การกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์โดย Test-Driven Development

Encouraging Students' Interest in Software Development by Test-Driven Development

สุชาดา พงศ์พรม

Suchada Pongprom

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา
วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree the Master of Science in Information Technology
Prince of Songkla University

2560

ชื่อวิทยานิพนธ์ การกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์โดย Test-Driven Development

ผู้เขียน นางสาวสุชาดา พงศ์พรม

สาขาวิชา เทคโนโลยีสารสนเทศ

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

.....

(ผู้ช่วยศาสตราจารย์ ดร.อซีส นันทอมรพงศ์)

คณะกรรมการสอบ

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.รัตนา เวทย์ประสิทธิ์)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.อซีส นันทอมรพงศ์)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.ทรงศรี ตั้งศรีไพโรจน์)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

.....

(รองศาสตราจารย์ ดร.ธีระพล ศรีชนะ)

คณบดีบัณฑิตวิทยาลัย

ขอรับรองว่า ผลงานวิจัยนี้มาจากการศึกษาวิจัยของนักศึกษาเอง และได้แสดงความขอบคุณบุคคลที่มีส่วนช่วยเหลือแล้ว

ลงชื่อ.....
(ผู้ช่วยศาสตราจารย์ ดร.อซีส นันทอมรพงศ์)
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

ลงชื่อ.....
(นางสาวสุชาดา พงศ์พรม)
นักศึกษา

ข้าพเจ้าขอรับรองว่า ผลงานวิจัยนี้ไม่เคยเป็นส่วนหนึ่งในการอนุมัติปริญญาในระดับใดมาก่อน และไม่ได้ถูกใช้ในการยื่นขออนุมัติปริญญาในขณะนี้

ลงชื่อ.....

(นางสาวสุชาดา พงศ์พรม)

นักศึกษา

ชื่อวิทยานิพนธ์ การกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์โดย Test-Driven Development

ผู้เขียน นางสาวสุชาดา พงศ์พรม

สาขาวิชา เทคโนโลยีสารสนเทศ

ปีการศึกษา 2559

บทคัดย่อ

ในปัจจุบันภาคอุตสาหกรรมซอฟต์แวร์ของไทยได้ประสบปัญหาการขาดแคลนนักพัฒนาซอฟต์แวร์ที่มีศักยภาพสูง สาเหตุประการหนึ่ง คือ นักศึกษาที่เรียนทางด้านการพัฒนาซอฟต์แวร์ให้ความสนใจในการเป็นนักพัฒนาซอฟต์แวร์น้อยลง ทั้งที่ในหลายมหาวิทยาลัยมีการเรียนการสอนในหลักสูตรเกี่ยวกับด้านการพัฒนาซอฟต์แวร์มากขึ้น แต่พบว่าบุคลากรที่จบจากมหาวิทยาลัยไม่สามารถช่วยเพิ่มขีดความสามารถในการแข่งขันให้กับภาคอุตสาหกรรมซอฟต์แวร์ได้ดีเท่าที่ควร จึงมีนักวิจัยและบุคลากรทางด้านการศึกษาหลายท่านได้นำวิธีการปฏิบัติทางด้านวิศวกรรมซอฟต์แวร์ไปใช้ในการเรียนการสอนกับนักศึกษาในระดับอุดมศึกษา เช่น การนำ Test-Driven Development (TDD) ไปใช้ในการสอนเพื่อช่วยให้นักศึกษาสามารถทำการพัฒนาซอฟต์แวร์ในช่วงการเรียนได้อย่างมีคุณภาพ มีงานวิจัยหลายงานระบุให้เห็นถึงข้อดีหรือผลกระทบในการนำ TDD ไปใช้ในการเรียนการสอน อย่างไรก็ตามในงานวิจัยทางด้าน TDD ยังไม่มีการศึกษาถึงความสนใจและทัศนคติของนักศึกษาที่มีต่อการใช้ TDD ในการเรียน

จากปัญหาการขาดแคลนนักพัฒนาซอฟต์แวร์ในอุตสาหกรรมซอฟต์แวร์ไทยและงานวิจัยด้าน TDD ยังขาดหลักฐานเชิงประจักษ์ ในด้านการใช้ TDD โดยเฉพาะในกลุ่มนักศึกษาที่กำลังศึกษาในระดับอุดมศึกษา ผู้วิจัยเล็งเห็นถึงความจำเป็นในการศึกษาการนำ TDD ไปช่วยกระตุ้นความสนใจของนักศึกษาที่เรียนทางด้านการพัฒนาซอฟต์แวร์ ผู้วิจัยได้ดำเนินโครงการวิจัยร่วมกับอาจารย์ผู้สอน โดยนักศึกษาจะได้เรียนรู้ TDD และนำ TDD ไปใช้พัฒนาซอฟต์แวร์ในชั่วโมงปฏิบัติการและโครงการซอฟต์แวร์ที่ได้รับมอบหมาย หลังจากนั้นผู้วิจัยได้ทำการประเมินผลที่ได้จากการนำ TDD ไปใช้ในการเรียนด้วยวิธีการวิศวกรรมซอฟต์แวร์เชิงประจักษ์

จากผลงานวิจัยพบว่าการนำ TDD ไปใช้ในการเรียนของนักศึกษาสามารถช่วยกระตุ้นให้นักศึกษาให้ความสนใจในการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น ซึ่งจากผลงานวิจัยจะเป็นแนวทางหนึ่งในการช่วยส่งเสริมการเรียนด้านการพัฒนาซอฟต์แวร์ในสถาบันการศึกษาและอาจจะทำให้สามารถเพิ่มบุคลากรที่มีความสนใจทางด้านการพัฒนาซอฟต์แวร์ให้เข้าสู่ภาคอุตสาหกรรมซอฟต์แวร์มากขึ้น นอกจากนี้งานวิจัยนี้จะช่วยเพิ่มหลักฐานเชิงประจักษ์ซึ่งจะเป็นประโยชน์ต่อนักวิจัยด้านวิศวกรรมซอฟต์แวร์สำหรับการคิดค้นวิธีการที่ทำให้นักศึกษาเกิดความสนใจในการเรียนด้านการพัฒนาซอฟต์แวร์มากขึ้น

คำสำคัญ: วิศวกรรมซอฟต์แวร์ วิศวกรรมซอฟต์แวร์เชิงประจักษ์ การพัฒนาซอฟต์แวร์ Test-Driven Development

Thesis Title Encouraging Students' Interest in Software Development by Test-Driven Development

Author Miss Suchada Pongprom

Major Program Information Technology

Academic Year 2016

ABSTRACT

Nowadays, Thailand's software industry encounters the problem of an insufficient number of high quality software developers. One of the reasons is that only a few graduates who studied software development related courses are interested in software developer professionals. Although many universities in Thailand have launched software development programs, new graduates could not break into a competitive software industry. Researchers and educational practitioners have been adopting more software engineering practices in software development courses. For example, the Test-Driven Development (TDD) approach has been taught at the software development class to improve software quality. Many studies indicate the advantages and impacts of TDD on teaching in an academic environment. However, none of them explore students' attitudes on using TDD.

With the shortage of Thai software developers and lacking empirical evidence, the author has seen the essence of how TDD can encourage students to be more interested in software development. The author conducted the research project with the professor to provide TDD knowledge to students who later employed TDD on their assigned class projects. Then, the author empirically evaluated the outcome of applying TDD to software development projects.

The results of this study show that the students' interest in software development was enhanced, TDD would be an alternative approach to improve the study of software development existed in universities. Subsequently, the academies

would also produce high quality software developers for the software industry. Additionally, the empirical evidence of this research would be beneficial for software engineering researchers to invent an approach to enhance students' attention to software development.

Keywords: Software Engineering, Empirical Software Engineering, Test-Driven Development

กิตติกรรมประกาศ

งานวิจัยนี้สำเร็จลุล่วงได้ด้วยดีโดยความอนุเคราะห์และความกรุณาอย่างยิ่งจากผู้มีพระคุณหลาย ๆ ฝ่ายโดยเฉพาะอย่างยิ่ง ผู้ช่วยศาสตราจารย์ ดร.อชีส นันทอมรพงศ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ผู้ซึ่งให้ความรู้ คำปรึกษา ข้อเสนอแนะสำหรับงานวิจัย และให้การอนุเคราะห์ในการทดลองกับกลุ่มตัวอย่างที่เป็นนักศึกษาสาขาวิชาวิศวกรรมซอฟต์แวร์ภายในรายวิชาโปรแกรมเชิงอ็อบเจกต์ขั้นสูง รวมไปถึงการติดตามความก้าวหน้าในการดำเนินงานวิจัย และชี้แนะแนวทางในการทำวิทยานิพนธ์ ผู้วิจัยรู้สึกซาบซึ้งในความกรุณาของอาจารย์เป็นอย่างยิ่งและขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณผู้ช่วยศาสตราจารย์ ดร.รัตนา เวทย์ประสิทธิ์ ผู้เป็นประธานกรรมการสอบวิทยานิพนธ์ และ ผู้ช่วยศาสตราจารย์ ดร.ทรงศรี ตั้งศรีไพโรจน์ ผู้เป็นคณะกรรมการสอบวิทยานิพนธ์ รวมถึงคณาจารย์ทุกท่าน ที่สละเวลาเพื่อช่วยให้ความรู้ แนะนำข้อผิดพลาดและแนะแนวทางในการปรับปรุงวิทยานิพนธ์

ขอขอบคุณคณะเทคโนโลยีและสิ่งแวดล้อม และขอขอบคุณบัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ ซึ่งมอบโอกาสในการศึกษาและทุนสนับสนุนการวิจัยในครั้งนี้ ขอขอบคุณ คุณวรนุช ญาณศักดิ์ เจ้าหน้าที่งานวิจัยและพัฒนานักศึกษา และคุณฐิติมา วศินพัฒนวิศิษฎ์ เจ้าหน้าที่งานบัณฑิตศึกษา ที่คอยชี้แนะแนวทางให้คำแนะนำและช่วยเหลือในด้านการติดต่อประสานงานต่าง ๆ เกี่ยวกับการทำวิทยานิพนธ์ รวมถึงบุคลากรทุกท่านที่ให้ความช่วยเหลือด้วยดีตลอดมา

นอกจากนี้ผู้วิจัยยังได้รับกำลังใจจากบิดา มารดา และครอบครัว ที่คอยให้กำลังใจ สนับสนุน ส่งเสริม และช่วยเหลือในทุกด้านจนสำเร็จการศึกษา ตลอดจนบุคคลต่าง ๆ ที่ให้ความช่วยเหลืออีกมากมาย ซึ่งผู้วิจัยอาจไม่สามารถกล่าวนามได้หมดในที่นี้ ผู้วิจัยรู้สึกซาบซึ้งในความกรุณาและความปรารถนาดีของทุกท่านเป็นอย่างยิ่ง ประโยชน์อันใดที่เกิดจากวิทยานิพนธ์เล่มนี้ย่อมเป็นผลมาจากความกรุณาของทุกท่านดังกล่าวข้างต้น ผู้วิจัยขอขอบพระคุณมา ณ โอกาสนี้

สุชาดา พงศ์พรม

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

รายการตาราง

รายการภาพประกอบ

รายการภาพประกอบ (ต่อ)

รูปที่ **หน้า**

บทที่ 1

บทนำ

1.1 ความสำคัญและที่มาของการวิจัย

ปัจจุบันมีจำนวนนักศึกษาที่ให้ความสนใจในการเป็นนักพัฒนาซอฟต์แวร์น้อยลง ซึ่งส่งผลให้ภาคอุตสาหกรรมซอฟต์แวร์มีความต้องการนักพัฒนาซอฟต์แวร์ที่สูงขึ้น เนื่องจากมีนักพัฒนาซอฟต์แวร์ไม่เพียงพอต่อความต้องการของภาคอุตสาหกรรม ทั้งที่ในปัจจุบันหลายมหาวิทยาลัยในประเทศไทยมีการเปิดหลักสูตรเกี่ยวกับการพัฒนาซอฟต์แวร์เพิ่มขึ้น แต่กลับพบว่าบุคลากรที่จบการศึกษาจากมหาวิทยาลัยไม่สามารถเพิ่มขีดความสามารถในการแข่งขันให้กับภาคอุตสาหกรรมซอฟต์แวร์ (คณะกรรมการอาเซียน สภาหอการค้าไทยและเขตอุตสาหกรรมซอฟต์แวร์แห่งชาติ, 2015) โดยทั่วไปการเรียนการสอนด้านการพัฒนาซอฟต์แวร์มีรูปแบบการสอนที่เริ่มจากการสอนให้รู้จักกับพื้นฐานการเขียนโปรแกรม คือ การประกาศตัวแปรและโครงสร้างการทำงานของภาษานั้น ๆ เช่น if-else, for, while และหลังจากนั้นจะมอบหมายให้นักศึกษาทำแบบฝึกหัดตามโจทย์หรือปัญหาที่กำหนดให้ สำหรับวิธีการสอนรูปแบบนี้มีความสอดคล้องกับกระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก (Waterfall Model) (Royce, 1970) การพัฒนาซอฟต์แวร์ในลักษณะแบบจำลองนี้จะเริ่มต้นจากการที่ต้องรู้ความต้องการของระบบ (Software Requirement) ให้มากที่สุด หลังจากนั้นจะทำการออกแบบระบบ (Software Design) และจึงเริ่มเขียนโค้ดตามการออกแบบที่ได้ระบุไว้ (Implementation) เมื่อโค้ดได้ถูกเขียนเสร็จสิ้น ในขั้นตอนสุดท้ายนักพัฒนาจะทำการทดสอบระบบ (Testing) ซึ่งการทดสอบระบบในขั้นตอนสุดท้ายนี้โดยทั่วไปผู้พัฒนาซอฟต์แวร์จะมุ่งเน้นที่จะให้ซอฟต์แวร์ทำงานได้โดยไม่ติดปัญหาในขณะทำงาน (No

Runtime Error) กล่าวคือ เมื่อเขียนโค้ดเสร็จสิ้น ผู้พัฒนาซอฟต์แวร์ก็จะทดสอบโปรแกรมกับกรณีทดสอบเพียงไม่กี่กรณี เมื่อโปรแกรมสามารถทำงานได้โดยไม่ติดปัญหาก็จะถือว่าการทดสอบนั้นเสร็จสิ้น แทนที่จะทำการทดสอบให้ครอบคลุมทุกเงื่อนไขการทำงาน ในการทดสอบในขั้นตอนสุดท้ายนี้ยังมีแนวโน้มที่จะสามารถทำให้โครงการการพัฒนาซอฟต์แวร์ล้มเหลวได้ จากระยะเวลาที่ใช้ในการพัฒนาซอฟต์แวร์ที่เพิ่มมากขึ้น เนื่องจากในการแก้ไขความผิดพลาดหลังการทดสอบนั้นจะใช้เวลามากกว่าการพัฒนาซอฟต์แวร์ โดยปัญหาดังกล่าวสามารถเกิดขึ้นได้กับนักศึกษาที่กำลังศึกษาในด้านการพัฒนาซอฟต์แวร์เช่นกัน ซึ่งในสภาพแวดล้อมของการเรียนวิชาทางด้านการพัฒนาซอฟต์แวร์นั้น บ่อยครั้งจะพบว่านักศึกษาจะไม่เข้าใจถึงความต้องการของระบบได้อย่างแท้จริง หรือไม่สามารถหาวิธีแก้ปัญหาโจทย์ที่ได้รับมอบหมาย จึงทำให้นักศึกษาไม่สามารถเขียนโค้ดให้ตรงตามความต้องการได้ หรือเกิดปัญหาในขณะเขียนโปรแกรม เช่น โปรแกรมที่เขียนเกิดข้อผิดพลาดไม่สามารถทำงานต่อได้ เป็นต้น

Test-Driven Development (TDD) เป็นวิธีการปฏิบัติที่สำคัญอย่างหนึ่งในการพัฒนาซอฟต์แวร์แบบเอกซ์ตรีมโปรแกรมมิ่ง (Extreme Programming) ซึ่งเป็นวิธีการหนึ่งในการพัฒนาซอฟต์แวร์แบบเอไจล์ (Agile) การพัฒนาซอฟต์แวร์แบบเอไจล์ได้เข้ามาเป็นที่นิยมของกลุ่มนักพัฒนาซอฟต์แวร์ทั่วโลกตั้งแต่ในปี พ.ศ. 2544 และมีแนวโน้มการนำไปใช้ในการพัฒนาซอฟต์แวร์มากขึ้นเรื่อย ๆ เนื่องจากปัจจุบันระบบซอฟต์แวร์มีความซับซ้อนเพิ่มมากขึ้น รวมถึงการใช้การพัฒนาซอฟต์แวร์แบบเอไจล์ทำให้ผู้พัฒนาระบบกับผู้ใช้งานระบบหรือลูกค้าสามารถสื่อสารและทำความเข้าใจเรื่องความต้องการของระบบได้ตรงกันได้มากขึ้น เพราะการพัฒนาซอฟต์แวร์แบบเอไจล์มีรูปแบบการทำงานเป็นรอบสั้น ๆ โดยในแต่ละรอบการทำงาน ผู้พัฒนาซอฟต์แวร์จะเลือกฟังก์ชันการทำงานมาจำนวนหนึ่งเพื่อทำการพัฒนา และเมื่อฟังก์ชันดังกล่าวได้รับการพัฒนาเสร็จสิ้นผู้พัฒนาซอฟต์แวร์จะส่งมอบงานให้แก่ลูกค้า ทำให้ลูกค้าเห็นในสิ่งที่ลูกค้าต้องการได้เร็วขึ้นและเป็นรูปธรรมว่าสิ่งที่นักพัฒนาซอฟต์แวร์ทำการพัฒนาขึ้นนั้นตรงกับความต้องการของลูกค้าหรือไม่ หากซอฟต์แวร์ที่พัฒนาไม่ตรงตามความต้องการที่ลูกค้ากำหนดไว้ลูกค้าก็จะสามารถให้ความคิดเห็นหรือข้อเสนอแนะเพื่อแก้ไขให้ตรงตามความต้องการโดยสามารถเริ่มแก้ไขได้ตั้งแต่เริ่มพัฒนาทำให้โอกาสที่ระบบจะไม่ตรงตามความต้องการของลูกค้ามีโอกาสเกิดขึ้นได้น้อยลง และทำให้ได้ซอฟต์แวร์ที่มีคุณภาพสูงขึ้น (Matharu, *et al.*, 2015) นอกจากนี้ยังมีหนังสือที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ที่ได้รับการตีพิมพ์ออกมาใหม่มีการแนะนำให้นำ TDD ไปใช้ในการสอนเพื่อให้นักศึกษามีระเบียบวินัยในการพัฒนาซอฟต์แวร์ด้วยวิธีการ TDD เนื่องจาก TDD ช่วยปรับปรุงในส่วนของกระบวนการในการ

พัฒนาซอฟต์แวร์ (Desai, *et al.*, 2008) โดยรูปแบบของ TDD เป็นการเขียนโค้ดที่ใช้ในการทดสอบก่อนการเขียนโปรแกรมจริง หลังจากเขียนโค้ดที่ใช้ในการทดสอบแล้วผู้พัฒนาซอฟต์แวร์จึงจะเริ่มเขียนโค้ดของโปรแกรมจริง และจะนำโปรแกรมที่เขียนขึ้นมาทดสอบกับโปรแกรมทดสอบที่สร้างไว้ โดยในการเขียนโปรแกรมจะทำการเขียนโปรแกรมเป็นรอบ ๆ จนครบทุกฟังก์ชัน เมื่อทำการทดสอบผ่านแล้ว นักพัฒนาซอฟต์แวร์จะทำการปรับปรุงโค้ดให้มีคุณภาพดีขึ้นเนื่องจากโค้ดที่เขียนขึ้นนั้นยังมีคุณภาพที่ไม่ดีเนื่องจากในการเขียนโค้ดในขั้นตอนนี้จะทำการเขียนโค้ดเพียงเพื่อให้โค้ดสามารถทำงานได้ถูกต้อง ซึ่งในขั้นตอนการปรับปรุงโค้ดนี้จะเรียกว่า รีแฟคทอริง (Refactoring) ซึ่งรีแฟคทอริงเป็นการเปลี่ยนโครงสร้างภายในของซอฟต์แวร์ แต่ไม่ได้เป็นการเปลี่ยนแปลงการทำงานของซอฟต์แวร์โดยการทำรีแฟคทอริงมีวัตถุประสงค์เพื่อเพิ่มคุณภาพของซอฟต์แวร์ เช่น ลดความซ้ำซ้อนของโค้ด (Duplicated code) (Wu and Lin, 2001) ซึ่งการซ้ำซ้อนของโค้ดทั้งที่เป็นการทำงานเดียวกันเพียงแต่อยู่คนละฟังก์ชันการทำงานนั้นสามารถปรับปรุงได้โดยการทำรีแฟคทอริงด้วยการเพิ่มซุปเปอร์คลาส (Super-class) โดยนำการทำงานที่มักเกิดซ้ำไปเป็นซุปเปอร์คลาส แล้วค่อยให้ฟังก์ชันอื่นทำการเรียกใช้การทำงานในซุปเปอร์คลาสผ่านคลาสปกติ จะทำให้โค้ดไม่เกิดการซ้ำซ้อนและสามารถลดจำนวนบรรทัดของโค้ดได้อีกด้วย จากงานวิจัยก่อนหน้านี้ของนักวิจัยทางด้าน TDD ได้มีงานวิจัยที่ทำการศึกษากับกลุ่มตัวอย่างผู้ใช้งาน TDD ซึ่งแบ่งออกได้เป็น 3 ลักษณะ คือ 1) การทดลองการใช้ TDD ในกลุ่มของนักพัฒนาซอฟต์แวร์มืออาชีพ 2) การทดลองการใช้ TDD ในกลุ่มของนักศึกษา และ 3) การทดลองโดยเปรียบเทียบการใช้ TDD ระหว่างกลุ่มของนักพัฒนามืออาชีพและกลุ่มของนักศึกษา ซึ่งจากงานวิจัยของ Janzen และคณะมีความใกล้เคียงกับสิ่งที่ผู้วิจัยสนใจ คือ การศึกษาการนำ TDD ไปใช้กับกลุ่มของนักศึกษา (Janzen and Saiedian, 2008) กล่าวคือ ในงานวิจัยของ Janzen และคณะได้ทำการศึกษาการนำ TDD มาช่วยในการเรียนการสอนเพื่อให้นักศึกษาสามารถพัฒนาระบบซอฟต์แวร์ได้ดีขึ้น จากการศึกษาพบว่าระบบซอฟต์แวร์มีคุณภาพที่ดีขึ้น แต่ในงานวิจัยนั้นไม่ได้ทำการศึกษาถึงความรู้สึก หรือทัศนคติของนักศึกษาผู้ใช้ TDD ในการทำงานที่ได้รับมอบหมาย

จากปัญหาความต้องการนักพัฒนาซอฟต์แวร์ในภาคอุตสาหกรรมซอฟต์แวร์ และงานวิจัยทางด้าน TDD ยังขาดหลักฐานเชิงประจักษ์ในด้านทัศนคติของผู้นำ TDD ไปใช้ โดยเฉพาะกับกลุ่มนักศึกษา นอกจากนี้ Booch ผู้คิดค้นยูเอ็มแอล (UML) ได้กล่าวถึงทิศทางของงานวิจัยด้านวิศวกรรมซอฟต์แวร์ในฐานะผู้บรรยายรับเชิญพิเศษในงานประชุมวิชาการ International Conference on Software

Engineering (ICSE) ในปี ค.ศ. 2015 (Booch, 2015) ไว้ว่าอนาคตงานวิจัยด้านวิศวกรรมซอฟต์แวร์จะไม่เน้นไปด้านความเร็วในการประมวลผล หรือเทคโนโลยีในการพัฒนาซอฟต์แวร์ แต่จะให้ความสนใจกับพฤติกรรมของมนุษย์ทั้งในทางด้านของผู้ใช้งานระบบและผู้พัฒนาระบบซอฟต์แวร์มากขึ้น ซึ่งจากการบรรยายนี้สนับสนุนกับแนวความคิดที่ผู้วิจัยสนใจ คือ เป็นการศึกษาความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ โดยการสังเกตและเปลี่ยนแปลงพฤติกรรมของผู้เรียนให้มีความสนใจในการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น สิ่งที่ผู้วิจัยสนใจ คือ การให้นักศึกษาได้นำ TDD ไปใช้ในการพัฒนาซอฟต์แวร์ในการเรียนวิชาทางด้านการพัฒนาซอฟต์แวร์ โดยผู้วิจัยได้ทำงานร่วมกับอาจารย์ผู้สอนในการดำเนินงานวิจัย โดยให้นักศึกษาได้ทำการเรียนรู้ TDD และนำ TDD ไปใช้พัฒนาซอฟต์แวร์ และโจทย์แบบฝึกหัดในการพัฒนาซอฟต์แวร์ที่ได้รับมอบหมายในวิชา สำหรับคำถามวิจัยหลักในงานวิจัยนี้ คือ *การใช้ Test-Driven Development สามารถช่วยกระตุ้นความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษาได้หรือไม่*

เพื่อให้ได้คำตอบสำหรับคำถามงานวิจัยข้างต้น ผู้วิจัยได้ใช้วิธีการต่าง ๆ ในด้านวิศวกรรมซอฟต์แวร์เชิงประจักษ์ (Seaman, 1999) ได้แก่ การทดลองแบบควบคุม (Controlled Experiment) การใช้แบบสอบถาม (Survey) การสัมภาษณ์ (Interview) และการสังเกต (Observation) โดยข้อมูลเชิงประจักษ์ที่ได้จะถูกนำไปวิเคราะห์ทั้งในเชิงปริมาณ (Quantitative) และคุณภาพ (Qualitative) หลังจากการวิเคราะห์ข้อมูลแล้ว ผู้วิจัยได้จัดทำรายงานเพื่อแสดงผลที่ได้จากงานวิจัย ผู้วิจัยเชื่อว่าผลการดำเนินงานของงานวิจัยนี้จะเป็นแนวทางหนึ่งที่ช่วยปรับเปลี่ยนวิธีการเรียนการสอนด้านการพัฒนาซอฟต์แวร์ในระดับมหาวิทยาลัย และอาจจะทำให้ภาคการศึกษาสามารถผลิตบุคลากรที่มีความสนใจทางด้านการพัฒนาซอฟต์แวร์เข้าสู่ภาคอุตสาหกรรมซอฟต์แวร์เพิ่มมากขึ้น รวมถึงเป็นการเพิ่มหลักฐานเชิงประจักษ์ของการนำ TDD ไปใช้ในการเรียนการสอน

1.2 วัตถุประสงค์

1.2.1 เพื่อศึกษาและประเมินผลการนำ TDD ไปช่วยนักศึกษาในการพัฒนาซอฟต์แวร์

1.2.2 เพื่อเป็นการกระตุ้นให้นักศึกษามีความสนใจในการพัฒนาซอฟต์แวร์มากขึ้น

1.2.3 เพิ่มหลักฐานเชิงประจักษ์ทางด้านวิศวกรรมซอฟต์แวร์ให้กับชุมชนนักวิจัยที่สนใจผลกระทบของการนำ TDD มาใช้

1.3 ความสำคัญและประโยชน์ของการวิจัย

ในงานวิจัยนี้ผู้วิจัยเชื่อว่าผลของงานวิจัยนี้จะมีประโยชน์ต่อทั้งนักวิจัยด้านวิศวกรรมซอฟต์แวร์ และบุคลากรทางด้านการศึกษาโดยเฉพาะทางด้านวิศวกรรมซอฟต์แวร์ในประเทศไทยดังต่อไปนี้

1.3.1 พัฒนา และปรับปรุงวิธีการสอนรายวิชาที่เกี่ยวกับการพัฒนาซอฟต์แวร์ในระดับมหาวิทยาลัย หรือที่สูงกว่า

1.3.2 เพิ่มหลักฐานเชิงประจักษ์ด้านวิศวกรรมซอฟต์แวร์ในเรื่อง TDD แก่ผู้วิจัยอื่น

1.4 ขอบเขตการวิจัย

งานวิจัยนี้ผู้วิจัยใช้กลุ่มตัวอย่างจากนักศึกษาสาขาวิชาวิศวกรรมซอฟต์แวร์ ชั้นปีที่ 2 จำนวนทั้งหมด 52 คนในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ขั้นสูง (Advanced Object Oriented Programming : AOOP) จากนักศึกษาทั้งหมดของนักศึกษาสาขาวิชาวิศวกรรมซอฟต์แวร์ทั้งหมด จำนวน 4 ชั้นปี ซึ่งกลุ่ม

ตัวอย่างที่ทำการเลือกเป็นนักศึกษาสาขาวิชาวิศวกรรมซอฟต์แวร์ เนื่องจากงานวิจัยนี้เกี่ยวข้องกับการทดสอบระบบและการทำรีแฟคทอริงซึ่งเป็นส่วนหนึ่งของการเรียนด้านวิศวกรรมซอฟต์แวร์ และนักศึกษาในกลุ่มนี้มีพื้นฐานโปรแกรมมิ่งจากรายวิชาการโปรแกรมเชิงอ็อบเจกต์ (Object Oriented Programming : OOP)

บทที่ 2

วรรณกรรมที่เกี่ยวข้อง

บทนี้เป็นส่วนของการนำเสนอเนื้อหา แนวคิด ทฤษฎีต่าง ๆ และงานวิจัยที่มีผู้ดำเนินการแล้ว ซึ่งมีความเกี่ยวข้องและสำคัญต่องานวิจัยที่ผู้วิจัยศึกษา เนื้อหาในบทนี้แบ่งออกเป็น 2 ส่วน คือ ทฤษฎีที่เกี่ยวข้องและวรรณกรรมที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

ทฤษฎีที่เกี่ยวข้องในงานวิจัยประกอบไปด้วย 6 ส่วน คือ

1) เอกซ์ตรีมโปรแกรมมิ่ง (Extreme Programming or XP)

2) Test-Driven Development (TDD)

3) กระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก (Waterfall Model)

4) วิจัยเชิงประจักษ์ด้านวิศวกรรมซอฟต์แวร์ (Empirical Software Engineering Research)

5) คำนิยามเรื่องความสนใจ (Interest)

6) การแสดงทางอารมณ์ (Emotional and Mood)

โดยมีรายละเอียดทฤษฎีที่เกี่ยวข้องในงานวิจัยในแต่ละส่วนมี ดังนี้

2.1.1 เอกซ์ตรีมโปรแกรมมิ่ง (Extreme Programming or XP)

การพัฒนาซอฟต์แวร์แบบเอกซ์ตรีมโปรแกรมมิ่ง (Abrahamsson, *et al.*, 2009) เป็นวิธีการหนึ่งในการพัฒนาซอฟต์แวร์รูปแบบเอไจล์ ซึ่งการพัฒนาซอฟต์แวร์แบบเอกซ์ตรีมโปรแกรมมิ่งเป็นกระบวนการพัฒนาซอฟต์แวร์ที่ได้กำหนดวิธีการที่จะทำให้ผู้ใช้และผู้พัฒนาซอฟต์แวร์ทำงานร่วมกันในทีมพัฒนาเพื่อก่อให้เกิดการสื่อสารขึ้นภายในทีมมากยิ่งขึ้น ทำให้การทำงานร่วมกันสามารถแก้ปัญหาจำนวนมากที่เกิดขึ้นระหว่างและหลังการพัฒนาซอฟต์แวร์ โดยแนวคิดหลักจะเน้นไปที่การทำงานร่วมกัน (Collaboration) การสื่อสาร (Communication) และการยึดมั่นในระเบียบวินัย (Discipline) ซึ่งการพัฒนาซอฟต์แวร์แบบเอกซ์ตรีมโปรแกรมมิ่งมีแนวทางปฏิบัติ 12 ข้อ ที่สามารถนำมาใช้เป็นแนวทางในการพัฒนาซอฟต์แวร์ดังนี้

(1) การทดสอบ (Test-Driven Development) เป็นการเขียนโค้ดการทดสอบก่อนการเขียนโปรแกรมจริง แล้วจึงทำการเขียนโปรแกรม หลังจากนั้นจึงนำโปรแกรมที่เสร็จแล้วมาทดสอบกับการโค้ดในการทดสอบจนกว่าจะผ่านการทดสอบ แล้วทำการรีแฟคทอริง

(2) การส่งมอบงานขนาดเล็ก (Small Release) การส่งมอบงานขนาดเล็กมีส่วนช่วยให้มั่นใจได้ว่าผู้พัฒนาและผู้ใช้งานสามารถทำงานร่วมกันได้ ทำให้มีโอกาสในการแลกเปลี่ยนความคิดเห็นอย่างต่อเนื่อง

(3) รีแฟคทอริง (Refactoring) เป็นการเปลี่ยนแปลงโครงสร้างการทำงานภายในของโปรแกรม แต่ไม่ได้ทำการเปลี่ยนแปลงฟังก์ชันการทำงานของโปรแกรม เพื่อให้โปรแกรมมีคุณภาพสูงและเป็นการเปลี่ยนแปลงโครงสร้างให้มีความยืดหยุ่นรองรับการเปลี่ยนแปลงที่อาจจะเกิดขึ้นในอนาคต ตัวอย่างของการรีแฟคทอริง เช่น การเปลี่ยนชื่อตัวแปรให้สื่อความหมาย การเพิ่มซุปเปอร์คลาส (Add Super-class) (Wu and Lin, 2001)

(4) มีการออกแบบที่เข้าใจง่าย (Simple Design) เป็นการลดความซับซ้อน ซึ่งมีความสำคัญมากต่อการพัฒนาซอฟต์แวร์

(5) การวางแผน (Planning Game) เป็นการสัมภาษณ์ผู้ใช้งานซอฟต์แวร์ และให้ผู้ใช้งานเขียนความต้องการระบบที่รับผิดชอบเป็นรูปแบบเหตุการณ์ (User Story) แล้วนำข้อมูลมาทำการสรุปงานออกเป็นส่วนย่อย ๆ แล้วประมาณเวลาที่ใช้ในการเขียนโปรแกรมของงานย่อย

(6) การเขียนโปรแกรมเป็นคู่ (Pair Programming) การเขียนโปรแกรมโดยใช้นักพัฒนาซอฟต์แวร์ 2 คนในการทำงานต่อคอมพิวเตอร์ 1 เครื่อง เพื่อให้สามารถช่วยกันตรวจทานและแก้ไขข้อผิดพลาด

(7) สถานที่ของลูกค้า (Onsite Customer) การทำงานในสถานที่ที่เดียวกับลูกค้าเป็นการเพิ่มความมั่นใจว่าปัญหา หรือข้อสงสัยจะได้รับคำตอบทันที เนื่องจากมีผู้ให้ข้อมูลร่วมทำงานกับผู้พัฒนา

(8) การอธิบายความหมายของระบบ (System Metaphor) เป็นการกำหนดนิยามความหมายของคำศัพท์เฉพาะทางเทคนิคของลูกค้าให้เข้าใจตรงกับทีมพัฒนา

(9) การเป็นเจ้าของโค้ดร่วมกัน (Collective Code Ownership) เป็นตัวช่วยให้ทุกคนภายในทีมต้องมีการทำงานร่วมกัน เพื่อลดความเสี่ยงในการที่มีผู้เข้าใจระบบภายในทีมเพียงผู้เดียว

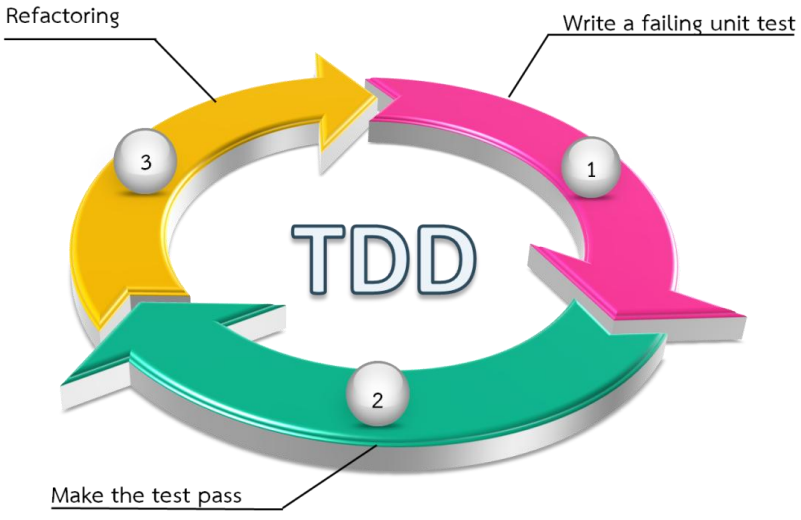
(10) การบูรณาการอย่างต่อเนื่อง (Continuous Integration) งานส่วนที่ทำการเปลี่ยนแปลงถูกนำไปแทนที่งานเก่าทันที เพื่อให้ผู้พัฒนาได้ทำงานในเวอร์ชันล่าสุดเสมอ การทำเช่นนี้จะช่วยให้พบปัญหาได้เร็วขึ้น

(11) ข้อกำหนดในการเขียนโค้ด (Coding Conventions) การเขียนโค้ดให้มีแบบแผนที่ตรงกันภายใต้ข้อตกลงที่ถูกกำหนดไว้เป็นการเพิ่มการสื่อสารภายในทีม

(12) การทำงานแบบยั่งยืน (Sustainable Pace) ในการพัฒนาซอฟต์แวร์ ควรใช้เวลาทำงานอย่างมีประสิทธิภาพภายใต้เวลาที่จำกัด เช่น ไม่ควรทำงานเกิน 40 ชั่วโมงต่อสัปดาห์ ซึ่งจะเป็นการช่วยลดความเครียดและกระตุ้นความสนใจในการทำงานในสัปดาห์ต่อไป

2.1.2 Test-Driven Development (TDD)

TDD เป็นวิธีการปฏิบัติหนึ่งในการพัฒนาซอฟต์แวร์แบบเอกซ์ตรีมโปรแกรมมิ่ง (Beck, 2003; Fraser, *et al.*, 2003; Janzen and Saiedian, 2005; Madeyski, 2009) ซึ่งมีวัตถุประสงค์เพื่อช่วยลดความยุ่งยากในการพัฒนา และเพิ่มคุณภาพซอฟต์แวร์ หลักการสำคัญของ TDD คือ ผู้พัฒนาซอฟต์แวร์ต้องเขียนโค้ดในการทดสอบก่อนทำการเขียนโปรแกรมจริงซึ่งเป็นวิธีการที่ช่วยให้นักพัฒนามั่นใจว่าซอฟต์แวร์ที่ได้สร้างขึ้นนั้นตรงตามความต้องการของผู้ใช้ และยังช่วยให้นักพัฒนาซอฟต์แวร์มั่นใจว่าขอบเขตการทำงานของซอฟต์แวร์ได้ถูกดำเนินการครบถ้วนสมบูรณ์



รูปที่ 2.1 แสดงกระบวนการทำงานของ TDD

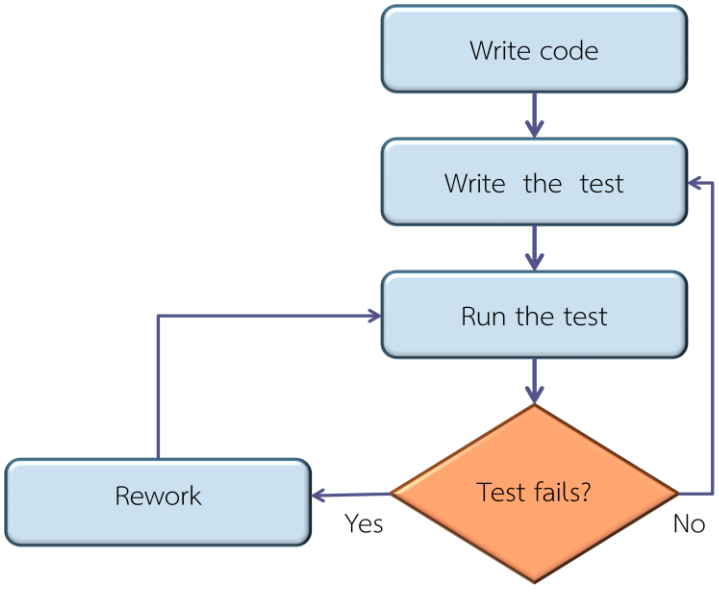
รูปที่ 2.1 เป็นการอธิบายกระบวนการของ TDD โดยขั้นตอนแรกนักพัฒนาซอฟต์แวร์ต้องทำการพัฒนาตัวทดสอบย่อย (Unit Test) สำหรับฟังก์ชันการทำงาน ในขั้นแรกนี้การทดสอบฟังก์ชันจะล้มเหลวเพราะโค้ดของฟังก์ชันยังไม่ได้รับการพัฒนา หลังจากนั้นนักพัฒนาจะต้องพัฒนาโค้ดของฟังก์ชันเพื่อทำให้ผ่านการทดสอบ เนื่องจากฟังก์ชันที่ถูกพัฒนาขึ้นมุ่งเน้นให้ผ่านการทดสอบ ซึ่งอาจจะไม่ได้รับความสนใจในเรื่องของคุณภาพ ดังนั้นนักพัฒนาอาจจะต้องทำการรีแฟคทอริงโค้ดหลังการทดสอบของฟังก์ชันนั้นผ่านการทดสอบเพื่อให้ได้โปรแกรมที่มีคุณภาพที่ดี โดยนักพัฒนาซอฟต์แวร์จะดำเนินการตามขั้นตอนแบบนี้ไปทุกฟังก์ชันจนครบ TDD มีวงจรการทำงานที่สั้นกว่าการพัฒนาซอฟต์แวร์แบบดั้งเดิมที่เป็นกระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกซึ่งประกอบไปด้วยการออกแบบ (Design) การเขียนโปรแกรม (Code) การทดสอบโปรแกรม (Test) และการแก้ไขจุดบกพร่องของโปรแกรม (Debug) เพราะ TDD จะมุ่งเน้นไปที่การสร้างสิ่งที่จำเป็นและใช้ทันทีเพื่อลดข้อผิดพลาดที่จะเกิดขึ้น ซึ่งการใช้การพัฒนาซอฟต์แวร์แบบ TDD ที่เริ่มต้นการพัฒนาซอฟต์แวร์จากตัวทดสอบย่อยจากนั้นนักพัฒนาจึงทำการเขียนโค้ดจริงของระบบ โดยในการพัฒนาตัวทดสอบและการเขียนโค้ดจริงนั้นจะทำให้นักพัฒนาสามารถออกแบบระบบไปในตัวด้วยได้

จากข้างต้นที่กล่าวมาแสดงให้เห็นว่า TDD มีการนำไปใช้ในส่วนของการเขียนโค้ด แต่จากงานวิจัยของ Fraser และคณะเป็นกลุ่มผู้มีประสบการณ์ในด้านเอไจล์และเอกซ์ตรีมโปรแกรมมิ่ง ได้แสดงให้เห็นว่า TDD สามารถนำไปใช้ได้ในหลายรูปแบบ ในงานวิจัยของ Fraser และคณะมีวัตถุประสงค์

เพื่อเป็นการแลกเปลี่ยนความคิดเห็นเกี่ยวกับแนวทางและประโยชน์ของการใช้ TDD ซึ่งได้มีการระบุเป้าหมายของ TDD ไว้ว่าเป็นการเขียนโค้ดที่ไม่มีความซับซ้อน นอกจากนี้ยังมีการนำ TDD ไปประยุกต์ใช้กับงานในส่วนอื่น ๆ นอกเหนือจากการเขียนโค้ดเพียงอย่างเดียว เช่น การใช้ TDD กับฐานข้อมูล (Databases) การใช้ TDD กับหน้าจอการแสดงผล และการใช้ TDD กับระบบประมวลแบบกระจาย (Distributed Computing Systems) จากงานวิจัยของ Fraser และคณะแสดงให้เห็นว่าการใช้ TDD สามารถนำไปประยุกต์ในงานพัฒนาซอฟต์แวร์หลายด้าน (Fraser, *et al.*, 2003)

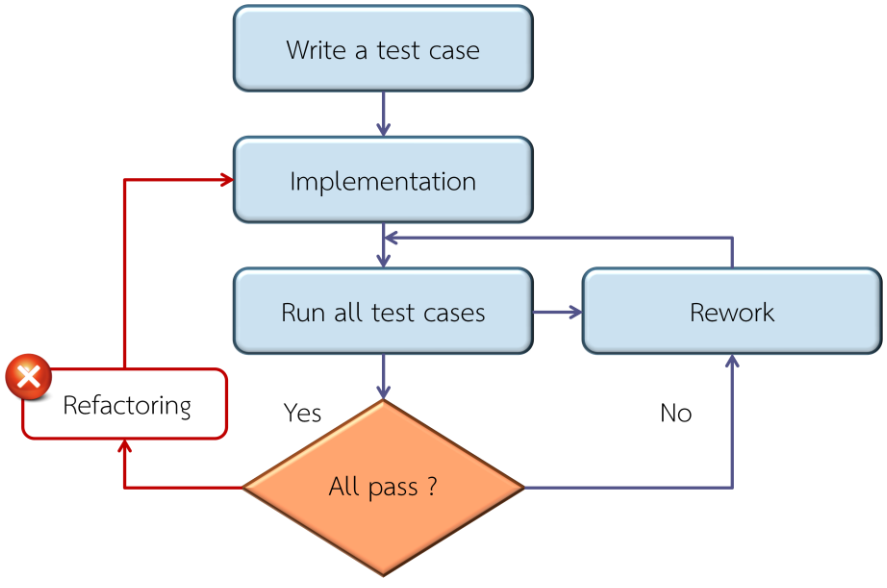
โดยทั่วไปเมื่อกล่าวถึง TDD จะมีความเกี่ยวข้องกับการทดสอบ 2 ลักษณะ คือ 1) การทดสอบแบบ Test-Last และ 2) การทดสอบแบบ Test-First ซึ่งการทดสอบในแต่ละลักษณะมีรายละเอียดดังนี้

(1) การทดสอบแบบ Test-Last เป็นส่วนหนึ่งของการพัฒนาซอฟต์แวร์ในรูปแบบจำลองน้ำตก คือ เริ่มต้นจากการที่ต้องรู้ความต้องการของระบบและทำการออกแบบระบบซอฟต์แวร์ หลังจากนั้นจึงทำการเขียนโปรแกรมตามการออกแบบที่วางไว้ และขั้นตอนสุดท้าย คือ การทดสอบระบบ ซึ่งหากระบบเกิดข้อผิดพลาดจะต้องทำการแก้ไขระบบ โดยการทดสอบระบบใหม่จนกว่าจะผ่าน แสดงดังรูปที่ 2.2 ซึ่งการทดสอบระบบในรูปแบบ Test-Last ผู้พัฒนาซอฟต์แวร์อาจจะมุ่งเน้นเพียงเพื่อให้ซอฟต์แวร์ทำงานได้โดยไม่ติดปัญหาในขณะทำงาน แต่ผู้พัฒนาซอฟต์แวร์อาจจะไม่ได้ทำการทดสอบระบบให้มีความถูกต้องทุกกรณีของการทดสอบ ซึ่งอาจจะทำให้ซอฟต์แวร์เกิดปัญหาขึ้นในอนาคต



รูปที่ 2.2 แสดงกระบวนการทำงานของการทดสอบแบบ Test-Last

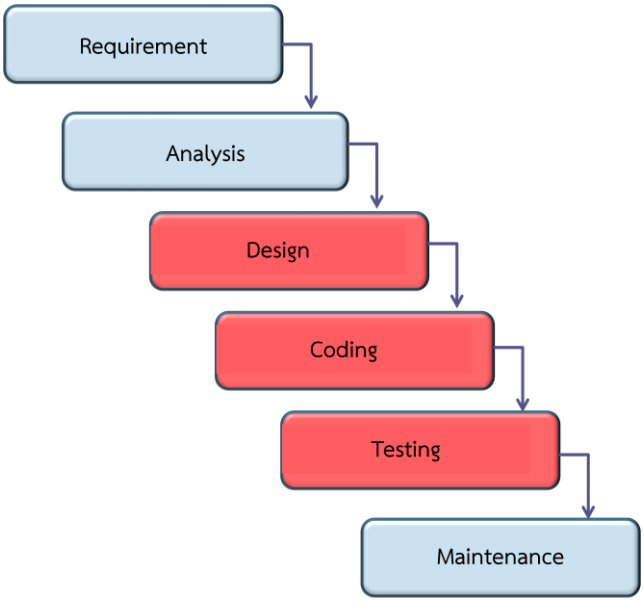
(2) การทดสอบแบบ Test-First เป็นการพัฒนาซอฟต์แวร์รูปแบบหนึ่ง และเป็นหนึ่งในองค์ประกอบสำคัญของ TDD และเอกซ์ตรีมโปรแกรมมิ่ง วิธีการทดสอบแบบ Test-First มีกระบวนการทำงาน แสดงดังรูปภาพที่ 2.3 โดยเริ่มต้นจากการเขียนโค้ดการทดสอบของฟังก์ชันให้เป็น Unit Test สำหรับฟังก์ชันก่อน แล้วจึงทำการเขียนโค้ดของฟังก์ชันจริงซึ่งจะมีการทดสอบซ้ำ เมื่อมีการเพิ่มฟังก์ชันของการทำงานใหม่ เพื่อให้แน่ใจว่าฟังก์ชันที่พัฒนาขึ้นมาใหม่ไม่ส่งผลกระทบต่อการทดสอบก่อนหน้า และเมื่อทำการทดสอบฟังก์ชันทั้งหมดจนผ่านแล้วจึงเริ่มทำฟังก์ชันการทำงานใหม่ วิธีการทดสอบแบบ Test-First แม้จะทำให้ได้ระบบตรงตามความต้องการ แต่ผู้พัฒนาซอฟต์แวร์อาจไม่ได้ทำการปรับปรุงโค้ดที่สามารถทำงานได้ตามฟังก์ชันการทำงานแล้วให้โค้ดมีคุณภาพที่ดีด้วยวิธีการรีแฟคทอริงเหมือนใน TDD



รูปที่ 2.3 แสดงกระบวนการทำงานของการทดสอบแบบ Test-First

2.1.3 กระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก (Waterfall Model)

กระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกเป็นกระบวนการพัฒนาซอฟต์แวร์ที่คิดค้นขึ้นในปี ค.ศ. 1987 โดยมี Winston W.Royce เป็นผู้คิดค้น (Royce, 1987)



รูปที่ 2.4 กระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก

กระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีขั้นตอนการทำงาน แสดงดังรูปที่ 2.4 ซึ่งเริ่มจากการรวบรวมความต้องการ (Requirement) การวิเคราะห์ความต้องการ (Analysis) การออกแบบ (Design) การเขียนโปรแกรม (Coding) การทดสอบ (Testing) และการบำรุงรักษา (Maintenance) ซึ่งแบบจำลองนี้สามารถนำไปประยุกต์ใช้เพื่อการพัฒนาซอฟต์แวร์ให้ตรงตามความต้องการของผู้ใช้ได้ แต่อาจจำเป็นต้องใช้เวลาในการดำเนินงานกับบางขั้นตอน โดยเฉพาะขั้นตอนการวิเคราะห์และออกแบบ ซึ่งปกติต้องได้รับการออกแบบระบบให้เสร็จทั้งหมดก่อน แล้วจึงค่อยลงมือพัฒนาโปรแกรม โดยเมื่อเข้าสู่ขั้นตอนใดแล้วจะไม่มีการย้อนกลับมาทำขั้นตอนก่อนหน้าอีก เช่น เมื่อเข้าสู่ขั้นตอนการเขียนโปรแกรมก็จะพร้อมที่จะนำโปรแกรมไปทดสอบและใช้งานจริง ไม่สามารถกลับไปแก้ไขได้ ซึ่งความจริงแล้วในทางปฏิบัติ นักพัฒนาซอฟต์แวร์อาจจำเป็นต้องย้อนกลับไปทำขั้นตอนก่อนหน้า เช่น นักวิเคราะห์ระบบอาจมองเห็นปัญหาหรือพบข้อผิดพลาดที่เกิดขึ้นภายหลัง หรือมีอะไรบางอย่างจำเป็นต้องได้รับการเปลี่ยนแปลงอย่างเร่งด่วน จำเป็นต้องกลับไปแก้ไขในขั้นตอนก่อนหน้า

ในกระบวนการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกนั้น ก่อนที่ระบบจะได้รับการพัฒนาจำเป็นต้องมีการออกแบบระบบให้เสร็จสมบูรณ์ทั้งหมดก่อน แล้วจึงค่อยดำเนินการเขียนโปรแกรม ซึ่งการพัฒนาซอฟต์แวร์ในปัจจุบันเป็นไปได้ยากที่จะออกแบบโดยรวมทั้งหมดให้มีความสมบูรณ์ในครั้งเดียว อีก

ทั้งขั้นตอนการทดสอบอยู่ในลำดับท้ายในกระบวนการพัฒนาซอฟต์แวร์ ดังนั้นหากระบบได้รับการทดสอบแล้วพบข้อผิดพลาดซึ่งไม่ตรงตามความต้องการของผู้ใช้ ย่อมเกิดปัญหาและส่งผลกระทบต่อโครงการในการพัฒนาซอฟต์แวร์ จึงมีโอกาสต้องวนกลับไปยังขั้นตอนก่อนหน้า คือ ขั้นตอนการเขียนโปรแกรม การออกแบบโปรแกรม หรืออาจต้องย้อนกลับไปยังขั้นตอนแรกที่ต้องเก็บรวบรวมความต้องการใหม่ หรือเมื่อย้อนกลับไปแก้ไขขั้นตอนก่อนหน้านี้อาจใช้เวลาในการแก้ไขนานมากกว่าเดิม ซึ่งทำให้สูญเสียเวลาและค่าใช้จ่าย เนื่องจากเหมือนกับการเริ่มต้นทำใหม่อีกครั้ง

2.1.4 การวิจัยเชิงประจักษ์ด้านวิศวกรรมซอฟต์แวร์ (Empirical Software Engineering Research)

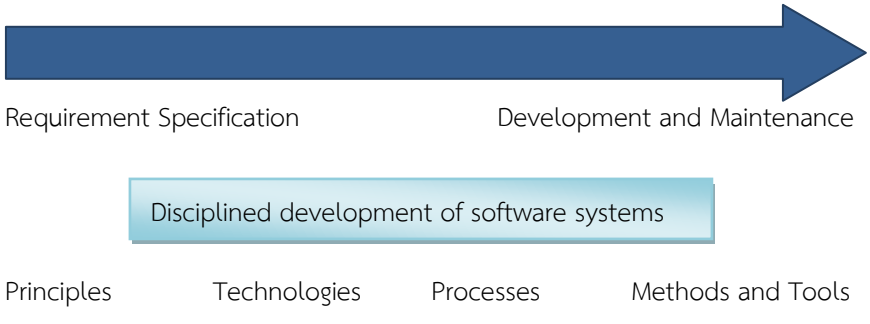
การวิจัยเชิงประจักษ์ในด้านวิศวกรรมซอฟต์แวร์มักจะใช้ในการสำรวจและอธิบายปรากฏการณ์หรือผลที่ได้จากการปฏิบัติ การวิจัยเชิงประจักษ์นี้ช่วยให้นักวิจัยสามารถตรวจสอบทฤษฎีเพื่อระบุปัจจัยที่สำคัญของทฤษฎีที่ทำการวิจัยและช่วยสร้างแบบจำลองที่เกิดขึ้นจากการทำการวิจัย โดยลักษณะของงานวิจัยเชิงประจักษ์แสดงดังรูปที่ 2.5 แสดงให้เห็นว่างานวิจัยที่เป็นลักษณะของงานวิจัยเชิงประจักษ์มีจุดมุ่งหมายในการให้ความสำคัญกับการปรับปรุงกระบวนการในการพัฒนาซอฟต์แวร์และเพื่อเพิ่มคุณภาพซอฟต์แวร์ โดยลักษณะของงานเชิงประจักษ์ด้านวิศวกรรมซอฟต์แวร์มี 4 ลักษณะ คือ

(1) หลักการและทฤษฎีที่เกี่ยวข้องทางด้านวิศวกรรมซอฟต์แวร์

(2) เทคโนโลยีที่ช่วยในการพัฒนาและปรับปรุงคุณภาพซอฟต์แวร์

(3) กระบวนการในการพัฒนาซอฟต์แวร์

(4) วิธีการและเครื่องมือที่ใช้ในการพัฒนาซอฟต์แวร์



รูปที่ 2.5 แสดงลักษณะของงานวิจัยเชิงประจักษ์ด้านวิศวกรรมซอฟต์แวร์

การวิจัยเชิงประจักษ์ในวิศวกรรมซอฟต์แวร์มีวิธีการหลายรูปแบบ นักวิจัยต้องเลือกวิธีการที่เหมาะสมในการวิจัยเพื่อให้เป็นไปตามเป้าหมาย สำหรับวิธีการที่เกี่ยวข้องกับงานวิจัยของผู้วิจัยในครั้งนี้ประกอบไปด้วย 1) การทดลอง (Experiment) 2) การสำรวจ (Survey) และ 3) การสังเกต (Observation) สำหรับในงานวิจัยนี้ ผู้วิจัยไม่ได้ใช้การวิจัยแบบกรณีศึกษา (Case Study) แต่มีความเกี่ยวข้องในส่วนของการทบทวนวรรณกรรม สำหรับรายละเอียดของแต่ละวิธีการมีรายละเอียดดังนี้

(1) การทดลอง (Experiment)

การทดลองในเชิงวิทยาศาสตร์เป็นหนึ่งในการศึกษาพฤติกรรม โดยในการทดลองนี้สามารถแบ่งได้เป็น 2 ประเภท คือ 1) Uncontrolled Experiment และ 2) Controlled Experiment โดยการทดลองในแต่ละประเภทมีรายละเอียดดังนี้

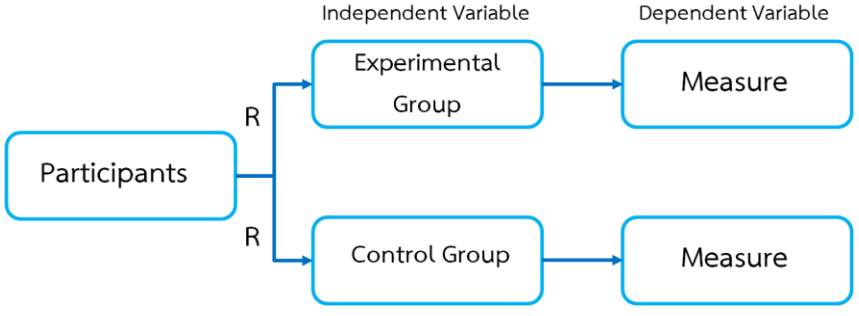
(1.1) Uncontrolled Experiment เป็นการศึกษาโดยการสังเกตการวัดค่าบางอย่างที่สนใจโดยไม่ได้มีการควบคุมตัวแปร การทดลองประเภทนี้ให้ผลลัพธ์ในการวิจัยรวดเร็วแต่ผลลัพธ์ที่ได้อาจจะไม่มีความน่าเชื่อถือมากพอ เช่น นักวิจัยให้ความสนใจว่าทำไมในบริษัทหนึ่งมีโครงการที่มีความล้มเหลวทุกโครงการ นักวิจัยมีหน้าที่ไปสังเกตการณ์ โดยนักวิจัยสังเกตจากการแก้ไขข้อผิดพลาด (Bug) ของบริษัทในแต่ละวัน

(1.2) Controlled Experiment การทดลองที่มีการควบคุมเป็นกระบวนการค้นหาความจริง ทฤษฎีหลักการ เทคโนโลยี หรือองค์ความรู้ใหม่ ๆ เน้นการศึกษาความเปลี่ยนแปลงของตัวแปรที่เกี่ยวข้องภายใต้เงื่อนไขที่มีการควบคุม เพื่อศึกษาพฤติกรรมว่าเป็นสาเหตุของการเปลี่ยนแปลงหรือไม่ โดยใช้วิธีการเปรียบเทียบความแตกต่างของตัวแปรที่เปลี่ยนระหว่างพฤติกรรมในสภาพปกติกับพฤติกรรมที่เกิดขึ้นในสภาพที่ถูกควบคุม เพื่อสรุปสิ่งที่ค้นพบและนำไปใช้อธิบายพฤติกรรมในเชิงเหตุผลได้อย่างชัดเจน การวิจัยเชิงทดลองจึงเป็นการศึกษาวิจัยจากสาเหตุไปหาผลลัพธ์ เพื่อหาความสัมพันธ์เชิงเหตุผลของปรากฏการณ์ต่าง ๆ ซึ่งได้รับการยอมรับว่าเป็นการวิจัยที่ให้ผลความเชื่อถือดีที่สุด การทดลองลักษณะนี้มีการควบคุมตัวแปรต่าง ๆ โดยมีตัวแปรหนึ่งได้รับการควบคุม ส่วนอีกกลุ่มหนึ่งไม่มีการควบคุม เป้าหมายหลักของการทำการทดลองรูปแบบนี้ คือ การลดตัวแปรที่ไม่มีความเกี่ยวข้องหรือตัวแปรกวน (Confounding Variable) เพื่อเป็นการควบคุมสิ่งที่ไม่เกี่ยวข้องภายในงานที่กำลังทำการศึกษา เช่น การทำโครงการเดียวกัน 2 กลุ่ม โดยให้กลุ่มที่ 1 เป็นกลุ่มประชากรที่เป็นผู้ไม่มีประสบการณ์เป็นกลุ่มที่มีการควบคุม และกลุ่มที่ 2 เป็นกลุ่มที่มีประสบการณ์มาแล้ว 5 ปี เป็นกลุ่มที่ไม่มีการควบคุม ในกลุ่มที่ไม่มีการควบคุมต้องทำให้มั่นใจว่าจะไม่มีผลกระทบอื่นที่เกิดขึ้นโดยที่เป็นไปตามธรรมชาติ ในการทดลองพื้นฐาน

โดยทั่วไปมี 2 ลักษณะ คือ 1) Posttest-Only Design และ 2) Pretest-Posttest Design โดยแต่ละแบบมีรายละเอียดดังนี้

(1.2.1) Posttest-Only Design

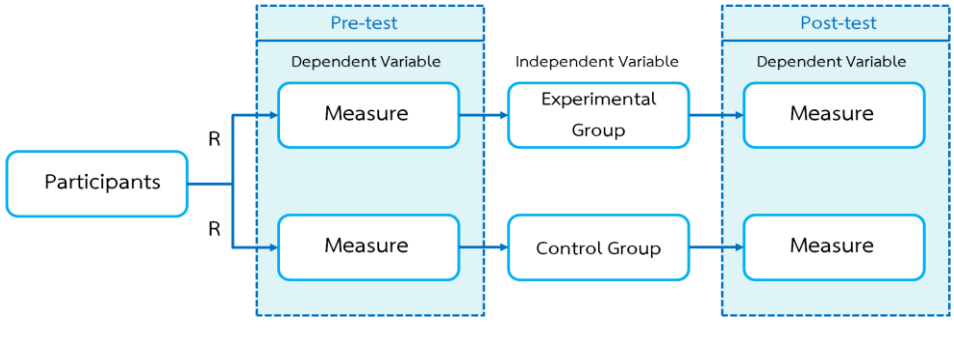
การทดลองแบบ Posttest-Only Design มีกลุ่มการทดลอง 2 กลุ่ม การทดลอง คือ กลุ่มควบคุม (Control Group) และกลุ่มทดลอง (Experimental Group) หลังจากได้รับการทดลองตามเงื่อนไขการทดลองแล้วมีการวัดผลหลังการทดลอง ข้อมูลที่ได้จึงมีเพียงค่าของตัวแปรตาม โดยลักษณะของการทดลองแบบ Posttest-Only Design ซึ่งแสดงดังรูปที่ 2.6 เมื่อทำการวัดผลหลังการทดลองถ้าผลที่ได้จากการวัดเหมือนกันแสดงว่าสิ่งที่ต้องการวัดไม่มีผล



รูปที่ 2.6 แสดงการทดลองพื้นฐานแบบการทดสอบหลังการทดลองเท่านั้น

(1.2.2) Pretest-Posttest Design

การทดลองแบบ Pretest-Posttest Design มุ่งเน้นการดำเนินการทดลองกับกลุ่มทดลอง 2 กลุ่ม คือ กลุ่มควบคุม (Control Group) และกลุ่มทดลอง (Experimental Group) ซึ่งแสดงดังรูปที่ 2.7 โดยมีการดำเนินการสังเกตผู้เข้าร่วมการทดลองก่อนและหลังการทดลอง หลังจากนั้นจึงนำผลที่ได้จากการวัดหรือการสังเกตไปเปรียบเทียบกันเพื่อทดสอบดูว่าแตกต่างกันหรือไม่



รูปที่ 2.7 แสดงการทดลองพื้นฐานแบบการทดสอบก่อนและหลังการทดลอง

(2) การสำรวจ (Survey)

การสำรวจเป็นการเก็บรวบรวมข้อมูลที่ได้จากประชากรที่อยู่ในความสนใจ หรือกลุ่มตัวอย่างซึ่งเป็นตัวแทนของประชากรทั้งหมด วิธีการสำรวจนี้สามารถกระทำได้โดยใช้วิธีการสัมภาษณ์ หรือการตอบแบบสอบถาม เพื่อเป็นการอธิบายเปรียบเทียบ หรืออธิบายทัศนคติและพฤติกรรม (Abramson and Abramson, 2008)

จากการศึกษางานวิจัยที่เกี่ยวกับการสำรวจได้กำหนดขั้นตอนในการดำเนินงานวิจัยเชิงสำรวจ (Mukherji and Albon, 2014) เป็นลำดับขั้นตอนดังนี้

(2.1) การระบุวัตถุประสงค์ของการวิจัย โดยนักวิจัยจะต้องระบุถึงปัญหาของงานวิจัยและระบุวิธีการในการสำรวจเพื่อตอบโจทย์ปัญหาที่ตั้งขึ้น

(2.2) การระบุกลุ่มเป้าหมาย นักวิจัยต้องระบุกลุ่มเป้าหมายที่เป็นกลุ่มเป้าหมายหลักของงานวิจัย ในทางปฏิบัติผู้วิจัยไม่สามารถสำรวจประชากรได้ทั้งหมด จึงต้องเลือกตัวอย่างที่สามารถเป็นตัวแทนของประชากรทั้งหมด

(2.3) การสุ่มตัวอย่าง ในขั้นตอนนี้นักวิจัยควรตัดสินใจเลือกขนาดกลุ่มตัวอย่างตามที่ต้องการและเลือกลักษณะผู้ตอบแบบสำรวจ

(2.4) การออกแบบสอบถาม ผู้วิจัยต้องออกแบบและพัฒนาแบบสอบถามที่สามารถนำข้อมูลจากประชากรมาตอบคำถามวิจัยได้

(2.5) การดำเนินการศึกษานำร่อง เมื่อแบบสอบถามได้รับการพัฒนาเสร็จสิ้นนักวิจัยควรทำการศึกษานำร่องเพื่อตรวจสอบคำถาม ซึ่งนักวิจัยสามารถทำการสำรวจกลุ่มตัวอย่างที่เป็นตัวแทนของกลุ่มการศึกษาจริง โดยการศึกษานำร่องมีวัตถุประสงค์ คือ เป็นการตรวจสอบความถูกต้องของแบบสอบถาม

(2.6) การกระจายแบบสอบถาม เมื่อนักวิจัยมั่นใจว่าแบบสอบถามได้รับการตรวจสอบแล้วจากการศึกษานำร่องและผ่านการประเมินยืนยันทางสถิติถึงความสอดคล้องของแบบสอบถามที่ผู้วิจัยได้จัดทำขึ้น หลังจากนั้นผู้วิจัยจึงทำการกระจายแบบสอบถามให้ผู้ตอบแบบสอบถาม โดยแบบสอบถามสามารถอยู่ในรูปแบบที่เป็นกระดาษหรืออิเล็กทรอนิกส์ เช่น อีเมล เว็บไซต์ นักวิจัยอาจจะส่งหนังสือแจ้งไปยังกลุ่มตัวอย่างก่อนที่จะส่งแบบสอบถามไปยังกลุ่มตัวอย่าง นอกจากนี้นักวิจัยอาจต้องทำการแจ้งเตือนในเวลาที่เหมาะสม เพื่อขอให้ผู้ตอบแบบสอบถามส่งแบบสอบถามแก่ผู้วิจัยเพื่อให้ผู้วิจัยทำการวิเคราะห์และสรุปผลเพื่อจัดทำรายงานแสดงผลของงานวิจัย

(3) การสังเกต (Observation)

การสังเกตเป็นวิธีการหนึ่งที่ช่วยให้นักวิจัยเข้าใจพฤติกรรมของผู้เข้าร่วมวิจัยในขั้นตอนการทดลองแบบควบคุมได้ดียิ่งขึ้น วิธีการนี้เป็นการสังเกตพฤติกรรม ปรากฏการณ์ที่เกิดขึ้นในกิจกรรมหรือการทดลอง ผู้วิจัยทำการเก็บข้อมูลพฤติกรรมหรือลักษณะการแสดงออกทางอารมณ์ของผู้เข้าร่วม ซึ่งสามารถเกิดขึ้นได้ในระยะเวลาไม่จำกัดตัวอย่าง เช่น การสังเกตพฤติกรรมของนักพัฒนาซอฟต์แวร์ในการทำงานเป็นทีมภายใต้เงื่อนไขบางอย่าง การเก็บข้อมูลนี้สามารถทำได้โดยการจดบันทึกหรือบันทึกโดยเครื่องมืออิเล็กทรอนิกส์อื่น ๆ เช่น เครื่องบันทึกภาพ ซึ่งข้อดีของการสังเกต คือ สามารถกระทำได้ง่ายและจะให้ผลลัพธ์ที่รวดเร็ว แต่การสังเกตมีข้อจำกัด คือ มักจะเป็นเรื่องยากที่จะเห็นพฤติกรรมบางพฤติกรรม เช่น ในกรณีที่ผู้ที่โดนสังเกตการณ์มีการใช้แป้นพิมพ์ลัดในการออกคำสั่งและการทำงาน หากเป็นการสังเกตทั่วไปที่สามารถสังเกตที่ได้ชัดเจน เช่น รู้ว่านักพัฒนาซอฟต์แวร์กำลังทำการแก้ปัญหา การใช้การสังเกตการณ์สามารถทำให้สามารถเก็บข้อมูลได้ดีโดยผู้สังเกตการณ์ต้องมีความเข้าใจในสภาพแวดล้อมหรือเงื่อนไขที่ผู้ถูกสังเกตทำงานอยู่

(4) กรณีศึกษา (Case Study)

การวิจัยเชิงกรณีศึกษาเป็นวิธีการหนึ่งที่ใช้ตอบคำถามที่มุ่งเน้นหาเหตุผลของผลลัพธ์ วิธีการ หรือกระบวนการที่ก่อให้เกิดผลลัพธ์ซึ่งต่างจากวิธีวิจัยอื่น ๆ เนื่องจากเป็นการดำเนินงานในสภาพแวดล้อมจริงไม่มีการควบคุมตัวแปรใด ๆ เช่นเดียวกับวิธีวิจัยเชิงการทดลอง อีกทั้งการวิจัยเชิงกรณีศึกษายังมุ่งเน้นถึงการศึกษาเชิงลึก (In-depth Study) เพื่อทำความเข้าใจในเรื่องใดเรื่องหนึ่งซึ่งต่างจากการวิจัยแบบสำรวจและการวิจัยแบบการใช้ข้อมูลที่บันทึก ซึ่งงานวิจัยสองแบบหลังนี้มุ่งตอบคำถามเกี่ยวกับ ใคร อะไร ที่ไหน หรือไม่ โดยใช้ข้อมูลเชิงปริมาณและเครื่องมือทางสถิติในการวิเคราะห์และสรุปผล งานวิจัยเชิงกรณีศึกษาสามารถแบ่งตามวัตถุประสงค์การศึกษาได้เป็น 3 ประเภท ตามหนังสือ Case study research: Design and methods (Yin, 2013) ดังนี้

(4.1) กรณีศึกษาเพื่อค้นหา (Exploratory Case Study)

การวิจัยแบบกรณีศึกษาเพื่อค้นหาเป็นวิธีที่ที่มักจะใช้เป็นส่วนใหญ่ในงานวิจัยแบบกรณีศึกษา เนื่องจากงานวิจัยเชิงกรณีศึกษานอกจากจะเหมาะสมในการตอบคำถามว่าทำไมหรืออย่างไรแล้วยังเหมาะสำหรับการศึกษาเรื่องที่ยังใหม่อยู่ เรื่องที่อยู่ในช่วงเริ่มต้น มีผู้ศึกษาน้อยหรืออาจจะทราบข้อมูลน้อยเกี่ยวกับเรื่องบางอย่าง (Benbasat, *et al.*, 1987) โดยกรณีศึกษาเพื่อค้นหามี

วัตถุประสงค์หลักเพื่อเป็นการค้นหาทฤษฎี หรือคำอธิบายเกี่ยวกับเรื่องใดเรื่องหนึ่งก็ได้ งานวิจัยประเภทนี้ถือเป็นส่วนหนึ่งของการตั้งคำถามของงานวิจัย ซึ่งอาจจะต้องใช้วิธีวิจัยอื่นเพื่อช่วยตอบคำถามที่ตั้งขึ้น

(4.2) กรณีศึกษาเพื่ออธิบาย (Explanatory Case Study)

การวิจัยประเภทกรณีศึกษาเพื่ออธิบาย เป็นการอธิบายถึงเหตุและผลของสิ่งที่เกิดขึ้น โดยเฉพาะอย่างยิ่ง ในสภาวะแวดล้อมที่ซับซ้อน

(4.3) กรณีศึกษาเพื่ออภิปราย (Descriptive Case Study)

การวิจัยประเภทกรณีศึกษาเพื่ออภิปรายเป็นการแสดงให้เห็นสิ่งที่เกิดขึ้นอย่างละเอียดในงานที่ทำการศึกษา โดยไม่มุ่งเน้นที่จะแสดงให้เห็นความเป็นเหตุเป็นผลของตัวแปรต่าง ๆ ในงานวิจัย

นอกจากนั้นการวิจัยเชิงกรณีศึกษายังอาจแบ่งได้ตามระยะเวลาในการศึกษาวิจัยและวิเคราะห์เช่นเดียวกับวิธีวิจัยอื่น ๆ ดังนี้

(4.4) กรณีศึกษา ณ ช่วงเวลาหนึ่ง (Snapshot Case Study) ซึ่งเลือกศึกษาเฉพาะช่วงเวลาที่กำหนด

(4.5) กรณีศึกษาแบบช่วงเวลา (Longitudinal Case Study) เป็นการศึกษาสิ่งใดสิ่งหนึ่งในช่วงเวลาที่ต่างกัน ซึ่งอาจมีการนำผลการศึกษามาวิเคราะห์เปรียบเทียบให้เห็นถึงความเปลี่ยนแปลง การศึกษาในรูปแบบนี้มีการรวบรวมข้อมูลเชิงปริมาณเพื่อใช้วิเคราะห์ผล

(4.6) กรณีศึกษาที่ศึกษาเหตุการณ์ก่อนและหลัง (Pre-post Case Study) กรณีศึกษาแบบนี้จะเลือกเอาเหตุการณ์สำคัญหนึ่ง ๆ ซึ่งมีผลในการอธิบายหรือหักล้างทฤษฎีที่เกี่ยวเนื่องมาใช้คั่นการศึกษา

(4.7) กรณีศึกษาแบบผสม (Patchwork Case Study) เป็นการออกแบบการศึกษาวิจัยแบบผสมมีการนำข้อมูลเฉพาะจุดและข้อมูลช่วงเวลา รวมไปถึงข้อมูลก่อนและหลังเหตุการณ์สำคัญมาใช้ประกอบเพื่ออธิบายสิ่งที่ต้องการศึกษา

(4.8) กรณีศึกษาเปรียบเทียบ (Comparative Case Study) การศึกษาสิ่งใดสิ่งหนึ่งที่ผู้วิจัยต้องการทำการศึกษาในช่วงเวลาเดียวกัน เพื่อนำมาเปรียบเทียบหาความเหมือนและความแตกต่าง กรณีศึกษารูปแบบนี้ใกล้เคียงกับการวิจัยเฉพาะช่วงเวลาในกลุ่มตัวอย่าง (Cross-sectional Research)

2.1.5 คำนิยามเรื่องความสนใจ (Interest)

เนื่องจากในงานวิจัยนี้มีความเกี่ยวข้องกับการกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ ซึ่งงานวิจัยในด้าน “ความสนใจ” มีการนิยามเรื่อง “ความสนใจ” ไว้ค่อนข้างหลากหลายและยังไม่พบตัวบ่งชี้ที่เป็นมาตรฐานทั้งในทางด้านความสนใจในทางจิตวิทยาและความสนใจในทางวิศวกรรมซอฟต์แวร์ ผู้วิจัยจึงต้องทำการศึกษานิยามและงานวิจัยที่เกี่ยวข้องด้านความสนใจเพื่อนำไปกำหนดตัวบ่งชี้ภายในงานวิจัย ผู้วิจัยได้แบ่งการศึกษาออกเป็น 2 ส่วน คือ 1) ความหมายของความสนใจ และ 2) งานวิจัยที่เกี่ยวข้องกับความสนใจ ซึ่งมีรายละเอียดดังนี้

(1) ความหมายของความสนใจ

จากการทำการศึกษานิยามที่เกี่ยวข้องกับด้านความสนใจจากนักวิจัย นักการศึกษา และนักจิตวิทยาหลายท่านได้อธิบายถึงความหมายของความสนใจไว้ดังตารางที่ 2.1

ตารางที่ 2.1 การศึกษานิยามด้านความสนใจ

คณะผู้วิจัย	นิยามด้านความสนใจ
เสถียร ศรีรัตน์ (2522)	ความสนใจเป็นความรู้สึกที่ใจจดจ่อต่อบุคคลที่มีต่อสิ่งใดสิ่งหนึ่ง บุคคลใดบุคคลหนึ่ง หรือสถานการณ์ใดสถานการณ์หนึ่งอันมีแนวโน้มให้อยากเข้าใกล้อยากรู้อยากเห็นกับสิ่งนั้นบุคคล หรือสถานการณ์นั้นความสนใจของแต่ละบุคคลจึงมีอิทธิพลอย่างมากต่อการเรียนรู้ที่จะได้รับจากประสบการณ์ต่าง ๆ ไปละครั้ง
สุชา จันทร์เอม (2542)	ความสนใจเป็นความรู้สึก หรือทัศนคติที่มีต่อสิ่งหนึ่งสิ่งใดโดยเฉพาะและความรู้สึกนั้นทำให้บุคคลเอาใจใส่และกระทำการจนบรรลุถึงความมุ่งหมายที่บุคคลมีต่อสิ่งนั้น บ้างก็กล่าวว่าความสนใจหมายถึง ความสะเทือนใจอันเกิดจากปฏิกิริยาระหว่างสิ่งเร้ากับร่างกาย กล่าวคือ สิ่งเร้าจากการกระตุ้นร่างกายก่อให้เกิดความสะเทือนใจอย่างแรงจะทำให้เกิดความตั้งใจจดจ่อต่อสู้กับสิ่งเร้านั้น
รัตนา คงคาเนาวรัตน์ (2542)	ความสนใจเป็นความรู้สึกที่บุคคลแต่ละคนมีความรักชอบต่อสิ่งใดสิ่งหนึ่ง และพร้อมที่จะเข้าร่วมกิจกรรมที่ให้ความสนใจ

ตารางที่ 2.1 การศึกษานิยามด้านความสนใจ (ต่อ)

คณะผู้วิจัย	นิยามด้านความสนใจ
จิราพร สุจริต (2543)	ความสนใจเป็นความรู้สึกชอบ หรือพอใจเอาใจใส่กระตือรือร้นตลอดจนความอยากรู้อยากเห็น ความอยากมีส่วนร่วมและเข้าร่วมกิจกรรมซึ่งความสนใจเกิดจากสภาพแวดล้อมซึ่งส่งผลให้บุคคลเกิดความพากเพียรพยายามให้เกิดความสำเร็จ
จิตมณี อะเมกอง (2545)	ความสนใจเป็นการที่บุคคลแสดงความรู้สึกชอบพอใจเอาใจใส่ต่อสิ่งใดสิ่งหนึ่งอยากรู้อยากเห็นอยากแสวงหาคำตอบและซาบซึ้งรวมทั้งเห็นคุณค่าต่อสิ่งใดสิ่งหนึ่ง ซึ่งเป็นจุดเริ่มต้นของการเรียนรู้นั่นเอง
Charters and Good (1945)	ความสนใจเป็นความรู้สึกชอบของคนเราที่แสดงออกต่อสิ่งใดสิ่งหนึ่ง ซึ่งความรู้สึกนี้อาจจะมีชั่วขณะหนึ่ง หรืออาจจะมีถาวรต่อไปก็ได้ขึ้นอยู่กับความอยากรู้อยากเห็นของบุคคลนั้นโดยมีอิทธิพลจากประสบการณ์ของบุคคลนั้น
Dwyer (1993)	ความสนใจเป็นความรู้สึกความพึงพอใจที่บุคคลมีต่อสิ่งใดสิ่งหนึ่ง แนวคิดใดแนวคิดหนึ่ง หรือกิจกรรมใดกิจกรรมหนึ่ง

จากการศึกษาเรื่องความสนใจ สามารถเขียนเป็นนิยามได้ว่า *ความสนใจเป็นความรู้สึกถึงความพึงพอใจหรือทัศนคติที่มีต่อสิ่งหนึ่ง แนวคิดใดแนวคิดหนึ่ง หรือสถานการณ์ใดสถานการณ์หนึ่งอันมีแนวโน้มให้อยากเข้าใกล้อยากรู้อยากเห็นสิ่งนั้น ซึ่งทำให้เกิดความเพียรพยายามและสามารถกระทำจนบรรลุจุดมุ่งหมาย ซึ่งความสนใจเป็นแรงผลักดันที่กระตุ้นให้บุคคลกระทำการใด ๆ หรือแสดงถึงแนวโน้มที่ตัวบุคคลจะทำการเลือก ซึ่งในด้านของการศึกษาความสนใจมีความสัมพันธ์กับความสำเร็จในด้านการเรียนวิชาการ และเป็นองค์ประกอบหนึ่งในการพัฒนาความสามารถในการประกอบอาชีพ* (เสถียร ศรีรัตน์, 2522; รัตนา คงคาเนาวรัตน์, 2542; สุชา จันทร์เอม, 2542; จิราพร สุจริต, 2543; จิตมณี อะเมกอง, 2545; Charters and Good, 1945; Dwyer, 1993)

(2) งานวิจัยที่เกี่ยวข้องกับความสนใจ

งานวิจัยที่เกี่ยวข้องทางด้านความสนใจ เป็นการศึกษางานวิจัยทางด้านจิตวิทยาและทางด้านวิศวกรรมซอฟต์แวร์ที่มีการนำงานทางด้านจิตวิทยาไปใช้ เพื่อศึกษาความเป็นมาของการศึกษาด้านความสนใจ นิยามด้านความสนใจ และการกำหนดตัวบ่งชี้ทางด้านความสนใจ ซึ่งจากการศึกษางานวิจัย ผู้วิจัยได้สรุปงานวิจัยที่ทำการศึกษาโดยมีรายละเอียดดังนี้

งานวิจัยของ Schiefele และคณะได้ทำการรวบรวมเรื่องที่มีความเกี่ยวข้องกับแนวคิดเรื่อง “ความสนใจ” (Schiefele, 1991) ซึ่งเป็นเรื่องที่ได้รับการถกเถียงกันถึงโครงสร้างของแรงจูงใจและความหมายของ “ความสนใจ” โดยใช้ทฤษฎีการศึกษาของประเทศสหรัฐอเมริกาและเยอรมัน จากแนวคิดทฤษฎีการศึกษามีการระบุไว้ว่า “ความสนใจ” ถูกกำหนดให้เป็นลักษณะในการสร้างแรงจูงใจที่เป็นสิ่งกระตุ้นให้เกิดความสนใจ โดยมีผลจากการศึกษาเพื่อสนับสนุนถึงความสำคัญในเรื่อง “ความสนใจ” สำหรับการทำความเข้าใจเรื่อง “ความสนใจ” ในเชิงลึกมีการศึกษาโดยใช้วิธีการเรียนรู้และประสบการณ์ทางด้านอารมณ์ในขณะที่ทำการเรียนรู้ ซึ่งจากการศึกษางานวิจัยของ Schiefele และคณะสามารถสรุปการนำเสนอเรื่อง “ความสนใจ” ได้ดังนี้

(2.1) ความเป็นมาของการศึกษาเรื่องความสนใจ: เกิดจากการที่มีนักวิจัยหลายท่านให้ความสนใจถึงความคลุมเครือของความหมายของ “ความสนใจ” โดย “ความสนใจ” เป็นอารมณ์ที่ได้รับการตรวจสอบอย่างละเอียดทางจิตวิทยาที่สามารถสร้างแรงจูงใจ แต่งานวิจัยเกี่ยวกับการสร้างแรงจูงใจไม่ได้มีการศึกษาเรื่องความสนใจบางมุมมอง สำหรับ “ความสนใจ” ที่มีการกล่าวถึงในทฤษฎีของการสร้างแรงจูงใจ (Deci and Ryan, 1985; Deci, 1992) ได้มีการอ้างถึง “ความสนใจ” ไว้ว่าความสนใจมีบทบาทสำคัญในการกระตุ้นพฤติกรรมของมนุษย์

(2.2) แนวคิดในการศึกษาเรื่องความสนใจ: ได้มีการศึกษามาเป็นระยะเวลานานในด้านจิตวิทยา โดยมีผู้ที่ทำการศึกษาแนวคิดด้าน“ความสนใจ” ไว้ดังนี้

(2.2.1) แนวคิดในการศึกษาเรื่องความสนใจของ Herbart (Herbart, 1806, 1841) ซึ่งเป็นหนึ่งในผู้บุกเบิกงานทางด้านจิตวิทยาที่ทำการศึกษาแนวคิดเรื่อง “ความสนใจ” ซึ่งในมุมมองของ Herbart มองว่าความสนใจจะต้องมีความเกี่ยวข้องกับการเรียนรู้

(2.2.2) แนวคิดในการศึกษาเรื่องความสนใจของ Dewey (Dewey, 1913, 1997, 2007) ซึ่งเป็นผู้บุกเบิกงานวิจัยเกี่ยวกับความสนใจในหนังสือชื่อ “Interest an Effort in

Education” (Dewey, 1913) เป็นการศึกษาเรื่องความสนใจโดยการเปรียบเทียบระหว่างการเรียนรู้ที่มุ่งเน้นความสนใจและการเรียนรู้ที่ละเลยความสนใจของนักเรียน จากการศึกษาของ Dewey ที่ทำการเปรียบเทียบระหว่างความสนใจในการเรียนรู้ 2 ลักษณะ มีความแตกต่างกันในเชิงคุณภาพจากผลของการเรียน

(2.2.3) แนวคิดในการศึกษาเรื่องความสนใจของ James ได้ทำการศึกษาเรื่องความสนใจและการเปลี่ยนแปลงในทางจิตวิทยา (James, 2013) ซึ่ง James กล่าวว่า “ความสนใจ” เป็นแรงกระตุ้นในจิตใจของมนุษย์ จากการศึกษาทางด้าน “ความสนใจ” ของ James ไม่ได้มีการพัฒนาทฤษฎีความสนใจ จึงทำให้งานของ Dewey ได้รับการยอมรับมากกว่างานของ James เนื่องจากเป็นงานที่มีความเกี่ยวข้องกับการสร้างแนวคิดของความสนใจมากกว่า

(2.2.4) แนวคิดเรื่องการศึกษาความสนใจของ Schiefel กล่าวไว้ว่า “ความสนใจ” เป็นสิ่งที่ทำให้เกิดแรงจูงใจที่มีความเกี่ยวข้องกับการศึกษา โดยแนวคิดของ Schiefel ได้มีการตีพิมพ์เป็นหนังสือชื่อ “Learn motivation and Motive learner Motivation to Learn and Acquisition of Motives” โดย Schiefel ได้ยืนยันแนวคิดเรื่องความสนใจว่า “ความสนใจ” เป็นแรงจูงใจที่ทำให้นักศึกษามีความมุ่งมั่นในการเรียนอย่างมีประสิทธิภาพ โดยมีความเป็นไปได้ว่านักศึกษาที่มีความสนใจในเรื่องนั้น ๆ จะพยายามเรียนรู้ โดย “ความสนใจ” มีคุณสมบัติดังต่อไปนี้

(1) ความสนใจเป็นสิ่งที่สามารถอธิบายทางเลือกของนักศึกษาที่มีความมุ่งมั่นในการปฏิบัติงาน

(2) ความสนใจมีบทบาทสำคัญเนื่องจากเป็นปัจจัยที่ใช้ในการอธิบายทฤษฎีทัศนะของครู

(3) ความสนใจไม่ได้เป็นลักษณะทางด้านกายภาพ

(4) ความสนใจเป็นเรื่องที่นักศึกษามีโอกาสคล้อยตามมากขึ้นในการเรียนการสอน

สำหรับคำจำกัดความเรื่อง “ความสนใจ” ได้มีนักวิจัยทำการศึกษาและสรุปไว้ว่า “ความสนใจ” มีความสำคัญใน 2 แนวคิดที่แตกต่างกันระหว่าง “ความสนใจ” ของแต่ละบุคคลและ “ความสนใจ” ภายใต้สถานการณ์ โดย “ความสนใจ” ของแต่ละบุคคลจะเป็นความรู้สึกที่เป็นความชอบที่ค่อนข้างมีความยั่งยืน ส่วน “ความสนใจ” ภายใต้สถานการณ์เป็นสภาวะอารมณ์ที่มีความ

เกี่ยวข้องกับสิ่งเร้าต่อสภาวะแวดล้อม ซึ่งแนวคิดทั้ง 2 นี้มีความเกี่ยวข้องในการกระตุ้นความสนใจและผลกระทบของความสนใจ

งานวิจัยของ Graziotin และคณะได้ทำการศึกษาผลกระทบต่ออารมณ์ความรู้สึกที่มีต่อความรู้ความเข้าใจในการประมวลผลและประสิทธิภาพการทำงานของบุคคล ซึ่งเป็นทฤษฎีทางด้านจิตวิทยาและการวัดผลในด้านวิศวกรรมซอฟต์แวร์เรียกว่า “จิตวิทยาวิศวกรรมซอฟต์แวร์เชิงประจักษ์ (Psychoempirical Software Engineering)” (Graziotin, *et al*., 2015) โดยผู้วิจัยมักพบงานวิจัยด้านวิศวกรรมซอฟต์แวร์ที่มีความเข้าใจผิดเกี่ยวกับผลกระทบต่อนักพัฒนา เนื่องจากยังขาดทฤษฎีและวิธีการในการวัดผลทางจิตวิทยา โดยภายในงานวิจัยของ Daniel และคณะ มีการศึกษา 3 หัวข้อ คือ

(1) จุดเด่นในการดำเนินการศึกษาที่ส่งผลกระทบทางด้านจิตวิทยา

(2) การทบทวนวรรณกรรมภายในงานวิจัยที่ส่งผลกระทบต่อทฤษฎี

(3) การนำเสนอแนวทางในการดำเนินการทางจิตวิทยาวิศวกรรมซอฟต์แวร์เชิงประจักษ์

งานวิจัยของ Krapp และคณะซึ่งเป็นงานวิจัยของกลุ่มนักวิจัยที่แสดงความคิดเห็นเรื่องความสนใจ (Krapp, 1999) โดยเฉพาะอย่างยิ่งในสาขาจิตวิทยาการศึกษา จากการดำเนินการวิเคราะห์การเรียนรู้และปัจจัยที่สร้างแรงบันดาลใจและความรู้ความเข้าใจซึ่งมีการเชื่อมต่อกับความสนใจของแต่ละบุคคล โดยความสนใจเป็นตัวแปรอิสระที่ขึ้นกับตัวแปรบางส่วนของผลการเรียน ซึ่งตัวแปรที่เกิดจากผลของการเรียนสามารถนำไปอธิบายผลกระทบเกี่ยวกับความสนใจ เช่น การเรียนรู้กลยุทธ์ให้ความสนใจ ประสบการณ์ทางอารมณ์

2.1.6 การแสดงออกทางอารมณ์ (Emotional and Mood)

อารมณ์เป็นสิ่งที่บุคคลแสดงออกแต่ไม่สามารถสัมผัสหรือสังเกตเห็นได้อย่างชัดเจน แต่สามารถทราบถึงสภาวะทางอารมณ์ของบุคคลจากสภาวะแวดล้อมรอบข้าง เช่น สังเกตจากพฤติกรรมการแสดงออกทางการเคลื่อนไหวร่างกายหรือการแสดงออกทางสีหน้า จากการศึกษาทฤษฎีด้านอารมณ์ผู้วิจัยได้แบ่งการศึกษาออกเป็น 2 ส่วน คือ 1) ทฤษฎีการแสดงออกทางอารมณ์ และ 2) งานวิจัยที่เกี่ยวข้องกับการวิเคราะห์อารมณ์ โดยมีรายละเอียดดังนี้

(1) ทฤษฎีการแสดงออกทางอารมณ์

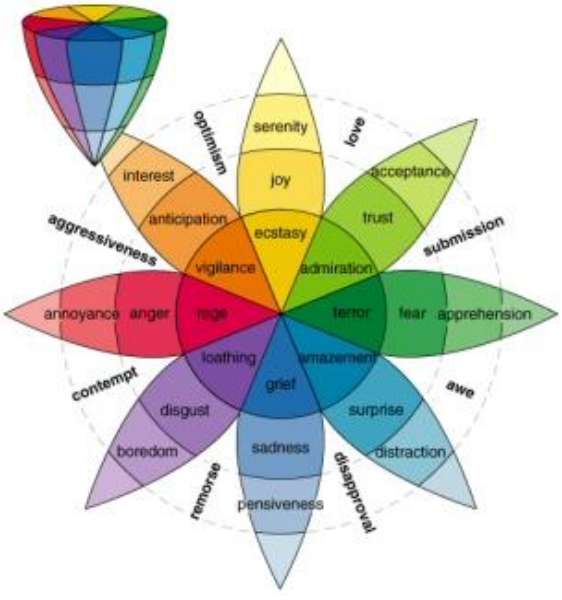
อารมณ์มีหลายชนิดจากการนำเสนอของนักจิตวิทยา โดยการแสดงออกทางอารมณ์ที่มีลักษณะเป็นสากล คือ การแสดงออกทางใบหน้า จากการศึกษามีการจัดประเภทของอารมณ์ของนักจิตวิทยาหลายท่าน เช่น

(1.1) การกำหนดประเภทของอารมณ์ของนักจิตวิทยา

(1.1.1) Izard นักจิตวิทยาได้จำแนกอารมณ์ออกเป็น 10 ประเภท คือ 1) Interest-excitement เป็นอารมณ์ที่ช่วยให้บุคคลเกิดแรงจูงใจในการเรียนรู้ 2) Joy เป็นอารมณ์ที่ก่อให้เกิดความเชื่อมั่น 3) Surprise เป็นอารมณ์ที่ก่อให้เกิดความเปลี่ยนแปลงในทางที่ดีอย่างฉับพลัน 4) Distress-anguish เป็นอารมณ์ที่เกิดขึ้นเมื่อบุคคลต้องประสบหรือพบความล้มเหลว 5) Anger-rage เป็นอารมณ์ที่เกิดขึ้นเมื่อบุคคลพบกับอุปสรรคด้านร่างกายหรือด้านจิตใจ ซึ่งมีผลต่อเป้าหมายของบุคคลนั้น 6) Disgust เป็นอารมณ์ที่เกิดจากผลกระทบภายนอก เช่น การได้กลิ่นเหม็น อาหารไม่ถูกปาก ซึ่งอารมณ์ประเภทนี้ทำให้บุคคลหาทางเปลี่ยนแปลงเพื่อให้พ้นสภาวะนั้น ๆ 7) Contempt-scorn เป็นอารมณ์ที่เกิดจากอารมณ์โกรธซึ่งเป็นต้นเหตุของพฤติกรรมความโกรธเคือง 8) Fear-terror เป็นอารมณ์ที่เกิดเมื่อบุคคลกำลังเผชิญกับสิ่งที่ตนไม่สามารถเข้าใจได้ 9) Shame sin shyness-humiliation เป็นอารมณ์ที่เกิดเมื่อบุคคลถูกลงโทษเนื่องจากทำพฤติกรรมไม่เหมาะสม และ 10) Guilt เป็นอารมณ์ที่มีความเกี่ยวข้องกับความผิดหวังหรือความกังวล (Izard, 1991)

(1.1.2) Plutchick ทำการศึกษาวิจัยเรื่องอารมณ์และกำหนดอารมณ์พื้นฐาน 8 ชนิด คือ กลัว ประหลาดใจ เสียใจ รังเกียจ โกรธ คาดหวัง รื่นเริง และยอมรับ ซึ่งอารมณ์แต่ละอารมณ์มีระดับของอารมณ์ที่แตกต่างกัน (Plutchik, 1984) ซึ่งแสดงดังรูปที่ 2.8

(1.1.3) Panksepp เสนอแนวคิดการจำแนกอารมณ์ซึ่งแตกต่างจาก Plutchick คือ จำแนกอารมณ์เป็น 4 ชนิด คือ คาดหวัง เดือดดาล ตื่นตระหนก และหวาดกลัว ซึ่งอารมณ์พื้นฐานแต่ละชนิดเกิดขึ้นสัมพันธ์กับตำแหน่งของสมองในส่วนไฮโปธาลามัสซึ่งแต่ละจุดบนสมองจะตอบสนองต่ออารมณ์ต่างชนิดกัน (Panksepp, 1982)



รูปที่ 2.8 อารมณ์พื้นฐานของ Plutchick (Plutchik, 1984)

จากการจำแนกอารมณ์ได้มีนักจิตวิทยาที่ทำการศึกษาและกำหนดทฤษฎีทางอารมณ์ที่เป็นทฤษฎีอธิบายการเกิดอารมณ์ ซึ่งมีทฤษฎีอธิบายการเกิดอารมณ์ที่ได้รับความนิยม 3 ทฤษฎี คือ 1) ทฤษฎีเจมส์-แลง 2) ทฤษฎีแคนนอน-บาร์ค และ 3) ทฤษฎีแชคเกอร์-ซิงเกอร์ โดยมีรายละเอียดแต่ละทฤษฎี ดังนี้

(1.2) ทฤษฎีการเกิดอารมณ์ของนักจิตวิทยา

(1.2.1) ทฤษฎีเจมส์-แลง (James-Lang Theory) William James นักจิตวิทยาได้อธิบายไว้ว่า ร่างกายจะแสดงปฏิกิริยาโต้ตอบแล้วจึงแสดงออกมาเป็นอารมณ์ ซึ่งแนวคิดของ William James ตรงกับความเห็นของนักจิตวิทยา Carl Lang แต่ในทฤษฎีไม่ได้ระบุว่าการเปลี่ยนแปลงอย่างไรของร่างกายที่จะเกิดควบคู่กับอารมณ์แต่ละชนิด (James, 1982)

(1.2.2) ทฤษฎีแคนนอน-บาร์ค (Cannon-Bard Theory) เป็นทฤษฎีที่คัดค้านทฤษฎีของเจมส์-แลง โดยทฤษฎีนี้ระบุไว้ว่าขณะบุคคลกำลังเผชิญหน้าอยู่กับสิ่งเร้าที่สามารถก่อให้เกิดการกระตุ้นอารมณ์อยู่นั้น แรงกระตุ้นจากประสาทจะส่งไปยังสมอง 2 ส่วน เพื่อประมวลผลแล้วมาแสดงเป็นการเปลี่ยนแปลงทางด้านร่างกาย (Cannon, 1927)

(1.2.3) ทฤษฎีแชคเกอร์-ซิงเกอร์ (Schachter-Singer Theory) ได้อ้างอิงทฤษฎีจากทั้ง 2 ทฤษฎี คือ ทฤษฎีเจมส์-แลง และทฤษฎีแคนนอน-บาร์ค ไว้ว่าถึงแม้จะมีประโยชน์ต่อการศึกษา แต่ทฤษฎีทั้ง 2 ทฤษฎีก็ไม่ได้ทำการอธิบายถึงการตีความสถานการณ์ หรือปฏิกิริยาตอบสนองอัตโนมัติทางกายว่าเกิดขึ้นได้อย่างไร แต่ในทฤษฎีนี้ได้ระบุไว้ว่า อารมณ์เกิดจากการเปลี่ยนปฏิกิริยาตอบสนองอัตโนมัติทางกายและการคิดหาสาเหตุของการตอบสนองนั้นจะใช้กระบวนการคิดการเข้าใจในการตีความ 2 ครั้ง คือ ครั้งที่ 1 เมื่อรับรู้สถานการณ์ให้ตอบสนองทางกายภาพ และครั้งที่ 2 เมื่อระบุอาการตอบสนองแล้วจึงกลายมาเป็นอารมณ์ (Schachter and Singer, 1962)

(2) งานวิจัยที่เกี่ยวกับการวิเคราะห์อารมณ์

งานวิจัยที่เกี่ยวกับการวิเคราะห์อารมณ์ที่ผู้วิจัยได้ทำการศึกษาแบ่งเป็น 2 ส่วน คือ 1) งานวิจัยเกี่ยวกับการวิเคราะห์อารมณ์ และ 2) งานวิจัยเกี่ยวกับอารมณ์ที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ ซึ่งมีรายละเอียดดังนี้

(2.1) งานวิจัยเกี่ยวกับการวิเคราะห์อารมณ์

งานวิจัยของ Koelstra และคณะ (Koelstra, *et al.*, 2012) ทำการนำเสนอชุดข้อมูลในการวิเคราะห์สถานะความรู้สึกและการแสดงออกทางร่างกาย โดยทำการบันทึกข้อมูลของผู้เข้าร่วมการทดลองจำนวน 32 คน จากการให้ผู้เข้าร่วมดูวิดีโอ ในการทดลองเพื่อเก็บข้อมูลการแสดงออกของมนุษย์

(2.2) งานวิจัยเกี่ยวกับอารมณ์ที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์

งานวิจัยของ Graziotinc และคณะ (Graziotin, *et al.*, 2014) ได้ทำการศึกษาเกี่ยวกับจิตวิทยาของผู้ทำงานในด้านการพัฒนาซอฟต์แวร์ โดยในงานวิจัยแสดงให้เห็นว่านักพัฒนาซอฟต์แวร์มีความสุขกว่านักวิเคราะห์ระบบ ซึ่งจากงานวิจัยได้แนะนำไว้ว่านักพัฒนาซอฟต์แวร์เป็นกลุ่มบุคคลที่มีเอกลักษณ์แตกต่างจากงานในวิชาชีพอื่น ๆ และควรมีงานวิจัยที่ทำการศึกษาเกี่ยวกับสถานะความรู้สึกของนักพัฒนาซอฟต์แวร์

งานวิจัยของ Lesiuk (Lesiulk, 2005) ได้คัดเลือกผู้ที่เป็นนักพัฒนาซอฟต์แวร์จำนวน 56 คน เพื่อทำความเข้าใจเกี่ยวกับผลกระทบของการฟังเพลงในขณะที่ทำการออกแบบซอฟต์แวร์ โดยทำการเก็บข้อมูลเป็นระยะเวลา 5 สัปดาห์ โดยให้ผู้เข้าร่วมประเมินตนเองว่าการฟังเพลงมีผลในการออกแบบซอฟต์แวร์หรือไม่ จากผลการวิจัยพบว่าการฟังเพลงขณะทำการออกแบบซอฟต์แวร์ส่ง

ผลกระทบในเชิงบวก ซึ่งจากผลของงานวิจัยแสดงให้เห็นถึงข้อดีของการฟังเพลงเพื่อเป็นการเปลี่ยนแปลงอารมณ์ในเชิงบวกและการรับรู้ที่ดีขึ้นเกี่ยวกับการออกแบบซอฟต์แวร์ขณะทำงาน

งานวิจัยของ Khan (Khan, *et al.*, 2011) ทำการศึกษาผลกระทบทางด้านอารมณ์ที่เกิดกับนักพัฒนาซอฟต์แวร์ขณะที่ทำการแก้ไขข้อผิดพลาดภายในโปรแกรมโดยใช้วิธีการศึกษาด้วยการทดลอง 2 การทดลอง โดยการทดลองครั้งที่ 1 มีนักพัฒนาซอฟต์แวร์เป็นผู้เข้าร่วมการทดลองจำนวน 72 คน ซึ่งในการทดลองแรกได้กำหนดให้นักพัฒนาซอฟต์แวร์ดูคลิปวิดีโอสั้น โดยคลิปที่กำหนดให้ดูนั้นเป็นคลิปที่ช่วยในการกระตุ้นอารมณ์ที่มีความเฉพาะเจาะจง ซึ่งจากผลลัพธ์แสดงให้เห็นว่าคลิปวิดีโอมีผลต่อการแก้ไขข้อผิดพลาดของนักพัฒนาซอฟต์แวร์ ส่วนในการทดลองครั้งที่ 2 มีการขออาสาสมัครเพื่อเป็นผู้เข้าร่วมการทดลองซึ่งมีผู้เข้าร่วมจำนวน โดยการให้ผู้เข้าร่วมทำการออกกำลังกายก่อนทำการตรวจสอบอัลกอริทึมเป็นเวลา 16 นาที ซึ่งพบว่านักพัฒนาซอฟต์แวร์สามารถตรวจสอบอัลกอริทึมได้มีประสิทธิภาพอย่างมีนัยสำคัญ

งานวิจัยของ Wrobel (Wrobel, 2013) นำเสนอผลการสำรวจเกี่ยวกับประสบการณ์ของอารมณ์ในการทำงานของนักพัฒนาซอฟต์แวร์ การวิเคราะห์เกี่ยวกับอารมณ์ที่ส่งผลต่อนักพัฒนาซอฟต์แวร์และผลกระทบต่อประสิทธิภาพ ซึ่งแนวคิดเรื่องความเสี่ยงทางอารมณ์ต่อการพัฒนาซอฟต์แวร์จะช่วยให้สามารถเลือกสภาวะทางอารมณ์ที่ควรหลีกเลี่ยง

2.2 วรรณกรรมที่เกี่ยวข้อง

การศึกษาผลงานวิจัยที่เกี่ยวข้องเพื่อเป็นแนวทางในการดำเนินงานวิจัย ซึ่งจากการศึกษาผู้วิจัยทำการแบ่งกลุ่มประเภทการศึกษาที่เกี่ยวกับ TDD เป็น 4 กลุ่ม แสดงดังรูปที่ 2.9 ประกอบด้วย กลุ่มที่ 1 เป็นการศึกษาการทดลอง กลุ่มที่ 2 การศึกษาจากกรณีศึกษา กลุ่มที่ 3 การศึกษาจากการสำรวจ ซึ่งทั้ง 3 กลุ่มดังกล่าวเป็นกลุ่มการศึกษาเชิงประจักษ์ (Empirical Study) ส่วนการศึกษาในกลุ่มที่ 4 เป็นการศึกษาจากงานวิจัยที่ทำการรวบรวมจากงานวิจัยก่อนหน้า คือ กลุ่มที่ทำการศึกษาในลักษณะการทบทวนวรรณกรรมแบบเป็นระบบ (Systematic Literature Review)

รูปที่ 2.9 แสดงผลงานวิจัยที่เกี่ยวข้องในการศึกษา

2.2.1 การศึกษาการทดลอง

ในกลุ่มนี้ผู้วิจัยจะใช้วิธีการเปรียบเทียบความแตกต่างของตัวแปรที่เปลี่ยนแปลงไประหว่างปรากฏการณ์ที่เกิดขึ้นในสภาพปกติกับที่เกิดขึ้นในสภาพที่ได้รับการควบคุมตามเงื่อนไขต่าง ๆ เพื่อให้ได้ข้อสรุปที่เป็นความจริงต่าง ๆ โดยมีรายละเอียดดังนี้

(1) การทดลอง TDD กับกลุ่มนักศึกษา

การทดลองกับกลุ่มตัวอย่างในระดับมหาวิทยาลัยสามารถพบได้ใน 2 รูปแบบ ดังนี้

(1.1) รูปแบบที่ 1 การนำ TDD มาเปรียบเทียบการทำงานกับ Test-Last โดยใช้กลุ่มนักศึกษาในการทดลอง 2 กลุ่มเพื่อเปรียบเทียบผลที่เกิดขึ้น ในงานวิจัยที่ผ่านมาเป็นการจำลองการศึกษาด้าน TDD ซึ่งได้กำหนดให้กระบวนการ Test-First คือ กระบวนการ TDD ในการวัดผลความสัมพันธ์ระหว่างจำนวนของการทดสอบ คุณภาพของโค้ด และกำลังการผลิตของนักพัฒนา นอกจากนี้ในการทดลองยังให้มีการเปลี่ยนแปลงตัวแปรให้อยู่ในบริบทต่าง ๆ แต่ยังให้ผลลัพธ์ดังเดิม (Fucci and Turhan, 2014) และอีกหนึ่งงานวิจัยเป็นการศึกษาการนำ TDD มาช่วยในการเรียนการสอน เพื่อให้นักศึกษา

สามารถพัฒนาระบบซอฟต์แวร์ได้ดีขึ้น จากการศึกษาพบว่าระบบซอฟต์แวร์มีคุณภาพที่ดีขึ้น แต่ในงานวิจัยนี้ไม่ได้ทำการศึกษาถึงความรู้สึกหรือทัศนคติของนักศึกษาผู้ใช้ TDD ในการทำงานที่ได้รับมอบหมาย (Janzen and Saiedian, 2008) และอีกหนึ่งงานวิจัยโดย Pančur และคณะ (Pančur, *et al.*, 2003) ผู้วิจัยได้ทำการทดลองโดยแบ่งกลุ่มนักพัฒนาออกเป็น 2 กลุ่ม ได้แก่ กลุ่มที่ 1 เป็นกลุ่มที่ใช้การทดสอบแบบดั้งเดิม คือ ทดสอบหลังจากเขียนโปรแกรมโดยตั้งใช้ชื่อกลุ่มนี้ว่า “Iterative test-last (ITL)” และกลุ่มที่ 2 เป็นกลุ่มที่ใช้ TDD โดยตั้งใช้ชื่อกลุ่มนี้ว่า “TDD” จากการเปรียบเทียบเผยให้เห็นถึงความแตกต่างของ Code Coverage จากการใช้ ITL ซึ่งมีความแตกต่างเพียงเล็กน้อย และไม่มีนัยสำคัญ ส่วนในด้านคุณภาพนั้นมีความแตกต่างอย่างมีนัยสำคัญ โดย TDD จะให้คุณภาพของโค้ดที่ดีกว่า

(1.2) รูปแบบที่ 2 เป็นการใช้ TDD ในการเรียนการสอนเพื่อวัดผลจากการใช้ TDD โดยในงานวิจัยของ Edwards เป็นการสอนทักษะการทดสอบซอฟต์แวร์เพื่อนักศึกษาระดับปริญญาตรีสาขาวิชาวิทยาการคอมพิวเตอร์ ในห้องเรียน จากการทดสอบนักศึกษาแสดงให้เห็นถึงประสิทธิภาพและความถูกต้องจากการทดสอบของงาน จากงานวิจัยพบว่านักศึกษาได้คะแนนที่สูงขึ้นและมีข้อบกพร่องในการเขียนโปรแกรมน้อยลง (Edwards, 2003) และจากงานวิจัยโดย Janzen เป็นการแนะนำให้นำ TDD ไปใช้ในการเรียนการสอน โดยภายในงานวิจัยได้บอกว่าในหนังสือหรือในตำราเรียนใหม่ ๆ แนะนำให้นำ TDD ไปใช้ในหลักสูตรการเรียน เพื่อให้นักศึกษามีระเบียบวินัยในการพัฒนาซอฟต์แวร์ด้วย TDD เพราะ TDD จะสามารถช่วยปรับปรุงคุณภาพในการสร้างซอฟต์แวร์ (Software Construction) (Janzen, 2005) และจากงานวิจัยของ Buffardi เป็นการศึกษาการนำ TDD ไปใช้ในกลุ่มของนักศึกษา โดยกล่าวว่า TDD สามารถช่วยให้การผลิตซอฟต์แวร์มีคุณภาพสูงขึ้น ผู้วิจัยจึงต้องการสร้างแรงจูงใจให้แก่นักพัฒนาซอฟต์แวร์มือใหม่หันมาใช้ TDD โดยภายในงานวิจัยมีวัตถุประสงค์เพื่อเป็นการวัดความเชื่อมั่นในการใช้ TDD โดยทำการทดลองภายในรายวิชาที่นักศึกษากำลังศึกษา (Buffardi, 2012)

(2) การทดลอง TDD กับกลุ่มนักพัฒนามืออาชีพ

มีงานวิจัยที่ศึกษาถึงการที่นักพัฒนามืออาชีพได้นำ TDD ไปใช้ในการทำงาน โดยงานวิจัยได้ระบุให้เห็นว่า TDD ได้รับความนิยมในวงการอุตสาหกรรมซอฟต์แวร์ (George and Williams, 2003) จากงานวิจัยที่ผ่านมาในการทดสอบการใช้ TDD ในกลุ่มผู้เชี่ยวชาญสำหรับการพัฒนาซอฟต์แวร์พบว่านักพัฒนาซอฟต์แวร์ที่ใช้วิธีการ TDD พัฒนาซอฟต์แวร์ที่มีคุณภาพสูงขึ้น และสรุปได้ว่าการใช้ TDD เป็นการกระตุ้นให้มีการตรวจสอบที่เข้มงวด แสดงให้เห็นถึงแนวโน้มในกลุ่มผู้พัฒนาในการใช้ TDD เพื่อเพิ่ม

ศักยภาพการทดสอบในอุตสาหกรรมซอฟต์แวร์ (George and Williams, 2004; Canfora, *et al.*, 2004; Munir, *et al.*, 2004)

(3) การทดลอง TDD โดยการเปรียบเทียบกลุ่มนักศึกษากับนักพัฒนามืออาชีพ

การศึกษาการทดลองเปรียบเทียบการใช้ TDD ระหว่างกลุ่มตัวอย่างในระดับมหาวิทยาลัยและกลุ่มนักพัฒนามืออาชีพซึ่งนำ TDD ไปใช้ในการทำงานในสภาวะแวดล้อมการทำงานจริง จากงานวิจัยพบว่ากลุ่มนักพัฒนามืออาชีพให้ความสำคัญและมีความต้องการใช้ TDD มากกว่ากลุ่มนักศึกษา (Müller and Höfer, 2007)

2.2.2 การศึกษาจากกรณีศึกษา

จากงานวิจัยของ Bhat และคณะทำการศึกษาถึงการพัฒนาซอฟต์แวร์โดยใช้วิธีการ TDD ในสภาพแวดล้อมที่แตกต่างกันในแผนกการพัฒนาระบบปฏิบัติการ Windows และแผนกการพัฒนาแอปพลิเคชัน MSN ของบริษัทไมโครซอฟท์ ในทั้งสองกรณีวัดการศึกษาในบริบทต่าง ๆ ของผลิตภัณฑ์ จากการศึกษาพบว่าคุณภาพของโค้ดดีขึ้นสำหรับโครงการที่พัฒนาโดยใช้ TDD เมื่อเทียบกับกลุ่มโครงการที่คล้ายกันในองค์กรเดียวกันที่ไม่ได้ใช้ TDD อีกทั้งยังใช้เวลาน้อยกว่า 15% สำหรับการเขียนการทดสอบ (Bhat and Nagappan, 2006)

2.2.3 การศึกษาจากการสำรวจ

จากงานวิจัยของ Desai และคณะเป็นการสำรวจการใช้ TDD ในหลักสูตรการเรียนการสอนได้มีการแนะนำให้นักศึกษาใช้ TDD ในการแก้ปัญหาเพื่อเป็นการฝึกฝนทักษะการทดสอบซอฟต์แวร์ และให้นักศึกษาตระหนักถึงประโยชน์ของการทดสอบซอฟต์แวร์เพื่อปรับปรุงคุณภาพซอฟต์แวร์ จากการศึกษาแสดงให้เห็นว่าการนำ TDD ไปใช้ทำให้นักศึกษาได้เรียนรู้ทักษะการวิเคราะห์ซอฟต์แวร์ (Desai, *et al.*, 2008) และอีกหนึ่งงานวิจัยของ Mäkinen และคณะ (Mäkinen and Münch, 2014) เป็นการศึกษาเปรียบเทียบผลกระทบและศักยภาพของการนำTDDไปใช้โดยใช้วิธีการศึกษาโดยการสังเกตการณ์ ผลจากการเปรียบเทียบจากคุณภาพของโค้ดแสดงให้เห็นว่า TDD สามารถช่วยลดข้อบกพร่องและแสดงให้เห็นถึงการบำรุงรักษาดีขึ้นรวมไปถึงใช้เวลาในการบำรุงรักษาน้อยลง

2.2.4 การศึกษาทบทวนวรรณกรรม

งานวิจัยของ Causevic และคณะ (Causevic, *et al.*, 2011) ได้ทำการศึกษาวรรณกรรมที่เกี่ยวกับข้อจำกัดในการนำ TDD ไปใช้ในอุตสาหกรรมซอฟต์แวร์ จากหลักฐานเชิงประจักษ์ที่ได้รวบรวมจากผลงานวิจัยที่ได้ตีพิมพ์ซึ่งเกี่ยวกับ TDD ทั้งหมด 48 ฉบับ คณะผู้วิจัยได้ระบุถึงข้อจำกัดในการนำ TDD ไปใช้ในอุตสาหกรรมซอฟต์แวร์ดังนี้

(1) เวลาในการพัฒนาระบบซอฟต์แวร์เพิ่มมากขึ้น

(2) นักพัฒนาซอฟต์แวร์ขาดประสบการณ์และความรู้ในการนำ TDD ไปปฏิบัติงานจริง

(3) การออกแบบซอฟต์แวร์ไม่ดีเพียงพอเมื่อมีการใช้ TDD

(4) นักพัฒนาซอฟต์แวร์ขาดทักษะการทดสอบซอฟต์แวร์

(5) ในขณะที่นำ TDD ไปปฏิบัติ นักพัฒนาซอฟต์แวร์ไม่สามารถปฏิบัติได้ตามขั้นตอนของ TDD

(6) มีข้อจำกัดในการใช้เครื่องมือสำหรับช่วยการทดสอบซอฟต์แวร์

(7) มีโค้ดเก่าที่ยังไม่เคยทำการทดสอบ (Legacy Code) ซึ่งยังไม่มีโค้ดที่ใช้ทดสอบหน่วยย่อย

บทที่ 3

วิธีดำเนินการวิจัย

ภายในงานวิจัยใช้วิธีการวิจัยด้านวิศวกรรมซอฟต์แวร์เชิงประจักษ์เพื่อตอบคำถามของงานวิจัย คือ *การใช้ Test-Driven Development สามารถช่วยกระตุ้นความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษาได้หรือไม่* โดยผู้วิจัยได้ตั้งสมมติฐานงานวิจัยไว้ดังนี้

สมมติฐานหลัก H_0 : นักศึกษาที่ได้รับการเรียนการสอนโดยใช้ TDD มีความสนใจในการพัฒนาซอฟต์แวร์ไม่ต่างกันกับการใช้แบบจำลองน้ำตก

สมมติฐานรอง H_1 : นักศึกษาที่ได้รับการเรียนการสอนโดยใช้ TDD มีความสนใจในการพัฒนาซอฟต์แวร์ต่างกับการใช้แบบจำลองน้ำตก

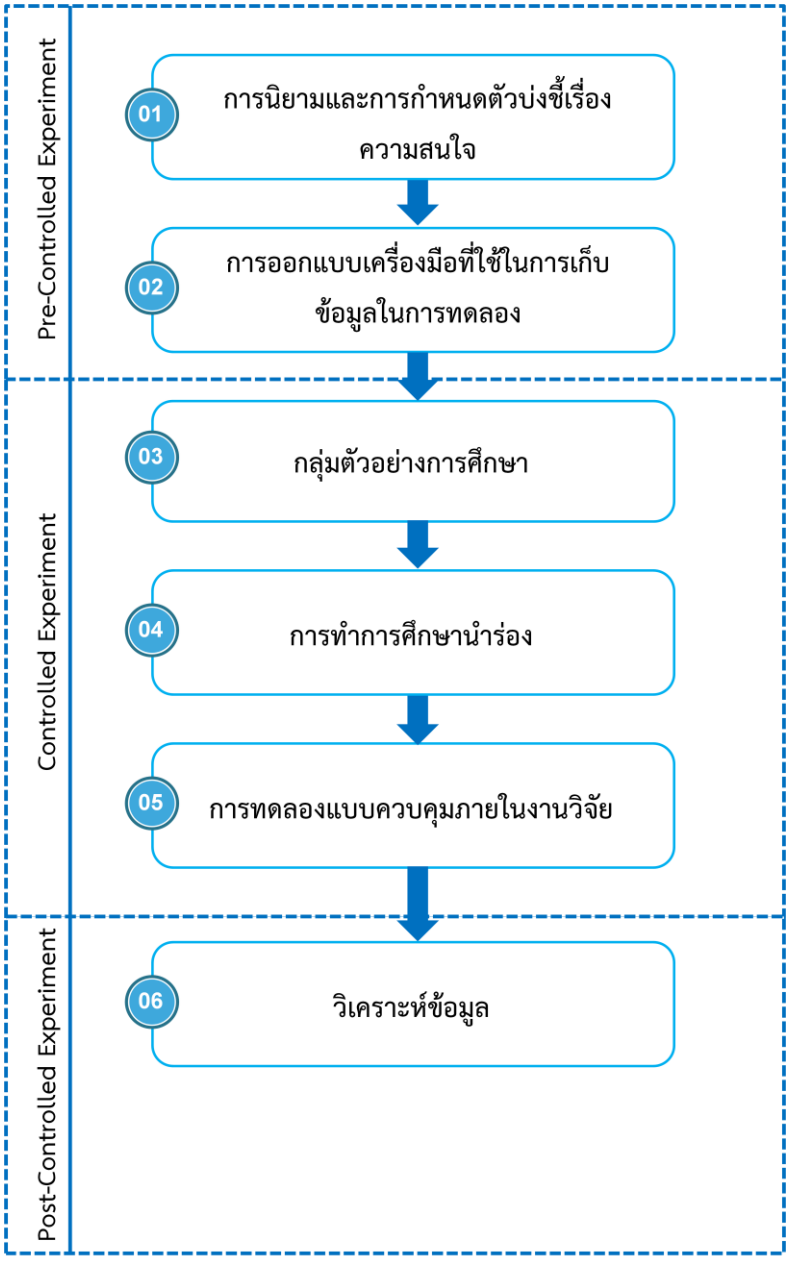
เพื่อตอบคำถามของงานวิจัยผู้วิจัยจึงใช้วิธีการทดลองแบบควบคุม โดยมีวิธีการดำเนินงานวิจัย แสดงดังรูปที่ 3.1 วิธีการดำเนินงานแบ่งได้เป็น 3 ส่วน ดังนี้

ส่วนที่ 1 เป็นการเตรียมตัวก่อนการทดลองซึ่งเป็นการศึกษานิยามและการกำหนดตัวบ่งชี้เรื่องความสนใจ เนื่องจากความสนใจมีนิยามที่หลากหลายและยังไม่มีการกำหนดตัวบ่งชี้ทางด้านความสนใจ หลังจากนั้นจึงทำการออกแบบเครื่องมือที่ใช้ในการเก็บข้อมูลซึ่งแบ่งออกเป็น 3 ส่วน คือ 1) การออกแบบแบบสอบถาม (Questionnaire Design) 2) การออกแบบคำถามในการสัมภาษณ์ (Interview Design) และ 3) การออกแบบแบบฟอร์มในการสังเกตการณ์ (Observaion Form Design) เพื่อใช้ในการเก็บข้อมูลของนักศึกษาภายในการทดลองแบบควบคุม

ส่วนที่ 2 คือ การทดลองแบบควบคุม โดยในการทดลองแบบควบคุมมีการให้ผู้เข้าร่วมการทดลองพัฒนาซอฟต์แวร์ 2 ในรูปแบบ คือ การพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและการพัฒนา

ซอฟต์แวร์ด้วยวิธีการ TDD ซึ่งการทดลองแบบควบคุมภายในงานวิจัยมีระยะเวลาในการทดลองและเก็บข้อมูลเป็นเวลา 5 เดือน (1 ภาคการศึกษา)

ส่วนที่ 3 คือ การวิเคราะห์ข้อมูลหลังการทดลอง



รูปที่ 3.1 แสดงวิธีการดำเนินงานภายในงานวิจัย

3.1 การออกแบบก่อนการทดลอง (Pre-Controlled Experiment)

การเตรียมตัวก่อนการทดลองเป็นการออกแบบแบบสอบถาม คำถามในการสัมภาษณ์ และแบบฟอร์มในการสังเกตการณ์เพื่อเก็บข้อมูลภายในการทดลอง โดยก่อนที่จะทำการออกแบบแบบสอบถาม คำถามภายในการสัมภาษณ์ และแบบฟอร์มในการสังเกตการณ์ ผู้วิจัยได้ทำการศึกษานิยามด้านความสนใจเพื่อนำนิยามมากำหนดเป็นตัวบ่งชี้ในการกำหนดข้อคำถามในแบบสอบถามและคำถามในการสัมภาษณ์สำหรับการออกแบบก่อนการทดลองมีรายละเอียดดังนี้

3.1.1 ตัวแปรภายในงานวิจัย

เนื่องจากในงานวิจัยเป็นการวิจัยเชิงทดลอง ผู้วิจัยจึงกำหนดตัวแปรอิสระ (Independent Variable) ในงานวิจัย คือ วิธีการพัฒนาซอฟต์แวร์ และได้กำหนดให้มีตัวแปรทดสอบ (Test Factor Variable) เพื่อดูผลของความสัมพันธ์ระหว่างตัวแปรอิสระกับตัวแปรตามโดยกำหนดตัวแปรทดสอบ 2 ตัวแปร คือ 1) การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และ 2) การพัฒนาซอฟต์แวร์แบบ TDD สำหรับตัวแปรตาม (Dependent Variable) ในงานวิจัย คือ ความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ โดยในการทดลองผู้วิจัยได้ควบคุมสภาพแวดล้อมของงานวิจัยที่มีการทดลองภายในห้องปฏิบัติการทั้ง 2 ครั้ง ให้มีสภาพแวดล้อมที่เหมือนกันทั้ง 2 ครั้ง รวมถึงก่อนเริ่มทำการทดลองผู้วิจัยได้ทำการแจ้งข้อตกลงในการเป็นผู้ร่วมการทดลอง ให้ผู้เข้าร่วมทดลองทราบว่าผู้วิจัยจะมีการเก็บข้อมูลจากการแสดงพฤติกรรมการตอบแบบสอบถาม และการตอบคำถามในการสัมภาษณ์ ซึ่งการร่วมการทดลองนี้จะไม่มีผลต่อคะแนนภายในรายวิชาที่ผู้วิจัยขอเข้าไปทำการทดลอง

3.1.2 การนิยามและการกำหนดตัวบ่งชี้เรื่องความสนใจ

จากการศึกษานิยามด้านความสนใจได้มีการกำหนดนิยามที่ค่อนข้างแตกต่างกันซึ่งแสดงดังตารางที่ 2.1 ผู้วิจัยจึงได้ทำการศึกษาและนำมาสรุปเป็นนิยามเพื่อนำมาใช้ในงานวิจัย ซึ่งผู้วิจัยได้สรุปนิยามด้านความสนใจได้ว่า *“ความสนใจเป็นความรู้สึกถึงความพึงพอใจหรือทัศนคติที่มีต่อสิ่งหนึ่ง แนวคิดใดแนวคิดหนึ่ง หรือสถานการณ์ใดสถานการณ์หนึ่ง อันมีแนวโน้มให้อยากเข้าใกล้อยากรู้อยากเห็น*

สิ่งนั้น ซึ่งทำให้เกิดความเพียรพยายามและสามารถกระทำจนบรรลุจุดมุ่งหมาย ซึ่งความสนใจเป็นแรงผลักดันที่กระตุ้นให้บุคคลกระทำการใด ๆ หรือแสดงถึงแนวโน้มที่ตัวบุคคลจะทำการเลือก ซึ่งในด้านของการศึกษาความสนใจมีความสัมพันธ์กับความสำเร็จในด้านการเรียนวิชาการ และเป็นองค์ประกอบหนึ่งในการพัฒนาความสามารถในการประกอบอาชีพ” (เสถียร ศรีรัตน์, 2522; รัตนา คงคาเนาวรัตน์, 2542; สุชา จันทร์เอม, 2542; จิราพร สุจริต, 2543; จิตมณี อะเมกอง, 2545; Charters and Good, 1945; Dwyer, 1993) หลังจากได้นิยามด้านความสนใจแล้วจึงนำนิยามที่สรุป มากำหนดตัวบ่งชี้ทางด้านความสนใจเพื่อเป็นตัวกำหนดคำถามภายในแบบสอบถามที่ใช้ในการเก็บข้อมูลภายในการทดลอง โดยผู้วิจัยได้กำหนดตัวบ่งชี้ด้านความสนใจไว้ทั้งหมด 5 ตัวบ่งชี้ ดังนี้

(1) ความกระตือรือร้นในสิ่งที่สนใจ สำหรับตัวบ่งชี้นี้มีวัตถุประสงค์เพื่อเป็นการศึกษาข้อมูลสิ่งที่นักศึกษาทำการเรียนรู้ก่อนการทดลอง หลังการทดลอง และขณะทำการทดลอง โดยความกระตือรือร้นในสิ่งที่สนใจจะเป็นแรงผลักดันที่กระตุ้นให้นักศึกษาแสดงแนวโน้มของการเลือกสิ่งที่นักศึกษาทำการศึกษา เช่น หากนักศึกษามีความสนใจเรื่องการเขียนโปรแกรมอาจจะเป็นแรงผลักดันให้นักศึกษาสนใจที่จะประกอบอาชีพเกี่ยวกับการพัฒนาซอฟต์แวร์

(2) การศึกษาเพิ่มเติมด้วยตนเอง สำหรับตัวบ่งชี้นี้มีวัตถุประสงค์เพื่อเป็นการศึกษาข้อมูลสิ่งที่นักศึกษาศึกษาเพิ่มเติมด้วยตนเองโดยอาจจะเกี่ยวข้อง หรือไม่เกี่ยวข้องกับด้านการพัฒนาซอฟต์แวร์ หรืออาจเป็นสิ่งที่เกี่ยวข้องกับด้านการพัฒนาซอฟต์แวร์แต่ไม่มีอยู่ภายในหลักสูตรการเรียนการสอนภายในมหาวิทยาลัย เช่น นักศึกษาศึกษาเพิ่มเติมถึงเรื่องการเขียนโปรแกรมในภาษา Python เป็นการศึกษาเพิ่มเติมด้านการพัฒนาซอฟต์แวร์แต่ไม่มีในหลักสูตรการเรียนการสอนในมหาวิทยาลัย

(3) การทำแบบฝึกหัด สำหรับตัวบ่งชี้นี้มีวัตถุประสงค์เพื่อเป็นการศึกษาข้อมูลการทำแบบฝึกหัดของนักศึกษา โดยวิชาที่นักศึกษาทำแบบฝึกหัดอาจจะเป็นวิชาที่นักศึกษาให้ความสนใจ เช่น นักศึกษาบางส่วนจะทำแบบฝึกหัดด้วยตนเองเฉพาะวิชาที่เกี่ยวข้องกับการวิเคราะห์ระบบแต่จะไม่ทำแบบฝึกหัดเกี่ยวกับการเขียนโปรแกรม

(4) เวลาในการทำแบบฝึกหัด สำหรับตัวบ่งชี้นี้มีวัตถุประสงค์เพื่อเป็นการศึกษาเวลาในการทำแบบฝึกหัด เนื่องจากเวลาในการทำแบบฝึกหัดของนักศึกษาแต่ละคนใช้เวลาในการทำแบบฝึกหัดแตกต่างกัน เช่น นักศึกษาที่ใช้เวลาในการทำแบบฝึกหัดนานอาจจะเป็นนักศึกษาที่ไม่

เข้าใจในเรื่องรายวิชาที่เรียน หรือนักศึกษาที่ใช้เวลาในการทำแบบฝึกหัดนานอาจจะเป็นนักศึกษาที่ทำการศึกษาเนื้อหาที่ทำแบบฝึกหัดเพิ่มเติม

(5)เวลาในการศึกษาด้วยตนเอง สำหรับตัวบ่งชี้นี้มีวัตถุประสงค์เพื่อเป็นการศึกษาเวลาในการศึกษาด้วยตัวเอง เช่น ถ้านักศึกษาสนใจศึกษาเพิ่มเติมเรื่องการพัฒนาซอฟต์แวร์ด้วยภาษาจาวา นักศึกษาอาจจะใช้เวลาในการศึกษาเพิ่มเติมด้วยตัวเองมากกว่าการศึกษาเรื่องอื่น ๆ

โดยเมื่อได้ตัวบ่งชี้ด้านความสนใจแล้วผู้วิจัยจึงนำตัวบ่งชี้ที่ได้ไปทำการออกแบบคำถามในแบบสอบถาม ซึ่งประกอบด้วยคำถามปลายเปิด และคำถามแบบ Likert Scale

3.1.3 การออกแบบเครื่องมือที่ใช้ในการเก็บข้อมูลในการทดลอง

การออกแบบเครื่องมือที่ใช้ในการเก็บข้อมูลภายในการทดลองมีทั้งหมด 3 ส่วน คือ 1) การออกแบบแบบสอบถาม 2) การออกแบบคำถามในการสัมภาษณ์ และ 3) การออกแบบแบบฟอร์มในการสังเกตการณ์ โดยการออกแบบเครื่องมือที่ใช้ในการเก็บข้อมูลทั้ง 3 ส่วนนี้ มีจุดมุ่งหมายเพื่อใช้เก็บข้อมูลในด้านความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษาในการทดลอง โดยแบบสอบถามและคำถามในการสัมภาษณ์เป็นการเก็บข้อมูลเชิงปริมาณและเชิงคุณภาพ เพื่อนำข้อมูลที่ได้จากนักศึกษาก่อนการทดลองและหลังการทดลองไปวิเคราะห์ผลว่าการนำ TDD มาใช้ในการพัฒนาซอฟต์แวร์สามารถกระตุ้นความสนใจของนักศึกษาได้หรือไม่ และเป็นการศึกษาผลกระทบในการนำ TDD มาใช้ในการศึกษา

สำหรับแบบสอบถามนั้นจะประกอบไปด้วยการเก็บข้อมูลเชิงปริมาณที่อยู่ในรูปแบบ Likert Scale และมีการเก็บข้อมูลเชิงคุณภาพซึ่งอยู่ในรูปแบบคำถามปลายเปิด ส่วนคำถามในการสัมภาษณ์อ้างอิงมาจากคำถามในแบบสอบถามปลายเปิด ซึ่งผู้วิจัยต้องการข้อมูลในเชิงลึกที่นักศึกษาอาจจะไม่ได้ตอบในแบบสอบถามเนื่องจากการสัมภาษณ์ทำให้ผู้ถูกสัมภาษณ์มีอิสระในการตอบคำถามมากกว่าการตอบคำถามในแบบสอบถาม และในการตอบแบบอธิบายด้วยคำพูดสามารถตอบได้ง่ายกว่าการเขียนอธิบาย รวมถึงผู้สัมภาษณ์และผู้ตอบคำถามสามารถใช้คำศัพท์เฉพาะด้านในการตอบคำถามได้ รวมถึงทำให้ได้ข้อมูลที่มีความครอบคลุมมากกว่า ส่วนแบบฟอร์มในการสังเกตการณ์เป็นการเก็บข้อมูลระหว่างการทำการทดลองเพื่อสังเกตพฤติกรรมที่เกิดขึ้นกับนักศึกษาขณะทำการทดลอง โดยนำผลจากการสังเกตการณ์มาเปรียบเทียบพฤติกรรมว่านักศึกษามีพฤติกรรมที่แตกต่างกันหรือไม่ ในการทดลองที่ใช้การ

พัฒนาซอฟต์แวร์แบบจำลองน้ำตกและ TDD สำหรับรายละเอียดการออกแบบแบบสอบถาม คำถามในการสัมภาษณ์ และแบบฟอร์มในการสังเกตการณ์มี ดังนี้

(1) การออกแบบแบบสอบถาม

ผู้วิจัยได้พัฒนาแบบสอบถามขึ้นมาเพื่อให้กลุ่มตัวอย่างได้ตอบคำถามก่อนการศึกษา (Pre-test) และหลังทำการศึกษา (Post-test) รวมถึงการทำแบบฝึกหัดเรื่อง TDD โดยคำถามในแบบสอบถามได้รับการตรวจสอบจากผู้เชี่ยวชาญทางด้านวิศวกรรมซอฟต์แวร์ ซึ่งมีประสบการณ์ในการทำงานทางด้านการพัฒนาซอฟต์แวร์ และงานวิจัยทางด้าน TDD จำนวน 1 คน สำหรับคำถามที่ได้พัฒนาขึ้นมานั้นมีด้วยกัน 2 รูปแบบ คือ

(1.1) แบบสอบถามปลายเปิด การสร้างแบบสอบถามปลายเปิดเป็นการเก็บข้อมูลเบื้องต้นของนักศึกษาและข้อมูลด้านการพัฒนาซอฟต์แวร์ในรูปแบบโครงการในรายวิชาการโปรแกรมเชิงอ็อบเจกต์และการทำโครงการพัฒนาซอฟต์แวร์ด้วยภาษาจาวา ซึ่งใช้กระบวนการในการสอนในรูปแบบการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก รวมไปถึงเป็นการเก็บข้อมูลกระบวนการที่นักศึกษาใช้พัฒนาซอฟต์แวร์และความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษาด้วย

(1.2) แบบสอบถาม Likert Scale การสร้างแบบสอบถามแบบ Likert Scale เป็นการเก็บข้อมูลด้านความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษา โดยมีการกำหนดระดับความสนใจ 5 ระดับ คือ 1) ความสนใจน้อยที่สุด 2) ความสนใจน้อย 3) ความสนใจปานกลาง 4) ความสนใจมาก และ 5) ความสนใจมากที่สุด

การสร้างแบบสอบถามนี้ได้ถูกพัฒนามาจากตัวบ่งชี้ด้านความสนใจที่ผู้วิจัยได้กำหนดขึ้น โดยในตารางที่ 3.1 แสดงคำถามที่ได้พัฒนาขึ้นมาทั้งที่เป็นคำถามปลายเปิดและปลายปิด รวมถึงตัวบ่งชี้ของแต่ละคำถาม ที่แสดงดังตารางที่ 3.1 ผู้วิจัยได้แบ่งการแสดงคำถามตามตัวบ่งชี้ทั้ง 5 ตัวบ่งชี้ โดยแบบสอบถามที่ใช้ในงานวิจัยแสดงอยู่ในภาคผนวก ก

ตารางที่ 3.1 คำถามในแบบสอบถามและตัวบ่งชี้

คำถามในแบบสอบถาม	ตัวบ่งชี้
ข้อมูลนักศึกษา (คำถามปลายเปิด)	
1. เกรดในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ (OOP)	ความกระตือรือร้นในสิ่งที่สนใจ
2. โครงการเทอมในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ (OOP) เกี่ยวกับอะไร	
คำถามพื้นฐานก่อนการทดลอง (คำถามปลายเปิด)	
1. เหตุผลที่นักศึกษาเข้าศึกษาในสาขาวิชาวิศวกรรมซอฟต์แวร์	ความกระตือรือร้นในสิ่งที่สนใจ
2. ความเข้าใจของนักศึกษาของคำว่า “นักพัฒนาซอฟต์แวร์”	
3. นักศึกษามีความสนใจในด้านการพัฒนาซอฟต์แวร์หรือไม่ เพราะเหตุใด (“ความสนใจ เป็นความรู้สึกถึงความพึงพอใจเป็นความอยากรู้อยากเห็น ความแสวงหาความรู้สึกชอบความพอใจของบุคคลต่อกิจกรรมนั้น ซึ่งทำให้เพียรพยายามและสามารถกระทำสิ่งการจนบรรลุจุดมุ่งหมาย”)	
4. กระบวนการที่นักศึกษาใช้ในการทำโครงการในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ (OOP) ตั้งแต่เริ่มต้นการทำโครงการจนถึงการนำเสนอโครงการ	
5. ภาษาโปรแกรมมิ่งที่นักศึกษามีความสนใจ	
6. การทำโครงการพัฒนาซอฟต์แวร์ในรายวิชาที่ผ่านมานักศึกษามีการทดสอบระบบที่พัฒนาขึ้นอย่างไร	
7. จากการศึกษาที่ผ่านมานักศึกษาให้ความสำคัญกับวิชาใดบ้างในการเรียนในสาขาวิชาวิศวกรรมซอฟต์แวร์ (“ความสำคัญ เป็นวิชาที่นักศึกษาคิดว่ามีผลต่อการศึกษาต่อในรายวิชา หรือจำเป็นต้องใช้งานในงานด้านการพัฒนาซอฟต์แวร์”)	

ตารางที่ 3.1 คำถามในแบบสอบถามและตัวบ่งชี้ (ต่อ)

คำถามในแบบสอบถาม	ตัวบ่งชี้
8. จากการศึกษาที่ผ่านมานักศึกษามีความสนใจในวิชาใดบ้างในการเรียนในสาขาวิชาวิศวกรรมซอฟต์แวร์ (“ความสนใจ เป็นความรู้สึกพึงพอใจในการเรียนวิชาดังกล่าว มีความรู้สึกอยากเรียนรู้เพิ่มเติมจากงาน หรือมีการศึกษาเพิ่มเติมด้วยตนเอง”)	
9. หากมีการจัดอบรมทั้งในเชิงบรรยายหรือเชิงปฏิบัติการนักศึกษาอยากให้มีการจัดอบรมหัวข้อใด เพราะเหตุใด	
10. อาชีพที่นักศึกษาคาดว่าหลังจากจบการศึกษาจะเข้าทำงาน	ศึกษาเพิ่มเติมด้วยตนเอง
11. นักศึกษาใช้เวลาเฉลี่ยเท่าใดในการศึกษาเพิ่มเติมด้วยตนเอง (เวลาเฉลี่ยต่อสัปดาห์)	
12. นักศึกษามักจะทำการศึกษาเรื่องใดเมื่อศึกษาเพิ่มเติมด้วยตนเอง	
13. หากมีการจัดอบรมทั้งในเชิงบรรยายหรือเชิงปฏิบัติการนักศึกษาอยากให้มีการจัดอบรมหัวข้อใด เพราะเหตุใด	
14. นักศึกษาใช้เวลาเฉลี่ยเท่าใด ในการพัฒนาซอฟต์แวร์ตามแบบฝึกหัดที่ได้รับมอบหมายต่อครั้ง (เวลาเฉลี่ยต่อสัปดาห์)	การทำแบบฝึกหัด
15. นักศึกษาใช้เวลาเฉลี่ยเท่าใด ในการพัฒนาซอฟต์แวร์ตามแบบฝึกหัดที่ได้รับมอบหมายต่อครั้ง (เวลาเฉลี่ยต่อสัปดาห์)	เวลาในการทำแบบฝึกหัด
ความสนใจของนักศึกษาเกี่ยวกับการเรียนด้านการพัฒนาซอฟต์แวร์ (Likert Scale)	
1. นักศึกษาให้ความสนใจในการเรียนในรายวิชาด้านการพัฒนาซอฟต์แวร์	ความกระตือรือร้นในสิ่งที่สนใจ
2. นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากแบบฝึกหัด (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)	

ตารางที่ 3.1 คำถามในแบบสอบถามและตัวบ่งชี้ (ต่อ)

คำถามในแบบสอบถาม	ตัวบ่งชี้
3. นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากการศึกษาหาข้อมูลด้วยตนเอง (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)	ความกระตือรือร้นในสิ่งที่สนใจ
4. นักศึกษาให้ความสนใจในการศึกษาเกี่ยวกับวิศวกรรมซอฟต์แวร์เพิ่มเติมด้วยตนเอง (การศึกษาเพิ่มเติมด้านวิศวกรรมซอฟต์แวร์ เช่น การศึกษาเพิ่มเติมถึงกระบวนการพัฒนาซอฟต์แวร์รูปแบบใหม่ ภาษาในการเขียนโปรแกรม เป็นต้น)	
5. นักศึกษามีความสนใจในการดัดแปลงการแสดงผลจากแบบฝึกหัดที่ได้รับมอบหมาย	
6. นักศึกษาให้ความสนใจในการมีส่วนร่วมในการถาม-ตอบภายในห้องเรียน	
7. นักศึกษาให้ความสนใจในการใช้เวลาในการทำแบบฝึกหัด	การทำแบบฝึกหัด
8. นักศึกษาให้ความสนใจในการใช้เวลาในการทำแบบฝึกหัด	เวลาในการทำแบบฝึกหัด
9. นักศึกษาให้ความสนใจในการใช้เวลาในการศึกษาด้วยตนเอง	ศึกษาเพิ่มเติมด้วยตนเอง
10. นักศึกษาให้ความสนใจในการใช้เวลาในการศึกษาด้วยตนเอง	เวลาในการศึกษาเพิ่มเติมด้วยตนเอง
คำถามหลังการทดลอง (คำถามปลายเปิด)	
1. การใช้ Test-Driven Development ในการเรียนการสอนมีผลต่อนักศึกษาในการให้ความสนใจในการพัฒนาซอฟต์แวร์ หรือไม่เพราะเหตุใด	ความกระตือรือร้นในสิ่งที่สนใจ
2. การใช้ Test-Driven Development ส่งผลต่อนักศึกษาอย่างไรในการเขียนโปรแกรมในห้องปฏิบัติการ	

ตารางที่ 3.1 คำถามในแบบสอบถามและตัวบ่งชี้ (ต่อ)

คำถามในแบบสอบถาม	ตัวบ่งชี้
คำถามหลังการทดลอง (คำถามปลายเปิด)	
3. การใช้ Test-Driven Development ในการเรียนการสอนมีผลต่อนักศึกษาในการให้ความสนใจในการพัฒนาซอฟต์แวร์ หรือไม่เพราะเหตุใด	ความกระตือรือร้นในสิ่งที่สนใจ
4. การใช้ Test-Driven Development ส่งผลต่อนักศึกษาอย่างไรในการเขียนโปรแกรมในห้องปฏิบัติการ	
5. จากการทำ Test-Driven Development ในห้องปฏิบัติการหากให้โจทย์เดียวกันแต่ให้นักศึกษาใช้วิธีการแบบจำลองน้ำตก (Waterfall Model) ในความคิดเห็นของนักศึกษาคิดว่าวิธีการใดทำให้นักศึกษาพบข้อผิดพลาดในโปรแกรมน้อยลง เพราะเหตุใด	
6. ในอนาคตหากมีการพัฒนาโครงการในรายวิชา นักศึกษาจะเลือกใช้วิธีการทดสอบแบบใดในการทดสอบโปรแกรมที่นักศึกษาทำการพัฒนา	
7. ในการทำการทดสอบโปรแกรม นักศึกษามีความคิดเห็นว่าวิธีการทดสอบแบบใดที่สามารถทำให้การทดสอบมีความละเอียดมากขึ้น เพราะเหตุใด	
8. ในมุมมองของนักศึกษาคิดว่า Test-Driven Development มีประโยชน์ต่อการพัฒนาซอฟต์แวร์หรือไม่ เพราะเหตุใด	

(2) การออกแบบคำถามในการสัมภาษณ์

การออกแบบคำถามในการสัมภาษณ์เป็นการออกแบบเพื่อใช้ในการสัมภาษณ์แบบกึ่งโครงสร้าง (Semi-Structured Interview) โดยการสัมภาษณ์แบบกึ่งโครงสร้างนั้น ผู้สัมภาษณ์จะสามารถเพิ่มเติมหรือปรับเปลี่ยนคำถามได้ระหว่างการสัมภาษณ์ โดยคำถามที่ใช้ในการสัมภาษณ์ใช้คำถามเดียวกับแบบสอบถามปลายเปิด แต่จะมีวัตถุประสงค์เพื่อเก็บข้อมูลเพิ่มเติมจากการตอบแบบสอบถามของนักศึกษาทั้งก่อนการทดลองและหลังการทดลอง เนื่องจากการสัมภาษณ์ทำให้ได้ข้อมูลเชิงลึกกว่าแบบสอบถาม

(3) การออกแบบแบบฟอร์มในการสังเกตการณ์

การออกแบบแบบฟอร์มในการสังเกตการณ์จัดทำขึ้นเพื่อทำการเก็บข้อมูลจากนักศึกษาขณะที่นักศึกษากำลังดำเนินการการทดลอง โดยในแบบฟอร์มในการสังเกตการณ์มีการเก็บข้อมูลที่จะนำไปวิเคราะห์ 3 ส่วนหลัก คือ 1) การสังเกตการณ์ในการแสดงออกทางกายภาพ 2) การสังเกตการณ์ในการเขียนโปรแกรม และ 3) การสังเกตการณ์ในการสืบค้นข้อมูล ซึ่งการเก็บข้อมูลจากการสังเกตการณ์มีวัตถุประสงค์เพื่อระบุพฤติกรรมของนักศึกษาขณะที่ทำการทดลองว่ามีพฤติกรรมที่แตกต่างกันหรือไม่ระหว่างการเขียนโปรแกรมในรูปแบบการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและการพัฒนาซอฟต์แวร์รูปแบบ TDD โดยแบบฟอร์มในการสังเกตการณ์ แสดงดังในภาคผนวก ข

3.2 การดำเนินการทดลองแบบควบคุม (Controlled Experiment)

3.2.1 กลุ่มตัวอย่างการศึกษา

กลุ่มตัวอย่างการศึกษาในงานวิจัยนี้เป็นนักศึกษาสาขาวิชาวิศวกรรมซอฟต์แวร์ ชั้นปีที่ 2 จำนวน 52 คน ในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ขั้นสูง (Advanced Object Oriented Programming) จากนักศึกษาของสาขาวิชาวิศวกรรมซอฟต์แวร์จำนวน 4 ชั้นปี เนื่องจากงานวิจัยนี้เกี่ยวข้องกับการทดสอบและการทำรีแฟคทอริง ซึ่งเป็นส่วนหนึ่งของการเรียนด้านวิศวกรรมซอฟต์แวร์ และกลุ่มตัวอย่างทุกคนได้รับการเรียนพื้นฐานการเขียนโปรแกรมจากรายวิชาการโปรแกรมเชิงอ็อบเจกต์และการทำโครงการ

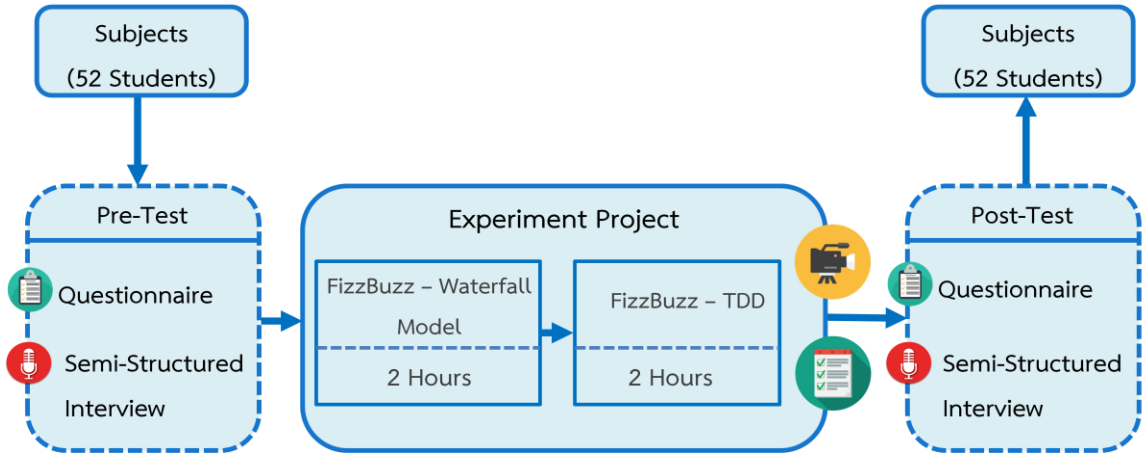
พัฒนาซอฟต์แวร์ด้วยภาษาจาวามาก่อน ซึ่งรายวิชาดังกล่าวมีการเรียนการสอนแบบจำลองน้ำตก ทำให้นักศึกษาสามารถตอบคำถามจากการสัมภาษณ์และคำถามจากแบบสอบถามได้ถึงความสนใจในการพัฒนาซอฟต์แวร์ ซึ่งผู้วิจัยสามารถนำคำตอบจากการสัมภาษณ์และการตอบแบบสอบถามเพื่อไปวิเคราะห์ความสนใจในด้านของการพัฒนาซอฟต์แวร์ของนักศึกษาได้

3.2.2 การทำการศึกษานำร่อง (Pilot Study)

การทำการศึกษานำร่องเป็นการดำเนินการการทดลองที่ได้ออกแบบไว้กับกลุ่มตัวอย่างขนาดเล็ก (Van Tejlingen and Hundley, 2002) สำหรับการศึกษานำร่องในงานวิจัยนี้มีวัตถุประสงค์เพื่อตรวจสอบความเที่ยงแบบสอดคล้องภายใน (Internal Consistency Reliability) ของชุดคำถาม และลดความผิดพลาดของการตอบคำถามจากคำถามที่คลุมเครือ หรือการใช้คำที่ไม่เหมาะสม (Kaufmann and Janzen, 2003) โดยกลุ่มตัวอย่างที่ใช้ในการศึกษานำร่องเป็นกลุ่มผู้เข้าร่วมทดลองที่เป็นตัวแทนของกลุ่มการศึกษาจริง ซึ่งผู้วิจัยได้ทำการศึกษานำร่องในการทดลองโดยให้กลุ่มตัวอย่างทำแบบสอบถามในการทดลองและทำการสัมภาษณ์กลุ่มตัวอย่างจำนวน 32 คน โดยกลุ่มตัวอย่างใช้เวลาในการทำแบบสอบถามโดยเฉลี่ยเป็นเวลา 30 นาทีต่อคน และทำการสัมภาษณ์กลุ่มตัวอย่างด้วยการสัมภาษณ์แบบหนึ่งต่อหนึ่งโดยใช้เวลาเฉลี่ยในการสัมภาษณ์เป็นเวลา 30 นาทีต่อคน และใช้เวลาในการสอบถามเพื่อขอคำแนะนำในการปรับปรุงแบบสอบถามเป็นเวลาโดยเฉลี่ยคนละ 15 นาที จากการที่ให้กลุ่มตัวอย่างทำแบบสอบถามเพื่อตรวจสอบว่าแบบสอบถามมีความเหมาะสมสำหรับใช้ในการทดลองหรือไม่ ผู้วิจัยได้ทำการคำนวณความเที่ยงแบบสอดคล้องภายใน โดยใช้ค่าสัมประสิทธิ์แอลฟาของครอนบัค จากการคำนวณได้ค่าสัมประสิทธิ์เท่ากับ 0.89 ซึ่งแสดงให้เห็นขนาดของความสอดคล้องกันของข้อคำถามที่มุ่งวัดในสิ่งเดียวกันได้ค่าที่ค่อนข้างเที่ยงตรง (มากกว่า 0.7) จากการศึกษานำร่องพบปัญหาจากแบบสอบถามและการสัมภาษณ์ คือ ในการถามคำถามเกี่ยวกับการทดสอบระบบในโครงการในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ กลุ่มตัวอย่างมีความเข้าใจว่าการทดสอบในโครงการในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ คือการใช้คำสั่ง Exception เพื่อป้องกันการเกิดความผิดพลาดของโปรแกรมที่ทำขึ้น

3.2.3 การดำเนินการทดลอง (Controlled Experiment)

การดำเนินการทดลองควบคุมสำหรับงานวิจัยนี้มีขั้นตอนในการทดลอง แสดงดังรูปที่ 3.2 โดยมีรายละเอียดในแต่ละขั้นตอนดังนี้



รูปที่ 3.2 แสดงการทดลองแบบควบคุมภายในงานวิจัย

(1) การสำรวจก่อนการทดลอง (Pre-Experiment Survey)

การสำรวจก่อนการทดลองสามารถแบ่งออกเป็น 2 ส่วน คือ สำรวจโดยใช้แบบสอบถามและสำรวจโดยใช้การสัมภาษณ์ สำหรับการสำรวจโดยใช้แบบสอบถามนั้นได้ทำการออกแบบไว้ในรูปแบบที่เป็นกระดาษ โดยให้นักศึกษาทำแบบสอบถามพร้อมกันในห้องเรียนก่อนทำการนำเสนอเรื่อง TDD แก่นักศึกษา โดยนักศึกษาใช้เวลาในการทำแบบสอบถามโดยเฉลี่ยคนละ 30 นาที และเก็บรวบรวมคืนหลังจากนักศึกษาทำแบบสอบถามเสร็จเรียบร้อยแล้ว ซึ่งในแบบสอบถามประกอบไปด้วยคำถามปลายเปิดและคำถามปลายปิดทั้งหมด 28 ข้อ

สำหรับการสำรวจโดยการสัมภาษณ์มีรูปแบบการใช้คำถามในการสัมภาษณ์แบบกึ่งโครงสร้าง คือ มีการวางแผนการสัมภาษณ์ไว้ล่วงหน้า โดยผู้ที่ทำการสัมภาษณ์เป็นผู้ที่มีความรู้ทางด้านวิศวกรรมซอฟต์แวร์จำนวน 4 คน และคำถามในการสัมภาษณ์นักศึกษานั้นทางผู้วิจัยได้กำหนดข้อคำถามที่ใช้ในการสัมภาษณ์ให้แก่ผู้ทำการสัมภาษณ์ โดยเนื้อหาของการสัมภาษณ์เป็นการสอบถามในเรื่องความสนใจในการพัฒนาซอฟต์แวร์และรูปแบบการพัฒนาซอฟต์แวร์ที่นักศึกษาเคยใช้และเรียนก่อนทำการทดลอง ซึ่งผู้สัมภาษณ์สามารถถามคำถามเพิ่มเติมจากคำถามที่กำหนดไว้ได้ สำหรับรูปแบบการ

สัมภาษณ์นั้นจะเป็นแบบหนึ่งต่อหนึ่ง คือ ผู้สัมภาษณ์ 1 คนต่อการสัมภาษณ์นักศึกษา 1 คนต่อครั้ง โดยกำหนดเวลาในการสัมภาษณ์ไม่เกิน 30 นาทีต่อคน โดยมีการเก็บข้อมูลด้วยวิธีการบันทึกเสียง (Record) และการจดบันทึก

(2) การแนะนำการพัฒนาซอฟต์แวร์แบบ TDD (TDD Training)

หลังจากนักศึกษาจัดทำแบบสอบถามและให้สัมภาษณ์เรียบร้อยแล้ว ผู้วิจัยจึงทำการนำเสนอเรื่อง TDD ด้วยการบรรยายเพื่อให้นักศึกษามีความรู้พื้นฐานเกี่ยวกับ TDD โดยหัวข้อในการบรรยายประกอบไปด้วย 1) ความรู้พื้นฐานเกี่ยวกับ TDD 2) การพัฒนาซอฟต์แวร์แบบ TDD 3) เครื่องมือในการพัฒนาซอฟต์แวร์แบบ TDD และ 4) ตัวอย่างการพัฒนาซอฟต์แวร์ด้วย TDD

(3) การพัฒนาซอฟต์แวร์ภายในงานวิจัย (Experiment Project)

สำหรับการพัฒนาซอฟต์แวร์ภายในงานวิจัยมีการพัฒนาซอฟต์แวร์ทั้งหมด 2 ครั้ง โดยใช้ห้องปฏิบัติการภายในมหาวิทยาลัยสงขลานครินทร์ที่ทำการติดตั้งโปรแกรมที่ใช้ในการทดลองเรียบร้อยแล้ว ซึ่งการพัฒนาซอฟต์แวร์แต่ละครั้งใช้เวลาในการพัฒนาซอฟต์แวร์จำนวน 2 ชั่วโมง โดยการทดลองทั้ง 2 ครั้ง ได้กำหนดให้นักศึกษาทำการพัฒนาโปรแกรม FizzBuzz (Fenton, 2014) เนื่องจากโปรแกรมนี้สามารถพัฒนาได้ง่าย ซึ่งจะส่งผลให้นักศึกษาสามารถเข้าใจถึงขั้นตอนในกระบวนการ TDD ได้ง่าย ซึ่ง FizzBuzz เป็นเกมที่ช่วยในการเรียนรู้ในเรื่องของการหารโดยกำหนดให้มีการรับค่าจำนวนเต็มหนึ่งค่า และมีข้อกำหนดทั้งหมด 4 กรณี คือ

กรณีที่ 1 จำนวนเต็มหารด้วย 3 ลงตัวให้แสดงคำว่า Fizz

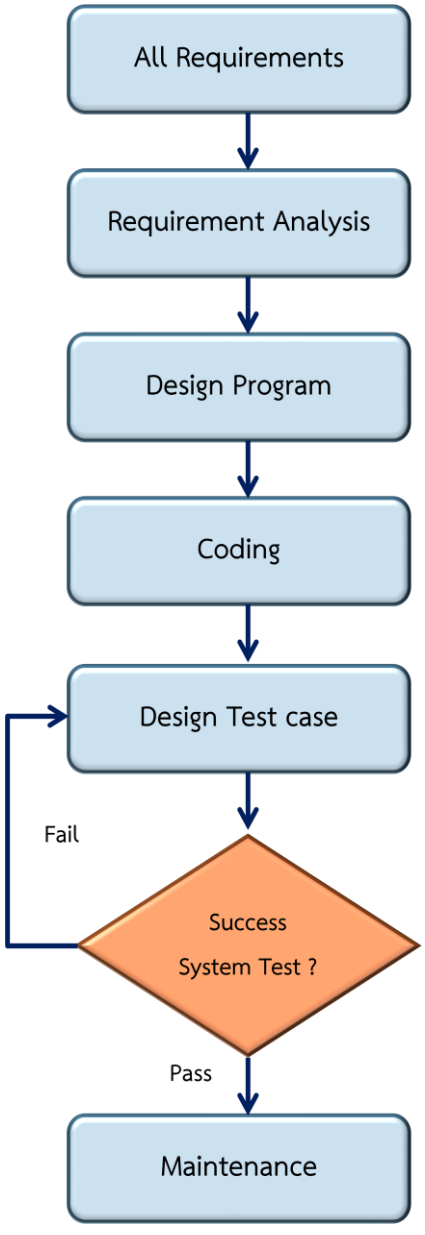
กรณีที่ 2 จำนวนเต็มหารด้วย 5 ลงตัวให้แสดงคำว่า Buzz

กรณีที่ 3 จำนวนเต็มหารด้วย 3 และ 5 ลงตัวให้แสดงคำว่า FizzBuzz

กรณีที่ 4 จำนวนเต็มเป็นเลขที่ไม่สามารถหารได้ตามทั้ง 3 กรณีข้างต้นให้แสดงค่าเลขจำนวนเต็มที่รับเข้ามา

สำหรับการพัฒนาซอฟต์แวร์ในครั้งที่ 1 นั้น นักศึกษาพัฒนาโปรแกรมโดยใช้วิธีการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และครั้งที่ 2นักศึกษาพัฒนาโปรแกรมโดยใช้วิธีการพัฒนาซอฟต์แวร์แบบ TDD โดยการพัฒนาโปรแกรมแต่ละครั้งมีรายละเอียดดังนี้

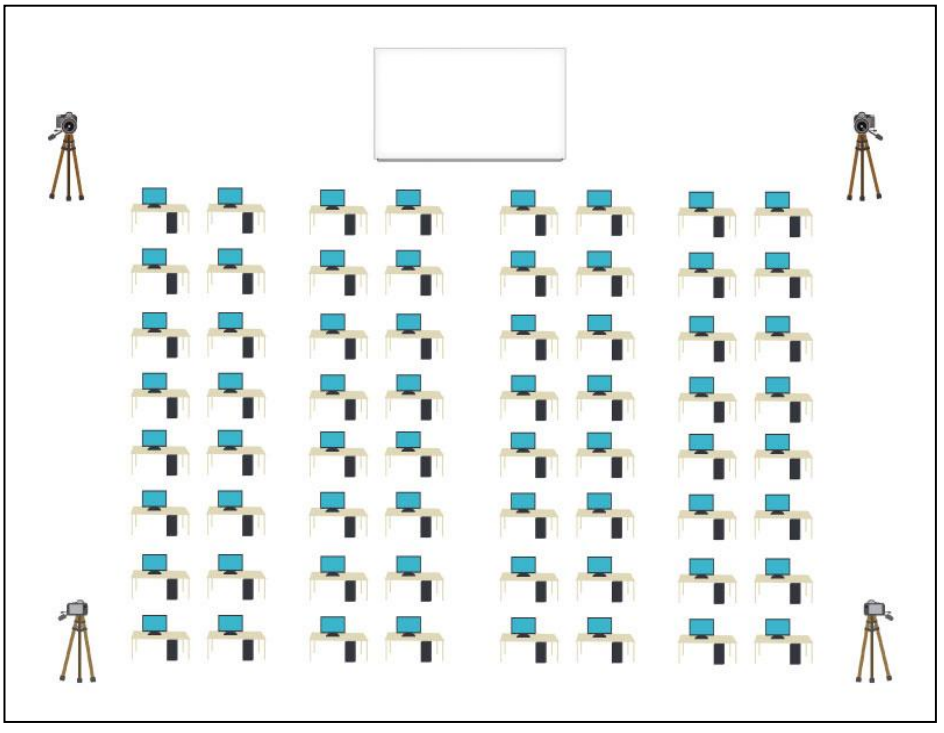
ครั้งที่ 1 ให้นักศึกษาทำการเขียนโปรแกรมโดยให้พัฒนาโปรแกรม FizzBuzz โดยให้นักศึกษาใช้วิธีการพัฒนาด้วยวิธีการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก แสดงดังรูปที่ 3.3



รูปที่ 3.3 วิธีการในการพัฒนาซอฟต์แวร์แบบการจำลองน้ำตก

จากการทดลองครั้งที่ 1 นั้น นักศึกษาสามารถเลือกใช้เครื่องมือในการเขียนโปรแกรมได้ตามความต้องการ ในขณะที่นักศึกษาทำการพัฒนาโปรแกรมภายในห้องปฏิบัติการจะมีการบันทึกวิดีโอเพื่อสังเกตพฤติกรรมของนักศึกษา โดยผู้วิจัยได้ใช้กล้องวิดีโอจำนวน 4 ตัว ตั้งภายในห้องปฏิบัติการเพื่อทำการบันทึก แสดงดังรูปที่ 3.4 นอกจากการบันทึกวิดีโอแล้วในห้องปฏิบัติการยังมีผู้สังเกตการณ์อีกจำนวน 4 คน เพื่อทำการสังเกตพฤติกรรมของนักศึกษาโดยทำการยืนสังเกตการณ์เป็นเวลา 5 นาทีต่อ

นักศึกษา 1 คน และทำการจดบันทึกพฤติกรรมของนักศึกษาในแบบฟอร์มการสังเกตการณ์ตามที่ผู้วิจัยได้แจกให้แก่ผู้สังเกตการณ์

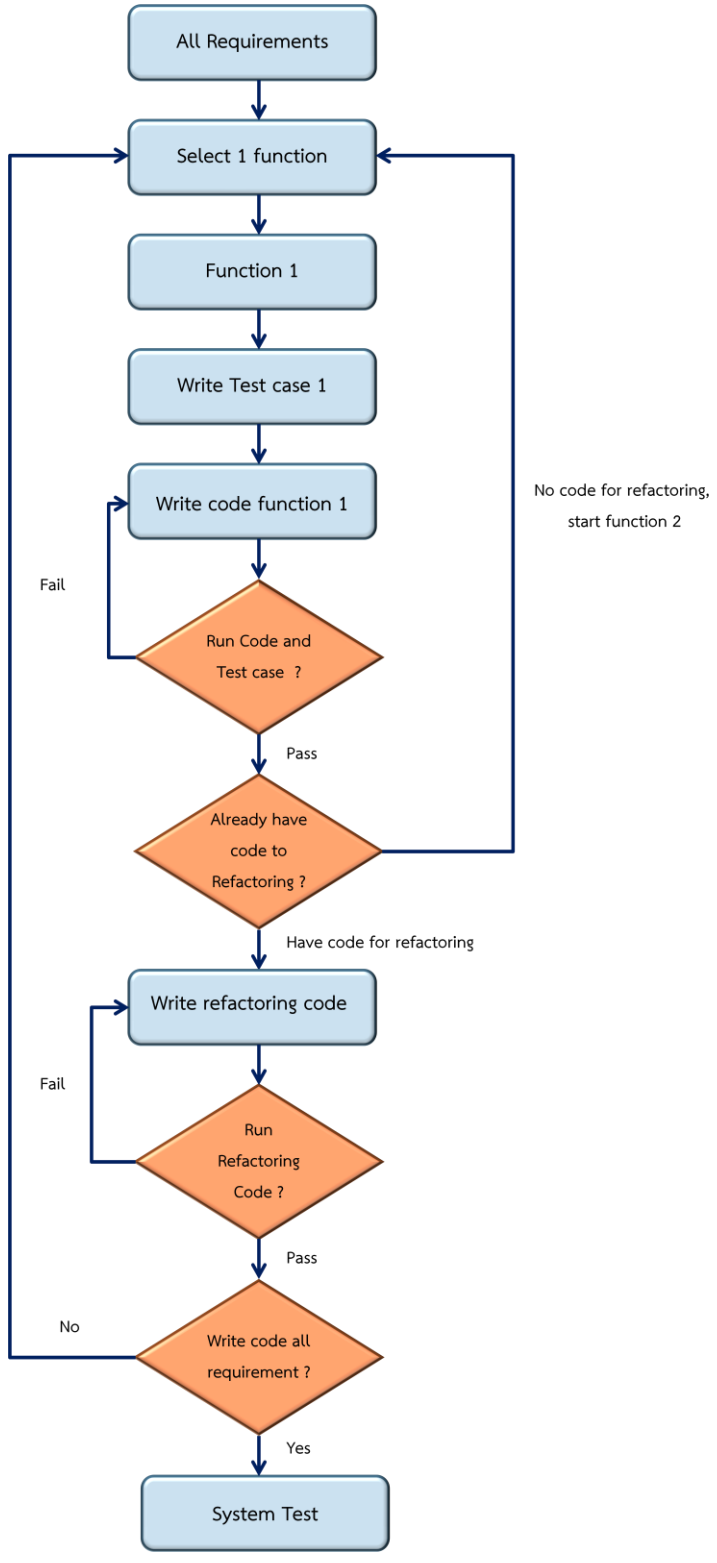


รูปที่ 3.4 แสดงการตั้งกล้องวิดีโอเพื่อสังเกตพฤติกรรมของนักศึกษาขณะทำการทดลอง

ในการทดลองครั้งที่ 2 นักศึกษาทำการเขียนโปรแกรมโดยให้พัฒนาโปรแกรม FizzBuzz โดยใช้วิธีการ TDD โดยกำหนดให้นักศึกษาใช้เครื่องมือในการเขียนโปรแกรม คือ Junit[1] เวอร์ชัน 4 ซึ่ง Junit เป็นซอฟต์แวร์โอเพนซอร์สในการทดสอบหน่วยย่อยของโปรแกรม เช่น เมธอด ในลักษณะอัตโนมัติ โดยเฉพาะโปรแกรมที่เขียนโดยภาษาจาวา ซึ่ง Junit ที่ใช้นี้เป็น plug in เพิ่มเติมใน Eclipse ซึ่งเป็น IDE ในการพัฒนาโปรแกรม โดย Eclipse[2] มีคำสั่งต่าง ๆ สำหรับการทดสอบโปรแกรมและมีเครื่องมือที่ช่วยให้การสร้างการทดสอบโปรแกรม

[1] สามารถดาวน์โหลดโปรแกรมได้ในเว็บไซต์ www.junit.org

[2] สามารถดาวน์โหลดโปรแกรมได้ในเว็บไซต์ https://eclipse.org



รูปที่ 3.5 วิธีการในการพัฒนาซอฟต์แวร์แบบ TDD

สำหรับในการทดลองครั้งที่ 2 ผู้วิจัยให้นักศึกษาทำการพัฒนาโปรแกรมตามแบบฝึกหัด ซึ่งในแบบฝึกหัดมีแบบฝึกหัดย่อยจำนวน 2 ข้อ โดยข้อที่ 1 เป็นการให้นักศึกษาทำการติดตั้ง Junit เวอร์ชัน 4 ภายใน Eclipse ด้วยตนเองและทำการทดสอบ Junit ว่าสามารถทำงานได้ปกติหรือไม่ ส่วนในข้อที่ 2 มีการกำหนดให้นักศึกษาพัฒนาโปรแกรม FizzBuzz โดยใช้วิธีการ TDD ในการพัฒนาซึ่งการใช้วิธีการแบบ TDD โดยมีขั้นตอนดังรูปที่ 3.5 และในขณะที่นักศึกษาทำการพัฒนาโปรแกรมในห้องปฏิบัติการจะมีการบันทึกวิดีโอเพื่อสังเกตพฤติกรรมของนักศึกษาโดยใช้กล้องวิดีโอจำนวน 4 ตัว ที่ตั้งภายในห้องปฏิบัติการซึ่งแสดงดังรูปที่ 3.4 และมีผู้สังเกตการณ์จำนวน 4 คน ที่ทำการสังเกตพฤติกรรมของนักศึกษาโดยทำการยืนสังเกตการณ์เป็นเวลา 5 นาทีต่อนักศึกษา 1 คน และทำการจดบันทึกพฤติกรรมของนักศึกษาในแบบฟอร์มการสังเกตการณ์ตามที่ผู้วิจัยได้แจกให้แก่ผู้สังเกตการณ์

(4) การสำรวจหลังการทดลอง (Post-Experiment Survey)

หลังจากนักศึกษาได้ทำแบบฝึกหัดเสร็จสิ้นแล้ว ผู้วิจัยได้ทำการสำรวจโดยใช้แบบสอบถามและการสัมภาษณ์ สำหรับการสำรวจโดยใช้แบบสอบถามที่ได้ทำการออกแบบไว้ในรูปแบบแบบสอบถามที่เป็นกระดาษนั้น คำถามหลังการทดลองเป็นแบบสอบถามชุดเดียวกับแบบสอบถามก่อนการทดลองแต่มีคำถามเพิ่มเติมเรื่องผลของการใช้ TDD โดยให้นักศึกษาทำแบบสอบถามพร้อมกันภายในห้องปฏิบัติการทันทีหลังจากนักศึกษาพัฒนาโปรแกรมด้วยวิธีการ TDD ในห้องปฏิบัติการเสร็จสิ้น โดยนักศึกษาใช้เวลาในการทำแบบสอบถามโดยเฉลี่ยคนละ 30 นาที

สำหรับการสำรวจโดยการสัมภาษณ์มีรูปแบบการใช้คำถามในการสัมภาษณ์แบบกึ่งโครงสร้าง โดยผู้ที่ทำการสัมภาษณ์เป็นผู้ที่มีความรู้ทางด้านวิศวกรรมซอฟต์แวร์จำนวน 4 คน ซึ่งคำถามสำหรับทำการสัมภาษณ์นักศึกษานั้น ผู้วิจัยได้กำหนดข้อคำถามที่ใช้ในการสัมภาษณ์ให้แก่ผู้ทำการสัมภาษณ์ โดยเนื้อหาของการสัมภาษณ์เป็นการสอบถามในเรื่องความสนใจในการพัฒนาซอฟต์แวร์และผลของการนำ TDD มาใช้ในการทดลอง โดยรูปแบบในการสัมภาษณ์เป็นการสัมภาษณ์แบบหนึ่งต่อหนึ่ง คือ ผู้สัมภาษณ์ 1 คนต่อการสัมภาษณ์นักศึกษา 1 คนต่อครั้ง โดยกำหนดเวลาในการสัมภาษณ์ใช้เวลาไม่เกิน 30 นาทีต่อคน โดยใช้การเก็บข้อมูลด้วยวิธีการบันทึกเสียง (Record) และมีการจดบันทึกโดยผู้ที่ทำการสัมภาษณ์นักศึกษา

3.3 การวิเคราะห์ข้อมูลหลังการทดลอง (Post-Controlled Experiment)

จากข้อมูลที่ได้จากการทดลองในรูปแบบแบบสอบถามแบบ Likert Scale และแบบสอบถามปลายเปิด การสัมภาษณ์ และการสังเกตการณ์ โดยข้อมูลที่ได้จากแบบสอบถามแบบ Likert Scale เป็นข้อมูลเชิงปริมาณซึ่งทางผู้วิจัยได้นำข้อมูลในส่วนนี้ไปทำการวิเคราะห์ผลทางสถิติ ส่วนข้อมูลที่ได้จากแบบสอบถามปลายเปิด การสัมภาษณ์ และการสังเกตการณ์ ซึ่งเป็นข้อมูลเชิงคุณภาพ โดยผู้วิจัยได้ทำการจัดการข้อมูลก่อนนำไปวิเคราะห์ผล ดังนี้

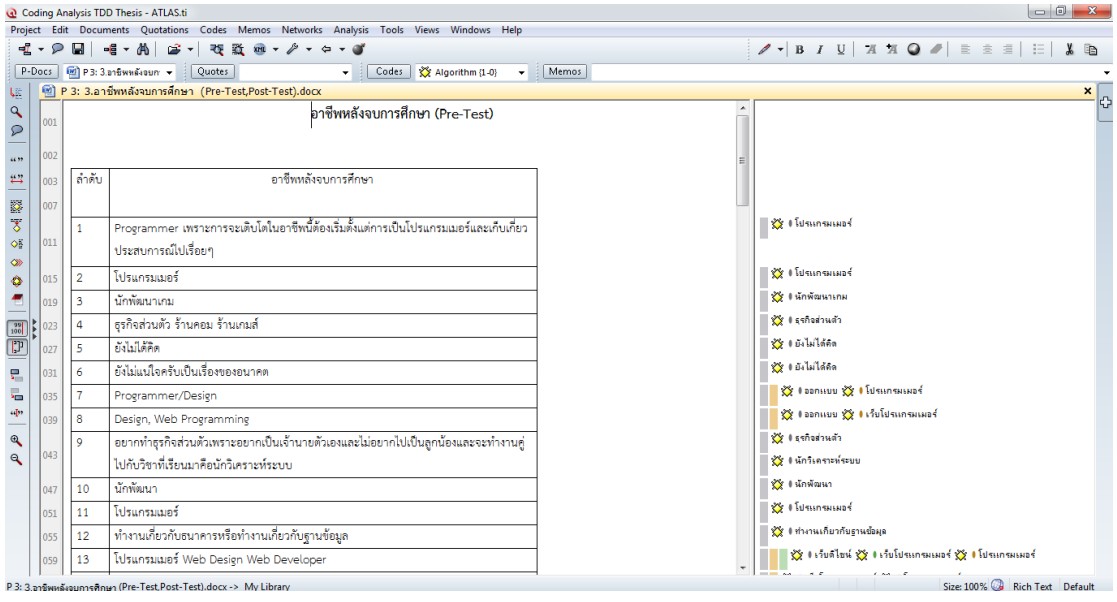
3.3.1 ข้อมูลจากแบบสอบถามปลายเปิด เป็นข้อมูลที่อยู่ในรูปแบบกระดาษ ผู้วิจัยจึงจัดการข้อมูลให้อยู่ในรูปแบบตารางด้วย Microsoft Excel โดยการเก็บเป็นไฟล์ตามข้อคำถามเพื่อนำข้อมูลไปวิเคราะห์ต่อโดยใช้ซอฟต์แวร์ ATLAS.ti ในการวิเคราะห์ด้วยวิธีการ Coding Analysis

3.3.2 ข้อมูลจากการสัมภาษณ์ เป็นข้อมูลการสัมภาษณ์ในลักษณะรูปแบบไฟล์เสียงจึงสามารถใช้ซอฟต์แวร์ ATLAS.ti เพื่อวิเคราะห์ข้อมูลได้เลย เนื่องจากซอฟต์แวร์ ATLAS.ti เป็นซอฟต์แวร์ที่รองรับการนำไฟล์เสียงในการวิเคราะห์ข้อมูลอยู่แล้ว

3.3.3 ข้อมูลจากการสังเกตการณ์ เป็นข้อมูลการสังเกตการณ์ในลักษณะรูปแบบกระดาษและวิดีโอ ซึ่งข้อมูลในแบบฟอร์มสังเกตการณ์ที่อยู่ในกระดาษผู้วิจัยได้จัดการนำข้อมูลให้อยู่ในลักษณะตารางด้วย Microsoft Excel เพื่อนำข้อมูลไปวิเคราะห์ต่อโดยใช้ซอฟต์แวร์ ATLAS.ti ส่วนข้อมูลในรูปแบบวิดีโอสามารถนำเข้าซอฟต์แวร์ ATLAS.ti เพื่อเข้าทำการวิเคราะห์ข้อมูลได้เลย

ในงานวิจัยนี้ผู้วิจัยได้ใช้ซอฟต์แวร์ ATLAS.ti ในการวิเคราะห์ข้อมูลทั้ง 3 ส่วน เนื่องจากซอฟต์แวร์นี้เป็นซอฟต์แวร์ที่มีความสามารถในการวิเคราะห์ข้อมูลเชิงคุณภาพ โดยสามารถรองรับข้อมูลที่จะนำมาวิเคราะห์ได้หลากหลายรูปแบบ เช่น ข้อความตัวอักษร เสียง และวิดีโอ หรือภาพเคลื่อนไหว ผู้วิจัยได้ใช้ซอฟต์แวร์ ATLAS.ti สำหรับการวิเคราะห์ข้อมูลด้วยวิธีการ Coding Analysis (Seaman, 1999)

ซึ่งเป็นการถอดรหัสข้อความโดยการกำหนดรูปแบบของกลุ่มในการถอดรหัส (Theme) เพื่อเป็นการแบ่งข้อมูลออกเป็นส่วนย่อยจากข้อความที่เป็นประโยคซึ่งเป็นการลดความซับซ้อนของข้อมูล และทำให้ผู้วิจัยสามารถจับประเด็นสำคัญของข้อมูลได้ โดยเมื่อทำการจัดกลุ่มของข้อความได้แล้วจะสามารถนำข้อความจากการจัดกลุ่มไปทำการวิเคราะห์ผลหรือสรุปผลเป็นประเด็นได้ ตัวอย่างเช่นในรูปที่ 3.6 แสดงตัวอย่างการจัดกลุ่มของข้อความเพื่อจัดกลุ่มคำตอบของคำถามเรื่อง อาชีพที่นักศึกษาต้องการเข้าทำงานหลังจบการศึกษา ซึ่งมีการกล่าวถึงการเป็นนักพัฒนาซอฟต์แวร์ การเป็นโปรแกรมเมอร์ ซึ่งจัดอยู่ในรูปแบบเดียวกัน คือ อาชีพเกี่ยวกับการเขียนโปรแกรม



รูปที่ 3.6 แสดงตัวอย่างการจัดกลุ่มคำตอบของนักศึกษาในการตอบแบบสอบถามเรื่องอาชีพที่นักศึกษาต้องการทำงานหลังเรียนจบ

บทที่ 4

ผลการวิจัย

จากการดำเนินงานวิจัย ผู้วิจัยสามารถนำผลที่ได้มาวิเคราะห์ ซึ่งสามารถแบ่งผลการดำเนินการวิจัยเป็น 3 ส่วน คือ 1) การตรวจสอบตัวบ่งชี้ด้านความสนใจด้วยการวิเคราะห์องค์ประกอบเชิงยืนยันภายในแบบสอบถาม 2) ผลการดำเนินการวิจัยจากแบบสอบถามและการสัมภาษณ์ และ 3) ผลการดำเนินการวิจัยจากการสังเกตการณ์ โดยมีรายละเอียดของผลจากการดำเนินงานดังนี้

4.1 การตรวจสอบตัวบ่งชี้ด้านความสนใจด้วยการวิเคราะห์องค์ประกอบเชิงยืนยันในแบบสอบถาม

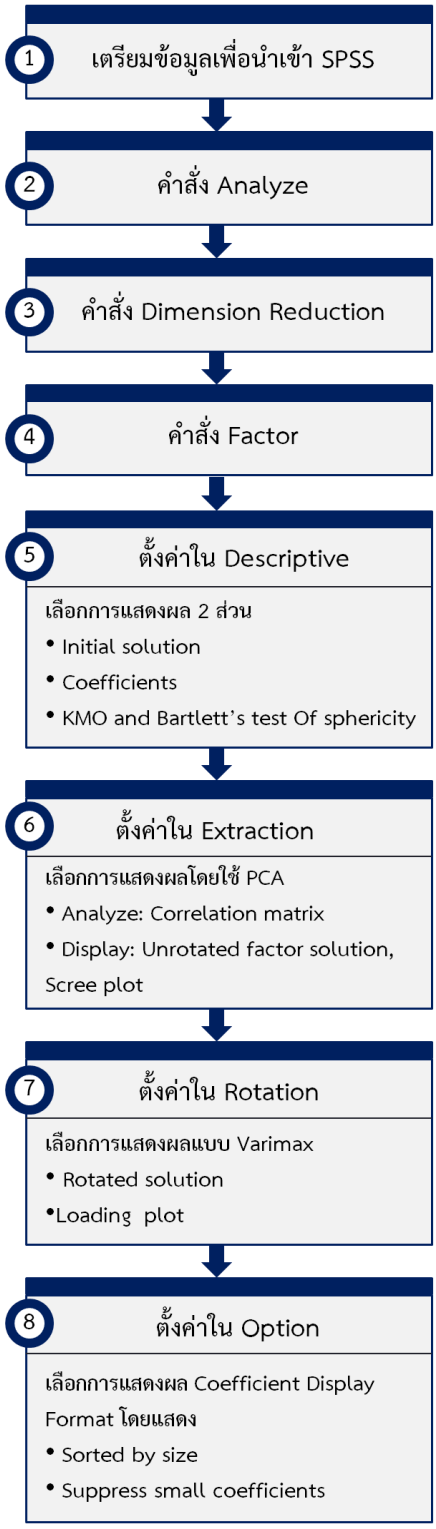
การวิเคราะห์องค์ประกอบเชิงยืนยัน (Confirmatory Factor Analysis) ในแบบสอบถาม มีวัตถุประสงค์เพื่อตรวจสอบตัวบ่งชี้ด้านความสนใจที่ผู้วิจัยกำหนดขึ้นจากนิยามเรื่องความสนใจว่าตัวบ่งชี้ทั้ง 5 ตัวบ่งชี้ ที่แสดงดังตารางที่ 4.1 จัดอยู่ในองค์ประกอบเดียวกันหรือไม่

ตารางที่ 4.1 แสดงตัวบ่งชี้ที่ใช้ในการตั้งคำถามในแบบสอบถาม

รหัส	ตัวบ่งชี้
I1	ความกระตือรือร้นในสิ่งที่สนใจ
I2	ศึกษาเพิ่มเติมด้วยตนเอง
I3	การทำแบบฝึกหัด
I4	เวลาในการทำแบบฝึกหัด
I5	เวลาในการศึกษาเพิ่มเติมด้วยตนเอง

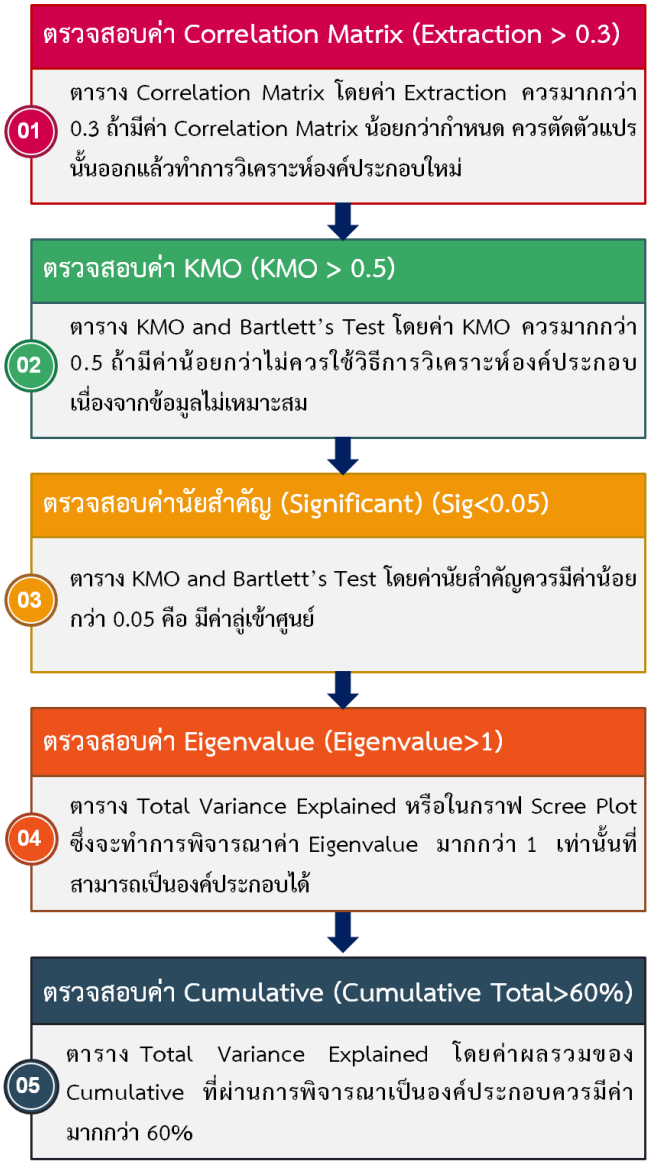
สำหรับการวิเคราะห์องค์ประกอบ “ความสนใจ” ในงานวิจัยนั้น ผู้วิจัยเลือกใช้วิธีการวิเคราะห์องค์ประกอบเชิงยืนยัน เนื่องจากการวิเคราะห์องค์ประกอบเชิงยืนยันมีวัตถุประสงค์เพื่อเป็นการพิสูจน์ ตรวจสอบและยืนยันทฤษฎีที่ผู้อื่นทำการค้นพบ โดยในงานวิจัยนี้ คือ ทฤษฎีด้านความสนใจ (Fox, 1983) ซึ่งตรงกับวัตถุประสงค์ของผู้วิจัยที่ต้องการตรวจสอบตัวบ่งชี้ด้านความสนใจว่าตัวบ่งชี้ทั้ง 5 ตัวบ่งชี้จัดอยู่ในองค์ประกอบ “ความสนใจ” หรือไม่ โดยในการวิเคราะห์องค์ประกอบเชิงยืนยันในงานวิจัยผู้วิจัยใช้เครื่องมือในการวิเคราะห์ด้วยโปรแกรม SPSS[3] เวอร์ชัน 22.0 (Denzin and Lincoln, 1994) ซึ่งเป็นโปรแกรมที่ใช้ในการวิเคราะห์ข้อมูลทางสถิติ โดยการวิเคราะห์องค์ประกอบเชิงยืนยันในงานวิจัยมีขั้นตอนในการวิเคราะห์แสดงดังรูปที่ 4.1 โดยรายละเอียดวิธีการในการวิเคราะห์องค์ประกอบและการพิจารณาค่าที่ได้จากการวิเคราะห์แสดงในภาคผนวก ค

[3] โปรแกรม IBM SPSS Statistics Base (SPSS) เป็นเครื่องมือในการวิเคราะห์ข้อมูลทางสถิติ



รูปที่ 4.1 ขั้นตอนในการวิเคราะห์องค์ประกอบด้วยโปรแกรม SPSS

เมื่อได้ค่าจากการคำนวณจากโปรแกรม SPSS แล้ว ผู้วิจัยทำการพิจารณาค่าที่ได้รับว่าจากข้อมูลที่นำมาวิเคราะห์ด้วยวิธีการวิเคราะห์องค์ประกอบ ได้ผลลัพธ์ที่ผ่านการพิจารณาหรือไม่ โดยค่าที่ต้องทำการพิจารณามีทั้งหมด 5 ส่วน (Yong and Pearce, 2013) แสดงดังรูปที่ 4.2



รูปที่ 4.2 ขั้นตอนในการพิจารณาค่าจากการวิเคราะห์องค์ประกอบ

การวิเคราะห์องค์ประกอบเชิงยืนยันในแบบสอบถามแบบ Likert Scale เป็นการนำคำถามทั้งหมด 8 ข้อ ที่แสดงดังตารางที่ 4.2 มาทำการวิเคราะห์องค์ประกอบเชิงยืนยัน โดยคะแนนที่นำมาใช้ในการวิเคราะห์องค์ประกอบเชิงยืนยันเป็นคะแนนเฉลี่ยก่อนและหลังการทดลองของนักศึกษา

ตารางที่ 4.2 แสดงคำถามในรูปแบบ Likert Scale และรหัสของข้อคำถาม

คำถามในรูปแบบ Likert Scale	**รหัส**
คำถามข้อ 1 นักศึกษาให้ความสนใจในการเรียนในรายวิชาด้านการพัฒนาซอฟต์แวร์	Question 1
คำถามข้อ 2 นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากแบบฝึกหัด (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)	Question 2
คำถามข้อ 3 นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากการศึกษาหาข้อมูลด้วยตนเอง (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)	Question 3
คำถามข้อ 4 นักศึกษาให้ความสนใจในการศึกษาเกี่ยวกับวิศวกรรมซอฟต์แวร์เพิ่มเติมด้วยตนเอง (การศึกษาเพิ่มเติมด้านวิศวกรรมซอฟต์แวร์ เช่น การศึกษาเพิ่มเติมถึงกระบวนการพัฒนาซอฟต์แวร์รูปแบบใหม่ ภาษาในการเขียนโปรแกรม เป็นต้น)	Question 4
คำถามข้อ 5นักศึกษาให้ความสนใจในการใช้เวลาในการทำแบบฝึกหัด	Question 5
คำถามข้อ 6 นักศึกษามีความสนใจในการดัดแปลงการแสดงผลจากแบบฝึกหัดที่ได้รับมอบหมาย	Question 6
คำถามข้อ 7 นักศึกษาให้ความสนใจในการใช้เวลาในการศึกษาด้วยตนเอง	Question 7
คำถามข้อ 8 นักศึกษาให้ความสนใจในการมีส่วนร่วมในการถาม-ตอบภายในห้องเรียน	Question 8

จากการวิเคราะห์องค์ประกอบเชิงยืนยันได้ผลในการวิเคราะห์ทั้งหมด 4 ส่วน คือ 1) Descriptive Statistics 2) KMO and Bartlett's Test 3) Total Variance Explained และ 4) Component Matrix[a] โดยมีรายละเอียดจากการวิเคราะห์ข้อมูลดังนี้

4.1.1 Descriptive Statistics

ค่าจาก Descriptive Statistics เป็นการอธิบายคุณลักษณะของสิ่งที่ทำการศึกษาโดยทำการแสดงจำนวนข้อมูล ค่าเฉลี่ย และค่าเบี่ยงเบนมาตรฐานของตัวแปรที่ทำการศึกษาในแบบสอบถามแบบ Likert Scale ดังตารางที่ 4.3 ซึ่งแสดงข้อมูลค่าเฉลี่ย (Mean) ค่าเบี่ยงเบนมาตรฐาน (Std. Deviation) และจำนวนข้อมูล โดยจำนวนข้อมูลเป็นจำนวนผู้ที่ตอบแบบสอบถามซึ่งมีทั้งหมด 42 คน (Analysis N)

ตารางที่ 4.3 แสดงจำนวนข้อมูล ค่าเฉลี่ย และค่าเบี่ยงเบนมาตรฐานภายในแบบสอบถาม Likert Scale

Variable	Mean	Std. Deviation	Analysis N
Question 1	3.738	0.7262	42
Question 2	3.500	0.7490	42
Question 3	3.429	0.8379	42
Question 4	3.238	0.8642	42
Question 5	3.548	0.7309	42
Question 6	3.500	0.8625	42
Question 7	3.405	0.7670	42
Question 8	3.179	0.8251	42

4.1.2 KMO and Bartlett's Test

KMO and Bartlett's Test เป็นการแสดงค่า KMO (Kaiser-Mayer-Olkin) และ Barlett's test โดย KMO เป็นค่าที่ใช้วัดความเหมาะสมของข้อมูลตัวอย่างที่จะนำมาวิเคราะห์โดยใช้เทคนิคการวิเคราะห์องค์ประกอบ ซึ่งโดยทั่วไปถ้าค่า KMO $<$ 0.5 จะถือว่าข้อมูลไม่เหมาะสมที่จะใช้เทคนิคการวิเคราะห์

องค์ประกอบ และ Barlett's test เป็นค่าสถิติที่ใช้ทดสอบสมมติฐานที่ตั้งขึ้น โดยผู้วิจัยได้ตั้งสมมติฐาน Barlett's test ดังนี้

H_0 : ตัวบ่งชี้ที่ใช้ในการตั้งคำถามไม่มีความสัมพันธ์กัน

H_1 : ตัวบ่งชี้ที่ใช้ในการตั้งคำถามมีความสัมพันธ์กัน

ตารางที่ 4.4 แสดงค่า KMO and Bartlett's Test ภายในแบบสอบถาม Likert Scale

Kaiser-Meyer-Olkin Measure of Sampling Adequacy		0.858
Bartlett's Test of Sphericity	Approx. Chi-Square	276.882
	Df	28
	Sig.	0.000

จากการวิเคราะห์ข้อมูลจากแบบสอบถามแบบ Likert Scale พบว่าค่า KMO ในแบบสอบถามแบบ Likert Scale ที่แสดงดังตารางที่ 4.4 มีค่า KMO เท่ากับ 0.858 ซึ่งมากกว่า 0.5 ดังนั้นจึงสามารถสรุปได้ว่า *ข้อมูลมีความเหมาะสมที่จะใช้เทคนิคการวิเคราะห์องค์ประกอบ*

ในทำนองเดียวกันค่า Barlett's test มีค่าน้อยกว่า 0.05 จึงทำการปฏิเสธ H_0 และยอมรับ H_1 ดังนั้นจึงสามารถสรุปได้ว่า *ตัวบ่งชี้ที่ใช้ในการตั้งคำถามมีความสัมพันธ์กัน จึงสามารถใช้การวิเคราะห์องค์ประกอบในการวิเคราะห์ผลต่อไปได้*

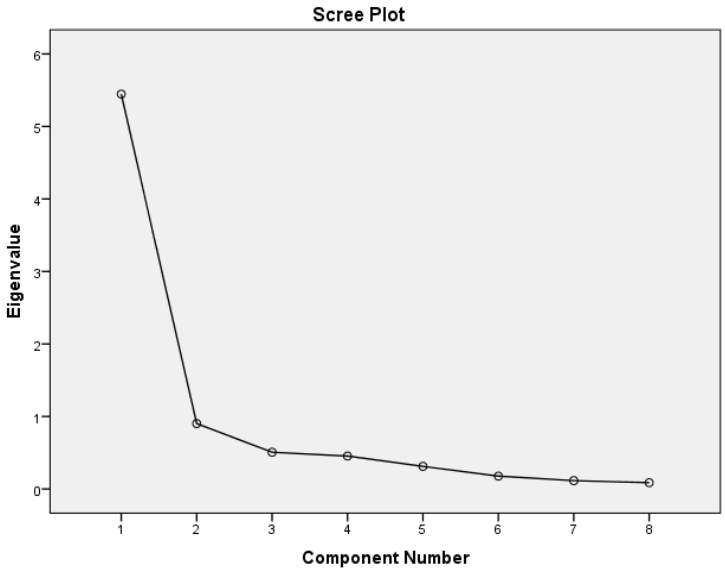
4.1.3 Total Variance Explained

Total Variance Explained เป็นการแสดงค่าทางสถิติสำหรับแต่ละองค์ประกอบทั้งก่อนและหลังการสกัดปัจจัยด้วยวิธีการ Principal Component Analysis ซึ่งแสดงข้อมูลดังตารางที่ 4.5

ตารางที่ 4.5 แสดงค่า Total Variance Explained ในแบบสอบถาม Likert Scale

Component	Initial Eigenvalues			Extraction Sums of Squared Loadings		
	Total	% of Variance	Cumulative %	Total	% of Variance	Cumulative %
1	5.446	68.079	68.079	5.446	68.079	68.079
2	0.903	11.283	79.362			
3	0.507	6.336	85.698			
4	0.454	5.673	91.371			
5	0.312	3.895	95.266			
6	0.177	2.216	97.482			
7	0.115	1.435	98.916			
8	0.087	1.084	100			

จากข้อมูลในตารางที่ 4.5 แสดงค่า Component ทั้งหมด 8 Components ซึ่งค่า Component แต่ละตัว คือ องค์ประกอบหรือปัจจัยที่นำมาสกัดปัจจัย และค่า Eigenvalue เป็นค่าความแปรปรวนทั้งหมดในตัวแปรเดิมที่สามารถอธิบายได้ โดยจะทำการพิจารณา Component ที่มีค่า Eigenvalue > 1 เท่านั้น จากตารางที่ 4.5 มี Component ที่มีค่า Eigenvalue > 1 เพียง 1 Component คือ Component ที่ 1 สามารถแสดงในรูปแบบกราฟ Scree plot ดังรูปที่ 4.3



รูปที่ 4.3 กราฟ Scree plot ของค่า Eigenvalue ภายในแบบสอบถามแบบ Likert Scale

4.1.4 Component Matrix[a]

Component Matrix[a] เป็นการแสดงค่าสัมประสิทธิ์ (factor loading) ที่เป็นค่าแสดงความสัมพันธ์ของตัวแปรที่ผ่านการพิจารณาการเป็นองค์ประกอบ แสดงดังตารางที่ 4.6 โดยเรียงลำดับจากตัวแปรที่มีค่าสัมประสิทธิ์มากที่สุดไปยังตัวแปรที่มีค่าสัมประสิทธิ์น้อยที่สุด

ตารางที่ 4.6 แสดงผลจาก Component Matrix[a] ภายในแบบสอบถามแบบ Likert Scale

Variable	Component
	1
Question 7	0.909
Question 4	0.898
Question 2	0.894
Question 3	0.863
Question 1	0.837
Question 5	0.784
Question 6	0.726
Question 8	0.653

จากตารางที่ 4.6 ค่าของสัมประสิทธิ์ความสัมพันธ์ของตัวแปรกับองค์ประกอบที่ผ่านการพิจารณาเพียง 1 องค์ประกอบ โดยทำการพิจารณาค่าตัวแปรว่าตัวแปรใดควรอยู่ในองค์ประกอบใดซึ่งพิจารณาจากค่า factor loading ถ้าค่า factor loading ของตัวแปรมีค่าเข้าสู่ ±1 แสดงว่าค่าดังกล่าวควรจัดอยู่ในองค์ประกอบ จากตารางที่ 4.6 แสดงให้เห็นว่าค่าที่ได้เป็นค่าที่จัดอยู่ในองค์ประกอบที่ 1 ทั้งหมด โดยแสดงค่าเรียงลำดับจากค่าที่มีค่า factor loading สูงสุด คือ นักศึกษาให้ความสนใจในการใช้เวลาในการศึกษาด้วยตนเอง (Question 7) ซึ่งมีค่าเท่ากับ 0.909 แต่ไม่มีค่าที่ต่ำกว่า 0.5 ดังนั้นตัวแปรทั้งหมดจึงจัดอยู่ใน 1 องค์ประกอบ

หลังจากที่ทำการจัดกลุ่มตัวแปรในการวิเคราะห์องค์ประกอบเชิงยืนยันแล้ว จะทำการนำค่าของตัวแปรที่อยู่ในองค์ประกอบแต่ละองค์ประกอบมาทำการหาค่าของ Cronbach's Alpha ซึ่งจากการทำการวิเคราะห์ได้ค่าของ Cronbach's Alpha เท่ากับ 0.929

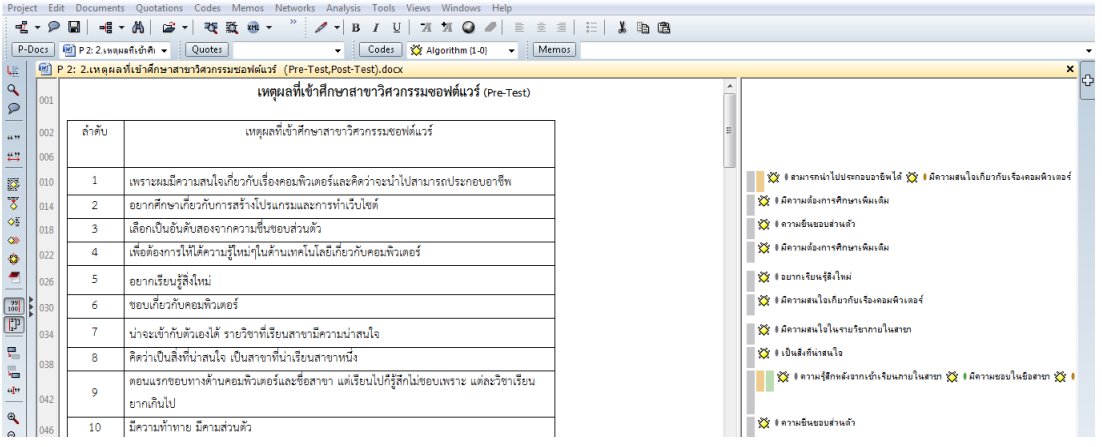
จากการทำการวิเคราะห์องค์ประกอบเชิงยืนยันพบว่าจากคำถามในแบบสอบถามแบบ Likert Scale ทั้ง 8 ข้อที่ตั้งขึ้นจากตัวบ่งชี้ทั้ง 5 ตัวบ่งชี้ในตารางที่ 4.2 จัดอยู่ในองค์ประกอบเดียวกัน และเมื่อทำการหาค่า Cronbach's Alpha พบว่าได้ค่าเท่ากับ 0.929 แสดงให้เห็นว่าตัวบ่งชี้ทั้ง 5 ตัวบ่งชี้จัดอยู่ในองค์ประกอบเดียวกันและตัวแปรมีความสัมพันธ์กันสูง ซึ่งจากค่าที่ได้จากการวิเคราะห์ผลด้วย Cronbach's Alpha แสดงให้เห็นว่าเครื่องมือที่ใช้ในการวัดผลด้านความสนใจ ซึ่งในที่นี้ คือ แบบสอบถาม ซึ่งสามารถวัดค่าและตรวจจับค่าด้านความสนใจได้ดี

4.2 ผลการดำเนินการวิจัยจากแบบสอบถามและการสัมภาษณ์

แบบสอบถามที่ใช้ในงานวิจัย 1 ชุดประกอบด้วยแบบสอบถาม 2 รูปแบบ คือ 1) แบบสอบถามปลายเปิด และ 2) แบบสอบถามแบบ Likert scale โดยแบบสอบถามได้มีการออกแบบจากตัวบ่งชี้ด้านความสนใจ ซึ่งมีการเก็บข้อมูลในรูปแบบต่างกัน โดยแบบสอบถามปลายเปิดเป็นการเก็บข้อมูลเชิงคุณภาพ และแบบสอบถามในรูปแบบ Likert scale เป็นการเก็บข้อมูลในเชิงปริมาณ โดยมีผู้ตอบแบบสอบถามทั้งหมด 42 คนจากผู้เข้าร่วมการทดลองทั้งหมด 52 คน สำหรับคำถามในการสัมภาษณ์นั้นมีการอ้างอิงคำถามมาจากแบบสอบถามปลายเปิด โดยการสัมภาษณ์ในงานวิจัยมีผู้เข้าร่วมทำการสัมภาษณ์ทั้งหมด 52 คน

4.2.1 ผลการดำเนินการวิจัยจากแบบสอบถามปลายเปิดและการสัมภาษณ์

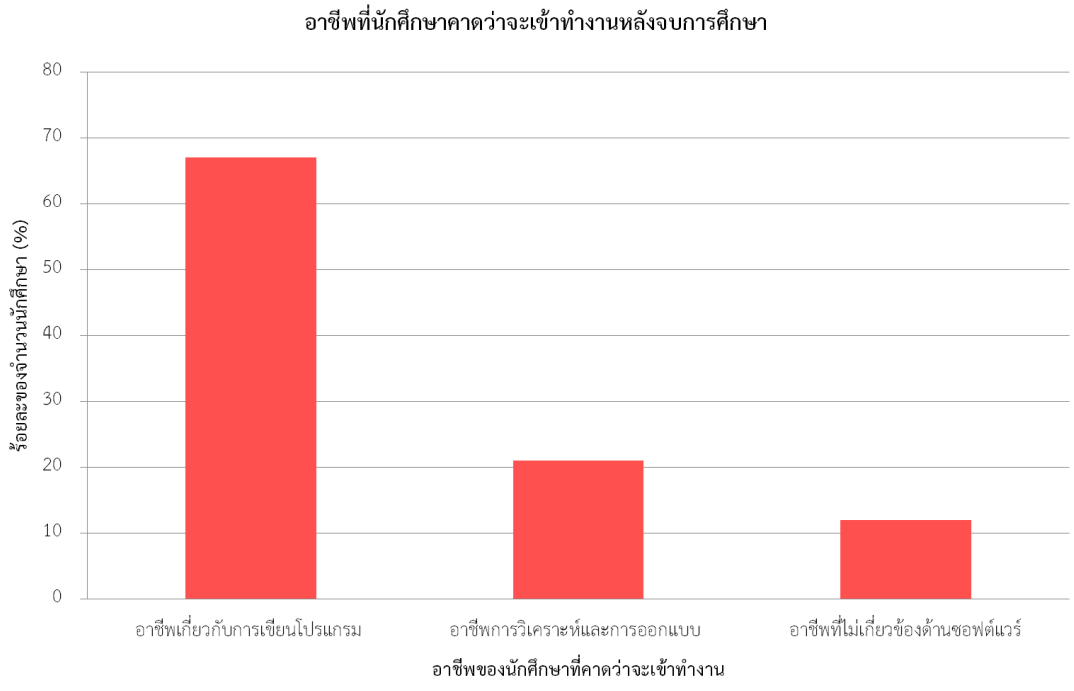
แบบสอบถามปลายเปิดและการสัมภาษณ์ในงานวิจัยมีการเก็บข้อมูลจำนวน 2 ครั้งคือ 1) การตอบแบบสอบถามและการสัมภาษณ์ก่อนทำการทดลอง และ 2) การตอบแบบสอบถามและการสัมภาษณ์หลังทำการทดลอง ซึ่งมีความแตกต่างกันในเรื่องของคำถามก่อนและหลังการทดสอบ โดยก่อนการทดลองเป็นคำถามเกี่ยวกับพื้นฐานการเขียนโปรแกรม วิธีการพัฒนาซอฟต์แวร์ และความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษา ส่วนคำถามหลังการทดลองเป็นการถามเกี่ยวกับผลของการนำ TDD ไปใช้ในการทดลองว่าส่งผลอย่างไรต่อนักศึกษา โดยคำตอบที่ได้จากแบบสอบถามปลายเปิดและการสัมภาษณ์เป็นข้อมูลเชิงบรรยายซึ่งทำการวิเคราะห์ข้อมูลด้วยซอฟต์แวร์ ATLAS.ti โดยผู้วิจัยได้วิเคราะห์ข้อมูลด้วยวิธีการ Coding Analysis คือ เป็นการจัดกลุ่มข้อมูลที่ได้จากแบบสอบถามและการสัมภาษณ์ที่เป็นรูปแบบเชิงบรรยายให้อยู่ในกลุ่มที่มีคำตอบลักษณะเดียวกันโดยการใส่เป็นรหัสกลุ่มแสดงดังรูปที่ 4.4 เมื่อทำการลงรหัสข้อมูลแล้วสามารถสรุปออกมาเป็นประเด็นได้ทั้งหมด 14 ประเด็น โดยทำการเปรียบเทียบกันระหว่างการเก็บข้อมูลที่ได้จากแบบสอบถามปลายเปิดและการสัมภาษณ์ มีรายละเอียดแต่ละประเด็นดังต่อไปนี้



รูปที่ 4.4 แสดงตัวอย่างการจัดกลุ่มข้อมูลเชิงบรรยายในแบบสอบถามปลายเปิด

ประเด็นที่ 1 *อาชีพที่นักศึกษาคาดว่าจะเข้าทำงานหลังจบการศึกษา* จากการตอบแบบสอบถามและการสัมภาษณ์สามารถแบ่งกลุ่มของรูปแบบงานที่นักศึกษาสนใจในการทำงานทั้งหมด 3 รูปแบบ คือ 1) อาชีพเกี่ยวกับการเขียนโปรแกรม 2) อาชีพการวิเคราะห์และการออกแบบ และ 3) อาชีพที่ไม่มีความเกี่ยวข้องด้านวิศวกรรมซอฟต์แวร์ จากรูปที่ 4.5 พบว่ามีนักศึกษาที่ต้องการประกอบอาชีพเกี่ยวกับ

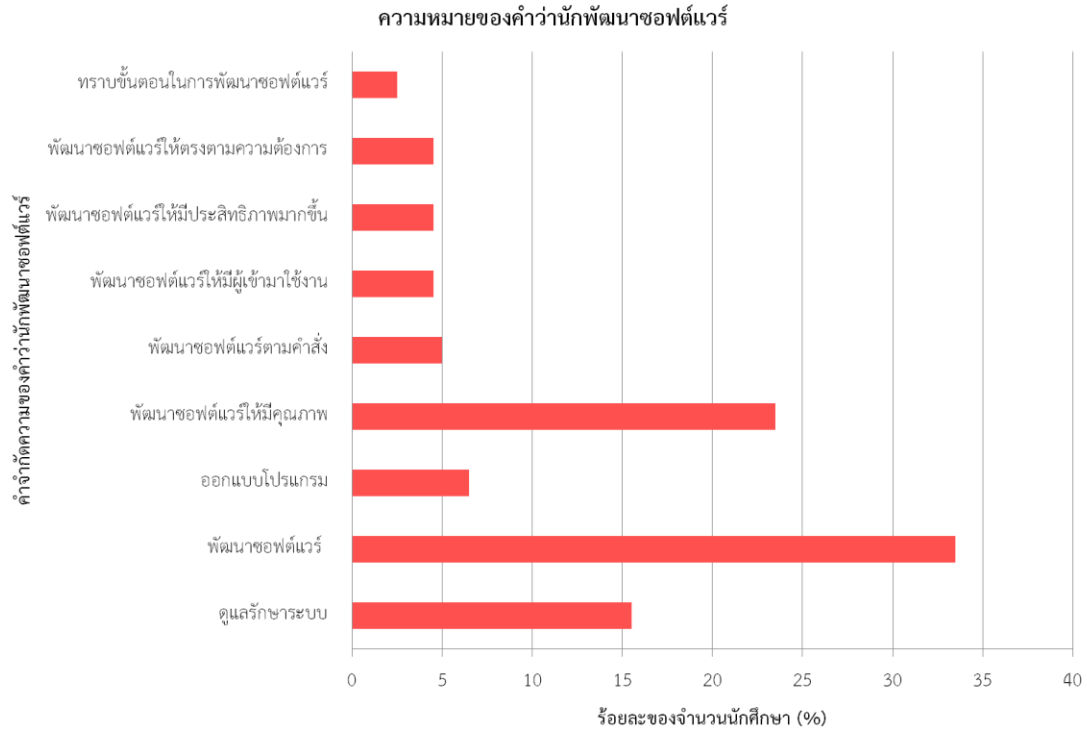
การเขียนโปรแกรมมากที่สุด คือ 67% นักศึกษาที่ต้องการประกอบอาชีพเกี่ยวกับการวิเคราะห์และการออกแบบ จำนวน 21% และมีนักศึกษาที่ต้องการประกอบอาชีพไม่มีความเกี่ยวข้องด้านวิศวกรรมซอฟต์แวร์ จำนวน 12%



รูปที่ 4.5 กราฟแสดงอาชีพที่นักศึกษาคาดว่าจะเข้าทำงานหลังจบการศึกษาภายในแบบสอบถามปลายเปิด

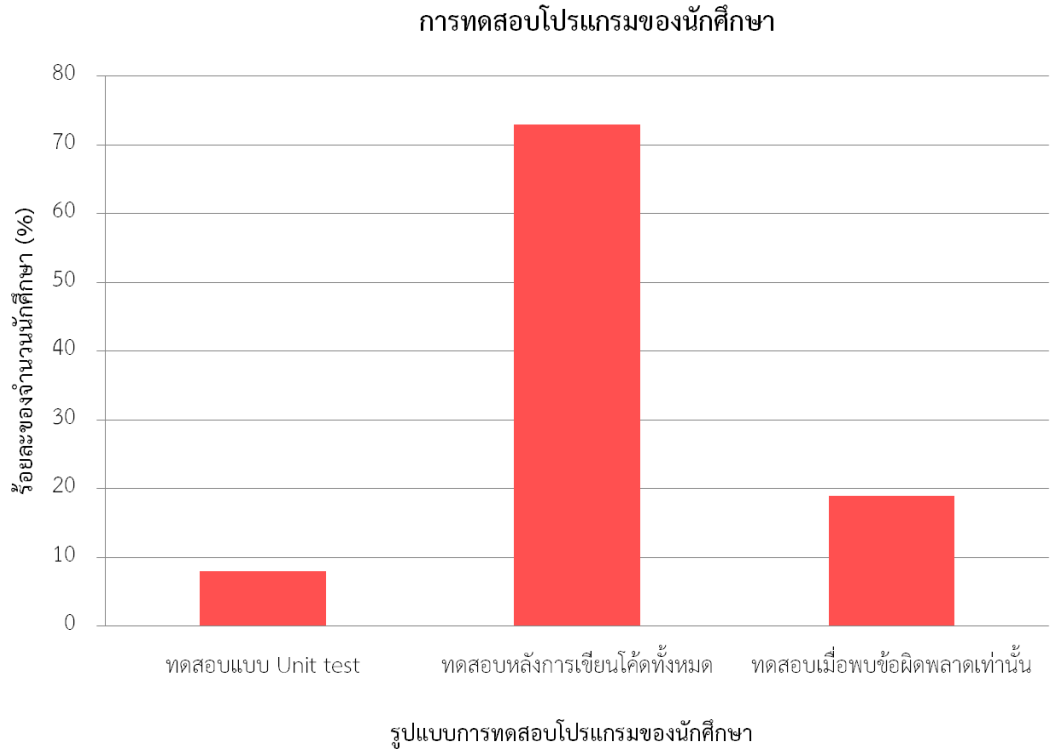
ประเด็นที่ 2 *ความเข้าใจถึงความหมายของคำว่า นักพัฒนาซอฟต์แวร์ ของนักศึกษา* จากการตอบแบบสอบถามและจากการสัมภาษณ์สามารถสรุปได้ว่า นักพัฒนาซอฟต์แวร์ คือ ผู้พัฒนาและออกแบบโปรแกรม แอปพลิเคชัน หรือซอฟต์แวร์ ที่มีอยู่เดิมให้ดีขึ้น หรือพัฒนาขึ้นมาใหม่ตามที่ได้รับคำสั่ง หรือพัฒนาขึ้นมาใหม่เพื่ออำนวยความสะดวกในชีวิตประจำวัน รวมถึงเป็นผู้ที่ต้องทำการดูแลรักษาระบบให้มีคุณภาพและยังต้องทราบทุกอย่างในกระบวนการพัฒนาซอฟต์แวร์ โดยความหมายของคำว่านักพัฒนาซอฟต์แวร์มาจากคำจำกัดความที่นักศึกษากล่าวไว้แสดงดังรูปที่ 4.6 คือ 1) เป็นผู้พัฒนาซอฟต์แวร์ จำนวน 33.5% 2) เป็นผู้พัฒนาซอฟต์แวร์ที่มีอยู่แล้วให้มีคุณภาพมากขึ้น จำนวน 23.5% 3) เป็นผู้ที่บำรุงรักษาระบบ จำนวน 15.5% 4) เป็นผู้ออกแบบโปรแกรม จำนวน 6.5% 5) เป็นผู้พัฒนาซอฟต์แวร์ตามคำสั่ง จำนวน 5% 6) เป็นผู้พัฒนาซอฟต์แวร์ให้มีผู้เข้ามาใช้งาน จำนวน 4.5% 7) เป็น

ผู้พัฒนาซอฟต์แวร์ให้มีประสิทธิภาพมากขึ้น จำนวน 4.5% 8) เป็นผู้พัฒนาซอฟต์แวร์ที่มีอยู่เดิมให้ตรงตามความต้องการมากขึ้น จำนวน 4.5% และ 9) ต้องทราบทุกอย่างภายในกระบวนการพัฒนาซอฟต์แวร์ จำนวน 2.5%



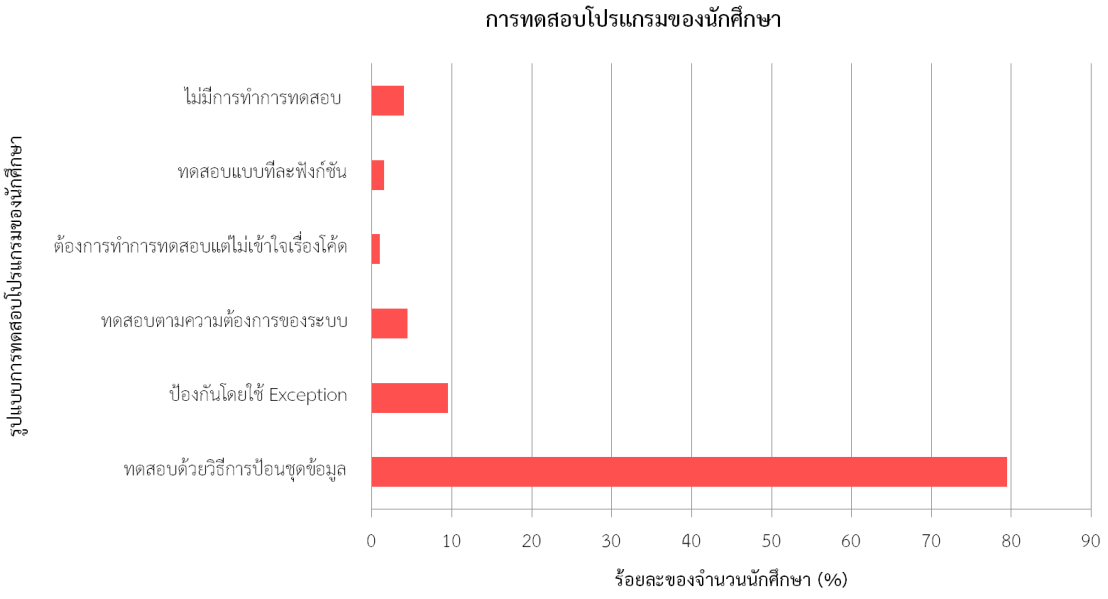
รูปที่ 4.6 กราฟแสดงคำจำกัดความความหมายของนักพัฒนาซอฟต์แวร์

ประเด็นที่ 3 *การพัฒนาโครงการของนักศึกษา* จากคำถามเกี่ยวกับกระบวนการที่นักศึกษาใช้ในการทำโครงการในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ ซึ่งเป็นรายวิชาก่อนทำการทดลองเรื่อง TDD ตั้งแต่เริ่มต้นการทำโครงการจนถึงการนำเสนอโครงการ นักศึกษาใช้รูปแบบในการทดสอบแบ่งออกเป็น 3 รูปแบบ แสดงดังรูปที่ 4.7 คือ 1) การทดสอบระบบหลังการเขียนโค้ดทั้งหมด จำนวน 73% 2) ทำการทดสอบระบบเมื่อพบข้อผิดพลาดขณะรันโปรแกรมเท่านั้น จำนวน 19% และ 3) การทดสอบระบบแบบ Unit test จำนวน 8% จากการจัดกลุ่มคำตอบพบว่ามีจำนวนนักศึกษาเพียง 8% เท่านั้นที่ทำการทดสอบระบบแบบ Unit test ซึ่งเป็นการทดสอบฟังก์ชันหลังจากทำการเขียนโค้ดเสร็จแล้วในแต่ละฟังก์ชัน โดยนักศึกษาที่เหลือใช้วิธีการทดสอบระบบหลังการเขียนโค้ดทั้งหมดเสร็จสิ้น



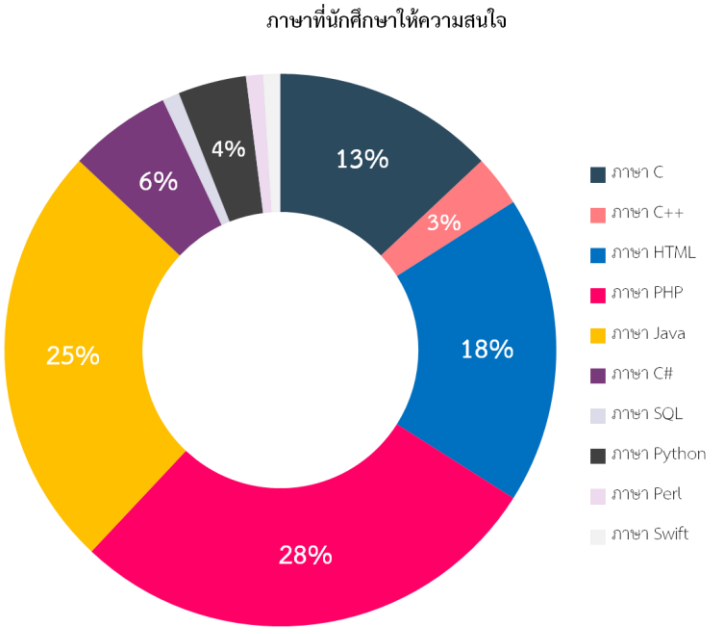
รูปที่ 4.7 กราฟแสดงการเลือกวิธีการทดสอบโปรแกรมของนักศึกษาในการพัฒนาซอฟต์แวร์

ประเด็นที่ 4 *การทดสอบโครงการของนักศึกษา* จากคำถามเกี่ยวกับการทดสอบโครงการในรายวิชาที่ผ่านมาด้านการเขียนโปรแกรม จากการทำ Coding Analysis สามารถแบ่งรูปแบบในการทดสอบได้ 6 รูปแบบ คือ 1) การทดสอบโดยทดสอบด้วยวิธีการป้อนชุดข้อมูลที่คิดว่าจะทำให้ระบบเกิดข้อผิดพลาดเพื่อทดสอบระบบ จำนวน 79.5% 2) ไม่มีการทดสอบโปรแกรมเพราะใช้วิธีป้องกันโดยใช้ Exception ในโปรแกรม จำนวน 9.5% 3) ทดสอบโปรแกรมตามความต้องการของระบบ จำนวน 4.5% 4) ไม่มีการทำการทดสอบ จำนวน 4% 5) ทดสอบแบบทีละฟังก์ชัน จำนวน 1.5% และ 6) ต้องการทำการทดสอบแต่ไม่เข้าใจเรื่องโค้ด จำนวน 1% จากการตอบแบบสอบถามและการสัมภาษณ์แสดงให้เห็นว่านักศึกษาคิดว่าการใช้ Exception เป็นการทดสอบระบบ และนักศึกษาใช้การทดสอบด้วยวิธีการป้อนชุดข้อมูลที่คิดว่าจะทำให้ระบบเกิดข้อผิดพลาดเพื่อทดสอบระบบมากที่สุดทั้งในแบบสอบถามและการสัมภาษณ์ซึ่งแสดงดังรูปที่ 4.8



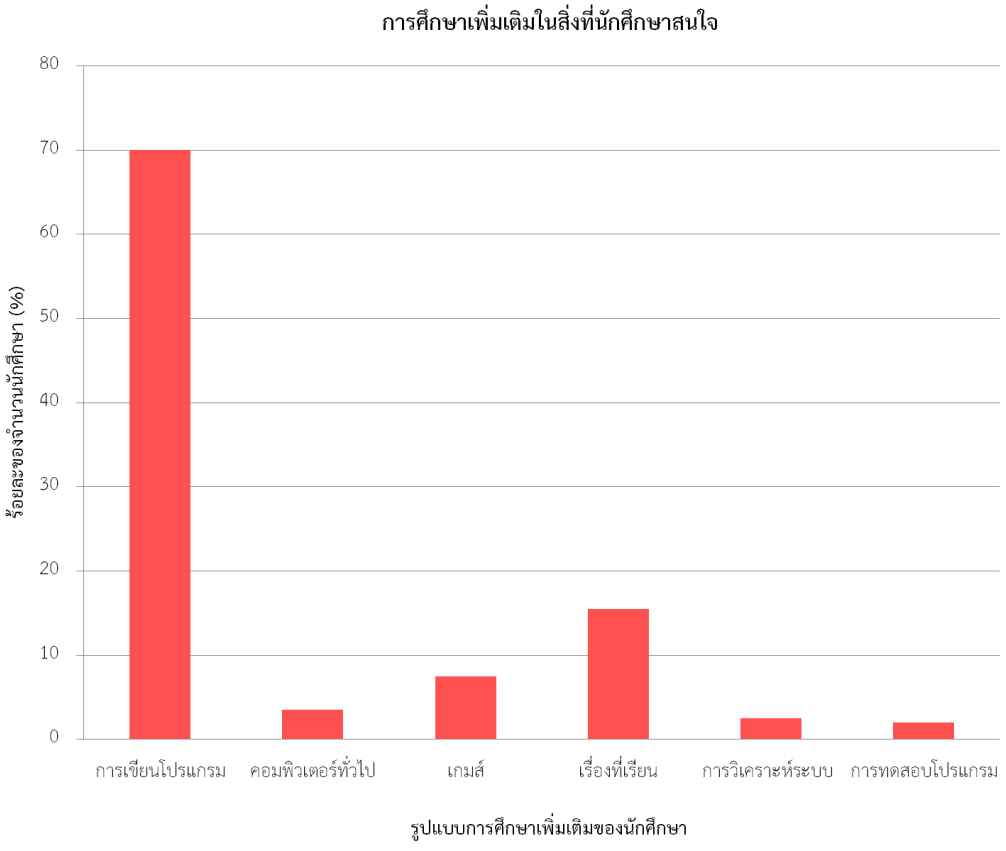
รูปที่ 4.8 กราฟแสดงรูปแบบการทดสอบโปรแกรมของนักศึกษาในการพัฒนาซอฟต์แวร์

ประเด็นที่ 5 *ภาษาในการเขียนโปรแกรมที่นักศึกษาให้ความสนใจ* สำหรับภาษาในการเขียนโปรแกรมที่นักศึกษาให้ความสนใจมีทั้งหมด 10 ภาษา คือ 1) ภาษา PHP จำนวน 28% 2) ภาษา Java จำนวน 25% 3) ภาษา HTML จำนวน 18% 4) ภาษา C จำนวน 13% 5) ภาษา C# จำนวน 6% 6) ภาษา Python จำนวน 4% 7) ภาษา C++ จำนวน 3% 8) ภาษา SQL จำนวน 1% 9) ภาษา Perl จำนวน 1% และ 10) ภาษา Swift จำนวน 1% ซึ่งแสดงดังรูปที่ 4.9 โดยจากภาษาที่นักศึกษาสนใจมีภาษาที่มีการสอนในมหาวิทยาลัยสงขลานครินทร์เพียง 6 ภาษา คือ 1) ภาษา C 2) ภาษา HTML 3) ภาษา PHP 4) ภาษา Java 5) ภาษา C# 6) ภาษา SQL และภาษาที่นักศึกษาสนใจมากที่สุด คือ ภาษา PHP ซึ่งมีนักศึกษาสนใจถึง 28%



รูปที่ 4.9 กราฟแสดงภาษาในการเขียนโปรแกรมที่นักศึกษาให้ความสนใจภายในแบบสอบถามปลายเปิด

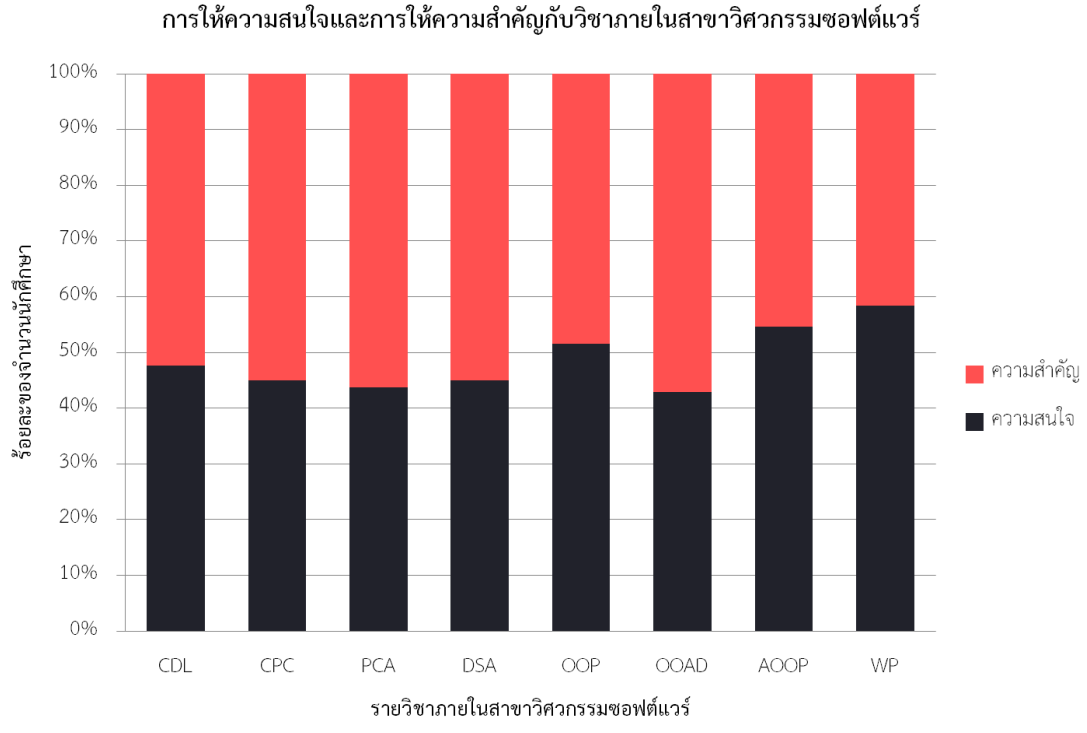
ประเด็นที่ 6 *เรื่องการศึกษาเพิ่มเติมในสิ่งที่นักศึกษาสนใจ* จากการทำ Coding Analysis สามารถจัดกลุ่มสิ่งที่นักศึกษาทำการศึกษาเพิ่มเติมได้ 6 กลุ่ม แสดงดังรูปที่ 4.10 คือ 1) การศึกษาเพิ่มเติมเกี่ยวกับการเขียนโปรแกรม จำนวน 70% 2) การศึกษาเพิ่มเติมเกี่ยวกับเรื่องที่เรียน จำนวน 15.5% 3) การศึกษาเพิ่มเติมเรื่องเกมส์ จำนวน 7.5% 4) การศึกษาเพิ่มเติมเกี่ยวกับคอมพิวเตอร์ จำนวน 3.5% 5) การศึกษาเพิ่มเติมเกี่ยวกับวิเคราะห์ระบบ จำนวน 2.5% และ 6) การทดสอบโปรแกรม จำนวน 2% โดยรูปแบบที่นักศึกษาให้ความสนใจในการทำการศึกษาเพิ่มเติมมากที่สุด คือ การศึกษาเพิ่มเติมเกี่ยวกับการเขียนโปรแกรม ซึ่งนักศึกษาทำการศึกษาเกี่ยวกับเรื่องการเขียนโปรแกรมในภาษา ต่าง ๆ การออกแบบหน้าจอการทำงาน การเขียนโค้ดเชื่อมต่อระหว่างฐานข้อมูลและหน้าจอแสดงผล การทำเว็บไซต์ การจัดการฐานข้อมูล เทคนิคกระบวนการในการเขียนโปรแกรม และการแก้ไขความผิดพลาดของโปรแกรมที่นักศึกษาเขียนและไม่สามารถแก้ไขได้ โดยจาก 6 กลุ่มนี้แสดงให้เห็นว่านักศึกษาไม่ได้ให้ความสนใจในเรื่องกระบวนการในการพัฒนาซอฟต์แวร์รูปแบบใดรูปแบบหนึ่งโดยเฉพาะ



รูปที่ 4.10 กราฟแสดงการให้ความสนใจในการศึกษาเพิ่มเติม

ประเด็นที่ 7 *การให้ความสนใจและการให้ความสำคัญกับวิชาภายในสาขาวิชาวิศวกรรมซอฟต์แวร์* สำหรับวิชาในสาขาวิชาวิศวกรรมซอฟต์แวร์ที่นักศึกษาให้ความสนใจและให้ความสำคัญ ประกอบไปด้วยรายวิชา 8 รายวิชา แสดงดังรูปที่ 4.11 คือ 1) การโปรแกรมบนเว็บ (Web Programming: WP) โดยนักศึกษาให้ความสนใจ จำนวน 21% และให้ความสำคัญ จำนวน 15% 2) การโปรแกรมเชิงอ็อบเจ็กต์ขั้นสูง (Advanced-Object-Oriented Programming: AOOP) โดยนักศึกษาให้ความสนใจ จำนวน 18% และให้ความสำคัญ จำนวน 15% 3) การโปรแกรมเชิงอ็อบเจ็กต์ (Object-Oriented Programming: OOP) โดยนักศึกษาให้ความสนใจ จำนวน 17% และให้ความสำคัญ จำนวน 16% 4) การใช้คอมพิวเตอร์ในชีวิตประจำวัน (Computer in Daily Life: CDL) โดยนักศึกษาให้ความสนใจ จำนวน 10% และให้ความสำคัญ จำนวน 11% 5) การวิเคราะห์และออกแบบเชิงอ็อบเจ็กต์ (Object-Oriented Analysis and Design: OOAD) โดยนักศึกษาให้ความสนใจ จำนวน 9% และให้ความสำคัญ จำนวน 12% 6) คอมพิวเตอร์และหลักโปรแกรม (Computer and

Programming Concept: CPC) โดยนักศึกษาให้ความสนใจ จำนวน 9% และให้ความสำคัญ จำนวน 11% 7) โครงสร้างข้อมูลและอัลกอริทึม (Data Structures and Algorithms: DSA) โดยนักศึกษาให้ความสนใจ จำนวน 9% และให้ความสำคัญ จำนวน 11% และ 8) สถาปัตยกรรมคอมพิวเตอร์และระบบปฏิบัติการ (Principles of Computer Architecture: PCA) โดยนักศึกษาให้ความสนใจ จำนวน 7% และให้ความสำคัญ จำนวน 9% จากการตอบแบบสอบถามของนักศึกษาจำนวน 42 คน แสดงให้เห็นว่าวิชาที่นักศึกษาให้ความสำคัญมากที่สุด คือ วิชาการโปรแกรมเชิงอ็อบเจ็กต์และวิชาที่นักศึกษาให้ความสนใจมากที่สุด คือ วิชาการโปรแกรมบนเว็บ

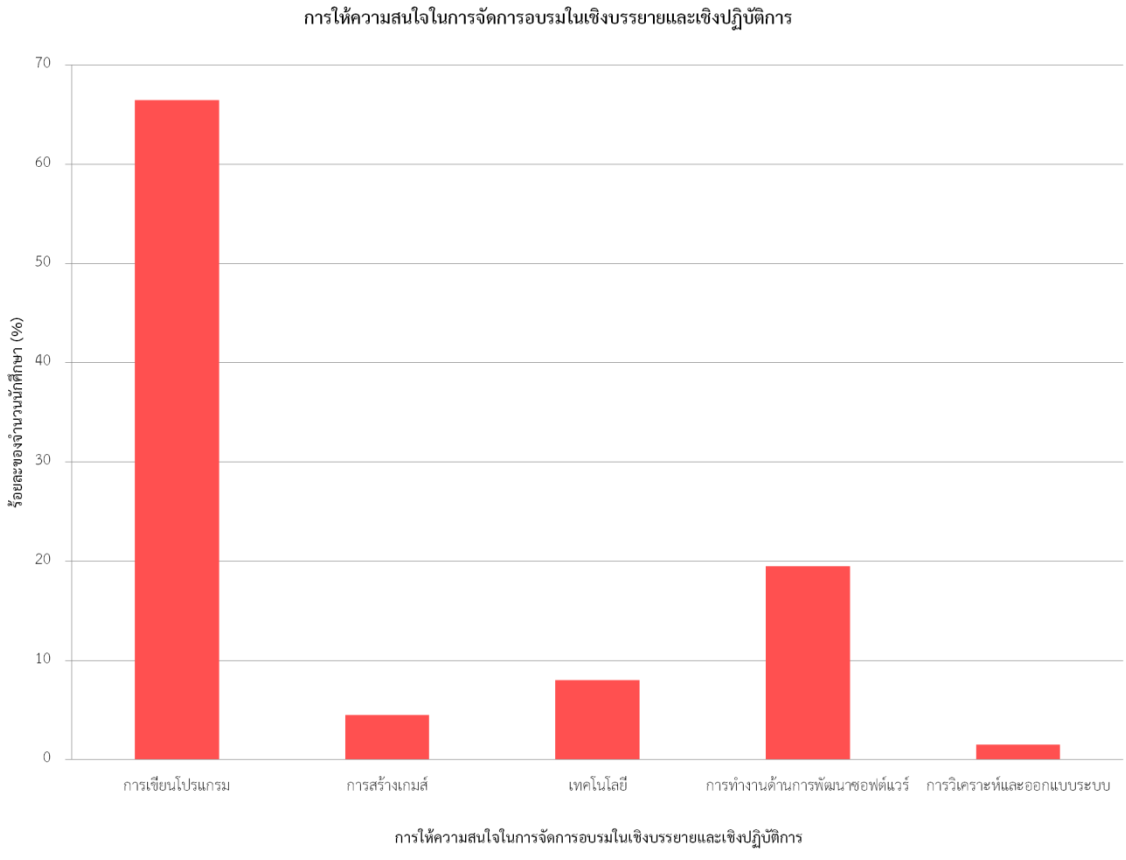


รูปที่ 4.11 กราฟแสดงการให้ความสนใจและการให้ความสำคัญกับวิชาภายในสาขาวิศวกรรมซอฟต์แวร์ภายในแบบสอบถามปลายเปิด

ประเด็นที่ 8 *การให้ความสนใจในการจัดการอบรมในเชิงบรรยายและเชิงปฏิบัติการ*

จากการทำการ Coding Analysis สามารถจัดกลุ่มความสนใจของนักศึกษาที่ให้ความสนใจในการอบรมได้ทั้งหมด 5 กลุ่ม คือ 1) การอบรมเกี่ยวกับการเขียนโปรแกรมมีนักศึกษาให้ความสนใจ จำนวน 66.5% 2) การอบรมเกี่ยวกับการทำงานด้านการพัฒนาซอฟต์แวร์มีนักศึกษาให้ความสนใจ จำนวน 19.5% 3) การ

อบรมเกี่ยวกับเทคโนโลยีมีนักศึกษาให้ความสนใจ จำนวน 8% 4) การอบรมเกี่ยวกับการสร้างเกมส์มีนักศึกษาให้ความสนใจ จำนวน 4.5% และ 5) การอบรมเกี่ยวกับการวิเคราะห์และออกแบบระบบมีนักศึกษาให้ความสนใจ จำนวน 1.5% ซึ่งรูปแบบที่ได้รับความสนใจจากนักศึกษามากที่สุด คือ การอบรมเกี่ยวกับการเขียนโปรแกรมโดยมีนักศึกษาสนใจถึง 66.5% แสดงดังรูปที่ 4.12



รูปที่ 4.12 กราฟแสดงการให้ความสนใจในการจัดอบรมในเชิงบรรยายและเชิงปฏิบัติการ

ประเด็นที่ 9 *ผลต่อความสนใจของนักศึกษาในการใช้ TDD ในการพัฒนาซอฟต์แวร์*

จากการวิเคราะห์พบว่า TDD มีผลต่อความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ทั้งหมด 86% โดยนักศึกษาที่พบว่า TDD มีผลต่อความสนใจในการพัฒนาซอฟต์แวร์นั้น ได้กล่าวถึงประโยชน์ของการนำ TDD ไปใช้ในการเรียนการสอนโดยแบ่งออกเป็นทั้งหมด 6 ด้าน คือ 1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม 2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม 3) ด้านคุณภาพของโปรแกรม 4) กระบวนการ

การใช้ TDD 5) ด้านความรู้สึกของผู้ใช้ TDD และ 6) ด้านเวลาที่ใช้ในการทำ TDD ในการเขียนโปรแกรม โดยประโยชน์ทั้ง 6 ด้านเกิดจากการวิเคราะห์ข้อมูลจากการทำ Coding Analysis แล้วนำคำตอบที่ได้มาจัดกลุ่มข้อมูล โดยประโยชน์ของการนำ TDD ไปใช้ในการเรียนการสอนจากคำตอบของนักศึกษามีรายละเอียดดังนี้

(1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม

(1.1) เป็นการตรวจสอบหาข้อผิดพลาดของโปรแกรม

(1.2) ทำให้พบข้อผิดพลาดในโปรแกรมลดน้อยลง

(1.3) ลดปัญหาในการเขียนโปรแกรม

(1.4) หาจุดบกพร่องของโปรแกรมได้ตั้งแต่เริ่มทำการเขียนโปรแกรม

(2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม

(2.1) คอยปรับปรุงโปรแกรมให้ตรงกับความต้องการของผู้ใช้งานซอฟต์แวร์

(2.2) ทำให้ทราบว่าโปรแกรมที่เขียนขึ้นเกิดปัญหาจากส่วนใด

(2.3) แก้ไขข้อผิดพลาดในการเขียนโปรแกรมได้รวดเร็วขึ้นจากเดิม

(2.4) ทำให้ง่ายต่อการแก้ไขข้อผิดพลาดที่เกิดขึ้น

(2.5) สามารถแก้ไขข้อผิดพลาดได้ดีขึ้น

(3) ด้านคุณภาพของโปรแกรม

(3.1) ทำให้โปรแกรมที่พัฒนามีคุณภาพ

(3.2) ช่วยทำให้โค้ดที่เขียนมีความถูกต้อง

(3.3) ทำให้โปรแกรมมีความซับซ้อนน้อยลง

(3.4) ทำให้เข้าใจการเขียนโค้ดมากยิ่งขึ้น

(3.5) ทำให้ทราบวิธีการเขียนโค้ดที่มีบรรทัดการทำงานน้อย

(3.6) ผ่านการทำการทดสอบแล้วจึงทำการปรับปรุงโค้ดให้ได้มาตรฐานมากขึ้น

(3.7) เป็นเทคนิคแบบใหม่ที่ทำให้การพัฒนาโปรแกรมดีขึ้น

(3.8) ทำให้โปรแกรมมีประสิทธิภาพเพิ่มมากขึ้น

(3.9) ช่วยปรับปรุงการเขียนโปรแกรมได้ดีขึ้น

(4) กระบวนการการใช้ TDD

(4.1) เป็นการพัฒนาโปรแกรมแบบเป็นขั้นตอนวนซ้ำไปเรื่อย ๆ

(4.2) ทำให้ทำงานอย่างมีลำดับขั้น

(4.3) มีความรอบคอบในการเขียนโค้ดเพิ่มมากขึ้น

(4.4) ทำการเขียนโค้ดง่ายขึ้น

(4.5) เป็นการฝึกการเขียนโค้ดว่าต้องเขียนโค้ดให้ออกมาให้ดี

(4.6) ช่วยทำให้มีความรอบคอบในการเขียนโปรแกรม

(4.7) มีขั้นตอนในการเขียนโปรแกรมได้ละเอียด

(4.8) ไม่ต้องทำการเขียนโค้ดแบบสุ่มแล้วไปทำการทดสอบภายหลังทำให้เขียนโปรแกรมเข้าใจง่ายกว่าแบบที่เคยเรียนมา

(4.9) ไม่ทราบว่าการทดสอบโปรแกรมจริง ๆ เป็นแบบใด

(4.10) ทำให้มีแนวทางในการเขียนโปรแกรม

(4.11) มีการวางเป้าหมายในการเขียนโปรแกรมอย่างชัดเจน

(4.12) เป็นการแก้ปัญหาจากจุดเล็ก ๆ ไปเป็นโปรแกรม

(4.13) ทราบข้อผิดพลาดก่อนการเขียนโค้ด

(5) ด้านความรู้สึกของผู้ใช้ TDD

(5.1) มีความมั่นใจในการเขียนโค้ดมากขึ้น

(5.2) เป็นวิธีการที่คิดแล้วทำให้รู้สึกสนุกทำการลองไปเรื่อย ๆ

(5.3) ทำให้ทราบความต้องการของนักพัฒนา

(5.4) เพิ่มความสนใจในการเรียนด้านการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น

(5.5) สามารถสื่อสารเรื่องเกี่ยวกับการเขียนโค้ดกับคนภายในกลุ่มได้เข้าใจกันมากขึ้น

(5.6) รู้สึกอยากเขียนโปรแกรมเพิ่มมากขึ้น

(6) ด้านเวลาที่ใช้ในการทำ TDD ในการเขียนโปรแกรม

(6.1) เมื่อมีการทำการทดสอบตั้งแต่เริ่มต้นทำให้เวลาเขียนโปรแกรมจริงเร็วขึ้น

(6.2) ใช้ระยะเวลาในการโค้ดน้อยลง

ประเด็นที่ 10 *ทัศนคติของนักศึกษาในการนำ TDD ไปใช้ในการเขียนโปรแกรมในห้องปฏิบัติการ* ผู้วิจัยแบ่งกลุ่มของผลที่เกิดต่อนักศึกษาได้เป็น 2 กลุ่ม คือ 1) ความรู้สึกด้านบวกของการใช้ TDD ในการเขียนโปรแกรมภายในห้องปฏิบัติการซึ่งนักศึกษาได้ให้ความคิดเห็นถึงความรู้สึกด้านบวก จำนวน 82% และ 2) ความรู้สึกด้านลบของการใช้ TDD ในการเขียนโปรแกรมภายในห้องปฏิบัติการซึ่งนักศึกษาได้ให้ความคิดเห็นถึงความรู้สึกด้านลบ จำนวน 18% ซึ่งในความรู้สึกด้านลบและความรู้สึกด้านบวกของการใช้ TDD ในการเขียนโปรแกรมภายในห้องปฏิบัติการนั้น นักศึกษาให้เหตุผลที่สามารถแบ่งออกเป็นทั้งหมด 6 ด้าน คือ 1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม 2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม 3) ด้านคุณภาพของโปรแกรม 4) กระบวนการการใช้ TDD 5) ด้านความรู้สึกของผู้ใช้ TDD และ 6) ด้านเวลาที่ใช้ในการทำ TDD ในการเขียนโปรแกรม ซึ่งทั้ง 6 ด้านเกิดจากการวิเคราะห์ข้อมูลจากการทำ Coding Analysis แล้วนำคำตอบที่ได้มาจัดกลุ่มข้อมูลโดยความรู้สึกของนักศึกษาทั้งด้านบวกและด้านลบ จากคำตอบของนักศึกษามีรายละเอียดดังนี้

(1) ความรู้สึกด้านบวกของการใช้ TDD ในการเขียนโปรแกรมภายในห้องปฏิบัติการ

(1.1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม

(1.1.1) ทำให้ทราบตำแหน่งและสาเหตุที่ทำให้เกิดข้อผิดพลาด

(1.1.2) หาจุดบกพร่องของโปรแกรมได้ง่ายขึ้น

(1.1.3) ทำให้โปรแกรมสามารถตรวจสอบข้อผิดพลาดได้ง่ายขึ้น

(1.1.4) สามารถหาข้อผิดพลาดได้เร็วขึ้น

(1.1.5) ช่วยให้ทราบข้อผิดพลาดที่อาจเกิดขึ้นได้ในโปรแกรมก่อนทำการเขียนโค้ดจริงในโปรแกรม

(1.2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม

(1.2.1) ทำให้การทดสอบความผิดพลาดทำให้ได้ง่าย

(1.2.3) ทำให้การป้องกันข้อผิดพลาด (Debug) ง่ายขึ้น

(1.2.4) ช่วยแก้ปัญหาเมื่อพบข้อผิดพลาดแล้วไม่มีแนวทางแก้ไข

(1.2.5) เมื่อเกิดข้อผิดพลาดในโปรแกรมสามารถทำการแก้ไขได้เลย

(1.3) ด้านคุณภาพของโปรแกรม

(1.3.1) ทำให้โค้ดที่เขียนขึ้นไม่มีความซับซ้อน

(1.3.2) ทำให้เกิดการคิดที่จะพัฒนาซอฟต์แวร์ให้มีโค้ดที่สั้นลงและไม่มีข้อผิดพลาดทำให้โปรแกรมที่ได้มีประสิทธิภาพมากขึ้น

(1.3.3) ช่วยลดข้อผิดพลาดที่อาจจะเปิดขึ้นในโปรแกรม

(1.3.4) โค้ดที่ทำการเขียนขึ้นมีความกระชับมากขึ้น

(1.3.5) มีการทำการปรับปรุงโค้ดให้สามารถเข้าใจได้ง่ายขึ้น (Refactoring)

(1.3.6) สามารถทำการเขียนโค้ดได้ตรงตามความต้องการมากขึ้น

(1.4) กระบวนการการใช้ TDD

(1.4.1) ช่วยทำให้มีความรอบคอบในการเขียนฟังก์ชันการทำงาน

(1.4.2) เป็นการเริ่มกระบวนการเขียนโปรแกรมจากยากไปง่าย

(1.4.3) ทำให้โค้ดที่เขียนขึ้นมีจำนวนน้อยลง

(1.4.4) รู้จักวิธีการแก้ปัญหามากขึ้น

(1.4.5) การทำ TDD สามารถแบ่งออกเป็นส่วนการทำงานได้

(1.4.6) เป็นการทำการเขียนโปรแกรมไปทีละส่วน

(1.4.7) ช่วยในเรื่องการเรียงลำดับการทำงานการเขียนโค้ดมากขึ้น

(1.5) ทำให้มีความเข้าใจโค้ดที่เขียนเพิ่มมากขึ้น

(1.5.1) ทำให้การเขียนโค้ดง่ายขึ้น

(1.5.2) เขียนโปรแกรมได้ถนัดขึ้น

(1.5.3) มีความตั้งใจในการเขียนโค้ดมากขึ้น

(1.5.4) มีความต้องการศึกษาเรื่อง TDD ต่อไป

(1.5.5) ทำให้อยากเขียนโปรแกรมและนำไปใช้ฝึกเขียนเองหลังจากจบการทดลอง

(1.5.6) ทำให้การเขียนโค้ดสนุกขึ้น

(1.6) ด้านเวลาที่ใช้ในการทำ TDD ในการเขียนโปรแกรม

(1.6.1) ประหยัดเวลาในการเขียนโปรแกรม

(1.6.2) ทำให้ใช้เวลาในการเขียนโปรแกรมเร็วกว่าแบบเดิม

(1.6.3) ทำให้เพื่อนเข้าใจโค้ดที่เราเป็นคนเขียนขึ้นด้วย

(1.6.4) วิธีเดิมใช้เวลาค่อนข้างเยอะเมื่อเกิดปัญหาที่ต้องกลับไปแก้ไข

(2) ความรู้สึกด้านลบของการใช้ TDD ในการเขียนโปรแกรมภายในห้องปฏิบัติการ

(2.1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม

(2.1.1) ทำให้การเขียนโปรแกรมมีความยากขึ้น

(2.1.2) ยังไม่เข้าใจหลักการทำงานของ TDD

(2.1.3) การใช้วิธีการแบบ TDD มีการทำงานหลายขั้นตอน

(2.2) ด้านความรู้สึกของผู้ใช้ TDD

(2.2.1) รู้สึกว่าการใช้ TDD มีความยากเพราะมีประสบการณ์น้อย

(2.2.2) การใช้ TDD ใช้เวลาในการเขียนโปรแกรมเพิ่มขึ้น

ประเด็นที่ 11 *การเลือกวิธีการในการพัฒนาซอฟต์แวร์ระหว่างวิธีการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และการพัฒนาซอฟต์แวร์โดยใช้ TDD* ผู้วิจัยจัดกลุ่มของคำตอบด้วยวิธีการ Coding Analysis ได้ทั้งหมด 4 กลุ่ม คือ 1) นักศึกษาเลือกใช้วิธีการ TDD ในการทำแบบฝึกหัด จำนวน 73% 2) นักศึกษาเลือกใช้วิธีการจำลองน้ำตกในการทำแบบฝึกหัด จำนวน 15% 3) นักศึกษาเลือกใช้วิธีการเขียนโปรแกรมทีละฟังก์ชันแล้วทดสอบในการทำแบบฝึกหัด จำนวน 11% และ 4) นักศึกษาคิดว่าวิธีการทั้ง 2 วิธี พบข้อผิดพลาดไม่ต่างกัน จำนวน 1% แสดงดังรูปที่ 4.13 โดยวิธีการที่นักศึกษาเลือกมากที่สุด คือ วิธีการ TDD โดยนักศึกษาให้เหตุผลที่เลือกวิธีการ TDD ไว้ดังนี้

(1) วิธีการ TDD เป็นวิธีการที่เน้นการทำกรณีทดสอบ

(2) ทำให้อ็อบเจ็กต์มีความสมบูรณ์มากกว่า

(3) ต้องทำการเขียนโปรแกรมให้ผ่านไปทีละขั้น

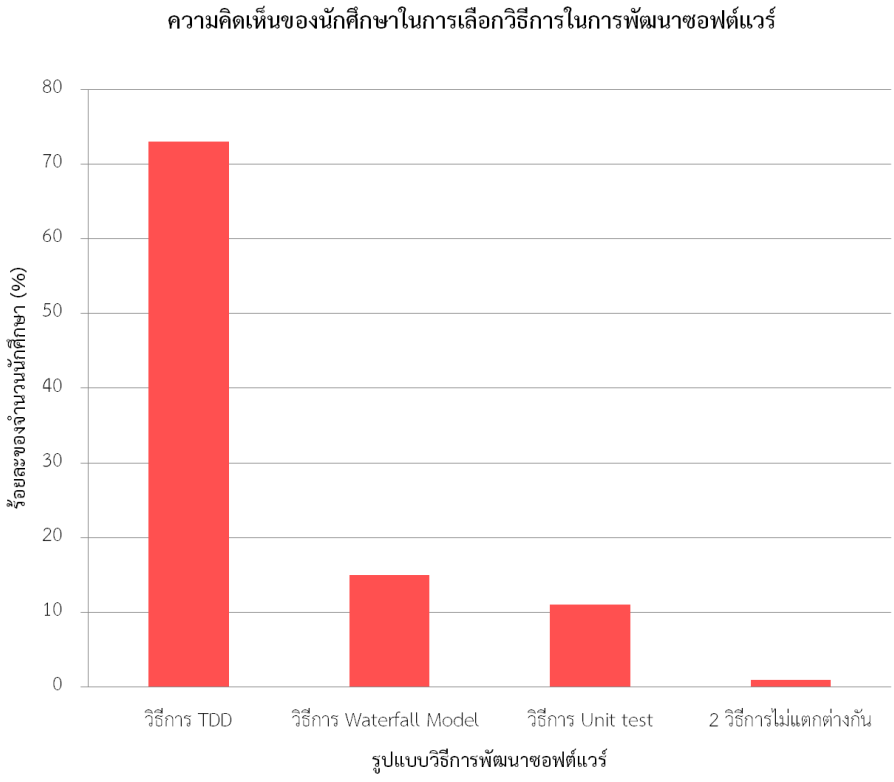
(4) ทำให้สามารถตรวจสอบกระบวนการในโปรแกรมได้ง่าย

(5) มีการทดสอบก่อนเริ่มทำการเขียนโปรแกรมจริง

(6) สามารถหาข้อผิดพลาดได้ง่าย

(7) มีการทดสอบตลอดเวลาที่ทำการเขียนโปรแกรม

(8) ไม่เสียเวลาในการแก้ปัญหาใหม่ตั้งแต่ต้น



รูปที่ 4.13 กราฟแสดงการเลือกวิธีการของนักศึกษาในการตรวจสอบข้อผิดพลาด

สำหรับวิธีการที่นักศึกษาเลือกมากที่สุด คือ วิธีการ TDD ซึ่งนักศึกษาที่เลือกวิธีการ TDD ได้ให้เหตุผลในการเลือกวิธีการ TDD ไว้ 2 รูปแบบ คือ กล่าวถึงข้อดีของการใช้ TDD และข้อจำกัดจากประสบการณ์ของนักศึกษาที่เคยใช้วิธีการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก สามารถแบ่งออกเป็นทั้งหมด 6 ด้าน คือ 1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม 2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม 3) ด้านคุณภาพของโปรแกรม 4) กระบวนการการใช้ TDD 5) ด้านความรู้สึกของผู้ใช้ TDD และ 6) ด้านเวลาที่ใช้ในการทำ TDD ในการเขียนโปรแกรม ซึ่งทั้ง 6 ด้านเกิดจากการวิเคราะห์ข้อมูลจากการทำ Coding Analysis แล้วนำคำตอบที่ได้มาจัดกลุ่มข้อมูล โดยมีรายละเอียดดังนี้

(1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม

(1.1) เป็นการเขียนการทดสอบก่อนการเขียนโค้ดจริง จึงทำให้มีโอกาสพบข้อผิดพลาดน้อยกว่า

(1.2) เมื่อเกิดข้อผิดพลาดทำให้ทราบว่าเกิดข้อผิดพลาดจากส่วนไหน

(1.3) สามารถหาข้อผิดพลาดได้ดีกว่า

(2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม

(2.1) สามารถทำให้แก้ปัญหาได้ตรงจุด

(2.2) เมื่อพบข้อผิดพลาดสามารถทำการแก้ไขได้เลย

(3) ด้านคุณภาพของโปรแกรม

(3.1) สามารถนำโค้ดที่ทำงานได้ตามต้องการมารวมกันให้โค้ดทำงานที่ถูกต้อง

(3.2) การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก พบปัญหาเยอะกว่าการพัฒนาซอฟต์แวร์แบบ TDD

(3.3) การพัฒนาซอฟต์แวร์แบบจำลองน้ำตกเมื่อแก้ข้อผิดพลาดจุดหนึ่งอาจจะไปกระทบกับโค้ดส่วนอื่นด้วย

(4) กระบวนการการใช้ TDD

(4.1) ก่อนการเขียนโปรแกรมมีการคิดถึงการทำงานของระบบ

(4.2) เมื่อทำการใช้ TDD สามารถทำการตรวจสอบได้หลายครั้ง

(4.3) มีการทำกรณีทดสอบก่อนการเขียนโปรแกรมจริง

(4.4) การใช้วิธีการ TDD ต้องเข้าใจผลลัพธ์ก่อนแล้วทำการหาวิธีเขียนโค้ดให้ได้ตามที่ตั้งไว้

(4.5) การทำ TDD มีการวางเป้าหมายไว้ก่อนลงมือทำการเขียนโค้ด

(4.6) การใช้ TDD ทำให้ได้ผลลัพธ์ตามวัตถุประสงค์มากกว่าการพิมพ์ไปเรื่อย ๆ แบบการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก

(4.7) คิดว่าการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกเป็นการเขียนแบบสุ่ม

(4.8) การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก มีการทำงานเป็นขั้นตอนแต่อาจทำให้นักศึกษาลืมรายละเอียดของโปรแกรม

(4.9)การพัฒนาซอฟต์แวร์แบบจำลองน้ำตกต้องเขียนโค้ดให้เสร็จเรียบร้อยก่อนถึงจะเริ่มทดสอบ

(5) ด้านความรู้สึกของผู้ใช้ TDD

(5.1) ถ้ามีการสอนแบบ TDD น่าจะทำให้การเขียนโปรแกรมดีขึ้น

(5.2) มีความเคยชินกับการทำการพัฒนาซอฟต์แวร์แบบ จำลองน้ำตกมากกว่า

(6) ด้านเวลาที่ใช้ในการทำ TDD ในการเขียนโปรแกรม

(6.1) การใช้วิธีการพัฒนาซอฟต์แวร์แบบ TDD มีโอกาสพบข้อผิดพลาดได้เร็วกว่าการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกก่อนเริ่มเขียนโปรแกรม

(6.2) ใช้เวลาในการค้นหาข้อผิดพลาดน้อยกว่าการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก

(6.3) การใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก มีการพัฒนาซอฟต์แวร์ที่ใช้เวลามากกว่า

ประเด็นที่ 12 *การเลือกวิธีการในการทดสอบในอนาคตของนักศึกษาในการทดสอบภายในโครงการในรายวิชา* ผู้วิจัยจัดกลุ่มของคำตอบได้ทั้งหมด 4 กลุ่ม คือ 1) นักศึกษาเลือกวิธีแบบ TDD จำนวน 78% 2) เลือกวิธีทดสอบแบบ Test-Last ในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก จำนวน 18.5% 3) นักศึกษาเลือกวิธีที่เหมาะสมสำหรับโครงการ จำนวน 2% และ 4) นักศึกษาเลือกนำไปใช้ทั้ง 2 วิธี จำนวน 1.5% โดยวิธีการทดสอบในอนาคตที่นักศึกษาเลือกใช้ในการทำโครงการในรายวิชามากที่สุด คือ วิธีแบบ TDD ซึ่งมีจำนวนนักศึกษาเลือกมากถึง 78%

ประเด็นที่ 13 *ประโยชน์ของการนำ TDD ไปใช้ในการพัฒนาซอฟต์แวร์ในมุมมองของนักศึกษา* ผู้วิจัยพบว่ามีนักศึกษา จำนวน 88.5% ที่เห็นว่า TDD มีประโยชน์ในการนำไปใช้ในการพัฒนาซอฟต์แวร์

ประเด็นที่ 14 *ข้อจำกัดของการนำ TDD ไปใช้ในการพัฒนาซอฟต์แวร์ในมุมมองของนักศึกษา* ผู้วิจัยได้รวบรวมข้อจำกัดที่นักศึกษาบางส่วนได้กล่าวถึงการนำ TDD ไปใช้ภายในห้องปฏิบัติขณะทำการทดลองและข้อจำกัดในการนำ TDD ไปใช้ต่อหลังจากจบการทดลองโดยมีข้อจำกัดทั้งหมดดังนี้

(1) ถ้าคนที่เป็นคนเขียนกรณีทดสอบและเป็นคนเขียนโค้ดด้วยก็จะต้องทราบอยู่ว่าจะต้องเขียนโปรแกรมอย่างไรให้ผ่านกรณีทดสอบ

(2) มีความเคยชินกับการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก

(3) TDD เป็นเรื่องใหม่นักศึกษาจึงมองว่าเป็นความยุ่งยาก

(4) การใช้ TDD มีหลายขั้นตอนจึงอาจต้องทำการศึกษาเพิ่มเติม แต่สนใจการเขียนโปรแกรมเพิ่มมากขึ้น

(5) นักศึกษาทราบเพียงรูปแบบพื้นฐานของการใช้ TDD จากการทดลอง

(6) เป็นความรู้ใหม่ ไม่เคยเรียนวิธีการแบบนี้มาก่อน

(7) เวลาเรียน/เวลาในการทดลองค่อนข้างจำกัด

(8) ใช้เครื่องมือในการทดลอง (Eclipse) ไม่เป็น

(9) ยังไม่ได้มีการนำเอา TDD ไปใช้เท่าที่ควร

(10) หากได้เรียน TDD มากกว่านี้นักศึกษาคิดว่าน่าจะสนใจเพิ่มขึ้นมาก เพราะช่วยสร้างความมั่นใจในการเขียนโค้ด

(11) ไม่ได้นำ TDD ไปใช้ต่อในรายวิชาอื่น

(12) มีความถนัดในการใช้การพัฒนาซอฟต์แวร์รูปแบบจำลองน้ำตกมากกว่า คือ เมื่อคิดอะไรได้ก็เขียนโปรแกรมลงไปเลย ถ้าเจอข้อผิดพลาดขณะที่โปรแกรมทำงานเมื่อแสดงผลลัพธ์แล้วจึงจะดำเนินการแก้ไข

(13) การใช้ TDD ต้องทำการคิดผลลัพธ์ตั้งแต่เริ่มต้นซึ่งนักศึกษาไม่มีความถนัดในการใช้วิธีดังกล่าว

(14) นักศึกษาคิดว่า TDD เป็นแค่กระบวนการพัฒนาซอฟต์แวร์แบบใหม่ ไม่ใช่เทคโนโลยีใหม่จึงไม่ค่อยสนใจ

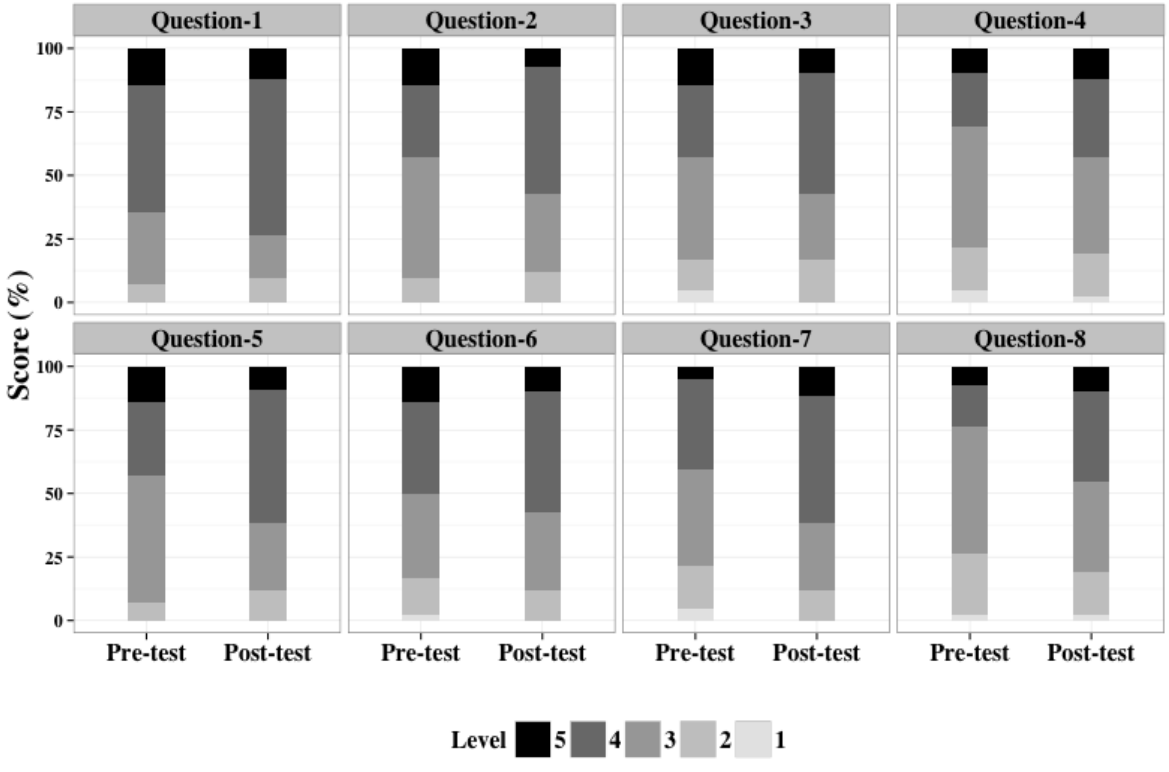
(15) ใช้วิธีการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก ซึ่งเป็นวิธีการแบบเดิมที่ใช้อยู่สามารถเข้าใจได้ดีกว่า

(16) หากต้องการใช้วิธีการ TDD ต้องทำการเรียงลำดับในการเขียนโปรแกรมของตัวเองใหม่

4.2.2 ผลการดำเนินการวิจัยจากแบบสอบถามแบบ Likert Scale

ผลการดำเนินการวิจัยจากแบบสอบถามแบบ Likert Scale ที่สร้างขึ้นจากตัวบ่งชี้ด้านความสนใจในงานวิจัยนั้น มีทั้งหมด 8 ข้อ โดยจัดทำขึ้นเพื่อวัดผลด้านความสนใจในการพัฒนาซอฟต์แวร์ของนักศึกษาก่อนและหลังการทำการสอน TDD ซึ่งได้ทำการสรุปคะแนนเฉลี่ยของคำถามในแต่ละข้อโดย

ทำการเปรียบเทียบจากคะแนนเฉลี่ยก่อนและหลังการทำการทดลอง โดยใช้เครื่องมือที่ช่วยในการวิเคราะห์ข้อมูล คือ โปรแกรม R โดยมีรายละเอียดดังรูปที่ 4.14



รูปที่ 4.14 แสดงคะแนนเฉลี่ยของคำถามจากแบบสอบถามแบบ Likert Scale

จากรูปที่ 4.14 แสดงให้เห็นถึงคะแนนเฉลี่ยในการตอบคำถามของแต่ละข้อจากนักศึกษาจำนวน 42 คน ในแต่ละข้อว่ามีคะแนนเฉลี่ยของแต่ระดับเป็นอย่างไรและเป็นการเปรียบเทียบข้อมูลการตอบแบบสอบถามของนักศึกษาก่อนและหลังการทดลอง โดยมีรายละเอียดของคำถามแต่ละข้อดังนี้

คำถามข้อ 1 *นักศึกษาให้ความสนใจในการเรียนในรายวิชาด้านการพัฒนาซอฟต์แวร์* ผลการตอบแบบสอบถามชี้ให้เห็นว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่ให้ความสนใจในการเรียนในรายวิชาด้านการพัฒนาซอฟต์แวร์ ในระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 4 สูงขึ้น 11.9%

คำถามข้อ 2 *นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากแบบฝึกหัด (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)* ผู้วิจัยพบว่าหลังการเรียน TDD มี

จำนวนนักศึกษาที่ให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากแบบฝึกหัดในระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 4 สูงขึ้น 21.30%

คำถามข้อ 3 *นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากการศึกษาหาข้อมูลด้วยตนเอง (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)* จากคำตอบในแบบสอบถามพบว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่ให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากการศึกษาหาข้อมูลด้วยตนเอง ในระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 4 สูงขึ้น 19.05% และมีค่าเฉลี่ยของคะแนนในระดับที่ 1 (ให้ความสนใจน้อยที่สุด) ซึ่งมีค่าเฉลี่ยของคะแนนในระดับที่ 1 ลดลง -4.76%

คำถามข้อ 4 *นักศึกษาให้ความสนใจในการศึกษาเกี่ยวกับวิศวกรรมซอฟต์แวร์เพิ่มเติมด้วยตนเอง (การศึกษาเพิ่มเติมด้านวิศวกรรมซอฟต์แวร์ เช่น การศึกษาเพิ่มเติมถึงกระบวนการพัฒนาซอฟต์แวร์รูปแบบใหม่ ภาษาในการเขียนโปรแกรม เป็นต้น)* ผู้วิจัยพบว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่ให้ความสนใจในการศึกษาเกี่ยวกับวิศวกรรมซอฟต์แวร์เพิ่มเติมด้วยตนเอง ในระดับที่ 5 (ให้ความสนใจมากที่สุด) และระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 5 และระดับที่ 4 สูงขึ้น 2.38% และ 9.52% ตามลำดับ และมีค่าเฉลี่ยของคะแนนในระดับที่ 1 (ให้ความสนใจน้อยมาก) มีค่าเฉลี่ยของคะแนนในระดับ 1 ลดลง -2.38%

คำถามข้อ 5 *นักศึกษาให้ความสนใจในการใช้เวลาในการทำแบบฝึกหัด* จากผลการตอบแบบสอบถามพบว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่ให้ความสนใจในการใช้เวลาในการทำแบบฝึกหัดในระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 4 สูงขึ้น 23.81%

คำถามข้อ 6 *นักศึกษามีความสนใจในการดัดแปลงการแสดงผลจากแบบฝึกหัดที่ได้รับมอบหมาย* ผู้วิจัยพบว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่มีความสนใจในการดัดแปลงการแสดงผลจากแบบฝึกหัดที่ได้รับมอบหมาย ในระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 4 สูงขึ้น 11.91% และมีค่าเฉลี่ยของคะแนนในระดับที่ 1 (ให้ความสนใจน้อยมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 1 ลดลง -2.38%

คำถามข้อ 7 *นักศึกษาให้ความสนใจในการใช้เวลาในการศึกษาด้วยตนเอง* ผลที่ได้พบว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่ให้ความสนใจในการใช้เวลาในการศึกษาด้วยตนเอง ในระดับที่ 5 (ให้ความสนใจมากที่สุด) และระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 5

และระดับที่ 4 สูงขึ้น 7.14% และ 14.29% ตามลำดับ และมีค่าเฉลี่ยของคะแนนในระดับที่ 1 (ให้ความสนใจน้อยมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 1 ลดลง -4.76%

คำถามข้อ 8 *นักศึกษามีให้ความสนใจในการมีส่วนร่วมในการถาม - ตอบภายในห้องเรียน* จากรูปที่ 4.14 พบว่าหลังการเรียน TDD มีจำนวนนักศึกษาที่ให้ความสนใจในการมีส่วนร่วมในการถาม-ตอบภายใน ห้องเรียน ในระดับที่ 5 (ให้ความสนใจมากที่สุด) และระดับที่ 4 (ให้ความสนใจมาก) มีค่าเฉลี่ยของคะแนนในระดับที่ 5 และระดับที่ 4 สูงขึ้น 2.38% และ 19.04% ตามลำดับ

นอกจากผู้วิจัยจะทำการวิเคราะห์ผลรายข้อในแบบสอบถามแบบ Likert Scale แล้ว ผู้วิจัยยังได้ทำการวิเคราะห์ผลทางสถิติด้วยวิธี Paired Student's T-test โดยใช้โปรแกรมที่ช่วยในการวิเคราะห์ข้อมูล คือ โปรแกรม R ในการวิเคราะห์ข้อมูลเพื่อตรวจสอบสมมติฐานของงานวิจัย คือ

สมมติฐานหลัก H_0: นักศึกษาที่ได้รับการเรียนการสอนโดยใช้ TDD มีความสนใจในการพัฒนาซอฟต์แวร์ไม่ต่างกันกับการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก

สมมติฐานรอง H_1: นักศึกษาที่ได้รับการเรียนการสอนโดยใช้ TDD มีความสนใจในการพัฒนาซอฟต์แวร์ต่างกับการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก

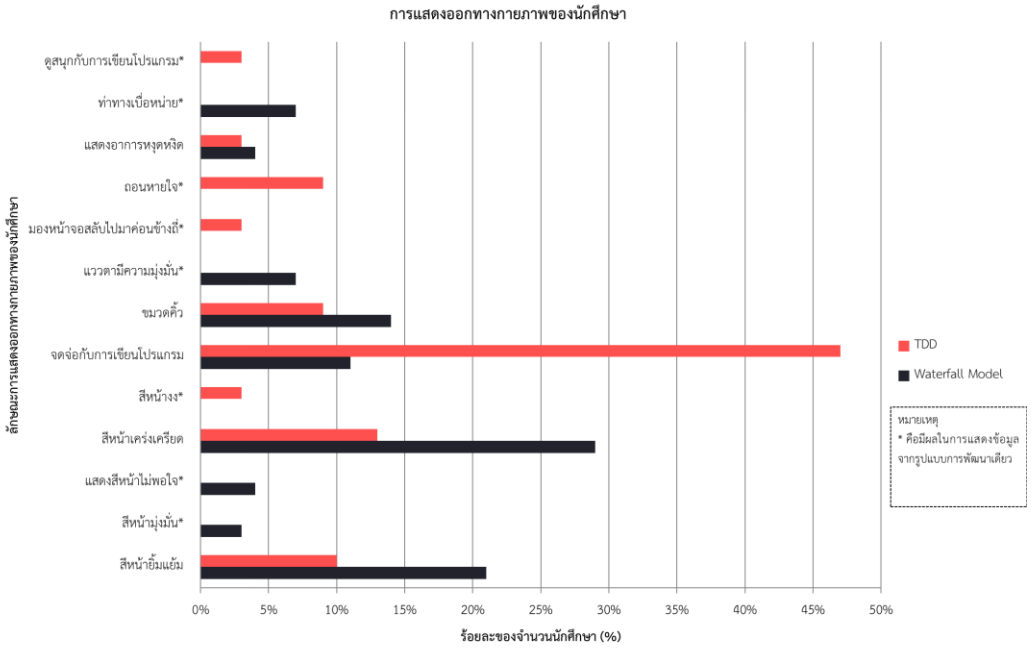
ซึ่งจากการวิเคราะห์ผลทางสถิติด้วยวิธี Paired Student's T-test มีการวิเคราะห์การเปรียบเทียบผลของ Pre-test และ Post-test ของนักศึกษาโดยรวมคะแนนคำตอบของแต่ละคน (1 คะแนน คือ สนใจน้อยสุด และ 5 คะแนน คือ สนใจมากที่สุด) ซึ่งผลลัพธ์ที่ได้จากการใช้วิธี Paired Student's T-test พบว่าค่าคะแนนเฉลี่ยของ Post-test มากกว่า Pre-test อย่างมีนัยสำคัญทางสถิติ (p-value = 0.00365 ที่ระดับความเชื่อมั่น 95%) ดังนั้นจึงปฏิเสธ H_0 ยอมรับ H_1 คือ นักศึกษาที่ได้รับการเรียนการสอนโดยใช้ TDD มีความสนใจในการพัฒนาซอฟต์แวร์ต่างกับการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก หรืออาจกล่าวอีกนัยหนึ่งได้ว่าการใช้ TDD ทำให้กลุ่มตัวอย่างมีความสนใจในการพัฒนาซอฟต์แวร์มากขึ้น

4.3 ผลการดำเนินการวิจัยจากการสังเกตการณ์

ผลการดำเนินการวิจัยจากการสังเกตมาจากข้อมูลที่ได้จากแบบฟอร์มการสังเกตการณ์และวิดีโอขณะทำการทดลอง โดยสามารถแบ่งผลการดำเนินการวิจัยจากการสังเกตการณ์ได้เป็น 3 ส่วน คือ 1) ด้านการแสดงออกทางกายภาพของนักศึกษา คือ ในขณะที่นักศึกษาเขียนโปรแกรมมีการแสดงออกทางกายภาพอย่างไรบ้าง 2) ด้านการเขียนโปรแกรมของนักศึกษา คือ ในขณะที่นักศึกษาเขียนโปรแกรมนักศึกษามีการเขียนโปรแกรมอย่างไร และ 3) ด้านการสืบค้นข้อมูลของนักศึกษา คือ ระหว่างที่นักศึกษาเขียนโปรแกรมนักศึกษามีการสืบค้นข้อมูลเพิ่มเติมจากอินเทอร์เน็ตหรือไม่ โดยผลจากการสังเกตแต่ละด้านมีรายละเอียดดังนี้

4.3.1 การแสดงออกทางกายภาพของนักศึกษา

การแสดงออกทางกายภาพที่มีการแสดงออกระหว่างการเขียนโปรแกรม แบ่งเป็น 2 ส่วนตามการสังเกตการณ์ 2 ครั้ง ดังนี้ 1) การแสดงออกทางกายภาพจากการสังเกตในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และ 2) การแสดงออกทางกายภาพจากการสังเกตในการพัฒนาซอฟต์แวร์โดยใช้ TDD ซึ่งมีลักษณะที่แสดงออกทางกายภาพ ดังรูปที่ 4.15



รูปที่ 4.15 แสดงการแสดงออกทางกายภาพของนักศึกษาในการทดลอง

การแสดงออกทางกายภาพที่สังเกตได้จากการทดลองภายในการพัฒนาซอฟต์แวร์ 2 ครั้ง คือ การพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD มีลักษณะที่แสดงให้เห็น คือ

(1) สีหน้ายิ้มแย้มในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 21% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 10%

(2) สีหน้ามุ่งมั่นในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 3% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 0%

(3) แสดงสีหน้าไม่พอใจในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 4% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 0%

(4) สีหน้าเคร่งเครียดในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 29% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 13%

(5) สีหน้างงในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 0% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 3%

(6) จดจ่อกับการเขียนโปรแกรมในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 11% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 47%

(7) ขมวดคิ้วในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 14% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 9%

(8) แววตามีความมุ่งมั่นในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 7% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 0%

(9) มองหน้าจอสลับไปมาค่อนข้างถี่ในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 0% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 3%

(10) ถอนหายใจในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 0% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 9%

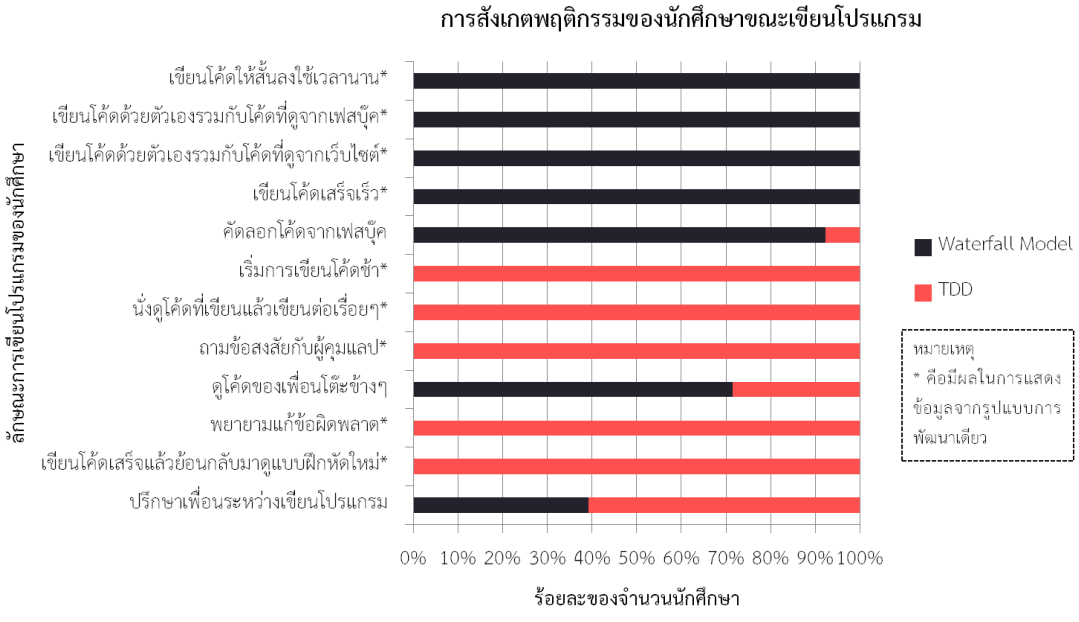
(11) แสดงอาการหงุดหงิดในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 4% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 3%

(12) ท่าทางเบื่อหน่ายในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 7% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแสดงออก จำนวน 0%

(13) ดูสนุกกับการเขียนโปรแกรมในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกมีการแสดงออก จำนวน 0% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD จำนวน 3%

4.3.2 การเขียนโปรแกรมของนักศึกษา

การสังเกตผู้เข้าร่วมทดลองในระหว่างการเขียนโปรแกรมประกอบด้วย 2 ส่วน คือ 1) การเขียนโปรแกรมในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และ 2) การเขียนโปรแกรมโดยใช้ TDD ซึ่งสามารถสังเกตได้จากพฤติกรรมในการเขียนโปรแกรมของนักศึกษา แสดงดังรูปที่ 4.16



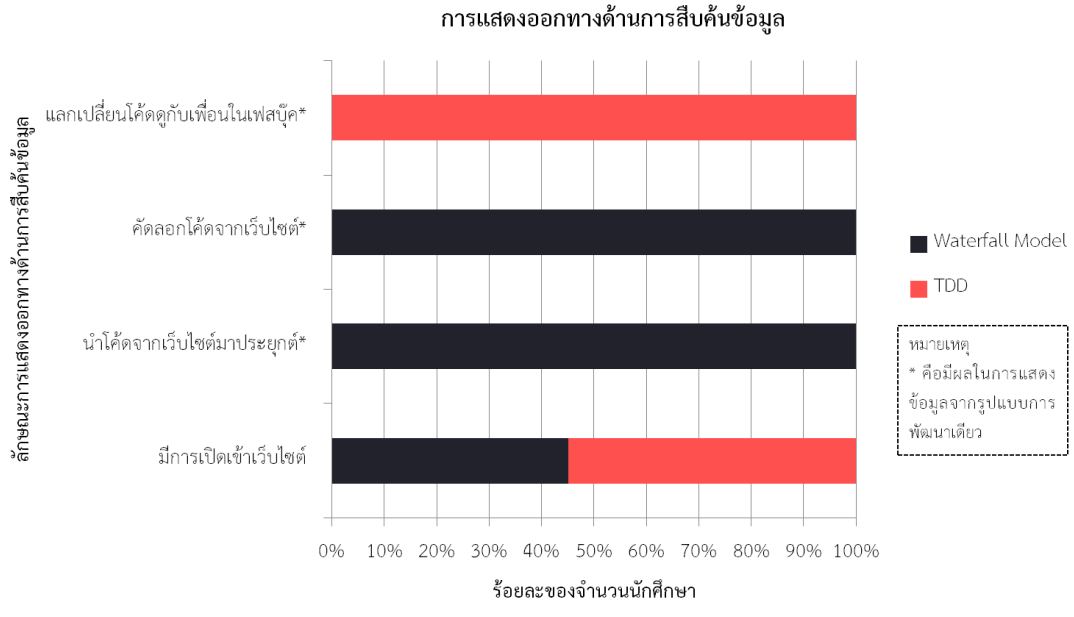
รูปที่ 4.16 แสดงการเขียนโปรแกรมของนักศึกษาจากการสังเกตในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD

การเขียนโปรแกรมที่สามารถสังเกตได้ระหว่างการเขียนโปรแกรมของนักศึกษา ในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD ในการทดลองสามารถสังเกตการเขียนโปรแกรมของนักศึกษาได้มีทั้งหมด 12 รูปแบบ โดยค่าเปอร์เซ็นต์ที่แสดงภายในวงเล็บ 2 ค่าเป็นค่าที่ได้จากการสังเกตด้านการเขียนโปรแกรม ซึ่งค่าที่ 1 เป็นค่าที่แสดงจากการสังเกตในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก ส่วนค่าที่ 2 เป็นค่าที่แสดงจากการสังเกตในการพัฒนาซอฟต์แวร์แบบ TDD คือ 1) ปรึกษาเพื่อนระหว่างเขียนโปรแกรม (45%, 70%) 2) เขียนโค้ดเสร็จแล้วย้อนกลับมาดูแบบฝึกหัดใหม่ (0%, 2%) 3) พยายาม

แก้ข้อผิดพลาด (0%, 7%) 4) ดูโค้ดของเพื่อนโต๊ะข้าง ๆ (10%, 4%) 5) ถามข้อสงสัยกับผู้คุมห้องปฏิบัติการ (0%,10%) 6) นั่งดูโค้ดที่เขียนแล้วเขียนต่อเรื่อย ๆ (0%, 2%) 7) เริ่มการเขียนโค้ดช้า (0%, 2%) 8) คัดลอกโค้ดจากเฟซบุ๊ก (24%, 2%) 9) เขียนโค้ดเสร็จเร็ว (7%, 0%) 10) เขียนโค้ดด้วยตัวเองรวมกับโค้ดที่ดูจากเว็บไซต์ (4%, 0%) 11) เขียนโค้ดด้วยตัวเองรวมกับโค้ดที่ดูจากเฟซบุ๊ก (7%, 0%) และ 12) เขียนโค้ดให้สั้นลงใช้เวลานาน (3%, 0%)

4.3.3 การสืบค้นข้อมูลของนักศึกษา

ผู้วิจัยได้สังเกตพฤติกรรมการการสืบค้นข้อมูลของผู้เข้าร่วมวิจัยด้วยกัน 2ครั้ง ประกอบด้วย 1) การสืบค้นข้อมูลจากการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และ 2) การสืบค้นข้อมูลจากการสังเกตในการพัฒนาซอฟต์แวร์โดยใช้ TDD ซึ่งสามารถสังเกตได้จากสิ่งที่นักศึกษาสืบค้นข้อมูล แสดงดังรูปที่ 4.17



รูปที่ 4.17 แสดงการสืบค้นข้อมูลของนักศึกษาจากการสังเกตในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD

การสืบค้นข้อมูลในการทดลองที่สามารถสังเกตได้ระหว่างการเขียนโปรแกรมของนักศึกษาในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD ในการทดลองสามารถสังเกตได้ คือ 1) มีการเปิดเข้าเว็บไซต์ในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก จำนวน 74% ส่วนในการพัฒนาซอฟต์แวร์

แบบ TDD มีการเปิดเข้าเว็บไซต์ จำนวน 90% 2) มีการเปิดเข้าเว็บไซต์แล้วนำมาประยุกต์ ในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก จำนวน 5% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการเปิดเข้าเว็บไซต์แล้วนำมาประยุกต์ จำนวน 0% 3) คัดลอกโค้ดจากเว็บไซต์ในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก จำนวน 21% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการคัดลอกโค้ดจากเว็บไซต์ จำนวน 0% และ 4) มีการแลกเปลี่ยนโค้ดดูกับเพื่อนในการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก จำนวน 0% ส่วนในการพัฒนาซอฟต์แวร์แบบ TDD มีการแลกเปลี่ยนโค้ดดูกับเพื่อน จำนวน 10%

บทที่ 5

อภิปรายและบทสรุปผลการวิจัย

5.1 อภิปรายผลการวิจัย (Discussion)

การอภิปรายผลในงานวิจัยมาจากผลการตอบแบบสอบถามและการสัมภาษณ์ของนักศึกษาทั้งก่อนและหลังการทดลอง รวมถึงผลจากการสังเกตการณ์ขณะทำการทดลอง

จากการกำหนดตัวบ่งชี้ด้านความสนใจทั้งหมด 5 ตัวบ่งชี้ และได้ทำการตรวจสอบตัวบ่งชี้ด้วยวิธีการวิเคราะห์องค์ประกอบเชิงยืนยันจากแบบสอบถามแบบ Likert Scale โดยผลลัพธ์จากการวิเคราะห์องค์ประกอบแสดงให้เห็นว่า ตัวบ่งชี้ด้านความสนใจประกอบด้วยตัวบ่งชี้ 5 ตัวบ่งชี้ คือ ความกระตือรือร้นในสิ่งที่สนใจ ศึกษาเพิ่มเติมด้วยตนเอง การทำแบบฝึกหัด เวลาในการทำแบบฝึกหัด และเวลาในการศึกษาเพิ่มเติมด้วยตนเอง ซึ่งจากตัวบ่งชี้ทั้ง 5 ตัวบ่งชี้ สามารถนำไปใช้ในการตั้งคำถามเพื่อวัดผลด้านความสนใจของกลุ่มผู้ทดลองได้

จากการตอบแบบสอบถามและการสัมภาษณ์ของนักศึกษาถึงความหมายของคำว่านักพัฒนาซอฟต์แวร์ สามารถสรุปคำตอบทั้งหมดออกมาเป็นนิยามได้ว่า นักพัฒนาซอฟต์แวร์ คือ ผู้พัฒนาและออกแบบโปรแกรม แอปพลิเคชัน หรือซอฟต์แวร์ ที่มีอยู่เดิมให้ดีขึ้นหรือพัฒนาขึ้นมาใหม่ตามที่ได้รับคำสั่ง หรือพัฒนาขึ้นมาใหม่เพื่ออำนวยความสะดวกในชีวิตประจำวัน รวมถึงเป็นผู้ที่ต้องทำการดูแลรักษาระบบให้มีคุณภาพและยังต้องทราบทุกอย่างในกระบวนการพัฒนาซอฟต์แวร์ ซึ่งอาจถือเป็นนิยามนิยามหนึ่งของนักพัฒนาซอฟต์แวร์จากนักศึกษาที่เรียนในสาขาวิชาวิศวกรรมซอฟต์แวร์

จากการตอบคำถามในแบบสอบถามและการสัมภาษณ์ก่อนการทดลองของนักศึกษาแสดงให้เห็นว่านักศึกษาที่ใช้วิธีการเรียนด้วยรูปแบบการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกจะไม่ค่อยให้

ความสนใจกับการทดสอบระบบ รวมถึงในการทดสอบระบบของนักศึกษาในการทำโครงการ นักศึกษาทำการทดสอบโดยใช้วิธีการสุ่มป้อนค่าข้อมูลเพื่อใช้ในการทดสอบข้อมูลเท่านั้น แต่นักศึกษาไม่ได้ทำการทดสอบกระบวนการในการทำงานของโค้ดภายใน และนักศึกษาจะทำการแก้ไขระบบเมื่อพบข้อผิดพลาดที่ส่งผลในขณะที่โปรแกรมทำงานเท่านั้น อีกทั้งจากการตอบแบบสอบถามและการสัมภาษณ์เรื่องการทดสอบโครงการที่ผ่านมาของนักศึกษา นักศึกษามีความเข้าใจว่าการใช้คำสั่ง Exception ในภาษาจาวาเป็นการทดสอบระบบวิธีหนึ่ง

จากการตอบแบบสอบถามและการสัมภาษณ์เรื่องการศึกษาเพิ่มเติมในสิ่งที่นักศึกษาสนใจ พบว่าจากการตอบแบบสอบถามและการสัมภาษณ์นักศึกษาให้ความสนใจในการศึกษาเพิ่มเติมเกี่ยวกับการเขียนโปรแกรมสูงสุดถึง 70% แต่พบว่าในการศึกษาเพิ่มเติม นักศึกษาไม่ได้ให้ความสนใจศึกษาเรื่องการทดสอบระบบและกระบวนการในการเขียนโปรแกรม หรือวิธีการพัฒนาซอฟต์แวร์

จากการศึกษาพบว่าการนำ TDD ไปใช้ในการเรียนการสอนมีส่วนช่วยในการกระตุ้นความสนใจของนักศึกษาให้มีความสนใจในการพัฒนาซอฟต์แวร์ จากการทำการทดลองพบว่ามีจำนวนนักศึกษาให้ความสนใจการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น โดยมีการวัดผลจากแบบสอบถามและการสัมภาษณ์ก่อนการเรียนและหลังการเรียน TDD ซึ่งนักศึกษาที่มีความสนใจเพิ่มมากขึ้นมีทัศนคติเกี่ยวกับ TDDโดยแบ่งเป็นออกเป็น 5 ด้าน คือ

(1) การพบข้อผิดพลาดที่เกิดขึ้นในโปรแกรม โดยนักศึกษาให้ความเห็นว่าการนำ TDD มาใช้การพัฒนาซอฟต์แวร์สามารถช่วยในการหาข้อผิดพลาดของโปรแกรมและช่วยลดปัญหาในการพบข้อผิดพลาดและการเขียนโปรแกรม

(2) การแก้ไขข้อผิดพลาดที่เกิดขึ้นของโปรแกรม โดยนักศึกษาให้ความคิดเห็นว่าการนำ TDD มาใช้ในการพัฒนาซอฟต์แวร์เป็นการช่วยปรับปรุงโปรแกรมให้มีความต้องการตรงกับความต้องการทำให้ทราบข้อผิดพลาดของโปรแกรมว่าเกิดจากส่วนใดและสามารถแก้ไขข้อผิดพลาดนั้นได้เร็วขึ้น

(3) คุณภาพของโปรแกรม โดยนักศึกษาให้ความคิดเห็นว่าการนำ TDD มาใช้ในการพัฒนาซอฟต์แวร์สามารถช่วยให้การเขียนโค้ดมีความถูกต้องเพิ่มขึ้นและโปรแกรมมีความซับซ้อนน้อยลงทำให้สามารถเข้าใจโค้ดได้ดียิ่งขึ้น รวมถึงทำให้มีการคำนึงถึงการปรับปรุงโค้ดให้ได้มาตรฐาน และเป็นการฝึกให้ทำการเขียนโค้ดให้มีคุณภาพ

(4) กระบวนการในการใช้ TDD โดยนักศึกษาให้ความคิดเห็นว่าการนำ TDDมาใช้ในการพัฒนาซอฟต์แวร์ช่วยให้มีการทำงานเป็นลำดับขั้นและมีความรอบคอบในการเขียนโค้ด

(5) ความรู้สึกของผู้ใช้ TDD โดยนักศึกษาให้ความคิดเห็นว่าการนำ TDD มาใช้ในการพัฒนาซอฟต์แวร์ เป็นการช่วยเพิ่มความมั่นใจในการเขียนโค้ดมากขึ้น และเป็นวิธีการคิดที่ทำให้รู้สึกสนุกในการเขียนโปรแกรม

นอกจากผลการทดลองจากแบบสอบถามและการสัมภาษณ์ในระหว่างการทดลองได้มีการสังเกตการณ์เพื่อศึกษาพฤติกรรมของนักศึกษาในการทดลอง โดยจากการสังเกตการณ์ สามารถแบ่งออกได้เป็น 3 ส่วน คือ

(1) จากการสังเกตการณ์ด้านการแสดงออกทางกายภาพ ซึ่งทำการจัดกลุ่มของข้อมูลที่ได้จากการสังเกตการณ์ทั้งหมด 13 กลุ่ม ในการสังเกตการณ์ในการทดลอง 2 ครั้ง คือ การพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD โดยสามารถจัดกลุ่มข้อมูลจากการสังเกตการณ์ทางด้านกายภาพซึ่งผู้วิจัยได้นำผลที่ได้มาแบ่งกลุ่มของข้อมูลเป็น 2 รูปแบบ คือ 1) อารมณ์ด้านบวก และ 2) อารมณ์ด้านลบ โดยอารมณ์ที่จัดอยู่ภายในอารมณ์ด้านบวก คือ สีหน้ายิ้มแย้ม สีหน้ามุ่งมั่น จดจ่อกับการเขียนโปรแกรม แววตามีความมุ่งมั่น และดูสนุกกับการเขียนโปรแกรม ส่วนอารมณ์ที่จัดอยู่อารมณ์ด้านลบ คือ แสดงสีหน้าไม่พอใจ สีหน้าเคร่งเครียด สีหน้างง ขมวดคิ้ว มองหน้าจอสลับไปมาค่อนข้างถี่ ถอนหายใจ แสดงอาการหงุดหงิด และท่าทางเบื่อหน่าย จากข้อมูลพบว่าในการทดลองโดยใช้วิธีการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกนักศึกษาแสดงอารมณ์ด้านบวก จำนวน 42% และแสดงออกในอารมณ์ด้านลบ จำนวน 58% ส่วนในการทดลองโดยใช้วิธีการพัฒนาซอฟต์แวร์แบบ TDD นักศึกษาแสดงอารมณ์ทางด้านบวก จำนวน 60% และแสดงออกถึงอารมณ์ทางด้านลบ จำนวน 40% โดยในงานวิจัยเป็นเพียงการเก็บข้อมูลจากการสังเกตการณ์ขั้นพื้นฐานเท่านั้น จากการสังเกตการณ์แสดงให้เห็นว่าขณะทำการทดลองการใช้วิธีการพัฒนาซอฟต์แวร์แบบ TDD นักศึกษาแสดงอารมณ์ด้านบวกมากกว่าการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก 18%

(2) จากการสังเกตการณ์ด้านเขียนโปรแกรมของนักศึกษาเปรียบเทียบระหว่างการพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและแบบ TDD แสดงให้เห็นว่านักศึกษาส่วนใหญ่ทำการปรึกษาเพื่อนจากโต๊ะที่นั่งคู่กันระหว่างการเขียนโปรแกรม จากการทดลองอาจจะแสดงให้เห็นว่าการทำการพัฒนาซอฟต์แวร์โดยใช้วิธีการ TDD ควรใช้วิธีการเขียนโปรแกรมแบบคู่มากกว่าการทำแบบเดี่ยว

(3) จากการสังเกตการณ์ด้านการสืบค้นข้อมูลของนักศึกษา เนื่องจากในการทดลองไม่ได้จำกัดในการสืบค้นข้อมูลทางอินเทอร์เน็ต จากการสังเกตการณ์พบว่าระหว่างการพัฒนาซอฟต์แวร์แบบจำลองน้ำตก และการพัฒนาซอฟต์แวร์แบบ TDD พบว่าในการพัฒนาซอฟต์แวร์แบบ TDD ไม่มีนักศึกษาที่ทำการคัดลอกโค้ดจากแหล่งสืบค้นข้อมูล

จากผลลัพธ์ที่ได้จากการดำเนินการซึ่งเป็นผลการวิจัยทางด้านความรู้สึกของผู้ใช้และมีความสอดคล้องกับงานวิจัยก่อนหน้านี้ทางด้าน TDD ซึ่งกล่าวว่าการนำ TDD ไปใช้ในการทำงานทำให้ซอฟต์แวร์และโค้ดภายในซอฟต์แวร์มีคุณภาพที่ดีขึ้น (Jeanzen and Saiedian, 2005; Pančur, *et al.*, 2003; Buffardi, 2012) และการใช้ TDD แสดงให้เห็นว่านักศึกษามีข้อบกพร่องในการเขียนโปรแกรมน้อยลง (Edwards, 2003) รวมถึงในงานวิจัยก่อนหน้านี้มีการแนะนำให้นำ TDD ไปใช้ในการเรียนการสอนเพื่อให้นักศึกษามีระเบียบวินัยในการพัฒนาซอฟต์แวร์ เนื่องจาก TDD จะสามารถช่วยปรับปรุงคุณภาพในการสร้างซอฟต์แวร์ (Jeanzen and Saiedian, 2005) จากที่กล่าวมาข้างต้นถึงงานวิจัยด้าน TDD แสดงให้เห็นว่าการใช้ TDD สามารถทำให้โปรแกรมมีประสิทธิภาพและคุณภาพของโค้ดเพิ่มขึ้น รวมถึงช่วยลดข้อบกพร่องในการเขียนโปรแกรมของนักศึกษา ซึ่งผลของงานวิจัยนี้ระบุให้เห็นว่า TDD สามารถช่วยกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ให้นักศึกษามีความสนใจในการพัฒนาซอฟต์แวร์เพิ่มขึ้น เนื่องจากนักศึกษาในปัจจุบันมีจำนวนผู้สนใจในการเป็นนักพัฒนาซอฟต์แวร์ลดน้อยลง ซึ่งการนำ TDD ไปใช้นอกจากจะสามารถช่วยกระตุ้นความสนใจของนักศึกษาแล้ว ยังทำให้นักศึกษาผลิตซอฟต์แวร์ให้มีประสิทธิภาพและคำนึงถึงคุณภาพของโค้ดเพิ่มขึ้น จากที่กล่าวมาข้างต้นอาจเป็นแนวทางหนึ่งในการปรับเปลี่ยนรูปแบบการเรียนการสอนเพื่อที่จะสามารถผลิตนักพัฒนาซอฟต์แวร์เข้าสู่ภาคอุตสาหกรรมให้มีจำนวนและคุณภาพมากขึ้น

จากผลในการสังเกตการณ์ในการทดลองซึ่งเป็นผลด้านอารมณ์ของผู้ใช้ โดยการนำผลทางการแสดงออกทางกายภาพมาวิเคราะห์อารมณ์ของนักศึกษาระหว่างการเขียนโปรแกรมทั้ง 2 รูปแบบ คือ การพัฒนาซอฟต์แวร์แบบจำลองน้ำตกและ TDD โดยผลที่นำมาวิเคราะห์ได้ทำการแบ่งการแสดงออกทางอารมณ์เป็น 2 ส่วน คือ อารมณ์ด้านบวกและอารมณ์ด้านลบ โดยจากผลการทดลองแสดงให้เห็นว่าขณะที่นักศึกษาใช้วิธีการ TDD นักศึกษาแสดงอารมณ์ทางด้านบวกมากกว่าการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก ซึ่งการศึกษาทางด้านอารมณ์ของนักพัฒนาซอฟต์แวร์จะเป็นหลักฐานเชิงประจักษ์เกี่ยวกับอารมณ์ของผู้พัฒนาซอฟต์แวร์ซึ่งยังมีอยู่ค่อนข้างน้อย (Graziotin, *et al.*, 2014, 2015) แต่ในการเก็บข้อมูลในการแสดงออกทางอารมณ์ของนักศึกษาเป็นเพียงการเก็บข้อมูลพื้นฐานเท่านั้น ซึ่งผลลัพธ์ที่ได้อาจจะเป็นแนวทางหนึ่งในการปรับเปลี่ยนรูปแบบการเรียนการสอนภายในการเรียนในห้องปฏิบัติการเพื่อทำให้นักศึกษามีอารมณ์ด้านบวกในการเรียนเพิ่มมากขึ้น ซึ่งอาจจะทำให้นักศึกษามีความรู้สึกสนใจในการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น นอกจากการสังเกตการณ์ทางด้านกายภาพแล้วยังได้มีการสังเกตการณ์ทางด้านการเขียนโปรแกรมของนักศึกษาด้วย ซึ่งจากผลลัพธ์แสดงให้เห็นว่านักศึกษาที่ใช้วิธีการ TDD ถึง 70% ทำการปรึกษาเพื่อนระหว่างการเขียนโปรแกรม จากผลลัพธ์แสดงให้เห็นว่าการใช้

วิธีการ TDD อาจเหมาะสำหรับการทำงานเป็นคู่ หรืออาจควรให้นักศึกษาได้ทำการพัฒนาซอฟต์แวร์เป็นคู่ (Pair programming) ซึ่งอาจสามารถกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น

5.2 ภัยคุกคามต่อความถูกต้อง (Threats to Validity)

ภัยคุกคามต่อความถูกต้อง เป็นปัจจัยที่ส่งผลต่อความถูกต้อง ความเชื่อถือ และความเที่ยงตรงของการดำเนินงานวิจัย โดยภัยคุกคามต่อความถูกต้องมี 3 รูปแบบ คือ 1) ภัยคุกคามต่อความถูกต้องเชิงโครงสร้าง 2) ภัยคุกคามต่อความถูกต้องภายใน และ 3) ภัยคุกคามต่อความถูกต้องภายนอก ซึ่งในงานวิจัยนี้ผู้วิจัยได้มีการระบุถึงภัยคุกคามดังนี้

5.2.1 ภัยคุกคามต่อความถูกต้องเชิงโครงสร้าง (Construct Validity)

ภัยคุกคามต่อความถูกต้องเชิงโครงสร้าง เป็นการพิจารณาว่าเครื่องมือในงานวิจัยสามารถวัดผลได้ตามรูปแบบ โครงสร้าง หรือลักษณะที่ควรเป็นในเรื่องที่ต้องการวัดตามวัตถุประสงค์ของงานวิจัย เช่น หากผู้วิจัยต้องการวัดสมรรถภาพสมองตามทฤษฎี X เครื่องมือที่ใช้จะต้องวัดได้ทุกองค์ประกอบ จึงจะเรียกว่าเครื่องมือชุดนี้มีความถูกต้องเชิงโครงสร้าง ซึ่งในงานวิจัยผู้วิจัยได้ระบุถึงภัยคุกคามที่อาจจะส่งผลต่อความถูกต้องเชิงโครงสร้างไว้ดังนี้

(1) ในงานวิจัยมีการใช้เครื่องมือในเก็บข้อมูล 3 รูปแบบ คือ การเก็บข้อมูลจากแบบสอบถาม โดยการเก็บข้อมูลจากแบบสอบถามมีวัตถุประสงค์ เพื่อเป็นการเก็บข้อมูลเชิงปริมาณที่อยู่ในรูปแบบ Likert Scale เพื่อใช้เปรียบเทียบคะแนนความสนใจก่อนและหลังการทดลอง และมีคำถามปลายเปิดเพื่อเก็บข้อมูลในเชิงบรรยาย การเก็บข้อมูลจากการสัมภาษณ์ ในงานวิจัยใช้คำถามในการสัมภาษณ์ที่อ้างอิงมาจากคำถามในแบบสอบถามปลายเปิด โดยในการสัมภาษณ์มีวัตถุประสงค์เพื่อ ให้ผู้ตอบสามารถตอบคำถามได้อย่างมีอิสระและสามารถใช้คำศัพท์เทคนิคในการตอบเพื่ออธิบายคำตอบของตัวเองได้ ซึ่งการอธิบายโดยการพูดจะทำได้ง่ายกว่าการเขียนตอบ รวมถึงผู้สัมภาษณ์สามารถถามคำถามเพิ่มเติมจากคำตอบที่ผู้ตอบตอบกลับมาได้ และการเก็บข้อมูลจากการสังเกตการณ์ มีวัตถุประสงค์เพื่อนำ

สิ่งที่เก็บข้อมูลได้ไปจัดเป็นรูปแบบเพื่อดูว่าจากการทดลองในแต่ละครั้ง นักศึกษามีสิ่งที่แสดงออกแตกต่างกันอย่างไร รวมถึงอาจจะทราบตัวแปรสังเกตที่ไม่สามารถระบุได้ในตัวแปรภายในงานวิจัย

(2) ในงานวิจัยนี้ผู้วิจัยได้ทำการวัดความสนใจของนักศึกษาจากการกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ด้วยการ TDD โดยใช้เครื่องมือในการวัดอยู่ในรูปแบบแบบสอบถาม ซึ่งการจัดทำแบบสอบถามในงานวิจัยผู้วิจัยได้ทำการศึกษาทฤษฎีต่าง ๆ ที่เกี่ยวข้องกับนิยามเรื่องความสนใจ เพื่อนำมาออกแบบแบบสอบถาม เนื่องจากการศึกษาทฤษฎีด้านความสนใจพบว่ายังไม่ได้มีการกำหนดตัวบ่งชี้ที่ชัดเจนทางด้านความสนใจ ทางผู้วิจัยจึงได้ทำการศึกษาทฤษฎีทางด้านความสนใจและได้นำทฤษฎีมากำหนดเป็นตัวบ่งชี้เพื่อใช้ในการกำหนดข้อคำถามภายในแบบสอบถาม อย่างไรก็ตามจากการศึกษานิยามเรื่องความสนใจ แสดงให้เห็นว่าคำนิยามด้านความสนใจมีการกำหนดนิยามที่ค่อนข้างหลากหลายทั้งในทางด้านจิตวิทยา การศึกษา และทางด้านวิศวกรรมซอฟต์แวร์ และคำนิยามด้านความสนใจยังไม่มีการกำหนดที่เป็นที่ยอมรับ ซึ่งในการออกแบบสอบถามในงานวิจัยได้ผ่านการตรวจสอบโดยผู้เชี่ยวชาญทางด้านวิศวกรรมซอฟต์แวร์จำนวน 1 คน และได้ทำการตรวจสอบตัวบ่งชี้ด้านความสนใจที่กำหนดขึ้นและนำไปใช้ในแบบสอบถามด้วยการวิเคราะห์องค์ประกอบเชิงยืนยัน จากการวิเคราะห์องค์ประกอบเชิงยืนยันแบบสอบถามแสดงให้เห็นว่า ตัวบ่งชี้ที่ทำการกำหนดขึ้นจัดอยู่ในองค์ประกอบเดียวกันจริง

5.2.2 ภัยคุกคามต่อความถูกต้องภายใน (Internal Validity)

ภัยคุกคามต่อความถูกต้องภายในเป็นคุณลักษณะที่สามารถสรุปได้ว่าผลการทดลองที่เกิดขึ้นเป็นผลอันเนื่องมาจากสิ่งที่ดำเนินการในการทดลองจริง สำหรับในงานวิจัยนี้ปัจจัยที่มีผลต่อความถูกต้องภายในเช่น วุฒิภาวะของกลุ่มตัวอย่าง เครื่องมือที่ใช้ในงานวิจัย การถดถอยทางสถิติ เป็นต้น สำหรับในงานวิจัยผู้วิจัยได้ระบุถึงภัยคุกคามที่อาจจะส่งผลต่อความถูกต้องภายในไว้ดังนี้

(1) เนื่องจากในงานวิจัยเป็นการทำงานวิจัยเชิงการทดลองโดยใช้แบบสอบถามและการสัมภาษณ์ที่ผู้วิจัยได้ทำการออกแบบขึ้น ซึ่งก่อนที่ผู้วิจัยจะทำการทดลองจริงได้ทำการศึกษานำร่องก่อนการทดลองจริง เพื่อตรวจสอบแบบสอบถามและคำถามในการสัมภาษณ์ว่านักศึกษาสามารถเข้าใจคำถามตรงกับที่ผู้วิจัยต้องการถามหรือไม่ รวมถึงก่อนทำการทดลองจริงทางผู้วิจัยได้แจ้งข้อตกลงในการเป็นผู้เข้าร่วมการทดลองแก่นักศึกษาว่าผลจากการทำการตอบแบบสอบถามและการสัมภาษณ์ในการทดลอง ไม่มีผลต่อคะแนนในรายวิชาที่ผู้วิจัยเข้าไปทำการทดลอง

(2) เนื่องจากในงานวิจัยมีการเก็บรวบรวมข้อมูลจากนักศึกษา จำนวน 52 คนในรูปแบบของแบบสอบถาม การสัมภาษณ์ และการสังเกตการณ์ โดยการเก็บรวบรวมข้อมูลแบบสอบถามและการสัมภาษณ์มีการเก็บข้อมูล 2 ครั้ง คือ ก่อนการทดลองและหลังการทดลอง พบว่าแบบสอบถามที่ได้กลับมาไม่ครบตามจำนวนนักศึกษาในแบบสอบถามหลังการทดลอง จึงทำการคัดเอาแบบสอบถามของนักศึกษาที่ส่งเพียงก่อนการทดลอง หรือหลังการทดลองเพียงอย่างเดียวออก เนื่องจากข้อมูลที่ต้องการนำไปวิเคราะห์ผลนั้นต้องใช้ข้อมูลก่อนการทดลองและหลังการทดลอง ส่วนในการสัมภาษณ์นั้นเนื่องจากในการสัมภาษณ์มีผู้ทำการสัมภาษณ์ทั้งหมด 4 คน โดยผู้สัมภาษณ์นักศึกษาทั้ง 4 คน จะได้รับคำถามที่จำเป็นต้องถามแก่นักศึกษาจากผู้วิจัย เนื่องจากรูปแบบการสัมภาษณ์เป็นลักษณะการสัมภาษณ์กึ่งโครงสร้างจึงอาจทำให้มีคำถามที่ทำการสัมภาษณ์นักศึกษาแต่ละคนมีความแตกต่างกัน และในส่วนของข้อมูลจากการสังเกตการณ์เนื่องจากมีการจดบันทึกด้วยแบบฟอร์มการสังเกตการณ์โดยผู้สังเกตการณ์อาจไม่สามารถจดบันทึกการแสดงออกของนักศึกษาได้ทั้งหมด ทางผู้วิจัยจึงทำการอัดวิดีโอเพื่อเป็นการเก็บข้อมูลสำรองในการสังเกตการณ์ในการทดลอง รวมถึงในการทดลองภายในห้องปฏิบัติการทั้ง 2 ครั้ง เป็นการให้นักศึกษาทำการพัฒนาโปรแกรม FizzBuzz ซึ่งโปรแกรม FizzBuzz เป็นการพัฒนาซอฟต์แวร์ขนาดเล็กที่เป็นเพียงพื้นฐานของการพัฒนาซอฟต์แวร์ โดยในการทดลองได้กำหนดให้นักศึกษาพัฒนาซอฟต์แวร์ FizzBuzz ทั้ง 2 ครั้ง แต่ใช้รูปแบบการพัฒนาซอฟต์แวร์แตกต่างกัน คือ ครั้งที่ 1 ให้นักศึกษาทำการพัฒนาโปรแกรม FizzBuzz โดยใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก ส่วนครั้งที่ 2 ให้นักศึกษาทำการพัฒนาโปรแกรม FizzBuzz โดยใช้การพัฒนาซอฟต์แวร์แบบ TDD ซึ่งอาจทำให้นักศึกษาเกิดการเรียนรู้เนื่องจากเป็นการพัฒนาโปรแกรมเดียวกัน

หลังจากที่ได้ข้อมูลจากการทดลองแล้วผู้วิจัยจึงนำข้อมูลที่ได้ไปทำการวิเคราะห์ผล โดยข้อมูลที่ได้จากการทดลองมี 2 รูปแบบ คือ 1) ข้อมูลเชิงปริมาณที่ได้มาจากการตอบแบบสอบถามแบบ Likert Scale ซึ่งนำไปวิเคราะห์ผลทางสถิติ และ 2) ข้อมูลเชิงบรรยายได้มาจากการตอบแบบสอบถามปลายเปิด การสัมภาษณ์ และการสังเกตการณ์ ข้อมูลทั้ง 3 ส่วนนี้ ผู้วิจัยต้องนำไปวิเคราะห์ผลโดยใช้วิธีการ Coding Analysis เพื่อทำให้สามารถจัดกลุ่มของข้อมูลได้ ซึ่งการวิเคราะห์ผลแบบ Coding Analysis จำเป็นต้องใช้ความชำนาญในเรื่องที่ทำการวิเคราะห์ผลเพื่อจัดกลุ่ม จึงอาจทำให้ผลที่ได้จากการวิเคราะห์ภายในงานวิจัยมีผลที่ไม่ถูกต้องสมบูรณ์ รวมถึงในการจัดกลุ่มจากการวิเคราะห์ข้อมูลอาจจะเกิดความลำเอียง (Bias) ในการวิเคราะห์ข้อมูล

(3) ในส่วนของข้อมูลที่ทำการเก็บในรูปแบบแบบสอบถาม มีการนำไปวิเคราะห์ผลทางสถิติด้วยวิธีการวิเคราะห์องค์ประกอบเชิงยืนยัน เพื่อตรวจสอบตัวบ่งชี้ที่ทางผู้วิจัยกำหนดขึ้นจาก

นิยามด้านความสนใจ แต่เนื่องจากกลุ่มข้อมูลที่ทำการศึกษามีจำนวนค่อนข้างน้อย คือ มีจำนวนกลุ่มข้อมูลจากการตอบแบบสอบถามจำนวน 42 คน ดังนั้นผลของการวิเคราะห์ด้วยวิธีการทางสถิติอาจจะได้ค่าที่มีความคลาดเคลื่อนเล็กน้อย

5.2.3 ภัยคุกคามต่อความถูกต้องภายนอก (External Validity)

ภัยคุกคามต่อความถูกต้องภายนอก เป็นการระบุว่าผลวิจัยที่ได้สามารถนำผลที่ได้ไปใช้กับประชากรที่มีขนาดใหญ่กว่าได้หรือไม่ และสามารถนำวิธีการทดลอง วิธีการวัด และเครื่องมือต่าง ๆ ไปใช้กับกลุ่มประชากรอื่นได้หรือไม่ โดยปัจจัยที่มีผลต่อความเที่ยงตรงภายนอก เช่น ปฏิกิริยาของผู้ถูกทดลอง การทดลองซ้ำหลายวิธี อิทธิพลร่วมของการทดลองก่อนการทดลองเป็นต้น ซึ่งในงานวิจัยผู้วิจัยได้ระบุถึงภัยคุกคามที่อาจจะส่งผลต่อความถูกต้องภายนอกไว้ดังนี้

เนื่องจากการทดลองในงานวิจัยเป็นการใช้กลุ่มตัวอย่างของนักศึกษาในสาขาวิชาวิศวกรรมซอฟต์แวร์ โดยผู้ที่เข้าร่วมในการทดลองต้องเป็นนักศึกษาที่ผ่านการเรียนพื้นฐานการเขียนโปรแกรมจากรายวิชาการโปรแกรมเชิงอ็อบเจกต์และการทำโครงการด้วยภาษาจาวามาก่อน อีกทั้งในงานวิจัยเป็นการศึกษาการนำ TDD ไปใช้เพื่อกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์กับกลุ่มตัวอย่างในสาขาวิชาวิศวกรรมซอฟต์แวร์ ซึ่งไม่ได้ครอบคลุมนักศึกษาที่กำลังศึกษาในสาขาวิชาอื่น เช่น สาขาวิชาเทคโนโลยีสารสนเทศ อย่างไรก็ตามผลจากการศึกษาจะเป็นแนวทางเบื้องต้นในการกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์

5.3 สรุปผลและข้อเสนอแนะงานวิจัย

งานวิจัยนี้เป็นการให้นักศึกษาได้เรียนรู้และนำ TDD ไปปฏิบัติใช้งาน เพื่อกระตุ้นความสนใจของนักศึกษาโดยใช้วิธีการทดลองแล้วเก็บข้อมูลจากแบบสอบถาม การสัมภาษณ์ และการสังเกตการณ์ ผลจากการทดลองสามารถตอบสมมติฐานงานวิจัยได้ว่า นักศึกษาที่ได้รับการเรียนการสอนโดยใช้ TDD มีความสนใจในการพัฒนาซอฟต์แวร์ต่างกับการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก โดยนักศึกษาที่ใช้ TDD ในการเรียนมีความสนใจในการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น โดยผลจากการ

ทดลองจะเป็นแนวทางหนึ่งในการช่วยส่งเสริมการเรียนด้านการพัฒนาซอฟต์แวร์ในสถาบันการศึกษาและอาจจะทำให้ภาคการศึกษาสามารถเพิ่มบุคลากรที่มีความสนใจทางด้านการพัฒนาซอฟต์แวร์เข้าสู่ภาคอุตสาหกรรมซอฟต์แวร์มากขึ้น นอกจากนี้งานวิจัยนี้จะช่วยเพิ่มหลักฐานเชิงประจักษ์ซึ่งจะเป็นประโยชน์ต่อนักวิจัยด้านวิศวกรรมซอฟต์แวร์เพื่อหาวิธีการให้นักศึกษามีความสนใจในการเรียนด้านการพัฒนาซอฟต์แวร์มากขึ้น รวมถึงเป็นการเพิ่มหลักฐานเชิงประจักษ์ด้านข้อจำกัดของการนำ TDD ไปใช้ในการเรียนการสอนของนักศึกษาเพื่อให้ผู้สนใจคนอื่นที่ต้องการนำ TDD ไปใช้ในการเรียนการสอนทราบถึงข้อจำกัดที่อาจจะเกิดขึ้นเมื่อนักศึกษาใช้วิธีการ TDD โดยจากการทำการทดลองการนำ TDD ไปช่วยในการกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์ผู้วิจัยมีข้อเสนอแนะที่ได้จากการทดลอง ดังนี้

(1) จากการทดลองแสดงให้เห็นว่านักศึกษามีความคุ้นเคยกับการใช้การพัฒนาซอฟต์แวร์แบบจำลองน้ำตก ซึ่งอาจทำให้การเปลี่ยนรูปแบบการพัฒนาซอฟต์แวร์เป็นแบบ TDD ค่อนข้างทำได้ยาก ดังนั้นโปรแกรมที่ใช้เพื่อการทดลองควรเป็นโปรแกรมที่ทำการพัฒนาได้ไม่ยากนัก เนื่องจากกระบวนการทั้ง 2 กระบวนการมีวิธีการคิดค่อนข้างแตกต่างกัน

(2) จากการทดลองในงานวิจัย ผู้วิจัยได้กำหนดให้นักศึกษาพัฒนาโปรแกรม FizzBuzz ซึ่งเป็นการพัฒนาโปรแกรมขนาดเล็ก ดังนั้นหากมีการนำการทดลองไปใช้เพื่อกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์เพิ่มเติม ผู้วิจัยอาจทำการเลือกการพัฒนาโปรแกรมอื่นที่มีลักษณะใกล้เคียงกัน

(3) ในการพัฒนาโปรแกรมผู้วิจัยได้กำหนดให้นักศึกษาใช้เครื่องมือ Eclipse ในการพัฒนาซอฟต์แวร์ เนื่องจาก Eclipse มี plug in ในการทำการทดสอบระบบแบบโอเพนซอร์ส แต่พบว่านักศึกษาค่อนข้างนิยมใช้โปรแกรมอื่นในการเขียนโปรแกรม เช่น NetBeans ดังนั้นก่อนทำการทดลองภายในห้องปฏิบัติการ ควรอธิบายถึงพื้นฐานของการใช้โปรแกรมที่ใช้ในการทดลองก่อนเพื่อให้นักศึกษารู้จักเครื่องที่ใช้ในการพัฒนาซอฟต์แวร์

(4) ระยะเวลาในการดำเนินการทดลองค่อนข้างมีผลต่อความสนใจของนักศึกษา เนื่องจากการทดลองมีระยะเวลาค่อนข้างจำกัดนักศึกษาจึงไม่มีความคุ้นชินกับการนำ TDD ไปใช้ต่อ และหลังจากจบการทดลองไม่มีรายวิชาที่ทำการสอน หรือนำ TDD ไปใช้ต่อ ดังนั้นควรกำหนดระยะเวลาในการทำการทดลองให้มีระยะเวลาเหมาะสม และอาจทำการศึกษาระยะยาวหลังจบการทดลองว่ามีวิชาที่รองรับการนำ TDD ไปใช้ต่อหรือไม่

จากการทำงานวิจัยนี้ผู้วิจัยเชื่อว่าผลลัพธ์ที่ได้จากงานวิจัยจะเป็นประโยชน์ต่อนักวิจัยทางด้านวิศวกรรมซอฟต์แวร์ เพื่อเป็นแนวทางในการศึกษาการกระตุ้นความสนใจของบุคลากรในการ

พัฒนาซอฟต์แวร์ รวมถึงเป็นการหันมาให้ความสำคัญกับวิธีการในพัฒนาซอฟต์แวร์และความสนใจทางด้านอารมณ์ของนักพัฒนาซอฟต์แวร์ นอกจากนี้ผลวิจัยที่ได้น่าจะเป็นประโยชน์กับบุคลากรทางด้านการศึกษาโดยเฉพาะทางด้านวิศวกรรมซอฟต์แวร์ในประเทศไทย เพื่อเป็นแนวทางหนึ่งในการปรับเปลี่ยนการเรียนการสอนเพื่อกระตุ้นให้นักศึกษามีความสนใจในการพัฒนาซอฟต์แวร์เพิ่มมากขึ้น และเป็นการเพิ่มหลักฐานเชิงประจักษ์ทางด้านวิศวกรรมซอฟต์แวร์ในเรื่องการนำ TDD ไปใช้ในการเรียนการสอน รวมถึงข้อจำกัดและข้อแสนอแนะที่ผู้วิจัยคนอื่น หรือบุคลากรทางด้านการศึกษาต้องการนำวิธีการ TDD ไปใช้กับการเรียนการสอน

เอกสารอ้างอิง

คณะกรรมการอาเซียน สภาหอการค้าไทยและเขตอุตสาหกรรมซอฟต์แวร์แห่งชาติ. (2015) “ซอฟต์แวร์ไทยจะก้าวอย่างไรในเวทีอาเซียน AEC 2015.” (ออนไลน์) เข้าถึงได้ที่: http://www.news.thaiware.com/material/aec/aec_conclusion1.pdf (วันที่ 15 มีนาคม, 2558).

จิตมณี อะเมกอง (2545). “การศึกษาความเข้าใจในการอ่านและความสนใจในการเรียนภาษาอังกฤษของนักเรียนชั้นมัธยมศึกษาปีที่ 1 ที่ได้รับการสอนแบบมุ่งประสบการณ์ภาษา (รูปแบบที่ 1) กับการสอนแบบเดิม.” วิทยานิพนธ์ปริญญาการศึกษามหาบัณฑิต, สาขาวิชาการมัธยมศึกษา, บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ.

จิราพร สุจริต. (2543). “การเปรียบเทียบความเข้าใจในการอ่านและความสนใจในการเรียน วิชาภาษาอังกฤษของนักเรียนชั้นมัธยมศึกษาปีที่ 3 ที่ได้รับการสอนตามทฤษฎีเพื่อการสื่อสารของ คีธ จอห์นสัน (Keith Johnson) กับการสอนตามคู่มือครู.” ปริญญานิพนธ์ กศ.ม. (การมัธยมศึกษา), บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ ประสานมิตร.

รัตนา คงคาเนาวรัตน์. (2542). “การศึกษาและความสนใจในการเขียนสะกดคำยากวิชาภาษาไทยของนักเรียนชั้นประถมศึกษาปีที่ 4 โดยใช้ชุดฝึกสะกดคำยาก.” ปริญญานิพนธ์ กศ.ม. กรุงเทพฯ, มหาวิทยาลัยศรีนครินทรวิโรฒ ประสานมิตร.

สุชา จันทร์เอม (2542). *จิตวิทยาวัยรุ่น*, ทยวฒันาพานิช.

เสถียร ศรีรัตน์. (2522). “ความสัมพันธ์ระหว่างความก้าวร้าวกับความสนใจในอาชีพของนักเรียน ชั้นมัธยมศึกษาปีที่ 3 ในโรงเรียนที่ใช้หลักสูตรกว้างในจังหวัดสงขลา.” วิทยานิพนธ์ปริญญาการศึกษามหาบัณฑิต, สาขาวิชาการบริหารการศึกษา, บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ ประสานมิตร.

Abrahamsson, P., Marchesi, M., and Maurer, F. (2009). “Agile Processes in Software Engineering and Extreme Programming*”, Proceeding of The 10th International Conference XP 2009,* Pula, Sardinia, Italy: 25-29 May, 2009.

Abramson, J., and Abramson, Z. (2008). “Research Methods in Community Medicine: Surveys, Epidemiological Research, Programme Evaluation. Clinical Trials.” *Applied Physiology*, 33(6), 1265-1266.

Beck, K. (2003). *Test-driven development: by example*. PearsonEducation,Inc., Boston.

Benbasat, I., Goldstein, D. K., and Mead, M. (1987). "The case research strategy in studies of information systems." *MIS quarterly, JSTOR*, 11(3), 369-386.

Bhat, T., and Nagappan, N. (2006). "Evaluating the Efficacy of Test-Driven Development: Industrial Case Studies.", *Proceeding of The 2006 ACM/IEEE International Symposium on Empirical Software Engineering*, Rio de Janeiro, Brazil: 21-22 September, 2006.

Booch, G. (2015). "The Future of Software Engineering (SEIP Keynote).", *Proceeding of The 37th IEEE International Conference on Software Engineering,* Florence, Italy: 16-24 May, 2015.

Buffardi, K. (2012). "Understanding and Persuading Adherence to Test-Driven Development.", *Proceeding of The 9th Annual International Conference on International Computing Education Research*, Auckland, New Zealand: 09-11 September, 2012.

Canfora, G., Cimitile, A., Garcia, F., Piattini, M., and Visaggio, C. A. (2006). "Evaluating Advantages of Test Driven Development: a Controlled Experiment with Professionals.", *Proceeding of The 2006 ACM/IEEE International Symposium on Empirical Software Engineering*, Rio de Janeiro, Brazil: 21-22 September, 2006.

Cannon, W. B. (1927). "The James-Lange theory of emotions: A critical examination and an alternative theory." *JSTOR*, 39(1/4), 106–124.

Causevic, A., Sundmark, D., and Punnekkat, S. (2011). "Factors Limiting Industrial Adoption of Test Driven Development: A Systematic Review.", *Proceeding of The 4th IEEE International Conference on Software Testing, Verification and Validation*, Berlin, Germany: 21-25 March, 2011.

Charters, W., and Good, C. V. (1945). *The Dictionary of Education,* Oxford University Press, United Kingdom.

Deci, E. L. (1992). *The relation of interest to the motivation of behavior: A self-determination theory perspective,* Lawrence Erlbaum Associates, Inc.

Deci, E., and Ryan, R. (1985). *Intrinsic Motivation and Self-Determination in Human Behavior,* Plenum Press, New York.

Denzin, N. K., and Lincoln, Y. S. (1994). *Handbook of qualitative research.* Sage Publications, Inc.,California.

Desai, C., Janzen, D., and Savage, K. (2008). "A survey of evidence for test-driven development in academia." *ACM SIGCSE Bulletin*, 40(2), 97–101.

Dewey, J. (1913). *Interest and effort in education*, Houghton Mifflin, Michigan.

Dewey, J. (1997). *How we think*, D. C. Heath & Co., Boston.

Dewey, J. (2007). *Experience and education*, Simon and Schuster,New York.

Dwyer, K. K. (1993). "Using the 4MAT System Learning Styles Model to Teach Persuasive Speaking in the Basic Speech Course." *Discover Report*, 15(8), 14-18.

Edwards, S. H. (2003). "Using Test-Driven Development in the Classroom: Providing Students with Automatic, Concrete Feedback on Performance.", *Proceeding of International Conference on Education and Information Systems: Technology and Applications,* Orlando, Florida: 21 November, 2003.

Fenton, S. (2014). *Pro TypeScript*. Apress, New York.

Fox, R. J. (1983). *Confirmatory factor analysis*, Wiley Online Library, United Kingdom.

Fraser, S., Astels, D., Beck, K., Boehm, B., McGregor, J., Newkirk, J., and Poole, C. (2003). "Discipline and Practices of TDD:(Test Driven Development).", *Proceeding of The 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, Anaheim, CA, USA: 26-30 October, 2003.

Fucci, D., and Turhan, B. (2014). "On the role of tests in test-driven development: A differentiated and partial replication." *Empirical Software Engineering*, Springer, 19(2), 277–302.

George, B., and Williams, L. (2003). "An Initial Investigation of Test Driven Development in Industry.", *Proceeding of The 2003 ACM Symposium on Applied Computing*, Melbourne, Florida: 09-12 March, 2003.

George, B., and Williams, L. (2004). "A structured experiment of test-driven development." *Information and software Technology*, Elsevier, 46(5), 337–342.

Graziotin, D., Wang, X., and Abrahamsson, P. (2014). "Happy software developers solve problems better: psychological measurements in empirical software engineering." *PeerJ*, PeerJ Inc., 2, e289.

Graziotin, D., Wang, X., and Abrahamsson, P. (2015). "Understanding the Affect of Developers: Theoretical Background and Guidelines for Psychoempirical Software Engineering.", *Proceeding of The 7th International Workshop on Social Software Engineering*, Bergamo, Italy: 1 September, 2015.

Herbart, J. F. (1806). *Allgemeine Pädagogik: aus dem Zweck der Erziehung abgeleitet*. Siegismund & Volkening, German.

Izard, C. E. (1991). *The psychology of emotions*. Springer Science & Business Media, London.

James, W. (1884). "What is an emotion?" *JSTOR*, 9(34), 188–205.

James, W. (2013). *The principles of psychology*. Read Books Ltd, United States.

Janzen, D. S. (2005). "Software Architecture Improvement through Test-Driven Development.", *Proceeding of The 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, San Diego, CA, USA: 16-20 October, 2005.

Janzen, D., and Saiedian, H. (2005). "Test-driven development concepts, taxonomy, and future direction." *Computer, IEEE*, 38(9), 43–50.

Janzen, D., and Saiedian, H. (2008). "Test-Driven Learning in Early Programming Courses.", *Proceeding of The 39th SIGCSE Technical Symposium on Computer Science Education*, Portland, OR, USA: 12-15 March, 2008.

Kaufmann, R., and Janzen, D. (2003). "Implications of Test-Driven Development: A Pilot Study.", *Proceeding of The 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, Anaheim, CA, USA: 26-30 October, 2003.

Khan, I. A., Brinkman, W.-P., and Hierons, R. M. (2011). "Do moods affect programmers' debug performance?" *Cognition, Technology & Work*, Springer, 13(4), 245–258.

Koelstra, S., Muhl, C., Soleymani, M., Lee, J.-S., Yazdani, A., Ebrahimi, T., Pun, T., Nijholt, A., and Patras, I. (2012). "Deap: A database for emotion analysis; using physiological signals." *IEEE Transactions on Affective Computing*, 3(1), 18–31.

Krapp, A. (1999). "Interest, motivation and learning: An educational-psychological perspective." *European Journal of Psychology of Education*, 14(1), 23–40.

Lesiuk, T. (2005). “The effect of music listening on work performance.” *Psychology of Music*, Sage Publications Sage CA: Thousand Oaks, CA, 33(2), 173–191.

Madeyski, L. (2009). *Test-driven development: An empirical evaluation of agile practice*. Springer Science & Business Media, New York.

Mäkinen, S., and Münch, J. (2014). “Effects of Test-Driven Development: A Comparative Analysis of Empirical Studies.”, *Proceeding of International Conference on Software Quality*, Vienna, Austria: 14-16 January, 2014.

Matharu, G. S., Mishra, A., Singh, H., and Upadhyay, P. (2015). “Empirical study of agile software development methodologies: A comparative analysis.” *ACM SIGSOFT Software Engineering Notes*, 40(1), 1–6.

Mukherji, P., and Albon, D. (2014). *Research methods in early childhood: An introductory guide*. SAGE Publication Ltd., London.

Müller, M. M., and Höfer, A. (2007). “The effect of experience on the test-driven development process.” *Empirical Software Engineering*, 12(6), 593–615.

Munir, H., Wnuk, K., Petersen, K., and Moayyed, M. (2014). “An Experimental Evaluation of Test Driven Development vs. Test-Last Development with Industry Professionals.” *Proceeding of The 18th International Conference on Evaluation and Assessment in Software Engineering*, London, England, United Kingdom: 13-14 May, 2014.

Ortony, A., and Turner, T. J. (1990). “What’s basic about basic emotions?” *Psychological review*, 97(3), 315.

Pancur, M., Ciglaric, M., Trampus, M., and Vidmar, T. (2003). “Towards Empirical Evaluation of Test-Driven Development in a University Environment.”, *Proceeding of IEEE Region 8 EUROCON 2003, Computer as a Tool,* Ljubljana, Slovenia, Slovenia: 22-24 September, 2003.

Panksepp, J. (1982). “Toward a general psychobiological theory of emotions.” *Cambridge Univ Press*, 5(3), 407–422.

Plutchik, R. (1984). “Emotions: A general psychoevolutionary theory.” *Approaches to emotion*, 1984, 197–219.

Royce, W. W. (1987). “Managing The Development of Large Software Systems: Concepts and Techniques.” *Proceeding of The 9th International Conference on Software Engineering*, California, USA: 30 March – 2 April, 1987.

Schachter, S., and Singer, J. (1962). “Cognitive, social, and physiological determinants of emotional state.” *American Psychological Association*, 69(5), 379.

Schiefele, U. (1991). “Interest, learning, and motivation.” *Educational psychologist*, 26 (3-4), 299–323.

Seaman, C. B. (1999). “Qualitative methods in empirical studies of software engineering.” *IEEE Transactions on Software Engineering*, 25(4), 557–572.

Van Teijlingen, E., and Hundley, V. (2002). “The importance of pilot studies.” *Nursing Standard*, 16(40), 33–36.

Wrobel, M. R. (2013). “Emotions in The Software Development Process.”, *Proceeding of The 6th International Conference on Human System Interaction,* Sopot, Poland: 6-8 June, 2013.

Wu, M.-W., and Lin, Y.-D. (2001). “Open source software development: an overview.” *Computer*, 34(6), 33–38.

Yin, R. K. (2013). *Case study research: Design and methods*. SAGE publications Ltd., New York.

Yong, A. G., and Pearce, S. (2013). “A beginner’s guide to factor analysis: Focusing on exploratory factor analysis.” *Tutorials in Quantitative Methods for Psychology*, 9(2), 79–94.

ภาคผนวก

ภาคผนวก ก

แบบสอบถามก่อนและหลังการทดลอง

[Group:]

แบบสอบถามเพื่อใช้ในการเก็บข้อมูลของนักศึกษา

เพื่อเป็นการศึกษาและเก็บข้อมูลนักศึกษาที่ให้ความสนใจในการเป็นนักพัฒนาซอฟต์แวร์

ข้อมูลนักศึกษา

ชื่อ – นามสกุล_____รหัสนักศึกษา_____

นักศึกษาชั้นปีที่ ☐ ชั้นปีที่ 1 ☐ ชั้นปีที่ 2 ☐ ชั้นปีที่ 3 ☐ ชั้นปีที่ 4 ☐ ชั้นปีที่ 5

เกรดในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ (OOP)_____

โครงการเทอมในรายวิชาการโปรแกรมเชิงอ็อบเจกต์ (OOP) เกี่ยวกับอะไร_____

_____

คำถามพื้นฐาน

1. เหตุผลที่นักศึกษาเข้าศึกษาในสาขาวิศวกรรมซอฟต์แวร์

   _____

   _____

   _____

2. อาชีพที่นักศึกษาคาดว่าหลังจบการศึกษาจะเข้าทำงาน

   _____

   _____

   _____

3. ความเข้าใจของนักศึกษาของคำว่า “นักพัฒนาซอฟต์แวร์”

   _____

   _____

   _____

[Group:]

4. นักศึกษามีความสนใจในด้านการพัฒนาซอฟต์แวร์หรือไม่ เพราะเหตุใด (ความสนใจ เป็นความรู้สึกถึงความพึงพอใจ เป็นความอยากรู้อยากเห็น ความแสวงหาความรู้สึกชอบความพอใจของบุคคลต่อกิจกรรมนั้น ซึ่งทำให้เพียรพยายามและสามารถกระทำสิ่งการจนบรรลุจุดมุ่งหมาย)

_____

_____

_____

5. กระบวนการที่นักศึกษาใช้ในการทำโครงการในรายวิชาโปรแกรมเชิงอ็อบเจกต์ (OOP) ตั้งแต่เริ่มต้นการทำโครงการจนถึงการนำเสนอโครงการ

_____

_____

_____

_____

_____

6. ภาษาโปรแกรมมิ่งที่นักศึกษามีความสนใจ (ตอบได้มากกว่า 1 ข้อ)

☐ ภาษา C ☐ ภาษา C++ ☐ ภาษา HTML ☐ ภาษา PHP ☐ ภาษา Java

☐ ภาษา C# ☐ ภาษาอื่นๆ (โปรดระบุ) .....

7. การทำโครงการพัฒนาซอฟต์แวร์ในรายวิชาที่ผ่านนักศึกษามีการทดสอบระบบอย่างไร

_____

_____

_____

8. นักศึกษาใช้เวลาเฉลี่ยเท่าใด ในการพัฒนาซอฟต์แวร์ตามแบบฝึกหัดที่ได้รับมอบหมายต่อครั้ง (เวลาเฉลี่ยต่อสัปดาห์)

_____

9. นักศึกษาใช้เวลาเฉลี่ยเท่าใดในการศึกษาเพิ่มเติมด้วยตนเอง (เวลาเฉลี่ยต่อสัปดาห์)

_____

[Group:]

10. นักศึกษามักจะทำการศึกษาเรื่องใดเมื่อศึกษาเพิ่มเติมด้วยตนเอง

_____

_____

_____

11. จากการศึกษาที่ผ่านมานักศึกษาให้ความสำคัญกับวิชาใดบ้างในการเรียนในสาขาวิศวกรรมซอฟต์แวร์ (ความสำคัญ เป็นวิชาที่นักศึกษาคิดว่ามีผลต่อการศึกษาต่อในรายวิชาหรือจำเป็นต้องใช้งานในงานด้านการพัฒนาซอฟต์แวร์)

☐ Computer in Daily Life ☐ Computer and Programming Concept

☐ Principles of Computer Architecture ☐ Data Structures and Algorithms

☐ Object-Oriented Programming ☐ Object-Oriented Analysis and Design

☐ Advanced-Object-Oriented Programming ☐ Web Programming

12. จากการศึกษาที่ผ่านมานักศึกษามีความสนใจในวิชาใดบ้างในการเรียนในสาขาวิศวกรรมซอฟต์แวร์ (ความสนใจ เป็นความรู้สึกพึงพอใจในการเรียนวิชาดังกล่าว มีความรู้สึกอยากเรียนรู้เพิ่มเติมจากงานหรือมีการศึกษาเพิ่มเติมด้วยตนเอง)

☐ Computer in Daily Life ☐ Computer and Programming Concept

☐ Principles of Computer Architecture ☐ Data Structures and Algorithms

☐ Object-Oriented Programming ☐ Object-Oriented Analysis and Design

☐ Advanced-Object-Oriented Programming ☐ Web Programming

13. หากมีการจัดอบรมทั้งในเชิงบรรยายหรือเชิงปฏิบัติการนักศึกษาอยากให้มีการจัดในหัวข้อใด เพราะเหตุใด

_____

_____

_____

[Group:]

ความสนใจของนักศึกษาเกี่ยวกับการเรียนด้านการพัฒนาซอฟต์แวร์

คำชี้แจง กรุณาให้คะแนนความสนใจของท่านโดยการทำเครื่องหมาย ✓ลงในช่องระดับความสนใจ 5 ระดับต่อไปนี้ (1 = ให้ความสนใจน้อยที่สุด, 2 = ให้ความสนใจน้อย, 3 = ให้ความสนใจปานกลาง, 4 = ให้ความสนใจมาก, 5 = ให้ความสนใจมากที่สุด)

หัวข้อในการทำแบบสอบถาม	ระดับความสนใจ				
	5	4	3	2	1
1. นักศึกษาให้ความสนใจในการเรียนในรายวิชาด้านการพัฒนาซอฟต์แวร์					
2. นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากแบบฝึกหัด (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)					
3. นักศึกษาให้ความสนใจในการฝึกฝนการพัฒนาซอฟต์แวร์จากการศึกษาหาข้อมูลด้วยตนเอง (การฝึกฝนการพัฒนาซอฟต์แวร์เป็นการฝึกฝนด้านการเขียนโปรแกรม)					
4. นักศึกษาให้ความสนใจในการศึกษาเกี่ยวกับวิศวกรรมซอฟต์แวร์เพิ่มเติมด้วยตนเอง (การศึกษาเพิ่มเติมด้านวิศวกรรมซอฟต์แวร์ เช่น การศึกษาเพิ่มเติมถึงกระบวนการพัฒนาซอฟต์แวร์รูปแบบใหม่ ภาษาในการเขียนโปรแกรม เป็นต้น)					
5. นักศึกษาให้ความสนใจในการใช้เวลาในการทำแบบฝึกหัด					
6. นักศึกษามีความสนใจในการดัดแปลงการแสดงผลจากแบบฝึกหัดที่ได้รับมอบหมาย					
7. นักศึกษาให้ความสนใจในการใช้เวลาในการศึกษาเพิ่มเติมด้านซอฟต์แวร์ด้วยตนเอง					
8. นักศึกษามีให้ความสนใจในการมีส่วนร่วมในการถาม-ตอบภายในห้องเรียน					

[Group:]

คำถามเพิ่มเติม

1. การใช้ Test-Driven Development ในการเรียนการสอนมีผลต่อนักศึกษาในการให้ความสนใจในการพัฒนาซอฟต์แวร์หรือไม่ เพราะเหตุใด

   _____

   _____

   _____

2. การใช้ Test-Driven Development ส่งผลต่อการนักศึกษาอย่างไร ในการเขียนโปรแกรมในห้องปฏิบัติการ

   _____

   _____

   _____

3. จากการทำ Test-Driven-Development ในห้องปฏิบัติการหากให้โจทย์เดียวกันแต่ให้นักศึกษาใช้วิธีการแบบจำลองน้ำตก (Waterfall Model) ในความคิดเห็นของนักศึกษาคิดว่าวิธีการใดทำให้นักศึกษาพบข้อผิดพลาดในโปรแกรมน้อยลง เพราะเหตุใด

   _____

   _____

   _____

4. ในอนาคตหากมีการพัฒนาโครงการในรายวิชา นักศึกษาจะเลือกใช้วิธีการทดสอบแบบใดในการทดสอบโปรแกรมที่นักศึกษาทำการพัฒนา

   _____

   _____

   _____

5. ในการทำการทดสอบโปรแกรม นักศึกษามีความคิดเห็นว่าวิธีการทดสอบแบบใดที่สามารถทำให้การทดสอบมีความละเอียดมากขึ้น เพราะเหตุใด

   _____

   _____

   _____

[Group:]

6. ในมุมมองของนักศึกษาคิดว่า Test-Driven Development มีประโยชน์ต่อการพัฒนาซอฟต์แวร์หรือไม่ เพราะเหตุใด

_____

_____

_____

ภาคผนวก ข

แบบสังเกตการณ์นักศึกษาในการทดลอง

การเก็บข้อมูลในการเรียนเชิงปฏิบัติการ (Lab)

กลุ่มการเข้าใช้ห้องปฏิบัติการ.....เวลาเฉลี่ยต่อคน.....

แบบสังเกตการณ์เพื่อใช้ในการเก็บข้อมูลของนักศึกษา

เพื่อเป็นการศึกษาและเก็บข้อมูลนักศึกษาที่ให้ความสนใจในการเป็นนักพัฒนาซอฟต์แวร์

ข้อมูลในการสังเกตการณ์

วันในการสังเกตการณ์วันที่.....เดือน.....พ.ศ.....เวลา.....

กลุ่มนักศึกษาเข้าเรียนห้องปฏิบัติการ☐ กลุ่มที่ 1 นักศึกษา.....คน ☐ กลุ่มที่ 2 จำนวน.....คน

การทำการทดลองครั้งที่.....แบบฝึกหัดที่ได้รับมอบหมายในห้องปฏิบัติการ.....

ข้อมูลนักศึกษา

ชื่อ – นามสกุล.....รหัสนักศึกษา.....โต๊ะที่นั่งในห้องปฏิบัติการ.....

นักศึกษาชั้นปีที่ ☐ ชั้นปีที่ 1 ☐ ชั้นปีที่ 2 ☐ ชั้นปีที่ 3 ☐ ชั้นปีที่ 4 ☐ ชั้นปีที่ 5

รูปแบบการพัฒนาซอฟต์แวร์ ☐ Waterfall Model ☐ Test-Driven Development

เวลาเริ่มต้นการทำแบบฝึกหัด.....เวลาที่ส่งแบบฝึกหัด.....

ข้อมูลการสังเกตการณ์ภายในห้องปฏิบัติการ

1. ด้านการแสดงออกทางกายภาพ

_____

_____

_____

_____

_____

_____

_____

_____

การเก็บข้อมูลในการเรียนเชิงปฏิบัติการ (Lab)

กลุ่มการเข้าใช้ห้องปฏิบัติการ.....เวลาเฉลี่ยต่อคน.....

2. ด้านการโปรแกรมมิ่งตามแบบฝึกหัดที่ได้รับมอบหมาย

1.1. การเขียนโปรแกรม

_____

_____

_____

_____

_____

_____

1.2. ด้านการสืบค้นข้อมูล

_____

_____

_____

_____

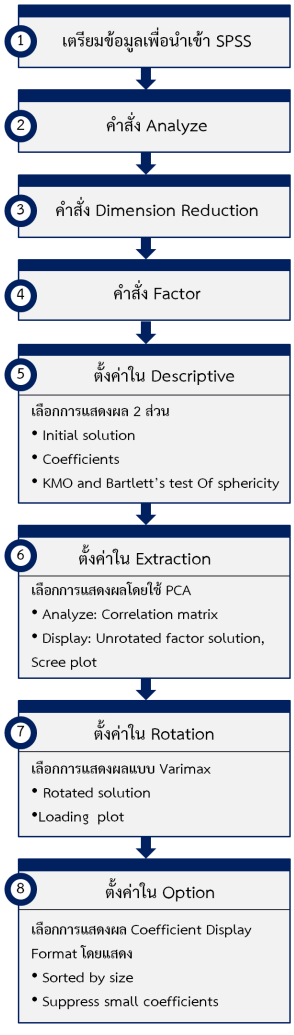
_____

_____

ภาคผนวก ค

วิธีการวิเคราะห์ผลและประเมินผลโดยใช้การวิเคราะห์องค์ประกอบ

สำหรับการวิเคราะห์องค์ประกอบโดยใช้เครื่องมือในการวิเคราะห์ด้วยโปรแกรม SPSS[4] เวอร์ชัน 22.0 (Denzin and Lincoln, 1994) เป็นโปรแกรมที่ใช้ในการวิเคราะห์ข้อมูลทางสถิติ โดยการวิเคราะห์องค์ประกอบเชิงยืนยันในงานวิจัยมีขั้นตอนในการทำการวิเคราะห์ ดังรูปภาคผนวกที่ 1 ซึ่งมีทั้งหมด 2 ขั้นตอนหลัก คือ



รูปภาคผนวกที่ 1 ขั้นตอนในการวิเคราะห์องค์ประกอบด้วยโปรแกรม SPSS

[4] โปรแกรม IBM SPSS Statistics Base (SPSS) เป็นเครื่องมือในการวิเคราะห์ข้อมูลทางสถิติ

ขั้นตอนที่ 1 เตรียมข้อมูลเพื่อนำเข้าโปรแกรม SPSS

ขั้นตอนที่ 2 ทำการวิเคราะห์องค์ประกอบโดยใช้คำสั่ง Analyze แล้วเลือกคำสั่ง Dimension Reduction และเลือกคำสั่ง Factor โดยในคำสั่ง Factor ทำการให้ตั้งค่าการแสดงผลลัพธ์ทั้งหมด 4 ส่วน คือ

(1) ตั้งค่าใน Descriptive ให้เลือกแสดงผล 3 ส่วน คือ Initial solution, Coefficient และ KMO and Bartlett's test of sphericity

(2) ตั้งค่าใน Extraction ให้เลือกการแสดงผลโดยใช้วิธีการ PCA (Principle Component Analysis) แล้วเลือกให้แสดงผลค่า Correlation matrix, Unrotated factor solution และ Scree plot ซึ่งการตั้งค่าในส่วนนี้ต้องเลือกการกำหนดวิธีการวิเคราะห์องค์ประกอบที่มี 2 รูปแบบ คือ หากต้องการวิเคราะห์องค์ประกอบเชิงสำรวจให้กำหนดใช้ Based on Eigenvalue ให้ตั้งค่า Eigenvalue > 1 หากต้องทำการวิเคราะห์องค์ประกอบเชิงยืนยันให้กำหนดโดยใช้ Fixed number of factors โดยตั้งค่า Factors to extract เป็นจำนวนองค์ประกอบตามที่ต้องการ

(3) ตั้งค่าใน Rotation ให้เลือกการแสดงผลโดยใช้วิธีการ Varimax แล้วให้เลือกแสดงผลค่า Rotated solution และ Loading plot

(4) ตั้งค่าใน Option ให้เลือกการแสดงผลโดยให้ค่าเรียงลำดับแบบ Sorted by size และ Suppress small coefficients

เมื่อตั้งค่าการแสดงผลแล้วจะได้ผลลัพธ์ออกมา แล้วจึงทำการพิจารณาค่าว่าจากข้อมูลที่นำมาวิเคราะห์ด้วยวิธีการวิเคราะห์องค์ประกอบ ได้ผลลัพธ์ที่ผ่านการพิจารณาหรือไม่ โดยค่าที่ต้องทำการพิจารณามีทั้งหมด 5 ส่วน (Yong and Pearce, 2013) ดังรูปภาคผนวกที่ 2 คือ

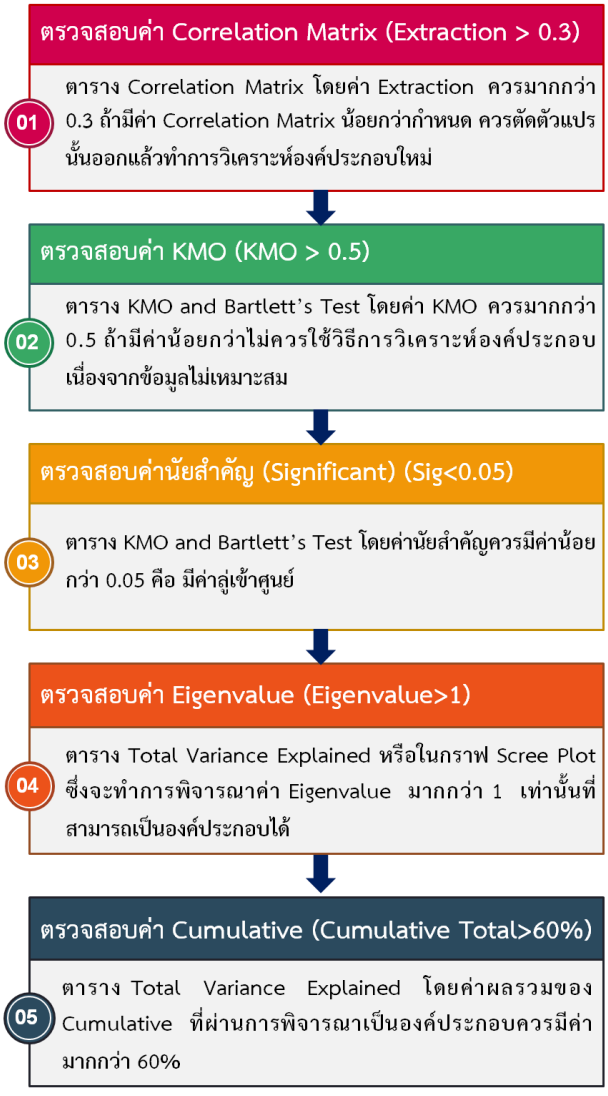
(1) ตรวจสอบค่า Extraction ในตาราง Correlation Matrix โดยค่า Extraction ควรมากกว่า 0.3 ถ้าตัวแปรใดมีค่า Extraction น้อยกว่า 0.3 ควรตัดตัวแปรนั้นออกจากการวิเคราะห์แล้วทำการวิเคราะห์องค์ประกอบใหม่

(2) ตรวจสอบค่า KMO ในตาราง KMO and Bartlett's Test โดยค่า KMO ควรมากกว่า 0.5 ถ้ามีค่า KMO น้อยกว่า 0.5 ไม่ควรใช้วิธีการวิเคราะห์องค์ประกอบเนื่องจากข้อมูลไม่เหมาะสม

(3) ตรวจสอบค่านัยสำคัญ (Significant) ในตาราง KMO and Bartlett's Test โดยค่านัยสำคัญควรมีค่าน้อยกว่า 0.05 คือ มีค่าลู่เข้า 0

(4) ตรวจสอบค่า Eigenvalue ในตาราง Total Variance Explained หรือในกราฟ Scree Plot โดยจะทำการพิจารณาค่า Eigenvalue มากกว่า 1 เท่านั้นที่สามารถจัดเป็นองค์ประกอบได้

(5) ตรวจสอบค่า Cumulative ในตาราง Total Variance Explained โดยค่าผลรวมของ Cumulative ที่มีค่า Eigenvalue มากกว่า 1 ควรมีค่ามากกว่า 60%



รูปภาคผนวกที่ 2 ขั้นตอนในการพิจารณาค่าจากการวิเคราะห์องค์ประกอบ

ประวัติผู้เขียน

ชื่อ-สกุล นางสาวสุชาดา พงศ์พรม

รหัสประจำตัวนักศึกษา 5830223013

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วิทยาศาสตรบัณฑิต (วิศวกรรมซอฟต์แวร์)	มหาวิทยาลัยสงขลานครินทร์	2558

การตีพิมพ์เผยแพร่ผลงาน

สุชาดา พงศ์พรม, อชีส นันทอมรพงศ์, และ รัตนา เวทประสิทธิ์. (2559). “การกระตุ้นความสนใจของนักศึกษาในการพัฒนาซอฟต์แวร์โดย Test-Driven Development”, งานประชุมวิชาการระดับชาติด้านคอมพิวเตอร์และเทคโนโลยีสารสนเทศ ครั้งที่ 12, โรงแรมอวานี ขอนแก่น: 7-8 กรกฎาคม 2559.