# บทที่ 5
# สรุปผลการศึกษาและข้อเสนอแนะ

## 5.1 สรุปผลการศึกษา

จากปัญหาของการพัฒนาซอฟต์แวร์ทั่วไป เมื่อเริ่มการเก็บความต้องการของระบบ โดยการสอบถามข้อมูลความต้องการจากผู้ที่จะใช้งาน ทำการวิเคราะห์ความต้องการ จัดทำเอกสารความต้องการซอฟต์แวร์ที่จะพัฒนา และเมื่อได้รายละเอียดความต้องการแล้ว จะทำการวางแผนการพัฒนา ตั้งแต่การกำหนดระยะเวลาและงบประมาณที่ต้องใช้ หลังจากนั้นนักพัฒนาจึงเริ่มการวิเคราะห์และออกแบบซอฟต์แวร์ เสร็จแล้วจึงแบ่งงานกำหนดหน้าที่รับผิดชอบให้กับผู้ที่เกี่ยวข้องในโครงการ ระหว่างการพัฒนามีการติดตามการพัฒนาให้เป็นไปตามแผนและงบประมาณที่วางไว้ กระบวนการต่างๆ เหล่านี้ครอบคลุมการพัฒนาซอฟต์แวร์ทั้งหมด ซึ่งเป็นงานใหญ่และเกี่ยวข้องกับคนหลายๆ กลุ่ม กว่าจะพัฒนาซอฟต์แวร์เสร็จ ก็ต้องอาศัยความร่วมมือของกลุ่มคนหลายๆ กลุ่ม และต้องดูแลเอาใจใส่เป็นอย่างมาก เมื่อโครงการเสร็จแล้วและต้องพัฒนาซอฟต์แวร์กับระบบอื่นอีก การพัฒนาจะต้องเริ่มใหม่ทั้งหมดอีกรอบ ฉะนั้นหากเป็นไปได้ ควรจะมีการนำเอาสิ่งที่ได้พัฒนาไปแล้วนั้นกลับมาใช้ได้กับโครงการใหม่อีก โดยนำเอาองค์ประกอบของซอฟต์แวร์ที่เคยได้พัฒนาไปแล้วนำกลับมาใช้ได้อีก หรือแม้กระทั่งแนวทางและกระบวนการพัฒนาซอฟต์แวร์ที่ดีก็นำกลับมาใช้ได้อีก เพื่อทำให้โครงการใหม่ ลดระยะเวลาและงบประมาณลง พร้อมทั้งทำให้ซอฟต์แวร์ที่ได้จากการพัฒนา มีมาตรฐานและคุณภาพด้วย

จากการศึกษาวิจัยการพัฒนาซอฟต์แวร์ด้วยเอ็มดีเอ สามารถช่วยลดงานที่ต้องใช้ในการพัฒนา โดยเอ็มดีเอเป็นเครื่องมือการพัฒนาซอฟต์แวร์ที่ครอบคลุมถึงการวิเคราะห์และออกแบบพัฒนา ซึ่งโครงสร้างการพัฒนาจะถูกแบ่งออกเป็น 4 ช่วงได้แก่

1. การออกแบบโดเมนโมเดล (Domain Modeling)
2. การออกแบบสถาปัตยกรรมโมเดล (Architecture Modeling)
3. การออกแบบซอฟต์แวร์ประยุกต์โมเดล (Application Model)

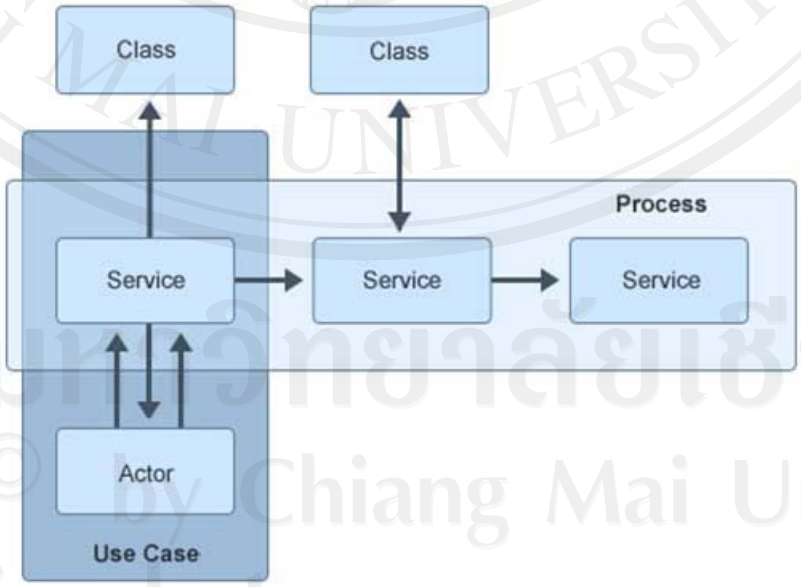
4. โมเดลของโค้ดโปรแกรม (Code Modeling)

### 5.1.1 การออกแบบโดเมนโมเดล (Domain Modeling)

โดเมนโมเดลเป็นส่วนขับเคลื่อนหลักของการเก็บองค์ประกอบความต้องการต่างๆ ของระบบติดตามงานวิจัย กระบวนการทำงานต่างๆ จะถูกออกแบบออกมาในรูปของโมเดล โดยไม่ขึ้นกับเทคโนโลยีใดๆ โดเมนโมเดลจะพยายามค้นหาความต้องการจากมุมมองของผู้ใช้งาน นั่นหมายความว่า เราจะสนใจการทำงานของผู้ใช้ระบบเป็นหลัก ปัจจุบันโดเมนโมเดลประกอบไปด้วยโมเดลต่างๆ ดังนี้

- ยูสเคสโมเดล (Use-Case Model)
- คลาสโมเดล (Class Model)
- เซอร์วิสโมเดล (Service Model)
- โปรเซสโมเดล (Process Model)

ทั้ง 4 โมเดล จำเป็นตัวกำหนดมุมมองซอฟต์แวร์ที่ได้ขององค์กรและเป็นตัวกำหนดการทำงานระหว่างผู้ใช้กับระบบ ความสัมพันธ์ระหว่างส่วนประกอบต่างๆ ขององค์กรและโมเดล ดังแสดงในรูปที่ 5.1




*รูปที่ 5.1 แสดง The Business Object and Their Associations*

การออกแบบยูสเคสโมเดล (Use-Case Model) จะแสดงให้เห็นการติดต่อระหว่างแต่ละบทบาทของผู้ใช้กับระบบติดตามงานวิจัย (Actor กับ System) ซึ่งจะแสดงมุมมองภายนอกของซอฟต์แวร์ที่ผู้ใช้ทำกับระบบ ยูสเคสจะประกอบไปด้วยลำดับขั้นตอนต่างๆ ที่ผู้ใช้กระทำกับระบบโดยขึ้นกับงานต่างๆ ขององค์กรที่มีหรือเปรียบกับงานต่างๆ ขององค์กรได้กับเซอร์วิสโมเดลนั่นเอง

คลาสโมเดล คือโมเดลของข้อมูลที่เกี่ยวเนื่องกับองค์กร (เช่น งานวิจัย กิจกรรม ผลของกิจกรรม และอื่นๆ) และความสัมพันธ์ระหว่างแต่ละโมเดลด้วย การอกอแบบคลาสโมเดล ผู้วิเคราะห์และออกแบบคลาสโมเดล จะต้องเป็นผู้ที่มีความรู้ความเข้าใจในความสัมพันธ์ระหว่างโมเดลด้วย เพื่อเวลาที่ออกแบบมาแล้ว สามารถนำไปใช้กับเอ็มดีเอได้อย่างมีประสิทธิภาพมากที่สุด และสามารถนำเอาโมเดลที่ได้ออกแบบไปแล้วนั้น นำไปใช้ได้กับการพัฒนาซอฟต์แวร์อื่นๆได้อีก ส่วนการออกแบบคลาสโมเดลนี้ จะเป็นส่วนที่สำคัญที่สุดของขั้นตอนการพัฒนาซอฟต์แวร์ด้วยเอ็มดีเอ เพราะการแก้ไขโดยการมองภาพรวมของระบบด้วยโมเดลที่เป็นรูปภาพ จะทำให้เข้าใจได้ง่ายกว่า สำหรับการแก้ไขโมเดลนั้น อาจไม่ได้มาจากความผิดพลาดจากการออกแบบทั้งหมด ซึ่งปกติการพัฒนาซอฟต์แวร์นั้น ความต้องการของผู้ใช้อาจจะมีการเปลี่ยนแปลงได้อีก การแก้ไขก็เริ่มกลับมาที่โมเดลเช่นเดียวกัน ซึ่งหลังจากการแก้ไข ตัวเครื่องมือก็สร้างส่วนต่างๆ ที่ได้ถูกแก้ไขไปจนถึงการสร้างโค้ดโปรแกรมใหม่ ทำให้ลดระยะเวลาในการพัฒนาซอฟต์แวร์ไปได้มาก เนื่องจากการพัฒนาซอฟต์แวร์ด้วยเอ็มดีเอ จะต้องกำหนดแบบโครงสร้างสถาปัตยกรรมไว้ก่อน ดังนั้นการสร้างส่วนต่างๆ ของระบบติดตามงานวิจัยหลักจากการแก้ไขโมเดลให้ตรงกับความต้องการมากที่สุดแล้ว สามารถกำหนดให้เป็นแนวทางมาตรฐานได้ ซึ่งการเลือกแบบโครงสร้างสถาปัตยกรรมนักออกแบบสามารถเลือกใช้ให้เหมาะกับขนาดและโครงสร้างของซอฟต์แวร์และโครงการที่จะทำการพัฒนาได้

เซอร์วิสโมเดล คือโมเดลการทำงานขององค์กร (Function Model) ที่ได้จากงานต่างๆของระบบติดตามงานวิจัย เซอร์วิสโมเดลถูกกำหนดเป็นงานที่ถูกใช้จากเครื่องลูกข่ายหนึ่งๆ (One Client)โดยถูกเรียกใช้งานเวลาหนึ่งๆ หรืออาจจะมองได้ว่าเซอร์วิสโมเดลจะให้บริการให้กับการทำงานของแต่ละคลาสโมเดล เมื่อออกแบบคลาสโมเดลแล้ว เครื่องมือจะสร้างเซอร์วิสโมเดลให้ ซึ่งในเซอร์วิสโมเดลนี้ นักพัฒนาสามารถที่จะปรับปรุงแก้ไขรายละเอียดได้อีก เป็นการปรับปรุง

เพื่อให้ได้การออกแบบตรงกับความต้องการมากที่สุด ในขั้นตอนนี้นักพัฒนาจะแก้ไขคุณลักษณะและความสัมพันธ์ของเซอร์วิสโมเดลเป็นหลัก ในคลาสโมเดลกับเซอร์วิสโมเดลจะเป็นโมเดลในขั้นตอนพีไอเอ็มโมเดล

โปรเซสโมเดล แสดงอยู่ในรูปของลำดับการทำงานของเซอร์วิสโมเดล ที่จำเป็นต้องถูกเรียกทำงาน ทำให้สามารถกำหนดลำดับเงื่อนไขการทำงานต่างๆ ที่องค์กรต้องการได้ แต่ในการค้นคว้าวิจัยนี้ยังไม่ใช้โปรเซสโมเดล

โดเมนโมเดลต่างๆ สามารถนำมารวมกันเป็นต้นแบบได้ เรียกว่า โมเดลต้นแบบอาร์คีไทก์ (Archetype Pattern) โดยที่โมเดลต้นแบบอาร์คีไทก์สามารถนำกลับมาใช้ได้อีก โดยนักออกแบบสามารถเพิ่มเติมโดเมนโมเดลอื่นๆ ได้ตามความต้องการของซอฟต์แวร์ใหม่ในองค์กรอื่นๆ ที่มีโครงสร้างการทำงานพื้นฐานเหมือนกันแต่รายละเอียดอื่นๆ ที่อาจจะแตกต่างกันไป โดเมนโมเดลของระบบติดตามงานวิจัย สามาราถนำไปเป็นต้นแบบอาร์คีไทก์ได้ในระดับหนึ่ง หากจะให้สมบูรณ์จะต้องปรับปรุงแก้ไขต่อไปอีก ซึ่งการศึกษาวิจัยนี้ได้ทำไว้เป็นแนวทางเท่านั้น

**5.1.2การออกแบบโครงสร้างโมเดล (Architecture Modeling)**

หลังจากที่สร้างโดเมนโมเดลแล้ว จะต้องตัดสินใจว่าจะเลือกโครงสร้างโมเดลแบบไหนในการศึกษาระบบติดตามงานวิจัย ได้เลือกใช้เครื่องมืออีซีโอ 6.0 (ECO 6.0) ซึ่งรองรับโครงการการพัฒนาบนพื้นฐานของเอเอสพีดอทเน็ทเฟรมเวิค4.0

การเปลี่ยนรูป (Transformation) ของโมเดลจะขึ้นกับเทคโนโลยีที่เลือกใช้ ในแบบโครงสร้างโมเดลจะประกอบไปด้วยรูปแบบของการเปลี่ยนรูปต่างๆที่จับคู่กับสมาชิกของแต่ละโมเดลไว้ เพื่อที่จะเปลี่ยนรูปโมเดลให้ได้ตามแบบโครงสร้างของซอฟต์แวร์ที่จะพัฒนา การเปลี่ยนรูปประกอบไปด้วยสองขั้นตอนได้แก่ การเปลี่ยนรูปจากโมเดลไปเป็นโมเดล (Model-to-Model) และการเปลี่ยนจากโมเดลไปเป็นโค้ด (Model-to-Code)

**5.1.3การออกแบบซอฟต์แวร์โมเดล (Software Modeling)**

การออกแบบซอฟต์แวร์โมเดลของระบบติดตามงานวิจัย เป็นขั้นตอนหลัก เพื่อสร้างส่วนต่างๆ ของซอฟต์แวร์โปรแกรม เช่น โครงสร้างของซอฟต์แวร์ (Application Architecture) หลักเกณฑ์และเงื่อนไขการทำงานของซอฟต์แวร์ (Application Rules) องค์ประกอบของซอฟต์แวร์และการติดต่อระหว่างองค์ประกอบนั้นๆ (Component) แบบซอฟต์แวร์โมเดลจะขึ้นกับเทคโนโลยีที่ใช้แต่ยังไม่มีการสร้างโค้ดโปรแกรม ซอฟต์แวร์โมเดลของระบบติดตามงานวิจัย ประกอบไปด้วยโมเดลย่อยดังต่อไปนี้

- โมเดลของฐานข้อมูล (Database Model) ประกอบด้วยโครงสร้างและตารางฐานข้อมูล (Schema และ Tables)
- โมเดลของเงื่อนไขการทำงานขององค์กร (Business Logic Model) ประกอบไปด้วยองค์ประกอบหลักๆ ที่ทำให้การทำงานของซอฟต์แวร์เป็นไปตามเงื่อนไข
- โมเดลของส่วนแสดงผล (Presentation Model) ประกอบไปด้วยส่วนติดต่อกับผู้ใช้ต่างๆ

**5.1.4โมเดลของโค้ดโปรแกรม (Code Modeling)**

โมเดลของโค้ดโปรแกรมในระบบติดตามงานวิจัย ประกอบไปด้วยโค้ดทั้งหมดที่มีอยู่ในซอฟต์แวร์ ได้แก่กลุ่มของแฟ้มข้อมูลโค้ด (Source Files) สคริปต่างๆ (Script) แฟ้มข้อมูลรายละเอียดต่างๆ ที่ซอฟต์แวร์ต้องใช้ (Descriptors)

อีซีโอ 6.0 สร้างการทำงานของโค้ดโปรแกรมมาจากซอฟต์แวร์โมเดลตรงๆ เช่น สร้างแฟ้มข้อมูลเอสคิวแอล(SQL: Structured Query Language) มาจากโมเดลของฐานข้อมูล และสร้างเอเอสพีเอ็กซ์(ASPX: Active Server Page Extended File) ในการเปลี่ยนรูปของโมเดลไปเป็นโค้ดจะถูกกำหนดโดยต้นแบบ (Patterns) ซึ่งสัมพันธ์กับโครงสร้างของโค้ดที่จำเป็นต้องมีในซอฟต์แวร์โมเดลและภาษาโปรแกรมที่เลือก เช่น จาวาหรือเอ็กซ์เอ็มแอล ถึงแม้จะสร้างแฟ้มข้อมูลมาหลายรูปแบบ แต่ก็มีรูปแบบที่เหมือนกัน ได้แก่

- ภายในแฟ้มข้อมูลที่สร้างขึ้นต่างถูกแบ่งออกเป็นส่วนป้องกัน (Guarded Block) และส่วนอิสระ (Free Block) ส่วนป้องกันจะประกอบไปด้วยโค้ดโปรแกรมที่ถูกจัดการโดนต้นแบบ (Pattern) ซึ่งสามารถอ่านได้อย่างเดียว ไม่สามารถแก้ไขได้ ส่วนอิสระถูกกำหนดขึ้นให้นักพัฒนาเพิ่มโค้ดโปรแกรมได้เองเพื่อจัดการกับเงื่อนไขการทำงานของซอฟต์แวร์
- การสร้างโค้ดจะสัมพันธ์กับองค์ประกอบซอฟต์แวร์โมเดล หรือเอ็มซีอาร์ (MCR: Model-Code Repository) เอ็มซีอาร์จะคอยจัดการสร้างแฟ้มข้อมูลที่จำเป็นต้องมีอยู่ในระบบ ไม่ว่าแฟ้มข้อมูลนั้นถูกลบไปแล้วหรือไม่ เมื่อองค์ประกอบของซอฟต์แวร์โมเดลถูกลบออกไป เอ็มซีอาร์ก็จะลบแฟ้มข้อมูลที่เกี่ยวข้องกับองค์ประกอบโมเดลที่ถูกลบไปด้วย
- เอ็มซีอาร์ยังช่วยในเรื่องจองโมเดลไปเป็นโค้ดด้วย (Model-to-Model) นักพัฒนาสามารถเลือกโมเดลและสามารถไปยังแฟ้มข้อมูลต่างๆ ที่ถูกสร้างจากโมเดลนั้นได้ โดยการเลือกใช้จากเมนู
- ตัวสร้างโค้ดของ อีซีโอ 6.0ยังสร้างการเชื่อมต่อกับซอฟต์แวร์ต่างๆที่ใช้ในการพัฒนาได้อีกด้วย และแบบโครงสร้างซอฟต์แวร์ที่ใช้เป็นหลักคือ ASP.Net Framework 4.0

**5.2 บทสรุป**

จากที่กล่าวมาแล้วนั้น จะเห็นว่าการพัฒนาซอฟต์แวร์ด้วยเครื่องมือ อีซีโอ 6.0 ทำให้การออกแบบและพัฒนาซอฟต์แวร์ได้รวดเร็ว อีกทั้งยังสามารถติดตั้งซอฟต์แวร์ที่ได้จากเอเอสพีดอทเน็ตสำหรับองค์กรได้อย่างมีประสิทธิภาพ เรียกได้ว่าเป็นเอจายล์(Agile) พัฒนาซอฟต์แวร์ด้วยพื้นฐานของเอ็มดีเอ ประกอบไปด้วยโมเดลต่างๆ ที่เป็นเอกลักษณ์ มีการเอาแบบโครงสร้างซอฟต์แวร์ประยุกต์ที่ดีมาใช้การพัฒนาซอฟต์แวร์ด้วย อีซีโอ 6.0 บนพื้นฐานของเอ็มดีเอสามารถนำเอาโมเดลนั้นกลับมาใช้กับการพัฒนาซอฟต์แวร์อื่นได้อีก ซึ่งเครื่องมือ อีซีโอ 6.0 ก็สามารถทำได้อย่างดีและรวดเร็ว

**5.3 ข้อเสนอแนะ**

ปัญหาที่พบในระหว่างการพัฒนานั้นคือตัวเครื่องมือ อีซีโอ 6.0 ที่มีความสามารถที่ครอบคลุมนั้น มีความซับซ้อนและต้องการศึกษาค่อนข้างมาก โดยการศึกษาไม่ได้มีเฉพาะการใช้งานเครื่องมือ อีซีโอ 6.0 เท่านั้น ยังต้องมีความเข้าใจในหลักการของโมเดลด้วย ซึ่งโมเดลที่ออกแบบจะเป็นโมเดลมาตรฐานที่ออกแบบด้วยยูเอ็มแอล(UML) และมาตรฐานการพัฒนาซอฟต์แวร์ด้วยเอ็มดีเอ ก็ถูกกำหนดจากโอเอ็มจี (OMG) อีกทั้งตัวเครื่องมือ อีซีโอ 6.0ไม่ได้เป็นโอเพนซอสซอฟต์แวร์ด้วย ทำให้การจัดหามาใช้สำหรับการพัฒนาอาจต้องมีค่าใช้จ่ายสำหรับการจัดหาเครื่องมือ อีซีโอ 6.0มาใช้ด้วย ซึ่งในการศึกษานี้ผู้ศึกษาได้ใช้ตัวซอฟต์แวร์ อีซีโอ 6.0ที่เป็นตัวทดลองใช้เท่านั้น แต่จากการศึกษาวิจัยนี้ทำให้เห็นว่าการพัฒนาซอฟต์แวร์โดยมองไปที่โมเดลของธุรกิจและใช้เครื่องมือเข้ามาช่วยเป็นอีกแนวทางหนึ่งที่ทำให้การนำเอาโมเดลกลับมาใช้อีกโดยไม่ต้องเริ่มออกแบบใหม่ทั้งหมด

ถ้าหากมองในแง่ของการลงทุนในเรื่องของเครื่องมือสำหรับพัฒนาแล้ว ถึงแม้ว่าเครื่องมือจะมีราคาสูงแต่ก็คุ้มค่าหากนำมาใช้ในการพัฒนาซอฟต์แวร์ ซึ่งในการพัฒนาซอฟต์แวร์ในแต่ละครั้ง ถ้าหากใช้เครื่องมือของเอ็มดีเอนั้นก็สามารถนำเอาโมเดลจากการพัฒนาซอฟต์แวร์ที่ได้พัฒนาไปแล้ว นำไปใช้ใหม่ได้ง่ายและรวดเร็ว ซึ่งเห็นได้ว่าช่วยให้ประหยัดเวลาในการพัฒนาเป็นอย่างมาก เพราะต้นทุนทางด้านเวลาไม่สามารถประเมินออกมาเป็นตัวเงินได้ง่ายๆ อีกทั้งยังลดขั้นตอนการพัฒนาซอฟต์แวร์ที่ทำซ้ำๆ กันออกไป ทำให้ใช้เวลาที่เหลือมาเพิ่มความถูกต้องและตรวจสอบประสิทธิภาพของซอฟต์แวร์ได้มากยิ่งขึ้น