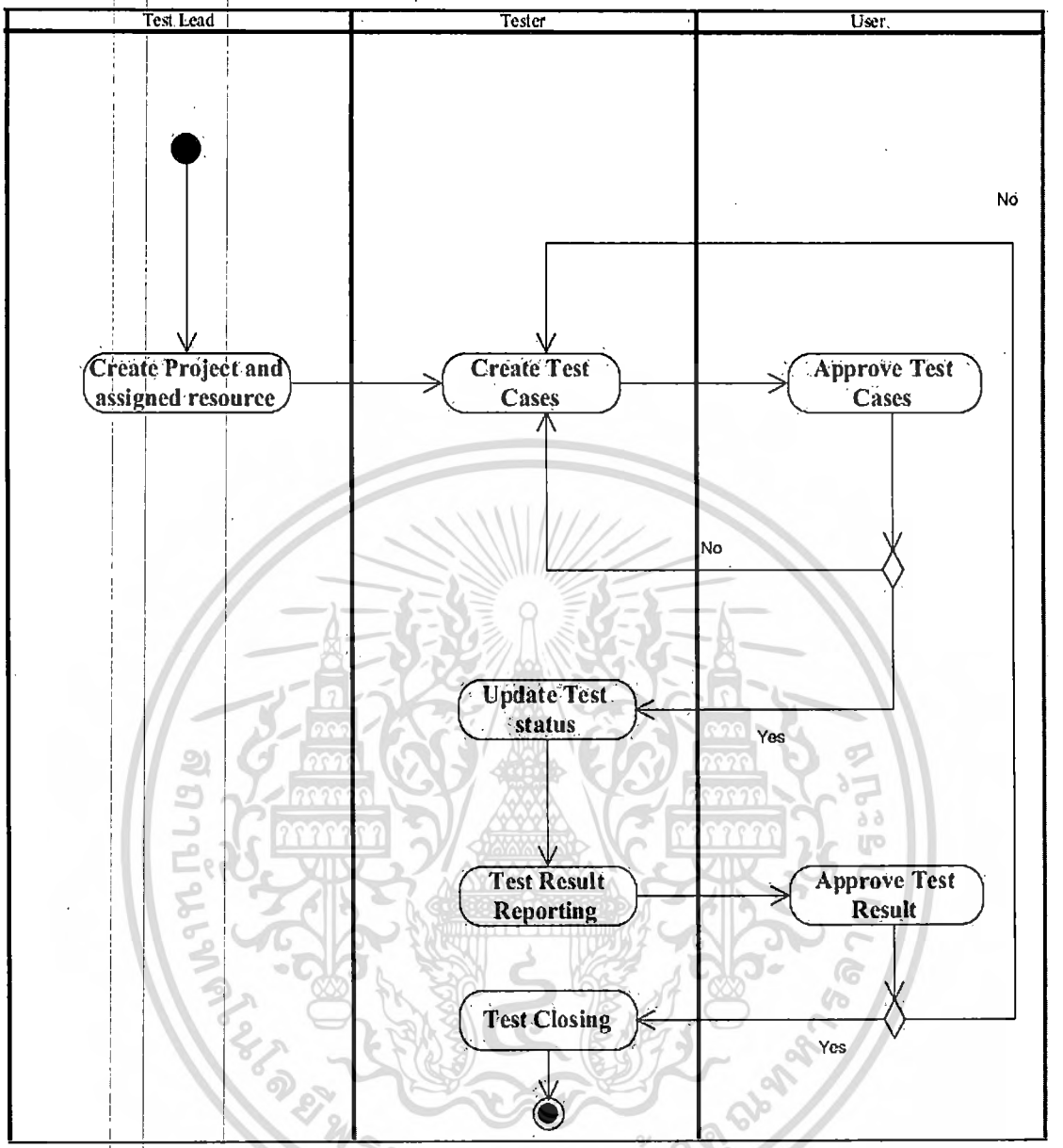
Use-case name	19. Create Report assign work	
Scenario	การสร้างรายงาน Assign work	
Brief Description	สามารถสร้างรายงาน Assign work ผ่านระบบ	
Actor	Test Lead	
Related Use-cases	-	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Resource	
Postconditions	สามารถรายงาน Assign work ผ่านระบบได้	
Flow of Events	Actor	System
	1.Test Lead เข้าสู่ระบบ 2. เลือกหน้า Test Resource >Create Report Assign Work 3. ทำการค้นหาAssign Work 4. กด Create Report Assign Work	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ Create report Assign Work 3. แสดงผลการค้นหา Assign Work 4. ระบบสร้างรายงาน
Exception Conditions	-	

ตารางที่ 3.20 Use Case Description ของส่วน View assign work

Use-case name	20. View assign work	
Scenario	การดูรายละเอียดข้อมูล Assign work	
Brief Description	สามารถดูรายละเอียดข้อมูล Assign work ผ่านระบบ	
Actor	Tester	
Related Use-cases	Create/Edit/Delete/Search assign work	
Preconditions	เข้าสู่ระบบด้วย user account และเข้าใช้หน้า Test Resource	
Postconditions	สามารถดูรายละเอียดข้อมูล Assign work ผ่านระบบได้	
Flow of Events	Actor	System
	1.Tester เข้าสู่ระบบ 2. เลือกหน้า Test Resource >Assign Work 3. ทำการเลือกช่วงเวลาเพื่อ View Assign Work และกด View	1. ตรวจสอบรหัสผู้ใช้ 2. แสดงแถบเครื่องมือให้ View Assign Work 3. ระบบแสดงข้อมูล Assign Work
Exception Conditions	-	

3.3.2 แอ็กทิวิตีไดอะแกรม

แอ็กทิวิตีไดอะแกรม แสดงขั้นตอนการปฏิบัติงานหรือกิจกรรมในการปฏิบัติงานของระบบพัฒนาขึ้น โดยมีการแสดงถึงลำดับของกิจกรรม ในระบบรวมถึงจุดที่ต้องตัดสินใจภายในกระบวนการทำงานดังนี้



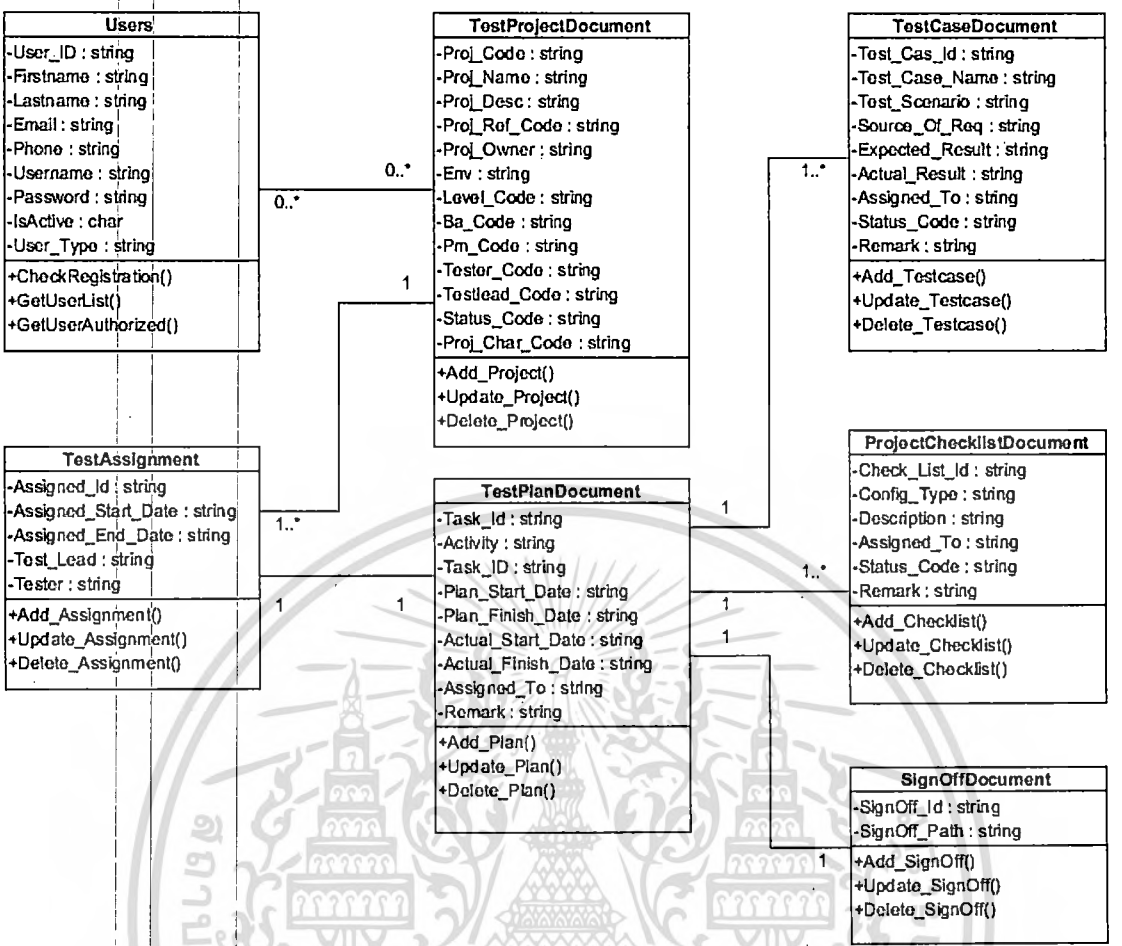
รูปที่ 3.5 แอ็กทิวิตีไดอะแกรมระบบงานใหม่ของระบบระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์

ระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ เริ่มต้นจากการสร้างโครงการใหม่ที่ผู้บริหารจัดการโครงการร้องขอโดยผู้จัดการหน่วยงาน IT Test Management เป็นผู้บันทึกลงในโปรแกรม พร้อมกับกำหนดผู้ที่ต้องทำการทดสอบโครงการนั้น หลังจากนั้นนักทดสอบซอฟต์แวร์จะทำการสร้างกรณีทดสอบสำหรับโครงการนั้น ๆ แล้วส่งให้ผู้ใช้งานระบบอนุมัติชุดกรณีทดสอบก่อนเริ่มทำการทดสอบโครงการ หากผู้ใช้งานพบว่ากรณีทดสอบมีรายละเอียดไม่เพียงพอหรือไม่ครอบคลุมตามความต้องการ จะส่งกลับมาให้นักทดสอบซอฟต์แวร์เพิ่มหรือแก้ไขกรณีทดสอบ ให้ครบถ้วนและส่งกลับให้ผู้ใช้งานอนุมัติอีกครั้ง หลังจากทำการอนุมัติเรียบร้อยแล้ว นักทดสอบ

ซอฟต์แวร์จะเริ่มทำการทดสอบโปรแกรมและปรับปรุงสถานะของกรณีทดสอบว่าทดสอบผ่านหรือไม่ ซึ่งระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์สามารถสรุปสถานะของกรณีทดสอบออกเป็นรายงาน เพื่อมองเห็นภาพรวมของโครงการว่าจากจำนวนกรณีทดสอบทั้งหมดของโครงการสามารถทดสอบผ่านไปแล้วกี่กรณีทดสอบ และยังทดสอบไม่ผ่านอีกเท่าใด เมื่อทำการทดสอบผ่านทุกกรณีทดสอบแล้วจะส่งให้ผู้ใช้งานระบบอนุมัติผลการทดสอบและปิดโครงการโดยนักทดสอบซอฟต์แวร์

3.3.3 คลาสไดอะแกรม

คลาสไดอะแกรม คือ แผนภาพที่แสดงความสัมพันธ์ของคลาสทั้งหมดที่ควรมีในระบบและแสดงโครงสร้างของแต่ละคลาส คลาสไดอะแกรมของระบบระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ ประกอบด้วยคลาสพื้นฐาน ดังรูปที่ 3.4



รูปที่ 3.6 คลาสไดอะแกรมของระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์

Class Diagram ของระบบนี้ประกอบด้วย 12 Class ดังนี้

1. TestProjectDocument : เป็น Class ที่ทำหน้าที่จัดการข้อมูลเกี่ยวกับรายละเอียดของ Project ที่ส่งมาให้ทำการทดสอบ

2. TestPlanDocument : เป็น Class ที่ทำหน้าที่จัดการข้อมูลเกี่ยวกับรายละเอียดTest Plan

3. TestCaseDocument : เป็น Class ที่ทำหน้าที่จัดการข้อมูลเกี่ยวกับ Test Case

4. ProjectChecklistDocument : เป็น Class ที่ทำหน้าที่จัดการข้อมูลเกี่ยวกับรายละเอียดของ Project Checklist

5. TestAssignment : เป็น Class ที่ทำหน้าที่จัดการเกี่ยวกับรายละเอียดของการ Assign work

6. SignOffDocument: เป็น Class ที่ทำหน้าที่จัดการเกี่ยวกับรายละเอียดของการ UAT Sign Off Sheet

7. User : เป็น Class ที่ทำหน้าที่จัดการเกี่ยวกับผู้ใช้ระบบ

3.3.4 การออกแบบฐานข้อมูล

จากการวิเคราะห์และออกแบบโครงสร้างของระบบระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ ทำให้สามารถออกแบบฐานข้อมูลเชิงสัมพันธ์ โดยการแปลงคลาสไดอะแกรมเป็นอีอาร์ไดอะแกรม โดยในอีอาร์ไดอะแกรม จะประกอบด้วยตารางต่างๆ จำนวน 11 เอนทิตี ดังนี้

รูปที่ 3.7 ตัวอย่างแผนภาพแสดงความสัมพันธ์ของข้อมูลในฐานข้อมูล

3.4 เครื่องมือและสภาพแวดล้อมในการพัฒนา

เครื่องมือและภาพแวดล้อมในการพัฒนาระบบ แบ่งได้เป็น 4 ส่วนด้วยกัน คือ

1. Web Server ด้านคอมพิวเตอร์ที่ทำหน้าที่เป็นเซิร์ฟเวอร์ได้ทำการติดตั้งระบบปฏิบัติการ Microsoft Windows 2000 Server ซึ่งภายในได้ทำการติดตั้ง Apache Tomcat เป็น web server ที่คอยให้บริการรับการร้องขอจากเครื่องไคลเอนต์และทำการประมวลผลเพื่อส่งเวบเพจกลับไปยังไคลเอนต์ โดยจะติดต่อกันผ่านทางเวบเบราเซอร์

2. Client คอมพิวเตอร์ที่ทำการร้องขอบริการจากเครื่องเซิร์ฟเวอร์ควรจะมีระบบปฏิบัติการตั้งแต่ Window XP ขึ้นไป และใช้ Microsoft Internet Explorer เป็นเวบเบราเซอร์ ซึ่งเป็นที่นิยมใช้กันอย่างแพร่หลายในปัจจุบัน

3. Database ระบบฐานข้อมูลที่ใช้ในการพัฒนาระบบได้นำเอา Microsoft SQL Server 2000 เข้ามาใช้เป็นฐานข้อมูลของระบบ

4. เทคโนโลยีที่ใช้ในการพัฒนาระบบ คือ J2EE เนื่องจากเป็นกลุ่มของเทคโนโลยีภาษาจาวาที่จำเป็นสำหรับการพัฒนาโปรแกรมประยุกต์ด้วยภาษาจาวาให้ใช้งานได้ในระดับองค์กร โครงสร้างของ J2EE นั้นอยู่บนพื้นฐานของภาษาจาวา ข้อดีของภาษาจาวา คือเขียนโค้ดโปรแกรมเพียงครั้งเดียวก็สามารถนำไปใช้งานที่ใดก็ได้ หรือแพลตฟอร์มใดๆ ก็ได้

บทที่ 4

บทสรุปและข้อเสนอแนะ

4.1 บทสรุป

โดยหลังจากการพัฒนาระบบสามารถสรุปได้ว่า การทำโครงงานนี้สามารถบรรลุตามวัตถุประสงค์ของโครงงานที่ผู้พัฒนาได้นำเสนอไว้ ดังนี้

1. ระบบจะช่วยจัดการกระบวนงานต่างๆ ตั้งแต่การมอบหมายงานของหัวหน้าหน่วยงานจนถึงขั้นตอนการส่งมอบงานที่แล้วเสร็จของพนักงาน

2. การติดตามความก้าวหน้าของงานได้ผ่านเวบไซต์ และการวัดประสิทธิภาพในการทำงานของพนักงาน

3. ไม่สามารถตรวจสอบได้ว่าโครงการนั้นจะได้รับการดำเนินการทดสอบจนสิ้นสุดเสร็จทันเวลาหรือไม่

4. หัวหน้าหน่วยงานไม่มีระบบที่ช่วยจัดเก็บข้อมูลการจัดสรรโครงการให้กับพนักงาน

4.2 ข้อเสนอแนะ

1. เนื่องจากข้อจำกัดทางด้านเวลาผู้พัฒนาจึงเลือกที่จะพัฒนาระบบเฉพาะส่วนของการจัดการ Test Case เท่านั้น ซึ่งยังไม่ครอบคลุมงานทั้งหมด เช่น ส่วนของการทำ Test Plan, เอกสาร UAT Sign-Off เป็นต้น ซึ่งช่วยให้หัวหน้างานสามารถมองเห็นภาพรวมของโครงการที่ทำการทดสอบและสามารถเห็นความก้าวหน้าของการดำเนินการทดสอบได้อย่างมีประสิทธิภาพ

2. ในส่วนของการใช้ UML ในการอธิบายส่วนต่างๆของระบบ ควรมีการใช้ Diagram ในการอธิบายระบบให้มากกว่านี้ เพื่อความเข้าใจในการทำงานระบบอย่างถ่องแท้ และสามารถวิเคราะห์และออกแบบระบบได้ครอบคลุมและมีประสิทธิภาพมากขึ้นรวมทั้งการออกแบบฐานข้อมูลด้วย

3. ในส่วนของรายงานสำหรับหัวหน้างาน ควรเพิ่มเติมรายงานเพื่อสรุปภาพรวมของโครงการทั้งหมด เพื่อให้สามารถมองเห็นภาพของแต่ละโครงการที่ตนเองเป็นผู้ดูแลรับผิดชอบ

ในการพัฒนาระบบให้เกิดประสิทธิภาพสูงสุดไม่ได้ขึ้นอยู่กับการเขียนโปรแกรมของผู้พัฒนาระบบเพียงอย่างเดียว แต่ยังต้องให้ความสำคัญกับความต้องการของผู้ใช้ระบบเป็นสำคัญด้วย การที่โปรแกรมออกมามีฟังก์ชันการทำงานครบถ้วนดีพร้อม ไม่ได้แปลว่าโปรแกรมนั้นจะเป็นโปรแกรมที่ดี หากผู้ใช้ไม่สามารถใช้งานฟังก์ชันต่างๆได้ครบหรือใช้งานได้อย่างถูกต้อง ผู้พัฒนาได้เล็งเห็นถึงความสำคัญของการการออกแบบ User Interface ให้เป็นระบบที่น่าใช้งาน ผู้ใช้ระบบสามารถเรียนรู้และใช้งานระบบอย่างง่าย อย่างคร่าวๆในช่วงของการวิเคราะห์ระบบงาน เพื่อนำการออกแบบนี้ไปปรับให้เข้ากับความต้องการของระบบ ซึ่งจะช่วยให้ระบบที่วิเคราะห์กับโปรแกรม มี

ความสอดคล้องและไปในแนวทางเดียวกันมากขึ้น สุดท้ายนี้ สำหรับระบบควบคุมการไหลของงานการทดสอบซอฟต์แวร์ที่ออกแบบโดยใช้หลักการเชิงวัตถุ (OOP) ถือเป็นประสบการณ์ที่ดีต่อผู้พัฒนาอย่างมาก ที่จะสามารถนำความรู้ที่เกิดจากการศึกษาค้นคว้า และลงมือปฏิบัติด้วยตัวเองนี้ไปใช้ประโยชน์ในการพัฒนาระบบอื่นต่อไป ทั้งนี้หากมีเวลาในการพัฒนามากกว่านี้ ผู้พัฒนาจะมีโอกาสที่จะนำระบบที่ได้พัฒนาขึ้นนี้ไปทดลองใช้งานจริง เพื่อแก้ไขข้อผิดพลาดที่อาจเกิดขึ้นได้

บรรณานุกรม

กิตติ ภักดีวัฒนกุล และพนิดา พานิชกุล.2552.วิศวกรรม**ซอฟต์แวร์ (Software Engineering)**. พิมพ์ครั้งที่ 3.กรุงเทพฯ: ไทยเจริญการพิมพ์.

โรเจอร์ เอส. เพรสแมน.2549.**วิศวกรรมซอฟต์แวร์**.แปลโดย พรฤดี เนติโสภากุล.กรุงเทพฯ:ท้อป.

Denisarona. 2011. **Software testing.** [Online] Available: http://en.wikipedia.org/wiki/Software_testing

Luckas-bot. 2012. **การทดสอบซอฟแวร์.** [Online] Available: http:// th.wikipedia.org/wiki/การทดสอบซอฟแวร์