การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น

SOFTWARE APPLICATION DESIGN USING LAYERED ARCHITECTURE

การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น

SOFTWARE APPLICATION DESIGN USING LAYERED ARCHITECTURE

วสันต์ มากธนะรุ่ง

การศึกษาเฉพาะบุคคลเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

วิทยาศาสตรมหาบัณฑิต

มหาวิทยาลัยกรุงเทพ

พ.ศ. 2554

มหาวิทยาลัยกรุงเทพ

ข้อตกลงว่าด้วยการอนุญาตให้ใช้สิทธิในวิทยานิพนธ์/สารนิพนธ์

วันที่ 1 เดือน กรกฎาคม พ.ศ. 2554

ข้าพเจ้า (นาย/นาง/นางสาว) วสันต์ มาตอนะรัง อยู่บ้านเลขที่ 67/33
ซอย ประชาราษฎร์ 10 ถนน ประชาราษฎร์ ตำบล/แขวง บางซื่อ
อำเภอ/เขต บางซื่อ จังหวัด กรุงเทพ รหัสไปรษณีย์ 10800
เป็นนักศึกษาของมหาวิทยาลัยกรุงเทพ รหัสประจำตัว 7520700142
ระดับปริญญา ☐ ตรี ☑ โท ☐ เอก
หลักสูตร วิทยาศาสตรมหาบัณฑิต สาขาวิชา เคมีอินทรีย์ประยุกต์ คณะ บัณฑิตวิทยาลัย
ซึ่งต่อไปนี้เรียกว่า "**ผู้อนุญาตให้ใช้สิทธิ**" ฝ่ายหนึ่ง และ

มหาวิทยาลัยกรุงเทพ ตั้งอยู่เลขที่ 119 ถนนพระราม 4 แขวงพระโขนง เขตคลองเตย กรุงเทพมหานคร 10110 ซึ่งต่อไปนี้เรียกว่า "**ผู้ได้รับอนุญาตให้ใช้สิทธิ**" อีกฝ่ายหนึ่ง

ผู้อนุญาตให้ใช้สิทธิ และ ผู้ได้รับอนุญาตให้ใช้สิทธิ ตกลงทำสัญญากันโดยมีข้อความดังต่อไปนี้

ข้อ 1. ผู้อนุญาตให้ใช้สิทธิขอรับรองว่าเป็นผู้สร้างสรรค์และเป็นผู้มีสิทธิแต่เพียงผู้เดียวในงานสารนิพนธ์/วิทยานิพนธ์หัวข้อ การประยุกต์ใช้สารประกอบแบบ ซอฟต์ แบบไม่สมมาตร

ซึ่งถือเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร วิทยาศาสตรมหาบัณฑิต ของมหาวิทยาลัยกรุงเทพ (ต่อไปนี้เรียกว่า "**สารนิพนธ์/วิทยานิพนธ์**")

ข้อ 2. ผู้อนุญาตให้ใช้สิทธิตกลงยินยอมให้ผู้ได้รับอนุญาตให้ใช้สิทธิโดยปราศจากค่าตอบแทนและไม่มีกำหนดระยะเวลาในการนำสารนิพนธ์/วิทยานิพนธ์ ซึ่งรวมถึงแต่ไม่จำกัดเพียงการทำซ้ำ ดัดแปลง เผยแพร่ต่อสาธารณชน ให้เช่าต้นฉบับหรือสำเนางาน ให้ประโยชน์อันเกิดจากลิขสิทธิ์แก่ผู้อื่น อนุญาตให้ผู้อื่นใช้สิทธิโดยจะกำหนดเงื่อนไขอย่างหนึ่งอย่างใดด้วยหรือไม่ก็ได้ ไม่ว่าทั้งหมดหรือเพียงบางส่วน หรือการกระทำอื่นใดในลักษณะทำนองเดียวกัน

ข้อ 3. หากกรณีมีข้อขัดแย้งในปัญหาลิขสิทธิ์ในสารนิพนธ์/วิทยานิพนธ์ระหว่างผู้อนุญาตให้ใช้สิทธิกับบุคคลภายนอกก็ดี หรือระหว่างผู้ได้รับอนุญาตให้ใช้สิทธิกับบุคคลภายนอกก็ดี หรือมีเหตุขัดข้องอื่นๆ เกี่ยวกับลิขสิทธิ์ อันเป็นเหตุให้ผู้รับอนุญาตให้ใช้สิทธิไม่สามารถนำงานนั้นออกทำซ้ำ เผยแพร่ หรือโฆษณาได้ ผู้อนุญาตให้ใช้สิทธิยินยอมรับผิดและชดใช้ค่าเสียหายแก่ผู้ได้รับอนุญาตให้ใช้สิทธิในความเสียหายต่างๆ ที่เกิดขึ้นแก่ผู้ได้รับอนุญาตให้ใช้สิทธิทั้งสิ้น

สัญญานี้ทำขึ้นสองฉบับ มีข้อความเป็นอย่างเดียวกัน คู่สัญญาได้อ่านและเข้าใจข้อความในสัญญานี้โดยละเอียดแล้ว จึงได้ลงลายมือชื่อให้ไว้เป็นสำคัญต่อหน้าพยาน และเก็บรักษาไว้ฝ่ายละฉบับ

ลงชื่อ.....ผู้อนุญาตให้ใช้สิทธิ

(.....)

ลงชื่อ.....ผู้ได้รับอนุญาตให้ใช้สิทธิ

(.....)

ผู้อำนวยการสำนักหอสมุด

ลงชื่อ.....พยาน

(ผู้ช่วยศาสตราจารย์ ดร. ศิวพร หวังพิพัฒน์วงศ์)

คณบดีบัณฑิตวิทยาลัย

ลงชื่อ.....พยาน

(.....)

บัณฑิตวิทยาลัย มหาวิทยาลัยกรุงเทพ

อนุมัติให้การศึกษาเฉพาะบุคคลนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

วิทยาศาสตรมหาบัณฑิต

เรื่อง การประยุกต์ใช้เสถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น

ผู้วิจัย นาย วสันต์ มากธนะรุ่ง

ได้พิจารณาเห็นชอบโดย

อาจารย์ที่ปรึกษา [redacted]

(ดร. ธนกร หวังพิพัฒน์วงศ์)

กรรมการผู้ทรงคุณวุฒิ [redacted]

(ดร. ธิรพล วงศ์สอาดสกุล)

[redacted]

(ผศ. ดร. สิวพร หวังพิพัฒน์วงศ์)

คณบดีบัณฑิตวิทยาลัย

วันที่ 1 เดือน กรกฎาคม พ.ศ. 2554

วสันต์ มากธนะรุ่ง. ปริญญาวิทยาศาสตรมหาบัณฑิต, กรกฎาคม 2554, บัณฑิตวิทยาลัย มหาวิทยาลัยกรุงเทพ.
การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น (64หน้า)
อาจารย์ที่ปรึกษา: ดร.ธนกร หวังพิพัฒน์วงศ์

บทคัดย่อ

ในการศึกษาครั้งนี้เป็นการทดสอบการออกแบบและพัฒนาซอฟต์แวร์ระบบลงทะเบียนเรียนโดยใช้สถาปัตยกรรมแบบลำดับชั้น ผลการทดสอบพบว่า เมื่อมีการเปลี่ยนแปลงเงื่อนไขหรือความต้องการทางธุรกิจสามารถแก้ไขระบบทุกส่วนที่ได้รับผลกระทบได้ง่ายขึ้น เนื่องจากระบบมีความยืดหยุ่นและแบ่งหน้าที่การทำงานแต่ละส่วนอย่างชัดเจน รวมถึงสามารถนำส่วนประกอบที่ได้พัฒนากลับมาใช้ใหม่ได้

Makthanarung, Wasan. Master of Science in Information Technology and Management, July 2011, Graduate School, Bangkok University.

Software Application Design Using Layered Architecture (64 pp.)

Advisor: Thanakorn Wangpipatwong ,Ph.D.

ABSTRACT

In this study, the design and development of an online registration system using layer architecture is conducted. The results show that layer architecture can help developer easily change the system when business requirements have changed. Since, each component of the system is defined clearly, it makes the system more flexible and easier to reuse.

กิตติกรรมประกาศ

โครงการศึกษาเฉพาะบุคคล เรื่อง การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น ได้ดำเนินการจนสำเร็จลุล่วง ด้วยความกรุณาจาก ดร.ธนกร หวังพิพัฒน์วงศ์ อาจารย์ที่ปรึกษาโครงการ อาจารย์เป็นผู้มอบความรู้คำแนะนำ และการชี้ทางไปสู่การพัฒนาตนเองให้พร้อมศึกษาหาความรู้อยู่เสมอ การศึกษาและการวิจัยนี้จะเกิดขึ้นไม่ได้ หากไม่ได้รับความไว้วางใจจากอาจารย์ให้ดำเนินการ

ขอขอบคุณ ดร. ถิรพล วงศ์สอาดสกุล กรรมการผู้ทรงคุณวุฒิ ที่ได้ให้ข้อซักถาม แนะนำที่เป็นประโยชน์ต่อการศึกษาโครงการ ขอบคุณ อาจารย์ สราวุธิ ราษฎร์นิยม ผู้ให้คำปรึกษา ข้อมูลและคำแนะนำมาโดยตลอด

ขอขอบคุณคณะผู้บริหารมหาวิทยาลัยกรุงเทพ ที่ได้อนุมัติทุนการศึกษาให้ผู้วิจัยได้ศึกษาในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศและการจัดการ

ขอขอบคุณ บิดามารดา ที่ให้การสนับสนุนและเป็นกำลังใจเสมอมา รวมถึงอาจารย์ทุกท่านที่ได้ถ่ายทอดวิชาความรู้ ทั้งในด้านการศึกษาและการดำเนินชีวิต

สุดท้ายนี้ หากผลการศึกษาวิจัยนี้เกิดประโยชน์กับบุคคลที่ได้นำไปใช้งานก็ดี ศึกษาก็ดี หรือเกิดคุณใด ๆ ก็ดี ขอให้คุณงามความดีนั้นสรรเสริญไปยังผู้มีพระคุณของข้าพเจ้า ทว่าหากผลงานการศึกษาชิ้นนี้ผิดพลาด มีข้อบกพร่องประการใด ล้วนแต่เกิดขึ้นด้วยความอ่อนด้อยของข้าพเจ้า ซึ่งเป็นผู้จัดทำทั้งสิ้น ข้าพเจ้าต้องขออภัยมา ณ ที่นี้

วสันต์ มากธนะรุ่ง

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญภาพ

สารบัญภาพ (ต่อ)

สารบัญภาพ (ต่อ)

บทที่ 1
บทนำ

เอกสารฉบับนี้ เป็นการนำเสนอข้อมูลที่ได้รับในการศึกษาค้นคว้า รวบรวมข้อมูล ทดลอง และปฏิบัติจนได้ผลลัพธ์ตามโครงการศึกษาเฉพาะบุคคล 1 และ 2 ซึ่งเป็นส่วนหนึ่งในการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาเทคโนลียีสารสนเทศและการจัดการ มหาวิทยาลัยกรุงเทพ

การศึกษาครั้งนี้ เป็นการศึกษาในเรื่องของการวิจัย โดยมีหัวเรื่องว่า “การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น (Software Application Design using Layered Architecture)”

1.1 ความสำคัญและที่มาของปัญหา

ในปัจจุบัน องค์กรธุรกิจมีการแข่งขันสูง ทำให้ความต้องการทางธุรกิจต้องปรับเปลี่ยนให้สอดคล้องกับกลยุทธ์หลักขององค์กรอยู่เสมอ ส่งผลให้ซอฟต์แวร์ที่ถูกพัฒนาโดยใช้วิธีแบบดั้งเดิมไม่สามารถรองรับการเปลี่ยนแปลงที่เกิดขึ้นได้ทัน เนื่องจากผู้ปรับปรุงซอฟต์แวร์จะต้องมีความเข้าใจในระบบงานเดิมอย่างดี และสามารถแก้ไขซอฟต์แวร์ในทุกส่วนที่ถูกกระทบจากการเปลี่ยนแปลงทางธุรกิจนั้นๆ ซึ่งใช้ระยะเวลานาน

สาเหตุที่การปรับปรุงแก้ไขซอฟต์แวร์ที่ถูกพัฒนาขึ้นมานั้นต้องใช้เวลานาน เนื่องจากซอฟต์แวร์ที่ถูกพัฒนาด้วยวิธีแบบดั้งเดิมนั้นมักจะมีโครงสร้างการทำงานที่ซับซ้อนและความสัมพันธ์ของข้อมูลจำนวนมากโดยเฉพาะอย่างยิ่งองค์กรขนาดใหญ่ที่มีการแบ่งระบบงานภายในองค์กร ออกเป็นส่วนย่อยๆ ตามหน้าที่ในแต่ละส่วนงาน ซึ่งโดยทั่วไปจะมีความสัมพันธ์เชื่อมโยงกันอย่างแนบแน่นและส่งผลให้การเปลี่ยนแปลงเงื่อนไขหรือความต้องการทางธุรกิจภายในแต่ละส่วน จะเกิดผลกระทบกับงานในส่วนอื่นๆทำให้ผู้ออกแบบและพัฒนาต้องคอยตามแก้ระบบงานที่เกี่ยวข้องทั้งหมดให้ตรงกับความต้องการทางธุรกิจที่เปลี่ยนแปลงไป (Meier, 2008; Microsoft Patterns & Practices Team, 2009, p. 23; Wiggerts, 1997)

แนวทางหนึ่งที่สามารถแก้ปัญหาดังกล่าวได้คือการออกแบบและพัฒนาซอฟต์แวร์แบบลำดับชั้น โดยซอฟต์แวร์ที่ออกแบบและพัฒนานั้นมีลักษณะแบ่งหน้าที่การทำงานเป็นชั้นๆ ซึ่งแต่ละชั้นจะทำหน้าที่สมบูรณ์ในตัวเอง (เฉลิมพล, 2553; Microsoft Pattern & Practices Team, 2009,

p. 39-40) และสามารถนำมาประกอบเป็นซอฟต์แวร์ขนาดใหญ่ได้ ซึ่งส่งผลดีให้การพัฒนาซอฟต์แวร์ขนาดใหญ่นั้นทำได้ง่าย เนื่องจากมีโครงสร้างที่ง่ายและชัดเจน อีกทั้งส่วนประกอบที่พัฒนาขึ้นในแต่ละส่วนนั้น สามารถนำกลับมาใช้ใหม่ได้

การศึกษานี้เป็นการนำแนวความคิดการออกแบบและพัฒนาซอฟต์แวร์แบบลำดับชั้น มาช่วยในการออกแบบและพัฒนาระบบลงทะเบียนเรียนแบบรายวิชา โดยมีสมมติฐานว่าการออกแบบและพัฒนาซอฟต์แวร์โดยใช้สถาปัตยกรรมแบบลำดับชั้น จะช่วยทำให้ระบบลงทะเบียนเรียนแบบรายวิชา

มีความยืดหยุ่นมากขึ้นสามารถแก้ไขและปรับปรุงซอฟต์แวร์รองรับความต้องการทางธุรกิจที่มีการเปลี่ยนแปลงได้ง่ายกว่าระบบลงทะเบียนที่ได้ออกแบบและพัฒนาด้วยสถาปัตยกรรมแบบดั้งเดิม

1.2 วัตถุประสงค์ของการศึกษา

โครงการศึกษาเฉพาะบุคคลในหัวข้อเรื่อง “การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น” มุ่งจัดทำเพื่อวัตถุประสงค์ดังต่อไปนี้

1.2.1 ระบบที่ออกแบบและพัฒนาโดยการใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น สามารถรองรับการเพิ่มเติม แก้ไข เปลี่ยนแปลง และการดูแลรักษาระบบได้ง่ายขึ้น

1.2.2 เพื่อให้การพัฒนาระบบงานใหม่ สามารถนำโมดูลที่พัฒนาแล้วสามารถนำกลับมาใช้ใหม่ได้

1.3 ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ที่คาดว่าจะได้รับจากการดำเนินโครงการศึกษาเฉพาะบุคคลในหัวข้อเรื่อง “การประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น” มีดังนี้

1.3.1 สามารถนำโมดูลที่พัฒนาไปแล้วสามารถนำกลับมาใช้ใหม่ได้

1.3.2 สามารถตรวจสอบการทำงานของระบบได้ง่ายขึ้นเนื่องจากมีการแยกชั้นกันอย่างชัดเจน

1.3.3 ง่ายต่อการแก้ไขและปรับปรุงระบบ

1.4 ขอบเขตการศึกษา

ในการศึกษานี้เป็นการนำระบบลงทะเบียนเรียนแบบรายวิชามหาวิทยาลัยกรุงเทพมาทำ

การทดสอบ ภายใต้แนวคิดการออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้นว่าสามารถแก้ไขปัญหาดังที่กล่าวมาข้างต้นได้

1.5 ขั้นตอนการพัฒนา

ในการศึกษาโครงการนี้จะทำการศึกษาโครงสร้างการทำงานเดิมของระบบที่ใช้การออกแบบแบบดั้งเดิม เพื่อมาพิจารณาออกแบบระบบโดยการประยุกต์ใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับขั้น โดยมีการแบ่งการทำงานต่างๆ ดังนี้

1.5.1 ศึกษาแนวทางการออกแบบซอฟต์แวร์โดยใช้สถาปัตยกรรมแบบลำดับชั้น

1.5.2 ศึกษาโครงสร้างและการทำงานของระบบเดิม

1.5.3 วางแผนการทำงานของระบบใหม่โดยใช้สถาปัตยกรรมแบบลำดับชั้น

1.5.4 ขั้นตอนการวิเคราะห์การทำงานของระบบ

1.5.5 ขั้นตอนการออกแบบระบบ

1.5.6 ขั้นตอนการพัฒนาระบบ

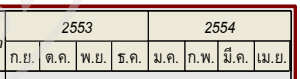
1.5.7 ขั้นตอนการวัดและการประเมินผลเปรียบเทียบว่าระบบที่พัฒนาขึ้นมานั้นสามารถ
รองรับการทำงานที่เพิ่มขึ้นและการเปลี่ยนแปลงแก้ไข เมื่อเปรียบเทียบกับระบบที่ไม่ได้ออกแบบโดยใช้สถาปัตยกรรมแบบลำดับชั้น

1.6 ระยะเวลาในการพัฒนางาน

การวิจัยนี้ใช้เวลา 9 เดือน นับตั้งแต่เดือนกันยายน พ.ศ. 2553 ถึงเดือนเมษายน พ.ศ. 2554 โดยมีแผนงานดังตารางที่ 1

ตารางที่ 1 : แผนการวิจัย

ID	Task Name	Start	Finish	Duration
1	ศึกษาแนวทางการออกแบบซอฟต์แวร์โดยใช้สถาปัตยกรรมแบบลำดับชั้น	1/9/2553	30/9/2553	30d
2	ศึกษาโครงสร้างและการทำงานของระบบเดิม	15/9/2553	14/10/2553	30d
3	วางแผนการทำงานของระบบใหม่โดยใช้สถาปัตยกรรมลำดับชั้น	15/10/2553	24/10/2553	10d
4	ขั้นตอนวิเคราะห์การทำงานของระบบ	**25/10/2553**	**19/12/2553**	56**d**
5	เก็บข้อมูลจากผู้ใช้ระบบ	25/10/2553	18/11/2553	25d
6	วิเคราะห์ข้อมูลของระบบ	9/11/2553	28/11/2553	20d
7	การพัฒนาเค้าโครงละเอียดของระบบ	20/11/2553	19/12/2553	30d
8	ขั้นตอนการออกแบบและพัฒนาระบบ	**14/12/2553**	**3/4/2554**	**111d**
9	ออกแบบระบบ	14/12/2553	22/1/2554	40d
10	ออกแบบส่วนต่อประสาน	1/1/2554	19/2/2554	50d
11	สร้างระบบต้นแบบพร้อมส่วนผู้ใช้งาน	20/1/2554	10/3/2554	50d
12	พัฒนาส่วนประกอบต่าง ๆ ของระบบ	3/2/2554	3/4/2554	60d
13	ขั้นตอนการทดสอบระบบ	**4/4/2554**	**23/4/2554**	**20d**
14	ทดสอบระบบ	4/4/2554	8/4/2554	5d
15	ติดตั้งระบบ	9/4/2554	13/4/2554	5d
16	ปรับปรุงแก้ไขระบบหลังจากพัฒนาเสร็จเรียบร้อย	14/4/2554	23/4/2554	10d
17	การประเมินผลและสรุปผลหลังจากใช้สถาปัตยกรรมแบบลำดับชั้น	24/4/2554	30/4/2554	7d



1.7 สมมติฐานการศึกษาและกรอบแนวคิดการวิจัย

การออกแบบและพัฒนาระบบลงทะเบียนเรียนแบบรายวิชามหาวิทยาลัยกรุงเทพ โดยใช้สถาปัตยกรรมแบบลำดับชั้น มีสมมติฐานดังต่อไปนี้

1.7.1 การออกแบบและพัฒนาระบบด้วยแนวคิดแบบลำดับชั้น สามารถตอบสนองการทำงานในปัจจุบันได้ดีกว่าระบบเดิม

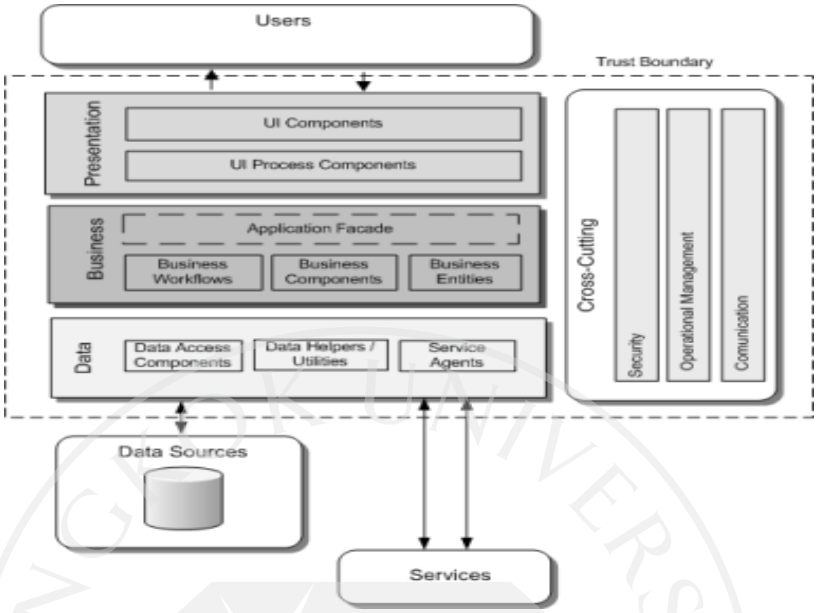
1.7.2 การออกแบบและพัฒนาระบบด้วยแนวคิดแบบลำดับชั้น ช่วยให้ระบบรองรับการเพิ่มเติมการทำงานหรือการปรับเปลี่ยนความต้องการของระบบได้เร็วกว่าระบบเดิม

บทที่ 2
การทบทวนวรรณกรรม

2.1 สถาปัตยกรรมแบบลำดับชั้น (Layered Architecture)

การออกแบบซอฟต์แวร์โดยใช้สถาปัตยกรรมลำดับชั้น จะมุ่งเน้นที่การออกแบบด้วยวิธีการจัดกลุ่มของงานที่มีความสัมพันธ์กันและงานประเภทเดียวกันเอาไว้ที่ชั้นเดียวกัน โดยจะแบ่งแต่ละลำดับชั้นให้ชัดเจนตามหน้าที่ของงานเพื่อให้แต่ละลำดับชั้นนั้นทำหน้าที่ชัดเจนและมีหน้าที่รับผิดชอบเฉพาะอย่างให้เกิดความยืดหยุ่นเพื่อรองรับความเปลี่ยนแปลงที่เกิดขึ้นและง่ายต่อการแก้ไขปรับปรุงดูแลรักษาระบบที่จะเกิดขึ้นในอนาคต โครงสร้างซอฟต์แวร์โดยใช้สถาปัตยกรรมแบบลำดับชั้นจะจัดกลุ่มและประเภทของงานที่เกี่ยวข้องกันไว้ในชั้นเดียวกัน (Kilian, Mühsig & Guhr, 2010; Arking & Millett, 2009, p.169-172) ดังภาพที่ 1 โดยแต่ละชั้นสามารถจำแนกเป็น 3 ส่วนหลัก ๆ คือ

ภาพที่ 1: โครงสร้างสถาปัตยกรรมแบบลำดับชั้น



ที่มา: Meier, J.D. (2008). Application architecture Visios now available. Retrieved by 10 December 2010 from http://blogs.msdn.com/b/jmeier/archive/2008/11/24/application-architecture-diagrams-added-to-codeplex.aspx

2.1.1 พรีเซนท์เทชันเลเยอร์ (Presentation layer) เป็นชั้นที่ใช้ติดต่อระหว่างผู้ใช้งานโดยประกอบไปด้วยส่วนของการแสดงผลและส่วนของการใช้งาน ในระดับชั้นนี้จะไม่มีโค้ดที่เกี่ยวกับฐานข้อมูลหรือภาษาเอสคิวแอล มีแต่การเรียกใช้ออบเจ็คและคลาสจากบิสซิเนสเลเยอร์ พรีเซนท์เทชันเลเยอร์ประกอบด้วย 2 ส่วนหลัก ๆ คือ (Meier, et al., 2009)

1) ยูสเซอร์อินเทอร์เฟสคอมโพเนนท์ (User Interface Component) เป็นส่วนที่ใช้แสดงผลข้อมูล และรับข้อมูลจากผู้ใช้งาน

2) พรีเซนท์เทชันลอจิกคอมโพเนนท์ (Presentation Logic components) เป็นส่วนที่กำหนดรูปแบบการติดต่อระหว่างผู้ใช้งานกับแอพพลิเคชัน

2.1.2 บิสซิเนสเลเยอร์ (Business layer) เป็นชั้นที่ทำหน้าที่เกี่ยวกับความต้องการของระบบ กฎเกณฑ์ดำเนินแผนงานทางธุรกิจอีกทั้งยังเป็นตัวกำหนดแนวทางหรือทิศทางในการทำงานของแอพพลิเคชันโดยอ้างอิงจากความต้องการทางธุรกิจเป็นหลักและผลลัพธ์ที่ได้ต้องสอดคล้องถูกต้องตามแผนธุรกิจที่วางไว้ ลำดับชั้นนี้ผู้ออกแบบและพัฒนาระบบต้องมีความเข้าใจในระบบ

มาก ในการออกแบบและพัฒนาระบบให้เกิดความยืดหยุ่น เพื่อรองรับการเปลี่ยนแปลงความต้องการทางธุรกิจที่เกิดขึ้นได้ตลอดเวลา บิสซิเนส เลเยอร์มีองค์ประกอบหลัก ๆ ดังนี้ (Meier, et al., 2009)

1) แอพพลิเคชันฟาซาด (Application façade) ทำหน้าที่รวบรวมการทำงานที่เกี่ยวข้องกับความต้องการทางธุรกิจหลายๆ การทำงานให้เป็นการทำงานเดียว เนื่องจากผู้ใช้งานภายนอกไม่จำเป็นต้องเข้าใจรายละเอียดและความสัมพันธ์นั้นๆ

2) ตรรกะทางธุรกิจ (Business Logic Components) เป็นการกำหนดแนวทางหรือทิศทางในการทำงานของแอพพลิเคชันโดยการอ้างอิงจากกระบวนการทำงานหรือแผนธุรกิจเป็นหลัก และผลลัพธ์ของข้อมูลที่ได้ต้องสอดคล้องและถูกต้องตามแผนธุรกิจที่วางไว้ ตรรกะทางธุรกิจสามารถแบ่งย่อยได้อีกเป็น 2 ประเภทดังนี้

2.1) ลำดับการทำงานทางธุรกิจ (Business Workflow Components) กล่าวคือ แอพพลิเคชันสามารถใช้ข้อมูลที่ได้จากการเก็บรวบรวมข้อมูลมาจากส่วนติดต่อผู้ใช้งาน โดยข้อมูลนั้นได้ถูกส่งไปยังบิสซิเนสเลเยอร์แล้ว ไปประมวลผลตามกระบวนการทางธุรกิจและต้องดำเนินงานในลำดับที่ถูกต้อง พร้อมทั้งต้องคอยประสานงานกับการทำงานส่วนอื่นๆ ด้วย

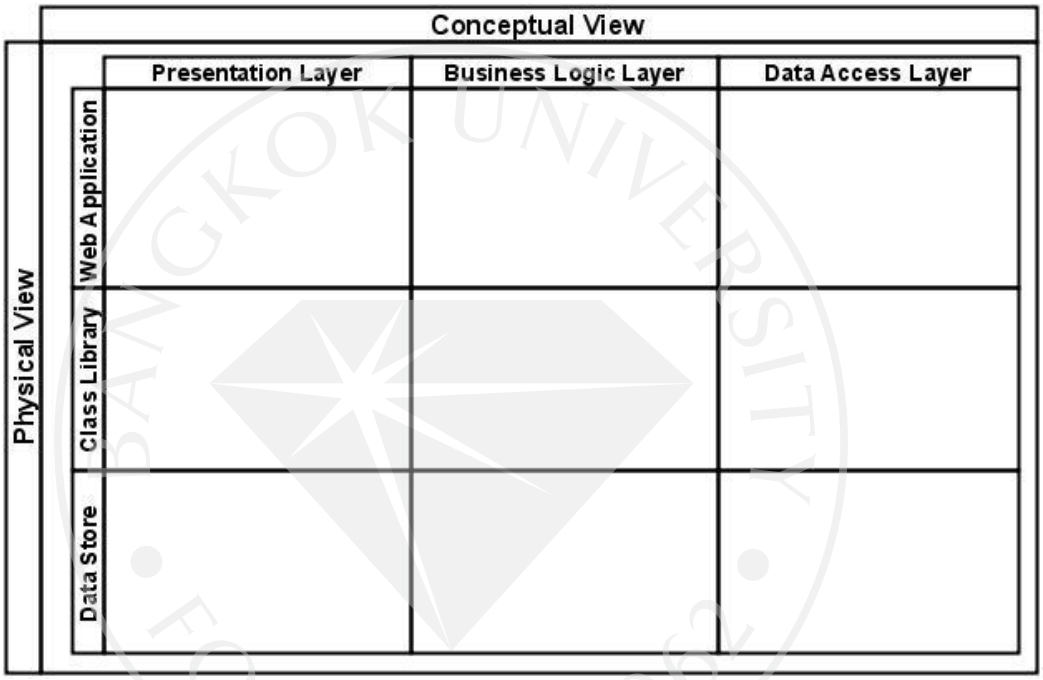
2.2) เอนทิตี้ทางธุรกิจ (Business Entity Components) จะเป็นส่วนที่ใช้ปกปิดข้อมูลตรรกะทางธุรกิจ และข้อมูลที่จำเป็นในการทำงานจริงๆ เช่น ข้อมูลลูกค้า ข้อมูลการสั่งซื้อ ข้อมูล
การขาย เป็นต้น ข้อมูลเหล่านี้จะเข้าถึงได้โดยผ่านทางคุณสมบัติของเอนทิตี้เท่านั้น โดยผ่านทาง Properties และภายในเอนทิตี้จะทำหน้าที่ตรวจสอบความถูกต้องของข้อมูล เช่น ประเภท และขนาดของข้อมูล เพื่อรับรองว่าข้อมูลที่นำไปใช้มีความถูกต้อง

2.1.3 ดาต้าแอคเซสเลเยอร์ (Data Access Layer) เป็นชั้นที่ติดต่อ และการเข้าถึงในส่วนของข้อมูลจะประกอบไปด้วยคลาสที่ทำหน้าที่เป็น ตัวแทน ของ DBMS ที่ใช้จริงๆ เช่นชั้นบิสซิเนสเลเยอร์อาจจะมาขอให้คลาสในชั้นดาต้าแอคเซสเลเยอร์ส่งข้อมูลออบเจ็กต์ของลูกค้าทั้งหมดมาให้ หรือเอาออบเจ็กต์ รายการสั่งซื้อไปบันทึกเก็บไว้ โดยการเข้าถึงข้อมูลสามารถผ่านเข้าถึงข้อมูลได้โดยผ่านเซอร์วิส ในชั้นดาต้าแอคเซสเลเยอร์ จะใช้อินเทอร์เฟสที่เป็นมาตรฐาน ซึ่งทำให้ชั้นบิสซิเนสเลเยอร์สามารถเรียกใช้งานได้ผ่านทางอินเทอร์เฟส ดาต้าแอคเซสเลเยอร์ประกอบด้วยองค์ประกอบหลัก ดังนี้(Meier, et al., 2009)

1) การเข้าถึงข้อมูล (Data Access Components) ในองค์ประกอบนี้จะรวมการเชื่อมต่อกับฐานข้อมูลไว้ที่ส่วนกลาง เพื่อง่ายต่อการตั้งค่า และการดูแลรักษาระบบ อีกทั้งสามารถนำส่วนประกอบที่พัฒนาไปแล้วนำกลับมาใช้ใหม่ได้อย่างง่ายในระบบอื่น ๆ ด้วย

2) เซอร์วิสเอเจนท์ (Service Agents) จะทำหน้าที่เป็นตัวเชื่อมต่อระหว่างดาต้าเลเยอร์กับบิสซิเนสเลเยอร์ ในกรณีที่บิซิเนสเลเยอร์ต้องการใช้บริการ ผู้ออกแบบและนักพัฒนาระบบต้องพัฒนาส่วนของโค้ดเพื่อจัดการในส่วนของการเชื่อมต่อกับระหว่างเลเยอร์ โดยเซอร์วิสเอเจนท์จะจำแนกลักษณะที่เหมาะกับระบบงานนั้นๆ และยังคอยติดต่อประสานงานกับรูปแบบข้อมูลที่ให้บริการกับรูปแบบข้อมูลที่ระบบร้องขอมาอีกด้วย

ภาพที่ 2: แอคทิวิตี้ไดอะแกรมแสดงการออกแบบและพัฒนาระบบด้วยแนวคิดลำดับชั้น



จากรูปที่ 2 แสดงให้เห็น มุมมองในการออกแบบและพัฒนาระบบโดยแบ่งเป็น 2 มุมมอง ดังนี้

1. มุมมองทางกายภาพ (Physical View) คือ เว็บแอพพลิชันที่ถูกพัฒนาขึ้นภายใต้การออกแบบด้วยเทคโนโลยีเชิงวัตถุและมีการแบ่งประเภทของงานต่างๆ ออกเป็น คลาสและการเก็บข้อมูลแบบคงทนถาวรซึ่งถูกจัดเก็บลงแหล่งข้อมูล (Dutta & Krishnamoorthy, 2010) ประกอบไปด้วย

1.1 Data Store คือแหล่งข้อมูลสำหรับบันทึกและค้นคืนข้อมูลที่จำเป็นสำหรับระบบ

1.2 Class Library คือชุดคำสั่งของการทำงานในแต่ละขั้นตอนของระบบ

1.3 Web Application คือเว็บเพจที่เชื่อมโยงผู้ใช้งานให้สามารถทำงานกับส่วนคลาสไลบารี่และแหล่งข้อมูลได้

2. มุมมองทางมโนภาพ (Conceptual View)คือ แนวคิดในการออกแบบและพัฒนาระบบ โดยแบ่งระบบออกเป็นส่วนๆ โดยจัดแต่ละส่วนที่มีหน้าที่เหมือนกันไว้ด้วยกัน (Microsoft Corporation, 2003, p. 161-162) ประกอบไปด้วย

2.1 Presentation Layer ส่วนที่ใช้ติดต่อระหว่างผู้ใช้งานโดยประกอบไปด้วยส่วนของการ
แสดงผลและส่วนของการใช้งาน

2.2 Business Logic Layer ส่วนที่ทำหน้าที่เกี่ยวกับความต้องการของระบบ กฎเกณฑ์ดำเนินแผนงานทางธุรกิจอีกทั้งยังเป็นตัวกำหนดแนวทางหรือทิศทางในการทำงานของระบบ

2.3 Data Access Layer ส่วนที่ทำหน้าที่ติดต่อและการเข้าถึงในส่วนของข้อมูล

2.2 ดีไซน์แพทเทิร์น (Design Patterns)

ดีไซน์แพทเทิร์น เป็นแนวทางที่ใช้ในการแก้ไขปัญหาที่เกิดขึ้นในการออกแบบและพัฒนาซอฟต์แวร์ ซึ่งดีไซน์แพทเทิร์นเป็นการอธิบายวิธีการทำงานเพื่อนำไปประยุกต์ใช้ในเหตุการณ์ต่าง ๆ ในทางการเขียนโปรแกรมเชิงวัตถุ ดีไซน์แพทเทิร์นจะแสดงความสัมพันธ์ต่อกันระหว่างคลาสหรือ
อ็อบเจกต์ต่าง ๆ

ดีไซน์แพทเทิร์นจะช่วยทำให้กระบวนการพัฒนาระบบรวดเร็วขึ้น เพราะเป็นรูปแบบที่ผ่านการพิสูจน์และทดสอบมาแล้ว การออกแบบซอฟต์แวร์ที่ดีต้องเตรียมพร้อมกับปัญหาที่อาจขึ้นหลังจากที่ระบบได้ถูกใช้งานไปแล้ว การออกแบบโดยใช้ดีไซน์แพทเทิร์นจะช่วยป้องกันปัญหาที่อาจจะเกิดขึ้นจนลุกลามใหญ่โตได้ การจัดหมวดหมู่และแพทเทิร์น ตามที่ แกมมา, เฮล์ม และ วริสไซด์ (Gamma, Helm, Johnson, & Vlissides, 1994) ได้ทำการรวบรวมและอธิบายรูปแบบของดีไซน์แพทเทิร์นที่ได้รับการยอมรับว่าเป็นปัญหาทั่วไปที่มักพบอยู่เสมอในการเขียนโปรแกรมเพื่อพัฒนาระบบ

บทที่ 3
ขั้นตอนการศึกษา

ในบทนี้จะกล่าวถึงขั้นตอนการศึกษาของโครงการส่วนบุคคลนี้ ได้มีขั้นตอนในการดำเนินงานตามที่ได้กล่าวถึงในบทที่ผ่านมา ซึ่งมีรายละเอียดแตกต่างกันไปในแต่ละขั้นตอนดังนี้

3.1 ขั้นตอนการวางแผน

ในขั้นตอนการวางแผนนั้น จะเป็นการปฏิบัติในด้านวางแผน เตรียมงาน ค้นคว้าข้อมูลทั้งหมดที่จำเป็นต่อโครงการ ซึ่งมีรายละเอียดดังต่อไปนี้

3.1.1 เตรียมโครงร่างการศึกษา

ในขั้นตอนนี้จะเป็นการศึกษาข้อมูลจาก “คู่มือการศึกษา วิชาโครงการเฉพาะบุคคล (Independent Study Project)” เพื่อให้รู้ขั้นตอนในการจัดทำเอกสารต่างๆ ที่จะเป็นในการนำเสนอโครงการต่อคณะกรรมการ

3.1.2 ศึกษาทฤษฎีต่าง ๆ ที่เกี่ยวข้อง

ขั้นตอนนี้เป็นขั้นตอนการศึกษาเพื่อ(รายละเอียดได้กล่าวไว้แล้วในบทที่ 2 วรรณกรรมทบทวน) นำเทคนิคและแนวคิดการออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้นมาเปรียบเทียบกับระบบงานเดิมได้พบปัญหาว่าระบบงานเดิมถูกออกแบบและพัฒนาโดยใช้แนวคิดการทำงานพื้นฐานกับข้อมูลเป็นหลักสำคัญ เมื่อตรวจสอบแหล่งข้อมูลของระบบงานเดิมพบว่า ขั้นตอน
การทำงานบางอย่างถูกจัดเก็บอยู่ในระบบฐานข้อมูล ซึ่งส่งผลทำให้ขั้นตอนการตรวจสอบในการพัฒนาเป็นไปได้ยาก ดังนั้นในเอกสารฉบับนี้จึงนำเสนอสถาปัตยกรรมแบบลำดับชั้นเข้ามาจัดการกับปัญหาดังกล่าว โดยมีเป้าหมายเพื่อทำให้งานต่างๆ ที่เกิดขึ้น ถูกแบ่งแยกและจัดเก็บลงในส่วนที่สามารถตรวจสอบและแก้ไขได้โดยง่ายสำหรับการออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้น

3.2 ขั้นตอนการวิเคราะห์

เป็นส่วนของการรวบรวมข้อมูลต่างๆ ที่เกี่ยวข้อง และนำมาวิเคราะห์ เพื่อใช้ในการออกแบบและพัฒนาระบบกับแนวคิดการออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้น

3.2.1 ศึกษาขั้นตอนการทำงานของระบบ

ขั้นตอนนี้เป็นขั้นตอนการศึกษาการทำงานทุกขั้นตอน เพื่อนำมาวิเคราะห์การทำงานต่างของระบบ เพื่อนำไปออกแบบและพัฒนาระบบด้วยแนวคิดที่แบ่งแยกแต่ละส่วนให้มีความชัดเจนและทำงานภายใต้หน้าที่ใดหน้าที่หนึ่งเท่านั้น มีเป้าหมายทำให้การพัฒนาและการแก้ไขงานในระบบสามารถทำได้โดยง่ายและสะดวกรวดเร็ว จากที่กล่าวมาเอกสารฉบับนี้ได้นำเทคนิคและแนวคิด
การออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้นมาใช้เพื่อทดสอบระบบหลังจากที่ได้ออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้น

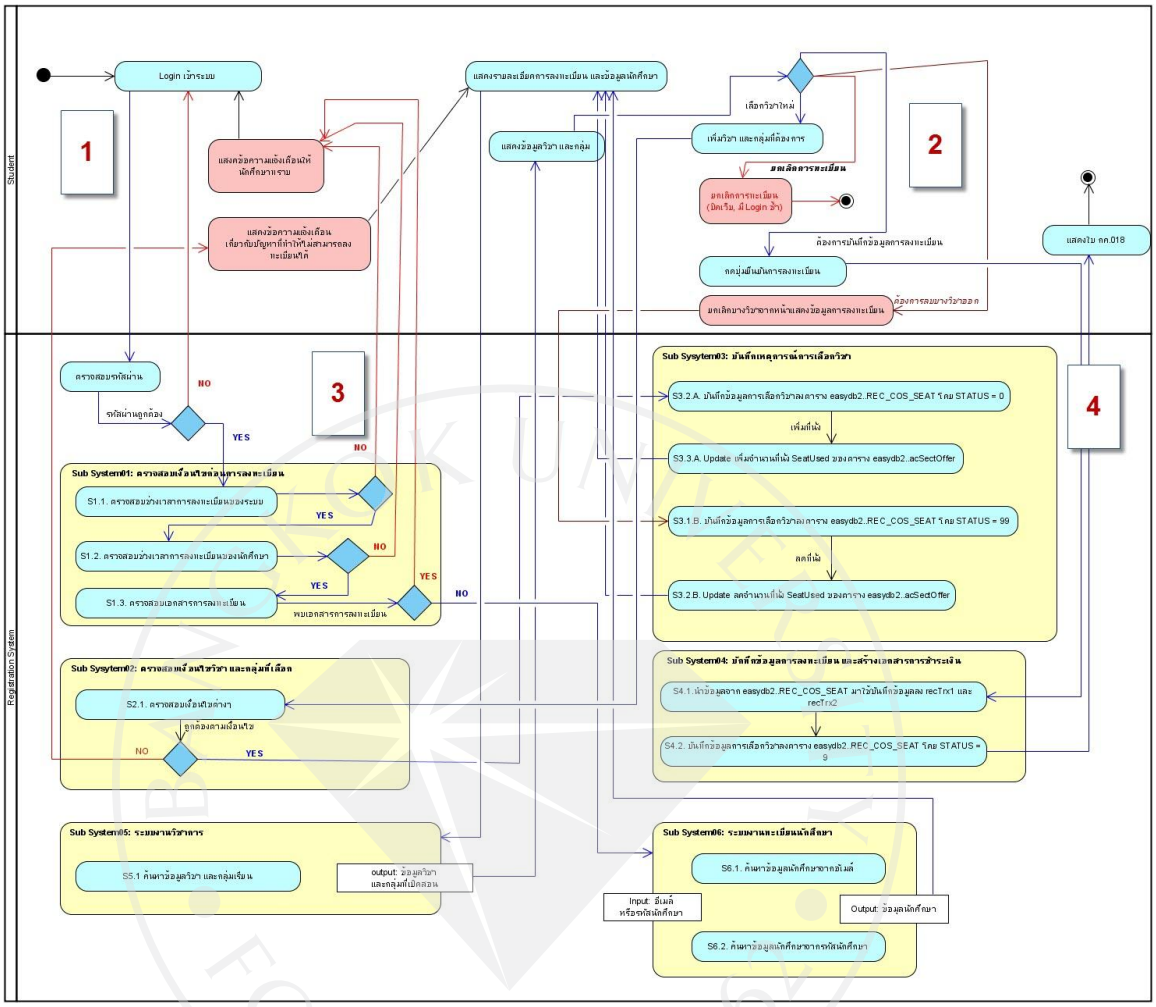
3.2.2 ขั้นตอนการออกแบบและพัฒนาระบบ

ในขั้นตอนการออกแบบและพัฒนาระบบโดยใช้แนวคิดการออกแบบและพัฒนาซอฟต์แวร์แบบลำดับชั้น ผู้ออกแบบและพัฒนาระบบต้องเข้าใจภาพรวมและความต้องการของระบบ รวมถึงรายละเอียดสิ่งต่างๆ ที่เกิดขึ้นในระบบและข้อมูลที่สำคัญต่อการทำงานของระบบ ซึ่งสามารถนำข้อมูลดังกล่าวมาใช้ในการแบ่งลำดับชั้นในการออกแบบซอฟต์แวร์ (Arking & Millett, 2009, p.194; Microsoft Patterns & Practices Team, 2009, p. 26) โดยมีลำดับการทำงานดังนี้

3.2.3 ศึกษาขั้นตอนการทำงานและจัดกลุ่มประเภทของการทำงาน

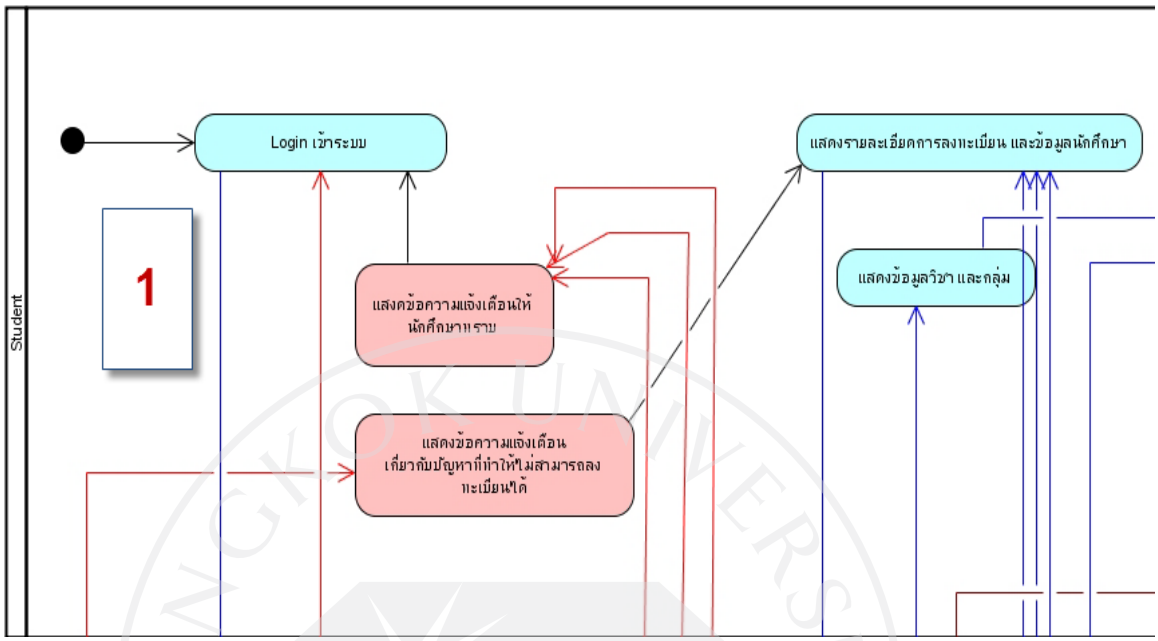
ดำเนินการศึกษาขั้นตอนการทำงานพร้อมทั้งจัดกลุ่มประเภทของการทำงานพร้อมทั้งศึกษาความสัมพันธ์ที่เชื่อมโยงในแต่ละกลุ่มตามประเภทของงาน มาเขียนลงแอคทิวิตี้ไดอะแกรม
ดังตัวอย่างในภาพที่ 3 โดยระบุด้วยว่างานแต่ละกลุ่มควรจะอยู่ในลำดับชั้นใด

ภาพที่ 3: แสดงแอคทิวิตี้ไดอะแกรมและขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษา

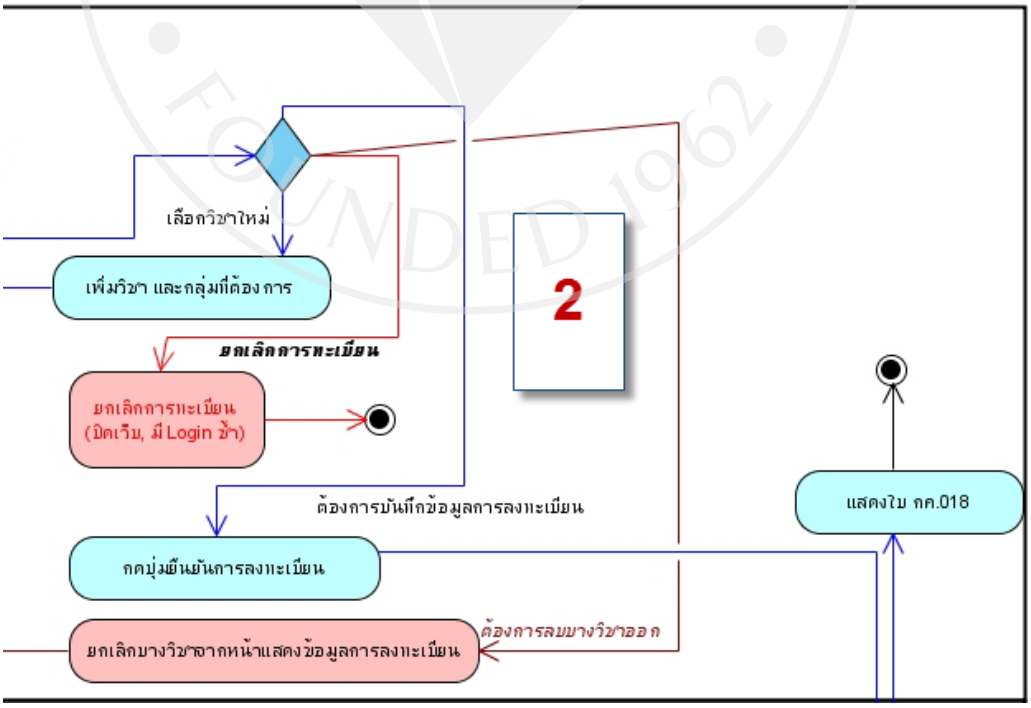


จากภาพที่ 3 ได้แบ่งออกเป็น 4 ส่วนดังภาพที่ 4 – 7 เพื่อแสดงแอคทิวิตี้ไดอะแกรมและขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาได้ชัดเจนยิ่งขึ้น

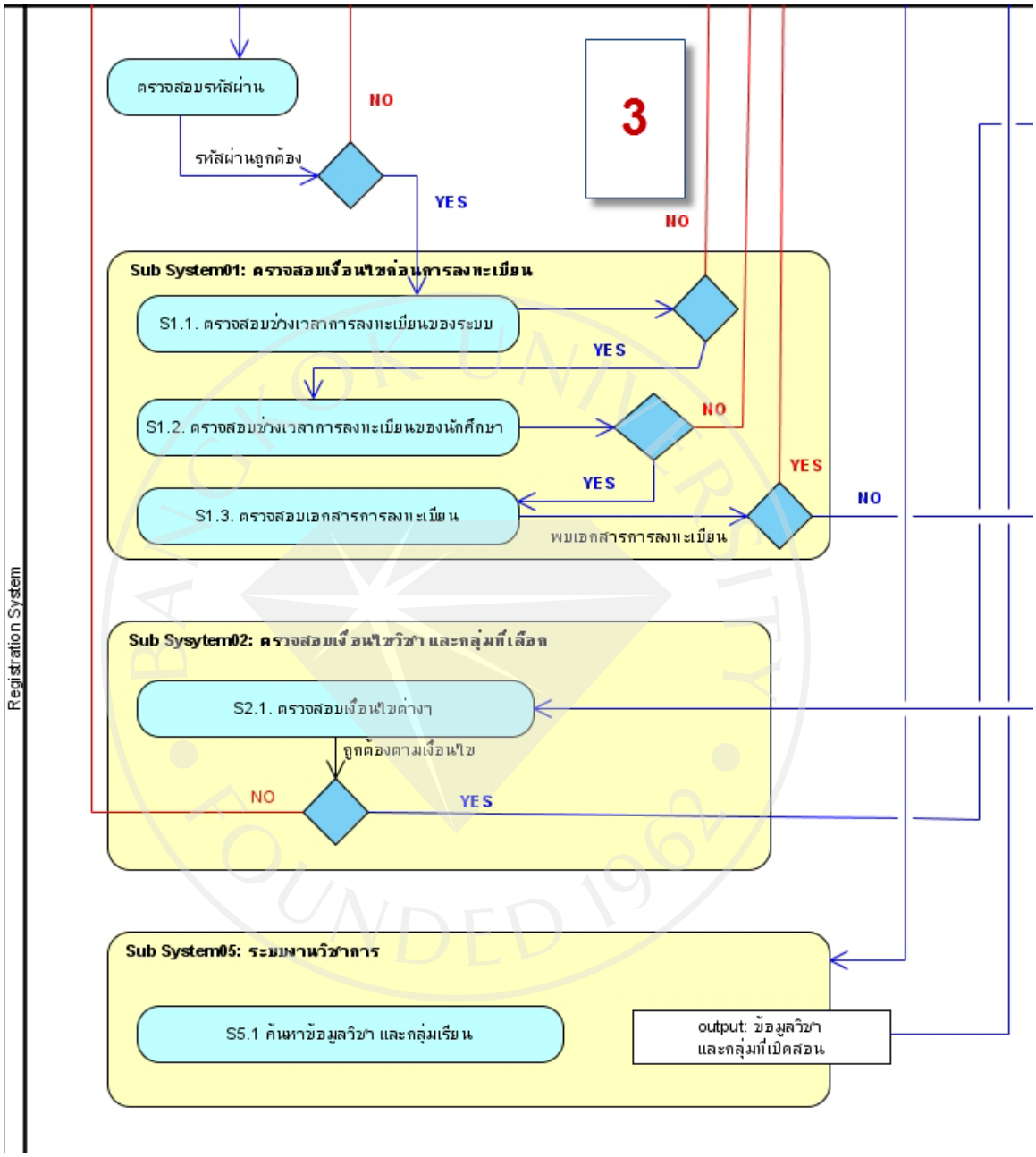
ภาพที่ 4: แสดงแอคทิวิตี้ไดอะแกรมและขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษา หมายเลข 1



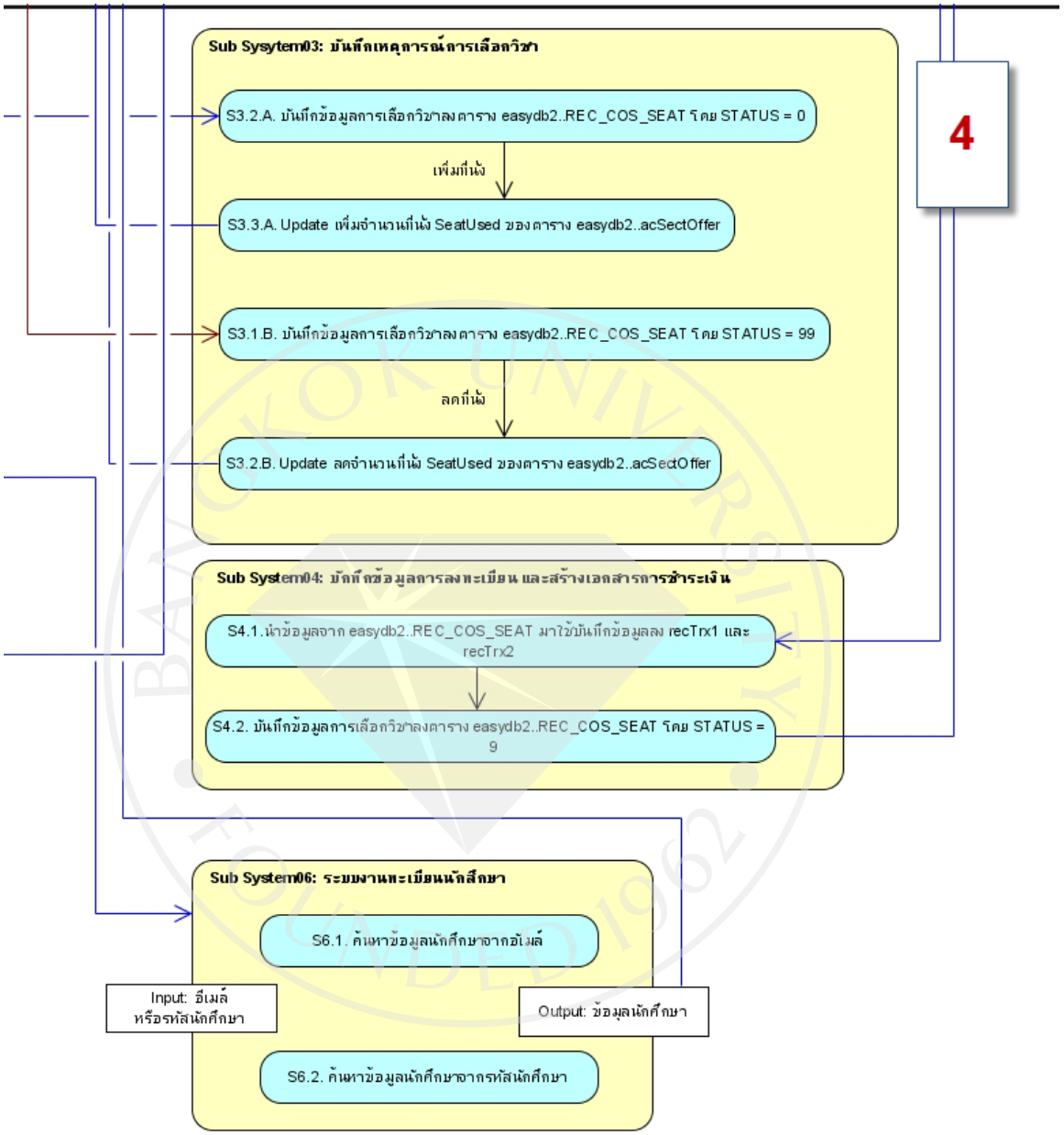
ภาพที่ 5: แสดงแอคทิวิตี้ไดอะแกรมและขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษา หมายเลข 2



ภาพที่ 6: แสดงแอคทิวิตี้ไดอะแกรมและขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาหมายเลข 3



ภาพที่ 7: แสดงแอคทิวิตี้ไดอะแกรมและขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษา หมายเลข 4



จากภาพที่ 3 แสดงให้เห็นขั้นตอนการทำงานของระบบลงทะเบียนเรียบแบบรายวิชาซึ่งมีขั้นตอนการทำงานดังต่อไปนี้

1. Login เข้าระบบ ทำงานต่อในข้อที่ 2
2. ตรวจสอบรหัสผ่าน ทำงานต่อในข้อที่ 3
3. ตรวจสอบเงื่อนไขก่อนการลงทะเบียน
   - 3.1 ตรวจสอบช่วงเวลาการลงทะเบียนของระบบ ทำงานต่อในข้อที่ 3.2
   - 3.2 ตรวจสอบช่วงเวลาการลงทะเบียนของนักศึกษา ทำงานต่อในข้อที่ 3.3
   - 3.3 ตรวจสอบเอกสารการลงทะเบียน กรณีที่ไม่พบเอกสาร ทำงานต่อในข้อที่ 14
   - 3.4 ตรวจสอบเงื่อนไขต่างๆ ทำงานต่อในข้อที่ 3.5
   - 3.5 กรณีที่เงื่อนไขถูกต้องทั้งหมด ทำงานต่อในข้อที่ 4
   - 3.6 กรณีที่เงื่อนไขไม่ถูกต้อง ทำงานต่อในข้อที่ 1
4. แสดงรายละเอียดการลงทะเบียนเรียน และข้อมูลนักศึกษา
   - 4.1 ทำงานในข้อที่ 15 เพื่อค้นคืนข้อมูลวิชาและกลุ่มของวิชาที่เปิดสอน
   - 4.2 กรณีที่เพิ่มวิชาใหม่ ทำงานต่อในข้อที่ 5
   - 4.3 กรณีที่ต้องการบันทึกข้อมูลการลงทะเบียน ทำงานต่อในข้อที่ 10
   - 4.4 กรณีที่ต้องการยกเลิกการลงทะเบียนเรียนบางวิชา ทำงานต่อในข้อที่ 7
   - 4.5 กรณีที่ปิดหน้าเว็บ หรือระบบไม่สามารถให้บริการต่อได้ ทำงานต่อในข้อที่ 9
5. เพิ่มวิชา และกลุ่มที่ต้องการ ทำงานต่อในข้อที่ 6
6. ตรวจสอบเงื่อนไขวิชา และกลุ่มที่เลือก
   - 6.1 ตรวจสอบเงื่อนไขต่างๆ เกี่ยวกับวิชาและกลุ่มวิชาที่ทำการลงทะเบียน ทำงานต่อในข้อที่ 6.2
   - 6.2 กรณีที่พบว่าไม่ตรงเงื่อนไขก็จะแสดงข้อความแจ้งเตือนให้นักศึกษาทราบ ทำงานต่อในข้อที่ 6.3
   - 6.3 กรณีที่พบว่าตรงตามเงื่อนไขที่กำหนด ทำงานต่อในข้อที่ 12
7. ยกเลิกบางวิชาจากหน้าแสดงข้อมูลการลงทะเบียน ทำงานต่อในข้อที่ 4
8. แสดงข้อความแจ้งเตือน เกี่ยวกับปัญหาที่ทำให้ไม่สามารถลงทะเบียนได้ ทำงานต่อในข้อ

ที่ 4

9. ยกเลิกการทะเบียน เช่น ปิดหน้าเว็บเพ็จ, มี Login เข้าใช้ระบบซ้ำ และจบการทำงาน
10. กดปุ่มยืนยันการลงทะเบียนเรียน ทำงานต่อในข้อที่ 13

11. แสดงใบ กค.018 และจบการทำงาน

12. บันทึกเหตุการณ์การเลือกวิชา

12.1 บันทึกข้อมูลการเลือกวิชา ทำงานต่อในข้อที่ 12.2

12.2 กรณีเพิ่มที่นั่ง: Update เพิ่มจำนวนที่นั่ง ทำงานต่อในข้อที่ 12.3

12.3 กรณีลดที่นั่ง: Update ลดจำนวนที่ ทำงานต่อในข้อที่ 12.4

12.4 เมื่อทำงานเสร็จครบถ้วนแล้ว ทำงานต่อในข้อที่ 11

13. บันทึกข้อมูลการลงทะเบียน และสร้างเอกสารการชำระเงิน

13.1 นำข้อมูลจากการลงทะเบียนมาใช้ในการบันทึกเอกสารการชำระเงิน ทำงานต่อ ใน

ข้อที่ 13.2

13.2 บันทึกข้อมูลการเลือกวิชา ทำงานต่อในข้อที่ 13.3

13.3 ทำงานต่อในข้อ 11

14. ระบบงานทะเบียนนักศึกษา

14.1 ค้นหาข้อมูลนักศึกษาจากอีเมลนักศึกษา หรือ จากรหัสนักศึกษา

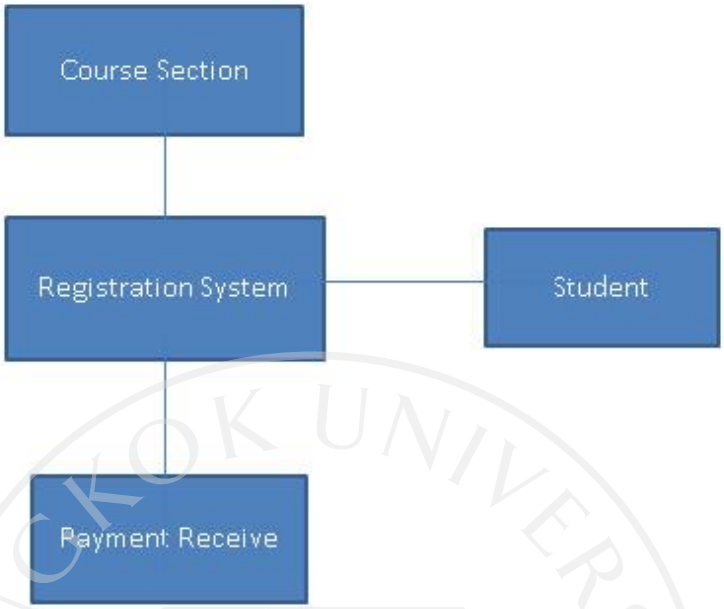
15. ระบบงานวิชาการ

15.1 ค้นหาข้อมูลวิชาและกลุ่มเรียนที่เปิดสอน

3.2.4 สร้างความสัมพันธ์

หลังจากได้กำหนดขั้นตอนการลงทะเบียนเรียบแบบรายวิชาเสร็จเรียบร้อยแล้ว สามารถนำข้อมูลดังกล่าวมาสร้างความสัมพันธ์ของข้อมูลในระบบลงทะเบียนแบบรายวิชาดังภาพที่ 8

ภาพที่ 8: แสดงความพันธ์ของข้อมูลในระบบลงทะเบียนเรียนแบบรายวิชา

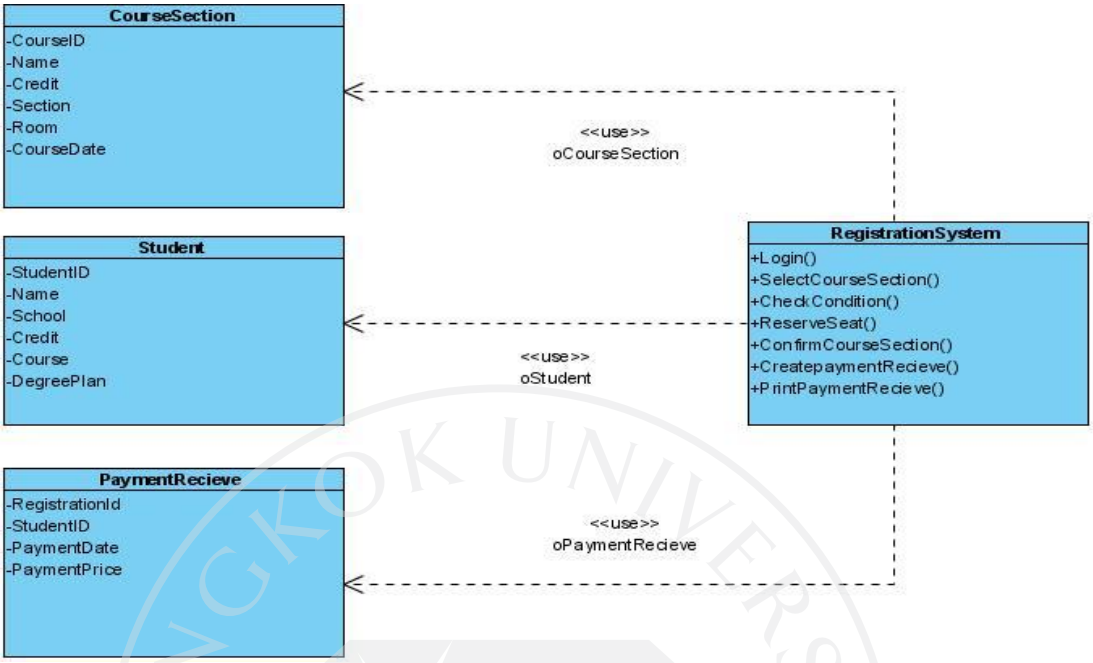


จากภาพที่ 8 สามารถอธิบายได้ว่าระบบลงทะเบียนประกอบไปด้วยข้อมูล 3 ส่วน เช่น การลงทะเบียนในแต่ละรายวิชาจำเป็นต้องใช้ข้อมูลของนักศึกษาที่ลงทะเบียนและต้องมีการตรวจสอบในส่วนของข้อมูลวิชาด้วย เพราะการลงทะเบียนเรียนในแต่ละรายวิชาจำเป็นต้องตรวจสอบแผน
การเรียนให้ตรงกับแผนการเรียนของนักศึกษา และเมื่อทำการลงทะเบียนเรียนเสร็จเรียบร้อยแล้ว นักศึกษาจำเป็นต้องนำเอกสารกำชำระเงินไปชำระเงินค่าลงทะเบียนเรียนตามวันและเวลาที่กำหนดในใบลงทะเบียน

3.2.5 กำหนดคุณสมบัติและพฤติกรรมการทำงาน

กำหนดคุณสมบัติและพฤติกรรมการทำงานที่ได้จากการทำความสัมพันธ์ของข้อมูลเพื่ออธิบายการทำงานทั้งหมด (Arking & Millett, 2009, p.203-204) ได้ดังภาพที่ 9

ภาพที่ 9: แสดงคุณสมบัติและพฤติกรรมการทำงานหลักของระบบ



จากภาพที่ 9 แสดงการทำงานหลักของระบบลงทะเบียนเรียน เช่น นักศึกษาทำการลงทะเบียนเรียนเลือกวิชาและกลุ่มของวิชาที่ต้องการลงทะเบียน (SelectCourseSection) ระบบลงทะเบียนเรียนจะทำการตรวจสอบเงื่อนไขการลงทะเบียนเรียน(CheckCondition)ว่าเงื่อนไขถูกต้องหรือไม่ โดยขั้นตอนต่างๆ ที่ได้มาจากพฤติกรรมหลักของระบบสามารถนำมาออกแบบเพื่อนำมาใช้

ในการแบ่งลำดับชั้นจากที่กล่าวมาส่วนที่เชื่อมต่อกับนักศึกษาสามารถนำมาจัดให้อยู่ในกลุ่มของลำดับชั้นพรีเซนท์เทชัน แต่เนื่องจากกระบวนการดังกล่าวต้องอาศัยกระบวนการอื่นๆ และข้อมูลที่เกี่ยวข้องด้วย ดังนั้นภาพที่ 8 เป็นการแสดงให้เห็นถึงการแบ่งส่วนงานต่างๆ ออกเป็น 3 ลำดับชั้น ซึ่งสามารถจัดออกเป็นลำดับชั้นได้ดังต่อไปนี้

1. ลำดับชั้นพรีเซนท์เทชันเป็นส่วนสำคัญสำหรับผู้ใช้งานเนื่องจากเป็นส่วนที่ติดต่อกับผู้ใช้งาน ซึ่งผู้ใช้งานกระทำบางอย่างในส่วนนี้รวมถึงผลลัพธ์ที่ต้องการแสดงให้ผู้ใช้งานได้เห็น (Microsoft Corporation, 2010) จึงสามารถจำแนกได้ดังนี้

- เพิ่มวิชาเป็นส่วนที่ผู้ใช้งานทำการเพิ่มวิชาในการลงทะเบียนเรียน
- แสดงข้อความให้ผู้ใช้งานทราบเป็นส่วนที่แจ้งข้อความของการลงทะเบียนเรียนให้ผู้ใช้งานทราบเช่น ไม่สามารถลงทะเบียนเรียนได้
- ยืนยันการลงทะเบียนเรียน เป็นส่วนที่ติดต่อกับผู้ใช้งานเพื่อทำการยืนยันการลงทะเบียนเรียนจากผู้ใช้งาน

2. ลำดับชั้นบิสซิเนสเป็นส่วนสำคัญสำหรับประมวลผลความต้องการและเงื่อนไขทางธุรกิจ (Espositi & Saltarello, 2008, p.178) เนื่องจากในลำดับชั้นนี้ทำหน้าที่ประมวลผลและจัดการขั้นตอนการลงทะเบียนเรียนให้ถูกต้องตามลำดับ และทำการตรวจสอบเงื่อนไขตามที่ได้กำหนดไว้ จึงสามารถจำแนกแยกเป็นขั้นตอนได้ดังนี้

- ตรวจสอบข้อมูลการลงทะเบียนเรียนเบื้องต้น ทำหน้าที่ตรวจสอบข้อมูลที่ใช้ในการ
ลงทะเบียนเบื้องต้น
- ตรวจสอบเงื่อนไขวิชาและข้อมูลนักศึกษา ทำหน้าที่ตรวจสอบเงื่อนไขต่างๆ ที่ใช้ในการลงทะเบียนเรียนว่านักศึกษาสามารถที่จะลงทะเบียนได้หรือไม่
- ตรวจสอบความถูกต้องของเงื่อนไขเพื่อนำไปสร้างข้อความแจ้งให้ผู้ใช้งานทราบ
- สร้างเอกสารการชำระเงิน

3. ลำดับชั้นดาต้าแอคเซส เป็นส่วนสำคัญสำหรับการติดต่อกับแหล่งข้อมูล เนื่องจากเป็นส่วนที่ทำหน้าที่บันทึก แก้ไข ลบ และสืบค้นข้อมูลจากแหล่งข้อมูลคืนมาในรูปแบบของออบแจ็คเพื่อส่งต่อไปยังส่วนที่ต้องการใช้ข้อมูลซึ่งข้อมูลที่ใช้อาจนำไปใช้ในการคำนวณหรือการแสดงผล (Microsoft Patterns & Practices Team, 2009, p.97) จึงสามารถจำแนกได้ดังนี้

- บันทึกแก้ไขและลบข้อมูลลงแหล่งข้อมูล ทำหน้าที่นำข้อมูลจัดเก็บลงแหล่งข้อมูล
- ค้นคืนข้อมูลตามความต้องการจากลำดับชั้นบิสซิเนส นำข้อมูลที่ได้เพื่อส่งไปให้ยังลำดับชั้นบิซิเนส

ระบบลงทะเบียนเรียนได้ถูกออกแบบเป็นคลาสและแบ่งเป็นลำดับชั้น 3 ลำดับชั้นดังนี้

1. ลำดับชั้นพรีเซนท์เทชัน (Presentation Layer) จากภาพที่ 6 แสดงให้เห็นตัวอย่างคลาสและเมธอดที่ทำหน้าที่แสดงผลเพื่อติดต่อกับผู้ใช้งานหรือรับข้อมูลมาจากผู้ใช้งานดังตารางที่ 2 – 4

ตารางที่ 2: แสดงคลาส index ในลำดับชั้นพรีเซนท์เทชัน

Presentation Layer		
Class Name:	index	
Return	None	
Method Name	**Return Type**	**Description**
OpenRegisterPage	-	ทำหน้าที่แสดงผลให้ผู้ใช้งานลงทะเบียนเรียน
btnLogin_Click()	-	ทำหน้าที่ตรวจสอบชื่อผู้ใช้งานและรหัสผ่าน

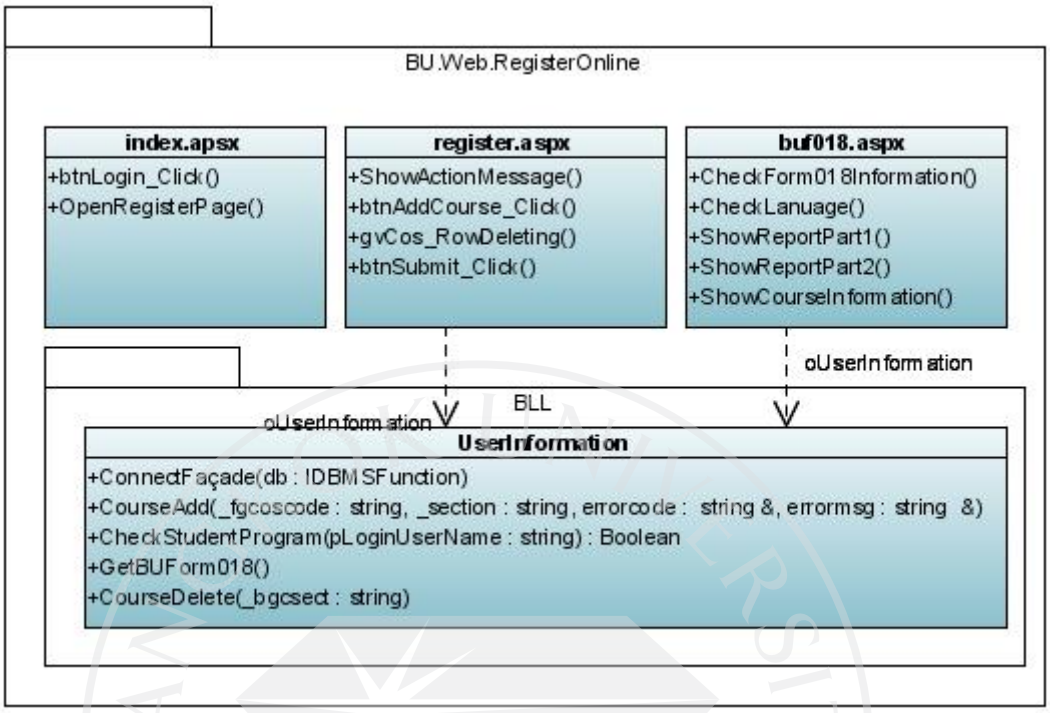
ตารางที่ 3: แสดงคลาส register ในลำดับชั้นพรีเซนท์เทชัน

Presentation Layer		
Class Name:	register	
Return	None	
Method Name	**Return Type**	**Description**
ShowActionMessage	-	ทำหน้าที่แสดงข้อความแจ้งเตือนให้ผู้ใช้งานทราบ
btnAddCourse_Click	-	ทำหน้าที่รับคำสั่งจากผู้ใช้งานเพื่อทำการเพิ่มวิชา
btnSubmit_Click	-	ทำหน้าที่รับคำสั่งจากผู้ใช้งานในการยืนยันผลการลงทะเบียนเรียน
gvCos_RowDeleting	-	ทำหน้าที่ลบวิชาที่ถูกเลือกออกจากรายการที่ได้ลงทะเบียนเรียน

ตารางที่ 4: แสดงคลาส register ในลำดับชั้นพรีเซนท์เทชัน

Presentation Layer		
Class Name:	buf018	
Return	None	
Method Name	**Return Type**	**Description**
CheckForm018Information	-	ทำหน้าที่แสดงข้อมูลนักศึกษาและผลการลงทะเรียน จำนวนเงินที่ต้องชำระ วันเดือนและปีที่ต้องการชำระเงิน และข้อมูลต่างๆที่เกี่ยวข้องกับการชำระเงินที่ธนาคาร
CheckLanguage	-	ทำหน้าที่ตรวจสอบภาษาที่ใช้แสดงผลในเอกสารการชำระเงิน โดยตรวจสอบจากโปรแกรมของนักศึกษาที่ลงทะเบียนเรียน เช่น โปรแกรมภาษาไทย หรือโปแกรมนานาชาติ
ShowReportPart1	-	ทำหน้าที่แสดงข้อมูลนักศึกษาด้านบนของใบชำระเงิน
ShowReportPart2	-	ทำหน้าที่แสดงข้อมูลนักศึกษาด้านล่างของใบชำระเงิน
ShowCourseInformation	-	ทำหน้าที่แสดงข้อมูลรายวิชาที่นักศึกษาลงทะเบียนเรียนลงใบชำระเงิน

ภาพที่ 10: แสดงคลาสบางส่วนในลำดับชั้นพรีเซนท์เทชัน



จากภาพที่ 10 คลาส “UserInformation” เป็นคลาสที่ทำหน้าที่เชื่อมต่อลำดับชั้นพรีเซนท์เทชันของคลาสในกลุ่ม BU Framework และเชื่อมต่อกับส่วนของลำดับชั้นดาต้าแอคเซส สำหรับงานลงทะเบียนเรียนแบบรายวิชา

ภาพที่ 11: หน้าจอลงทะเบียนเรียนของนักศึกษา

ภาพที่ 12: แสดงส่วนของโค้ดในลำดับชั้นพรีเซนท์เทชัน

```
protected void btnAddCos_Click(object sender, ImageClickEventArgs e)
{
    try
    {
        UserInformation objUserInfo =
            (UserInformation)Session[ObjectFactory.SessionUserInformation];


        int result_add;
        result_add = objUserInfo.CourseAdd(
            BUCourseSelector1.SelectedCourseValue,
            BUCourseSelector1.SelectedSectionText.Substring(0,4));

        if (result_add == -1)
        {
            lstbErrorMsg.Items.Add("หน่วยกิตที่ลงทะเบียนในภาคการศึกษานี้ไม่เกิน " +
                objUserInfo.MaxCredit.ToString() +
                " หน่วยกิต และไม่ต่ำกว่า " + objUserInfo.MinCredit +
                " หน่วยกิต (Maximum credits exceeded.)");
        }


        DataTable dt = objUserInfo.GetCourseUptoDate();
        if (dt != null && dt.Rows.Count > 0)
        {
            DisplayErrorList(dt);
        }

        ShowActionMessage(objUserInfo, dt);
        DisplayErrorList(dt);
```

จากภาพที่ 12 เมื่อผู้ใช้งานกดปุ่มในการเพิ่มวิชาระบบก็จะส่งข้อมูลไปที่ลำดับชั้นบิสซิเนส

เพื่อตรวจสอบเงื่อนไขตามความต้องการทางธุรกิจที่ได้ออกแบบไว้ เมื่อระบบทำการตรวจสอบตามเงื่อนไขที่ได้ออกแบบไว้ตามความต้องการทางธุรกิจก็จะส่งข้อความตอบกลับมาเพื่อแจ้งให้ผู้ใช้งานทราบ

1. ลำดับชั้นบิสซิเนส (Business Layer) จากภาพที่ 13 แสดงให้เห็นตัวอย่างคลาสที่เกี่ยวข้องกับความต้องการและเงื่อนไขทางธุรกิจ ดังตารางที่ 5-6

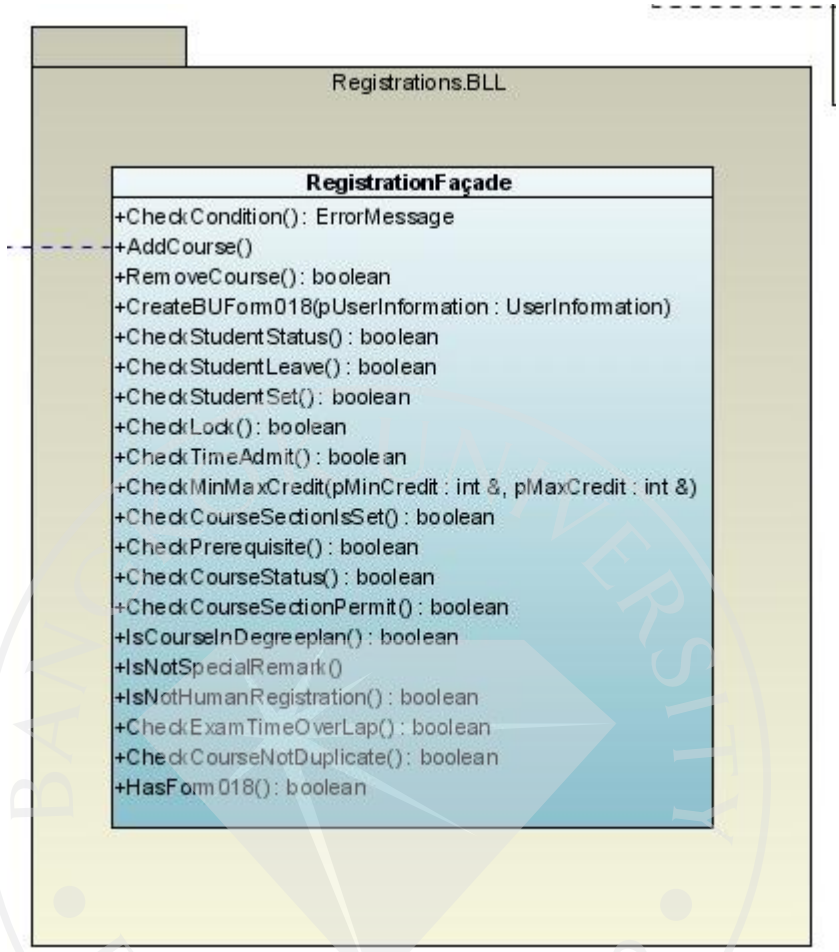
ตารางที่ 5 : แสดงคลาส BU.LOB.Academic.IAcademicFactory ในลำดับชั้นบิสซีเนส

Business Logic Layer		
Class Name:	BU.LOB.Academic.IAcademicFactory	
Return	None	
Method Name	**Return Type**	**Description**
ListCourseSection	List<<Course>>	ทำหน้าที่แสดงรายชื่อวิชาทั้งหมดที่เปิดให้สามารถลงทะเบียนได้
GetcurrentSemester	String	ทำหน้าค้นคืนเทอมการศึกษาที่ใช้ในการลงทะเบียนเรียน
ListAllcourse	List<<Course>>	ทำหน้าที่ค้นคืนข้อมูลวิชาทั้งหมด
ListCourse	List<<Course>>	ทำหน้าที่ค้นคืนข้อมูลวิชา

ตารางที่ 6: แสดงคลาส Registrations.BLL.Registrationfaçade ในลำดับชั้นบิสซีเนส

Business Logic Layer		
Class Name:	Registrations.BLL.Registrationfaçade	
Return	None	
Method Name	**Return Type**	**Description**
CheckMinMaxCredit	int	ทำหน้าที่ตรวจสอบว่าจำนวนหน่วยกิตที่สามารถลงทะเบียนได้
CheckStudentStatus	boolean	ทำหน้าที่ตรวจสอบสถานะนักศึกษา
CheckCouseSchedule	boolean	ทำหน้าที่ตรวจสอบตารางเรียนของนักศึกษาว่าวิชาที่ทำการลงทะเบียนตรงกับวิชาที่ได้ลงทะเบียนไปแล้วหรือไม่
CheckCourseValidation	boolean	ทำหน้าที่ตรวจสอบว่าวิชาที่เลือกสามารถลงทะเบียนเรียนได้หรือไม่
CheckCondition	boolean	ทำหน้าที่ตรวจสอบเงื่อนไขในการลงทะเบียนเรียน
AddCourse	boolean	ทำหน้าที่เพิ่มรายวิชาในการลงทะเบียนเรียน
RemoveCourse	boolean	ทำหน้าที่นำวิชาที่ลงทะเบียนออกจากการลงทะเบียนเรียน
CreateBUForm018	-	ทำหน้าที่สร้างใบชำระเงิน
CheckStudentStatus	boolean	ทำหน้าที่ตรวจสอบสถานะนักศึกษา
CheckStudentLeave	boolean	ทำหน้าที่ตรวจสอบสถานะนักศึกษาว่าได้ลาพักหรือไม่
CheckStudentSet	boolean	ทำหน้าที่ตรวจสอบนักศึกษาลงทะเบียนแบบกลุ่มหรือไม่
CheckCourseSectionIsSet	boolean	ทำหน้าที่ตรวจสอบรายวิชาที่ลงทะเบียนแบบเซ็ทหรือไม่
CheckPrerequisite	boolean	ทำหน้าที่ตรวจสอบว่าวิชาที่ลงทะเบียนต้องผ่านวิชาพื้นความรู้หรือไม่

ภาพที่ 13: แสดงคลาสบางส่วนในลำดับชั้นบิสซิเนส



ภาพที่ 14: ตัวอย่างโค้ดในเมธอด CheckCourseValidationในลำดับชั้นบิสซิเนส

```
private void CheckCourseValidation(
        BU.Entities.Academic.CourseSection _cs,
        int _i,
        ref string _errorcode,
        ref string _errormessage) {
        bool result;

        //string yearTemp = (Convert.ToInt32(regisForm.Semester.Year)+543).ToString().Substring(2,2);

        BU.LOB.Fa?ade.DAL.Registration reg
            = new BU.LOB.Fa?ade.DAL.Registration(
                ObjectFactory.GetDBProvider());

        string[] _SeatTextArr =
            reg.GetSeatUseAndSeatNum(
            RegisterYear,RegisterSemester,
            _cs.Course.SectionList[0].ID).Split('/');

        _cs.Course.SectionList[0].SeatUsed = Convert.ToInt32(_SeatTextArr[0]);
        _cs.Course.SectionList[0].SeatNum = Convert.ToInt32(_SeatTextArr[1]);

        if (_cs.Course.SectionList[0].SeatUsed <
            _cs.Course.SectionList[0].SeatNum) { //ตรวจสอบที่นั่ง
            // 3. string pAcdYear, string pSemester, string p0gCSectCode  // <returns>TRUE = SET / FALSE = NOT SET</returns>
            result = objFa?adeBLL.CheckCourseSectionIsSet(
                regisForm.Semester.Year,
                regisForm.Semester.Code,
                _cs.Course.SectionList[0].ID);

            if (result == false) { // ใช่ห้องไม่เป็น set
                // 4. ตรวจสอบวิชาพื้นความรู้ของวิชาที่ลง // <returns>TRUE = PASS / FALSE = NOT PASS</returns>
                result = objFa?adeBLL.CheckPrerequisite(regisForm.Semester.Year, regisForm.Semester.Code, regisForm.Student.ID, _cs.Course.Code, _cs.Course.SectionList[
```

```
if (result == true) {
    // 5. ตรวจสอบเวลาเรียนตรงกัน // ตรวจจากรายวิชาที่ลงทะเบียนไว้
    result = CheckCourseSchedule(_cs, _i, ref _errorcode, ref _errormessag
```

```
            // 7. ตรวจสอบว่าวิชานี้อยู่ในแผนที่กำหนดลงทะเบียนหรือไม่ // <returns>TRUE = PASS / FALSE = NOT PASS</returns>
            result = objFa?adeBLL.CheckCourseSectionPermit(_cs.Course.Code, _cs.Course.SectionList[0].SectionNo, ref _errorcode, ref _errormessage);
```

จากภาพที่ 14 เป็นคลาสที่รวบรวมความต้องการทางธุรกิจหลายๆความต้องการให้เป็นความต้องการเดียวโดยมีลำดับการทำงานที่เป็นขั้นตอนตามที่ได้ออกแบบจากภาพที่ 14 จะมีขั้นตอน

การทำงานส่วนหนึ่งที่ทำหน้าที่เรียกไปยังคลาส “CheckCourseSchedule” เพื่อทำการตรวจดูว่าตารางเรียนของนักศึกษาตรงกับวิชาที่ได้ลงทะเบียนไปแล้วหรือไม่ เมื่อทำการตรวจสอบเงื่อนไขเรียบร้อยแล้ว ก็จะทำการตรวจสอบเงื่อนไขอื่นๆ ที่เหลือจนครบตามเงื่อนไขที่ได้ออกแบบไว้ในลำดับชั้นนี้จะเกี่ยวข้องกับกระบวนการและเงื่อนไขทางธุรกิจเท่านั้น ในอนาคตถ้าเงื่อนไขในการตรวจสอบ

การลงทะเบียนเรียนเกิดการเปลี่ยนแปลงก็สามารถที่จะเข้ามาแก้ไขปรับปรุงหรือเพิ่มได้ที่คลาส “CheckCourseSchedule” เพราะมีการกำหนดหน้าที่การทำงานเฉพาะด้าน และสามารถแก้ไขปรับปรุงความต้องการทางธุรกิจได้ง่ายขึ้น

1. ลำดับชั้นดาต้าแอคเซส (Data Access Layer) จากภาพที่ 11 แสดงให้เห็นตัวอย่างคลาสที่เกี่ยวข้องกับการเข้าถึงแหล่งข้อมูล การเพิ่ม แก้ไข ลบและค้นหาแหล่งข้อมูล ดังที่จะแสดงให้เห็นในตารางที่ 7

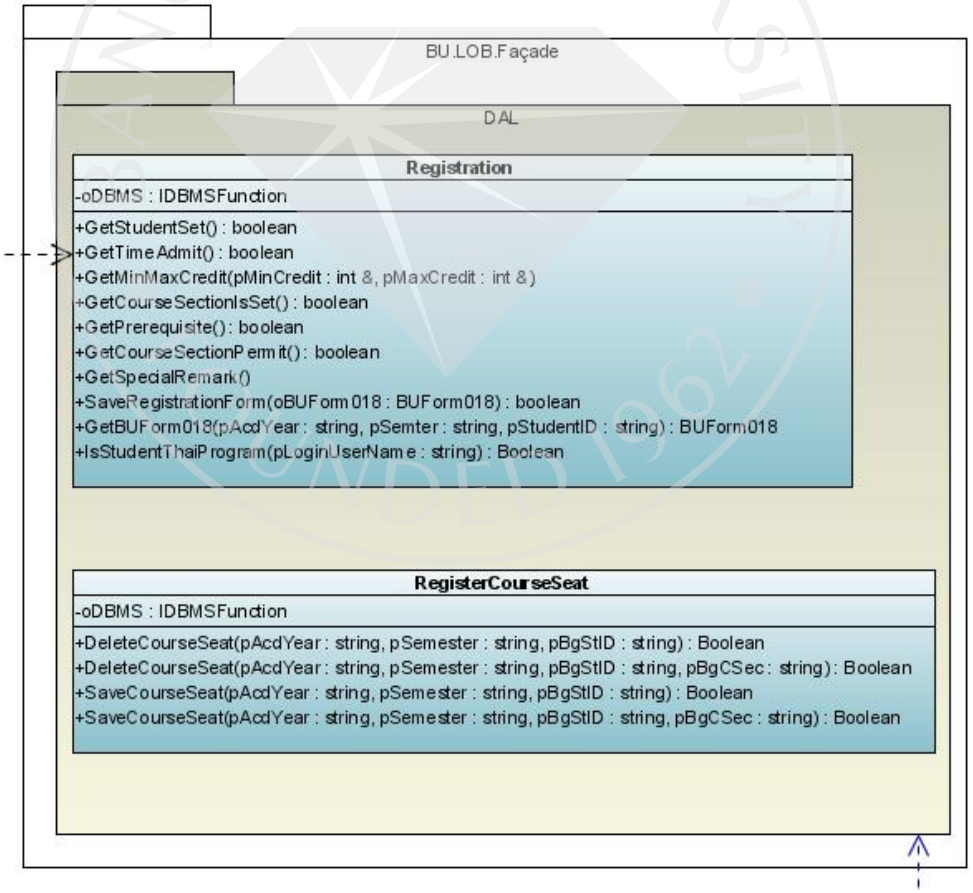
ตารางที่ 7: แสดงคลาส BU.LOB.DAL.façade ในลำดับชั้นดาต้าแอคเซส

Data Access Layer		
Class Name:	BU.LOB.DAL.façade	
Return	None	
Method Name	**Return Type**	**Description**
GetStudentSet	Boolean	ค้นคืนข้อมูลนักศึกษาว่าเป็นการลงทะเบียนเป็นเซ็ทหรือไม่
GetTimeAdmit	Boolean	ค้นคืนข้อมูลช่วงของเวลาที่เปิดให้นักศึกษาลงทะเบียนเรียน
GetMinMaxCredit	Int	ค้นคืนข้อมูลจำนวนหน่วยกิตที่เปิดให้ลงทะเบียนเรียนว่าลงทะเบียนได้กี่หน่วยกิต
GetCourseSectionSet	Boolean	ค้นคืนชื่อวิชาและกลุ่มของวิชาที่เปิดให้ลงทะเบียนเรียน
GetPrerequisite	Boolean	ค้นคืนข้อมูลวิชาที่ต้องผ่านวิชาพื้นความรู้มาก่อนถึงจะทำการลงทะเบียนเรียนได้
GetSpecialRemark	-	ค้นคืนข้อมูลว่านักศึกษามีเงื่อนไขพิเศษหรือไม่
SaveRegistrationForm	Boolean	บันทึกผลการลงทะเบียนเรียน
GetBUForm018	BUForm018	ค้นคืนข้อมูลแบบฟอร์มการชำระเงิน
isStudentThaiProgram	Boolean	ค้นคืนข้อมูลนักศึกษาว่าเป็นโปรแกรมภาษาไทย

ตารางที่ 8: แสดงคลาส BU.LOB.DAL.RegisterCourseSeat ในลำดับชั้นดาต้าแอคเซส

Data Access Layer		
Class Name:	BU.LOB.DAL.RegisterCourseSeat	
Return	None	
Method Name	**Return Type**	**Description**
DeletecourseSeat	Boolean	ทำหน้าที่ลบจำนวนที่นั่งในวิชาที่นักศึกษาได้ลงทะเบียนเรียน
SavecourseSeat	Boolean	ทำหน้าที่บันทึกจำนวนที่นั่งที่นักศึกษาได้ลงทะเบียนเรียนเรียบร้อยแล้ว

ภาพที่ 15: แสดงคลาสบางส่วนที่ใช้ในลำดับชั้นดาต้าแอคเซส

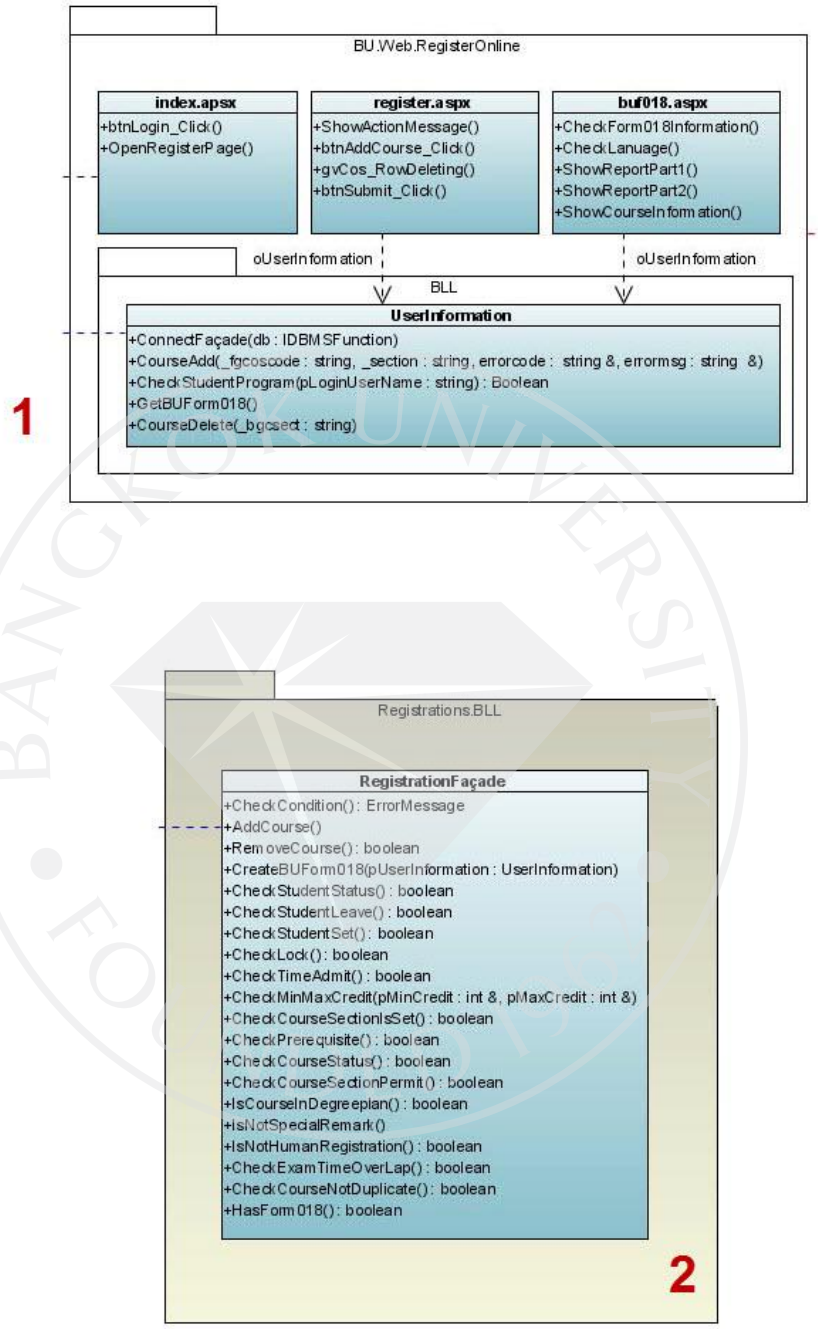


ภาพที่ 16: ตัวอย่างโค้ดค้นคืนข้อมูลชื่อวิชาและกลุ่มวิชาในลำดับชั้นดาต้าแอคเซส

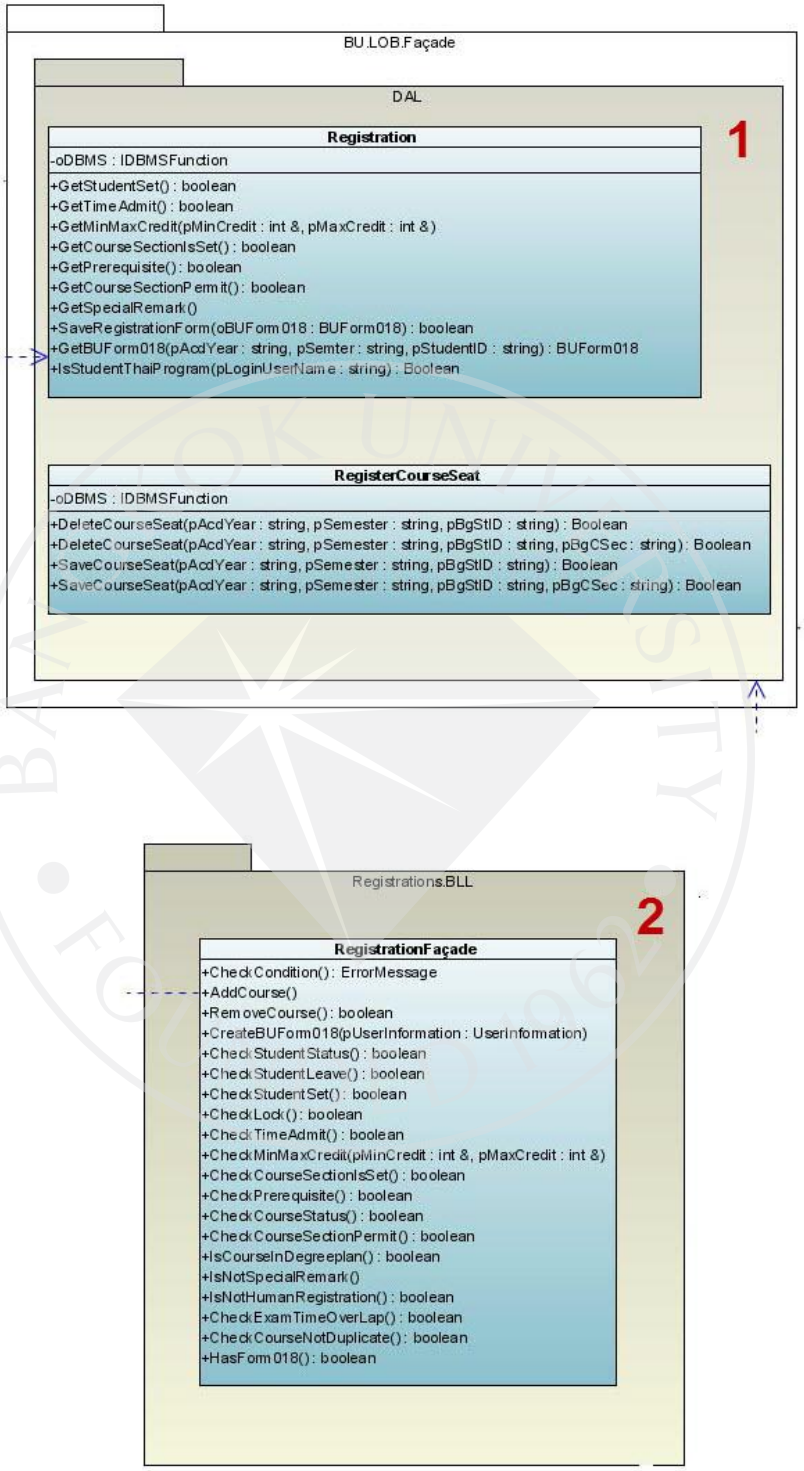
```
public List<Course> ListCourse(string acedemicYearCode, string acedemicSemesterCode)
{
    SetDefaultSQL();
    try
    {
        And += "and AcdYr = '" + acedemicYearCode + "' ";
        MainSQL = Select + From + Where + And;
        reader = ADb.LoadToDataTableReader(MainSQL);
    }
    catch (Exception ex)
    {
        throw new Exception("ERROR: From Method [ListCourse]", ex);
    }
    if (reader.HasRows)
    {
        aCourseList = new List<Course>();
        //aSemesterDao = aDaoFactory.GetSemesterDao();
        //aSemesterList = aSemesterDao.GetAll();
        aSemesterDao = new SemesterDaoSybase(ADb);
        try
        {
aSemesterDao.ADb = this.ADb;
aSemester = aSemesterDao.GetByDegreeProgram("2", acedemicYearCode, acedemicSemesterCode, "T");
        }
        catch
        {
            aSemester = null;
        }
        while (reader.Read())
        {
            aCourse = new Course();

            SetEntityProperties();

            aCourseList.Add(aCourse);
        }
        reader.Close();
    }
    else
    {
        aCourseList = null;
    }
    return aCourseList;
}
```

จากภาพที่ 16 เป็นเมธอดที่ทำหน้าที่ค้นคืนข้อมูลวิชาที่สามารถลงทะเบียนได้ในเทอมนั้น ๆ จากแหล่งข้อมูลแล้วนำข้อมูลที่ค้นคืนได้ส่งไปยังส่วนที่เรียกใช้งานมา ค่าที่ได้กลับมานั้นถูกเก็บลงตัวแปรประเภทออบเจ็กต์เพื่อนำผลลัพธ์ที่ได้จากการค้นคืนไปประมวลผลและนำมาแสดงให้ผู้ใช้งานเห็นวิชาที่เปิดให้ลงทะเบียนเรียน

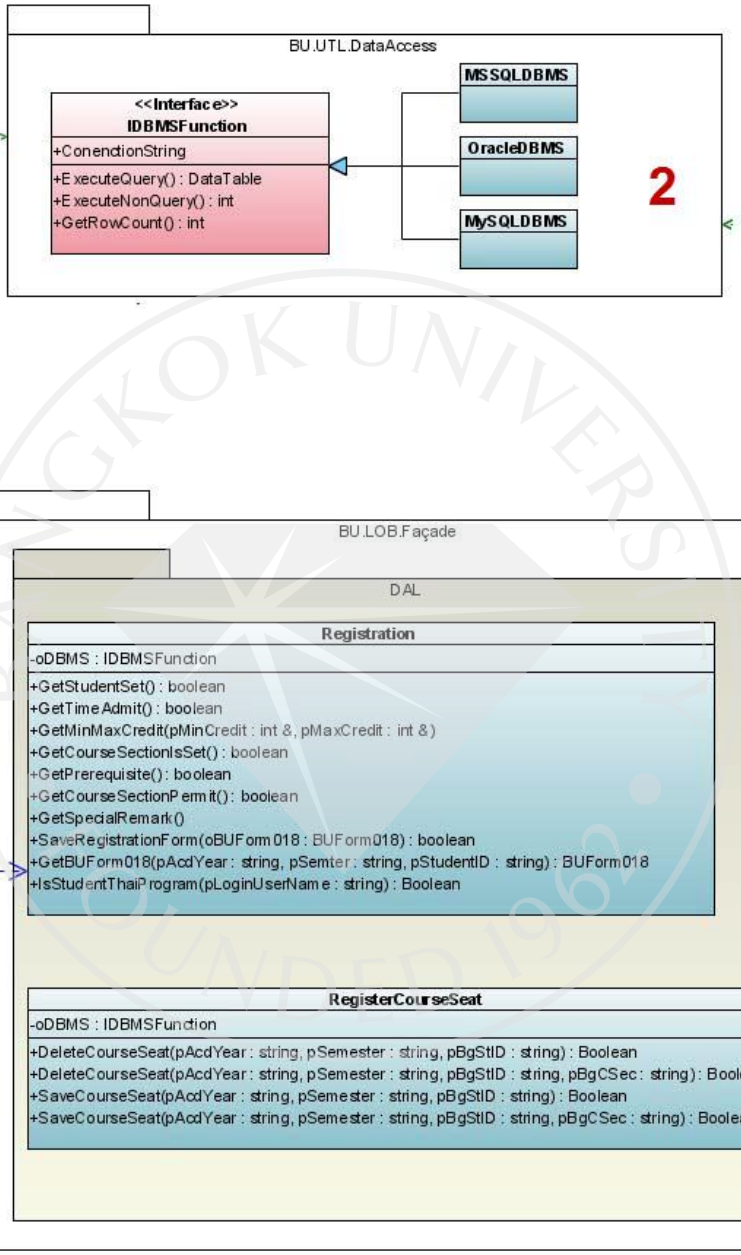
ภาพที่ 17: แสดงความสัมพันธ์ระหว่างลำดับชั้นพรีเซนท์เทชันกับลำดับชั้นบิสซิเนส



ภาพที่ 18: แสดงความสัมพันธ์ระหว่างลำดับชั้นบิสซิเนสกับลำดับชั้นดาต้าเอคเซส



ภาพที่ 19: แสดงความสัมพันธ์ระหว่างลำดับชั้นดาต้าเอคเซสกับแหล่งข้อมูล



จากภาพที่ 17, 18และ 19 แสดงให้เห็นการเชื่อมโยงความสัมพันธ์ของคลาสในลำดับชั้นพรีเซนท์เทชันกับคลาสที่อยู่ในลำดับชั้นบิสซีเนส การเชื่อมโยงความสัมพันธ์ของคลาสในลำดับชั้น
บิสซีเนสกับคลาสที่อยู่ในลำดับชั้นดาต้าแอคเซส และการเชื่อมโยงความสัมพันธ์ของคลาสในลำดับชั้นดาต้าแอคเซสกับแหล่งข้อมูลว่ามีการเชื่อมโยงระหว่างคลาสเป็นแบบใด รวมถึงมีการเรียกใช้งานระหว่างแต่ละลำดับชั้นอย่างไร ยกตัวอย่างเช่น นักศึกษาเข้าใช้งานระบบลงทะเบียนเรียนแบบรายวิชา ในระบบก็จะทำการตรวจสอบว่านักศึกษาที่เข้าใช้ระบบสามารถลงทะเบียนในช่วงเวลานี้ได้หรือไม่ โดยมีขั้นตอนการทำงานดังต่อไปนี้

1. นักศึกษาทำการเข้าใช้ระบบผ่านทางหน้าจอ “index.aspx” โดยอยู่คลาส “BU.Web.RegisterOnline.index” (ภาพที่ 17 หมายเลขที่ 1) หน้าสำหรับเข้าใช้งานจะเลือกเมธอด “btnLogin_click” เพื่อเข้าใช้งาน

2. ทำการตรวจสอบจากรหัสนักศึกษา ว่านักศึกษาอยู่ในช่วงของการลงทะเบียนเรียนแบบรายวิชาหรือไม่ โดยตรวจสอบจาก คลาส “BU.LOB.Records.Registrations.BLL.RegistrationFaçade” (ภาพที่ 17 หมายเลขที่ 2)

3. คลาส “BU.LOB.Records.Registrations.BLL.RegistrationFaçade” (ภาพที่ 18 หมายเลขที่ 1) จะไปเรียกขอข้อมูลจากเมธอด “BU.LOB. Façade.DAL.Registration.GetTimeAdmit” (ภาพที่ 18 หมายเลขที่ 2)โดยจะมีการส่งข้อมูลรหัสนักศึกษา ปีเทอมที่ลงทะเบียน เพื่อไปทำการตรวจสอบว่านักศึกษาสามารถที่จะลงทะเบียนแบบรายวิชาได้หรือไม่

4. เมธอด “BU.LOB. Façade.DAL.Registration.GetTimeAdmit” ทำหน้าที่ค้นคืนข้อมูลช่วงเวลาที่นักศึกษาสามารถลงทะเบียนได้จากแหล่งข้อมูลในคลาส “BU.UTL.DataAccess.IDBMSFunction” คลาส “BU.UTL.DataAccess.IDBMSFunction” (ภาพที่ 19 หมายเลขที่ 2) ก็จะคืนค่าช่วงเวลาที่นักศึกษาสามารถลงทะเบียนได้กลับมา

5. คลาส “BU.LOB.Records.Registrations.BLL.RegistrationFaçade” เมื่อได้ข้อมูลช่วงเวลาของการลงทะเบียนที่นักศึกษาสามารถลงทะเบียนได้ก็จะทำการตรวจสอบว่านักศึกษาสามารถลงทะเบียนเรียนได้ตามวันและเวลาดังกล่าวหรือไม่ ถ้าไม่สามารถลงทะเบียนในช่วงเวลาดังกล่าวได้ก็จะส่งข้อความไปยัง หน้าจอเพื่อแจ้งให้นักศึกษาทราบว่าไม่สามารถลงทะเบียนเรียนในช่วงวันและเวลาดังกล่าวได้

โดยจะแสดงโค้ดตัวอย่างสำหรับการทำงานแต่ละลำดับชั้นว่ามีความสัมพันธ์ในแต่ละลำดับชั้นอย่างไรบ้าง ดังแสดงในภาพที่ 20

ภาพที่ 20: แสดงโค้ดตัวอย่างการเข้าใช้งาน

```
protected void btnLogin_Click(object sender, ImageClickEventArgs e)
{
    if (CheckLogin(txbUsername.Text.Trim(),
        txbPassword.Text.Trim()))
    {
        if (CheckStudentProgram(txbUsername.Text.Trim()))
        {
            Session.Clear();

            SetAcdYearAndUserLogin(txbUsername.Text.Trim());

            SaveLog(Session["AcdYear"].ToString(),
                Session["Semester"].ToString(), "",
                oUserLogin,
                "R", RegistrationAction.USER_STEP_ONE, "");

            Session["PageName"] = "INDEX";

            Page.Response.Redirect("step1.aspx");
        }
        else
        {
            lbMsgLogin.Text = "<B>นักศึกษาหลักสูตรนานาชาติ</B> <BR> ไม่มีสิทธิ์ลงทะเบียนเรียนในระบบนี้";
        }
    }
    else
    {
        lbMsgLogin.Text = "Login Failure.";
    }
}
```

ภาพที่ 21: แสดงโค้ดที่ทำหน้าที่ตรวจสอบการใช้งาน

```
private bool CheckLogin(string pLoginName, string pLoginPassword) {

    if (pLoginName != "" && pLoginPassword != "") {
        // ตรวจสอบ Login Name และ Password
        BU.Security.Authentication.ActiveDirectory ldap_verify =
            new BU.Security.Authentication.ActiveDirectory();

        bool isSuccess = false;
        ldap_verify.IsAuthenticate(pLoginName, pLoginPassword);
        return isSuccess;
    }
    else {
        lbMsgLogin.Text = "กรุณาระบุชื่อ และรหัสผ่าน <BR>User name and Password can not empty.";
        return false;
    }
}
```

จากภาพที่ 20 และ 21 แสดงให้เห็นโค้ดที่ใช้ติดต่อกันระหว่างลำดับชั้นพรีเซนท์เทชันกับลำดับชั้นบิสซีเนส

ภาพที่ 22 แสดงโค้ดสำหรับตรวจสอบสถานะของนักศึกษาในการลงทะเบียนเรียน

```
public bool CheckStudentStatus(
    string pAcdYear, string pSemester,
    string pBgStID, string pFullId, string pProgramCode, string pWebAppCode,
    ref string pSwichCode, ref string pWebNote,ref string pErrorMessage) {
    //2.1 ตรวจสอบการลาพักของนักศึกษา 10,20
    if (CheckStudentLeave(pAcdYear, pSemester, pBgStID)) {
        pErrorMessage = @"นักศึกษาที่มีสถานะลาพัก  ไม่สามารถลงทะเบียนเรียนทางอินเทอร์เน็ตได้";
        return false;
    }

    //2.2 ตรวจสอบเวลาให้บริการของระบบ WebAppCode => "0015" => การลงทะเบียนแบบไม่ใช่ Set
    if (!CheckStudentSet(pAcdYear, pSemester, pProgramCode, pWebAppCode, ref pSwichCode, ref pWebNote))
        pErrorMessage = @"นักศึกษาต้องลงทะเบียนเรียนแบบเซตเสรี";
        return false;
    }

    //2.3 ตรวจอสบสถานะการล็อคเอกสาร หรืออื่นๆ
    if (CheckLock(pFullId)) {
        pErrorMessage =
            @"นักศึกษาถูกระงับการออกเอกสาร โปรดติดต่อสำนักทะเบียนนักศึกษา <br>(Sorry, registration report of this stude
        return false;
    }

    //2.4 ตรวจสอบช่วงเวลาลงทะเบียน
    if (!CheckTimeAdmit(pAcdYear, pSemester, pWebAppCode, pFullId)) {
        pErrorMessage = @"ต้องลงทะเบียนเรียน/พิมพ์ใบ  ในช่วงเวลาที่กำหนดเท่านั้น <br>(Registration is allowed during spe
        return false;
    }

    pErrorMessage = "OK";
    return true;
}
```

จากภาพที่ 22 ในหัวข้อ 2.4 จะเป็นการตรวจสอบช่วงเวลาที่นักศึกษาสามารถลงทะเบียนเรียนได้โดยจะเรียกไปยัง เมธอด “CheckTimeAdmit” เพื่อขอข้อมูลช่วงเวลาที่นักศึกษาสามารถที่ลงทะเบียนได้

ภาพที่ 23: แสดงโค้ด เมธอด “CheckTimeAdmit” ในลำดับชั้นดาต้าแอคเซส

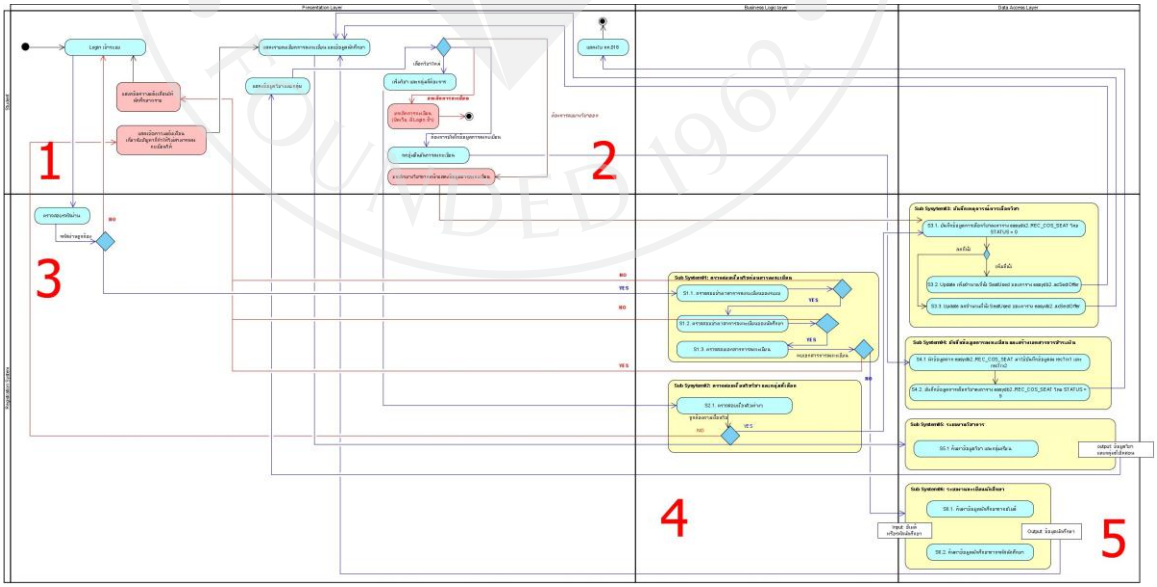
```
private bool CheckTimeAdmit
    (string pAcdYear, string pSemester,
    string pWebAppCode, string pFullId)
{
    return FaçadeDAL.CheckTimeAdmit(pAcdYear, pSemester, pWebAppCode, pFullId);
}
```

จากภาพที่ 23 แสดงให้เห็นโค้ดในลำดับชั้นดาต้าแอคเซส และมีการเรียกค้นคืนข้อมูลจากแหล่งข้อมูลในเมธอด “FaçadeDAL.CheckTimeAdmit”

ภาพที่ 24: แสดงโค้ดค้นคืนช่วงเวลาที่นักศึกษาสามารถลงทะเบียนได้จากแหล่งข้อมูล

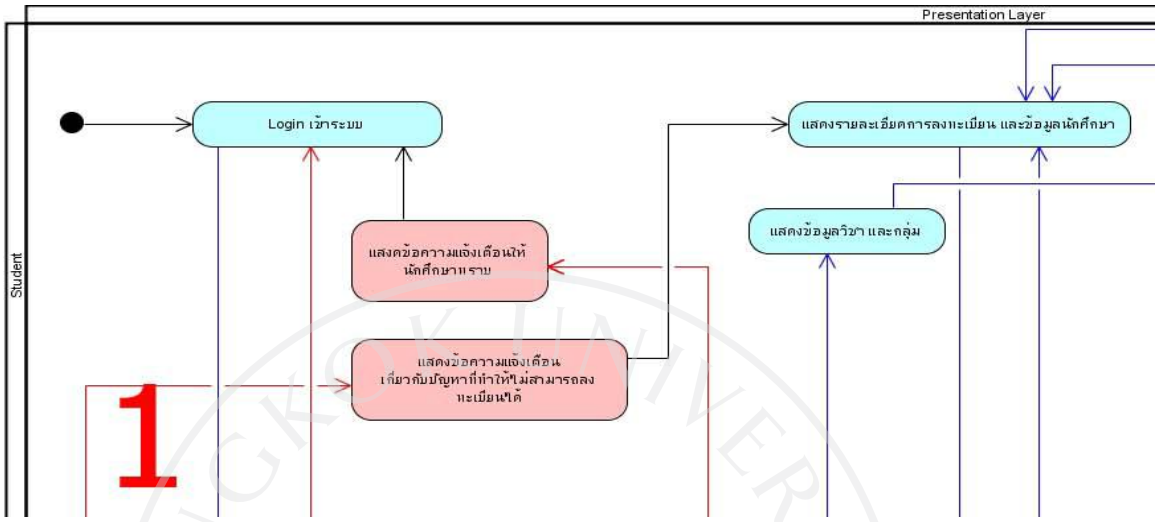
```
public bool CheckTimeAdmit(string pAcdYear, string pSemester,
    string pWebAppCode, string pFullId)
{
    sqlText = string.Format(
        "usp_aw_RGO_ROCS_checktime '{0}','{1}','{2}','{3}'",
        pAcdYear, pSemester, pWebAppCode, pFullId);

    string result = dbms.ReturnFirstRowFirstColumn(sqlText);

    if (result.Trim() == "1")
        return true;

    return false;
}
```

ภาพที่ 25: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้น

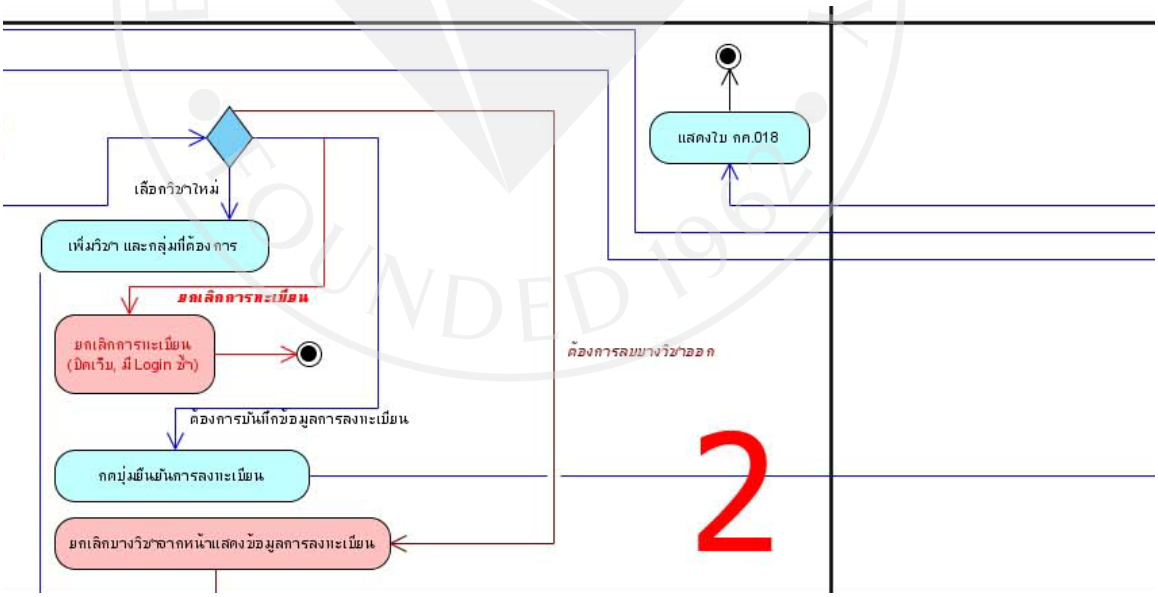


จากภาพที่ 25 ได้แบ่งออกเป็น 5 ส่วนดังภาพที่ 26 – 30 เพื่อแสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้นได้ชัดเจนขึ้น

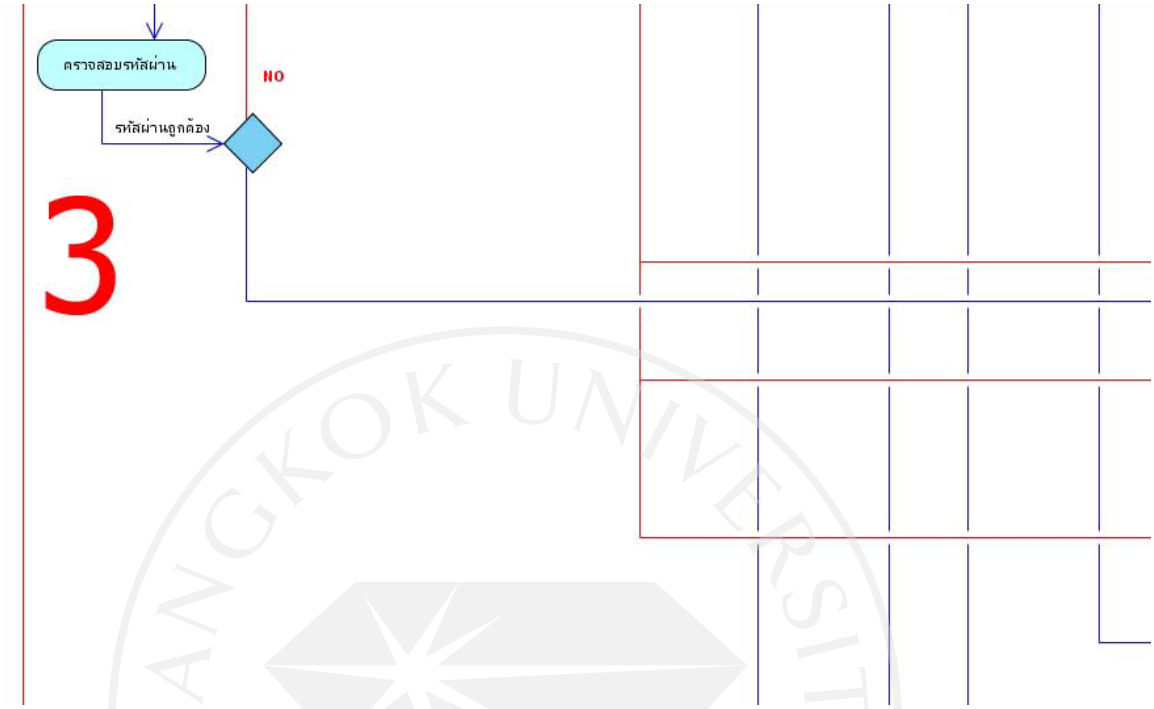
ภาพที่ 26: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 1



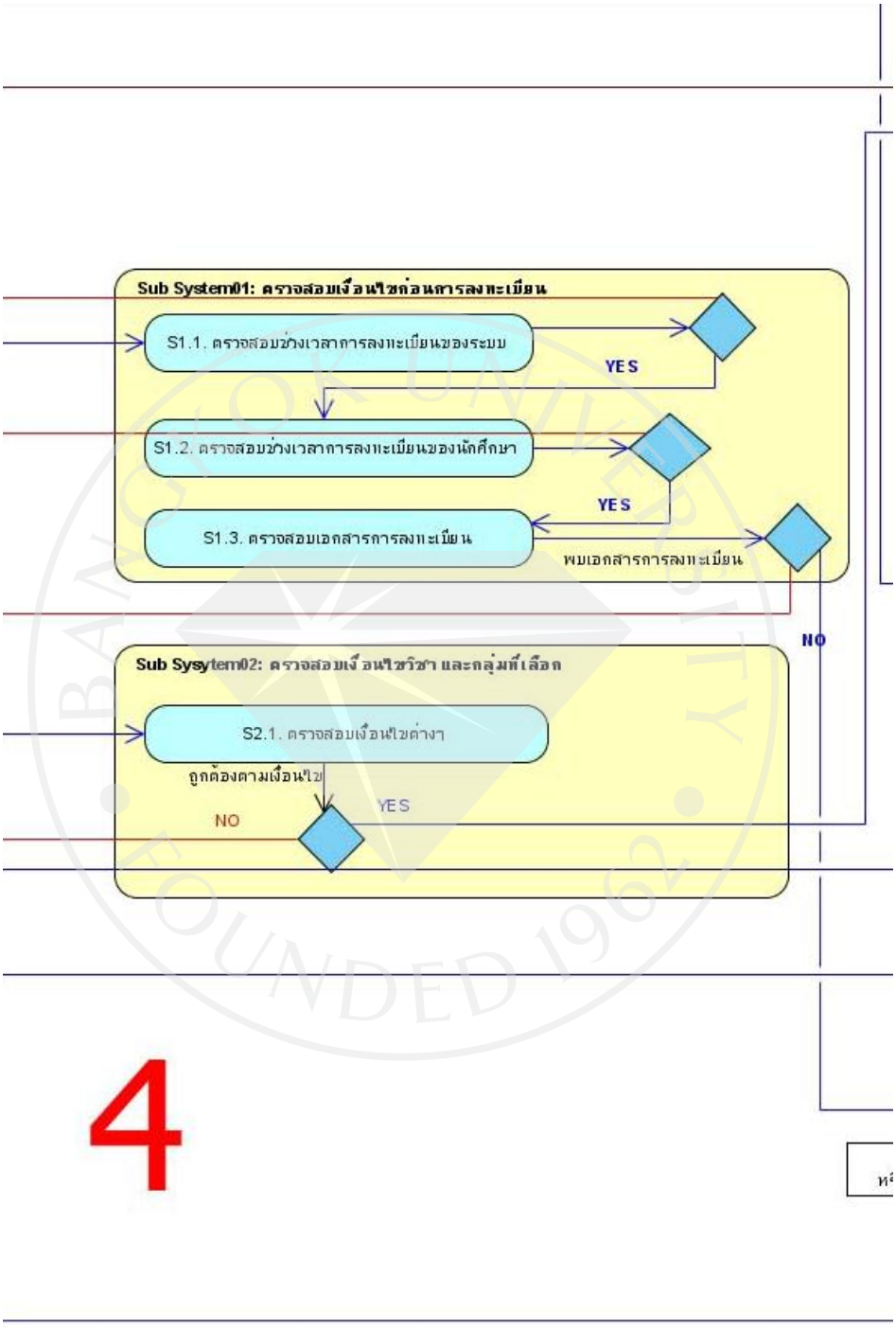
ภาพที่ 27: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 2



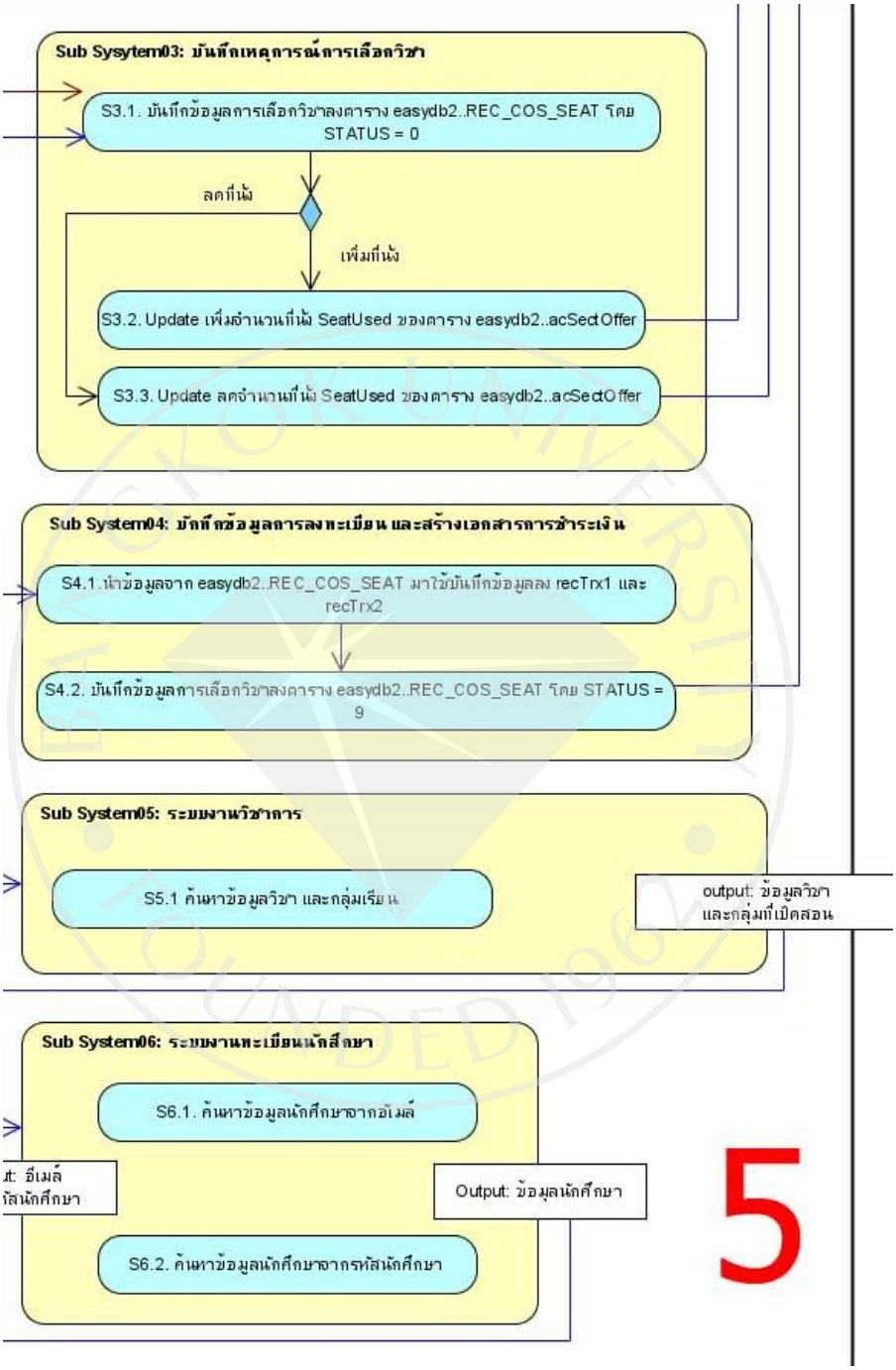
ภาพที่ 28: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 3



ภาพที่ 29: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 4



ภาพที่ 30: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 5



จากภาพที่ 25 อธิบายถึงขั้นตอนการทำงานสำหรับการลงทะเบียนเรียนแบบรายวิชาโดยแบ่งออกเป็น 2 มุมมองสำหรับผู้เล่นของระบบ คือ

1. ส่วนที่แสดงการทำงานในแนวนอน ซึ่งประกอบไปด้วย ระบบลงทะเบียนเรียนกับนักศึกษา

2. ส่วนที่แสดงการทำงานในแนวตั้ง เป็นส่วนที่แสดงการทำงานในแต่ละลำดับชั้นประกอบไปด้วย ลำดับชั้นพรีเซนท์เทชัน ลำดับชั้นบิสซีเนสและลำดับชั้นดาต้าแอคเซส

จากภาพที่ 21 สามารถอธิบายขั้นตอนการทำงานของระบบการลงทะเบียนได้ดังต่อไปนี้

1. เมื่อนักศึกษาเข้าใช้งานในส่วนของลำดับชั้นพรีเซนท์เทชันเพื่อตรวจสอบชื่อผู้ใช้งาน

2. ระบบทำการตรวจสอบชื่อผู้ใช้กับรหัสผู้ใช้งานว่าสามารถเข้าใช้งานได้หรือไม่

3. ในกรณีที่สามารถเข้าใช้งานได้ ระบบจะเข้าไปตรวจสอบเงื่อนไขต่างๆ (ลำดับชั้นบิสซีเนส) ที่ใช้ในการลงทะเบียน ในขั้นตอนการตรวจสอบเงื่อนไขต่างๆ จะมีการเข้าไปค้นคืนข้อมูล (ลำดับชั้นดาต้าแอคเซส) เพื่อนำข้อมูลดังกล่าวมาตรวจสอบเงื่อนไขต่างๆ ก่อนที่จะมีการลงทะเบียนเกิดขึ้น ในกรณีที่เข้าใช้งานไม่ได้ระบบก็จะแจ้งข้อความเตือนไปยังผู้ใช้งานว่าไม่สามารถเข้าใช้งานได้ด้วยเงื่อนไขใด (ลำดับชั้นพรีเซนท์เทชัน)

4. หน้าจอสำหรับลงทะเบียนเรียนแบบรายวิชา ตัวระบบลงทะเบียนเรียนแบบรายวิชาจะเข้าไปค้นคืนข้อมูลวิชาและกลุ่มวิชาที่เปิดให้นักศึกษาสามารถลงทะเบียนเรียนได้ มาแสดงผล

5. เมื่อนักศึกษาเลือกวิชาที่ลงทะเบียนเรียน (ลำดับชั้นพรีเซนท์เทชัน) ตัวระบบลงทะเบียนก็จะทำการตรวจสอบว่านักศึกษาสามารถลงทะเบียนเรียนวิชาที่นักศึกษาสามารถเลือกได้หรือไม่ (ลำดับชั้นบิสซีเนส) โดยมีการเข้าไปค้นคืนข้อมูลนักศึกษาและข้อมูลวิชา (ลำดับชั้นดาต้าแอคเซส) มาตรวจสอบเงื่อนไขการลงทะเบียนเรียน

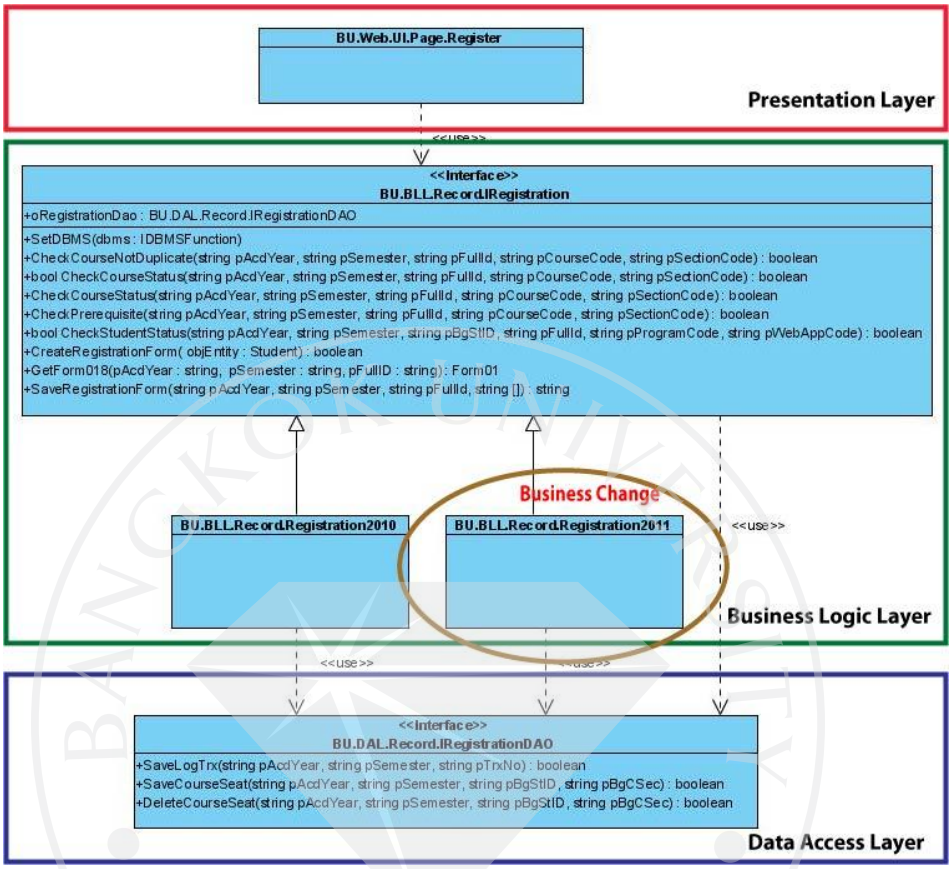
6. นักศึกษาบันทึกวิชาที่ได้ลงทะเบียนเรียน (ลำดับชั้นพรีเซนท์เทชัน) ระบบจะทำการตรวจสอบเงื่อนไขต่างๆ ในการลงทะเบียน(ลำดับชั้นบิสซีเนส) เช่น จำนวนหน่วยกิตที่ได้ลงทะเบียนตรงตามเงื่อนไขหรือไม่ ถ้าตรงตามเงื่อนไขระบบก็ทำการบันทึกข้อมูลวิชาที่ลงทะเบียนเรียนเข้าไปยังแหล่งข้อมูล(ลำดับชั้นดาต้าแอคเซส) พร้อมทั้งข้อความแจ้งให้นักศึกษาทราบว่าผลการลงทะเบียนสำเร็จหรือไม่ (ลำดับชั้นพรีเซนท์เทชัน)

7. เมื่อบันทึกข้อมูลการลงทะเบียนเรียบร้อย ตัวระบบลงทะเบียนจะมีการสร้างเอกสารการชำระเงินเพื่อให้นักศึกษาสามารถนำใบชำระเงินดังกล่าวไปชำระเงินที่ธนาคาร

บทที่ 4
ผลการศึกษา

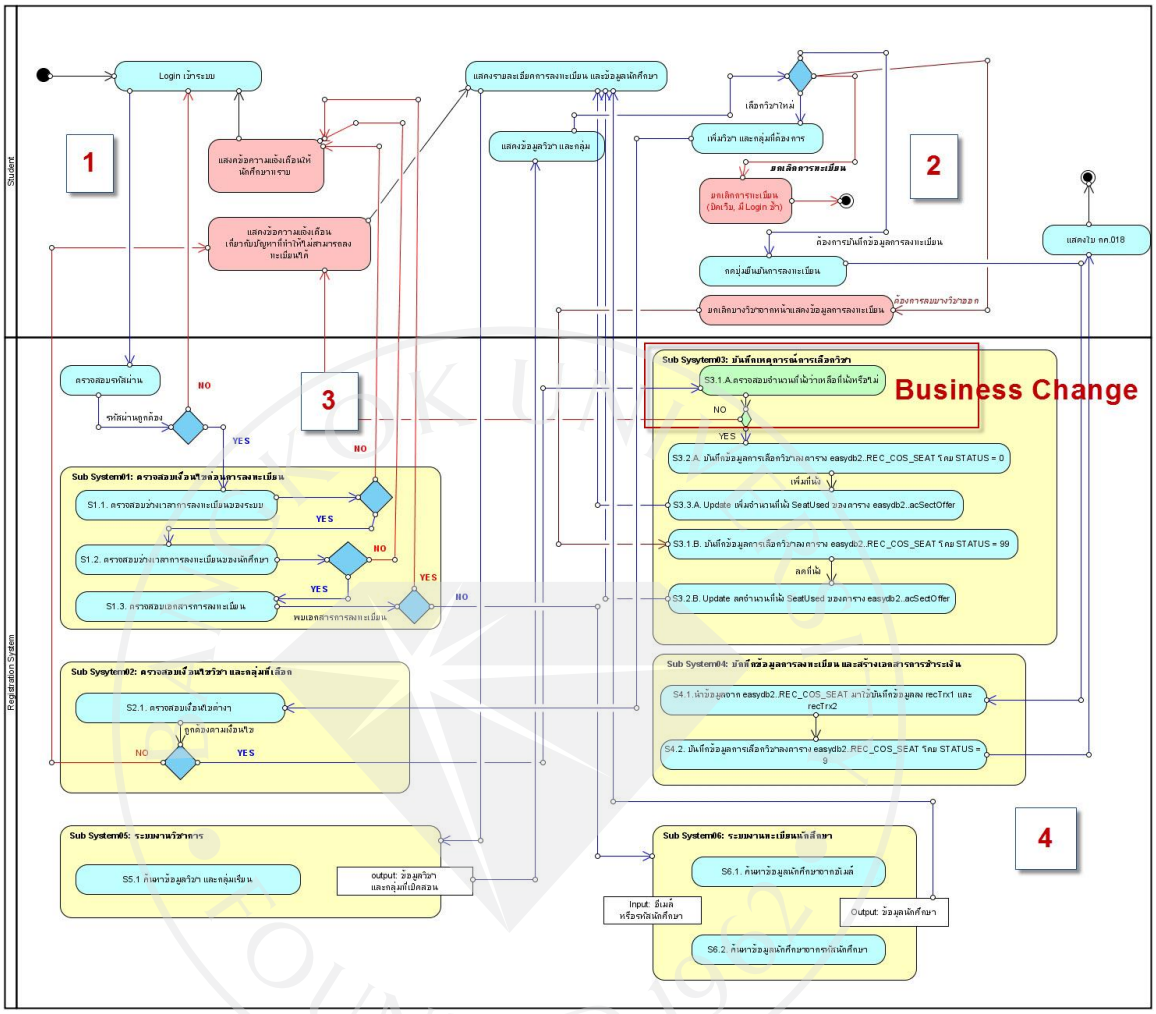
จากการศึกษาการออกแบบและพัฒนาระบบโดยใช้สถาปัตยกรรมแบบลำดับชั้นในการออกแบบและพัฒนาระบบลงทะเบียนเรียนแบบรายวิชามหาวิทยาลัยกรุงเทพ ได้ทำการทดสอบเมื่อระบบลงทะเบียนเรียนแบบรายวิชามีการเปลี่ยนแปลงเงื่อนไขในการลงทะเบียนเรียนคือ ต้องทำการตรวจสอบจำนวนที่นั่งว่ามีจำนวนที่นั่งว่างเหลือในวิชาที่นักศึกษาลงทะเบียนเรียนหรือไม่ ดังนั้นผู้ออกแบบและพัฒนาระบบทำการออกแบบและพัฒนาระบบโดยกำหนดอินเทอร์เฟสในการเชื่อมต่อระหว่างลำดับชั้น คือ เมธอดประเภท พารามิเตอร์ที่ใช้ติดต่อสื่อสารกันระหว่างลำดับชั้นดังภาพที่ 31 อินเทอร์เฟสชื่อ “BU.BLL.Record.IRegistration”

ภาพที่ 31: แสดงคลาสไดอะแกรมเมื่อมีการปรับเปลี่ยนส่วนงานทางธุรกิจ



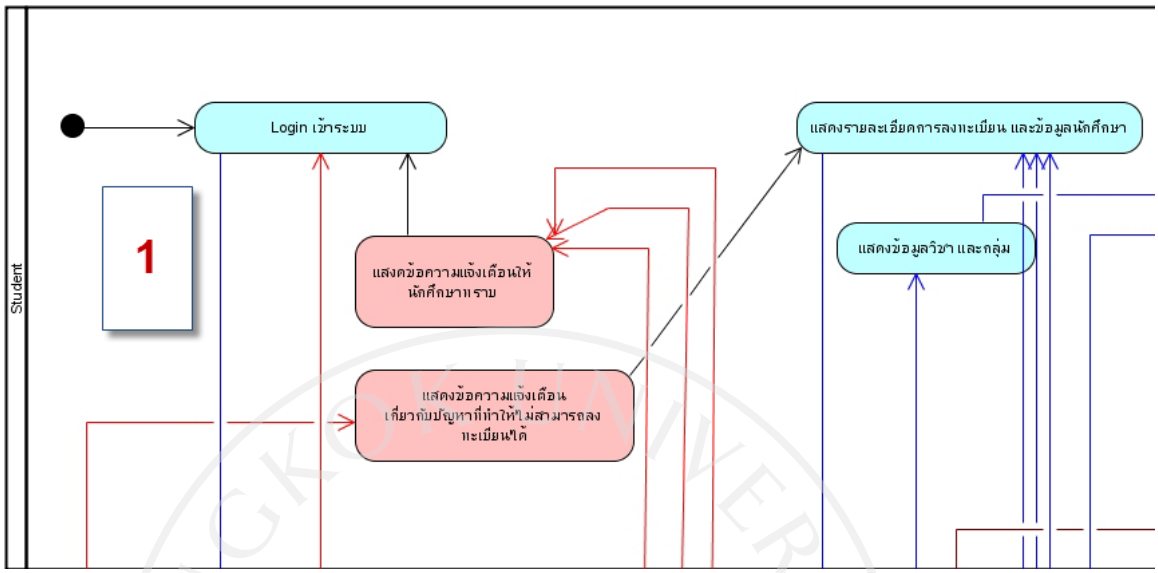
จากภาพที่ 31 เมื่อความต้องการทางธุรกิจเกิดการเปลี่ยนแปลงจากระบบงานเดิมที่มีอยู่แล้วได้ โดยการเพิ่มคลาสเพื่อตอบสนองเงื่อนไขทางธุรกิจแบบใหม่ (“BU.BLL.Record.IRegistration2011”) ถูกพัฒนาภายใต้การพัฒนาผ่านอินเทอร์เฟสของข้อกำหนดเดิมที่มีอยู่ทำให้คลาสที่เพิ่มขึ้นมีหน้าที่
การทำงานครบตามการเรียกใช้งานจากลำดับชั้นพรีเซนท์เทชันอยู่แล้ว เพราะฉะนั้นเมื่อมีการเพิ่มความต้องการทางธุรกิจสามารถทำได้โดยง่ายและสะดวก

ภาพที่ 32: แสดงขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาเมื่อมีการเปลี่ยนแปลงเงื่อนไข

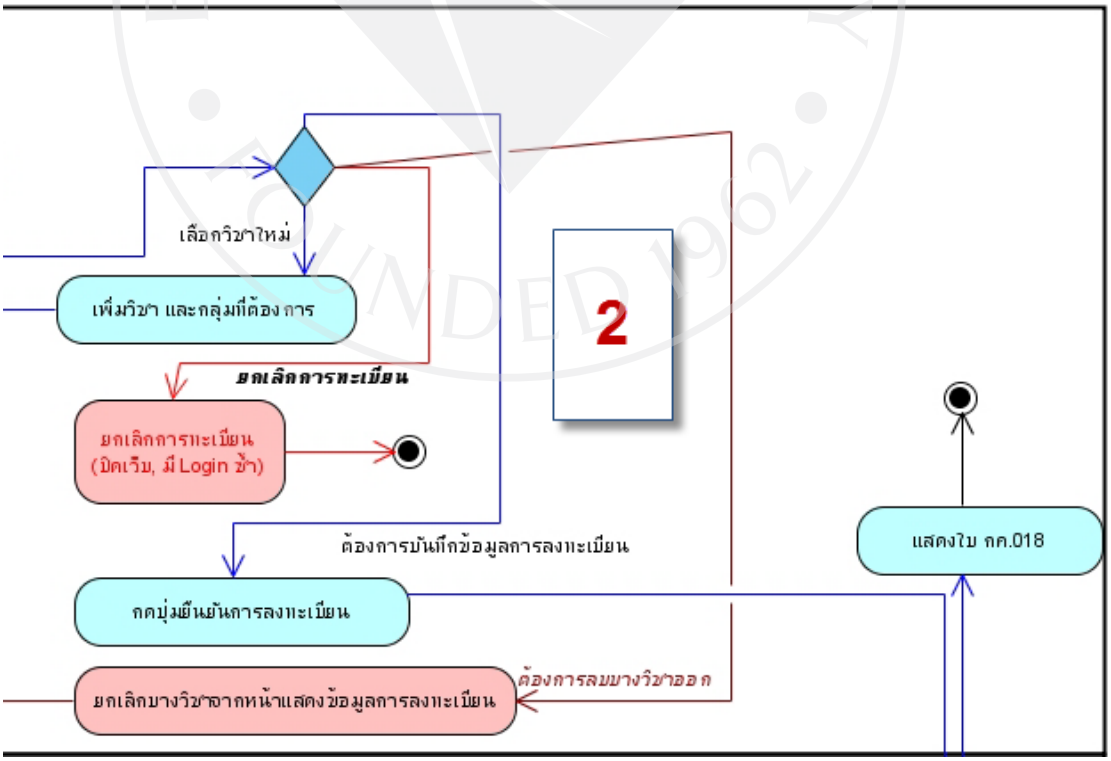


จากภาพที่ 32 ได้แบ่งออกเป็น 4 ส่วนดังภาพที่ 33 – 36 เพื่อแสดงขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาเมื่อมีการเปลี่ยนแปลงเงื่อนไขได้ชัดเจนขึ้น

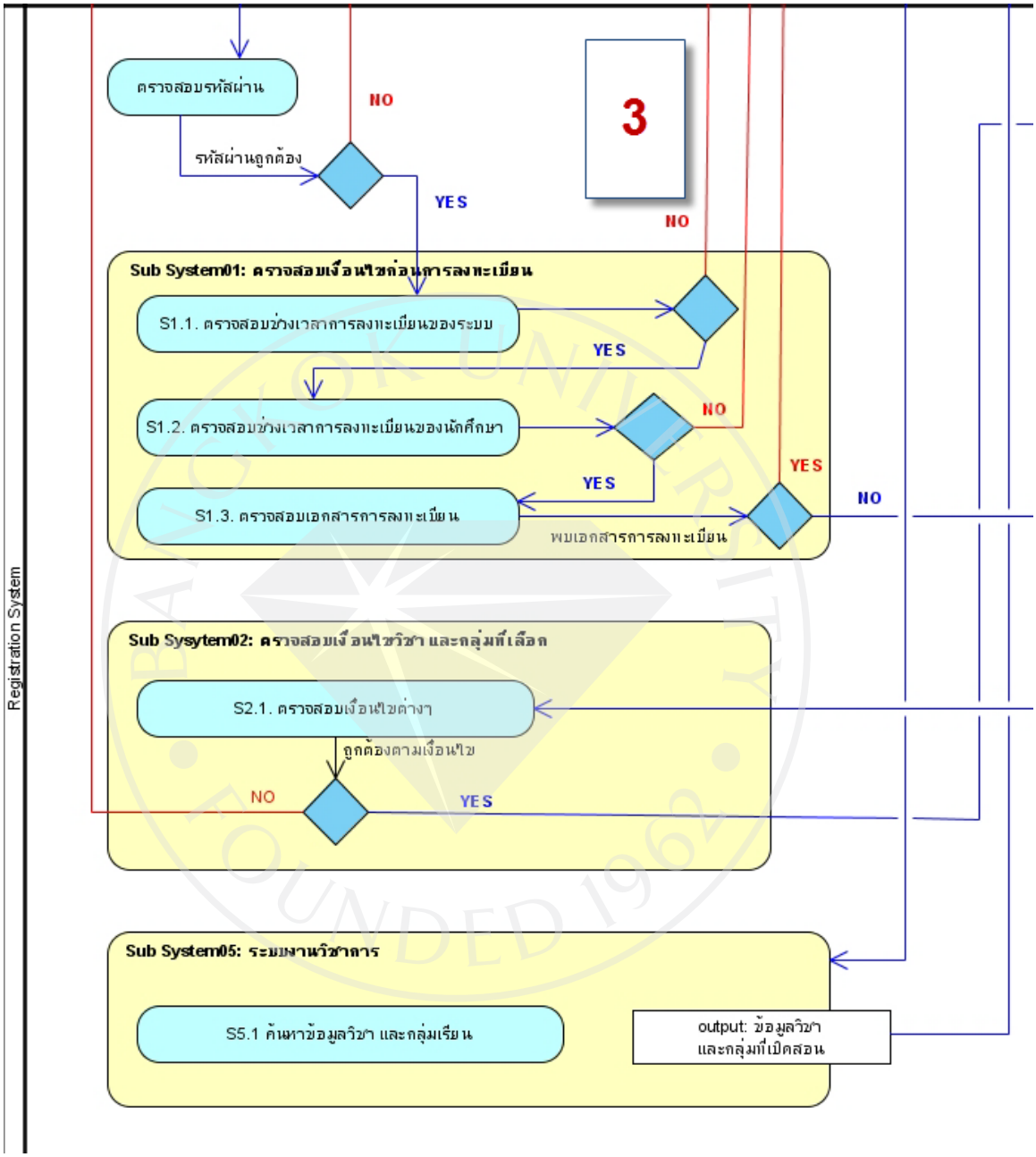
ภาพที่ 33: แสดงขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาเมื่อมีการเปลี่ยนแปลงเงื่อนไขหมายเลข 1



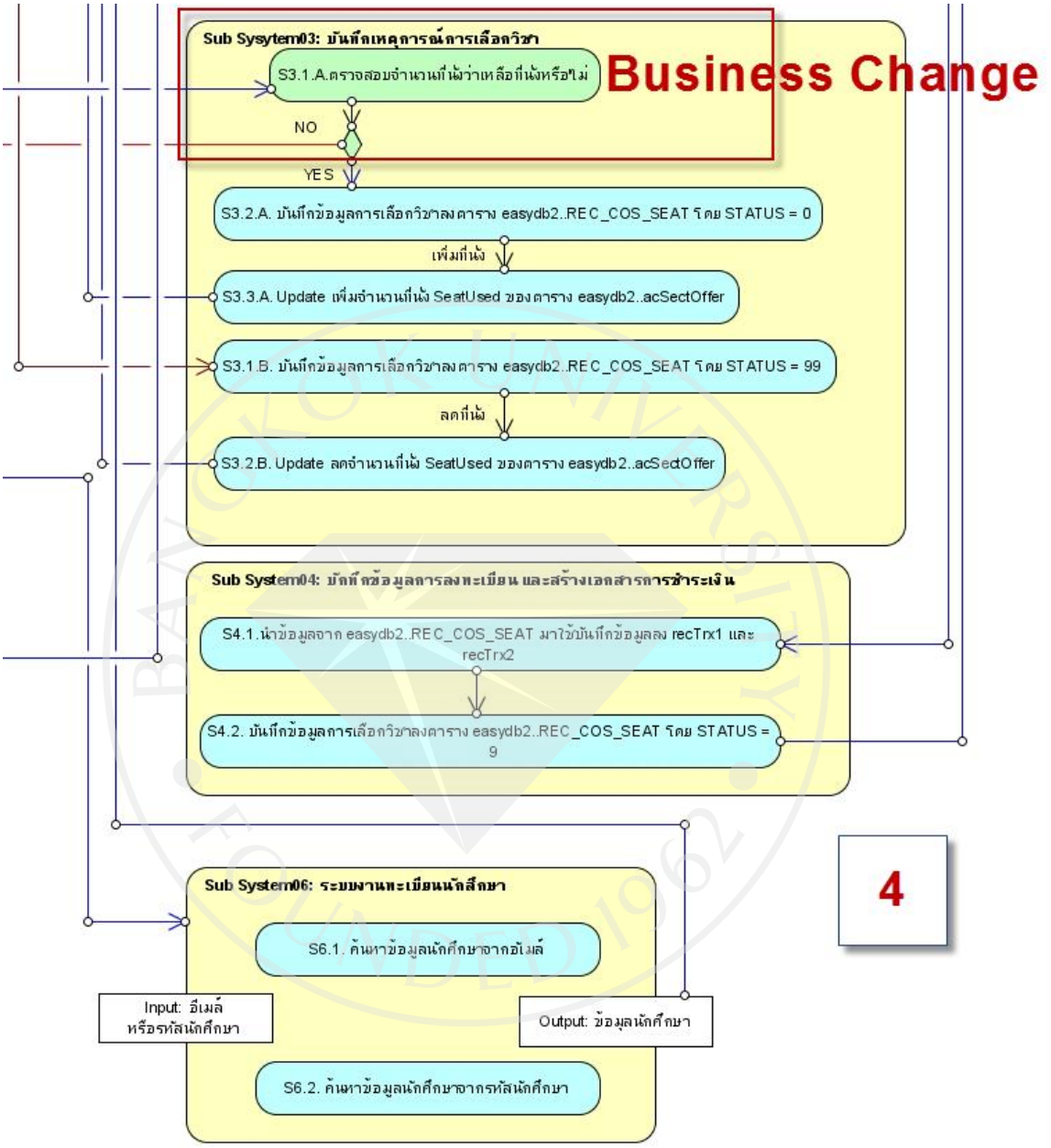
ภาพที่ 34: แสดงขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาเมื่อมีการเปลี่ยนแปลงเงื่อนไขหมายเลข 2



ภาพที่ 35: แสดงขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาเมื่อมีการเปลี่ยนแปลงเงื่อนไขหมายเลข 3

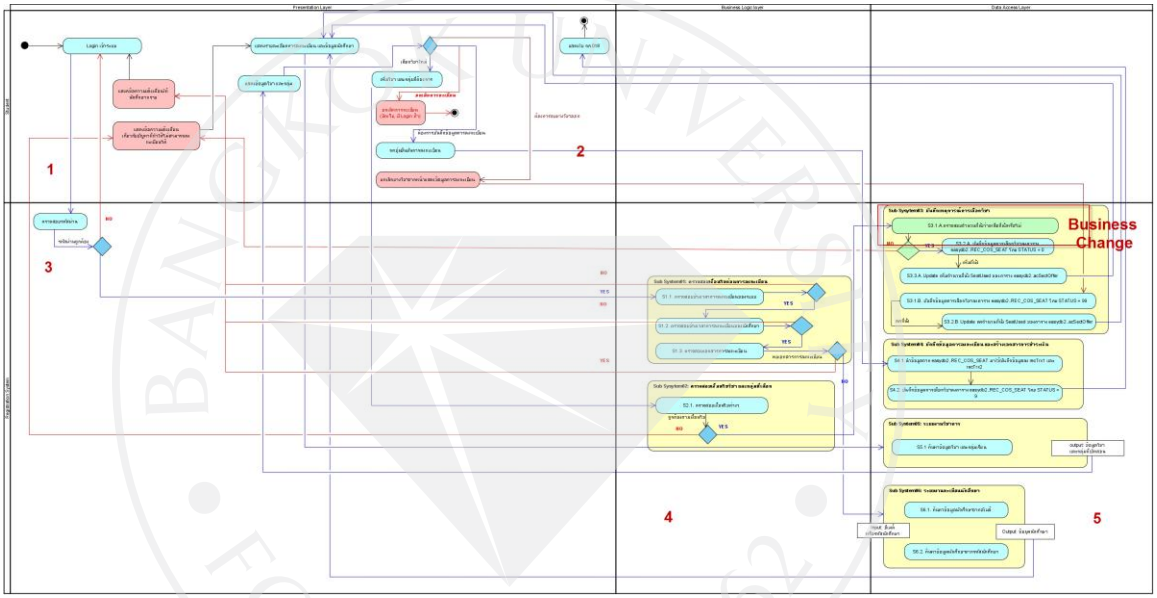


ภาพที่ 36: แสดงขั้นตอนการทำงานการลงทะเบียนเรียนสำหรับนักศึกษาเมื่อมีการเปลี่ยนแปลงเงื่อนไขหมายเลข 4



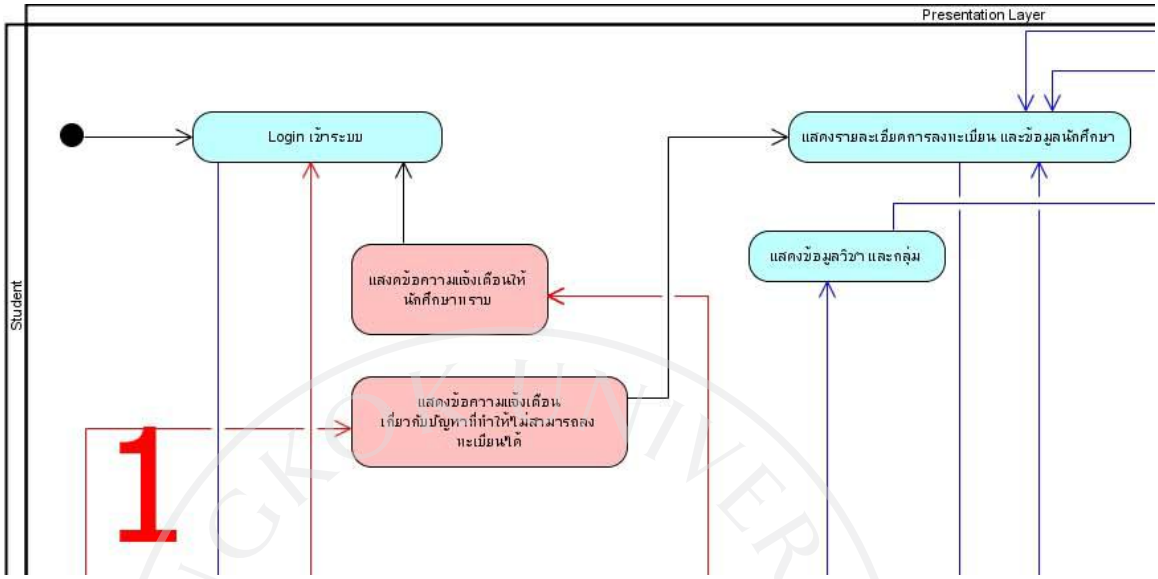
จากภาพที่ 32 แสดงให้เห็นขั้นตอนการทำงานเมื่อมีการเปลี่ยนแปลงความต้องการทางธุรกิจคือ ต้องการให้นักศึกษาได้ที่นั่งในรายวิชาที่นักศึกษาลงทะเบียนเรียนทันทีหลังจากการลงทะเบียนเรียนเสร็จ โดยระบบจะทำการตรวจสอบจำนวนที่นั่งของวิชาที่นักศึกษาเลือกในการลงทะเบียนเรียนก่อน ในกรณีที่จำนวนที่นั่งในวิชาที่นักศึกษาเลือกมีจำนวนที่นั่งไม่เพียงพอระบบจะทำการแจ้งให้นักศึกษาทราบว่าวิชาที่นักศึกษาได้เลือกลงทะเบียนเรียนมีจำนวนที่นั่งไม่เพียงพอ

ภาพที่ 37: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น

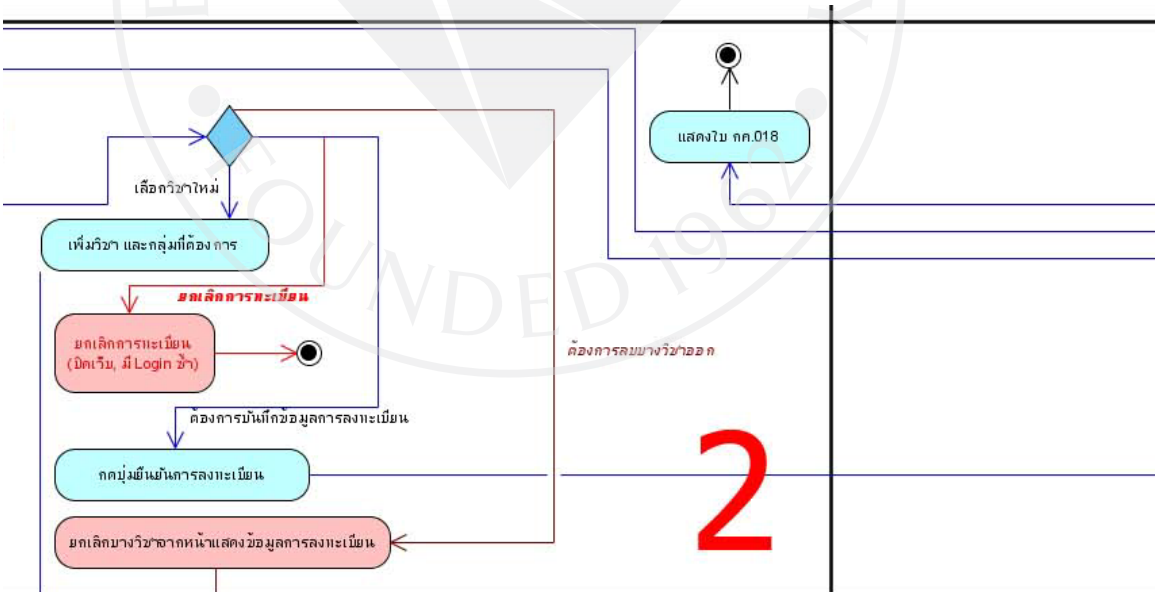


จากภาพที่ 37 ได้แบ่งออกเป็น 5 ส่วนดังภาพที่ 38 – 42 เพื่อแสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น

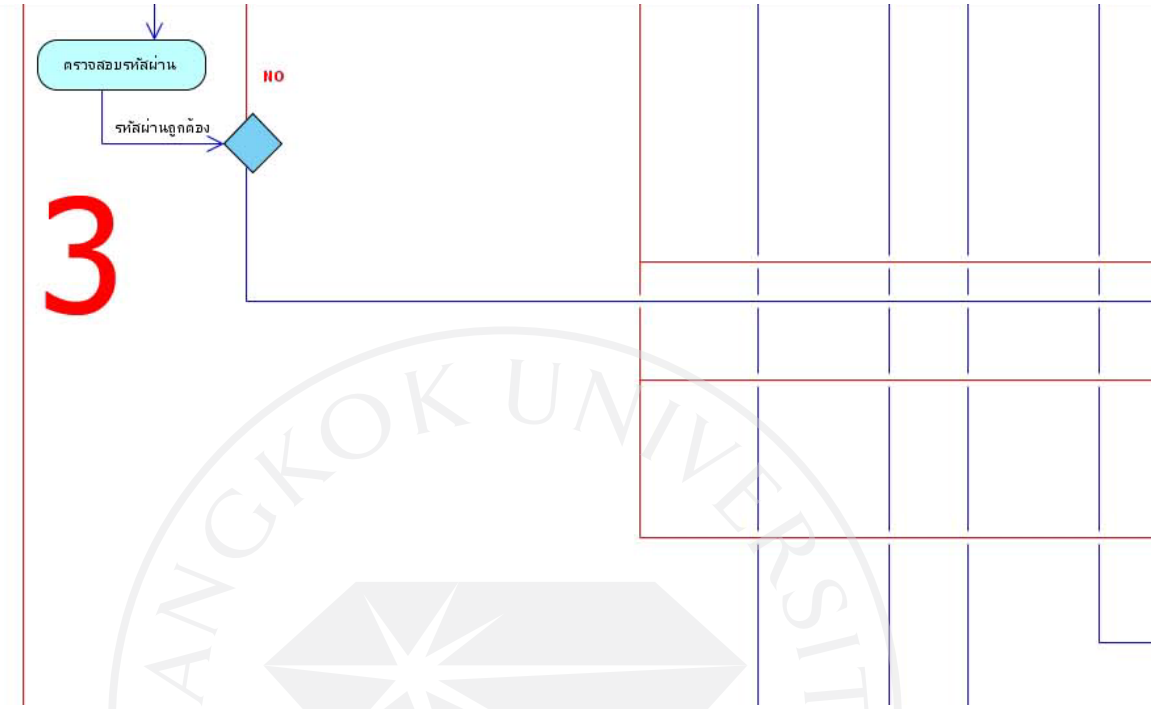
ภาพที่ 38: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 1



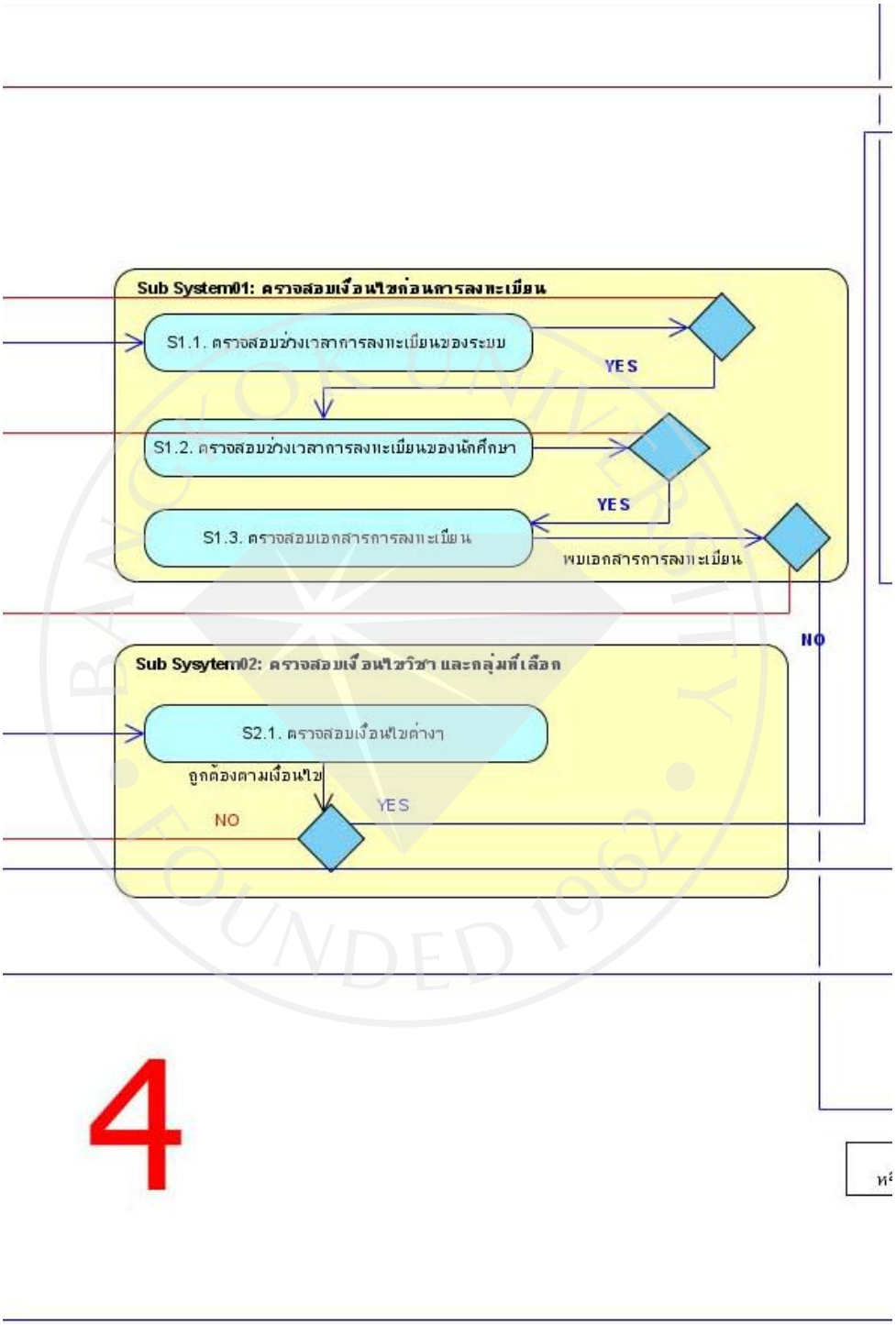
ภาพที่ 39: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 2



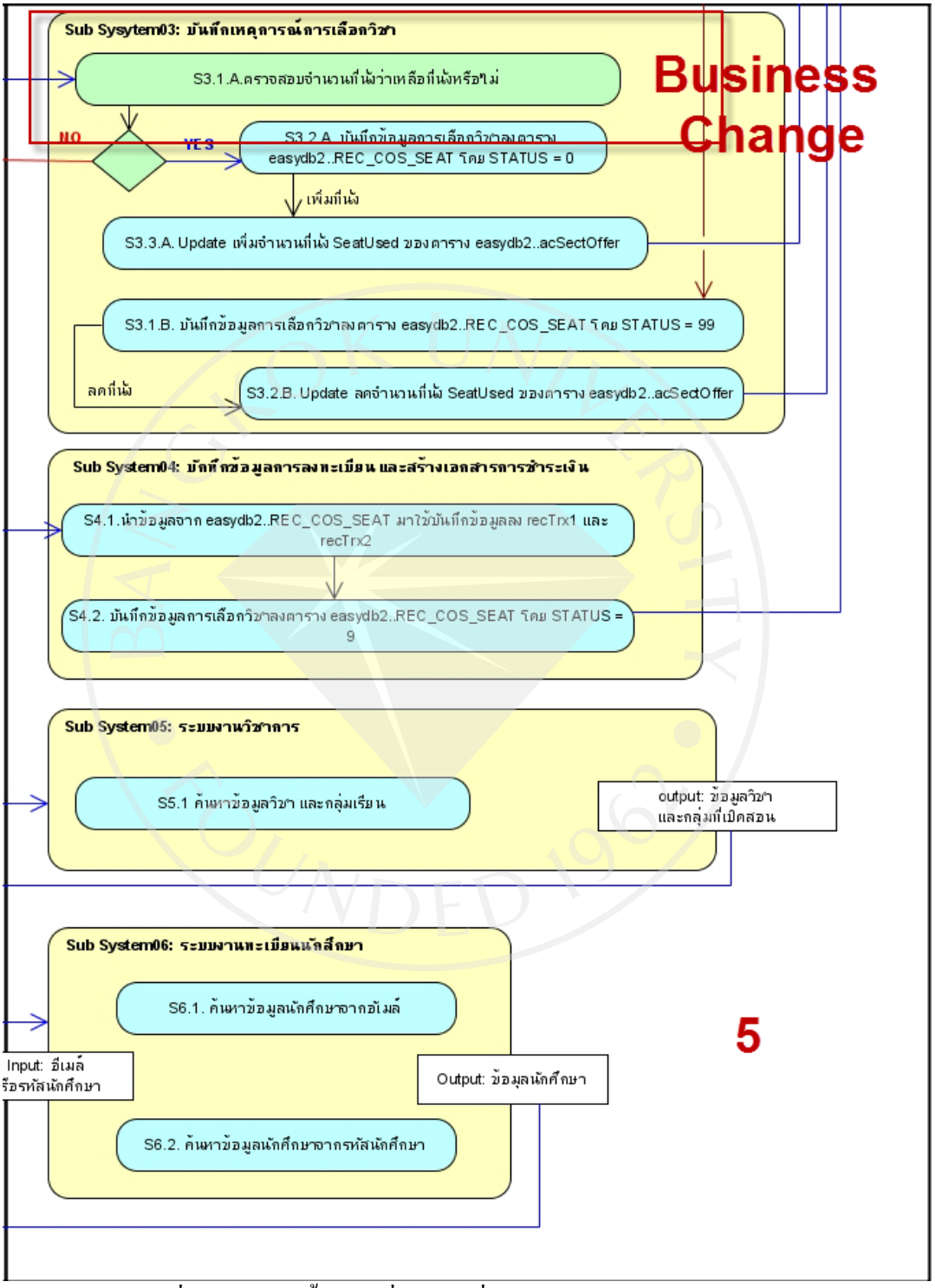
ภาพที่ 40: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 3



ภาพที่ 41: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 4



ภาพที่ 42: แสดงแอคทิวิตี้ไดอะแกรมการลงทะเบียนเรียนเมื่อมีการเปลี่ยนแปลงเงื่อนไขโดยมีการแบ่งการทำงานเป็นลำดับชั้น หมายเลข 5



จากภาพที่ 37 มีการนำขั้นตอนที่มีการเปลี่ยนแปลงความต้องการทางธุรกิจมาแบ่งเป็นลำดับชั้น จะเห็นได้ว่าการปรับปรุงแก้ไขความต้องการของระบบ สามารถระบุส่วนงานที่ต้องการแก้ไขได้

ชัดเจนว่ามีส่วนใดในลำดับชั้นใดที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการ ในภาพที่ 37
จะเห็นว่า เมื่อมีการเพิ่มเงื่อนไขในการตรวจสอบจำนวนที่นั่งก่อนที่จะลงทะเบียนเรียน ลำดับชั้นที่ได้รับผลกระทบจากการเปลี่ยนแปลงความต้องการ คือลำดับชั้นบิสซีเนสและลำดับชั้นดาต้าแอคเซส ผู้ออกแบบและพัฒนาระบบลงทะเบียนสามารถที่จะแก้ไขระบบได้ทันที โดยทำการปรับปรุงเฉพาะส่วนของงานที่ได้รับผลกระทบและในส่วนงานอื่นที่ไม่ได้รับผลกระทบจากเปลี่ยนแปลงสามารที่
จะทำงานได้ตามปกติ

ภาพที่ 43: แสดงโค้ดตัวอย่างในการตรวจสอบวิชาที่ลงทะเบียนเรียน

```
protected override void CheckCourseValidation(BU.Entities.Academic.CourseSection _cs,int _i,
        ref string _errorcode, ref string _errormessage)
    {
        Declare Variable

        // 3. ตรวจสอบวิชาที่ลงว่าเป็นแบบรายวิชาหรือไม่
        // <returns>TRUE = SET / FALSE = NOT SET</returns>
        result = objFaçadeBLL.CheckCourseSectionIsSet(
            regisForm.Semester.Year,
            regisForm.Semester.Code,
            _cs.Course.SectionList[0].ID);

        if (result == false) { // วิชาต้องไม่เป็น set
            // 4. ตรวจสอบวิชาพื้นความรู้ของวิชาที่ลง
            // <returns>TRUE = PASS / FALSE = NOT PASS</returns>
            result = objFaçadeBLL.CheckPrerequisite(
                regisForm.Semester.Year,
                regisForm.Semester.Code, regisForm.Student.ID,
                _cs.Course.Code, _cs.Course.SectionList[0].SectionNo,
                ref _errorcode, ref _errormessage);
                Condition 5-12
        }
    }
```

ภาพที่ 44: แสดงโค้ดตัวอย่างเมื่อมีความต้องการทางธุรกิจเกิดการเปลี่ยนแปลง

```
protected override void CheckCourseValidation(BU.Entities.Academic.CourseSection _cs,int _i,
        ref string _errorcode, ref string _errormessage)
    {
        Declare Variable

        if (_cs.Course.SectionList[0].SeatUsed <
            _cs.Course.SectionList[0].SeatNum) { //ตรวจสอบที่นั่ง
            // 3. string pActYear, string pSemester, string pDgCSectCode
            // <returns>TRUE = SET / FALSE = NOT SET</returns>
            result = objFaçadeBLL.CheckCourseSectionIsSet(
                regisForm.Semester.Year,
                regisForm.Semester.Code,
                _cs.Course.SectionList[0].ID);

            if (result == false) { // วิชาต้องไม่เป็น set
                // 4. ตรวจสอบวิชาพื้นความรู้ของวิชาที่ลง
                // <returns>TRUE = PASS / FALSE = NOT PASS</returns>
                result = objFaçadeBLL.CheckPrerequisite(
                    regisForm.Semester.Year,
                    regisForm.Semester.Code, regisForm.Student.ID,
                    _cs.Course.Code, _cs.Course.SectionList[0].SectionNo,
                    ref _errorcode, ref _errormessage);
                Condition 5-12
            }
        }
        else {
            _errorcode = "-1";
            _errormessage = "ที่นั่งเต็ม (No seat available.)";
        }
    }
```

จากภาพที่ 44 จะมีการตรวจสอบเงื่อนไขในการลงทะเบียนเรียน โดยทำการตรวจสอบว่าวิชาที่ลงทะเบียนต้องไม่เป็นแบบกลุ่ม และตรวจสอบว่าวิชาที่ลงทะเบียนต้องผ่านวิชาพื้นความรู้มาก่อนจึงจะทำการลงทะเบียนเรียนได้ แล้วจึงทำการประมวลผลการลงทะเบียนเรียนเพื่อจัดการที่นั่งให้กับนักศึกษาแต่ละคนหลังจากการลงทะเบียนเรียน เพื่อให้นักศึกษาสามารถพิมพ์ใบชำระเงินเพื่อนำไปชำระค่าลงทะเบียนเรียน แต่ในปีถัดมามีการเพิ่มความต้องการใหม่คือ ต้องการให้นักศึกษาได้ที่นั่งในรายวิชาที่นักศึกษาลงทะเบียนเรียนทันทีหลังจากการลงทะเบียนเรียนเสร็จ จากภาพที่ 44 จะเห็นได้ว่ามีการเพิ่มความต้องการทางธุรกิจโดยทำการตรวจสอบจำนวนที่นั่งว่ามีจำนวนที่นั่งว่างเหลือในวิชาที่นักศึกษาลงทะเบียนเรียนหรือไม่ กรณีที่จำนวนที่นั่งไม่เหลือก็จะแจ้งข้อความให้นักศึกษาทราบว่า

วิชาที่นักศึกษาทำการลงทะเบียนเรียนไม่สามารถลงทะเบียนเรียนได้แต่ถ้ามีจำนวนที่นั่งเหลือในรายวิชาที่ลงทะเบียนเรียนจึงทำการตรวจสอบเงื่อนไขอื่นๆ จนครบทุกเงื่อนไขของการลงทะเบียนเรียน กรณีตรวจสอบผ่านทุกเงื่อนไข จำนวนที่นั่งว่างจะถูกลดลงทันที ซึ่งการเปลี่ยนแปลงที่เกิดขึ้นนั้น นักศึกษาสามารถพิมพ์ใบชำระเงินค่าลงทะเบียนเรียนได้ทันทีโดยไม่ต้องรอการประมวลผลการลงทะเบียนเรียน

ภาพที่ 45: แสดงให้เห็นการนำโมดูลที่พัฒนาไปแล้วนำกลับมาใช้ใหม่

```
public virtual void CheckCourseValidation(
    BU.Entities.Academic.CourseSection _cs,int _i,
    ref string _errorcode,
    ref string _errormessage)
{
    throw new NotImplementedException();
}
```

จากภาพที่ 45 แสดงให้เห็นถึงเมธอดของคลาสแม่ ที่เตรียมกระบวนการตรวจสอบความถูกต้องของวิชาที่ลงทะเบียนเรียนแบบรายวิชาโดยอนุญาตให้คลาสลูกสามารถนำไปเพิ่มเติมรายละเอียดหรือเงื่อนไขในการตรวจสอบการลงทะเบียนเรียน ดังแสดงในภาพที่ 43 และ ภาพที่ 44

หลังจากที่ได้ออกแบบและพัฒนาระบบลงทะเบียนแบบรายวิชาโดยใช้แนวคิดการออกแบบและพัฒนาซอฟต์แวร์แบบลำดับชั้น เมื่อระบบลงทะเบียนแบบรายวิชาเกิดการแก้ไขปรับปรุงระบบเพื่อตอบสนองความต้องการทางธุรกิจที่เปลี่ยนแปลง ช่วยทำให้การออกแบบและพัฒนาระบบงานต่าง ๆ ให้มีความชัดเจนสะดวกรวดเร็วมีข้อกำหนดการพัฒนาที่เด่นชัด และมีความยืดหยุ่นสูงในการแก้ไข เพราะแต่ละลำดับชั้นมีหน้าที่รับผิดชอบที่ชัดเจนและรับผิดชอบเฉพาะอย่างตามที่ผู้ออกแบบได้ออกแบบไว้ และยังสามารถนำโมดูลที่พัฒนาไปแล้วสามารถนำกลับมาใช้ใหม่ได้

บทที่ 5
บทสรุปและข้อเสนอแนะ

สำหรับบทนี้จะกล่าวถึงบทสรุปของการวิจัยและการอภิปรายผล โดยมุ่งเน้นที่การประยุกต์ใช้สถาปัตยกรรมการออกและพัฒนาซอฟต์แวร์แบบลำดับชั้น ซึ่งเนื้อหาในบทนี้จะกล่าวถึงวัตถุประสงค์ สมมติฐานการวิจัย กระบวนการการวิจัย สรุปผลการวิจัย การอภิปรายผลและข้อเสนอแนะการวิจัย โดยมีรายละเอียดดังนี้

5.1. วัตถุประสงค์

5.1.1. ระบบที่ออกแบบและพัฒนาโดยการใช้สถาปัตยกรรมการออกแบบซอฟต์แวร์แบบลำดับชั้น สามารถรองรับการเพิ่มเติม แก้ไข เปลี่ยนแปลง และการดูแลรักษาระบบได้ง่ายขึ้น

5.1.2. เพื่อให้การพัฒนาระบบงานใหม่ สามารถนำโมดูลที่พัฒนาแล้วสามารถนำกลับมาใช้ใหม่ได้

5.2. สมมติฐานการวิจัย

5.2.1. การออกแบบและพัฒนาระบบด้วยแนวคิดแบบลำดับชั้น สามารถตอบสนองการทำงานในปัจจุบันได้ดีกว่าระบบเดิม

5.2.2. การออกแบบและพัฒนาระบบด้วยแนวคิดแบบลำดับชั้น ช่วยให้ระบบรองรับการเพิ่มเติมการทำงานหรือการปรับเปลี่ยนความต้องการของระบบได้เร็วกว่าระบบเดิม

5.3. กระบวนการศึกษา

ผู้ศึกษาได้ดำเนินการโดยนำระบบลงทะเบียนเรียนแบบรายวิชามหาวิทยาลัยกรุงเทพ มาทดสอบ โดยการออกแบบและพัฒนาระบบโดยใช้แนวคิดการออกแบบและพัฒนาระบบโดยใช้แนวคิดลำดับชั้น เครื่องมือที่ใช้ในการศึกษาครั้งนี้ประกอบไปด้วย Microsoft Visual Studio 2005 ภาษา C# ร่วมกับระบบฐานข้อมูล Microsoft SQL SERVER 2008 เพื่อทดสอบระบบหลังจากที่ได้ใช้แนวคิดแบบลำดับชั้น

5.4. สรุปผลการศึกษา

หลังจากที่กระบวนการศึกษาเสร็จสิ้นและได้ทำการประเมินแลสรุปผลการศึกษา โดยมีรายละเอียดดังนี้

5.4.1. ขั้นตอนการวางแผน

ขั้นตอนการวางแผนมีการใช้ระยะเวลามากกว่าที่ได้ประเมินไว้ โดยเฉพาะในส่วนของการศึกษาขั้นตอนการทำงานของระบบลงทะเบียนเรียนแบบรายวิชา และขั้นตอนการในการออกแบบและพัฒนาซอฟต์แวร์โดยใช้แนวคิดแบบลำดับชั้น

5.4.2. ขั้นตอนการวิเคราะห์

ขั้นตอนการวิเคราะห์ เป็นขั้นตอนที่มีการเก็บข้อมูล วิเคราะห์ข้อมูล และพัฒนาแบบจำลอง ในการแบ่งขั้นตอนการทำงานของระบบลงทะเบียนเรียนแบบรายวิชาเป็นส่วนๆ เพื่อง่ายต่อการออกแบบ

5.4.3. ขั้นตอนการออกแบบระบบ

ขั้นตอนการออกแบบ ในส่วนนี้การออกแบบระบบลงทะเบียนเรียนแบบรายวิชาโดยจัดกลุ่มการทำงานและหน้าที่การทำงานของระบบโดยแบ่งเป็นส่วนๆ ตามแต่ละลำดับชั้นดังนี้

- พรีเซนท์เทชันเลเยอร์ (Presentation layer) การทำงานที่ใช้ติดต่อระหว่างผู้ใช้งาน
- บิสซิเนสเลเยอร์ (Business layer) การทำงานที่เกี่ยวข้องกับความต้องการทางธุรกิจของระบบ
- ดาต้าแอคเซสเลเยอร์ (Data Access Layer) การทำงานที่เชื่อมแต่และติดต่อกับแหล่งข้อมูล

5.4.4. ขั้นตอนการพัฒนาระบบ

ขั้นตอนการพัฒนาระบบ ในส่วนของการพัฒนาส่วนประกอบต่างๆ ตามที่ได้ออกแบบไว้ ถึงแม้ว่าจะมีการพัฒนาระบบตามที่ได้ออกแบบไว้ เมื่อมีการนำแต่ละส่วนที่ได้พัฒนามาเชื่อมต่อกันก็ยังมีการกลับไปแก้ไข

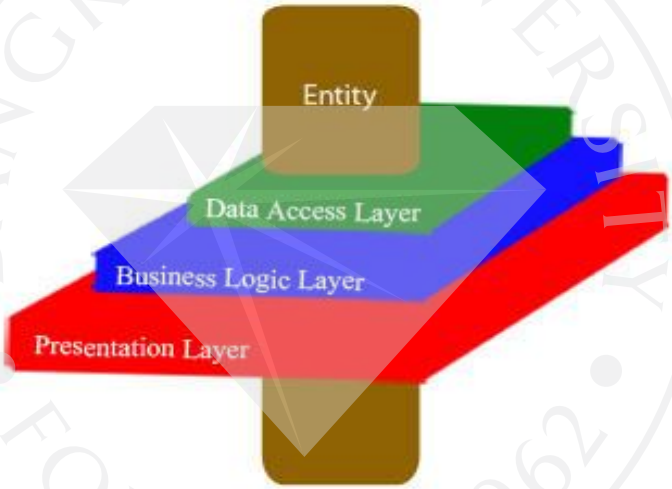
สาเหตุที่เป็นเช่นนั้นเพราะว่าการออกแบบระบบงานในบางส่วนหรือบางขั้นตอนยังแยกลำดับชั้นการทำงานออกจากกันไม่เด็ดขาด บางส่วนของการออกแบบเป็นการทำงานที่คาบเกี่ยวกันระหว่างลำดับชั้น หรือเป็นการทำงานที่ซ้ำซ้อนกัน จึงต้องมีการย้อนกลับไปแก้ไขปัญหาดังกล่าว

เมื่อได้ทำการประกอบส่วนต่างๆ เข้าด้วยกัน เพื่อทำการทดสอบระบบ ก็ยังมีการแก้ไขปรับปรุงส่วนประกอบต่างๆ ตามที่ได้ออกแบบไว้จนแล้วเสร็จ

5.5. การอภิปรายผลและข้อเสนอแนะ

จากข้อสรุปของการศึกษานี้เป็นการทดสอบและเป็นการอธิบายการออกแบบและพัฒนาระบบด้วยแนวคิดการออกแบบและพัฒนาระบบแบบลำดับชั้น เข้ามาช่วยจัดการส่วนต่างของระบบงานที่เกิดขึ้น ทำให้ระบบงานที่ได้ออกแบบและพัฒนาซอฟต์แวร์ด้วยแนวคิดแบบลำดับชั้นสามารถพัฒนางานในแต่ละส่วนไปพร้อมๆ กันและสะดวกต่อการจัดการความต้องการและเงื่อนไขทางธุรกิจขององค์กรที่ระบบจำเป็นต้องทำให้สำเร็จ พร้อมกับอำนวยความสะดวกให้กับผู้พัฒนาสามารถแสดงได้ภาพที่ 46

ภาพที่ 46: แสดงความสัมพันธ์โมเดลโดเมนเอนทิตี้



จากภาพที่ 46 แสดงให้เห็นถึงความสัมพันธ์ของแต่ละลำดับชั้นของการออกแบบและพัฒนาระบบลำดับชั้นดาต้าแอคเซสเป็นส่วนที่มีพื้นที่น้อยที่สุด เนื่องจากการทำงานที่เกิดขึ้นในส่วนนี้จะหน้าที่เฉพาะการเชื่อมต่อแหล่งข้อมูลเท่านั้น ดังนั้น ถ้ามีการการเปลี่ยนแปลงแหล่งข้อมูลการเปลี่ยนแปลงที่เกิดขึ้นก็สามารถทำได้ง่าย ภายใต้เงื่อนไขและข้อกำหนดแบบเดิม ในลำดับชั้นถัดมาเป็นการแสดงให้เห็นส่วนที่ทับซ้อนและไม่ทับซ้อนกันกับลำดับชั้นดาต้าแอคเซสในส่วนที่ทับซ้อนจะมีการยึดเหนี่ยวกันระหว่างโมดูลยังคงมีผลกระทบต่อกันเมื่อมีการแก้ไขหรือเปลี่ยนแปลงในส่วนที่ทับซ้อนแต่ในส่วนที่ไม่ทับซ้อน เมื่อมีการแก้ไขเพิ่มเปลี่ยนแปลงสามารถทำได้ง่ายสะดวกรวดเร็ว ผู้ออกแบบและพัฒนาระบบต้องคำนึงถึงรายละเอียดในส่วนนี้ด้วย ในทางปฏิบัติผู้พัฒนาอาจนำเทคนิค Inversion of Control (IOC) เข้ามาใช้แก้ไขปัญหาเรื่องการยึดเกาะตรงนี้ได้ เนื่องจากเทคนิค IOC เป็นการแยกเอาส่วนของข้อมูลที่เป็นการกำหนดการทำงานต่างๆ ไปเขียนไว้ในส่วนที่

เรียกว่า Container ส่งผลให้สามารถนำโค้ดต่างๆ ที่ได้รับการพัฒนาและใช้งานได้ดี สามารถนำไปใช้กับระบบอื่นๆ ที่ต้องการในภายหลังได้ ในลำดับชั้นถัดมา ลำดับชั้นพรีเซนท์เทชันเป็นส่วนของหน้าจอหรือส่วนที่แสดงข้อความเพื่อเชื่อมต่อกับผู้ใช้งานมีทั้งส่วนที่ทับซ้อนกับและไม่ทับซ้อนกับลำดับชั้นบิสซิเนส ส่วนที่ทับซ้อนยังเกิดผลกระทบเมื่อมีการแก้ไขเปลี่ยนแปลง แต่ในส่วนที่ไม่ทับซ้อนจะสามารถแก้ไขเปลี่ยนแปลงได้ง่ายกว่า จากรูปที่ 46 จะมีโดเมนเอนทิตี้ที่ทะลุผ่านทุกลำดับชั้นเพื่อเป็นส่วนที่ทำให้ทุกลำดับชั้นทำงานภายใต้ขอบเขตข้อมูลเดียวกันได้อย่างถูกต้องตามที่ได้ออกแบบไว้ ผู้ออกแบบและพัฒนาระบบสามารถใช้หลักการของ Data Transfer Object (DTO)เข้ามาช่วยเพื่อทำให้การทำงานสามารถพัฒนาและทำงานได้ง่ายขึ้น

สำหรับการศึกษาครั้งต่อไปสิ่งที่ควรออกแบบและพัฒนาเพิ่มเติมคือการทำให้ลำดับชั้นต่าง ๆ สามารถอยู่อย่างเป็นอิสระบนฮาร์ดแวร์ที่ต่างกัน โดยผู้ออกแบบและพัฒนาอาจมีการใช้เทคนิค Service-Oriented Architecture (SOA) เข้ามาช่วยในการออกแบบและพัฒนาระบบ เนื่องจาก SOA เป็นแนวคิดในการออกแบบระบบให้เป็นระบบเชิงบริการ โดยไม่ยึดติดกับระบบปฏิบัติการโดยใช้ XML ในการติดต่อสื่อสารระหว่างแพลตฟอร์ม

บรรณานุกรม

สื่ออิเลคโทรนิคส์

เฉลิมพล อารีพงศ์. (2553). ASP.NET MVC Series I: Foundation of ASP.NET MVC. สืบค้นวันที่ 10 ธันวาคม 2553 จาก http://nine69.wordpress.com/2010/06/

Books

Arking, J., & Millett, S. (2009). Professional enterprise .Net. Indianapolis, Indiana: Wiley.

Espositi, D., & Saltarello, A. (2008). Architecting microsoft® .Net solutions for the enterprise. Redmond, Washington: Microsoft Press.

Microsoft Patterns, & Practices Team. (2009). Microsoft application architecture guide (patterns & practices). (2nd ed.). Redmond, Washington: Microsoft Press.

Microsoft Corporation. (2003). Enterprise solution patterns using microsoft® .Ner. Redmond, Washington: Microsoft Press.

Journal

Wiggerts, T. (1997). Using clustering algorithms in lagacy systems remodularization. In Proceedings of the Fourth Working Conference on Reverse Engineering ,33-43.

Internet

Dutta, S., & Krishnamoorthy, A. (2010). Patterns & practices application architecute guide layer diagrams. Retrieved by 25 October 2010 from http://visualstudiogallery.msdn.microsoft.com/237f823c-45b4-4f1f-b9e2-607fe66eaae7/

Kilian, A., Mühsig, R., & Guhr, O. (2010). HowTo: 3-tier / 3-Layer architecture. Retrieved by 27 October 2010 from http://code-inside.de/blog-in/2010/10/23/howto-3-tier-3-layer-architecture/

Meier, J.D. (2008). Application architecture Visios now available. Retrieved by 10 December 2010 from http://blogs.msdn.com/b/jmeier/archive/2008/11/24/application-architecture-diagrams-added-to-codeplex.aspx

Microsoft Corporation. (2010). N-Tier data applications overview. Retrieved by 15 October 2010 from http://msdn.microsoft.com: http://msdn.microsoft.com/en-us/library/bb384398.aspx

ประวัติผู้เขียน

ชื่อ-นามสกุล นายวสันต์ มากธนะรุ่ง

อีเมล wasan.m@bu.ac.th

ประวัติการศึกษา วท.บ. (วิทยาศาสตร์)
มหาวิทยาลัยกรุงเทพ, 2544

ประสบการณ์ทำงาน

2544 – ปัจจุบัน อาจารย์ประจำมหาวิทยาลัยกรุงเทพ (นักเขียนวิเคราะห์และออกแบบพัฒนาระบบ)