# การสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความ

วิทยานิพนธ์

ของ

คณิน งามมานะ

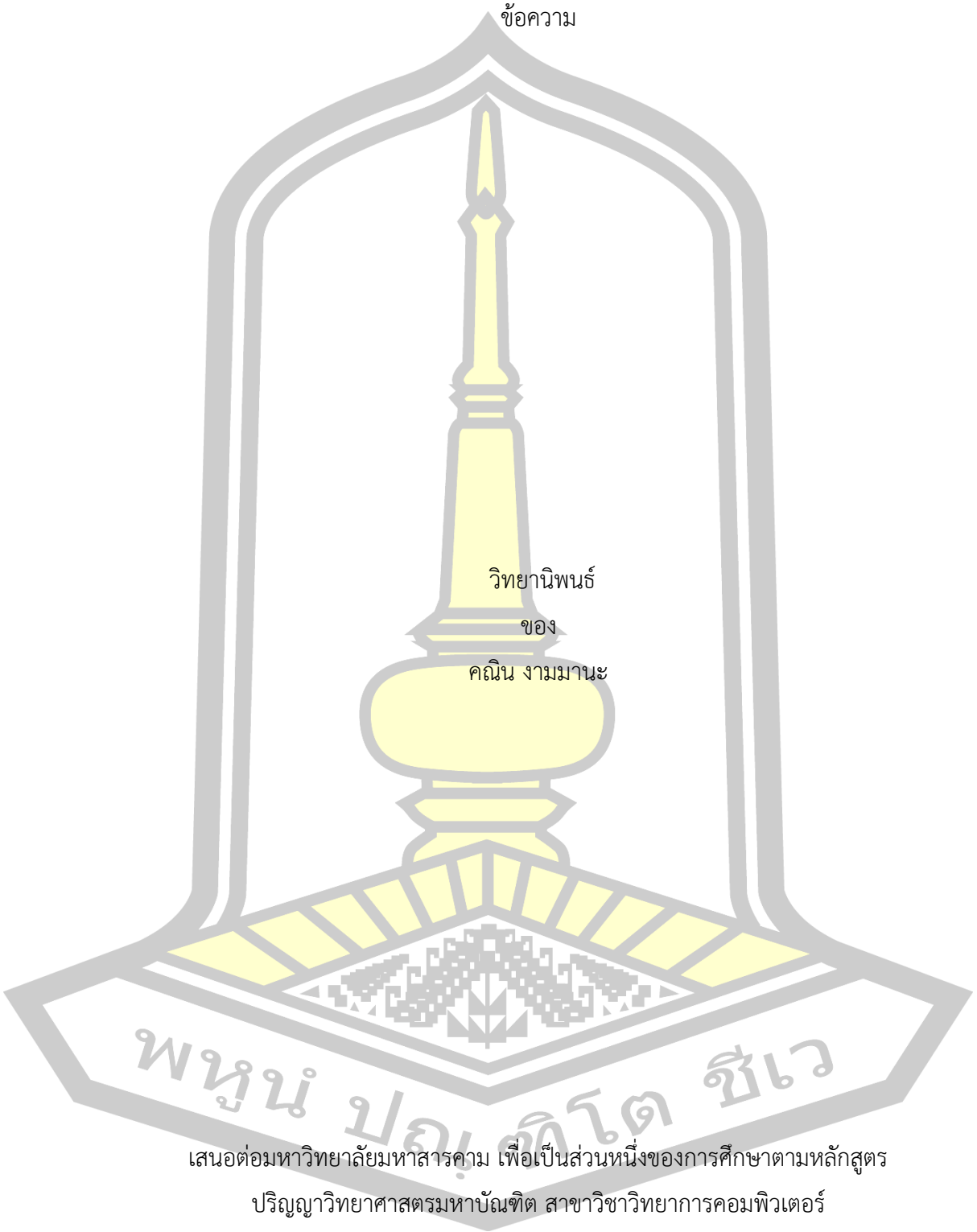
เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

มิถุนายน 2562

สงวนลิขสิทธิ์เป็นของมหาวิทยาลัยมหาสารคาม

# การสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความ



วิทยานิพนธ์

ของ

คณิน งามมานะ

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

มิถุนายน 2562

สงวนลิขสิทธิ์เป็นของมหาวิทยาลัยมหาสารคาม

# Automatic Software Process Modelling from Textual Software Requirements

Kanin Ngammana

A Thesis Submitted in Partial Fulfillment of Requirements

for Master of Science (Computer Science)

June 2019

Copyright of Mahasarakham University

คณะกรรมการสอบวิทยานิพนธ์ ได้พิจารณาวิทยานิพนธ์ของนายคณิน งามมานะ แล้ว เห็นสมควรรับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ของมหาวิทยาลัยมหาสารคาม

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(ผศ. ดร. ธัชพงศ์ กตัญญูกุล )

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(ผศ. ดร. จันทิมา พลพินิจ )

.....กรรมการ
(ผศ. ดร. พนิดา ทรงรัมย์ )

.....กรรมการ
(ผศ. ดร. รพีพร ช่ำชอง )

มหาวิทยาลัยอนุมัติให้รับวิทยานิพนธ์ฉบับนี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ของมหาวิทยาลัยมหาสารคาม

.....
(ผศ. ศศิธร แก้วมั่น )
คณบดีคณะวิทยาการสารสนเทศ

.....
(ผศ. ดร. กริสน์ ชัยมูล )
คณบดีบัณฑิตวิทยาลัย

พหุน ปญฺ ฑิโต ชีเว

**ชื่อเรื่อง** การสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความ

**ผู้วิจัย** คณิน งามมานะ

**อาจารย์ที่ปรึกษา** ผู้ช่วยศาสตราจารย์ ดร. จันทิมา พลพินิจ

**ปริญญา** วิทยาศาสตรมหาบัณฑิต **สาขาวิชา** วิทยาการคอมพิวเตอร์

**มหาวิทยาลัย** มหาวิทยาลัยมหาสารคาม **ปีที่พิมพ์** 2562

## บทคัดย่อ

การระบุข้อกำหนดซอฟต์แวร์อัตโนมัติ (Software Requirements Specification: SRS) จากเอกสารความต้องการซอฟต์แวร์เป็นปัญหาอย่างหนึ่งในวิศวกรรมความต้องการ (Requirement Engineering) เนื่องจากว่างานในรูปแบบนี้เป็นงานที่ต้องใช้เวลานานและใช้แรงงานอย่างมากในขั้นตอนการวิเคราะห์และออกแบบระบบ โดยเฉพาะถ้าหากเป็นการทำงานด้วยมือกับเอกสารความต้องการของระบบซอฟต์แวร์ขนาดใหญ่ ดังนั้น ในงานวิจัยฉบับนี้จึงมุ่งเน้นในการนำเสนอกระบวนการในการสกัดซีนาริโอจากเอกสารความต้องการซอฟต์แวร์ที่เป็นภาษาอังกฤษแบบอัตโนมัติ เมื่อซีนาริโอในที่นี้คือลำดับการกระทำ (Action) ของแต่ละผู้กระทำ (Actor) ในซอฟต์แวร์หนึ่งๆ ซึ่งหลังจากสกัดผู้กระทำและการกระทำได้แล้วจะถูกจำลองเป็นกระบวนการของซอฟต์แวร์ (Software process) โดยเครื่องมือหลักที่ใช้ในกระบวนการที่นำเสนอคือเทคนิคของการประมวลผลภาษาธรรมชาติ (Natural Language Processing Techniques) หลังจากทดสอบการสกัดผู้กระทำจากเอกสารความต้องการซอฟต์แวร์แบบอัตโนมัติด้วยค่าความระลึก ค่าความแม่นยำ และค่าเอฟปรากฏว่าให้ผลลัพธ์ที่ 1.00, 0.67 และ 0.8 ตามลำดับ ในขณะที่การทดสอบการสกัดการกระทำของแต่ละผู้กระทำจากเอกสารความต้องการซอฟต์แวร์แบบอัตโนมัติพบว่าให้ค่าความระลึกที่ 0.88 แม้ว่าผลลัพธ์ที่ได้ในเทอมของค่าความแม่นยำจะค่อนข้างต่ำ แต่ก็ให้ผลลัพธ์ในเทอมของค่าความระลึกที่น่าพอใจ ด้วยการทำเช่นนี้ ก็น่าจะช่วยให้เกิดความประหยัดในการพัฒนาซอฟต์แวร์ของแต่ละองค์กร โดยเฉพาะในการพัฒนาซอฟต์แวร์จากความต้องการระบบขนาดใหญ่และซับซ้อน

คำสำคัญ : ความต้องการของซอฟต์แวร์ที่เป็นข้อความ, ซีนาริโอ, การประมวลผลภาษาธรรมชาติ, ผู้กระทำ, การกระทำ, กระบวนการของซอฟต์แวร์

**TITLE** Automatic Software Process Modelling from Textual Software Requirements

**AUTHOR** Kanin Ngammana

**ADVISORS** Assistant Professor Jantima Polpinij , Ph.D.

**DEGREE** Master of Science **MAJOR** Computer Science

**UNIVERSITY** Mahasarakham University **YEAR** 2019

## ABSTRACT

An automatically finding of software requirements specification (SRS) from software requirements is a problem in requirement engineering. This is because this task becomes a time-consuming and labor-intensive in system analysis and design stage, when it is manually working on a large scale of textual software requirements. Therefore, this work aims to present a method of automatically extracting scenarios from textual software requirements written with English, where a scenario is a narrative of actions of each user (called actor) in a software. After extracting “actor” and its “actions” from textual software requirements, those will be modelled as software process. The main mechanisms of the proposed method are natural language processing (NLP) techniques. After testing actor extraction by recall, precision, and F-measure, the results are 1.00, 0.67 and 0.8 respectively. Meanwhile, extracting of actor’s actions evaluated by recall, reveals the recall score of 0.88. Although the result in term of precision score is quite poor, the proposed method may still return the satisfactory results in term of recall. By doing so, it may provide a substantial saving for major software development of an organization, especially in developing a software from large and complex software requirements.

Keyword : Textual Software Requirements, Scenarios, Natural Language Processing, Actor, Action, Softwre Process

# กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จสมบูรณ์ได้ด้วยความกรุณาและความช่วยเหลืออย่างสูงยิ่งจาก ผศ. ดร.จันทิมา พลพินิจ ประธานกรรมการควบคุมวิทยานิพนธ์ ผศ. ดร.ธัชพงศ์ กตัญญูกุล ประธานกรรมการสอบ และ ผศ. ดร.พนิดา ทรงรัมย์ และ ผศ. ดร.รพีพร ชำชอง กรรมการสอบ

ขอขอบพระคุณบิดา มารดา เพื่อนร่วมงาน นิสิตระดับปริญญาตรี และระดับบัณฑิตศึกษา ที่ให้คำปรึกษาและความช่วยเหลือ จนทำให้งานวิทยานิพนธ์ฉบับนี้เสร็จสมบูรณ์

คณิน งามมานะ

พหูน ปณฺฑิโต ชีเว

# สารบัญ

หน้า


บทคัดย่อภาษาไทย ..... ง

บทคัดย่อภาษาอังกฤษ ..... จ

กิตติกรรมประกาศ ..... ฉ

สารบัญ ..... ช

สารบัญตาราง ..... ฎ

สารบัญภาพประกอบ ..... ฎ

บทที่ 1 บทนำ ..... 15

1.1 หลักการและเหตุผล ..... 15

1.2 วัตถุประสงค์ของการวิจัย ..... 17

1.3 ขอบเขตของงานวิจัย ..... 17

1.4 ความสำคัญของงานวิจัย ..... 18

1.5 นิยามศัพท์เฉพาะ ..... 19

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง ..... 20

2.1 แบบจำลองกระบวนการของซอฟต์แวร์ ..... 20

2.1 ซีนาริโอ ..... 21

2.2.1 ความหมายของซีนาริโอ ..... 21

2.2.2 รูปแบบการนำเสนอซีนาริโอจากความต้องการซอฟต์แวร์ ..... 23

2.2.3 ภาษาและโครงสร้างประโยคของเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 23

2.3 เทคนิคการประมวลผลภาษาธรรมชาติ ..... 25

2.3.1 POS Tagging ..... 25

2.3.2 แบบจำลองมาร์คอฟและเอ็นแกรม ..... 27

2.3.3 แบบจำลองมาร์คอฟซ่อนเร้น ..... 30
2.3.4 การแจกแจงรูปประโยค ..... 31
2.3.5 การลดรูปคำ ..... 32
2.4 การแทนองค์ความรู้ ..... 37
2.5 การประเมินประสิทธิภาพ ..... 39
2.7.1 ค่าความแม่นยำ ..... 40
2.7.2 ค่าความระลึก ..... 40
2.7.3 ค่า F-measure ..... 40
2.6 แผนภาพที่แสดงการทำงานของผู้ใช้ระบบ ..... 40
2.7 งานวิจัยที่เกี่ยวข้อง ..... 41
บทที่ 3 วิธีดำเนินการวิจัย ..... 45
3.1 ชุดข้อมูล ..... 45
3.2 กรอบการดำเนินการ ..... 46
3.3 การสร้างกฎ (Rules Modelling) ..... 46
3.3.1 กฎในการสกัด Actor ..... 47
3.3.2 กฎในการแทนคำสรรพนาม ..... 52
3.3.3 กฎในการแยกประโยคความรวม ..... 54
3.4 การสกัด Actor และ Action (Extracting of Actors and Actions) ..... 60
3.4.1 การเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ (Textual Software Requirement Preparation) ..... 61
3.4.2 การสกัด Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ด้วยกฎในการสกัด Actor (Actor Extraction from Textual Software Requirements by Rules) ..... 62
3.4.3 การสกัด Action ของแต่ละ Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ (Actors' Action Extraction from Textual Software Requirements) ..... 64

3.5 การสร้างแบบจำลองกระบวนการของซอฟต์แวร์ ..... 67
บทที่ 4 ผลการวิจัย ..... 68
4.1 ชุดข้อมูลที่ใช้ในงานวิจัย (Dataset) ..... 68
4.1.1 ชุดข้อมูลที่ใช้ในการประเมินกฎในการสกัด Actor ..... 68
4.1.2 ชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 68
4.2 การประเมินกระบวนการทำความสะอาดประโยค ..... 69
4.3 การประเมินกฎในการสกัด Actor ..... 71
4.4 การประเมินผลการสกัด Actor ..... 74
4.5 การประเมินซีนาริโอที่สกัดได้ ..... 76
บทที่ 5 สรุปผล อภิปรายผล และข้อเสนอแนะ ..... 79
5.1 ความสำคัญและวัตถุประสงค์ของการวิจัย ..... 79
5.2 สรุปกระบวนการ ..... 79
5.3 สรุปผลและอภิปราย ..... 81
5.4 ปัญหาที่พบ ..... 83
5.5 ข้อเสนอแนะ ..... 83
บรรณานุกรม ..... 84
ภาคผนวก ..... 88
ภาคผนวก ก ตัวอย่างชุดข้อมูล SNLI ..... 89
ภาคผนวก ข ตัวอย่างชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 92
ภาคผนวก ค ตัวอย่างผลเฉลยของชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 94
ภาคผนวก ง ตัวอย่างกฎที่ใช้ในกระบวนการสกัดซีนาริโอ ..... 96
ภาคผนวก จ ตัวอย่างแบบจำลองกระบวนการของซอฟต์แวร์ที่สร้างจากซีนาริโอที่สกัดได้ ..... 99
ประวัติผู้เขียน ..... 103

# สารบัญตาราง

หน้า


ตารางที่ 2.1 ตัวอย่างคำจำนวน 8 คำที่มีความถี่สูงในคลังข้อมูล ..... 29

ตารางที่ 2.2 ความถี่แบบ Bigram ของคำ 8 คำนี้ในคลังข้อมูลตัวอย่าง ..... 29

ตารางที่ 2.3 ตารางความคลาดเคลื่อน ..... 39

ตารางที่ 3.1 ความน่าจะเป็นในการเปลี่ยนสถานะของ *tag* ..... 51

ตารางที่ 3.2 กฎในการสกัด Actor ..... 52

ตารางที่ 3.3 ตัวอย่างกฎในการแทนคำสรรพนาม ..... 54

ตารางที่ 3.4 ความน่าจะเป็นในการเปลี่ยนสถานะของ *Object* ..... 59

ตารางที่ 3.5 ตัวอย่างกฎในการแยกประโยคความรวม ..... 60

ตารางที่ 4.1 ตัวอย่างผลลัพธ์การทำความสะอาดประโยคเมื่อเทียบกับผลเฉลย ..... 70

ตารางที่ 4.2 ตารางความคลาดเคลื่อนของผลลัพธ์การทำความสะอาดประโยค ..... 71

ตารางที่ 4.3 ผลลัพธ์การทำความสะอาดประโยค ..... 71

ตารางที่ 4.4 ตัวอย่างผลลัพธ์การประเมินกฎในการสกัด Actor เมื่อเทียบกับผลเฉลย ..... 72

ตารางที่ 4.5 ตารางความคลาดเคลื่อนของผลลัพธ์การสกัด Actor ..... 73

ตารางที่ 4.6 ผลการประเมินกฎในการสกัด Actor ..... 73

ตารางที่ 4.7 ตัวอย่าง Actor ทั้งหมดที่สกัดได้และเปรียบเทียบกับผลเฉลยด้วยผู้เชี่ยวชาญ ..... 75

ตารางที่ 4.8 ตารางความคลาดเคลื่อนของผลการสกัด Actor ..... 75

ตารางที่ 4.9 ผลการสกัด Actor หลังทำการลดรูปคำ ..... 76

ตารางที่ 4.10 ผลลัพธ์การสกัดซีนาริโอหรือ *Action* และ *Actor* ..... 77

ตารางที่ 4.11 ตารางความคลาดเคลื่อนของผลลัพธ์การสกัดซีนาริโอ ..... 78

ตารางที่ 4.12 ผลการประเมินซีนาริโอที่สกัดได้ ..... 78

# สารบัญภาพประกอบ

หน้า


ภาพที่ 2.1 ตัวอย่างแบบจำลองกระบวนการ ..... 20

ภาพที่ 2.2 ตัวอย่างข้อมูลกึ่งโครงสร้าง ..... 21

ภาพที่ 2.3ตัวอย่างซีเนริโอการจองห้องประชุม ..... 21

ภาพที่ 2.4 มุมมองเกี่ยวกับซีเนริโอ ..... 23

ภาพที่ 2.5 ตัวอย่างตำแหน่งของ Actor และ Action ในประโยคความต้องการ ..... 24

ภาพที่ 2.6 Penn Treebank Tagset ..... 26

ภาพที่ 2.7 ตัวอย่างไดอะแกรมสถานะของสภาพอากาศแบบมาร์คอฟซ่อนเร้น ..... 30

ภาพที่ 2.8 ตัวอย่างต้นไม้ไวยากรณ์ของประโยค ..... 31

ภาพที่ 2.9 ตัวอย่างการลดรูปคำ ..... 32

ภาพที่ 2.10 ขั้นตอนการสร้างแม่แบบด้วยแนวคิดของการเรียนรู้แบบมีผู้สอน ..... 34

ภาพที่ 2.11 ขั้นตอนการสกัดสารสนเทศด้วยแนวคิดของการเรียนรู้แบบไม่มีผู้สอน ..... 36

ภาพที่ 2.12 ตัวอย่างการแทนองค์ความรู้เชิงเครือข่าย ..... 38

ภาพที่ 2.13 ตัวอย่างการแทนองค์ความรู้เชิงโครงสร้าง ..... 38

ภาพที่ 2.14 ตัวอย่าง Use Case Diagram ..... 41

ภาพที่ 3.1 ตัวอย่างประโยคจากคลังข้อมูล SNLI ..... 46

ภาพที่ 3.2 กรอบการดำเนินงานวิจัย ..... 46

ภาพที่ 3.3 กรอบการประมวลผลการสร้างกฎในการสกัด *Actor* ..... 47

ภาพที่ 3.4 กรอบการประมวลผลเพื่อการทำความสะอาดประโยค ..... 48

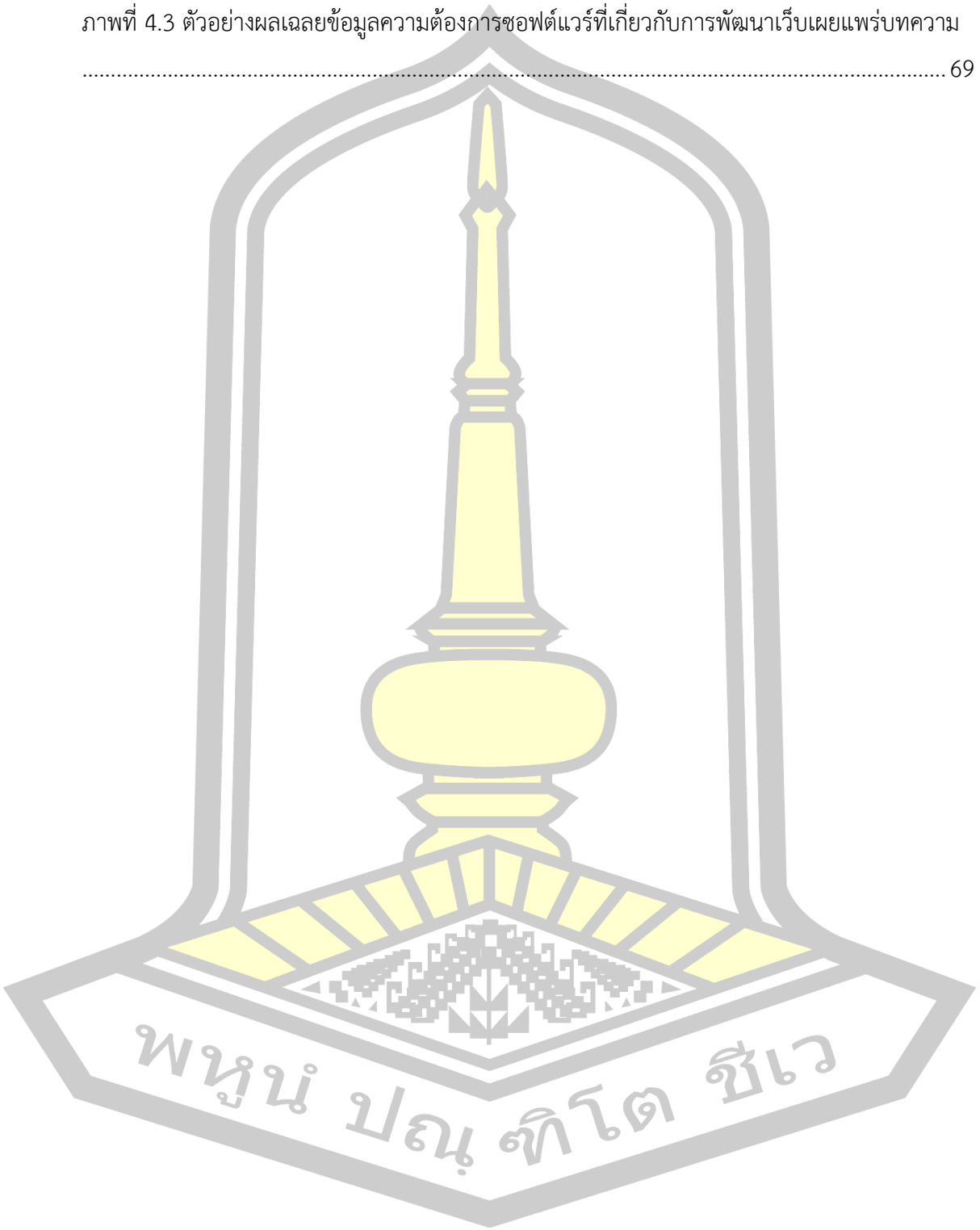
ภาพที่ 3.5 ประโยคที่ผ่านการทำ *POS tagging* และการทำความสะอาดประโยค ..... 48

ภาพที่ 3.6 ตัวอย่างการหาชุดลำดับของ *tag* ของกลุ่มคำนามที่เริ่มต้นประโยค ..... 49

ภาพที่ 3.7 ตัวอย่างชุดลำดับของ *tag* ..... 50

ภาพที่ 3.8 แผนภาพแบบจำลองความน่าจะเป็นของชุดลำดับของ *tag* ..... 51
ภาพที่ 3.9 ตัวอย่างการหารูปแบบการแทนคำสรรพนาม ..... 53
ภาพที่ 3.10 ตัวอย่างแบบจำลองมาร์คอฟซ่อนเร้นที่สร้างจากรูปแบบการแทนคำสรรพนาม ..... 54
ภาพที่ 3.11 กรอบการประมวลผลการสร้างกฎในแยกประโยคความรวม ..... 55
ภาพที่ 3.12 ตัวอย่างการหาตำแหน่งของคำสันธานในประโยคความรวม ..... 56
ภาพที่ 3.13 ตัวอย่างการสกัดเพื่อหาชุดลำดับของ *Object* ในประโยคความรวม ..... 57
ภาพที่ 3.14 ตัวอย่างลำดับของ *Object* ..... 58
ภาพที่ 3.15 ตัวอย่างแบบจำลองความน่าจะเป็นของลำดับของ *Object* ..... 59
ภาพที่ 3.16 กรอบการประมวลผลเพื่อการสกัด *Actor* และ *Action* จาก เอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 60
ภาพที่ 3.17 ตัวอย่างข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 61
ภาพที่ 3.18 ขั้นตอนการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 61
ภาพที่ 3.19 ตัวอย่างการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 62
ภาพที่ 3.20 ขั้นตอนการประมวลผลเพื่อการสกัดหา Actor จากเอกสารข้อความที่แสดง ความต้องการซอฟต์แวร์ ..... 63
ภาพที่ 3.21 ตัวอย่างการใช้กฎในการแทนคำสรรพนาม ..... 64
ภาพที่ 3.22 กรอบการประมวลผลการสกัด Action ของแต่ละ Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 65
ภาพที่ 3.23 ตัวอย่างการระบุตำแหน่งของ *Actor* ในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 65
ภาพที่ 3.24 ตัวอย่างการสกัดรูปแบบโครงสร้างของประโยคข้อความที่แสดงความต้องการซอฟต์แวร์ ..... 66
ภาพที่ 3.25 ตัวอย่างผลการสกัด *Action* และ *Actor* ออกจาก *Object* ..... 67
ภาพที่ 3.26 ตัวอย่างการแสดงแบบจำลองกระบวนการของซอฟต์แวร์ ..... 67
ภาพที่ 4.1 ตัวอย่างประโยคที่ใช้ประเมินกฎในการสกัด Actor ..... 68

ภาพที่ 4.2 ตัวอย่างข้อมูลความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บเผยแพร่บทความ..... 69
ภาพที่ 4.3 ตัวอย่างผลเฉลยข้อมูลความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บเผยแพร่บทความ
..... 69



# บทที่ 1

# บทนำ

## 1.1 หลักการและเหตุผล

ในการพัฒนาซอฟต์แวร์นั้น ขั้นตอนการวิเคราะห์เพื่อออกแบบระบบ (Software Analysis and Design) เป็นส่วนที่มีความสำคัญมาก โดยทั่วไปในขั้นตอนการออกแบบซอฟต์แวร์จะมีขั้นตอนที่สำคัญคือการสกัดซีนาริโอ (Scenario) [1] และการสร้างแบบจำลองกระบวนการ (Software Process Modeling) [2, 3]

ซีนาริโอ [4] คือ ลำดับของกิจกรรม (Activity) หรืองาน (Task) ที่มีประโยชน์อย่างมากในการช่วยให้นักพัฒนาซอฟต์แวร์เข้าใจการปฏิสัมพันธ์ในองค์ประกอบของซอฟต์แวร์และการตรวจสอบสถาปัตยกรรมซอฟต์แวร์ [4, 5] นอกจากนี้ยังแสดงให้เห็นแนวทางหรือพฤติกรรมของงานที่เกิดขึ้นในซอฟต์แวร์ นอกจากนี้ซีเนริโอได้ถูกนำมาใช้สำหรับรวบรวมและกลั่นกรองความต้องการของซอฟต์แวร์ โดยประสิทธิภาพและความสำเร็จของโครงการพัฒนาซอฟต์แวร์นั้นมักขึ้นอยู่กับคุณภาพของ ซีเนริโอที่รวบรวมมา [6-8]

จากการศึกษาที่ผ่านมาซีเนริโอไม่เพียงถูกใช้เป็นข้อมูลที่สำคัญในการพัฒนาซอฟต์แวร์สมัยใหม่ แต่ยังเป็นเครื่องมือที่ใช้ในการตรวจสอบและปรับปรุงซอฟต์แวร์อีกด้วย โดยเฉพาะอย่างยิ่งซอฟต์แวร์ที่มีโครงสร้างขนาดใหญ่ เพราะซีเนริโอสามารถถูกใช้ได้ในทุกๆ กระบวนการของวงจรการพัฒนาซอฟต์แวร์ (Software Development Life Cycle: SDLC) เนื่องจากซีเนริโอจะช่วยให้ผู้พัฒนาซอฟต์แวร์เข้าใจเชิงลึกในสภาพแวดล้อมต่างๆ ที่อาจเกิดขึ้นในขณะที่มีการพัฒนาและการทำงาน รวมถึงสิ่งที่จำเป็นต่อการดำเนินการของซอฟต์แวร์ที่กำลังจะพัฒนาในสถานการณ์ปัจจุบัน นอกจากนั้น ซีเนริโอยังลดความเห็นขัดแย้งในเรื่องความต้องการของซอฟต์แวร์ระหว่างเจ้าของระบบและผู้พัฒนาซอฟต์แวร์ [4, 5] ดังนั้นซีเนริโอเป็นเสมือนเครื่องมือในการตรวจสอบความต้องการของซอฟต์แวร์ก่อนเริ่มการพัฒนา ใช้เป็นแนวทางในการวิเคราะห์ข้อจำกัดของซอฟต์แวร์ (Software Specification) และการออกแบบซอฟต์แวร์ จนถึงการใช้ซีเนริโอยังสามารถใช้ในการวางแผนเพื่อการการทดสอบซอฟต์แวร์ที่พัฒนาขึ้นก่อนที่จะถูกนำไปใช้งานจริง [6-8]

ขณะที่แบบจำลองกระบวนการของซอฟต์แวร์ (Software Process Models) คือการนำความต้องการของซอฟต์แวร์ เช่น กระบวนการและข้อมูลที่เกี่ยวข้อง รวบรวมและแสดงให้อยู่ในรูป

แบบจำลองที่เข้าใจและใช้งานได้ง่าย แบบจำลองกระบวนการมีบทบาทสำคัญอย่างมากในขั้นตอนการพัฒนาซอฟต์แวร์ [2, 3, 9-11] โดยแบบจำลองที่ดีจะช่วยควบคุมแนวทางการพัฒนาซอฟต์แวร์ให้มีคุณภาพและเป็นไปตามวัตถุประสงค์ [2, 9-13] นอกจากนี้แบบจำลองกระบวนการยังช่วยให้ผู้พัฒนาเห็นโครงสร้าง การออกแบบ หรือเป้าหมายของแต่ละกระบวนการซอฟต์แวร์ อีกทั้งยังเป็นเครื่องมือที่ช่วยให้การสื่อสารระหว่างผู้เกี่ยวข้องเข้าใจกันง่ายขึ้น [2, 3, 9] ในการพัฒนาซอฟต์แวร์แบบจำลองกระบวนการถูกนำมาใช้เป็นเครื่องมือสำหรับช่วยจัดการกับความสัมพันธ์ที่เกิดขึ้นในกระบวนการพัฒนาซอฟต์แวร์ไม่ว่าจะเป็น ต้นทุน เวลา เครื่องมือ บุคคล หรือทรัพยากร [9, 11-13] โดยปัจจัยเหล่านี้มีผลอย่างยิ่งกับผลิตภัณฑ์ซอฟต์แวร์ที่พัฒนา นั่นแสดงให้เห็นถึงความสำคัญของแบบจำลองกระบวนการของซอฟต์แวร์ในขั้นตอนการพัฒนาซอฟต์แวร์

ในทศวรรษที่ผ่านมา พบว่ามีการศึกษาเพื่อการวิเคราะห์เพื่อการสกัดซีนาริโอและการสร้างแบบจำลองกระบวนการแบบอัตโนมัติ ได้กลายเป็นปัญหาที่สำคัญในงานวิศวกรรมซอฟต์แวร์ (Software Engineering) วิศวกรรมความต้องการ (Requirement Engineering) และการจัดการกระบวนการทำงาน (Business Process Management: BPM) [14] เพราะว่ามันเป็นงานที่ต้องใช้เวลาและความพยายามเป็นอย่างมากในการวิเคราะห์และทำความเข้าใจเพื่อสกัดความต้องการซอฟต์แวร์ โดยเฉพาะถ้าเอกสารความต้องการซอฟต์แวร์นั้นมีเป็นจำนวนมาก การใช้แรงงานคนเพียงอย่างเดียวอาจจะไม่ทันการ ดังนั้น ในปัจจุบันจึงมีความพยายามที่จะประโยชน์จากการประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) เข้ามาช่วยในการวิเคราะห์และสกัดซีเนริโอจากความต้องการของซอฟต์แวร์แบบอัตโนมัติ เพราะโดยทั่วไปความต้องการของซอฟต์แวร์มักอยู่ในรูปแบบของข้อความ (plain text) [14, 15]

อย่างไรก็ตาม การสกัดซีนาริโอและการสร้างแบบจำลองกระบวนการแบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความไม่ใช่เรื่องง่าย เนื่องจาก “ข้อความ (Text)” เป็นข้อมูลที่ไม่มีรูปแบบหรือโครงสร้างสำหรับการเข้าถึงข้อมูลที่ชัดเจน และการวิเคราะห์ข้อมูลที่เป็นข้อความจำเป็นต้องตระหนักถึงความซับซ้อนของข้อมูลเป็นอย่างมาก [16, 17] จากเหตุผลดังกล่าว การสกัดซีนาริโอและการสร้างแบบจำลองกระบวนการแบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความจึงไม่ใช่เรื่องง่าย [17, 18] ดังนั้นปัญหาดังกล่าวจึงกลายเป็นความท้าทายในการศึกษานี้

ในงานวิจัยนี้ นำเสนอการสกัดซีนาริโอและสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความบนพื้นฐานเทคนิคด้านการประมวลผลภาษาธรรมชาติ

**1.2 วัตถุประสงค์ของการวิจัย**

1) เพื่อสกัดซีนาริโอหรือลำดับการทำงานของซอฟต์แวร์สำหรับผู้ใช้งานในแต่ละโดเมนแบบอัตโนมัติจากเอกสารความต้องการด้านซอฟต์แวร์ที่เป็นข้อความด้วยเทคนิคด้านการประมวลผลภาษาธรรมชาติ
2) เพื่อนำซีนาริโอที่ได้มาสร้างเป็นแบบจำลองกระบวนการของซอฟต์แวร์ (Software Process) แบบอัตโนมัติ

**1.3 ขอบเขตของงานวิจัย**

1) การสกัดซีนาริโอคือการสกัดหา Actor (ผู้ใช้งาน) และงานทำงาน (Action) ของผู้ใช้นั้นๆ แบบอัตโนมัติจากเอกสารความต้องการด้านซอฟต์แวร์ที่เป็นข้อความด้วยเทคนิคด้านการประมวลผลภาษาธรรมชาติ โดยเครื่องมือที่ใช้ในการสกัด Actor และ Action จะอยู่ในรูปแบบกฎ (Rules)
2) นำซีนาริโอที่ได้มาสร้างเป็นแบบจำลองกระบวนการของซอฟต์แวร์ (Software Proces)
3) แบบจำลองกระบวนการของซอฟต์แวร์จะแสดงในรูปแบบข้อมูลกึ่งโครงสร้าง (Semi-Structured Data)
4) กระบวนการในการสร้างกฎจะเป็นการประยุกต์หลักการแบบ POS tagging และแบบจำลองมาร์คอฟและเอ็นแกรม
5) กระบวนการในการวิเคราะห์โครงสร้างของประโยคเพื่อสกัดซีนาริโอจะเป็นการประยุกต์ POS tagging และการวิเคราะห์สารสนเทศโดยใช้แม่แบบ (Template-based Information Analysis)
6) โครงสร้างของประโยคที่จะทำสกัดซีนาริโอจะอยู่ในรูปแบบ Active Voice คือประโยคที่ประธานของประโยคนั้นเป็นผู้กระทำกริยา

7) ชุดข้อมูลที่ใช้ในการสร้างกฎในการสกัด Actor และ Action จะเป็นประโยคภาษาอังกฤษ 300,000 ประโยค และประโยคที่ใช้ทดสอบจำนวนไม่น้อยกว่า 1,000 ประโยค และมาจากฐานข้อมูลมาตรฐาน SNLI
8) เอกสารที่จะใช้ในงานวิจัยเป็นเอกสารความต้องการของซอฟต์แวร์ที่เป็นภาษาอังกฤษ ความยาวต่อเนื่องไม่น้อยกว่า 20 หน้า
9) ในการประเมินผลลัพธ์ที่ได้จากกระบวนการที่นำเสนอ จะทำการประเมินใน 4 ส่วนคือ
    (1) การประเมินการทำความสะอาดประโยค (Sentence Cleaning) เป็นการนำประโยคที่ผ่านการทำความสะอาดมาเปรียบเทียบกับผลเฉลยที่สกัดด้วยผู้เชี่ยวชาญ โดยใช้ค่าความระลึก (Recall) ค่าความแม่นยำ (Precision) และค่าประสิทธิภาพ (F-measure)
    (2) การประเมินกฎในการสกัด Actor และ Action โดยเปรียบเทียบผลกับผลเฉลยที่สกัดจากประโยคภาษาอังกฤษจากชุดข้อมูล SNLI จำนวน 1,000 ประโยคโดยผู้เชี่ยวชาญ ด้วยค่าความระลึก ค่าความแม่นยำ และค่าประสิทธิภาพ
    (3) การประเมินผลการสกัด Actor เป็นการประเมินจาก Actor ทั้งหมดที่สกัดมาได้ด้วยการเปรียบเทียบกับผลเฉลยที่สกัดด้วยผู้เชี่ยวชาญ โดยใช้ค่าความระลึก ค่าความแม่นยำ และค่าประสิทธิภาพ
    (4) การประเมินซีนาริโอที่สกัดได้ ซึ่งจะเป็นการประเมินเชิงเปรียบเทียบกับผลเฉลยที่ผู้เชี่ยวชาญได้ทำเอาไว้ โดยเป็นการประเมินด้วยค่าความระลึก

**1.4 ความสำคัญของงานวิจัย**

1) ได้กระบวนการสกัดซีนาริโอจากเอกสารความต้องการของระบบ และการสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติ
2) ซีนาริโอที่สกัดได้สามารถทำให้ผู้ที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์มองเห็นขั้นตอนการทำงานที่ตรงกัน ทำให้ลดความเห็นขัดแย้งในเรื่องความต้องการของซอฟต์แวร์ระหว่างเจ้าของระบบและผู้พัฒนาซอฟต์แวร์
3) การสกัดซีนาริโอและการสร้างแบบจำลองกระบวนการแบบอัตโนมัติจากเอกสารความต้องการของระบบ น่าจะช่วยลดเวลาในการวิเคราะห์และพัฒนาซอฟต์แวร์ขนาดใหญ่ได้

**1.5 นิยามศัพท์เฉพาะ**

1) แบบจำลองกระบวนการทำงานของซอฟต์แวร์ (Software Process Models) คือการแสดงกระบวนการของซอฟต์แวร์ในรูปแบบที่เหมาะสมกับการแสดงภาพรวมของกระบวนการ เพื่อให้ง่ายต่อการทำความเข้าใจและอธิบาย
2) ความต้องการของซอฟต์แวร์ (Software Requirement) คือ ความต้องการของซอฟต์แวร์ที่กำลังจะพัฒนา โดยสามารถรวบรวมได้จากหลายๆ แหล่ง เช่น เอกสารของระบบเดิม (Old documents) การตอบแบบสอบถาม (Questionnaires) การสัมภาษณ์ผู้ใช้งาน (Interviews) หรือการสังเกตการณ์การทำงานของผู้ใช้งาน (Observation) เป็นต้น
3) ซีนาริโอ (Scenario) ในทางวิศวกรรมซอฟต์แวร์ หมายถึงลำดับของกิจกรรม (Activities) หรืองาน (Tasks) ที่เกิดขึ้นในแต่ละกระบวนการในการทำงานของซอฟต์แวร์ โดยในงานวิจัยฉบับนี้ ซีนาริโอคือลำดับการกระทำ (Action) ของแต่ละผู้กระทำ (Actor)
4) การวิเคราะห์สารสนเทศโดยใช้แม่แบบ (Template-based Information Analysis) คือการวิเคราะห์ตำแหน่งและชนิดของคำในประโยค ว่าตรงกับแม่แบบหรือไม่ ซึ่งแม่แบบที่ใช้ในงานวิจัยนี้อยู่ในรูปแบบกฎ (Rules) โดยจะแบ่งออกเป็น 3 กลุ่มตามจุดประสงค์การใช้งานคือ
    (1) กฎในการสกัด Actor
    (2) กฎการแทนคำสรรพนาม
    (3) กฎในการแยกประโยคความรวม

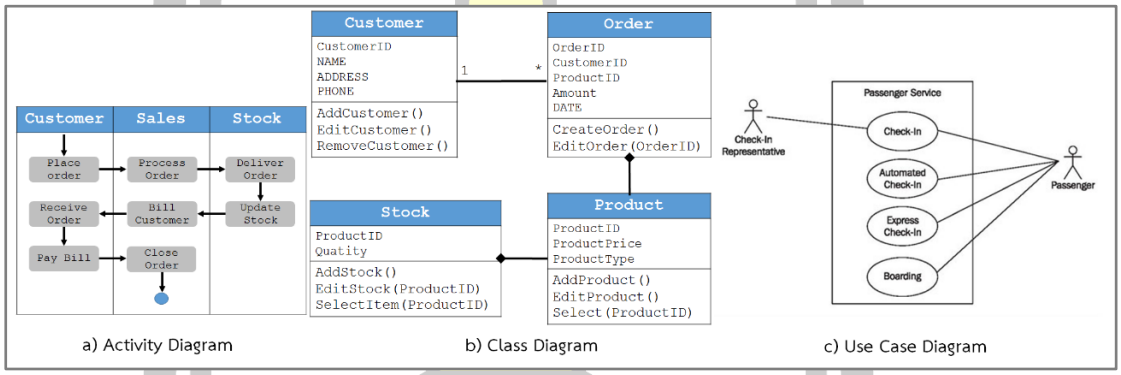
พหุน ปณ ฑิโต ชีเว

# บทที่ 2

# ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

## 2.1 แบบจำลองกระบวนการของซอฟต์แวร์

แบบจำลองกระบวนการของซอฟต์แวร์ (Software Process Models) คือการแสดงกระบวนการของซอฟต์แวร์ในรูปแบบที่เหมาะสมกับการแสดงภาพรวมของกระบวนการ เพื่อให้ง่ายต่อการทำความเข้าใจและอธิบาย โดยเน้นไปที่พฤติกรรมการโต้ตอบระหว่างผู้เกี่ยวข้องหรือระบบ [19] เช่น ใครทำกระบวนการอะไร และจะได้ผลลัพธ์อะไร (ถ้ามี) ในขณะที่กระบวนการของซอฟต์แวร์ (Software Process) จะเน้นความสำคัญไปที่ขั้นตอน ลำดับการทำงาน หรือวิธีการทำงานภายในกระบวนการของซอฟต์แวร์ [9]




ภาพที่ 2.1 ตัวอย่างแบบจำลองกระบวนการ

ในการพัฒนาซอฟต์แวร์ แบบจำลองกระบวนการถูกนำมาใช้เป็นเครื่องมือสำหรับช่วยจัดการกับความสัมพันธ์ที่เกิดขึ้นในกระบวนการพัฒนาซอฟต์แวร์ไม่ว่าจะเป็น ต้นทุน เวลา เครื่องมือ บุคคล หรือทรัพยากร โดยปัจจัยเหล่านี้มีผลอย่างยิ่งกับผลิตภัณฑ์ซอฟต์แวร์ที่พัฒนา ดังนั้นการสร้างแบบจำลองของกระบวนการจึงเป็นที่จำเป็นและสำคัญในขั้นตอนการพัฒนาซอฟต์แวร์

ซึ่งในงานวิจัยนี้ ได้ทำการสร้างแบบจำลองกระบวนการของซอฟต์แวร์โดยแสดงให้อยู่ในรูปแบบข้อมูลกึ่งโครงสร้างที่แสดงตัวอย่างดังภาพที่ 2.2 โดยที่แสดงกระบวนการของผู้ใช้งานแต่ละส่วน ซึ่งรวบรวมมาจากซีนาริโอที่สกัดจากเอกสารความต้องการของซอฟต์แวร์ที่เป็นข้อความ

| Actor : | Customer |
|---|---|
| Action: | Order<br>Confirm Order<br>Payment<br>Get Product |

| Actor : | Employee |
|---|---|
| Action: | Get Order<br>Check Stock<br>Get Paid<br>Send Product |

ภาพที่ 2.2 ตัวอย่างข้อมูลกึ่งโครงสร้าง

## 2.1 ซีนาริโอ

### 2.2.1 ความหมายของซีนาริโอ

ซีเนริโอ (Scenario) ในทางวิศวกรรมระบบและวิศวกรรมซอฟต์แวร์หมายถึงลำดับของกิจกรรมหรืองาน [4, 5] ซีเนริโอคือการอธิบายการทำงานตามลำดับไปสู่เป้าหมาย (ของซีเนริโอ) จึงสำคัญอย่างมากในกระบวนการพัฒนาซอฟต์แวร์ [1] สำหรับช่วยให้นักพัฒนาซอฟต์แวร์เข้าใจถึงกระบวนการทำงานต่างๆ ที่เกิดขึ้นในซอฟต์แวร์ว่าความต้องการในซอฟต์แวร์คืออะไรและควรดำเนินการต่อความต้องการนั้นอย่างไร นั่นแสดงให้เห็นว่าซีเนริโอมีประโยชน์อย่างมากสำหรับการวิเคราะห์ความต้องการของซอฟต์แวร์ การออกแบบ ตลอดจนถึงการทดสอบระบบ [6-8] ซึ่งซีเนริโอสามารถจัดเป็นกลุ่มได้จากข้อความที่มีประโยคเกี่ยวข้องกันในซีเนริโอเพื่อรวบรวมและแสดงลำดับการทำงานของแต่ละกิจกรรม หรือลำดับการทำงานของผู้เกี่ยวข้องกับซอฟต์แวร์ในแต่ละส่วน

[Title: Reservation of a meeting room]
[Viewpoints: citizen, system]{
1. A citizen enters reservation information to the system.
2. The system retrieves available room from the database using the information.
3. The system shows an available room to the citizen.
4. The citizen enters his name and telephone number to the system.
5. The system validates the citizen with the name and the telephone number.
6. The system shows the room rate to the citizen.
7. The citizen pays the rate to the system.
8. The system issues a receipt to the citizen.
9. The system shows the room number to the citizen.
}

ภาพที่ 2.3ตัวอย่างซีเนริโอการจองห้องประชุม

ซีเนริโอสามารถแบ่งได้เป็น 3 ประเภท [1] ได้แก่

1) Normal Scenario เป็นกลุ่มการทำงานที่อยู่ในระบบเดียวกัน เป็นพฤติกรรมทั่วไปที่แสดงถึงเป้าหมายของระบบ เช่น ซีเนริโอของระบบลงทะเบียนเรียนของนิสิตนักศึกษาที่จะประกอบไปด้วย การทำงานของเจ้าหน้าที่และนิสิต โดยระบบนี้จะประกอบด้วย 2 ซีเนริโอ คือซีเนริโอที่แสดงการทำงานต่างๆ ในระบบที่เป็นไปได้ของเจ้าหน้าที่ และ ซีเนริโอที่แสดงการทำงานต่างๆ ในระบบที่เป็นไปได้ของนิสิต

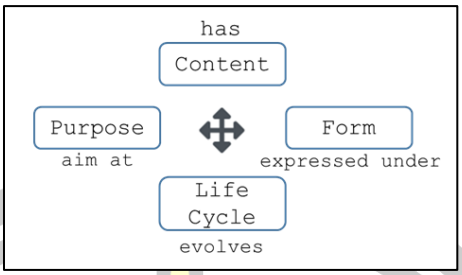
2) Alternative Scenario เป็นพฤติกรรมที่มีทางเลือกในระบบ เช่นระบบลงทะเบียนเรียนของนิสิตนักศึกษา กรณีที่รายวิชาและกลุ่มเรียนที่นิสิตต้องการลงทะเบียนเรียนไม่สามารถลงทะเบียนเรียนได้ ระบบมีกลุ่มเรียนอื่นๆ ในรายวิชาเดียวกันกับที่นิสิตต้องการลงทะเบียนเรียนเป็นทางเลือกให้แก่นิสิตที่ต้องการลงทะเบียนเรียน

3) Exceptional Scenario เป็นพฤติกรรมที่ผิดปกติในระบบ เช่นระบบลงทะเบียนเรียนของนิสิตนักศึกษา ในกรณีที่นิสิตต้องการลงทะเบียนเรียนหรือถอนรายวิชา แต่ไม่สามารถดำเนินการผ่านระบบปกติได้เนื่องจากอยู่นอกเหนือช่วงวันเวลาที่กำหนดไว้ในปฏิทินการศึกษา ระบบอาจแสดงข้อความหรือคำแนะนำต่างๆ ให้แก่นิสิต เพื่อดำเนินการลงทะเบียนเรียนหรือถอนรายวิชาเป็นกรณีพิเศษได้

ได้มีการสำรวจวรรณกรรมต่างๆ ที่เกี่ยวข้องกับซีเนริโอและได้สรุปออกเป็นมุมมองต่างๆ โดย Weidenhaupt และคณะ [19] ซึ่งได้แบ่งออกเป็น 4 มุมมอง ได้แก่

1) Form คือมุมมองที่เกี่ยวข้องกับการแสดงผล เช่นภาพเคลื่อนไหวหรือการโต้ตอบกับผู้ใช้
2) Purpose คือมุมมองที่อยู่ในกระบวนการพัฒนาซอฟต์แวร์ เช่นการอธิบายลำดับการทำงานในซอฟต์แวร์
3) Content คือมุมมองที่เกี่ยวข้องกับการปฏิสัมพันธ์ต่างๆ ทั้งภายในและภายนอกระบบ
4) Life cycle คือมุมมองที่พิจารณาซีเนริโอเสมือนว่าเป็นวัตถุที่ต้องเกี่ยวข้องกับการบริหารจัดการโครงการ เช่นการจัดการด้านเทคนิคและระยะเวลาในการพัฒนา

โดยความสัมพันธ์ของทั้ง 4 มุมมองข้างต้นสามารถแสดงได้ดังภาพที่ 2.4




ภาพที่ 2.4 มุมมองเกี่ยวกับซีเนริโอ

2.2.2 รูปแบบการนำเสนอซีนาริโอจากความต้องการซอฟต์แวร์

1) Structured Representation แสดงเป็นเอกสารที่จัดเก็บในรูปแบบที่ทันสมัย ง่ายต่อการอ่านและทำความเข้าใจ รวมถึง Use Case Diagram, Sequence Diagram, Swimlanes Diagrams ,Data flow Diagram (DFD) และ Flowcharts ที่แสดงตัวอย่างบางส่วนดังภาพที 2.2

2) Semi-structured Representation คือโครงสร้างข้อมูลที่ประกอบด้วยข้อมูลแบบมีโครงสร้างรวมอยู่กับข้อมูลที่ไม่มีโครงสร้างจึงเรียกว่าข้อมูลกึ่งโครงสร้าง โดยแสดงตัวอย่างดังภาพที่ 2.2

3) Sequential Representation แสดงลำดับขั้นตอนในการโต้ตอบระหว่างมนุษย์หรือเครื่องจักรที่ทำหน้าที่ในระบบในการทำงานหรือเหตุการณ์ต่างๆ

4) Storyboards แสดงถึงสถานการณ์ บทบาท และการทำงานแต่ละลำดับของเหตุการณ์ต่างๆ ที่อยู่ในระบบ

2.2.3 ภาษาและโครงสร้างประโยคของเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

โดยทั่วไปในซีเนริโอที่แสดงถึงลำดับในการทำงานซึ่งประกอบด้วยหลายๆ ประโยค แต่ละประโยคจะประกอบด้วย 2 ส่วนหลักคือ ส่วนของ *Actor* และ *Action* สำหรับ *Actor* จะแสดงถึง *“ผู้กระทำ”* ขณะที่ *Action* จะแสดงถึง *“การกระทำหรือการทำงาน”* ในแต่ละซีเนริโอ ซึ่งรูปแบบนี้ในทางวิศวกรรมความต้องการ หรือ Business Process Management (BPM) จะเรียกว่า *“Actor-Action-Object”* [20]

ในงานวิจัยชิ้นนี้ได้มุ่งเน้นไปที่เอกสารความต้องการของซอฟต์แวร์ที่เป็นภาษาอังกฤษ โดยทั่วไปประโยคภาษาอังกฤษประกอบไปด้วย *Subject* (ประธาน) และ *Predicate* (ภาคแสดง) ซึ่ง

สอดคล้องกับ *“Actor-Action-Object”* โดย *Actor* เปรียบเสมือนประธานและ *“Action-Object”* เปรียบเสมือนภาคแสดงในประโยค แสดงได้ดังรูปแบบต่อไปนี้

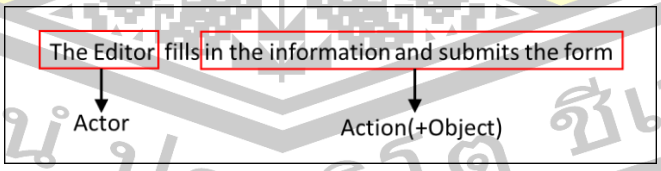
*Sentence = Subject + Predicate*

ประธานนั้นสามารถเป็นได้ทั้งบุคคลหรือสิ่งของที่แสดงการกระทำ ขณะที่ภาคแสดงเป็นส่วนหนึ่งของประโยคที่ประกอบไปด้วยคำกริยา (Verb) หรือคำกริยาวลี (Verb Phrase) และส่วนขยาย (Complements) คำกริยานั้นใช้เพื่ออธิบายการกระทำของประธาน ในขณะที่ส่วนขยายเป็นบุคคลหรือสิ่งของที่ได้รับผลจากการกระทำที่คำกริยาได้ระบุไว้ โดยคำกริยาสามารถอธิบายได้ดังต่อไปนี้

*<Predicate> = (<Verb> | <Verb Phrase>) <Complements>*

หากคำกริยาไม่มีส่วนเสริมจะถูกเรียกว่าอกรรมกริยา (Intransitive) แต่ทว่าถ้าคำกริยาต้องมีหนึ่งหรือสองส่วนขยายจะเรียกว่าสกรรมกริยา (Transitive) ในไวยากรณ์แบบเดิมส่วนขยายเหล่านี้จะถูกเรียกว่ากรรมรอง (Indirect object) และกรรมตรง (Direct object) ส่วนขยายของกริยาหรือกริยาวลีใช้บรรยายถึงตัวเลือกหรือส่วนประกอบไม่เพียงแต่วัตถุแต่ยังมี คำคุณศัพท์ (Adjectives) บุพบทวลี (Prepositional phrases) และอื่นๆ

เมื่อเทียบกับโครงสร้างประโยคในเอกสารความต้องการของซอฟต์แวร์ [14] จะเห็นว่าประธานมักจะหมายถึง *Actor* อยู่เสมอ ในขณะที่การกระทำของประธานจะอธิบายด้วยคำกริยาหรือกริยาวลี ดังนั้นตัวอย่างของ *Actor* และ *Action* ในประโยคความต้องการสามารถแสดงได้ดังภาพที่ 2.5




ภาพที่ 2.5 ตัวอย่างตำแหน่งของ Actor และ Action ในประโยคความต้องการ

**2.3 เทคนิคการประมวลผลภาษาธรรมชาติ**

การประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) มุ่งเน้นที่การวิเคราะห์และคำนวนภาษาของมนุษย์ [21] NLP เป็นสหวิทยาการประกอบด้วยภาษาศาสตร์ ภาษาศาสตร์คอมพิวเตอร์ และปัญญาประดิษฐ์ เพื่อจัดการกับความซับซ้อนของภาษาธรรมชาติ โดยปกติจะจำแนกภาษาคอมพิวเตอร์ได้เป็นหกระดับได้แก่ วิเคราะห์ระบบเสียง (Phonological analysis) การวิเคราะห์รูปพรรณ (Morphological analysis) การวิเคราะห์โครงสร้างประโยค (Syntactic analysis) การวิเคราะห์ความหมาย (Semantic analysis) การวิเคราะห์แบบปฏิบัตินิยม (Pragmatic analysis) และการวิเคราะห์บทสนนา (Discourse analysis)

มีหลายๆ เทคนิคที่ใช้สำหรับการวิเคราะห์ภาษาธรรมชาติ โดยเทคนิคบางส่วนที่เกี่ยวข้องกับงานวิจัยนี้จะอธิบายอย่างรวบรัดดังต่อไปนี้

2.3.1 POS Tagging

Part of Speech Tagging (POS Tagging) เป็นกระบวนการเพื่อกำหนดขอบเขตของคำพูดเพื่อระบุชนิดของคำในประโยค การกำหนดขอบเขตของคำเพื่อกำกับชนิดของคำโดยอัติโนมัติเรียกว่า “*Tagging*” ในขณะที่ตัวกำกับคำเรียกว่า “tag” [21]

ในภาษาอังกฤษ ไวยากรณ์สามารถแบ่งคำได้เป็น 8 ประเภทได้แก่ คำกริยา (verb) คำนาม (noun) คำสันธาน (conjunction) คำสรรพนาม (pronoun) คำคุณศัพท์ (adjective) คำวิเศษณ์ (adverb) คำบุพบท (preposition) และคำอุทาน (interjection)

ในการระบุ POS ในขั้นตอนแรกนั้นควรกำหนดชุดมาตรฐานของ tags ที่ใช้ในการทำ POS tagging ในปัจจุบันมีชุดของ Tag สำหรับภาษาอังกฤษสามมาตรฐานที่นิยมใช้กันได้แก่ Penn Treebank [22], Brown Corpus Tag-set และ British national corpus ซึ่งโดย Penn treebank กำหนดไว้ที่ 45 tag (แสดงตัวอย่างดังภาพที่ 2.6) ในขณะที่ Brown Corpus และ British national corpus กำหนดไว้ที่ 192 tag และ 61 tag ตามลำดับ

| Tag | Description | Example | Tag | Description | Example |
|---|---|---|---|---|---|
| CC | coordin. conjunction | *and, but, or* | SYM | symbol | +,%, & |
| CD | cardinal number | *one, two, three* | TO | "to" | *to* |
| DT | determiner | *a, the* | UH | interjection | *ah, oops* |
| EX | existential 'there' | *there* | VB | verb, base form | *eat* |
| FW | foreign word | *mea culpa* | VBD | verb, past tense | *ate* |
| IN | preposition/sub-conj | *of, in, by* | VBG | verb, gerund | *eating* |
| JJ | adjective | *yellow* | VBN | verb, past participle | *eaten* |
| JJR | adj., comparative | *bigger* | VBP | verb, non-3sg pres | *eat* |
| JJS | adj., superlative | *wildest* | VBZ | verb, 3sg pres | *eats* |
| LS | list item marker | *1, 2, One* | WDT | wh-determiner | *which, that* |
| MD | modal | *can, should* | WP | wh-pronoun | *what, who* |
| NN | noun, sing. or mass | *llama* | WP$ | possessive wh- | *whose* |
| NNS | noun, plural | *llamas* | WRB | wh-adverb | *how, where* |
| NNP | proper noun, singular | *IBM* | $ | dollar sign | *$* |
| NNPS | proper noun, plural | *Carolinas* | # | pound sign | # |
| PDT | predeterminer | *all, both* | " | left quote | ' or " |
| POS | possessive ending | *'s* | " | right quote | ' or " |
| PRP | personal pronoun | *I, you, he* | ( | left parenthesis | [, (, {, < |
| PRP$ | possessive pronoun | *your, one's* | ) | right parenthesis | ], ), }, > |
| RB | adverb | *quickly, never* | , | comma | , |
| RBR | adverb, comparative | *faster* | . | sentence-final punc | . ! ? |
| RBS | adverb, superlative | *fastest* | : | mid-sentence punc | : ; ... – - |
| RP | particle | *up, off* | | | |

ภาพที่ 2.6 Penn Treebank Tagset

โดยทั่วไปมีเทคนิคการทำ POS Tagging อยู่ 3 วิธี [21] ได้แก่ การจำแนกประเภทของคำด้วยกฎ (Rule-based POS Tagging), การจำแนกประเภทของคำด้วยแบบจำลองมาร์คอฟ (Markov Models) และการเรียนรู้บนพื้นฐานการเปลี่ยนแปลง (Transformation-based Learning)

ในงานนี้ได้ใช้ประโยชน์จากการทำ POS Tagging ด้วยเทคนิคที่อยู่บนพื้นฐานของความน่าจะเป็น ซึ่งเทคนิคนี้จะสกัดองค์ความรู้ทางภาษาศาสตร์จากคลังข้อมูลขนาดใหญ่และทำ POS Tagging โดยอัตโนมัติ ดำเนินด้วยหลักความน่าจะเป็นของความเป็นไปได้ที่จะเกิด tag ดังนั้น เทคนิค POS Tagging อยู่บนพื้นฐานของหลักความน่าจะเป็นและต้องมีชุดข้อมูลสำหรับเรียนรู้ความน่าจะเป็นที่จะเกิด tag และโมเดลมาร์คอฟมักถูกนำมาใช้ในการดำเนินการในขั้นตอนนี้

วัตถุประสงค์ของโมเดลมาร์คอฟคือการหาลำดับที่เหมาะสมที่สุดของ Tag $T = t_1, t_2, t_3, ..., t_n$ ของลำดับคำ $W = w_1, w_2, w_3, ..., w_n$ นั่นคือการหาลำดับของ tag ที่มีความน่าจะเป็นที่สุดสำหรับกลุ่มคำเหล่านั้น โดยถ้ายึดว่าโอกาสที่จะเกิด tag ขึ้นอยู่กับความน่าจะเป็นของ tag ที่เกิดขึ้นก่อนหน้ารูปแบบนี้เรียกว่าไบแกรมโมเดล (bigram model) แต่ละขั้นของไบแกรมโมเดลจะสอดคล้องกับ POS tag ความน่าจะเป็นระหว่าง POS state สามารถแสดงได้ด้วยสมการ $P(t_i, t_j)$ และความน่าจะเป็นของคำที่ถูกดึงออกมาจาก tag สามารถแสดงเป็น $P(w_i, t_j)$

2.3.2 แบบจำลองมาร์คอฟและเอ็นแกรม

แบบจำลองมาร์คอฟ (Markov Model) [23] คือ ตัวแบบทางคณิตศาสตร์ที่ใช้หลักความน่าจะเป็นในการวิเคราะห์พฤติกรรมของตัวแปรเพื่อคาดการณ์พฤติกรรมของตัวแปรที่จะเกิดขึ้นถัดไป โดยการใช้แบบจำลองมาร์คอฟที่ง่ายที่สุดคือลูกโซ่มาร์คอฟ (Markov chain)

ลูกโซ่มาร์คอฟ [24] คือลำดับของการเกิดเหตุการณ์ ซึ่งค่าความน่าจะเป็นของการเกิดเหตุการณ์ แต่ละเหตุการณ์จะขึ้นอยู่กับเหตุการณ์ที่เกิดขึ้นก่อนหน้า ด้วยการคำนวน $P(w|h)$ หมายถึงความน่าจะเป็นที่จะเกิดคำ w (word) ขึ้นอยู่กับข้อความ h (history) จากตัวอย่างสามารถอธิบายได้ดังต่อไปนี้ พิจารณาประโยคต่อไปนี้ “*its water is so transparent that*” โดยในการคำนวนหาค่าความน่าจะเป็นของคำว่า “*the*” ที่จะเกิดขึ้นหลังประโยคตัวอย่างสามารถคำนวนได้ด้วยสมการนี้

$$P(the|its\ water\ is\ so\ transparent\ that)$$

วิธีหนึ่งคือการประมาณค่าความน่าจะเป็นจากการนับความถี่สัมพัทธ์จากคลังข้อมูลขนาดใหญ่ นับจำนวนครั้งที่เห็นข้อความ “*its water is so transparent that*” และนับจำนวนครั้งที่ตามด้วยคำว่า “*the*” จะสามารถตอบคำถามที่ว่าจากจำนวนครั้งที่พบข้อความ h จากคลังข้อมูลจะตามด้วยข้อความ w จำนวนกี่ครั้งดังต่อไปนี้

$$\frac{C(the|its\ water\ is\ so\ transparent\ that\ the)}{C(the|its\ water\ is\ so\ transparent\ that)}$$

ในขณะที่วิธีการประมาณค่าความน่าจะเป็นใช้ได้ดีจากหลายๆ กรณี แต่มีจำนวนไม่น้อยที่ได้ค่าประมาณการที่ไม่ดีพอเนื่องจากคลังข้อมูลใหญ่ไม่เพียงพอ นั้นเป็นเพราะภาษามีประโยคหรือข้อความใหม่ๆ เกิดขึ้นตลอดเวลาและเราไม่สามารถที่จะคำนวนข้อความเหล่านี้ได้ทั้งหมด

เสมือนว่าถ้าเราอยากรู้ความน่าจะเป็นร่วมจากลำดับข้อความทั้งหมดที่เหมือน “*its water is so transparent*” เราสามารถทำได้โดยหาความน่าจะเป็นทั้งหมดจากลำดับที่มีคำ 5 คำ นั่นหมายถึงความถี่ของข้อความ “*its water is so transparent*” หารด้วยผลรวมจำนวนความถี่ของลำดับห้าคำที่เป็นไปได้ทั้งหมด ซึ่งดูมากเกินกว่าจะใช้ประมาณการ

ด้วยเหตุนี้จึงมีวิธีที่ดีกว่าเพื่อใช้ประเมินค่าความน่าจะเป็นของคำ *w* ที่ต่อจาก *h* หรือความน่าจะเป็นจากลำดับคำทั้งหมด เพื่อแสดงความน่าจะเป็นของตัวแปร *X* โดยกำหนดค่าให้เป็นคำว่า

*"the"* หรือ *P(X="the")* หรือเขียนอย่างง่ายเป็น *P(the)* และแทนชุดลำดับของคำจำนวน *N* คำด้วย $w_1,...,w_n$ หรือ $w_1^n$ สำหรับความน่าจะเป็นร่วมของแต่ละคำในลำดับใช้ $P(w_1,...,w_n)$ ซึ่งสามารถคำนวนความน่าจะเป็นจากลำดับทั้งหมดโดยจำแนกความน่าจะเป็นโดยใช้กฎลูกโซ่ (*Chain Rules*) ของความน่าจะเป็นได้จากสมการที่ 2.1

$$\begin{aligned} P(w_1,...,w_n) &= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1^2)...P(w_n \mid w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k \mid w_1^{k-1}) \end{aligned} \quad (2.1)$$

ลูกโซ่มาร์คอฟแสดงให้เห็นความเชื่อมโยงระหว่างการคำนวนความน่าจะเป็นร่วมของลำดับคำและคำนวนความน่าจะเป็นแบบมีเงื่อนไขของคำที่ขึ้นอยู่กับคำก่อนหน้า จากสมการที่ผ่านมาแสดงให้เห็นว่าเราสามารถคำนวนหาความน่าจะเป็นร่วมกันของลำดับทั้งหมดโดยผลรวมผลคูณของความน่าจะเป็นแบบมีเงื่อนไข ซึ่งอาจไม่สามารถคำนวนหาค่าความน่าจะเป็นที่แน่นอนจากลำดับคำที่มีความยาวได้เนื่องจากข้อความนั้นอาจไม่เคยเกิดขึ้น

จึงเกิดการประมาณนี้มีชื่อเรียกว่า การประมาณมาร์คอฟ (Markov assumption) ซึ่งจะทำให้ลดการเก็บข้อมูลจาก สถานะยกกำลัง *n* เหลือแค่สถานะยกกำลัง 2 เพราะสนใจข้อมูลเพียงแค่สองเวลา ดังแสดงในสมการที่ 2.2

$$P(w_1,...,w_n) = \prod_{i=1}^{n} P(w_i \mid w_{i-1}) \quad (2.2)$$

ด้วยเหตุนี้จึงเกิดแนวความคิดเอ็นแกรม โดยเอ็นแกรมเป็นลำดับต่อเนื่องที่จำนวน N ไอเท็ม (items) จากลำดับของคำทั้งหมดในประโยคหรือข้อความ [21, 25] ที่เรียกว่ารูปแบบภาษา (Language Model) ซึ่งไอเท็มสามารถเป็นหน่วยเสียง พยางค์ ตัวอักษรที่ประกอบรวมเป็นคำ หรือคำที่ประกอบรวมกันเป็นประโยค โดยพื้นฐานเอ็นแกรมโมเดลเป็นประเภทของความน่าจะเป็นของรูปแบบภาษาสำหรับคาดการณ์ items ถัดไป ที่ขึ้นอยู่กับลำดับคำทั้งหมด ซึ่งสามารถประมาณค่าได้โดยใช้การประมาณมาร์คอฟ โดยประมาณค่าจากคำสุดท้ายเพียงบางส่วน ยกตัวอย่างไบแกรมโมเดลจะประมาณค่าความน่าจะเป็นคำที่ขึ้นอยู่กับลำดับคำก่อนหน้า $P(w_n \mid w_1^{n-1})$ โดยใช้เพียงความน่าจะเป็นแบบมีเงื่อนไขของคำที่ผ่านมา $P(w_n \mid w_{n-1})$ จาก *P(the|its water is so transparent that)* จากนั้นจึงสามารถหาค่าความน่าจะเป็นของด้วยการประมาณค่าความน่าจะเป็นของ $P(the \mid that)$ โดยสามารถสรุปได้ว่าเอ็นแกรมจะสนใจตั้งแต่คำสุดท้ายย้อนกลับไปถึงคำที่ *N-1* ด้วยการประมาณค่าความน่าจะเป็นแบบมีเงื่อนไขของคำในลำดับถัดไปได้จากสมการที่ 2.3

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1}) \quad (2.3)$$

ยกตัวอย่างโดยใช้ข้อมูลจาก Berkeley Restaurant Project (BeRP) ซึ่งถอดความมาจากข้อมูลเสียงการสนทนา และนี่คือตัวอย่างของข้อมูลบางส่วนที่ถูกจัดให้อยู่ในรูปแบบข้อความ

```
I want to spend no more than seven dollars
I want to travel no more than three blocks
```

ตารางที่ 2.1 ตัวอย่างคำจำนวน 8 คำที่มีความถี่สูงในคลังข้อมูล

| i | want | to | eat | Chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|
| 2533 | 927 | 2417 | 746 | 158 | 1093 | 341 | 278 |

ตารางที่ 2.2 ความถี่แบบ Bigram ของคำ 8 คำนี้ในคลังข้อมูลตัวอย่าง

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| i | 5 | 827 | 0 | 9 | 0 | 0 | 0 | 2 |
| want | 2 | 0 | 608 | 1 | 6 | 6 | 5 | 1 |
| to | 2 | 0 | 4 | 686 | 2 | 0 | 6 | 211 |
| eat | 0 | 0 | 2 | 0 | 16 | 2 | 42 | 0 |
| chinese | 1 | 0 | 0 | 0 | 0 | 82 | 1 | 0 |
| food | 15 | 0 | 15 | 0 | 1 | 4 | 0 | 0 |
| lunch | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| spend | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

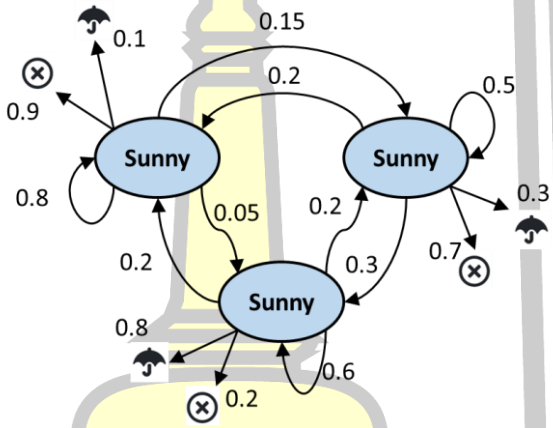
จากข้อมูลดังตารางที่ 2.1 และตารางที่ 2.2 จะสามารถคำนวนหาค่าความน่าจะเป็นของประโยค “*i want chinese food*” โดยกำหนดให้ค่าความน่าจะเป็นที่ i จะเป็นคำแรกของประโยค (<s>) ที่ 0.25 และ food เป็นคำสุดท้ายของประโยค (</s>) ที่ 0.68 จึงสามารถคำนวนได้ดังสมการต่อไปนี้

$$\begin{aligned} P(i\ want\ Chinese\ food) &= P(i \mid <s>)P(want \mid i)P(chinese \mid want) \\ &\quad P(food \mid chinese)P(</s> \mid food) \\ &= 0.25\ x\ 0.33\ x\ 0.006\ x\ 0.52\ x\ 0.68 \\ &= 0.00018 \end{aligned}$$

จากตัวอย่างเดียวกันนี้ หากจะทำการหาความน่าจะเป็นที่คำว่า “*please*” ที่จะเกิดต่อจากประโยค “*I want chinese food*” เราสามารถคำนวนเพื่อประมาณค่าความน่าจะเป็นได้โดยใช้โมเดล N-gram โดยหากกำหนดให้ N เท่ากับ 3 ดังนั้น เราจึงสามารถคำนวนโดยใช้คำเพียง 2 คำสุดท้ายของประโยคคือ *P*(*please*|*Chinese food*) เป็นต้น

2.3.3 แบบจำลองมาร์คอฟซ่อนเร้น

แบบจำลองมาร์คอฟซ่อนเร้น (Hidden Markov Model: HMM) [26] นั้นมีพื้นฐานมาจากลูกโซ่มาร์คอฟ โดยลูกโซ่มาร์คอฟมีประโยชน์เมื่อเราต้องการคำนวณความน่าจะเป็นสำหรับลำดับเหตุการณ์ที่สังเกตได้ อย่างไรก็ตาม พบว่าในบางกรณีนั้น สิ่งที่เราสนใจอาจถูกซ่อนอยู่ จึงทำให้ไม่สามารถสังเกตสิ่งเหล่านั้นโดยตรง จึงต้องใช้ปัจจัยอื่นมาเทียบเคียงด้วย ในขณะที่แบบจำลองมาร์คอฟซ่อนเร้น คือแบบจำลองมาร์คอฟที่เพิ่มสมมติฐานว่า ในแต่ละลำดับเหตุการณ์ที่สังเกต (Observations) ได้นั้น อาจมีสถานะที่ยังไม่ได้สังเกตการณ์ซ่อนอยู่ (Unobserved)




ภาพที่ 2.7 ตัวอย่างไดอะแกรมสถานะของสภาพอากาศแบบมาร์คอฟซ่อนเร้น

จากภาพที่ 2.7 เป็นแผนภาพไดอะแกรมสถานะของสภาพอากาศที่เป็นแบบมาร์คอฟซ่อนเร้น เนื่องด้วยมีผลลัพธ์ของสถานะสังเกต ดังตัวอย่างเช่น P(Umbrella|Sunny) = 0.1 ซึ่งหมายความว่าในวันที่เป็น Sunny มีความน่าจะเป็นที่ Umbrella (ในที่นี้หมายความว่าคนจะพกร่ม) มีค่าเท่ากับ 0.1 หรือ P(Umbrella|Rainy) = 0.8 นั่นหมายความว่าในวันที่เป็น Rainy มีความน่าจะเป็นที่คนจะพกร่มเป็น 0.8 เป็นต้น โดยในที่นี้จะเรียกความน่าจะเป็นในสถานะสังเกตนี้ว่า Emission probabilities

จากตัวอย่างในหัวข้อตัวแบบมาร์คอฟ หากต้องการพยากรณ์สภาพอากาศที่ยังไม่เกิดขึ้นจำเป็นที่จะต้องมีข้อมูลสภาพอากาศในอดีตก่อน แต่หากไม่ทราบข้อมูลในอดีต หรือข้อมูลดังกล่าวถูกปกปิดไว้ (Hidden) จะพยากรณ์ได้อย่างไร ซึ่งตัวแบบมาร์คอฟซ่อนเร้นนั้นสามารถใช้เหตุการณ์ที่เกี่ยวเนื่องกันเพื่อเทียบเคียงเพื่อใช้ในการพยากรณ์ได้โดยที่ไม่รู้เหตุการณ์ในอดีตว่าเป็นอย่างไรได้ โดยมีส่วนประกอบและปัญหาขั้นพื้นฐานของตัวแบบมาร์คอฟซ่อนเร้นดังต่อไปนี้

*ส่วนประกอบของตัวแบบมาร์คอฟซ่อนเร้น*

ส่วนประกอบต่างๆ ที่ตัวแบบมาร์คอฟซ่อนเร้นจะต้องมีคือ

$Q = \{q_1, q_2, \dots, q_n\}$ คือ เซตของสถานะ

$A = \{a_{11}, \dots, a_{ij}, \dots, a_{nn}\}$ คือ state transition probabilities

$O = \{o_1, o_2, \dots, o_n\}$ คือผลการสังเกต

$\pi = \pi_1, \pi_2, \dots \pi_n$ คือ initial state probabilities

$B = b_i(o_t)$ คือ emission probabilities

*ปัญหาขั้นพื้นฐานในแบบจำลองมาร์คอฟซ่อนเร้น*

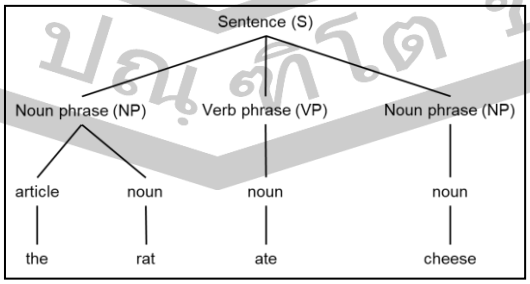
ปัญหาที่ 1 : ถ้าให้แบบจำลอง $\lambda = (A, B)$ และลำดับของการสังเกต $(O)$ จะ12 สามารถคำนวณหาค่าความน่าจะเป็นได้เท่าใด

ปัญหาที่ 2 : ถ้าให้ลำดับของการสังเกต $(O)$ และแบบจำลอง $\lambda$ จะหาลำดับสถานะที่ให้ค่าความเป็นไปได้มากที่สุดอย่างไร

ปัญหาที่ 3 : ถ้าให้ลำดับของการสังเกต $(O)$ และเซตของสถานะ จะสามารถสร้างแบบจำลอง $\lambda = (A, B)$ ที่ให้ค่าความน่าจะเป็นที่ดีที่สุดได้อย่างไร

2.3.4 การแจกแจงรูปประโยค

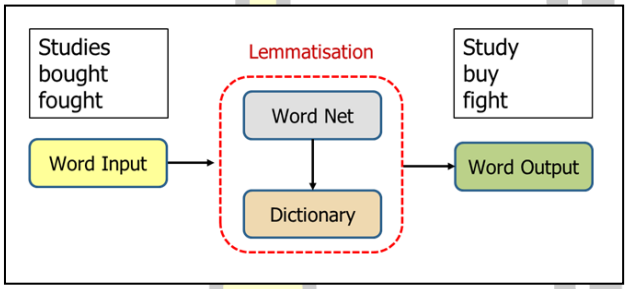
การแจกแจงรูปประโยค (Parsing / Parser) คือกระบวนการในการตรวจสอบโครงสร้างไวยากรณ์ของประโยค (Syntax Analysis) โดยทำหน้าที่ในการแจงแจกประโยคออกเป็นกลุ่มคำตามโครงสร้างไวยากรณ์ เช่น กลุ่มคำนาม กลุ่มคำกริยา และกลุ่มคำวิเศษณ์ เป็นต้น โดยเอาต์พุตที่ได้จะเป็นข้อมูลที่อยู่ในรูปแบบต้นไม้ที่แจกแจงออกมานั่นคือต้นไม้ไวยากรณ์ ยกตัวอย่างประโยค “the rat eat cheese” สามารถแจกแจงเป็นต้นไม้ไวยากรณ์ได้ดังตัวอย่างในภาพที่ 2.8




ภาพที่ 2.8 ตัวอย่างต้นไม้ไวยากรณ์ของประโยค

2.3.5 การลดรูปคำ

ในทางภาษาศาสตร์คอมพิวเตอร์กระบวนการลดรูปคำ (Lemmatization) [27] เป็นกระบวนการลดรูปคำให้อยู่ในรูปปกติแต่ยังคงความหมายเดิมไว้ มีประโยชน์ในการค้นหาคำที่มีรูปคำใกล้เคียงกัน ซึ่งกระบวนการลดรูปคำจะนำคำที่ต้องการลดรูป ไปทำการสืบค้นในคลังคำพจนานุกรม เพื่อหาคำให้อยู่ในรูปปกติหรือตามแบบพจนานุกรม ตัวอย่างเช่น คำว่า “*studies*” จะถูกลดรูปคำเป็น “*study*” โดยแสดงตัวอย่างดังภาพที่ 2.9




ภาพที่ 2.9 ตัวอย่างการลดรูปคำ

2.3.4 การสกัดสารสนเทศด้วยแม่แบบ

แนวความคิดของการสกัดสารสนเทศด้วยแม่แบบ (Template-based Information Extraction) เป็นเทคนิคที่ใช้ในงานการสกัดข้อมูลโดยเฉพาะ ซึ่งนำเสนอรูปแบบที่ใช้ในการสกัดข้อมูลส่วนต่างๆ จากข้อความต้นฉบับในขอบเขตที่เฉพาะเจาะจงตามที่ได้กำหนดไว้ในแม่แบบล่วงหน้า ซึ่งแนวคิดของการสกัดสารสนเทศด้วยแม่แบบ จะต้องกำหนดลักษณะที่เจาะจงในแต่ละเหตุการณ์ เช่น ใคร (Who), ทำอะไร (What), ที่ไหน (Where), เมื่อไหร่ (When) และทำไม (Why)

โดยทั่วไป แม่แบบที่ใช้ในการสกัดสารสนเทศจากข้อความมีอยู่ 4 ประเภท ได้แก่

*1) Cascaded finite-state transducers*

เป็นเทคนิคที่มีแนวคิดการแยกการประมวลผลออกเป็นขั้นๆ [28] เพื่อจะวิเคราะห์ข้อความในเอกสารในระดับกว้าง เพื่อดึงข้อมูลและความเชื่อมโยงในขอบเขตที่สนใจ โดยทั่วไปแบ่งการประมวลผลออกเป็น 5 ขั้นโดยได้แก่ *Complex Words*, *Basic Phrases*, *Complex Phrases*, *Domain Patterns* และ *Merging Structures*

*Complex Words*

เป็นขั้นตอนแรกของการประมวลผล โดยทำการระบุกลุ่มคำ รวมถึงชื่อบุคคล ชื่อบริษัท สถานที่ วันที่ เวลา เช่น “*set up*”, “*16 January 2015*” หรือ “*ABC Sportswear Co.*” เป็นต้น

*Basic Phrases*

ในขั้นตอนที่สองนี้ จะทำการระบุกลุ่มของคำนาม (noun groups) กลุ่มคำกริยา (verb groups) รวมถึงคำบุพบท (preposition) คำสันธาน (conjunction) สรรพนามเชื่อมความ (relative pronoun) ตัวอย่างกลุ่มคำที่ถูกระบุในขั้นตอนนี้ได้แก่ “*a joint venture*”, “*to produce*”, “*to*” และ “*approximately 5 kg*” เป็นต้น

*Complex Phrases*

เป็นการนำกลุ่มคำที่ได้จากขั้นตอน Basic Phrases มาทำการรวมและจำแนกเป็น Complex Phrases ในแต่ละโดเมน โดยพิจารณาจากคำหรือวลีที่อยู่เคียงข้างกัน กลุ่มคำนามที่เชื่อมกันด้วยคำสันธาน และมี “*of*” และ “*for*” อยู่หน้ากลุ่มคำนาม ยกตัวอย่างเช่น “*production of 10,000 shoes and 10,000 socks a month*” จัดเป็นโดเมน “*production*” เป็นต้น

*Domain Patterns*

จะเป็นการนำวลีพื้นฐานหรือวลีที่ซับซ้อนจากขั้นตอน Complex Phrases มาเป็นข้อมูลอินพุต (input) เพื่อเปรียบเทียบกับรูปแบบเหตุการณ์ที่สนใจ ยกตัวอย่างรูปแบบเหตุการณ์เช่น <Company><Start><Activity> in/on <Date> สำหรับสกัดสารสนเทศที่สนใจคือ บริษัทอะไร ดำเนินการอะไร และเมื่อไร่ เป็นต้น

*Merging Structures*

โดยในสี่ขั้นตอนก่อนหน้านี้เป็นการประมวลผลในระดับประโยค แต่ขั้นตอนสุดท้ายนี้เป็นการประมวลผลกับเอกสารทั้งหมด เพื่อดูข้อมูลในโดเมนต่างๆ หรือในกรณีที่มีโดเมนที่มีความสัมพันธ์กันโดเมนเหล่านั้นจะถูกรวมเข้าด้วยกัน เช่น “sportswear” มีความหมายเดียวกับ “shoes and socks”

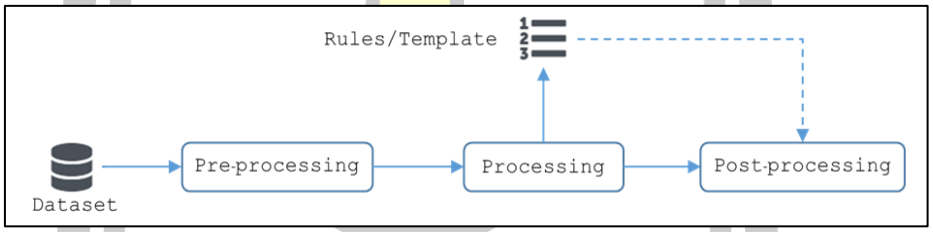
ตัวอย่างงานวิจัยที่มีการประยุกต์ใช้ Cascaded finite-state transducers ได้แก่ งานวิจัยของ Hobbs [29] ได้ใช้แนวคิดดังกล่าวในการสกัดสารสนเทศจากบทความชีวการแพทย์ โดย

สกัดข้อมูลชีวการแพทย์ในขอบเขตที่สนใจและจัดกลุ่มตามความสัมพันธ์ของข้อมูลเอนไซม์แต่ละตัว เช่นชื่อเอนไซม์ น้ำหนักโมเลกุล และส่วนประกอบทางเคมี

*2) Supervised learning approaches*

เป็นการประยุกต์ใช้แนวคิดของการเรียนรู้แบบมีผู้สอนเพื่อการสร้างกฎ (Rules) หรือโมเดล (Model) หรือรูปแบบ (Pattern) ที่แน่นอนไว้ก่อน ก่อนการนำเอาสิ่งเหล่านี้ไปสกัด (extract) องค์ความรู้ที่ต้องการจากข้อมูล [30]

โดยทั่วไปการประยุกต์ใช้แนวคิดแบบการเรียนรู้แบบมีผู้สอนในการสกัดสารสนเทศจากเอกสาร สามารถแบ่งการทำงานออกเป็น 3 ขั้นตอนได้แก่ การเตรียมข้อมูล (Pre-processing) การประมวลผลเพื่อสร้างแม่แบบ (Processing) และ การใช้งานแม่แบบ (Post-processing)




ภาพที่ 2.10 ขั้นตอนการสร้างแม่แบบด้วยแนวคิดของการเรียนรู้แบบมีผู้สอน

*การเตรียมข้อมูล (Pre-processing)*

ก่อนการดำเนินการคือการเตรียมความพร้อมของข้อมูลที่จะใช้เป็นข้อมูลตัวอย่างสำหรับการเรียนรู้พื่อสร้างแม่แบบ โดยการจัดการกับข้อมูลที่ไม่สมบูรณ์ ข้อมูลรบกวนหรือข้อมูลที่ไม่สอดคล้องกัน ตลอดจนถึงการผสานข้อมูล ลดรูปข้อมูล หรือการแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสมสำหรับการประมวลผลเพื่อสร้างแม่แบบ

*การประมวลผลเพื่อสร้างแม่แบบ (Processing)*

การดำเนินการคือการนำข้อมูลตัวอย่างที่ผ่านขั้นตอนการข้อมูล มาประมวลผลด้วยเพื่อสกัดสารสนเทศด้วยเทคนิคหรืออัลกอริธึมแบบมีผู้สอน (Supervised learning algorithm) เช่น ซัพพอร์ตเวคเตอร์แมชชีน (Support Vector Machines) โครงข่ายประสาทเทียม (Neural Networks) หรือต้นไม้ตัดสินใจ (Decision Trees) เป็นต้น ซึ่งผลลัพธ์ที่ได้อาจจะเป็นกฎ (Rules) หรือโมเดล (Model) หรือรูปแบบ (Pattern) ก็ได้

*การใช้งานแม่แบบ (Post-processing)*

ภายหลังการประมวลผลเพื่อสร้างแม่แบบ ในส่วนของการใช้งานแม่แบบจะเป็นขั้นตอนของการประเมินประสิทธิภาพของแม่แบบ เพื่อให้เกิดความมั่นใจว่า แม่แบบที่สร้างมานั้นสามารถนำไปใช้ในการสกัดสารสนเทสได้อย่างเหมาะสมและถูกต้อง โดยทั่วไปการวัดประสิทธิภาพของแม่แบบ สามารถทำได้การโดยผ่านการวิเคราะห์ในหลายๆ วิธี เช่น การวัดค่าความระลึก (Recall) ความแม่นยำ (Precision) เป็นต้น

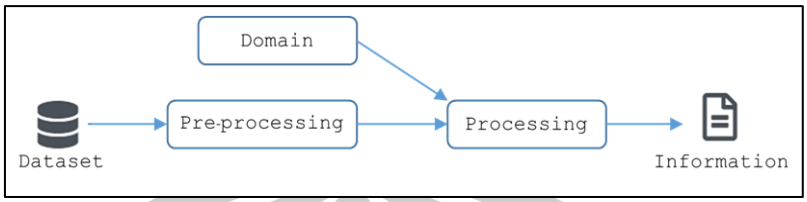
ตัวอย่างงานวิจัยที่มีการสร้างแม่แบบด้วยการประยุกต์ใช้แนวคิดของการเรียนรู้แบบมีผู้สอนมีหลายงานวิจัย เช่น ในงานของ Soderland [31] ได้เสนอการสร้างกฎสำหรับสกัดสารสนเทศสำหรับข้อความแบบกึ่งโครงสร้างและไม่มีโครงสร้าง โดยได้ใช้ข้อมูลในขอบเขตต่างๆ ในการสร้างกฎและทดลองเช่น กฎสำหรับสกัดจำนวนห้องนอน และราคาจากข้อมูลประกาศให้เช่าบ้าน กฎสำหรับสกัดข้อมูลวัน เวลา สถานที่ และวิทยากรจากประกาศสัมมนา และกฎสำหรับสกัดข้อมูลเมือง โปรแกรมประยุกต์ อายุงาน จากประกาศหางานด้านโปรแกรมเมอร์และนักวิเคราะห์ระบบ

และในงานของ Riloff และ Jones [32] ได้เสนอวิธีในการเรียนรู้คำศัพท์สำหรับการสกัดสารสนเทศ โดยใช้คำศัพท์ตัวอย่างในโดเมนที่ต้องการจำนวนหนึ่ง เพื่อสกัดหารูปแบบประโยคที่พบคำศัพท์ตัวอย่างเหล่านั้น จัดเก็บเป็นรูปแบบเพื่อใช้ในการสกัดคำศัพท์อื่นๆ ที่เป็นโดเมนเดียวกันในคลังข้อมูล ซึ่งใช้ข้อมูลในการสร้างรูปแบบและทดลองสองชุด ได้แก่ข้อมูลเว็บไซต์ของบริษัท และบทความข่าวการก่อการร้าย โดยได้สร้างกฎและสกัดหาคำศัพท์ในโดเมนต่างๆ เช่น ชื่อบริษัท สถานที่ตั้ง และอาวุธ และใช้ค่าความระลึก (Recall) และความแม่นยำ (Precision) ในการวัดประสิทธิภาพ

*3) Unsupervised learning approaches*

เป็นเทคนิคการเรียนรู้ของเครื่องแบบไม่มีผู้สอนหรือไม่จำเป็นต้องใช้ชุดข้อมูลตัวอย่างในการเรียนรู้ นิยมใช้เพื่อวิเคราะห์หารูปแบบหรือคุณลักษณะของที่ซ่อนอยู่ในชุดข้อมูล โดยใช้เทคนิคความน่าจะเป็นในการวิเคราะห์หาความคล้ายคลึงกันของข้อมูล หรือทำนายข้อมูลที่จะเกิดขึ้น [33]

โดยทั่วไปการประยุกต์ใช้แนวคิดแบบการเรียนรู้แบบไม่มีผู้สอนสำหรับการสกัดสารสนเทศจากเอกสาร สามารถแบ่งการทำงานออกเป็น 3 ขั้นตอนได้แก่ ได้แก่ การเตรียมข้อมูล (Pre-processing) การประมวลผล (Processing) และการประเมินประสิทธิภาพ (Evaluation)




ภาพที่ 2.11 ขั้นตอนการสกัดสารสนเทศด้วยแนวคิดของการเรียนรู้แบบไม่มีผู้สอน

*การเตรียมข้อมูล (Pre-processing)*

ขั้นตอนการเตรียมข้อมูล จะเป็นวิธีเดียวกันกับแนวคิดแบบการเรียนรู้แบบมีผู้สอน โดยแตกต่างกันที่ข้อมูลของแนวคิดการเรียนรู้แบบไม่มีผู้สอนจะไม่มีฉลากหรือคำตอบกำกับอยู่

*การประมวลผล (Processing)*

เป็นการนำข้อมูลที่ผ่านการเตรียมข้อมูล มาทำการประมวลผลร่วมกับโดเมนที่สนใจ เพื่อวิเคราะห์และสร้างแม่แบบที่ตรงความสนใจ ในขณะเดียวกันก็ใช้แม่แบบที่สร้างขึ้นในการสกัดสารสนเทศ โดยมีการประเมินผลแม่แบบเพื่อปรับปรุงตลอดจนกระทั่งการประมวลผลเสร็จสิ้น ตัวอย่างเทคนิคหรืออัลกอริธึมแบบไม่มีผู้สอน (Unsupervised learning algorithm) เช่น N-gram, Hidden Markov models (HMM) หรือ Apriori เป็นต้น

*การประเมินประสิทธิภาพ (Evaluation)*

คือการวัดประสิทธิภาพของขั้นตอนการประมวลผล โดยประเมินจากสารสนเทศที่สกัดได้ว่ามีความถูกต้องตรงความต้องการมากน้อยเพียงใด

ตัวอย่างงานวิจัยที่ใช้การประยุกต์แนวคิดการเรียนรู้แบบไม่มีผู้สอน เช่น ในงานของ Sekine [34] เป็นงานวิจัยเกี่ยวกับการสกัดข้อมูลตามความต้องการโดยอัตโนมัติ โดยใช้แนวคิดแบบไม่มีผู้สอนวิเคราะห์เพื่อประเมินว่าข้อความในเอกสารตรงหรือใกล้เคียงกับขอบเขตหรือรูปแบบที่ผู้ใช้กำหนดขึ้นหรือไม่ ถ้าได้จะทำการแยกข้อความที่สกัดได้เก็บรวบรวมเป็นตารางของแต่ละรูปแบบ

4) *Hybrid approaches*

เทคนิคนี้เป็นการผสมผสานหลายๆ เทคนิคการเรียนรู้ของเครื่องเพื่อเพิ่มประสิทธิภาพการทำงาน โดยมักประกอบไปด้วยการทำงานสองส่วน ส่วนแรกคือการนำเข้าข้อมูลดิบเพื่อสร้างข้อมูลที่เป็นส่วนกลาง ส่วนที่สองนำข้อมูลส่วนกลางเป็นข้อมูลนำเข้าเพื่อให้ได้ผลลัพธ์สุดท้าย [35]

ยกตัวอย่างงานวิจัยของ Feldman และคณะ [36] ได้ใช้เทคนิคแบบผสมผสาน (Hybrid) ระหว่างการคำนวนทางสถิติและฐานความรู้สำหรับสร้างแม่แบบเพื่อสกัดสารสนเทศที่เรียกว่า TEG (Trainable Extraction Grammar) โดยสารสนเทศที่สกัดได้คือเอนติทีและความสัมพันธ์ในระดับประโยค เทคนิคฐานความรู้คือการสร้างกฎที่ใช้สำหรับสกัดไวยากรณ์ด้วยมือ ส่วนเทคนิคทางสถิติจะใช้คลังข้อมูลในการเรียนรู้ค่าความน่าจะเป็น ซึ่งผลการทดลองที่ได้จากการใช้เทคนิคผสมผสมผสานนี้มีประสิทธิภาพดีกว่าการใช้เทคนิคอย่างใดอย่างหนึ่ง

ในงานนี้ได้ใช้แนวคิดการประยุกต์ใช้แนวคิดของการเรียนรู้แบบมีผู้สอนเป็นวิธีในการดำเนินการสร้างแม่แบบหรือกฎเชิงไวยากรณ์ ดังนั้นอัลกอริธึมเรียนรู้มักถูกนำมาใช้ในการสร้างแม่แบบจากกลุ่มข้อมูลตัวอย่าง จากนั้นกฎเชิงไวยากรณ์ดังกล่าวสามารถนำมาใช้ในการสกัดข้อมูลจากกลุ่มข้อมูลใหม่ ซึ่งความถูกต้องของกฎเชิงไวยากรณ์มักเพิ่มขึ้นตามจำนวนของกลุ่มข้อมูลตัวอย่างที่ใช้ในการเรียนรู้ [37]

**2.4 การแทนองค์ความรู้**

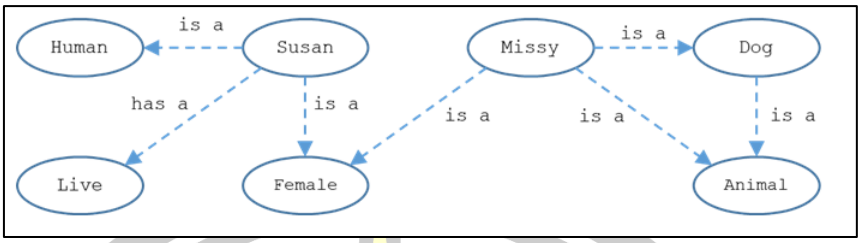
การแทนองค์ความรู้ (Knowledge Representation) คือการจัดการองค์ความรู้ที่มีอยู่หรือสกัดมาได้ มาแสดงให้อยู่ในรูปแบบที่เฉพาะเจาะจงหรือเหมาะสมกับองค์ความรู้ที่ต้องการแสดง โดยทั่วไปการแทนองค์ความรู้จะมีหลายวิธี [38] ได้แก่

*การแทนองค์ความรู้เชิงตรรกะ (Logical Knowledge Representation)*

การแทนองค์ความรู้เชิงตรรกะ เป็นการนำข้อเท็จจริงหรือข้อกำหนดที่สร้างขึ้น มาเข้าสู่กระบวนการทางตรรกะ เพื่อประมวลผลหาข้อเท็จจริงของสิ่งที่สนใจ โดยผลลัพธ์ที่ได้คือผลของการอนุมานเช่นจริง (True) หรือเท็จ (False) ยกตัวอย่างเช่น หากต้องการทราบว่าสมชายเปียกหรือไม่เปียก เราสามารถพิจารณาได้จากข้อเท็จจริงได้แก่ ฝนตกหรือไม่ หรือสมชายอยู่ในบ้านหรือนอกบ้าน เป็นต้น

*การแทนองค์ความรู้เชิงเครือข่าย (Network Knowledge Representation)*

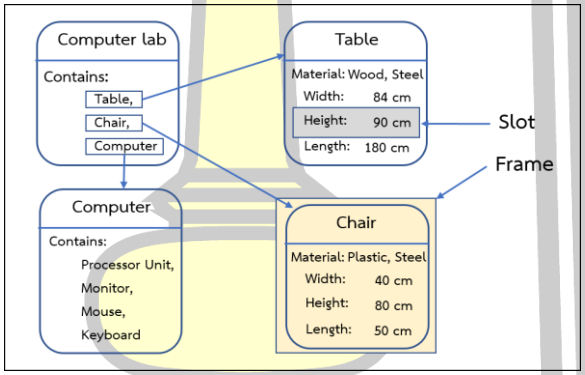
การแทนองค์ความรู้เชิงเครือข่ายจะสนใจไปที่วัตถุ และความสัมพันธ์ระหว่างวัตถุ โดยแสดงในรูปแบบกราฟระบุทิศทาง ประกอบด้วยโหนด (Nodes) แทนการกระทำ เหตุการณ์ หรือวัตถุ โดยมีเส้นเชื่อม (Arcs) ที่ใช้เชื่อมโยงความสัมพันธ์ระหว่างโหนดต่างๆ




ภาพที่ 2.12 ตัวอย่างการแทนองค์ความรู้เชิงเครือข่าย

*การแทนองค์ความรู้เชิงโครงสร้าง* (*Structured Knowledge Representation*)

การแทนองค์ความรู้เชิงโครงสร้างจะแบ่งองค์ความรู้ออกเป็นโครงสร้าง (Frame) ซึ่งในแต่ละโครงสร้างจะแบ่งออกเป็นส่วนย่อยๆ เรียกว่าสล็อต (Slot) และในแต่ละสล็อตจะจัดเก็บคุณลักษณะและค่าของสล็อต โดยแต่ละโครงสร้างอาจมีความสัมพันธ์ระหว่างกันได้




ภาพที่ 2.13 ตัวอย่างการแทนองค์ความรู้เชิงโครงสร้าง

อย่างไรก็ตาม ยังมีวิธีการแทนความรู้ที่ได้รับความนิยม ได้แก่การแทนองค์ความรู้เชิงระเบียบวิธี (Procedural Knowledge Representation) เนื่องจากสามารถแสดงองค์ความรู้ให้อยู่ในรูปแบบที่ง่ายต่อการอธิบายและทำความเข้าใจ ซึ่งการแทนองค์ความรู้เชิงระเบียบวิธีสามารถอธิบายและยกตัวอย่างได้ดังต่อไปนี้

การแทนองค์องค์ความรู้เชิงระเบียนวิธีหมายถึง ระเบียบ ขั้นตอน หรือลำดับการทำงานเพื่อให้การทำงานบรรลุวัตถุประสงค์หรือแก้ไขปัญหาต่างๆ และแต่ละการทำงานจะเชื่อมโยงกันด้วย *“เงื่อนไข”* (Condition) ซึ่งเงื่อนไขมีไว้เพื่อให้แน่ใจว่าการทำงานนั้น จะถูกดำเนินการตามระเบียบหรือลำดับที่ถูกต้อง

ยกตัวอย่างลำดับการทำงาน X → Y → Z เราสารมารถตรวจสอบได้ว่า การทำงานของลำดับการทำงานชุดนี้ มีแนวโน้มที่จะบรรลุวัตถุประสงค์ทั้งหมดได้หรือไม่ โดยการตรวจสอบการทำงานที่ลำดับต่างๆ เช่น X และ Y ว่าทำงานเป็นไปตามเงื่อนไขหรือระเบียบวิธีหรือไม่ ยกตัวอย่างเช่น

```
IF X Follow Conditions THEN Y
IF Y Follow Conditions THEN Z
```

สำหรับการนำการแทนองค์ความรู้เชิงระเบียบวิธีมาใช้ในงานวิจัยนี้ จะเป็นการแสดงลำดับการทำงานและเงื่อนไขการทำงานต่างๆ ของ Actor เช่น

```
IF (Have authentication) THEN... ELSE need login to the
system
THEN...
```

**2.5 การประเมินประสิทธิภาพ**

การประเมินประสิทธิภาพ จะเป็นการประเมินโดยการนำผลลัพธ์ที่โมเดลหรือโปรแกรมทำนาย เปรียบเทียบกับผลเฉลยหรือสิ่งที่เกิดขึ้นจริงด้วยตารางความคลาดเคลื่อน (Confusion Matrix) [21] ซึ่งเป็นรูปแบบตารางที่เฉพาะเจาะจงที่ช่วยให้เห็นผลการทำงานของโมเดลหรือโปรแกรมโดยการคำนวนค่าความระลึก ค่าความแม่นยำ และค่า F1

ตารางที่ 2.3 ตารางความคลาดเคลื่อน

| | | predicted | |
|---|---|---|---|
| | | positive | negative |
| Actual | positive | *TP* | *FN* |
| | negative | *FP* | *TN* |

จากตารางที่ 2.3 จะแสดงตารางความคลาดเคลื่อนเพื่อความสะดวกในการคำนวณ ซึ่งจะแทนที่ด้วยจำนวนผลลัพธ์ที่โมเดลหรือโปรแกมทำนาย โดยเปรียบเทียบกับผลเฉลยหรือสิ่งที่เกิดขึ้นจริง โดยที่

True Positive (TP) คือ สิ่งที่โมเดลทำนายว่าจริง และผลเฉลยบอกว่ามันจริง

True Negative (TN) คือ สิ่งที่โมเดลทำนายว่าไม่จริง และผลเฉลยบอกว่ามันไม่จริง

False Positive (FP) คือ สิ่งที่โมเดลทำนายว่าจริง แต่ผลเฉลยบอกว่าไม่จริง

False Negative (FN) คือ สิ่งที่โมเดลทำนายว่าไม่จริง แต่ผลเฉลยบอกว่าจริง

2.7.1 ค่าความแม่นยำ

ค่าความแม่นยำ (Precision) เป็นอัตราส่วนของจำนวนครั้งที่โมลเดลทำนายว่าจริงและตรงกับผลเฉลย กับจำนวนครั้งที่โมลเดลทำนายว่าจริงทั้งหมด สามารถแสดงได้ดังสมการที่ 2.4

$$Precision = \frac{TP}{TP + FP} \tag{2.4}$$

2.7.2 ค่าความระลึก

ค่าความระลึก (Recall) เป็นอัตราส่วนของจำนวนครั้งที่โมลเดลทำนายว่าจริงและตรงกับผลเฉลย กับจำนวนผลเฉลยที่บอกว่าจริงทั้งหมด สามารถคำนวนได้ดังสมการที่ 2.5

$$Recall = \frac{TP}{TP + FN} \tag{2.5}$$

2.7.3 ค่า F-measure

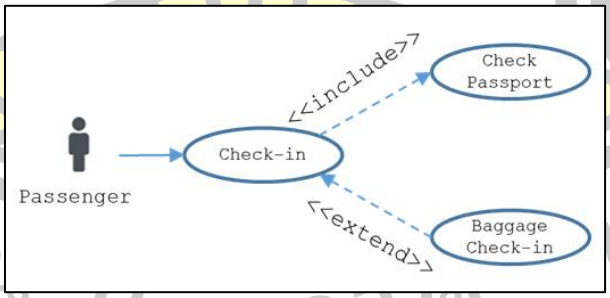
ค่า F-measure หรือ F1 เป็นการวัดประสิทธิภาพของระบบโดยใช้ค่าเฉลี่ยถ่วงน้ำหนักของค่าความแม่นยำและค่าความระลึก โดยมีประสิทธิภาพสูงสุดที่ 1 และต่ำสุดที่ 0 โดยสามารถหาค่าประสิทธิภาพได้ดังสมการที่ 2.6

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.6}$$

**2.6 แผนภาพที่แสดงการทำงานของผู้ใช้ระบบ**

แผนภาพที่แสดงการทำงานของผู้ใช้ระบบ (Use Case Diagram) เป็นส่วนหนึ่งของ UML (Unified Modeling Language) ซึ่งเป็นภาษาที่ใช้อธิบายแบบจำลองต่างๆ หรือใช้รูปภาพมาตรฐานเป็นสัญลักณ์แทนภาษา นิยมใช้สำหรับการสร้างแบบจำลองเชิงวัตถุ โดยแผนภาพที่แสดงการทำงานของผู้ใช้ระบบคือ แผนภาพที่แสดงการทำงานของผู้ใช้ระบบ (Actor) หน้าที่หรือการทำงานในระบบ (Use Case) และความสัมพันธ์ระหว่างผู้ใช้ระบบและการทำงาน (Relationship) โดยมีส่วนประกอบและสัญลักษณ์ดังต่อไปนี้

- Actor คือ ผู้เกี่ยวข้องกับระบบ ในที่นี้จะใช้สัญลักษณ์รูปคน พร้อมทั้งเขียนชื่อของ Actor ไว้ด้านล่างของสัญลักษณ์
- Use Case คือ หน้าที่ที่ระบบหรือผู้เกี่ยวข้องต้องกระทำ ใช้สัญลักษณ์รูปวงรี พร้อมทั้งเขียนชื่อ Use Case อยู่ภายในซึ่งปกติเป็นคำกริยาหรือกริยาวลี
- Connection เส้นที่ลากเชื่อมต่อระหว่าง Actor กับ Use Case ที่มีปฏิสัมพันธ์กัน มีสัญลักษณ์เป็นเส้นตรงไม่มีหัวลูกศร
- Relationship เป็นความสัมพันธ์ของ Use Case โดยแบ่งเป็น 3 ประเภทได้แก่
    1) Generalization หรือ เป็นความสัมพันธ์ที่ไม่มีการถ่ายทอดคุณลักษณะใดๆ เป็นความสัมพันธ์แบบทั่วไปที่แสดงลำดับการทำงาน
    2) Extend เป็นความสัมพันธ์แบบขยายหรือเพิ่มเติม เกิดขึ้นในกรณีที่บาง Use Case นั้นเกิดเหตุการณ์ที่ต้องทำให้เกิด Use Case อื่นเพิ่มเติม ในการพัฒนาซอฟต์แวร์จะหมายถึง *“ฟังก์ชันหรือหน่วย”* ของการทำงานที่ต้องมี แม้ว่าจะถูกใช้หรือไม่ก็ตาม
    3) Include คือ Use Case หนึ่ง อาจจะถูกผนวกเข้าไปรวมกับกิจกรรมของอีก Use Case หนึ่ง ในการพัฒนาซอฟต์แวร์จะหมายถึง “ฟังก์ชันหรือหน่วย” ที่ต้องถูกดำเนินการก่อน ก่อนที่ฟังก์ชันหรือหน่วยอื่นจะถูกดำเนินการ




ภาพที่ 2.14 ตัวอย่าง Use Case Diagram

## 2.7 งานวิจัยที่เกี่ยวข้อง

ในหลายๆ เอกสารความต้องการของระบบนั้นเขียนในรูปแบบเอกสารข้อความ (Textual software requirements) ดังนั้น เมื่อไม่นานมานี้จึงมีการประยุกต์เทคนิคทางด้าน NLP ในงานด้าน

วิศวกรรมซอฟท์แวร์ (Software Engineering) หรือวิศวกรรมความต้องการ (Requirement Engineering) โดยมีนักวิจัยหลายท่าน [39] ยืนยันว่าการประมวลผลภาษาธรรมชาติได้เข้ามามีบทบาทสำคัญในการพัฒนาระบบคอมพิวเตอร์หรือซอฟต์แวร์ และปัญหาที่สำคัญอย่างหนึ่งเกิดจากการที่ ความต้องการของระบบ (software requirements) ตลอดจนข้อกำหนดความต้องการของระบบ (software specific requirements) มักอยู่ในรูปแบบเอกสารข้อความ [18] หากเอกสารเหล่านั้นมีจำนวนไม่มาก การวิเคราะห์เพื่อทำความเข้าใจและใช้ในการออกแบบระบบ ก็จะไม่ใช่ปัญหา แต่ปัจจุบัน เอกสารที่แสดงความต้องการระบบ มักจะมีขนาดใหญ่และจำนวนมากตามขนาดของระบบที่ต้องการพัฒนา ดังนั้น จากสาเหตุนี้จึงกลายเป็นแรงจูงใจในการประยุกต์การประมวลผลภาษาธรรมชาติเพื่อใช้ในการวิเคราะห์ข้อความในงานวิศวกรรมความต้องการ [40]

Fantechi และคณะ [41] ได้นำเสนอการพัฒนากระบวนการในการทำความเข้าใจลักษณะประโยคที่ปรากฏในเอกสารข้อความบนพื้นฐานของการประมวลผลภาษาธรรวมชาติ หรือ NLP เพื่อแสดงความเชื่อมโยงหรือความสัมพันธ์ระหว่างระบบย่อยๆ ในระบบ

นอกจากนั้น Goldin และ Berry [42] ได้กล่าวว่าการระบุชื่อเฉพาะ (Name Entity) เช่น ชื่อตำแหน่ง ชื่อหน่วยงาน เป็นต้น ล้วนเป็นปัญหาสำคัญในการวิเคราะห์ความต้องการของระบบ เพราะชื่อเฉพาะดังกล่าวจึงมักพบในเอกสารข้อความที่รวบรวมจากผู้รับบริการและผู้ใช้งานระบบ ดังนั้น ชื่อเฉพาะที่ปรากฏในเอกสารความต้องการของระบบอาจจะมีความแตกต่าง ซึ่งสาเหตุดังกล่าวจะทำให้การวิเคราะห์ข้อจำกัดของระบบจากเอกสารเหล่านั้นเกิดความผิดพลาดได้

ในปัญหาที่ใกล้เคียงกัน Ambriola และ Gervasi [43] ได้เสนอ บนเทคนิคที่เรียกว่า *“Circe”* ที่อยู่บนพื้นฐานการประมวลผลภาษาธรรมชาติเช่นเดียวกัน โดยระบบนี้จะช่วยรวบรวม (Collect) คัดเลือก (Select) และตรวจสอบเอกสารความต้องการของระบบ (Verify) ยิ่งไปกว่านั้น Circe ยังสามารถใช้เพื่อการสกัดชื่อเฉพาะออกจากข้อความในเอกสารได้ด้วย จากนั้นจึงจะสร้างโมเดลของระบบอย่าง่ายสำหรับอธิบายความต้องการของระบบ รวมทั้งการตรวจสอบความถูกต้องของโมเดลที่จะใช้ประกอบการพัฒนาระบบและสร้างรายงานตัวชี้วัดการทำงาน

Huyck และ Abbas [44] นำเสนอและอภิปรายถึงข้อดีบางส่วนของเทคนิคด้านการประมวลผลภาษาธรรมชาติ ที่สามารถนำไปใช้กับงานวิศวกรรมความต้องการ หรือ Requirement Engineering โดยนักวิจัยเน้นย้ำว่าความคลุมเครือหรือกำกวม และการรวบรวมเอกสารที่ไม่ครอบคลุมต่อการพัฒนาระบบเป็น 2 ปัญหาสำคัญในงานวิศวกรรมความต้องการ ซึ่งปัญหาเหล่านี้มี

ความสัมพันธ์กับรูปแบบเอกสารและสามารถทำให้เกิดความเข้าใจผิดในการออกแบบระบบ ซึ่ง Huyck และ Abbas ได้ลงความเห็นว่าเทคนิค ด้านการประมวลผลภาษาธรรมชาติสามารถประยุกต์ใช้ เพื่อช่วยในการตรวจพบปัญหาและแนวทางการแก้ปัญหา นอกจากนี้ ในงานวิจัยของ Flores [45] ยังได้นำเสนอการเพิ่มเทคนิคด้านการตีความหรือให้ความหมาย (Semantic Technique) เพื่อใช้ในการลดความกำกวมด้านภาษาในระหว่างการวิเคราะห์ความต้องการของระบบ โดยเฉพาะความต้องการของระบบที่อยู่ในรูปแบบเอกสารข้อความขนาดใหญ่

ในงานวิจัยของ MacDonell และคณะ [15] ได้มุ่งเน้นการใช้เทคนิคด้านการประมวลผลภาษาธรรมชาติสำหรับงานวิศวกรรมซอฟต์แวร์ โดยจะครอบคลุมการนำเสนอวิธีการตรวจสอบความไม่สอดคล้องของคำศัพท์ในเอกสารและสกัดองค์ความรู้จากเอกสารความต้องการ ซึ่งในงานวิจัยของ Kof [16] ก็ได้เสนอแนวคิดและวิธีการที่สอดคล้องกับแนวคิดและการทำงานของ MacDonell และคณะ [15] แต่มีการปรับปรุงกระบวนการของการประมวลผลภาษาธรรมชาติในบางจุด เพื่อทำให้ครอบคลุมไปถึงการตรวจสอบเงื่อนไขที่เกี่ยวข้องกับการทำงานในระบบ

นอกจากนั้น ในงานวิจัยของ Ibrahim และ Ahmad [46] ได้เสนอวิธีและเครื่องมือเพื่อให้กระบวนการในการวิเคราะห์ความต้องการของระบบให้ดีขึ้น และยังสามารถแสดงความต้องการในรูปแผนภาพอย่างง่ายที่ใช้แสดงความต้องการ ซึ่งกระบวนการทั้งหมดล้วนเป็นการประยุกต์ใช้เทคนิคด้านการประมวลผลภาษาธรรมชาติทั้งสิ้น

Yang และคณะ [17] ยังคงให้ความคิดเห็นถึงปัญหาหลักๆ ของงานด้านการประยุกต์การประมวลผลภาษาธรรมชาติในการวิเคราะห์หรือสกัดความต้องการของระบบจากเอกสารความต้องการ ว่าปัญหาหลักยังคงเป็นเรื่องความกำกวมทางภาษา เนื่องจาก แม้ว่าโดยทั่วไปการรวมรวมเอกสารความต้องการจะมีรูปแบบที่เป็นมาตรฐาน แต่การรวบรวมความต้องการของระบบยังมีความยืดหยุ่นมาก ไม่ว่าจะเป็นเรื่องของภาษาที่ใช้ หรือโครงสร้างในการแสดงประโยค เป็นที่มาของความคลุมเครืออันไม่พึงประสงค์ และทำให้เกิดความตีความที่แตกต่างกันออกไป ปัญหาหลักอย่างหนึ่งคือเรื่องการใช้คำสรรพนาม ดังนั้น Yang และคณะ [17] จึงพยายามหาแนวทางในการควบคุมการตีความที่ผิดพลาดดังกล่าว

หลังจาก Yang และคณะ [17] นำเสนองานวิจัยของตนได้ไม่นาน Lindblad และ Wester [48] ได้นำเสนอการสร้างและใช้กฎทางภาษาอย่างง่าย (Language Rules) ที่สร้างจากเอกสาร

ข้อความที่แตกต่างกัน ก็สามารถนำมาใช้ในการสกัดความต้องการระบบ (software requirement specification) จากเอกสารความต้องการของระบบเช่นเดียวกัน

ในปี 2010 Ibrahim และ Ahmad [46] ได้สร้างแบบจำลองให้อยู่ในรูปแบบ Class diagram จากการสกัดจากความต้องการที่เป็นข้อความ เพื่อช่วยลดปัญหาการวิเคราะห์ความต้องการที่มีขนาดใหญ่ ด้วยเครื่องมือที่เรียกว่า Requirements Analysis and Class Diagram Extraction (RACE) โดยใช้เทคนิคการประมวลผลภาษาธรรมชาติได้แก่ POS Tagger, Parsing และ WordNet เพื่อสกัดคำนาม (กลุ่มคำนาม) คำกริยา และความสัมพันธ์ระหว่างกันโดยใช้ฐานความรู้เฉพาะทางและกระบวนการจัดแยกประเภท โดยจัดประเภทของความสัมพันธ์ออกเป็น 3 ประเภทได้แก่คลาส คุณลักษณะ และความสัมพันธ์ ซึ่งงานวิจัยนี้ได้ใช้ความต้องการของระบบห้องสมุดเป็นกรณีศึกษา

ปี 2012 Herchi และ Abdessalem [47] ได้สร้างเครื่องมือเพื่อช่วยจัดการกับเอกสารความต้องการขนาดใหญ่ โดยจัดเก็บในรูปแบบ XML และแสดงในรูปแบบ UML Class diagram ซึ่งใช้เทคนิคการประมวลผลภาษาธรรมชาติได้แก่ Tokenizer, POS tagger และ Parsing เพื่อแยกคำนามหรือกลุ่มคำนามออกจากประโยค และใช้กฎในการจัดประเภทว่าคำนามเหล่านั้นเป็นคลาส คุณลักษณะ หรือความสัมพันธ์ระหว่างกัน และใช้เทคนิคฐานความรู้เฉพาะทางเพื่อกรองผลลัพธ์ให้มีประสิทธิภาพมากขึ้น โดยผลการทดลองนั้นได้ใช้ค่าความระลึกและค่าความแม่นยำในการวัดประสิทธิภาพ

และปี 2014 Macdonell และคณะ [15] ได้พัฒนาเครื่องมือที่ช่วยให้นักวิเคราะห์ระบบสามารถเลือกและยืนยันเงื่อนไขหรือข้อกำหนดที่มีความเกี่ยวข้องกับระบบ ด้วยการแยกและจัดประเภทของวัตถุที่น่าสนใจจากเอกสารความต้องการ โดยใช้เทคนิคการประมวลผลภาษาธรรมชาติคือการทำ Parsing เพื่อแยกกลุ่มคำนามออกจากเอกสารความต้องการ และใช้ Term Management System เพื่อให้ผู้ใช้จัดประเภทของกลุ่มคำนามว่าเป็น Function, Entity หรือ Attribute หรือตัดกลุ่มคำนามที่ไม่จำเป็นหรือไม่มีความเกี่ยวข้องกับระบบทิ้งไปได้

พหุน ปณ ฑิโต ชีเว

# บทที่ 3

# วิธีดำเนินการวิจัย

ในบทนนี้จะเป็นการแสดงและอธิบายกระบวนการในการสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยแสดงรายละเอียดดังต่อไปนี้

## 3.1 ชุดข้อมูล

ชุดข้อมูล (Dataset) ที่ใช้ในงานวิจัยนี้จะมี 2 ชุดคือ ชุดข้อมูลที่ใช้ในการสร้างกฎเพื่อวิเคราะห์โครงสร้างภาษา และชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ (Textual Software Requirements) ใช้สำหรับการประเมินประสิทธิภาพของกระบวนการ โดยในหัวข้อนี้จะอธิบายถึงเฉพาะชุดข้อมูลที่ใช้ในการสร้างกฎ เพื่อใช้สำหรับขั้นตอนการสกัด *Actor* และ *Action*

ชุดข้อมูลที่ใช้ในการสร้างกฎเพื่อวิเคราะห์โครงสร้างภาษาจะใช้คลังข้อมูลมาตรฐานที่นิยมใช้ในการประมวลผลภาษาธรรมชาติ โดยคลังข้อมูลที่ใช้นั้นมีชื่อว่า *"The Stanford Natural Language Inference (SNLI) Corpus"* [48] โดยเป็นคลังข้อมูลขนาดใหญ่ที่รวบรวมประโยคภาษาอังกฤษที่เขียนในรูปแบบภาษาธรรมชาติ ซึ่งเหตุผลที่เลือกใช้คลังข้อมูลของ *SNLI* คือ

1) มีความหลากหลายของรูปแบบการนำเสนอประโยค (Writing)
2) เพื่อหลีกเลี่ยงปัญหาการที่โมเดลจดจำรูปแบบของข้อมูลที่ใช้ในการเรียนรู้ (Overfitting) กล่าวคือ กฎใดๆ ที่สกัดได้เป็นการสกัดจากคลังข้อมูลแบบทั่วไป ที่ไม่ใช่คลังข้อมูลที่เฉพาะเจาะจงเช่นเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

ในขั้นต้น ได้เลือกใช้เอกสารที่เป็นประโยคความเดียว (Simple Sentence) เพื่อใช้ในการสร้างกฎในการสกัด *Actor* และประโยคความรวม (Compound Sentence) สำหรับใช้ในการสร้างกฎในการแยกประโยคความรวม

ชุดข้อมูลที่ใช้ในการสร้างกฎเพื่อวิเคราะห์โครงสร้างภาษาในงานวิจัยนี้ จะอยู่ในรูปแบบประโยคภาษาอังกฤษจำนวน 300,000 ประโยค และประโยคที่ใช้ในการทดสอบกฎในการสกัด *Actor* ที่สร้างขึ้นจำนวนไม่น้อยกว่า 1,000 ประโยค

```
The children are walking in the afternoon.
People listening to a choir in a Catholic church.
A man buying a new laptop.
Three guys and a girl are all jumping in a pool together.
The woman is wondering if she left her car open.
A group of people point forwards while performing some kind of act.
Two woman beach volleyball players jump during a match.
```

ภาพที่ 3.1 ตัวอย่างประโยคจากคลังข้อมูล SNLI

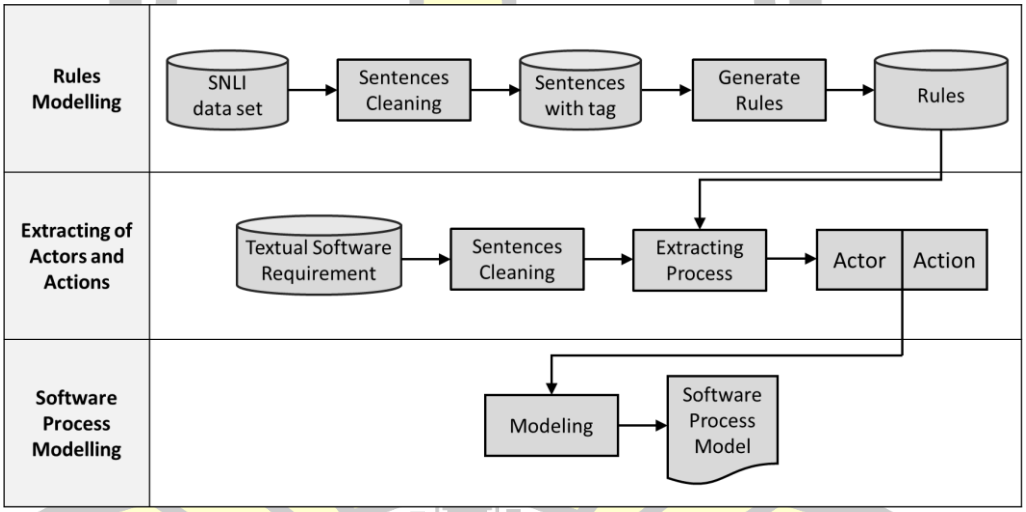
## 3.2 กรอบการดำเนินการ

การดำเนินการในงานวิจัยนี้จะแบ่งออกเป็น 3 ส่วนได้แก่

1) การสร้างกฎ (Rules Modelling)

2) การสกัด *Actor* และ *Action* (Extracting of Actors and Actions) และ

3) การสร้างเป็นแบบจำลองกระบวนการซอฟต์แวร์ (Software Process Modelling)




ภาพที่ 3.2 กรอบการดำเนินงานวิจัย

## 3.3 การสร้างกฎ (Rules Modelling)

ในขั้นตอนนี้จะอธิบายถึงการสร้างกฎเพื่อใช้สำหรับขั้นตอนการสกัด *Actor* และ *Action* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยกฎที่ใช้จะแบ่งออกเป็น 3 กลุ่มตามจุดประสงค์การใช้งานคือ

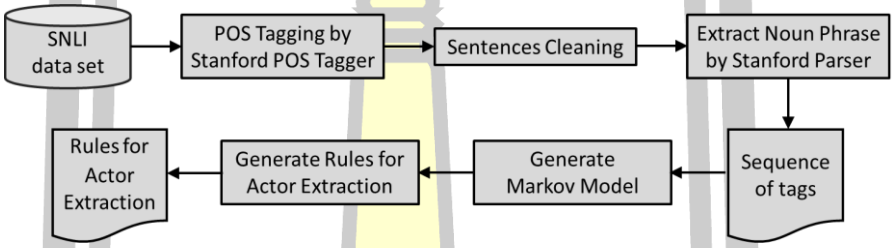
(1) กฎในการสกัด *Actor*

(2) กฎในการแทนคำสรรพนาม

(3) กฎในการแยกประโยคความรวม

3.3.1 กฎในการสกัด Actor

ในการสร้างกฎเพื่อการวิเคราะห์โครงสร้างภาษา ในกลุ่มของกฎในการสกัด *Actor* ซึ่ง *Actor* อาจจะเป็น “*บุคคล* (*person*)”, “*ระบบย่อย* (*sub-system*)”, “*หน่วยงาน* (*unit*)”, หรือ “*แผนก* (*department*)” ดังนั้นคำหรือกลุ่มคำเหล่านี้ มักจะอยู่ในรูปแบบคำหรือกลุ่มคำนาม (noun) ซึ่งสามารถแสดงกรอบการประมวลผลการสร้างกฎในการสกัด *Actor* ดังภาพที่ 3.3 และการสร้างกฎในการสกัด *Actor* มีกระบวนการดังนี้

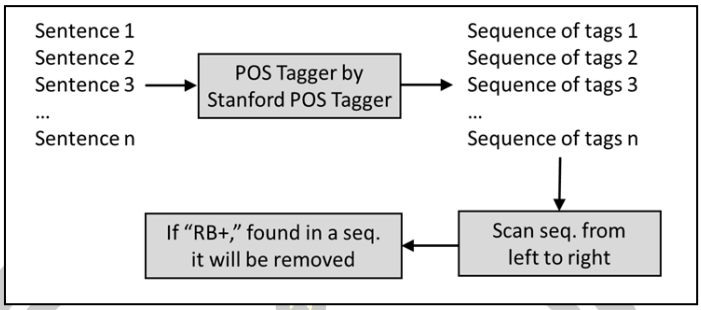



ภาพที่ 3.3 กรอบการประมวลผลการสร้างกฎในการสกัด *Actor*

*ขั้นตอน 1*: *การทำความสะอาดประโยค* (*Sentence Cleaning*)

ขั้นตอนนี้เป็นการเตรียมความเหมาะสมของประโยคที่จะนำไปสร้างกฎในการสกัด *Actor* โดยในขั้นตอนนี้ใช้ ชุดข้อมูล *SNLI* [48] สำหรับการสร้างกฎ ในลำดับแรกจะเป็นการทำ *POS Tagging* เพื่อระบุประเภทของคำด้วยการติด tag ประเภทของคำในประโยค โดยส่วนของประโยคที่จะถูกตัดออกจะเป็นคำที่อยู่ในรูปแบบ ***“คำวิเศษณ์ (Adverb: RB) + ,”*** ออกไป เนื่องจากรูปแบบดังกล่าวในประโยค มักจะเป็นการเกริ่นนำประโยคเพื่อให้ประโยคมีความสละสลวย หรือแสดงความต่อเนื่องจากประโยคก่อนหน้า แต่ในส่วนนี้ไม่ใช่ส่วนของประโยคที่มีความสำคัญในการนำเอามาสร้างกฎเพื่อการสกัด *Actor* เช่น “*Afterwards, the manager will approve the letter.*” จากประโยคตัวอย่าง จะเห็นได้ว่า “*Afterwards,*” เป็นส่วนของการเกริ่นนำประโยค แต่ไม่ใช่ส่วนที่เป็นความสำคัญในการนำเอามาสร้างกฎเพื่อการสกัด *Actor*

การทำงานของการทำความสะอาดประโยคจะสามารถแสดงได้ดังภาพที่ 3.4 และตัวอย่างสามารถแสดงได้ดังภาพที่ 3.5




ภาพที่ 3.4 กรอบการประมวลผลเพื่อการทำความสะอาดประโยค

| Sentence | Text |
|---|---|
| 1 | the manager will approve the letter. |
| 2 | Two football players tumbling to the ground. |
| 3 | The boy is wearing safety equipment. |
| 4 | A large dog is sitting on a hill. |

POS Tagging

| Sentence | Sentence with tag |
|---|---|
| 1 | Afterwards, the manager will approve the letter.<br>RB , DT NN MD VB DT NN . |
| 2 | Two football players tumbling to the ground.<br>CD NN NNS VBG TO DT NN . |
| 3 | The boy is wearing safety equipment.<br>DT NN VBZ VBG NN NN . |
| 4 | A large dog is sitting on a hill.<br>DT JJ NN VBZ VBG IN DT NN . |

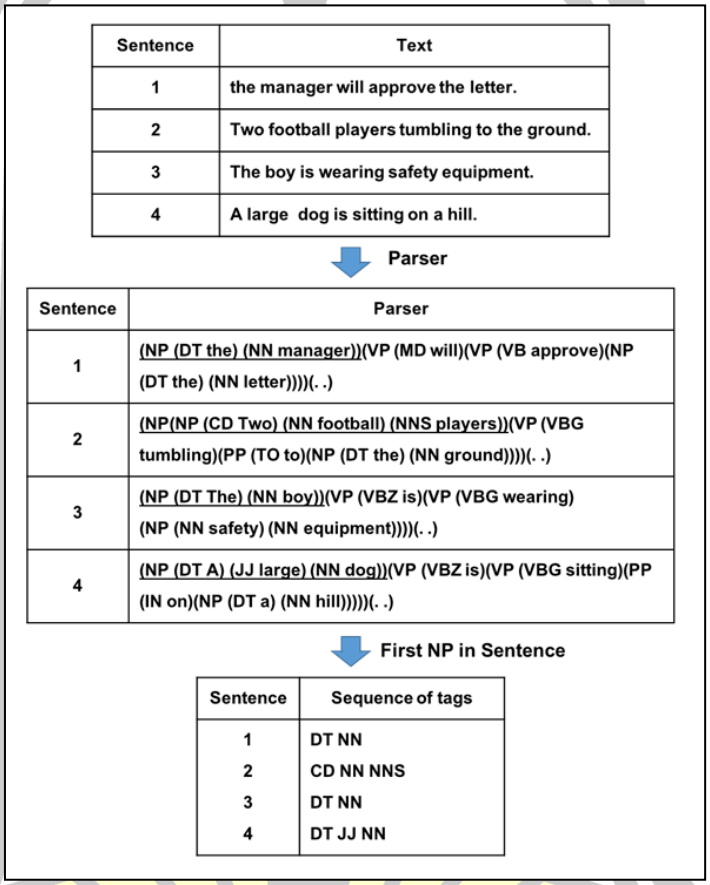
Sentence Cleaning

| Sentence | Sentence with tag |
|---|---|
| 1 | the manager will approve the letter.<br>DT NN MD VB DT NN . |
| 2 | Two football players tumbling to the ground.<br>CD NN NNS VBG TO DT NN . |
| 3 | The boy is wearing safety equipment.<br>DT NN VBZ VBG NN NN . |
| 4 | A large dog is sitting on a hill.<br>DT JJ NN VBZ VBG IN DT NN . |

ภาพที่ 3.5 ประโยคที่ผ่านการทำ *POS tagging* และการทำความสะอาดประโยค

*ขั้นตอน 2: การหาชุดลำดับของ tag ของกลุ่มคำนามในประโยคด้วย Parser*

ในขั้นตอนนี้ เป็นขั้นตอนของการใช้ *Stanford Parser* เพื่อสกัดเอา*ชุดลำดับของ tag* (Sequence of tags) ที่เป็นกลุ่มคำนาม (Noun Phrase: NP) ชุดแรกที่พบในประโยคนั้นๆ ดังแสดงในภาพที่ 3.6



| Sentence | Text |
|---|---|
| 1 | the manager will approve the letter. |
| 2 | Two football players tumbling to the ground. |
| 3 | The boy is wearing safety equipment. |
| 4 | A large dog is sitting on a hill. |

Parser

| Sentence | Parser |
|---|---|
| 1 | (NP (DT the) (NN manager))(VP (MD will)(VP (VB approve)(NP (DT the) (NN letter))))(. .) |
| 2 | (NP(NP (CD Two) (NN football) (NNS players))(VP (VBG tumbling)(PP (TO to)(NP (DT the) (NN ground))))(. .) |
| 3 | (NP (DT The) (NN boy))(VP (VBZ is)(VP (VBG wearing) (NP (NN safety) (NN equipment))))(. .) |
| 4 | (NP (DT A) (JJ large) (NN dog))(VP (VBZ is)(VP (VBG sitting)(PP (IN on)(NP (DT a) (NN hill)))))(. .) |

First NP in Sentence

| Sentence | Sequence of tags |
|---|---|
| 1 | DT NN |
| 2 | CD NN NNS |
| 3 | DT NN |
| 4 | DT JJ NN |

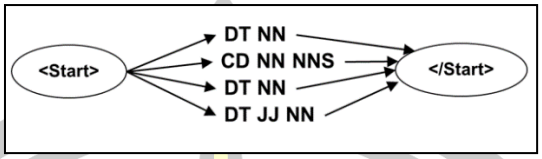
ภาพที่ 3.6 ตัวอย่างการหาชุดลำดับของ *tag* ของกลุ่มคำนามที่เริ่มต้นประโยค

*ขั้นตอน 3: การเรียนรู้กฎเพื่อการสกัด Actor ด้วยแบบจำลองมาร์คอฟ*

ภายหลังจากที่ได้ชุดลำดับของ *tag* ของกลุ่มคำนามทั้งหมดจากชุดข้อมูลที่ใช้ในการสร้างกฎในการสกัด *Actor* แต่ละชุดลำดับของ *tag* จะมีการติดสัญลักษณ์เริ่มต้น *<start>* และสิ้นสุด *<\start>* เพื่อที่ในระหว่างการเรียนรู้กฎจะได้ทราบว่าตำแหน่งใดคือตำแหน่งเริ่มต้น และตำแหน่งใดคือตำแหน่งสิ้นสุด ดังแสดงได้ต่อไปนี้

```
<start> DT NN <\start>
<start> CD NN NNS <\start>
<start> DT NN <\start>
<start> DT JJ NN <\start>
```

โดยสามารถแสดงในรูปแบบกราฟสำหรับการเรียนรู้กฎได้ดังนี้




ภาพที่ 3.7 ตัวอย่างชุดลำดับของ *tag*

เมื่อได้ชุดลำดับของ *tag* ดังตัวอย่างในภาพที่ 3.7 จากนั้นจึงสร้างแบบจำลองมาร์คอฟ โดยแบบจำลองมาร์คอฟคือตัวแบบทางคณิตศาสตร์ที่ใช้หลักความน่าจะเป็นในการวิเคราะห์พฤติกรรมของตัวแปรเพื่อคาดการณ์พฤติกรรมของตัวแปรที่จะเกิดขึ้นถัดไปซึ่งขึ้นอยู่กับเหตุการณ์ที่เกิดขึ้นก่อนหน้าโดยใช้ลูกโซ่มาร์คอฟดังสมการที่ 3.1

$$P\left(w_1^n\right) \approx \prod_{i=1}^{n} P(w_i \mid w_{i-1}) \quad \text{โดยที่ } w \text{ คือ tag} \tag{3.1}$$

จากสมการดังกล่าว สามารถทำการแยกคำนวณเพื่อหาค่าความน่าจะเป็นของการเปลี่ยนแปลงของแต่ละ tag เพื่อทำการสร้างแบบจำลองความน่าจะเป็นของชุดลำดับของ tag ได้ โดยสามารถคำนวนได้ด้วยสมการที่ 3.2

$$P\left(w_i \mid w_{i-1}\right) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})} \tag{3.2}$$

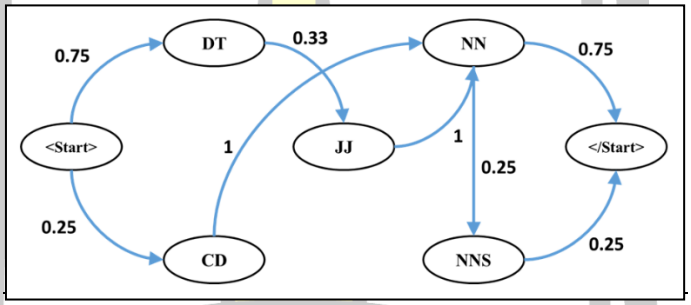
ซึ่งสามารถแสดงผลการคำนวณเพื่อหาค่าความน่าจะเป็นของการเปลี่ยนแปลง (Transition Probabilities) ของแต่ละ *tag* ได้ดังต่อไปนี้

$$P(DT| < start >) = \frac{3}{4} = 0.75$$
$$P(CD| < start >) = \frac{1}{4} = 0.25$$
$$P(NN|DT) = \frac{2}{3} = 0.66$$
$$P(NN|CD) = \frac{1}{1} = 1$$
$$P(NN|JJ) = \frac{1}{1} = 1$$
$$P(NNS|NN) = \frac{1}{4} = 0.25$$
$$P(< /start > |NN) = \frac{3}{4} = 0.75$$
$$P(< /start > |NNS) = \frac{1}{4} = 0.25$$

เมื่อคำนวณความน่าจะเป็นของการเปลี่ยนแปลงของแต่ละ tag ทั้งหมดด้วยสมการข้างต้น จะได้เมทริกซ์การเปลี่ยนสถานะ (Transition Matrix) ดังตารางที่ 3.1 และสามารถแสดงความน่าจะเป็นของการเปลี่ยนแปลงสถานะของแต่ละสถานะในรูปแบบแผนภาพดังภาพที่ 3.8

ตารางที่ 3.1 ความน่าจะเป็นในการเปลี่ยนสถานะของ *tag*

| จากสถานะ | เปลี่ยนไปเป็นสถานะ | | | | | | | รวม |
|---|---|---|---|---|---|---|---|---|
| | start | DT | NN | NNS | CD | JJ | /start | |
| start | 0 | 0.75 | 0 | 0 | 0.25 | 0 | 0 | 1 |
| DT | 0 | 0 | 0.66 | 0 | 0 | 0.33 | 0 | 1 |
| NN | 0 | 0 | 0 | 0.25 | 0 | 0 | 0.75 | 1 |
| NNS | 0 | 0 | 0 | 0 | 0 | 0 | 0.25 | 0.25 |
| CD | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| JJ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| /start | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |




ภาพที่ 3.8 แผนภาพแบบจำลองความน่าจะเป็นของชุดลำดับของ *tag*

*ขั้นตอน 4: Actor Rules Representation*

ภายหลังจากการเรียนรู้กฎเพื่อการสกัด *Actor* ในงานวิจัยนี้ได้ทำการสร้างกฎจากแบบจำลองที่ได้สร้างขึ้นด้วยเทคนิคเอ็นแกรม ได้ด้วยสมการที่ 3.3

$$P(w_1 \ldots w_n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1}) \tag{3.3}$$

โดยเอาเฉพาะชุดลำดับของ tag ที่มีผลการคำนวณจากแบบจำลองแล้วมีค่าความน่าจะเป็นมากกว่า 0 ซึ่งเมื่อคำนวณจากทุกลำดับของ tag ที่เป็นไปได้ โดยกำหนดให้กฎมีความยาวสูงสุด 3 ซึ่งสุดท้ายจะได้กฎทั้งสิ้น 584 กฎ และกฎเหล่านี้จะถูกจัดเก็บในรูปแบบ Text file และเมื่อทดสอบการสกัดกลุ่มคำนามด้วยกฎเหล่านี้กับชุดข้อมูล *SNLI* จำนวน 1,000 ประโยค พบว่ากฎทั้งหมดสามารถสกัดกลุ่มคำนามที่มีลักษณะเป็น *Actor* ได้ค่อนข้างดี อย่างไรก็ตาม เมื่อนำกฎเหล่านี้ไปทดสอบการสกัด *Actor* ในชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ พบว่าจาก

584 กฎ กฎที่จะถูกใช้จริงสำหรับชุดข้อมูลความต้องการของซอฟต์แวร์ มีจำนวนเพียง 3 กฎ ดังที่ได้สามารถแสดงในตารางที่ 3.2

ตารางที่ 3.2 กฎในการสกัด Actor

| Rule No. | Rule | Definition |
|---|---|---|
| 1 | DT +NN | Determiner +Noun |
| 2 | DT + NNP | Determiner +Proper Noun |
| 3 | DT +NN +NN | Determiner +Noun +Noun |

3.3.2 กฎในการแทนคำสรรพนาม

ในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ บ่อยครั้งที่พบว่า ประโยค 2 ประโยคมีการทำงานที่ต่อเนื่องของ *Actor* บุคคลเดียวกัน อย่างไรก็ตาม ตามหลักของไวยากรณ์ภาษาอังกฤษจะไม่มีการกล่าวถึง *"ประธาน"* ซ้ำ ในประโยคถัดไปหากมีการกล่าวถึง *"ประธาน"* นั้นๆ เช่น *"The users can access to the system. They can edit their personal information"*

ด้วยหลักไวยากรณ์ภาษาอังกฤษ จากตัวอย่างประโยคข้างต้นจะเห็นว่า *"The users"* ในประโยคแรกก็คือบุคคลคนเดียวกันกับ *"They"* ในประโยคที่สอง ประโยคในลักษณะดังกล่าว หากเป็นการพิจารณาโดยบุคคล ก็จะสามารถเข้าใจได้ได้ว่า *"They"* ก็คือ *"The users"* นั่นเอง แต่หากเป็นการประมวลผลแบบอัตโนมัติ การวิเคราะห์ดังกล่าวน่าจะต้องอาศัยกฎในการพิจารณา ดังนั้นในส่วนนี้ จึงจะแสดงการสร้างกฎสำหรับการแทนคำสรรพนามด้วย *Actor* เพื่อให้รู้ว่าคำสรรพนามที่พบในประโยคนั้นสื่อถึง *Actor* ใด โดยมีกระบวนการสร้างกฎในการแทนคำสรรพนามดังต่อไปนี้

*ขั้นตอน 1: การหารูปแบบการแทนคำสรรพนาม*

การหารูปแบบการแทนคำสรรพนาม เป็นขั้นตอนของการใช้ *Stanford Parser* เพื่อหารูปแบบการแทนคำสรรพนาม โดยจะเน้นความสนใจไปที่ประโยคที่ขึ้นต้นด้วยคำสรรพนามหรือ tag *PRP* และจะใช้ประโยคก่อนหน้าที่มีประธานหรือกลุ่มคำนาม (Noun Phrase: NP) ชุดแรกที่พบในประโยค ที่กล่าวถึงคำสรรพนามในประโยคถัดไป โดยเมื่อได้รูปแบบการทำสรรพนามทั้งหมดจากชุดข้อมูลแล้ว จะนำรูปแบบการทำสรรพนามเหล่านั้นไปใช้ในขั้นตอนต่อไป คือขั้นตอนการเรียนรู้กฎการแทนคำสรรพนามด้วยแบบจำลองมาร์คอฟซ่อนเร้น

| Sentence | Text |
|---|---|
| 1 | The hikers climb up the tower, they can see the view better. |
| 2 | The women in fancy uniforms, they are maids. |
| 3 | Smoking is not good for health, it harms your lungs. |
| 4 | A woman running in a race, she is going to win. |

Parser

| Sentence | Parser |
|---|---|
| 1 | (S(S(NP (DT The) (NNS hikers))(VP (VB climb)(PRT (RP up))(NP (DT the) (NN tower))))(, ,)(NP (PRP they))(VP (MD can)(VP (VB see)(NP (DT the) (NN view))(ADVP (RBR better))))(. .)) |
| 2 | (S(S(NP(NP (DT The) (NNS women))(PP (IN in)(NP (JJ fancy) (NNS uniforms)))))(, ,)(NP (PRP they))(VP (VBP are)(ADJP (JJ maids)))(. .)) |
| 3 | (S(S(NP (NN Smoking))(VP (VBZ is) (RB not)(ADJP (JJ good)(PP (IN for)(NP (NN health)))))(, ,)(NP (PRP it))(VP (VBZ harms)(NP (PRP$ your) (NNS lungs)))(. .)) |
| 4 | (S(S(NP(NP (DT A) (NN woman))(VP (VBG running)(PP (IN in)(NP (DT a) (NN race)))))(, ,)(NP (PRP she))(VP (VBZ is)(VP (VBG going)(S(VP (TO to)(VP (VB win))))))) |

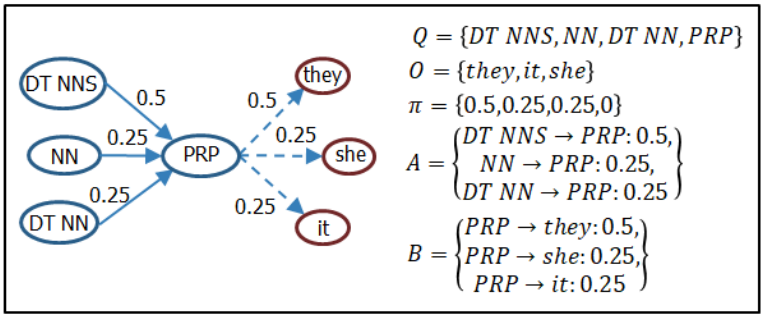
First NP in first sentence
and First PRP in second sentence

| Sentence | Noun Phrase | Refer to |
|---|---|---|
| 1 | DT NNS | PRP (they) |
| 2 | DT NNS | PRP (they) |
| 3 | NN | PRP (it) |
| 4 | DT NN | PRP (she) |

ภาพที่ 3.9 ตัวอย่างการหารูปแบบการแทนคำสรรพนาม

*ขั้นตอน 2: การเรียนรู้กฎการแทนคำสรรพนามด้วยแบบจำลองมาร์คอฟซ่อนเร้น*

การเรียนรู้กฎการแทนคำสรรพนามด้วยแบบจำลองมาร์คอฟซ่อนเร้น จะเป็นการนำรูปแบบการทำคำสรรพนามที่ได้จากขั้นตอนก่อนหน้า มาทำการคำนวนเพื่อหาค่าความน่าจะเป็นต่างๆ ประกอบไปด้วย initial probability ($\boldsymbol{\pi}$) transition probability ($A$) และ emission probability ($B$) โดยที่ $Q$ คือเซตของสถานะ และ $O$ คือผลการสังเกต ซึ่งสามารถแสดงตัวอย่างแบบจำลองมาร์คอฟซ่อนเร้นที่สร้างจากรูปแบบการแทนคำสรรพนามได้ดังภาพที่ 3.10




ภาพที่ 3.10 ตัวอย่างแบบจำลองมาร์คอฟซ่อนเร้นที่สร้างจากรูปแบบการแทนคำสรรพนาม

*ขั้นตอน 3: การสร้างกฎการแทนคำสรรพนาม*

ภายหลังจากการเรียนรู้กฎการแทนคำสรรพนามด้วยแบบจำลองมาร์คอฟซ่อนเร้นในงานวิจัยนี้ได้ทำการสร้างกฎจากแบบจำลองที่ได้สร้างขึ้นโดยคำนวนจาก $P(O|\lambda)$ ซึ่งจะทำการคำนวนจากทุกความเป็นไปได้ และรูปแบบที่คำนวณจากแบบจำลองแล้วมีค่าความน่าจะเป็นมากกว่า 0 จะถูกจัดเก็บเป็นกฎ ซึ่งเมื่อคำนวณจากทุกความเป็นไปได้แล้วจะได้กฎทั้งสิ้นจำนวน 231 กฎ โดยสามารถแสดงตัวอย่างกฎได้ดังตารางที่ 3.3

ตารางที่ 3.3 ตัวอย่างกฎในการแทนคำสรรพนาม

| Rule No. | Rule | Replacing |
|---|---|---|
| 1 | NN | It |
| 2 | NNS | They |
| 3 | DT + NN | She |
| 4 | CD + NNS | They |
| 5 | DT + JJ + NN | He |

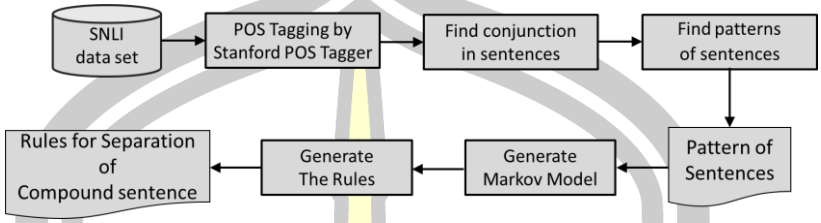
3.3.3 กฎในการแยกประโยคความรวม

ในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่มีการรวบรวมมานั้น บ่อยครั้งที่พบว่าประโยคมีการแสดงการกระทำหรือ *Action* สองการกระทำในหนึ่งประโยค เช่น

*"Before the manager approves a letter, the manager needs to verify it."*

จากประโยคดังกล่าว จะเห็นว่า *"the manager"* ซึ่งเป็น *Actor* มี *Action* สองอย่างคือ *"approves a letter"* และ *"needs to verify it"* ซึ่งหากเป็นการพิจารณาโดยบุคคล ก็จะสามารถแยกการทำงานของ *"the manager"* ได้ง่าย แต่หากเป็นการประมวลผลแบบอัตโนมัติ การวิเคราะห์

ดังกล่าวจำเป็นจะต้องอาศัยกฎในการพิจารณา ดังนั้นในส่วนนี้จึงแสดงการสร้างกฎในการแยกประโยคความรวม โดยมีกรอบการประมวลผลดังภาพที่ 3.11

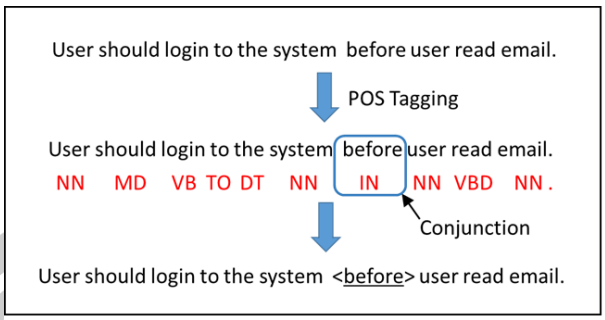



ภาพที่ 3.11 กรอบการประมวลผลการสร้างกฎในแยกประโยคความรวม

กฎในการแยกประโยคความรวมจะเน้นความสนใจที่ประโยคความรวม (Compound Sentence) ที่เชื่อมระหว่างประโยคด้วยคำสันธาน (Conjunction) ซึ่งคำสันธานเป็นคำที่ใช้เชื่อมอนุประโยคหรือประโยคย่อยเข้าด้วยกัน (ประโยคย่อยคือประโยคที่แสดงความหมายโดยสมบูรณ์ซึ่งจะประกอบด้วยประธานและกริยา) โดยคำสันธานอาจอยู่หน้าประโยคและใช้เครื่องหมายจุลภาค (Comma: ,) คั่นประโยคย่อย หรือคำสันธานอาจคั่นอยู่กลางระหว่างประโยคย่อยก็ได้ [49] ซึ่งในงานวิจัยนี้จะใช้คำสันธานที่พบบ่อยในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ เช่น *if*, *then*, *after*, *when* และ *before* เป็นต้น โดยการสร้างกฎในการแยกประโยคความรวมสามารถแสดงในแต่ละขั้นตอนได้ดังต่อไปนี้

*ขั้นตอน 1*: *การหาตำแหน่งของคำสันธานในประโยคความรวม*

ในขั้นตอนนี้เป็นการหาตำแหน่งของคำสันธานหรือ tag *“IN”* ในประโยค เพื่อใช้คำสันธานที่พบในประโยคเป็นตำแหน่งในการตัดแบ่งประโยค และใช้ในการหาตำแหน่งของคำสันธานในประโยคในขั้นตอนถัดไป โดยเริ่มจากการทำ POS Tagging เพื่อกำกับชนิดของคำในประโยค และทำการหาและระบุตำแหน่งของคำสันธานหรือ *“IN”* ที่อยู่ภายในประโยค โดยแสดงตัวอย่างดังภาพที่ 3.12

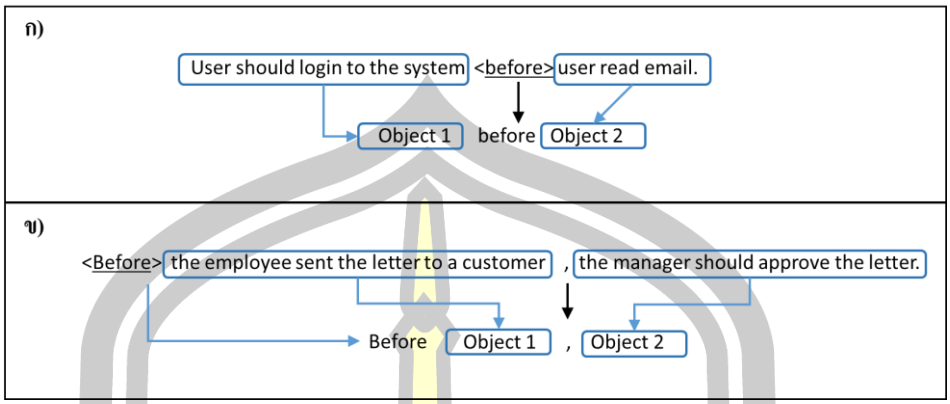



ภาพที่ 3.12 ตัวอย่างการหาตำแหน่งของคำสันธานในประโยคความรวม

จากตัวอย่างดังภาพที่ 3.12 จะเริ่มจากการนำประโยคมากำกับชนิดของคำในประโยค ด้วยการทำ *POS Tagging* และเริ่มอ่าน *tag* ในประโยคจากตำแหน่งซ้ายสุดของประโยค เพื่อค้นหา *tag* ที่เป็น *"IN"* โดยเมื่อพบคำสันธานจะระบุตำแหน่งของคำสันธานที่อยู่ในตำแหน่ง *"IN"* ในกรณีที่ไม่พบคำสันธานในประโยคให้ข้ามไปพิจารณายังประโยคถัดไป โดยตำแหน่งของคำสันธานที่พบในประโยค จะใช้ในการแบ่งประโยคออกเป็น *Object* ในขั้นตอนถัดไป คือการหาชุดลำดับของ Object ในประโยคความรวม ยกตัวอย่างประโยคในภาพที่ 3.12 เมื่อพบคำสันธานคือ *before* ดังนั้นจะสามารถแยกประโยคออกเป็น 2 *Object* ได้แก่ *"The three people are napping"* คือ *Object* 1 และ *"going to the swimming pool"* คือ *Object* 2 เป็นต้น

*ขั้นตอน 2: การหาชุดลำดับของ Object ในประโยคความรวม*

ในขั้นตอนนี้เป็นการหาชุดลำดับของ *Object* ในประโยคความรวม โดยเริ่มด้วยการอ่านคำในประโยคจากซ้ายไปขวาเพื่อหาตำแหน่งของคำสันธานที่ถูกระบุไว้จากขั้นตอนก่อนหน้า และใช้เป็นตำแหน่งเพื่อช่วยในการแยกประโยคความรวมออกเป็นประโยคย่อยหรือ *Object* ซึ่งจำนวนของ *Object* จะเป็น *Object 1, Object 2,...,Object N* ตามจำนวนประโยคย่อยที่พบในประโยคความรวม ซึ่งชุดลำดับของ *Object* ที่ได้จากขั้นตอนนี้ จะนำไปใช้สำหรับการเรียนรู้กฎในการแยกประโยคความรวมด้วยแบบจำลองมาร์คอฟในขั้นตอนถัดไป



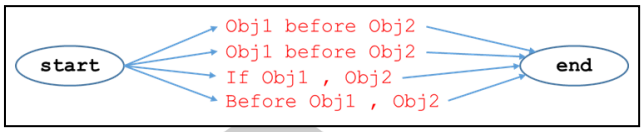

ภาพที่ 3.13 ตัวอย่างการสกัดเพื่อหาชุดลำดับของ *Object* ในประโยคความรวม

จากภาพที่ 3.13 (ก) เป็นตัวอย่างประโยคความรวมที่เชื่อมประโยคด้วยคำสันธาน จะพบว่าตำแหน่งของคำสันธานหรือคำว่า *before* อยู่กึ่งกลางประโยค ดังนั้นจึงสามารถแบ่งประโยคย่อยได้เป็น 2 ส่วน ได้แก่ส่วนที่อยู่หน้าคำสันธาน คือ *"User should login to the system"* และส่วนที่อยู่หลังคำสันธานคือ *"user read email"* โดยจะถูกมองเป็น *Object1* และ *Object2* ตามลำดับ สุดท้ายจะได้ชุดลำดับของ *Object* จากประโยคนี้คือ *"Object1 + before + Object2"* เป็นต้น

และจากภาพที่ 3.13 (ข) เป็นตัวอย่างประโยคความรวมที่มีคำสันธานอยู่หน้าประโยคและใช้เครื่องหมายจุลภาคในการคั่นประโยคย่อย ในกรณีนี้จะใช้ตำแหน่งของคำสันธานร่วมกับเครื่องหมายจุลภาคในการแบ่งประโยคย่อย โดยสามารถแบ่งประโยคย่อยได้เป็น 2 ส่วน ได้แก่ส่วนที่อยู่หลังคำสันธาน จนถึงเครื่องหมายจุลภาคคือ *"the employee sent the letter to a customer"* และส่วนที่อยู่หลังเครื่องหมายจุลภาคคือ *"the manager should approve the letter"* โดยกำหนดให้ประโยคย่อยที่แบ่งได้เป็น *Object1* และ *Object2* ตามลำดับ สุดท้ายจะได้ชุดลำดับของ *Object* จากประโยคนี้คือ *"Before + Object1 + , + Object2"* เป็นต้น

*ขั้นตอน 3: การเรียนรู้กฎในการแยกประโยคความรวมด้วยแบบจำลองมาร์คอฟ*

ภายหลังจากที่ได้ชุดลำดับของ *Object* ทั้งหมดจากชุดข้อมูลที่ใช้ในการสร้างกฎในการแยกประโยคความรวมที่ผ่านขั้นตอนที่ 1 และ 2 แล้ว จะทำการติดสัญลักษณ์เริ่มต้น *"<start>"* และสิ้นสุด *"<\start>"* แต่ละชุดลำดับของ *Object* เพื่อในระหว่างการเรียนรู้กฎจะได้ทราบว่าตำแหน่งใดคือตำแหน่งเริ่มต้น และตำแหน่งใดคือตำแหน่งสิ้นสุด สุดท้ายจะได้ชุดลำดับของ *Object* (*Obj*) ที่เป็นไปได้โดยแสดงในรูปแบบกราฟสำหรับการเรียนรู้กฎดังตัวอย่างในภาพที่ 3.14




ภาพที่ 3.14 ตัวอย่างลำดับของ *Object*

เมื่อได้ชุดลำดับของ *Object* ดังตัวอย่างในภาพที่ 3.14 จากนั้นจึงสร้างแบบจำลองมาร์คอฟ โดยแบบจำลองมาร์คอฟคือตัวแบบทางคณิตศาสตร์ที่ใช้หลักความน่าจะเป็นในการวิเคราะห์พฤติกรรมของตัวแปรเพื่อคาดการณ์พฤติกรรมของตัวแปรที่จะเกิดขึ้นถัดไปซึ่งขึ้นอยู่กับเหตุการณ์ที่เกิดขึ้นก่อนหน้าโดยใช้ลูกโซ่มาร์คอฟ ดังสมการที่ 3.4

$$P\left(w_1^n\right) \approx \prod_{i=1}^{n} P(w_i \mid w_{i-1}) \text{ โดยที่ } w \text{ คือ } Object \qquad (3.4)$$

จากสมการดังกล่าว สามารถทำการแยกคำนวณเพื่อหาค่าความน่าจะเป็นของการเปลี่ยนแปลงของแต่ละ *Object* เพื่อทำการสร้างแบบจำลองความน่าจะเป็นของชุดลำดับของ *Object* ได้ด้วยสมการที่ 3.5

$$P\left(w_i \mid w_{i-1}\right) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})} \qquad (3.5)$$

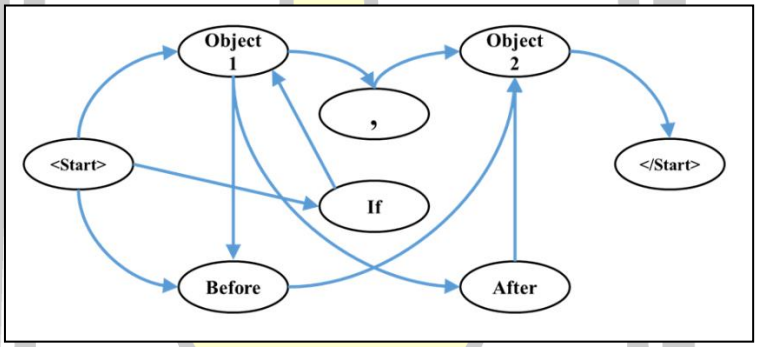
ซึ่งแสดงผลการคำนวณเพื่อหาค่าความน่าจะเป็นของการเปลี่ยนแปลงของแต่ละ *Object* ได้ดังต่อไปนี้

$$P(Object1| < start >) = \frac{2}{4} = 0.50$$
$$P(if| < start >) = \frac{1}{4} = 0.25$$
$$P(before| < start >) = \frac{1}{4} = 0.25$$
$$P(before|Object1) = \frac{2}{4} = 0.5$$
$$P(,|Object1) = \frac{2}{4} = 0.5$$
$$P(Object1|if) = \frac{1}{1} = 1$$
$$P(Object1|before) = \frac{1}{3} = 0.33$$
$$P(Object2|before) = \frac{2}{3} = 0.66$$
$$P(Object2|,) = \frac{2}{2} = 1$$
$$P(</start > |Obj2) = \frac{4}{4} = 1$$

เมื่อคำนวณสถานะทั้งหมดด้วยสมการข้างต้น จะได้เมทริกซ์การเปลี่ยนสถานะของแต่ละ *Object* (*obj*) ดังตารางที่ 3.4 และสามารถแสดงความน่าจะเป็นของการเปลี่ยนแปลงสถานะของแต่ละสถานะดังภาพที่ 3.15

ตารางที่ 3.4 ความน่าจะเป็นในการเปลี่ยนสถานะของ *Object*

| จากสถานะ | เปลี่ยนไปเป็นสถานะ | | | | | | | รวม |
|---|---|---|---|---|---|---|---|---|
| | start | Object1 | Object2 | before | if | , | /start | |
| start | 0 | 0.5 | 0 | 0.25 | 0.25 | 0 | 0 | 1 |
| Object1 | 0 | 0 | 0 | 0.5 | 0 | 0.5 | 0 | 1 |
| Object2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| before | 0 | 0.33 | 0.66 | 0 | 0 | 0 | 0 | 1 |
| if | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| , | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| /start | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |




ภาพที่ 3.15 ตัวอย่างแบบจำลองความน่าจะเป็นของลำดับของ *Object*

*ขั้นตอน 4*: *Rules Representation*

ภายหลังจากการเรียนรู้กฎในการแยกประโยคความรวม ในงานวิจัยนี้ได้ทำการสร้างกฎจากแบบจำลองที่ได้สร้างขึ้นด้วยเทคนิคเอ็นแกรม ซึ่งแต่ละกฎจะมีความยาว *N* คำ ได้ด้วยสมการที่ 3.6

$$P(w_1 \ldots w_n) \approx \prod_{k=1}^{n} P(w_k \mid w_{k-1}) \tag{3.6}$$

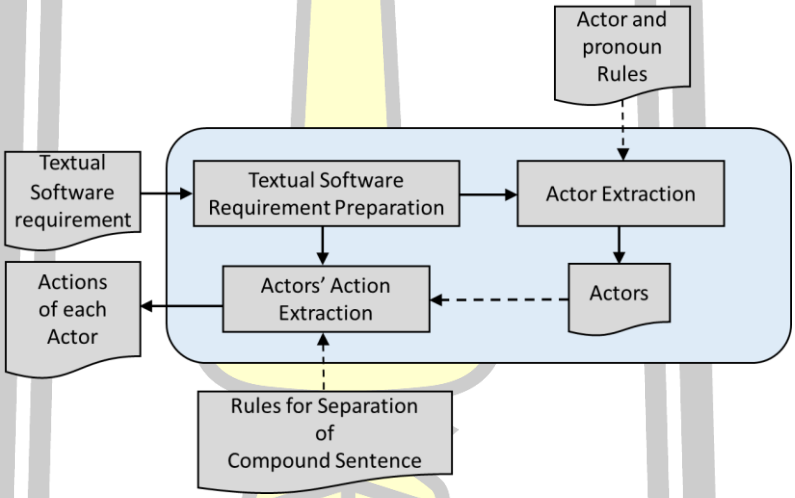
โดยเอาเฉพาะชุดลำดับของ *Object* ที่มีผลการคำนวณจากแบบจำลองแล้วมีค่าความน่าจะเป็นมากกว่า 0 ซึ่งเมื่อคำนวณจากทุกลำดับของ *Object* ที่เป็นไปได้แล้ว สุดท้ายจะได้กฎในการแยกประโยคความรวมจำนวน 39 กฎ โดยสามารถแสดงตัวอย่างได้ดังตารางที่ 3.4

ตารางที่ 3.5 ตัวอย่างกฎในการแยกประโยคความรวม

| Rule No. | Pattern | Description |
|---|---|---|
| 1 | Obj1 +before +Obj2 | Object 1 , Object 2 |
| 2 | Obj1 +after +Obj2 | Object 2 , Object 1 |
| 3 | Before +Obj1 +, +Obj2 | Object 2 , Object 1 |
| 4 | Obj1 +,+and +then +Obj2 | Object 1 , Object 2 |
| 5 | If +Obj1 +, +Obj2 | Object 1 , Object 2 |

### 3.4 การสกัด Actor และ Action (Extracting of Actors and Actions)

ขั้นตอนนี้คือการนำกฎเพื่อวิเคราะห์โครงสร้างภาษารูปแบบต่างๆ ที่ได้ทำการสร้างขึ้น มาทำการสกัดเพื่อหาซีนาริโอคือ *Actor* และ *Action* ของแต่ละ *Actor* จากข้อมูลภายในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยแสดงกรอบการทำงานดังภาพที่ 3.16




ภาพที่ 3.16 กรอบการประมวลผลเพื่อการสกัด *Actor* และ *Action* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

จากภาพที่ 3.16 ซึ่งแสดงกรอบการประมวลผลในการสกัด *Actor* และ *Action* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยจะมีขั้นตอนย่อย 3 ขั้นตอนได้แก่

1. การเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์
2. การสกัด *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ด้วยกฎสกัด *Actor*
3. การสกัด *Action* ของแต่ละ *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

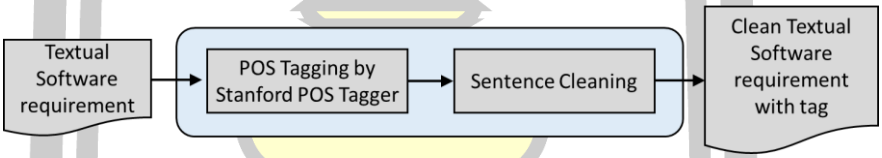
Before this process can be initiated, the Editors has already accessed the main page of the Article Manager.
The Editors selects a function for working. So, they can choose add or update function. The system presents a function required. Therefore, the Editors chooses to add or to update. If the Editor is updating an author, the System presents a list of authors to choose from and presents a grid filling in with the information; else the system presents a blank grid. The Editors fills in the information and submits the form. The system verifies the information and returns the Editors to the Article Manager main page.

ภาพที่ 3.17 ตัวอย่างข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

ซึ่งสามารถแสดงตัวอย่างการประมวลผลเพื่อการสกัด *Actor* และ *Action* ของแต่ละ *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ดังกล่าวได้ดังนี้

3.4.1 การเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ (Textual Software Requirement Preparation)

ขั้นตอนนี้เป็นขั้นตอนเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยมีการทำงานหลักๆ คือการกรองคำหรือกลุ่มคำที่ไม่จำเป็นของแต่ละประโยคออกไป ซึ่งในงานวิจัยนี้เรียกว่า “*การทำความสะอาดประโยค (Sentence Cleaning)*” สำหรับขั้นตอนในการทำความสะอาดประโยค สามารถแสดงได้ดังภาพที่ 3.19



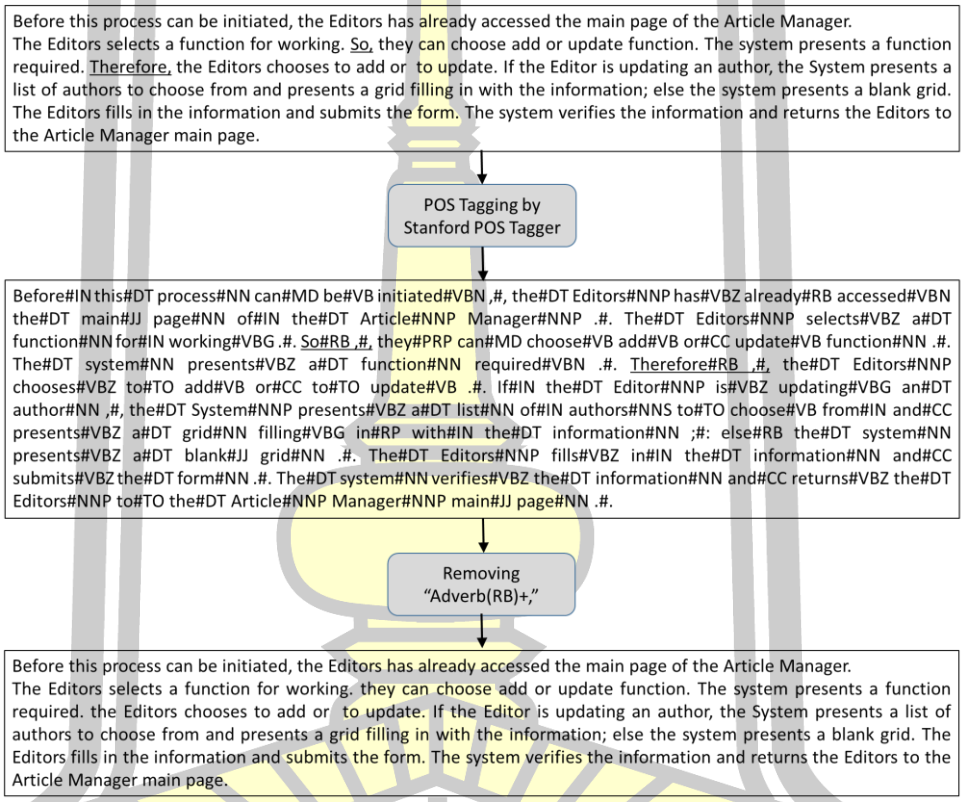

ภาพที่ 3.18 ขั้นตอนการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

จากภาพที่ 3.18 จะเห็นว่าในการทำความสะอาดประโยคเพื่อการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์นั้น จะประกอบด้วยการทำงานหลักๆ ใน 2 ขั้นตอนคือระบุชนิดของคำด้วย Stanford POS Tagger และการทำความสะอาดประโยคด้วยการกำจัดคำ ที่เป็นคำในกลุ่มของคำวิเศษณ์ (Adverb) หรือ tag “*RB*” ที่อยู่ตำแหน่งซ้ายมือสุดของประโยค และมีเครื่องหมาย “,” ตามหลังคำเหล่านั้นออกไป ซึ่งคำเหล่านั้น ได้แก่ Additionally, Furthurmore, และ Therefore เป็นต้น

สมมติมีประโยคว่า “*Therefore, the Editors chooses to add or to update.*” สำหรับประโยคนี้คำว่า “*Therefore,*” จะถูกตัดออกไป เพราะจากการระบุชนิดของคำด้วย Stanford POS Tagger พบว่า “*Therefore*” เป็นคำในกลุ่มที่กล่าวมาข้างต้น และตรงตามเงื่อนไขที่กำหนด ดังนั้นภายหลังจากการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ก็จะ

ได้ประโยค “*the Editors chooses to add or  to update*.” เพื่อนำไปใช้ในขั้นตอนถัดไป คือ ขั้นตอนการสกัดหา *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ด้วยกฎในการสกัด *Actor*

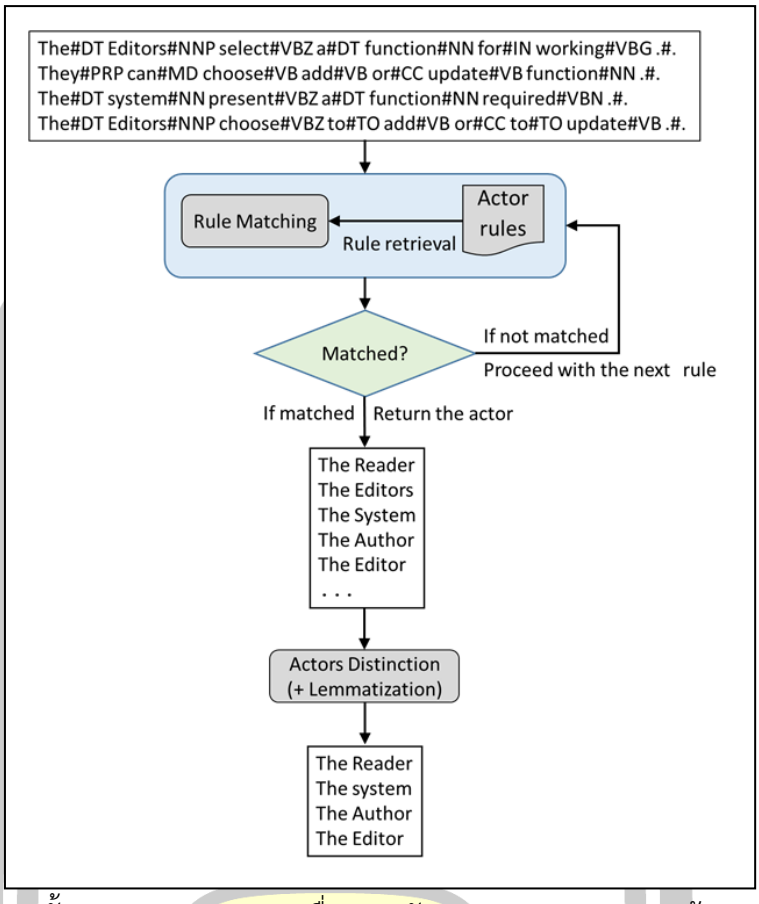
และจากตัวอย่างเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ในภาพที่ 3.17 จะสามารถแสดงการเตรียมข้อความที่แสดงความต้องการซอฟต์แวร์และผลลัพธ์ของขั้นตอนนี้ได้ดังตัวอย่างในภาพที่ 3.19




ภาพที่ 3.19 ตัวอย่างการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

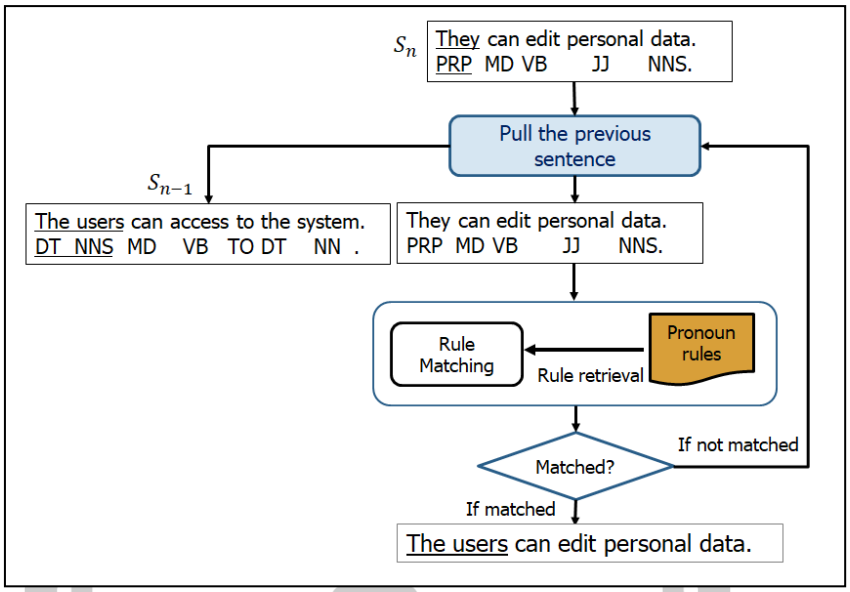
3.4.2 การสกัด Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ด้วยกฎในการสกัด Actor (Actor Extraction from Textual Software Requirements by Rules)

จากผลลัพธ์ที่ได้จากขั้นตอนการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ในขั้นตอนนี้เป็นการนำกฎในการสกัด *Actor* มาใช้ เพื่อสกัดหา *Actor* ทั้งหมดในชุดเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยมีขั้นตอนการทำงานดังแสดงในภาพที่ 3.20




ภาพที่ 3.20 ขั้นตอนการประมวลผลเพื่อการสกัดหา Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

จากภาพที่ 3.20 จะเป็นการนำเอากฎในการสกัด *Actor* ที่ได้อธิบายไว้แล้วในหัวข้อที่ 3.3.1 มาใช้เพื่อสกัดเอา *Actor* นั้นออกมาจากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์แต่ละประโยค โดยมีขั้นตอนการทำงานคือ นำผลลัพธ์ที่ได้จากการเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์แต่ละประโยคมาทำการเทียบกับกฎที่ใช้ในการสกัด *Actor* หากตรงกันก็สกัด *Actor* ในตำแหน่งนั้นออกมา โดยในกรณีที่ประโยคขึ้นต้นด้วยคำสรรพนาม ให้ทำการใช้กฎในการแทนคำสรรพนาม ในการแทนที่ตำแหน่งของคำสรรพนามที่พบด้วย *Actor* จากนั้นจึงทำการข้ามไปสกัด *Actor* ในประโยคถัดไป โดยตัวอย่างการใช้กฎในการแทนคำสรรพนามสามารถแสดงได้ดังภาพที่ 3.21




ภาพที่ 3.21 ตัวอย่างการใช้กฎในการแทนคำสรรพนาม

จากนั้นทำการหาความแตกต่าง (Distinction) เพื่อไม่ให้เกิด *Actor* ที่ซ้ำซ้อน โดยในการทำ *Distinction* ได้มีการใช้เทคนิคการลดรูปคำร่วมด้วย เพื่อเปลี่ยนรูปของคำให้ไปอยู่ในรูปดั้งเดิม ด้วยวิธีการนี้ ทำให้การลดความซ้ำซ้อนของคำที่แสดง *Actor* ไม่มีความผิดพลาด ยกตัวอย่างเช่น คำว่า “*Author*” และ “*Authors*” สำหรับในความต้องการซอฟต์แวร์นี้คือ กลุ่มคนกลุ่มเดียวกัน แต่ในภาษาอังกฤษเมื่อกล่าวถึงในสถานการณ์ที่แตกต่างกัน ก็จะเขียนแสดงในรูปแบบที่แตกต่างกัน ดังนั้นหากไม่มีการทำลดรูปคำก็อาจจะทำให้ในการสกัด *Actor* ได้ *Actor* ที่ซ้ำซ้อน

3.4.3 การสกัด Action ของแต่ละ Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ (Actors’ Action Extraction from Textual Software Requirements)

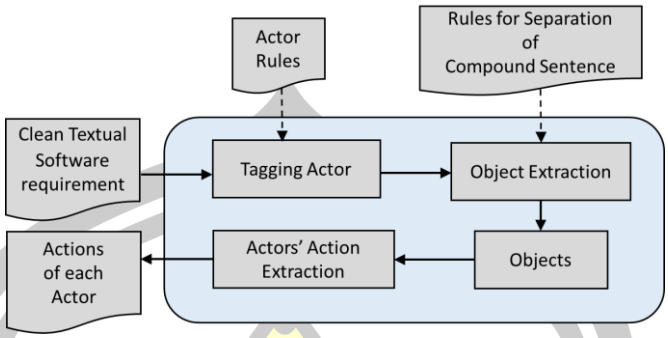
สำหรับในขั้นตอนการสกัด *Action* ของแต่ละ *Actor* นั้น จะแบ่งการทำงานออกเป็น 3 ขั้นตอนย่อย คือ

(1) การระบุตำแหน่งของ *Actor* ในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

(2) การสกัดรูปแบบโครงสร้างของประโยคด้วยกฎในการแยกประโยคความรวม

(3) การสกัด *Action* และ *Actor* ออกจาก *Object*

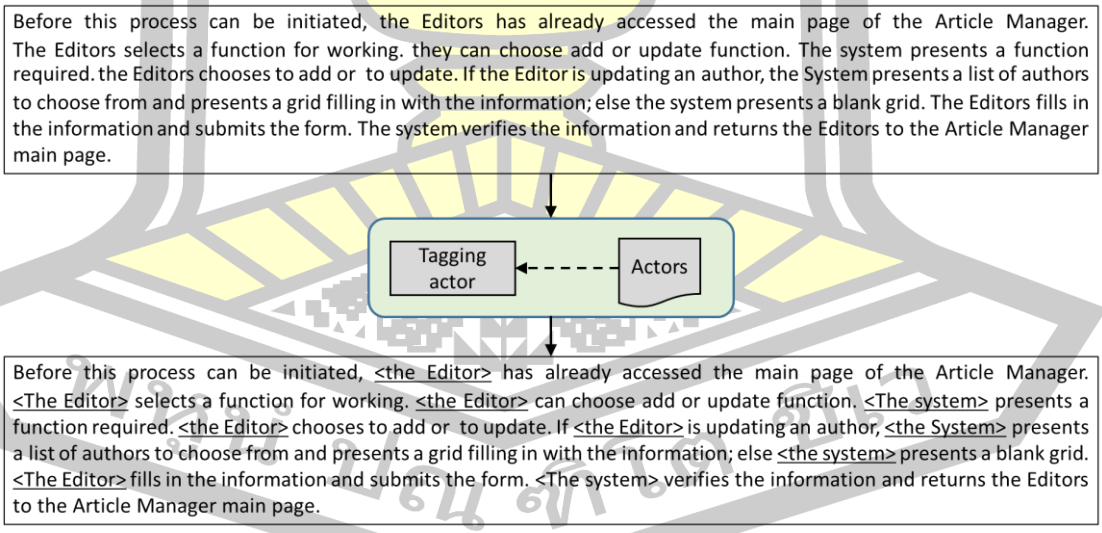
และสามารถแสดงกรอบการประมวลผลการสกัด *Action* ของแต่ละ *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ได้ดังภาพที่ 3.22




ภาพที่ 3.22 กรอบการประมวลผลการสกัด Action ของแต่ละ Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

*(1) การระบุตำแหน่งของ Actor ภายในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์*

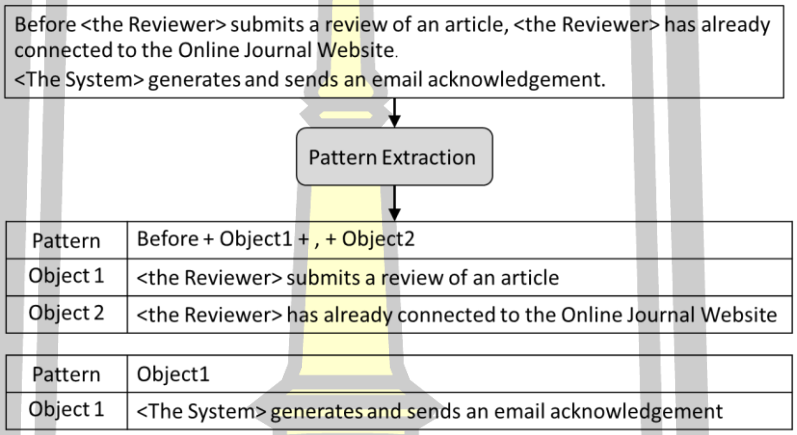
เมื่อได้ Actor ทั้งหมดจากขั้นตอนการสกัด Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ด้วยกฎในการสกัด Actor ที่ได้อธิบายขั้นตอนไว้ในหัวข้อที่ 3.4.2 จากนั้นจะนำ Actor เหล่านั้นมาทำการค้นหาและระบุตำแหน่ง Actor ที่อยู่ภายในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ เพื่อนำไปใช้ในขั้นตอนถัดไป คือขั้นตอนการสกัด Action ของแต่ละ Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยสามารถแสดงการระบุตำแหน่งของ Actor และผลลัพธ์ได้ดังตัวอย่างในภาพที่ 3.23




ภาพที่ 3.23 ตัวอย่างการระบุตำแหน่งของ *Actor* ในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

*(2) การสกัดรูปแบบโครงสร้างของประโยคด้วยกฎในการแยกประโยคความรวม*

ในขั้นตอนนี้ เป็นการหารูปแบบโครงสร้างของประโยคด้วยกฎในการแยกประโยคความรวม โดยเน้นไปที่ประโยคความรวมหรือประโยคข้อความที่สามารถสกัดได้มากกว่าหนึ่งประโยคย่อย หรือที่งานวิจัยนี้เรียกว่า *Object* ซึ่ง *Object* ที่สกัดได้จากขั้นตอนนี้จะนำไปใช้ในขั้นตอนถัดไป คือ การสกัด *Action* และ *Actor* ออกจาก *Object* โดยแสดงตัวอย่างผลการสกัดรูปแบบโครงสร้างของประโยคด้วยกฎในการแยกประโยคความรวมได้ดังภาพที่ 3.24




ภาพที่ 3.24 ตัวอย่างการสกัดรูปแบบโครงสร้างของประโยคข้อความที่แสดงความต้องการซอฟต์แวร์

*(3) การสกัด Action และ Actor ออกจาก Object*

เมื่อได้ผลลัพธ์หรือ *Object* ทั้งหมดจากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์จากขั้นตอนการสกัดประโยคย่อยจากประโยคความรวมแล้ว ในขั้นตอนนี้ จะเป็นขั้นตอนในการสกัด *Action* และ *Actor* ออกจาก *Object* เพื่อใช้สำหรับการสร้างแบบจำลองกระบวนการของซอฟต์แวร์ในขั้นตอนถัดไป

ยกตัวอย่างประโยคที่มีข้อความ “*<the Reviewer> submits a review of an article*” จะมีโครงสร้างประโยคสอดคล้องกับรูปแบบ “*Actor-Action-Object*” จากโครงสร้างประโยคของเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่ได้อธิบายไว้ในหัวข้อที่ 2.2.3 ดังนั้น เมื่อพบ *Actor* ที่ถูกระบุไว้ใน *Object* ซึ่งก็คือ “*the Reviewer*” ดังนั้นข้อความที่อยู่หลัง *Actor* ซึ่งก็คือ “*submits a review of an article*” จึงถูกกำหนดเป็น *Action* โดยแสดงตัวอย่างผลการสกัด *Action* และ *Actor* ออกจาก *Object* ได้ดังภาพที่ 3.25

| Object |
|---|
| <the Reviewer> submits a review of an article |
| <the Reviewer> has already connected to the Online Journal Website |
| <The System> generates and sends an email acknowledgement |

Action's Actor Extraction

| Actor | Action |
|---|---|
| the Reviewer | already connected to the Online Journal Website |
| the Reviewer | submits a review of an article |
| The System | generates and sends an email acknowledgement |

ภาพที่ 3.25 ตัวอย่างผลการสกัด *Action* และ *Actor* ออกจาก *Object*

### 3.5 การสร้างแบบจำลองกระบวนการของซอฟต์แวร์

การสร้างแบบจำลองกระบวนการของซอฟต์แวร์ (Software Process Modelling) คือการนำผลการสกัดซีนาริโอ ได้แก่การกระทำหรือ *Action* ของแต่ละผู้กระทำหรือ *Actor* มาทำการแสดงให้อยู่ในรูปแบบจำลองโดยทำการรวบรวมและจัดกลุ่ม *Action* ของแต่ละ *Actor* โดยงานวิจัยนี้ได้นำเสนอแบบจำลองในรูปแบบข้อมูลกึ่งโครงสร้าง ซึ่งสามารถแสดงผลการสร้างแบบจำลองกระบวนการของซอฟต์แวร์จากซีนาริโอที่สกัดได้ดังภาพที่ 3.26

| Actor | Action |
|---|---|
| the Reviewer | already connected to the Online Journal Website |
| the Reviewer | submits a review of an article |
| The System | generates and sends an email acknowledgement |

Software Process Modeling

| **Software Process** | |
|---|---|
| the Reviewer | The System |
| already connected to the Online Journal Website | generates and sends an email acknowledgement |
| submits a review of an article | |

ภาพที่ 3.26 ตัวอย่างการแสดงแบบจำลองกระบวนการของซอฟต์แวร์

# บทที่ 4

# ผลการวิจัย

ในบทนี้จะกล่าวถึงชุดข้อมูลที่ใช้ในการวิจัย และการประเมินประสิทธิภาพของกระบวนการสกัดซีนาริโอจากเอกสารความต้องการของซอฟต์แวร์ โดยสามารถแสดงรายละเอียดของชุดข้อมูลที่ใช้และการประเมินประสิทธิภาพและการวิจารณ์ผลในด้านต่างๆ ดังต่อไปนี้

## 4.1 ชุดข้อมูลที่ใช้ในงานวิจัย (Dataset)

### 4.1.1 ชุดข้อมูลที่ใช้ในการประเมินกฎในการสกัด Actor

ชุดข้อมูลที่ใช้ในการประเมินกฎในการสกัด *Actor* จะใช้ข้อมูลชุดเดียวกันกับชุดข้อมูลที่ใช้ในการสร้างกฎในการสกัด *Actor* คือคลังข้อมูล *SNLI* โดยจะสุ่มเลือกประโยคภาษาอังกฤษจำนวนประโยคไม่น้อยกว่า 1,000 ประโยคที่ไม่อยู่ในชุดข้อมูลที่ใช้ในการสร้างกฎในการสกัด *Actor* ซึ่งแสดงตัวอย่างประโยคที่ใช้ประเมินกฎในการสกัด *Actor* ดังภาพที่ 4.1

```
Three bikers stop in town.
Two people walk home after a tasty steak dinner.
The boy is wearing safety equipment.
The white crane is searching for food in a pond.
Two football players tumbling to the ground.
A large yellow dog is sitting on a hill.
```

ภาพที่ 4.1 ตัวอย่างประโยคที่ใช้ประเมินกฎในการสกัด Actor

### 4.1.2 ชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

เอกสารที่จะใช้ในงานวิจัยเป็นเอกสารความต้องการของซอฟต์แวร์ที่รวบรวมความต้องการของซอฟต์แวร์ที่กำลังจะพัฒนา เป็นภาษาอังกฤษ ความยาวต่อเนื่องไม่น้อยกว่า 20 หน้า ซึ่งในงานวิจัยนี้จะใช้ชุดข้อมูลความต้องซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บเผยแพร่บทความ (Web Publishing System) สำหรับบางส่วนของเอกสารทดสอบ สามารถแสดงตัวอย่างได้ดังภาพที่ 4.2

**Process: Submit Article**

Before this use case can be initiated, the Author has already connected to the Online Journal Website. The Author can chooses the Email Editor button. The System uses the sendto HTML tag to bring up the user's email system. The Author fills in the Subject line and attaches the files as directed and emails them. The System generates and sends an email acknowledgement.

**Process: Submit Review**

Before this use case can be initiated, the Reviewer has already connected to the Online Journal Website. The Reviewer chooses the Email Editor button. The System uses the sendto HTML tag to bring up the user's email system. The Reviewer fills in the Subject line and attaches the file as directed and emails it. The System generates and sends an email acknowledgement.

ภาพที่ 4.2 ตัวอย่างข้อมูลความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บเผยแพร่บทความ

**Use case: Submit Article**
**Initial Step-By-Step Description**
Before this use case can be initiated, the Author has already connected to the Online Journal Website.
1. The Author chooses the *Email Editor* button.
2. The System uses the *sendto* HTML tag to bring up the user's email system.
3. The Author fills in the Subject line and attaches the files as directed and emails them.
4. The System generates and sends an email acknowledgement.

**Use case: Submit Review**
**Initial Step-By-Step Description**
Before this use case can be initiated, the Reviewer has already connected to the Online Journal Website.
1. The Reviewer chooses the *Email Editor* button.
2. The System uses the *sendto* HTML tag to bring up the user's email system.
3. The Reviewer fills in the Subject line and attaches the file as directed and emails it.
4. The System generates and sends an email acknowledgement.

ภาพที่ 4.3 ตัวอย่างผลเฉลยข้อมูลความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บเผยแพร่บทความ

## 4.2 การประเมินกระบวนการทำความสะอาดประโยค

ในการทำความสะอาดประโยค มีวัตถุประสงค์เพื่อกำจัดคำที่เป็นคำในกลุ่มคำวิเศษณ์ ที่อยู่ตำแหน่งซ้ายมือสุดของประโยค เนื่องจากกลุ่มคำลักษณะนี้เป็นส่วนของการเกริ่นนำประโยคซึ่งไม่จำเป็นในกระบวนการสกัด *Actor* และ *Action*

สำหรับการประเมินกระบวนการทำความสะอาดประโยคนั้น เป็นการประเมินจากประโยคที่ผ่านกระบวนการทำความสะอาดประโยคด้วยเงื่อนไขที่ได้สร้างขึ้น โดยเปรียบเทียบกับผลการทำความสะอาดประโยคด้วยผู้เชี่ยวชาญด้านภาษา และใช้ตารางความคลาดเคลื่อนเพื่อช่วยในการคำนวณค่าความระลึก ค่าความแม่นยำ และค่า F1 โดยสามารถแสดงตัวอย่างผลลัพธ์การทำความ

สะอาดประโยคได้ดังตารางที่ 4.1 โดยที่ TP คือประโยคที่ทำความสะอาดประโยคได้และตรงกับผลเฉลยด้วยผู้เชียวชาญ และ FN คือประโยคที่ทำความสะอาดประโยคไม่ได้แต่ผลเฉลยด้วยผู้เชียวชาญสามารถทำความสะอาดประโยคได้

ตารางที่ 4.1 ตัวอย่างผลลัพธ์การทำความสะอาดประโยคเมื่อเทียบกับผลเฉลย

| ประโยคก่อนทำความสะอาดประโยค | |
|---|---|
| Finally, the system verifies the information and returns the Editor to the Article Manager main page. | |
| In addition, the Editor checks the status of all active articles. | |
| Firstly, the Editor selects to Assign Reviewer. | |
| Later, the Editor will send a response to an Author. | |
| To do this, the Editor selects to Receive Article, and then the system presents a choice of entering a new article or updating an existing article. | |
| Firstly, the Editor selects to publish article. | |
| ประโยคหลังผ่านการทำความสะอาดประโยค | ผลลัพธ์ |
| the system verifies the information and returns the Editor to the Article Manager main page. | TP |
| In addition, the Editor checks the status of all active articles. | FN |
| the Editor selects to Assign Reviewer. | TP |
| the Editor will send a response to an Author. | TP |
| To do this, the Editor selects to Receive Article, and then the system presents a choice of entering a new article or updating an existing article. | FN |
| the Editor selects to publish article. | TP |

ในการประเมินกระบวนการทำความสะอาดประโยค จะใช้ชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บสำหรับเผยแพร่บทความ ซึ่งมีประโยคในลักษณะข้างต้นจำนวน 32 ประโยค และเมื่อทำการทดสอบการทำความสะอาดประโยคตามกระบวนการที่นำเสนอ โดยการทำ POS tagging จากนั้นจะตัดคำวิเศษณ์ที่มี tag เป็น *IN* + “,” ที่อยู่ตำแหน่งซ้ายสุดของประโยคออกไป พบว่าให้ผลลัพธ์ดังที่แสดงในตารางที่ 4.2

ตารางที่ 4.2 ตารางความคลาดเคลื่อนของผลลัพธ์การทำความสะอาดประโยค

| | | ทำความสะอาดประโยคด้วยเงื่อนไขที่สร้างขึ้น | |
|---|---|---|---|
| | | yes | no |
| ทำความสะอาดประโยคด้วยผู้เชียวชาญด้านภาษา | yes | *27* | *5* |
| | no | *0* | *0* |

จากตารางที่ 4.2 สามารถใช้ค่าภายในตารางความคลาดเคลื่อนคำนวนเพื่อหาค่าความระลึก ค่าความถูกต้อง และค่า F1 ได้ดังสมการที่แสดงในตารางที่ 4.3

ตารางที่ 4.3 ผลลัพธ์การทำความสะอาดประโยค

| เทคนิคที่ใช้ | การคำนวณ | ผลลัพธ์ |
|---|---|---|
| ค่าความระลึก | $\frac{TP}{TP + FN} = \frac{27}{27 + 5}$ | 0.84 |
| ค่าความแม่นยำ | $\frac{TP}{TP + FP} = \frac{27}{27 + 0}$ | 1.00 |
| ค่า F1 | $2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{1 \times 0.84}{1 + 0.84} = 2 \times \frac{0.84}{1.84}$ | 0.91 |

จากผลลัพธ์ในตารางที่ 4.3 จะเห็นว่าในขั้นตอนการทำความสะอาดประโยคค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก ค่าความถูกต้อง และค่า F1 อย่างไรก็ตาม จะพบว่าในบางประโยคจะไม่สามารถทำความสะอาดได้ เนื่องจากประโยคเริ่มต้นด้วยกลุ่มคำบุพบทวลี (Prepositional Phrase) ที่ทำหน้าที่ขยายประโยค เช่นประโยคที่ขึ้นต้นด้วย “*In addition,*” และ “*To do this,*” ในตารางที่ 4.1 เป็นต้น ดังนั้น เงื่อนไขในการทำความสะอาดประโยคอาจยังไม่เพียงพอ ซึ่งอาจจะต้องมีการปรับปรุงเพิ่มเติมสำหรับงานวิจัยในอนาคต

### 4.3 การประเมินกฎในการสกัด Actor

ในโครงสร้างประโยคของเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์จะประกอบไปด้วย 2 ส่วนหลักคือ ส่วนของ *Actor* และ *Action* สำหรับ *Actor* จะแสดงถึง “ผู้กระทำ” ขณะที่ *Action* จะแสดงถึง “การกระทำหรือการทำงาน” ซึ่งกฎในการสกัด *Actor* นั้น จะใช้ในกระบวนการสกัด *Actor* หรือ “ผู้กระทำ” ที่อยู่ในตำแหน่งซ้ายสุดของประโยค

สำหรับการประเมินกฎในการสกัด *Actor* นั้น เป็นการประเมินจากจำนวนประโยคที่กฎในการสกัด *Actor* สารมารถสกัด *Actor* ออกมาได้ โดยนำผลการสกัด *Actor* แต่ละประโยค มาทำการเปรียบเทียบกับผลเฉลยด้วยผู้เชี่ยวชาญด้านภาษา และใช้ตารางความคลาดเคลื่อนเพื่อช่วยในการคำนวณค่าความระลึก ค่าความแม่นยำ และค่า F1

ในการประเมินกฎในการสกัด *Actor* จะทำการสุ่มเลือกประโยคจากชุดข้อมูล *SNLI* ซึ่งเป็นคลังข้อมูลที่รวบรวมประโยคภาษาอังกฤษที่เขียนในรูปแบบภาษาธรรมชาติ จำนวน 1,000 ประโยค และเมื่อทำการทดสอบกฎในการสกัด *Actor* ตามกระบวนการที่นำเสนอ โดยการทำ POS tagging และใช้ tag ในการเปรียบเทียบกับกฎในการสกัด *Actor* เพื่อทำการสกัดคำนามหรือกลุ่มคำนามที่อยู่ในตำแหน่งซ้ายสุดของแต่ละประโยค และทำการเปรียบเทียบกับผลเฉลยด้วยผู้เชียวชาญ ดังแสดงตัวอย่างในตารางที่ 4.4

ตารางที่ 4.4 ตัวอย่างผลลัพธ์การประเมินกฎในการสกัด Actor เมื่อเทียบกับผลเฉลย

| ประโยค | ผลลัพธ์ |
|---|---|
| [a young woman] is smiling and looking away from the camera.<br>[the elderly people] are crossing the crosswalk.<br>[a man] is sick next to the toilet.<br>[the young boy] is asleep while the woman is watching television.<br>[the police] are controlling an unruly crowd.<br>[a man] is laughing as he watches a show in a theater. | TP |
| [a man] and woman setup a camera.<br>[women] in red are performing with nun-chucks.<br>[the group] of people are smiling for their family photo.<br>[a woman] and kids climb up the.<br>[a group] of men and boys look upwards at something.<br>[a group] of people walk through tear gas. | FP |
| there are people sitting on benches outdoors.<br>swimming in the wide blue ocean.<br>is operating a small handheld electronic device.<br>there are two people walking in the sand.<br>there are a group of track runners bracing to take off. | TN |
| rose petals being thrown at an asian couple. | FN |

โดยเมื่อทำการเปรียบเทียบกับผลเฉลยด้วยผู้เชียวชาญจนครบทั้ง 1,000 ประโยคแล้ว สามารถแสดงในรูปแบบตารางความคลาดเคลื่อนได้ดังแสดงในตารางที่ 4.5 โดยที่

TP (True Positive) คือประโยคที่กฎสามารถสกัด Actor ได้และตรงกับผลเฉลย

TN (True Negative) คือประโยคที่กฎสกัด Actor ไม่ได้และตรงกับผลเฉลย

FP (False Positive) คือประโยคที่กฎสกัด Actor ได้ แต่ Actor ที่สกัดได้นั้นไม่ตรงกับผลเฉลย เหตุที่เกิดข้อผิดพลาดลักษณะนี้เป็นเพราะกฎในการสกัด *Actor* นั้น สามารถสกัดได้เพียง

คำนามหรือกลุ่มของคำนามที่อยู่ตำแหน่งซ้ายสุดของประโยคเท่านั้น แต่ *Actor* ที่สกัดได้นั้น มีกลุ่มคำอื่นๆ ที่ทำหน้าที่ขยายประธานในประโยคอยู่ Actor ที่ได้จึงไม่ตรงกับผลเฉลย

FN (False Negative) คือประโยคที่กฎสกัด Actor ไม่ได้ แต่มี Actor ในผลเฉลย เหตุที่เกิดข้อผิดพลาดลักษณะนี้เป็นเพราะผลการทำ POS Tagging มีผลในขั้นตอนการสกัด Actor ยกตัวอย่างประโยค *"rose petals being thrown at an asian couple."* ในตารางที่ 4.4 ซึ่งคำว่า *"rose petals"* เมื่อผ่านกระบวนการทำ POS Tagging ให้ผลลัพธ์เป็น *"VBD NNS"* ซึ่งเริ่มต้นประโยคด้วยคำกริยา จึงทำให้ไม่สามารถสกัด *Actor* ได้ แต่ผลเฉลยจะมองคำเดียวกันนี้เป็นกลุ่มคำนามซึ่งเป็นประธานของประโยค

ตารางที่ 4.5 ตารางความคลาดเคลื่อนของผลลัพธ์การสกัด Actor

| | | ผลการสกัดด้วย กฎในการสกัด Actor | |
|---|---|---|---|
| | | yes | no |
| ผลเฉลยด้วย ผู้เชี่ยวชาญด้านภาษา | yes | *676* | *1* |
| | no | *344* | *32* |

จากตารางที่ 4.5 สามารถใช้ค่าภายในตารางความคลาดเคลื่อนคำนวนเพื่อหาค่าความระลึก ค่าความถูกต้อง และค่า F1 ได้ดังสมการที่แสดงในตารางที่ 4.6

ตารางที่ 4.6 ผลการประเมินกฎในการสกัด Actor

| เทคนิคที่ใช้ | การคำนวณ | ผลลัพธ์ |
|---|---|---|
| ค่าความระลึก | $\frac{TP}{TP + FN} = \frac{676}{676 + 1}$ | 0.99 |
| ค่าความแม่นยำ | $\frac{TP}{TP + FP} = \frac{676}{676 + 344}$ | 0.66 |
| ค่า F1 | $2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.66 \times 0.99}{0.66 + 0.99} = 2 \times \frac{0.65}{1.65}$ | 0.79 |

จากผลลัพธ์ในตารางที่ 4.6 จะพบว่าผลการประเมินกฎในการสกัด *Actor* ค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก อย่างไรก็ตามค่าความถูกต้อง และค่า F1 จะพบว่าบางประโยคจะไม่สามารถสกัด *Actor* ได้ถูกต้อง เนื่องจากคำนามหรือกลุ่มคำนามนั้นตามด้วยคำหรือกลุ่มคำอื่นๆ ที่ทำหน้าที่ขยายคำนามหรือกลุ่มคำนาม ซึ่งเป็นประธานของประโยค โดยกลุ่มคำอื่นๆ ที่พบ เช่น บุพบทวลี โดยมีตัวอย่างประโยคดังเช่น

*“A man in an army uniform meeting someone foreign .”*

*“A group of people walk through tear gas.”*

จากตัวอย่างข้างต้น ผลลัพธ์การสกัด Actor ด้วยกฎในการสกัด Actor จะได้

*“<A man> in an army uniform meeting someone foreign .”*

*“<A group> of people walk through tear gas.”*

และจากตัวอย่างเดียวกันนี้ ผลลัพธ์ที่เฉลยโดยผู้เชี่ยวชาญทางภาษาจะได้

*“<A man in an army uniform> meeting someone foreign .”*

*“<A group of people> walk through tear gas.”*

เหตุที่เป็นเช่นนี้เป็นเพราะกฎในการสกัด *Actor* นั้น สามารถสกัดได้เพียงคำนามหรือกลุ่มของคำนามที่อยู่ตำแหน่งซ้ายสุดของประโยคเท่านั้น ดังนั้นอาจจะต้องเพิ่มคำหรือกลุ่มคำชนิดอื่นๆ ที่ทำหน้าที่ขยายคำนามสำหรับงานวิจัยในอนาคต อย่างไรก็ตาม เอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่ใช้ในงานวิจัยนี้ พบว่ามีลักษณะของ *“ประธานในประโยค”* หรือ *Actor* เป็นเพียงคำนามหรือกลุ่มของคำนามเท่านั้น จากเหตุนี้ ทำให้การสกัด *Actor* จึงไม่เกิดปัญหาใดๆ ทั้งสิ้น

**4.4 การประเมินผลการสกัด Actor**

ในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์จะประกอบไปด้วย *Actor* และ *Action* โดย *Actor* อาจจะเป็น *“บุคคล”*, *“ระบบย่อย”*, *“หน่วยงาน”*, หรือ *“แผนก”* โดยคำหรือกลุ่มคำเหล่านี้ มักจะอยู่ในรูปแบบคำนามหรือกลุ่มคำนาม ดังนั้น จึงจำเป็นต้องมีกฎในการสกัด *Actor* เพื่อใช้สำหรับการสกัดคำนามหรือกลุ่มคำนามที่เป็น *“ผู้กระทำ”* หรือ *Actor* ทั้งหมดที่อยู่ภายในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

สำหรับการประเมินในส่วนนี้ จะเป็นการประเมินจากผลลัพธ์คือ *Actor* ทั้งหมดที่สกัดได้จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ และนำผลลัพธ์ที่ได้ทำการเปรียบเทียบกับผลเฉลยด้วยผู้เชี่ยวชาญด้านภาษา และใช้ตารางความคลาดเคลื่อนเพื่อช่วยในการคำนวณค่าความระลึก ค่าความแม่นยำ และค่า F1 ซึ่งสามารถแสดงตัวอย่างผลลัพธ์การสกัด *Actor* ทั้งหมดที่อยู่ภายในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ดังตารางที่ 4.7

ตารางที่ 4.7 ตัวอย่าง Actor ทั้งหมดที่สกัดได้และเปรียบเทียบกับผลเฉลยด้วยผู้เชียวชาญ

| Actor ที่สกัดได้ | ผลเฉลยด้วยผู้เชียวชาญ | ตรงกับผลเฉลย (TP) | ไม่มีในผลเฉลย (FP) |
|---|---|---|---|
| THE AUTHOR REQUEST<br>THE EDITOR REPEAT<br>THE REVIEWER<br>THE ARTICLE<br>THE EDITOR<br>THE READER<br>THE SYSTEM<br>THE AUTHOR<br>THE WEB | THE REVIEWER<br>THE ARTICLE<br>THE EDITOR<br>THE READER<br>THE SYSTEM<br>THE AUTHOR | THE REVIEWER<br>THE ARTICLE<br>THE EDITOR<br>THE READER<br>THE SYSTEM<br>THE AUTHOR | THE AUTHOR REQUEST<br>THE EDITOR REPEAT<br>THE WEB |

ในการประเมินผลการสกัด *Actor* จะใช้ชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บสำหรับเผยแพร่บทความ และเมื่อทำการทดสอบผลการสกัด *Actor* ตามกระบวนการที่นำเสนอ โดยการทำ POS tagging และใช้ *tag* ในการเปรียบเทียบกับกฎในการสกัด *Actor* เพื่อทำการสกัดคำนามหรือกลุ่มคำนามที่อยู่ในตำแหน่งซ้ายสุดของแต่ละประโยค พบว่าให้ผลลัพธ์ดังที่แสดงในตารางที่ 4.7 และสามารถแสดงในรูปแบบตารางความคลาดเคลื่อนดังตารางที่ 4.8 โดยที่ TP คือ *Actor* ที่สกัดได้และตรงกับผลเฉลย และ FP คือ *Actor* ที่สกัดได้แต่ไม่มีในผลเฉลย โดยข้อผิดพลาดลักษณะนี้จะอธิบายในลำดับถัดไป

ตารางที่ 4.8 ตารางความคลาดเคลื่อนของผลการสกัด Actor

| | | ผลการสกัดด้วยกฎในการสกัด Actor | |
|---|---|---|---|
| | | yes | no |
| ผลเฉลยด้วยผู้เชี่ยวชาญด้านภาษา | yes | 6 | 0 |
| | no | 3 | 0 |

จากตารางที่ 4.8 สามารถใช้ค่าภายในตารางความคลาดเคลื่อน คำนวนเพื่อหาค่าความระลึก ค่าความถูกต้อง และค่า F1 ดังแสดงผลการสกัด Actor ดังตารางที่ 4.9

ตารางที่ 4.9 ผลการสกัด Actor หลังทำการลดรูปคำ

| เทคนิคที่ใช้ | การคำนวณ | ผลลัพธ์ |
|---|---|---|
| ค่าความระลึก | $\frac{TP}{TP + FN} = \frac{6}{6 + 0}$ | 1.00 |
| ค่าความแม่นยำ | $\frac{TP}{TP + FP} = \frac{6}{6 + 3}$ | 0.67 |
| ค่า F1 | $2 \times \frac{Precision \times Recall}{Precision + Recall} = 2 \times \frac{0.67 \times 1}{0.67 + 1} = 2 \times \frac{0.67}{1.67}$ | 0.8 |

จากผลลัพธ์ในตารางที่ 4.9 จะพบว่าผลการสกัด *Actor* ค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก อย่างไรก็ตามค่าความถูกต้อง และค่า F1 อาจจะได้ประสิทธิภาพที่ค่อนข้างต่ำ ซึ่งสาเหตุที่เป็นเช่นนี้เนื่องจากกระบวนการทำ POS Tagging มีผลต่อความถูกของ Actor ที่สกัดได้ ยกตัวอย่างประโยคดังเช่น

*"The Author request to submit the article by email to the Editor."*

จากตัวอย่างประโยคข้างต้น ผลลัพธ์ด้วยกฎในการสกัด Actor จะได้

*"<The Author request> to submit the article by email to the Editor."*

และจากตัวอย่างประโยคเดียวกันนี้ ผลลัพธ์ที่เฉลยโดยผู้เชี่ยวชาญทางภาษาจะได้

*"<The Author> request to submit the article by email to the Editor."*

เหตุที่เป็นเช่นนี้เป็นเพราะกระบวนการทำ POS Tagging ได้กำหนดให้คำว่า *"request"* เป็นคำนาม หรือ tag เป็น *NN* ดังนั้น กฎในการสกัด *Actor* จึงได้ผลการสกัดคือ *"The Author request"* ด้วยกฎ *"DT + NN + NN"* ในขณะที่ผู้เชียวชาญด้านภาษาได้กำหนดคำว่า *"request"* เป็นคำกริยา หรือ tag เป็น VB ดังนั้น ผู้เชียวชาญด้านภาษาจึงสามารถสกัด *Actor* ได้คือ *"The Author"* ซึ่งตรงกับรูปแบบกฎ *"DT + NN"* เป็นต้น

## 4.5 การประเมินซีนาริโอที่สกัดได้

ซีนาริโอ คือลำดับของกิจกรรมหรืองาน ที่แสดงให้เห็นแนวทางหรือพฤติกรรมของงานที่เกิดขึ้นในซอฟต์แวร์ โดยในงานวิจัยนี้คือ *Actor* และ *Action*

สำหรับการประเมินซีนาริโอที่สกัดได้นั้น เป็นการประเมินจากจำนวนของ *Action* ในแต่ละ *Actor* ที่ได้จากกระบวนการสกัด *Actor* และ *Action* โดยผลลัพธ์ที่นั้นจะนำมาเปรียบเทียบกับผลเฉลยด้วยผู้เชี่ยวชาญด้านภาษา และใช้ตารางความคลาดเคลื่อนเพื่อช่วยในการคำนวณค่าความระลึก โดยตัวอย่างชุดเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ สามารถแสดงได้ดังต่อไปนี้

*“ Before this use case can be initiated, the Author has already connected to the Online Journal Website. The Author can choose the Email Editor Button. The System uses the sendto HTML tag to bring up the user’s email system. The Author fills in the Subject line and attaches the files as directed and emails them. The System generates and sends an email acknowledgement.”*

และเมื่อทำการสกัด *Action* ของแต่ละ *Actor* จากชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์แล้ว สามารถแสดงตัวอย่างผลลัพธ์การสกัดซีนาริโอหรือ *Action* ของแต่ละ *Actor* ได้ดังตารางที่ตารางที่ 4.10

ตารางที่ 4.10 ผลลัพธ์การสกัดซีนาริโอหรือ *Action* และ *Actor*

| Actor | Action | ผลลัพท์ |
|---|---|---|
| THE SYSTEM | • use the sendto html tag to bring up the user 's email system<br>• generate and send an email acknowledgement | TP<br>TP |
| THE AUTHOR | • have already connect to the online journal website<br>• choose the email editor button<br>• fill in the subject line and attach the file a direct and email them | TP<br>TP<br>TP |
| THE AUTHOR REQUEST | • to submit [THE ARTICLE] by email to [THE EDITOR] | FP |
| THE EDITOR REPEAT | • step 3 and 4 until sufficient reviewer be assign | FP |

ในการประเมินซีนาริโอที่สกัดได้ จะใช้ชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บสำหรับเผยแพร่บทความ และเมื่อทำการสกัดซีนาริโอด้วยการสกัด *Actor* และ *Action* ตามกระบวนการที่นำเสนอได้แก่ การเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ และทำการสกัด *Actor* จากนั้นจึงกัด *Action* ของแต่ละ *Actor* พบว่าให้ผลลัพธ์ดังที่แสดงในตารางที่ 4.11 โดยที่

TP คือ *Action* และ *Actor* ที่สกัดได้ตรงกับผลเฉลยด้วยผู้เชียวชาญ

FN คือ Action และ Actor ที่ไม่สามารถสกัดได้แต่มีในผลเฉลยด้วยผู้เชียวชาญ โดยข้อผิดพลาดลักษณะนี้เกิดจาก ประโยคที่ผ่านการสกัด Action และ Actor นั้น อาจจะไม่ได้อยู่ในโครงสร้างประโยคที่เป็นลักษณะ *“Actor-Action-Object”* ดังนั้น จึงทำให้ไม่สามารถทำการสกัด Action และ Actor ที่ในประโยคได้

FP คือ *Action* และ *Actor* ที่สกัดได้แต่ไม่ตรงกับผลเฉลยด้วยผู้เชียวชาญ โดยข้อผิดพลาดลักษณะนี้เกิดจาก ประโยคที่ผ่านการสกัด *Action* และ *Actor* นั้น ในประโยคได้ถูกระบุ

ตำแหน่งของ *Actor* ที่ไม่ถูกต้อง หรือในกรณีที่คำสรรพนามถูกแทนที่ด้วย *Actor* ที่ไม่ถูกต้อง ด้วยเหตุนี้ จึงทำให้ประโยคดังกล่าว สกัดได้ *Action* และ *Actor* ที่ไม่ตรงกับผลเฉลยผู้เชียวชาญ

ตารางที่ 4.11 ตารางความคลาดเคลื่อนของผลลัพธ์การสกัดซีนาริโอ

| | | **ซีนาริโอที่สกัดได้** | |
|---|---|---|---|
| | | yes | no |
| **ผลเฉลยด้วยผู้เชียวชาญ** | yes | 298 | 42 |
| | no | 18 | 0 |

จากตารางที่ 4.11 สามารถใช้ค่าภายในตารางความคลาดเคลื่อนคำนวนเพื่อหาค่าความระลึก ได้ดังสมการที่แสดงในตารางที่ 4.12

ตารางที่ 4.12 ผลการประเมินซีนาริโอที่สกัดได้

| เทคนิคที่ใช้ | การคำนวณ | ผลลัพธ์ |
|---|---|---|
| ค่าความระลึก | $\frac{TP}{TP + FN} = \frac{298}{298 + 42}$ | 0.88 |

จากผลลัพธ์ในตารางที่ 4.12 จะพบว่าผลลัพธ์การแทนคำสรรพนามค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก อย่างไรก็ตาม บางประโยคจะพบว่า *Action* และ *Actor* ที่สกัดได้ แต่ไม่ตรงกับผลเฉลยด้วยผู้เชียวชาญ เนื่องจากเกิดข้อผิดผลาดในขั้นตอนก่อนหน้าที่ผ่านมา ก่อนถึงขั้นตอนการสกัดซีนาริโอ ดังนั้น จึงทำให้ผลลัพธ์ที่ได้จากขั้นตอนการสกัดซีนาริโอไม่ถูกต้องตามไปด้วย

# บทที่ 5

## สรุปผล อภิปรายผล และข้อเสนอแนะ

งานวิจัยนี้เป็นการวิจัยเพื่อนำเสนอการสกัดซีนาริโอและสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความบนพื้นฐานเทคนิคด้านการประมวลผลภาษาธรรมชาติ ซึ่งมีรายละเอียดดังต่อไปนี้

### 5.1 ความสำคัญและวัตถุประสงค์ของการวิจัย

1) เพื่อสกัดซีนาริโอหรือลำดับการทำงานของซอฟต์แวร์สำหรับผู้ใช้งานในแต่ละโดเมนแบบอัตโนมัติจากเอกสารความต้องการด้านซอฟต์แวร์ที่เป็นข้อความด้วยเทคนิคด้านการประมวลผลภาษาธรรมชาติ

2) เพื่อนำซีนาริโอที่ได้มาสร้างเป็นแบบจำลองกระบวนการของซอฟต์แวร์ (Process Modeling) แบบอัตโนมัติ

### 5.2 สรุปกระบวนการ

กระบวนการในการดำเนินการวิจัยนี้ ได้แบ่งออกเป็น 2 ส่วนหลัก คือการสร้างกฎเพื่อวิเคราะห์โครงสร้างทางภาษา และการสกัด *Actor* และ *Action*

สำหรับกฎเพื่อวิเคราะห์โครงสร้างทางภาษานั้น ใช้สำหรับการสกัด *Actor* และ *Action* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยกฎที่ใช้จะแบ่งออกเป็น 3 กลุ่มตามจุดประสงค์การใช้งานดังต่อไปนี้

กฎในการสกัด *Actor* ใช้สำหรับสกัดคำนามหรือกลุ่มคำนามในประโยค ดังนั้น คำนามหรือกลุ่มคำนามเหล่านี้จึงสามารถกำหนดเป็น *Actor* ซึ่งอาจจะเป็น *“บุคคล”*, *“ระบบย่อย”*, *“หน่วยงาน”*, หรือ *“แผนก”* ซึ่งการสร้างกฎในการสกัด *Actor* ได้ใช้ชุดข้อมูล *SNLI* จำนวน 300,000 ประโยค และประกอบไปด้วยขั้นตอนการทำความสะอาดประโยค การหาชุดลำดับของ tag ของกลุ่มคำนามในประโยคด้วย Parser การเรียนรู้และสร้างกฎในการสกัด *Actor* ด้วยแบบจำลองมาร์คอฟ โดยสามารถสร้างกฎในการสกัด *Actor* ได้ทั้งหมดจำนวน 584 กฎ และกฎที่จะถูกใช้จริงสำหรับชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ มีจำนวนเพียง 3 กฎ

กฎในการแทนคำสรรพนามนี้จะใช้ในขั้นตอนการสกัด *Actor* เพื่อช่วยแก้ปัญหาในกรณีที่ประโยคขึ้นต้นด้วยคำสรรพนาม หรือ tag เป็น *PRP* ด้วยการแทนที่ตำแหน่งของคำสรรพนามที่พบด้วย *Actor* ที่สกัดได้จากประโยคก่อนหน้า โดยทำการหารูปแบบการแทนคำสรรพนามจากประโยคที่มีความต่อเนื่องกัน จากนั้นนำรูปแบบการแทนคำสรรพนามไปทำการเรียนรู้และสร้างกฎจากแบบจำลองมาร์คอฟซ่อนเร้น ซึ่งสามารถสร้างกฎในการแทนคำสรรพนามได้ทั้งหมดจำนวน 231 กฎ และเมื่อใช้กฎกับชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ มีจำนวนเพียง 3 กฎ

กฎในการแยกประโยคความรวม จะใช้ในการวิเคราะห์โครงสร้างประโยคเพื่อทำการแยกประโยคย่อยที่อยู่ในประโยคความรวม หรือที่ในงานวิจัยนี้เรียกว่า *"Object"* ซึ่งมีจำนวน *Object* ตามจำนวนประโยคย่อยที่อยู่ในประโยคความรวม โดยกระบวนการสร้างกฎได้ใช้ชุดข้อมูล *SNLI* ที่เป็นลักษณะประโยคความรวม และประกอบด้วยขั้นตอนการหาตำแหน่งของคำสันธานด้วยการทำ POS Tagging และค้นหา tag ที่เป็นคำสันธานคือ *IN* เพื่อใช้เป็นตำแหน่งในการหาชุดลำดับคำของประโยคย่อยหรือ *Object* ที่อยู่ในประโยคความรวม จากนั้นจึงทำการเรียนรู้และสร้างกฎด้วยแบบจำลองมาร์คอฟร่วมกับแนวคิดข้อจำกัด โดยสามารถสร้างกฎในการแยกประโยคความรวมได้ทั้งหมดจำนวน 39 กฎ และกฎที่ถูกใช้จริงสำหรับชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ มีจำนวน 9 กฎ

กระบวนการในการสกัด *Actor* และ *Action* เป็นการนำกฎรูปแบบต่างๆ ที่กล่าวถึงข้างต้นมาทำการสกัดซีนาริโอ หรือ *Action* ของแต่ละ *Actor* โดยชุดข้อมูลที่ใช้ในงานวิจัยนี้คือเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ที่เกี่ยวกับการพัฒนาเว็บเผยแพร่บทความ ซึ่งเป็นภาษาอังกฤษ ที่มีความยาวต่อเนื่องไม่น้อยกว่า 20 หน้า โดยกระบวนการในการสกัด *Actor* และ *Action* มีขั้นตอนดังต่อไปนี้

การเตรียมเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ขั้นตอนนี้เป็นการกรองคำหรือกลุ่มคำที่ไม่จำเป็นของแต่ละประโยคออกไป ซึ่งในงานวิจัยนี้เรียกว่า *"การทำความสะอาดประโยค"* โดยเริ่มจากการทำ POS Tagging และจากนั้นจะตัดคำวิเศษณ์ หรือคำที่มี tag เป็น *RB* ที่อยู่ตำแหน่งซ้ายสุดของประโยคและมีเครื่องหมาย "," ตามหลังออกไป

การสกัด *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ด้วยกฎในการสกัด *Actor* ขั้นตอนนี้เป็นการใช้กฎในการสกัด *Actor* และกฎในการแทนคำสรรพนาม เพื่อสกัดหา *Actor* ทั้งหมดในชุดเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ โดยนำชุดข้อมูลที่ผ่านการทำความ

สะอาดประโยคมาทำการเทียบกับกฎที่ใช้ในการสกัด *Actor* ที่อยู่ในแต่ละประโยค เมื่อได้ *Actor* ทั้งหมดแล้วจะทำการหาความแตกต่างร่วมกับการลดรูปคำเพื่อไม่ให้เกิด *Actor* ที่ซ้ำซ้อน สุดท้ายจะได้ *Actor* ทั้งหมดที่อยู่ในชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

การสกัด *Action* ของแต่ละ *Actor* จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ได้แบ่งการทำงานออกเป็น 3 ขั้นตอนย่อย ได้แก่การระบุตำแหน่งของ *Actor* การสกัดรูปแบบโครงสร้างของประโยคด้วยกฎในการแยกประโยคความรวม และการสกัด *Action* และ *Actor* ออกจาก *Object*

การระบุตำแหน่งของ *Actor* เป็นการนำ *Actor* ทั้งหมดที่สกัดได้จากขั้นตอนก่อนหน้า มาทำการค้นหาและระบุตำแหน่ง *Actor* ที่อยู่ภายในเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ สำหรับการนำไปใช้ในขั้นตอนถัดไป คือขั้นตอนการสกัด *Action* ของแต่ละ *Actor*

การสกัดรูปแบบโครงสร้างของประโยคด้วยกฎในการแยกประโยคความรวม ในขั้นตอนนี้จะเน้นไปที่ประโยคความรวม โดยทำการสกัดรูปแบบของประโยคย่อยหรือ *Object* ก่อนที่จะนำ *Object* เหล่านั้นที่สกัดได้จากประโยคความรวม โดยที่ *Object* ทั้งหมดที่สกัดได้จากเอกสารความต้องการของซอฟต์แวร์จะถูกนำไปสกัดเพื่อหา *Action* และ *Actor* ในขั้นตอนถัดไป

การสกัด *Action* และ *Actor* ออกจาก *Object* โดยการทำงานหรือ *Action* เหล่านี้สามารถอธิบายถึงพฤติกรรมหรือการทำงานระหว่าง *Actor* กับระบบ หรืออาจเป็น *Actor* อื่นๆ ภายในกระบวนการของซอฟต์แวร์ โดยภายโครงสร้างประโยคภายใน *Object* จะเป็น *"Actor-Action-Object"* โดยสามารถสกัด *Actor* จากตำแหน่งที่ถูกระบุไว้ในขั้นตอนก่อนหน้า ดังนั้น จึงสามารถสกัดข้อความที่อยู่หลัง *Actor* เป็นการกระทำหรือ *Action* ที่สอดคล้องกับตำแหน่งของ *"Action-Object"* ในโครงสร้างประโยคของเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์นั่นเอง

เมื่อทำการสกัดซีนาริโอ หรือ *Action* ของแต่ละ *Actor* ทั้งหมดจากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์แล้ว จะทำการรวบรวมและจัดกลุ่มของ *Action* ของแต่ละ *Actor* และทำการแสดงผลลัพธ์ของแบบจำลองกระบวนการซอฟต์แวร์ในรูปแบบข้อมูลกึ่งโครงสร้าง

### 5.3 สรุปผลและอภิปราย

กระบวนการในการดำเนินการวิจัยนี้ ได้แบ่งการประเมินผลการวิจัยเป็น 4 ด้าน ได้แก่

1) การประเมินกระบวนการทำความสะอาดประโยค เมื่อประเมินด้วยค่าความระลึก ค่าความถูกต้อง และค่า F1 ปรากฏว่าให้ผลลัพธ์ที่ 0.84, 1.00 และ 0.91 ตามลำดับ ซึ่งค่อนข้างให้

ประสิทธิภาพที่ดี แต่อย่างไรก็ตาม จะพบว่าในบางประโยคจะไม่สามารถทำความสะอาดได้ เนื่องจากประโยคเริ่มต้นด้วยการเกริ่นนำประโยคด้วยกลุ่มคำอื่นๆ ที่อยู่นอกเหนือจากเงื่อนไขการทำความสะอาดประโยค เช่น บุพบทวลี เป็นต้น

2) การประเมินกฎในการสกัด *Actor* ค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก โดยสามารถทำได้ถึง 0.99 แต่อย่างไรก็ตามค่าความถูกต้องและค่า F1 ปรากฏว่าให้ผลลัพธ์ที่ 0.66 และ 0.79 ตามลำดับ ซึ่งเป็นผลลัพธ์ที่ค่อนข้างต่ำ เนื่องจากพบว่าบางประโยคจะไม่สามารถสกัด *Actor* ได้ถูกต้อง เนื่องจากคำนามหรือกลุ่มคำนามนั้น มีกลุ่มคำอื่นๆ ที่ทำหน้าที่ขยายประธานของประโยคประกอบอยู่ด้วย เช่นบุพบทวลี ซึ่งกฎในการสกัด *Actor* นั้น สามารถสกัดได้เพียงคำนามหรือกลุ่มคำนามที่อยู่ตำแหน่งซ้ายสุดของประโยคเท่านั้น ทั้งนี้ผลการประเมินจะขึ้นอยู่กับชุดข้อมูลที่นำมาใช้ในการประเมินกฎในการสกัด *Actor* ว่ามีความเหมาะสมกับกฎในการสกัด *Actor* เพียงใด

3) การประเมินผลการสกัด *Actor* ประเมินด้วยค่าความระลึก ค่าความถูกต้อง และค่า F1 ปรากฏว่าให้ผลลัพธ์ที่ 1.00, 0.62 และ 0.8 ตามลำดับ โดยค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก แต่อย่างไรก็ตามค่าความถูกต้องและค่า F1 อาจจะได้ประสิทธิภาพที่ค่อนข้างต่ำ ซึ่งสาเหตุที่เป็นเช่นนี้เนื่องจากกระบวนการทำ POS Tagging มีผลต่อความถูกต้องในกระบวนการสกัด Actor จากเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์ ทำให้ *Actor* ที่สกัดได้นั้น ไม่ตรงกับผลเฉลยด้วยผู้เชียวชาญ ซึ่งจะอธิบายในหัวข้อต่อไป

4) การประเมินซีนาริโอที่สกัดได้ ค่อนข้างให้ประสิทธิภาพที่ดีเมื่อประเมินด้วยค่าความระลึก ปรากฏว่าให้ผลลัพธ์ที่ 0.88 อย่างไรก็ตาม จะพบว่าในบางประโยคจะไม่สามารถสกัดซีนาริโอหรือ *Actor* และ *Action* ได้ เนื่องจากโครงสร้างประโยคอาจจะไม่ได้อยู่ในรูปแบบ *“Actor-Action-Object”* นั่นเป็นเพราะกฎเพื่อการวิเคราะห์โครงสร้างภาษาในงานวิจัยนี้ ถูกสร้างขึ้นเพื่อสกัดซีนาริโอจากโครงสร้างประโยคลักษณะดังกล่าวเท่านั้น และในกรณีที่เกิดข้อผิดผลาดในขั้นตอนก่อนหน้าที่จะมาถึงขั้นตอนการสกัดซีนาริโอ ดังนั้น จึงทำให้ผลลัพธ์ที่ได้จากขั้นตอนการสกัดซีนาริโอไม่ถูกต้องตามไปด้วย

## 5.4 ปัญหาที่พบ

ปัญหาที่พบในงานวิจัยฉบับนี้ แบ่งออกเป็น 2 ประเด็น

1) ปัญหาที่พบในกระบวนการทำความสะอาดประโยค โดยเงื่อนไขการทำความสะอาดประโยคคือการกำจัดคำวิเศษณ์ที่อยู่ตำแหน่งซ้ายสุดของประโยคออกไป ซึ่งพบว่าบางประโยคจะไม่สามารถทำความสะอาดได้ เนื่องจากพบกลุ่มคำอื่นๆ ที่ไม่ใช่คำวิเศษณ์ นั่นคือคำบุพบทวลีที่ทำหน้าที่ขยายประโยค เช่น “*In addition*,” และ “*To do this*,” เป็นต้น ดังนั้น เงื่อนไขในการทำความสะอาดประโยคอาจจะต้องเพิ่มกลุ่มคำชนิดอื่นๆ สำหรับงานวิจัยในอนาคต

2) กระบวนการทำ POS Tagging เป็นกระบวนการมีผลต่อความถูกต้องในการสกัดซีนาริโอจากเอกสารความต้องการของซอฟต์แวร์ จะพบว่าบางคำนั้น สามารถเป็นได้ทั้งคำนามหรือคำกริยาขึ้นอยู่กับบริบทในประโยค ดังนั้น จึงมีผลต่อการวิเคราะห์ประโยคเพื่อจัดรูปแบบประโยคออกเป็น *“Actor-Action-Object”* ซึ่งหากไม่สามารถจัดเป็นโครงสร้างประโยครูปแบบดังกล่าวได้ จะทำให้ไม่สามารถทำการสกัดซีนาริโอ หรือ *Action* และ *Actor* ได้ ด้วยเหตุนี้ ทำให้ผลลัพธ์ที่ได้ในด้านของค่าความแม่นยำจะค่อนข้างต่ำ ยกตัวอย่างประโยค *“The Author request to submit the article by email to the Editor.”* หากคำว่า *“request”* นั้นถูกกำหนดให้เป็นคำนามหรือ *NN* จะทำให้ *Actor* ที่สกัดได้เป็น *“The Author request”* แต่หากถูกกำหนดให้เป็นคำกริยาหรือ *VB* จะทำให้ *Actor* ที่สกัดได้เป็น *“The Author”* เป็นต้น

## 5.5 ข้อเสนอแนะ

อาจต้องมีการเพิ่มเงื่อนไข หรือกลุ่มคำชนิดอื่นๆ ในกระบวนการสร้างกฎในแต่ละรูปแบบเพื่อให้ได้กฎที่ครอบคลุมการสกัดซีนาริโอจากรูปแบบประโยคที่หลากหลายยิ่งขึ้นในอนาคต และควรมีการใช้เครื่องมืออื่นๆ สำหรับกระบวนการทำ POS Tagging เพื่อทำการเปรียบเทียบประสิทธิภาพของผลการสกัดซีนาริโอ ว่าเครื่องมือสำหรับการทำ POS Tagging ใด ที่ให้ประสิทธิภาพสูงสุดสำหรับกระบวนการสร้างแบบจำลองกระบวนการของซอฟต์แวร์แบบอัตโนมัติจากความต้องการของซอฟต์แวร์ที่เป็นข้อความ

# บรรณานุกรม


[1] Makino M, Ohnishi A. Alternative/Exceptional Scenario Generation with Differential Scenario; 2008.

[2] Canfora G, García F, Piattini M, Ruiz F, Visaggio CA. A family of experiments to validate metrics for software process models. Journal of Systems and Software 2005; 77113-129.

[3] Reijers HA, Mendling J. A Study Into the Factors That Influence the Understandability of Business Process Models. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 2011; 41449-462.

[4] Sutcliffe AG, Maiden NAM, Minocha S, Manuel D. Supporting scenario-based requirements engineering. IEEE Transactions on Software Engineering 1998; 241072-1088.

[5] Fantechi A, Gnesi S, Lami G, Maccari A. Application of linguistic techniques for Use Case analysis; 2002.

[6] Gargantini A, Riccobene E, Scandurra P, Carioni A. Scenario-based validation of embedded systems; 2008.

[7] Go K, Carroll JM. Scenario-based task analysis. The handbook of task analysis for human-computer interaction 2003; 117

[8] Kazman R, Abowd G, Bass L, Clements P. Scenario-Based Analysis of Software Architecture. IEEE software 1996; 1347-55.

[9] Acuna ST, Antonio AD, Ferre X, Lopez M, Mate L, Estero S, et al. The Software Process: Modelling, Evaluation and Improvement. 2000.

[10] Hurtado Alegria JA, Bastarrica MiC, Bergel A. Analyzing Software Process Models with AVISPA. New York, NY, USA: ACM; 2011.

[11] Kaur R, Sengupta J. Software process models and analysis on failure of software development projects. arXiv preprint arXiv:13061068 2013;

[12] Gruhn V. Validation and verification of software process models BT - Software Development Environments and CASE Technology. Berlin, Heidelberg: Springer Berlin Heidelberg; 1991.

[13] Humphrey WS, Kellner MI. Software Process Modeling: Principles of Entity Process Models. New York, NY, USA: ACM; 1989.

[14] Bures T, Hnetynka P, Kroha P, Simko V. Requirement specifications using natural languages.

[15] MacDonell SG, Min K, Connor AM. Autonomous requirements specification processing using natural language processing. arXiv preprint arXiv:14076099 2014;

[16] Kof L. Natural Language Processing: Mature Enough for Requirements Documents Analysis? BT - Natural Language Processing and Information Systems. Berlin, Heidelberg: Springer Berlin Heidelberg; 2005.

[17] Yang H, de Roeck A, Gervasi V, Willis A, Nuseibeh B. Analysing anaphoric ambiguity in natural language requirements. Requirements Engineering 2011; 16163.

[18] Shah US, Jinwala DC. Resolving Ambiguities in Natural Language Software Requirements: A Comprehensive Survey. SIGSOFT Softw Eng Notes 2015; 401-7.

[19] Curtis B, Kellner MI, Over J. Process Modeling. Commun ACM 1992; 3575-90.

[20] Mishra A, Sureka A. Automatic detection of semantic inconsistency between BPMN process model and SBVR rule model. 2015.

[21] Dale R, Moisl H, Somers H. Handbook of natural language processing. 2000;

[22] Marcus MP, Marcinkiewicz MA, Santorini B. Building a Large Annotated Corpus of English: The Penn Treebank. Comput Linguist 1993; 19313-330.

[23] Jurafsky D, Martin JH. Speech and language processing. 2014; 3

[24] Grinstead CMCM, Snell JLJL, Snell JLJL. Introduction to probability. 1997; 510.

[25] Indurkhya N, Damerau FJ. Handbook of natural language processing. 2010; 2

[26] Jurafsky D, Martin JH. Speech and language processing. Pearson London; 2014.

[27] Müller T, Cotterell R, Fraser A, Schütze H. Joint lemmatization and morphological tagging with lemming; 2015.

[28] Friburger N, Maurel D. Finite-state transducer cascades to extract named entities in texts. Theoretical Computer Science 2004; 31393-104.

[29] Hobbs JR. Information extraction from biomedical text. Journal of Biomedical Informatics 2002; 35260-264.

[30] Kotsiantis SB, Zaharakis I, Pintelas P. Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer

engineering 2007; 1603-24.

[31] Soderland S. Learning Information Extraction Rules for Semi-Structured and Free Text. Machine Learning 1999; 34233-272.

[32] Riloff E, Jones R, others. Learning dictionaries for information extraction by multi-level bootstrapping; 1999.

[33] Hastie T, Tibshirani R, Friedman J. Unsupervised Learning BT - The Elements of Statistical Learning: Data Mining, Inference, and Prediction. In: Hastie T, Tibshirani R, Friedman J, eds. New York, NY: Springer New York 2009:485-585.

[34] Sekine S. On-demand Information Extraction. Stroudsburg, PA, USA: Association for Computational Linguistics; 2006.

[35] Tsai C-F, Hsu Y-F, Lin C-Y, Lin W-Y. Intrusion detection by machine learning: A review. Expert Systems with Applications 2009; 3611994-12000.

[36] Feldman R, Rosenfeld B, Fresko M. TEG—a hybrid approach to information extraction. Knowledge and Information Systems 2006; 91-18.

[37] Li Y, Bontcheva K, Cunningham H. SVM Based Learning System for Information Extraction BT - Deterministic and Statistical Methods in Machine Learning. Berlin, Heidelberg: Springer Berlin Heidelberg; 2005.

[38] Dolk DR, Konsynski BR. Knowledge Representation for Model Management Systems. IEEE Transactions on Software Engineering 1984; SE-10619-628.

[39] !!! INVALID CITATION !!!;

[40] Kof L. Natural language processing for requirements engineering: Applicability to large requirements documents. 2004;

[41] Fantechi A, Gnesi S, Ristori G, Carenini M, Vanocchi M, Moreschini P. Assisting requirement formalization by means of natural language translation. Formal Methods in System Design 1994; 4243-263.

[42] Goldin L, Berry DM. AbstFinder, a prototype abstraction finder for natural language text for use in requirements elicitation: design, methodology, and evaluation; 1994.

[43] Ambriola V, Gervasi V. Processing natural language requirements; 1997.

[44] Huyck C, Abbas F. Natural Language Processing and Requirements Engineering: a Linguistics Perspective; 2000.

[45] Flores JJG. Semantic Filtering of Textual Requirements Descriptions BT - Natural Language Processing and Information Systems. Berlin, Heidelberg: Springer Berlin Heidelberg; 2004.

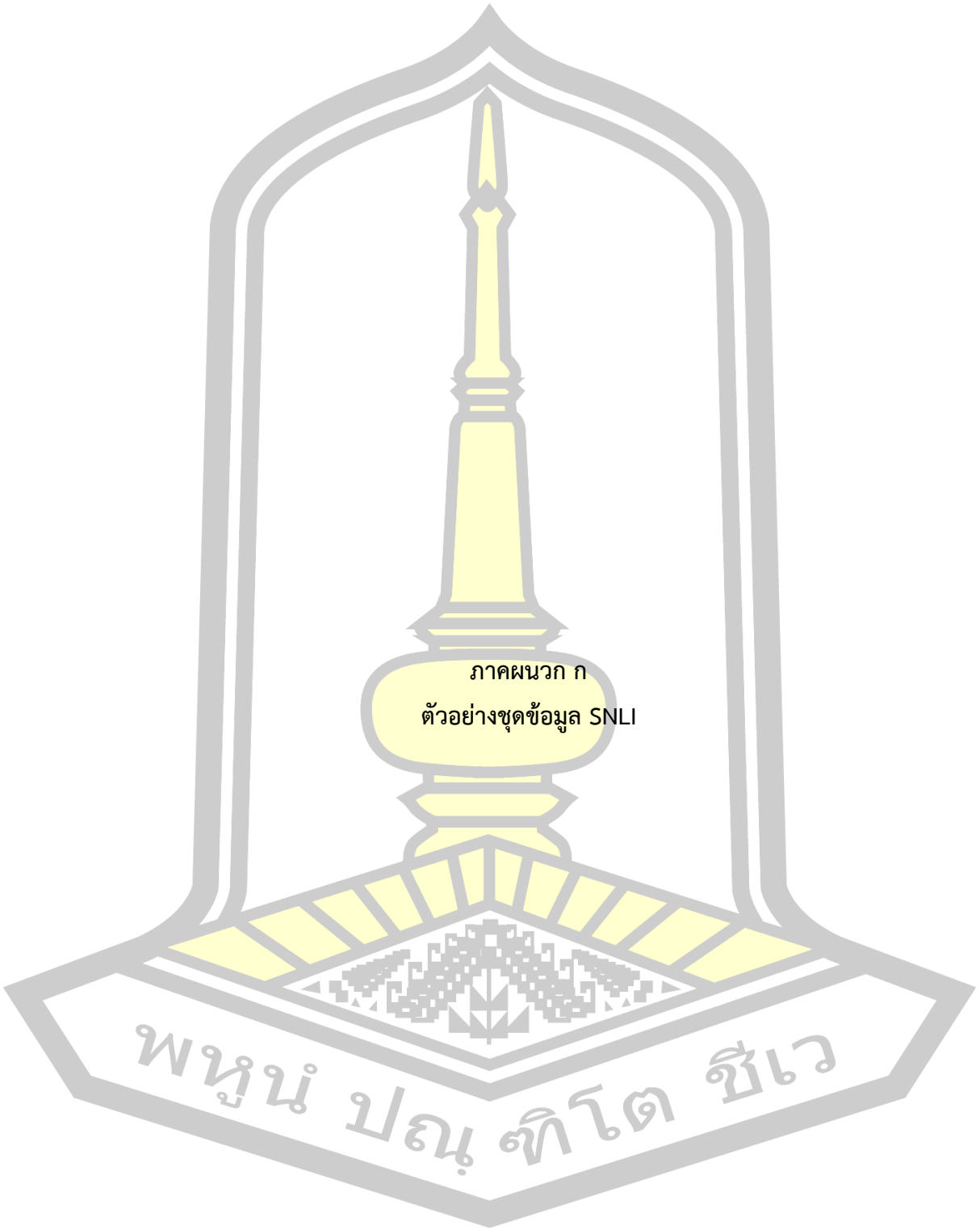
[46] Ibrahim M, Ahmad R. Class Diagram Extraction from Textual Requirements Using Natural Language Processing (NLP) Techniques; 2010.

[47] Herchi H, Abdessalem WB. From user requirements to UML class diagram. arXiv preprint arXiv:12110713 2012;

[48] Bowman SR, Angeli G, Potts C, Manning CD. A large annotated corpus for learning natural language inference. arXiv preprint arXiv:150805326 2015;

[49] Miltsakaki E, Prasad R, Joshi AK, Webber BL. The Penn Discourse Treebank.; 2004.

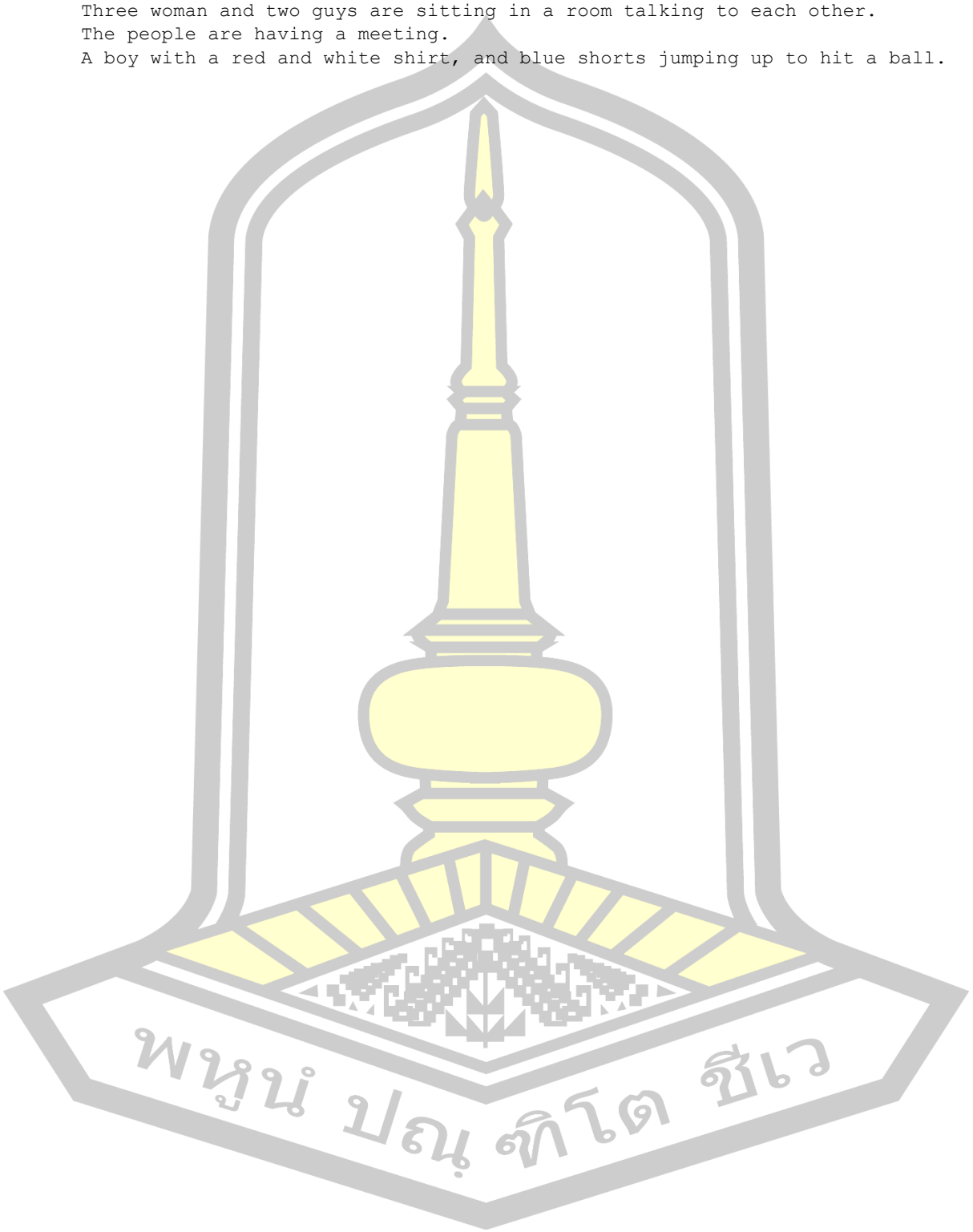
# ภาคผนวก

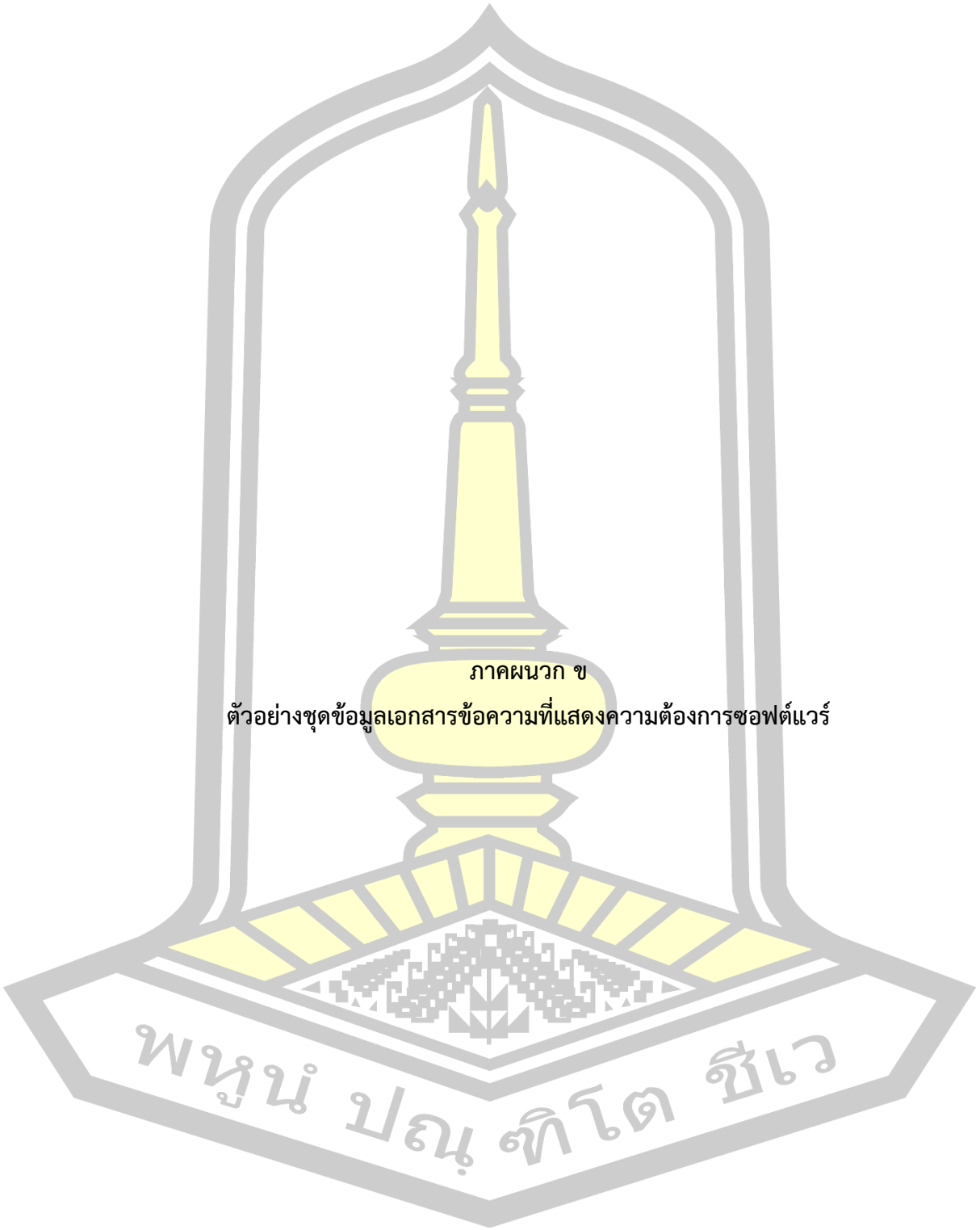


# ภาคผนวก ก
# ตัวอย่างชุดข้อมูล SNLI

Two women carrying grain on their head.
Two women collect grain to feed their families before the upcoming famine in the land.
Two women carrying grain on their head.
Two women are chatting while they carry grain.
An African woman carries a parcel on her head.
The woman is returning with her clean laundry.
A girl in a blue school uniform climbs a tree and stops for a picture.
A schoolgirl is sitting on a tree branch.
Women carrying items on their heads walk down a dirt path with greenery on each side.
Woman are carrying fruit baskets on their heads.
A woman walks up a path that cuts through cultivated plots while balancing straw on her head.
A woman walks up a wooded path.
A woman walks up a path that cuts through cultivated plots while balancing straw on her head.
The woman is bringing straw to the stables to feed the horses.
A woman walks up a path that cuts through cultivated plots while balancing straw on her head.
The woman is having a hard time balancing the straw on her head.
A woman walks up a path that cuts through cultivated plots while balancing straw on her head.
A woman is walking down a path while carrying straw and balancing some on her head.
A woman walks up a path that cuts through cultivated plots while balancing straw on her head.
The woman is walking by herself.
White dove glides above water.
Some birds are gliding back home.
A woman cooks a variety of meets on a grill while others stand in the background.
A woman is grilling hotdogs for her family.
A large white bird flies out of the water.
The dog scared the bird out of the water.
A large white bird flies out of the water.
The large white bird is a duck.
A woman in a white apron is cooking all kinds of food over a grill.
There is a womb cooking outdoors.
A woman grilling a variety of different kinds of meats at a cookout.
A woman grills meat for a family reunion.
Two women wearing coats and carrying bags walk down a street.
Two women walk down the street during a shopping trip.
Two dogs are chasing each other in a yard.
the dog is a lab.
A gray labradoodle jumps over another large dog.
A gray labradoodle jumps over another large hunting dog.
Two dogs playing in grass.
the dogs are playing fetch.
A group of people discussing the merits of the mullet.
a group of people discussing about mullets because it is a passion.
A group of people discussing the merits of the mullet.
some people are arguing and yelling at each other.
A group of 70's or 80's era Caucasian men and women seated and having a discussion.
the people are wearing crazy clothes.
Four people in a room sitting down while a woman in blue talks to another woman sitting behind her.
Four people in a room sitting down while a woman in blue talks to another woman sitting behind her on a rocking chair.

Two people sitting in orange carts.
People sitting down.
Three woman and two guys are sitting in a room talking to each other.
The people are having a meeting.
A boy with a red and white shirt, and blue shorts jumping up to hit a ball.





# ภาคผนวก ข

## ตัวอย่างชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์

**Submit Review**

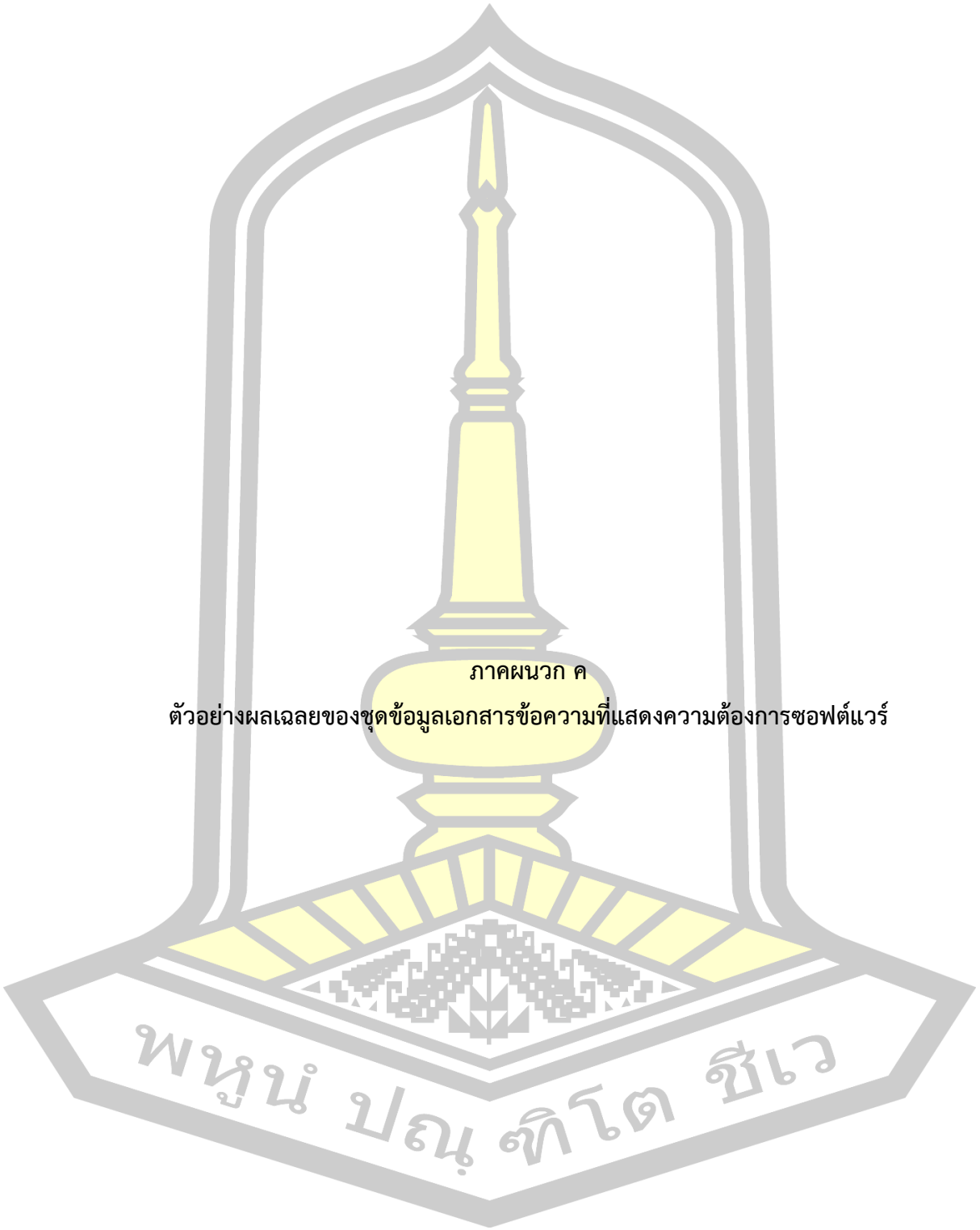
Before the Reviewer submits a review of an article, the Reviewer has already connected to the Online Journal Website. To submit the article, the Reviewer needs to submit a review of an article to the Editor. The System should provide the button of the email Editor on the Website for the Reviewer. The System should provide the HTML tag to bring up the user's email system. After the Reviewer fills in the information form and attaches the files as directed and emails it, the System generates and sends an email acknowledgement.

**Search Article**

The Reader has already accessed the Online Journal Website. The Reader chooses to search by author name, category, or keyword. Afterwards, the system displays the choices to the Reader. The Reader will select the article desired, and then the system presents the abstract of the article to the reader. The Reader chooses to download the article and the system provides the requested article.

**Update Author**

Before the Editor enters a new Author or updates information about a current Author, the Editor has already accessed the main page of the Article Manager. When the Editor selects to *Add/Update Author*, the system presents a choice of adding or updating. The Editor chooses to add or to update. If the Editor is updating an Author, the system presents a list of authors to choose from and presents a grid filling in with the information; else the system presents a blank grid. The Editor fills in the information and submits the form. Finally, the system verifies the information and returns the Editor to the Article Manager main page.



**ภาคผนวก ค**

**ตัวอย่างผลเฉลยของชุดข้อมูลเอกสารข้อความที่แสดงความต้องการซอฟต์แวร์**

## Use Case: Search Article

### Initial Step-By-Step Description

Before this use case can be initiated, the Reader has already accessed the Online Journal Website.

1. The Reader chooses to search by author name, category, or keyword.
2. The system displays the choices to the Reader.
3. The Reader selects the article desired.
4. The system presents the abstract of the article to the reader.
5. The Reader chooses to download the article.
6. The system provides the requested article.

## Use Case: Submit Review

### Initial Step-By-Step Description

Before this use case can be initiated, the Reviewer has already connected to the Online Journal Website.

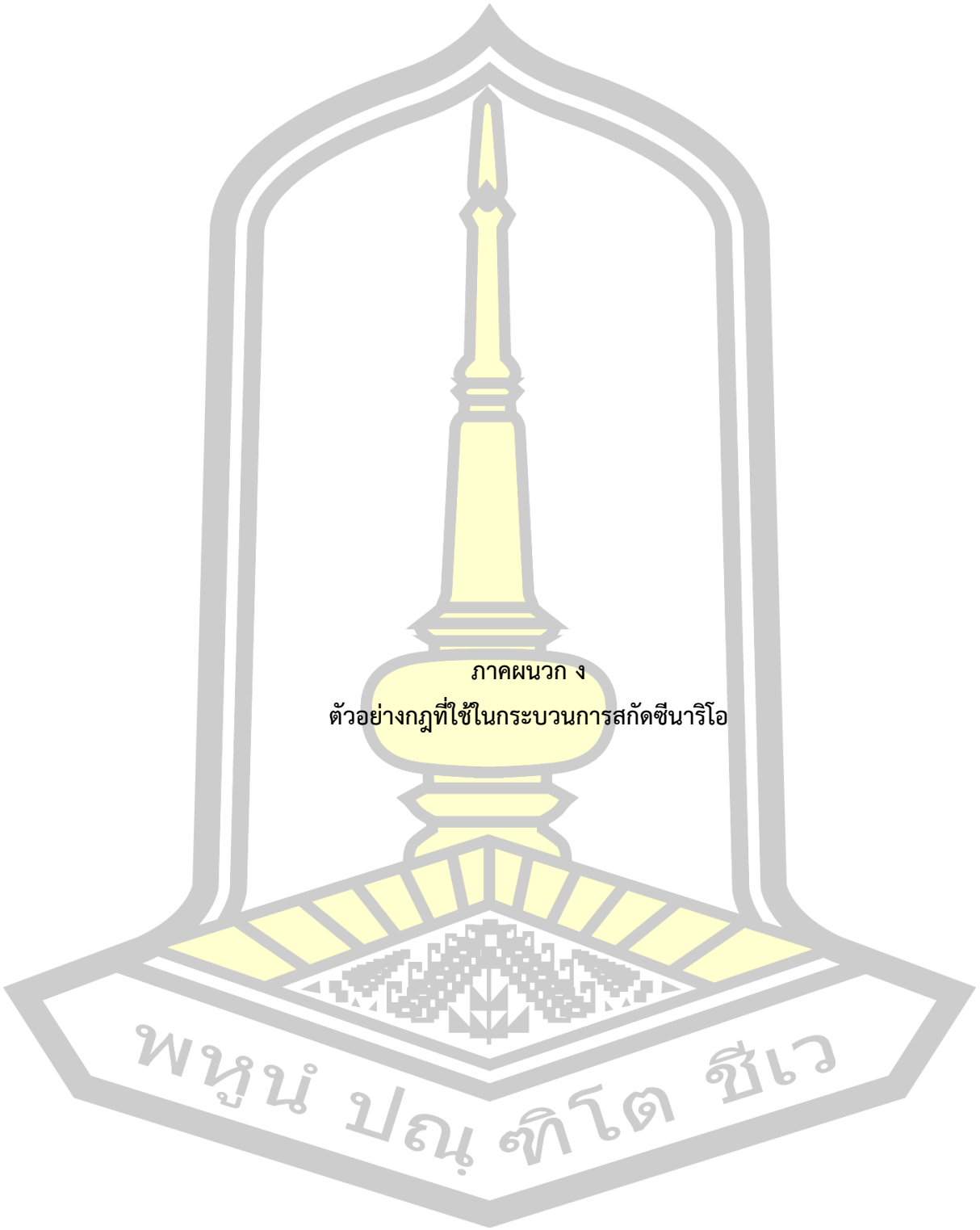
5. The Reviewer chooses the *Email Editor* button.
6. The System uses the *sendto* HTML tag to bring up the user's email system.
7. The Reviewer fills in the Subject line and attaches the file as directed and emails it.
8. The System generates and sends an email acknowledgement.

## Use Case: Submit Review

### Initial Step-By-Step Description

Before this use case can be initiated, the Editor has already accessed the main page of the Article Manager.

1. The Editor selects to *Add/Update Author*.
2. The system presents a choice of adding or updating.
3. The Editor chooses to add or to update.
4. If the Editor is updating an Author, the system presents a list of authors to choose from and presents a grid filling in with the information; else the system presents a blank grid.
5. The Editor fills in the information and submits the form.
6. The system verifies the information and returns the Editor to the Article Manager main page.




# ภาคผนวก ง
# ตัวอย่างกฎที่ใช้ในกระบวนการสกัดซีนาริโอ

| กฎในการสกัด Actor | กฎที่ถูกใช้งานจริง |
|---|---|
| JJ | DT NN |
| NN | DT NNP |
| NNP | DT NN NN |
| NNS | |
| CD JJ | |
| CD NNS | |
| DT JJ | |
| DT NN | |
| DT NNS | |
| JJ NN | |
| JJ NNS | |
| NN CD | |
| NN NN | |
| NN NNS | |
| NNS NN | |
| NNS NNS | |
| CD NNS NN | |
| CD NNS NNS | |
| DT JJ JJ | |
| DT JJ NN | |
| DT JJ NNS | |
| DT NN CD | |
| DT NN NN | |
| DT NN NNS | |

| กฎในการแทนคำสรรพนาม | กฎที่ถูกใช้งานจริง |
|---|---|
| DT JJ NN --> he | DT NN --> They |
| DT NN NN --> he | DT NNP --> It |
| DT NN --> he | DT NNP --> They |
| DT NN --> it | |
| JJ NN --> it | |
| CD NN --> he | |
| CD NN --> it | |
| NN NN --> it | |
| JJ JJ --> it | |
| JJ NN --> he | |
| NN NN --> he | |
| NN IN --> he | |
| NN IN --> it | |
| NN CD --> he | |
| NN JJ --> he | |
| NNP --> they | |
| NN --> they | |
| NNP --> she | |
| NNS --> she | |
| NNPS --> it | |
| NNP --> he | |
| NNS --> it | |
| NN --> she | |
| NNP --> it | |
| NNS --> he | |
| PDT --> it | |
| NN --> he | |
| NN --> it | |

| กฎในการแยกประโยคความรวม | กฎที่ถูกใช้งานจริง |
|---|---|
| Obj1 after Obj2 , and Obj3 | after $1 , $2 , and then $3 |
| Obj1 else Obj2 before Obj3 | if $1 , $2 else $3 |
| Obj1 , and Obj2 when Obj3 | $1 , and then $2 |
| Obj1 after Obj2 else Obj3 | before $1 , $2 |
| Obj1 when Obj2 , and Obj3 | after $1 , $2 |
| Obj1 after Obj2 when Obj3 | when $1 , $2 |
| after Obj1 , Obj2 , and then Obj3 | if $1 , $2 |
| if Obj1 , Obj2 else Obj3 | $1 , $2 |
| Obj1 then Obj2 then Obj3 | $1 |
| Obj1 , and Obj2 if Obj3 | |
| Obj1 , and then Obj2 | |
| Obj1 else after Obj2 | |
| Obj1 , before Obj2 | |
| before Obj1 , Obj2 | |
| after Obj1 , Obj2 | |
| Obj1 if Obj2 if Obj3 | |
| Obj1 , after Obj2 | |
| Obj1 before Obj2 | |
| Obj1 , when Obj2 | |
| Obj1 , then Obj2 | |
| Obj1 after Obj2 | |
| Obj1 , and Obj2 | |
| when Obj1 , Obj2 | |
| Obj1 when Obj2 | |
| Obj1 then Obj2 | |
| if Obj1 , Obj2 | |
| Obj1 , if Obj2 | |
| Obj1 if Obj2 | |
| Obj1 , Obj2 | |
| Obj1 | |

พหุน ปณ ฑิโต ชีเว

# ภาคผนวก จ

## ตัวอย่างแบบจำลองกระบวนการของซอฟต์แวร์ที่สร้างจากซีนาริโอที่สกัดได้

| ACTOR_ID | ACTOR_NAME |
|---|---|
| A | THE AUTHOR REQUEST |
| B | THE EDITOR REPEAT |
| C | THE REVIEWER |
| D | THE ARTICLE |
| E | THE EDITOR |
| F | THE READER |
| G | THE SYSTEM |
| H | THE AUTHOR |
| I | THE WEB |

| ACTION_ID | THE AUTHOR REQUEST ACTION |
|---|---|
| A1 | to submit [THE ARTICLE] by email to [THE EDITOR] |

| ACTION_ID | THE EDITOR REPEAT ACTION |
|---|---|
| B1 | to submit [THE ARTICLE] by email to [THE EDITOR] |

| ACTION_ID | THE REVIEWER ACTION |
|---|---|
| C1 | return their comment |
| C2 | submit a review of an article |
| C3 | have already connect to the online journal website |
| C4 | need to submit a review of an article to [THE EDITOR] |
| C5 | fill in the information form and attach the file a direct and email it |
| C6 | complete their review |

| ACTION_ID | THE ARTICLE ACTION |
|---|---|
| D1 | be accept a write ,1 decline |
| D2 | be accept ,1 possibly after a revision |
| D3 | be publish to the online journal |

| ACTION_ID | THE EDITOR ACTION |
|---|---|
| E1 | access the entire system directly |
| E2 | outline the process for each user separately a follow |
| E3 | enter it into [THE SYSTEM] and assign it to and send it to at least three reviewer |
| E4 | send a copyright form to [THE AUTHOR] |
| E5 | enter a new author or update information about a current author have already access the main page of [THE ARTICLE] manager |
| E6 | select to add/update author |
| E7 | choose to add or to update |
| E8 | be update an author |
| E9 | fill in the information and submit the form |
| E10 | enter a new reviewer or update information about a current |
| E11 | reviewer |
| E12 | have already access the main page of [THE ARTICLE] manager |
| E13 | select to add/update reviewer |
| E14 | choose to add or to update |
| E15 | be update a reviewer |
| E16 | fill in the information form and submit it |
| E17 | receive an article |
| E18 | enter a new or revise article into [THE SYSTEM] |
| E19 | have already access the main page of [THE ARTICLE] manager and have a file contain [THE ARTICLE] available |
| E20 | select to receive article |
| E21 | choose to add or to update |
| E22 | be update an article |

| | |
|---|---|
| E23 | fill in the information and submit the form |
| E24 | will assign one or more reviewer to an article |
| E25 | have already access [THE ARTICLE] use the update article use case |
| E26 | select to assign reviewer |
| E27 | select a reviewer |
| E28 | have already access the main page of [THE ARTICLE] manager |
| E29 | will select to receive review |
| E30 | fill the review in the information form and submit it |
| E31 | will send a response to an author |
| E32 | have already access the main page of [THE ARTICLE] manager |
| E33 | select to the response by a button |
| E34 | fill out the email text and send the message |
| E35 | a already access the main page of [THE ARTICLE] manager |
| E36 | select to check status |
| E37 | remove [THE ARTICLE] from the active category |
| E38 | have already access the main page of [THE ARTICLE] manager |
| E39 | need to select to remove an article from the active database |
| E40 | select an article for removal |
| E41 | will transfer an accept article to the online journal |
| E42 | have already access the main page of [THE ARTICLE] manager |
| E43 | select to publish article |

| ACTION_ID | THE READER ACTION |
|---|---|
| F1 | have already access the online journal website |
| F2 | choose to search by author name ,1 category ,2 or keyword |
| F3 | will select [THE ARTICLE] desire |
| F4 | choose to download [THE ARTICLE] and [THE SYSTEM] provide the request article |

| ACTION_ID | THE EDITOR ACTION |
|---|---|
| G1 | display the choice to [THE READER] |
| G2 | present the abstract of [THE ARTICLE] to [THE READER] |
| G3 | provide the button of the email editor on the website for [THE AUTHOR] |
| G4 | provide the html tag to bring up the user 's email system |
| G5 | generate and send an email acknowledgement |
| G6 | provide the button of the email editor on the website for [THE REVIEWER] |
| G7 | provide the html tag to bring up the user 's email system |
| G8 | generate and send an email acknowledgement |
| G9 | present a choice of add or update |
| G10 | present a list of author to choose from and present a grid fill in with the information; |
| G11 | present a blank grid |
| G12 | verify the information and return [THE EDITOR] to [THE ARTICLE] manager main page |
| G13 | present a choice of add or update |
| G14 | link to the historical society database |
| G15 | and present a grid with the information about [THE REVIEWER] ; |
| G16 | present list of member for [THE EDITOR] to select a reviewer and present a grid for the person select |
| G17 | will verify the information and return [THE EDITOR] to [THE ARTICLE] manager main page |
| G18 | present a choice of enter a new article or update an exist article |
| G19 | present a list of article to choose from and present a grid for fill with the information ; |
| G20 | present a blank grid |
| G21 | verify the information and return [THE EDITOR] to [THE ARTICLE] manager main page |
| G22 | present a list of reviewer with their status |

| | |
|---|---|
| G23 | verify that the person be still an active member use the historical society database |
| G24 | email [THE REVIEWER] |
| G25 | return [THE EDITOR] to the update article use case |
| G26 | present a grid for fill with the information |
| G27 | verify the information and return [THE EDITOR] to [THE ARTICLE] manager main page |
| G28 | return [THE EDITOR] to [THE ARTICLE] manager main page |
| G29 | return a scrollable list of all active article with their status |
| G30 | return [THE EDITOR] to [THE ARTICLE] manager main page |
| G31 | provide a list of article with the status of each |
| G32 | remove [THE ARTICLE] from the active article database |
| G33 | return [THE EDITOR] to [THE ARTICLE] manager main page |
| G34 | will transfer [THE ARTICLE] to the online journal and update the search information there |
| G35 | remove [THE ARTICLE] from the active article database and return [THE EDITOR] to [THE ARTICLE] manager home page |

| ACTION_ID | THE AUTHOR ACTION |
|---|---|
| H1 | be ask to make some change base on the review |
| H2 | either submit an original article or resubmit an edit article |
| H3 | have already connect to the online journal website |
| H4 | fill in the information form and attach the file a direct and email it |

| ACTION_ID | THE WEB ACTION |
|---|---|
| I1 | publish system have four active actor and one cooperate system |
| I2 | be [THE AUTHOR] ,1 reader |

พหุน ปณ ฑิโต ชีเว

| | |
|---|---|
| **ชื่อ** | นายคณิน งามมานะ |
| **วันเกิด** | วันที่ 3 มกราคม พ.ศ.2531 |
| **สถานที่เกิด** | อำเภอเมือง จังหวัดสกลนคร |
| **สถานที่อยู่ปัจจุบัน** | บ้านเลขที่ 214 หมู่ 6 ตำบลพังขว้าง อำเภอเมือง จังหวัดสกลนคร รหัสไปรษณีย์ 47000 |
| **ตำแหน่งหน้าที่การงาน** | นักวิชาการคอมพิวเตอร์ |
| **สถานที่ทำงานปัจจุบัน** | มหาวิทยาลัยราชภัฏสกลนคร เลขที่ 680 ตำบลธาตุเชิงชุม อำเภอเมือง จังหวัดสกลนคร รหัสไปรษณีย์ 47000 |
| **ประวัติการศึกษา** | พ.ศ. 2548 ประกาศนียบัตรวิชาชีพ วิทยาลัยเทคนิคสกลนคร อำเภอเมือง จังหวักสกลนคร |
| | พ.ศ. 2553 วิศวกรรมศาสตรบัณฑิต (วศ.บ.) สาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเทคโนโลยีมหานคร |
| | พ.ศ. 2562 วิทยาศาสตรมหาบัณฑิต (วท.ม.) สาขาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยมหาสารคาม |

พหูน ปณฺฑิโต ชีเว