บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

ในส่วนของบทที่ 2 เป็นการศึกษาเอกสารและงานวิจัยที่เกี่ยวข้อง ซึ่งเนื้อหาในบทนี้ประกอบด้วย หัวข้อทั้งหมดดังนี้

2.1 โรคหลอดเลือดสมองตีบหรืออุดตัน

2.2 การพัฒนาแอพพลิเคชั่นบนระบบปฏิบัติการไอโอเอส (iOS)

2.3 การวิเคราะห์ระบบงานโดยใช้ยูเอ็มแอล (Unified Modeling Language: UML)

2.4 ทฤษฎีเกี่ยวกับระบบฐานข้อมูล

2.1 โรคหลอดเลือดสมองตีบหรืออุดตัน

2.1.1 โรคหลอดเลือดสมอง (Stroke)

โรคหลอดเลือดสมอง เป็นปัญหาสำคัญทางสาธารณสุขที่พบบ่อยในเวชปฏิบัติทั่วไป โดยเป็นสาเหตุของการเสียชีวิตอันดับ 1 และอันดับ 3 ในหญิงไทยและชายไทยทุกอายุ อีกทั้งยังทำให้เกิดความพิการทุพพลภาพได้สูง การป้องกัน การวินิจฉัย การสืบค้นทางห้องปฏิบัติการ ตลอดจนการรักษาที่ถูกต้องเหมาะสม รวมทั้งการให้ความถูกต้อง ทันสมัย แก่บุคลากรทางการแพทย์และประชาชนทั่วไป นับได้ว่าเป็นส่วนสำคัญอย่างยิ่งในการลดอัตราการป่วย ความพิการ การเสียชีวิต ตลอดจนลดการสูญเสียทางเศรษฐกิจทั้งในระดับครอบครัว ชุมชน และระดับประเทศ ในปัจจุบันความรู้และวิทยาการต่าง ๆ เกี่ยวกับการรักษาโรคหลอดเลือดสมอง ได้พัฒนาก้าวไปเป็นอย่างมาก มีทั้งการให้ยาละลายลิ่มเลือด (Recombinant-tissue Plasminogen Activator) ทางหลอดเลือดดำในผู้ป่วยที่มีอาการภายใน 4.5 ชั่วโมง การให้ยาละลายลิ่ม เลือดทางหลอดเลือดแดง หรือการใช้เครื่องมือนำเอาลิ่มเลือดออกทางหลอดเลือดแดงในผู้ป่วยบาง รายการให้ยาต้านเกร็ดเลือด การรักษาภาวะความดันในกะโหลกศีรษะสูง การรักษาทั่วไปด้านอื่น ๆ เพื่อ ป้องกันภาวะแทรกซ้อน รวมทั้งการทำเวชศาสตร์ฟื้นฟู ซึ่งการรักษาแบบองค์รวมที่เป็นไปตามแนวทางการรักษาโรคหลอดเลือดสมองขาดเลือดเฉียบพลันอย่างสากลนี้ จะมีผลทำให้ผู้ป่วยกลุ่มนี้ มีผลการรักษาที่ดี สามารถกลับไปช่วยเหลือตัวเองได้มากขึ้นและลดภาวะทุพพลภาพที่เกิดขึ้นได้ (ดิษยา รัตนากร, 2556: 6)

โรคหลอดเลือดสมอง (Stroke) คือ ภาวะที่สมองขาดเลือดไปเลี้ยงเนื่องจากหลอดเลือดตีบ หลอดเลือดอุดตัน หรือหลอดเลือดแตก ส่งผลให้เนื้อเยื่อในสมองถูกทำลาย การทำงานของสมองหยุดชะงัก (โรงพยาบาลบำรุงราษฎร์, 2558) ความผิดปกติของหลอดเลือดสมองที่ทำให้สมองขาดเลือด แบ่งได้เป็น 2 ประเภท ดังนี้

2.1.2.1 หลอดเลือดสมองตีบหรืออุดตัน (Ischemic Stroke) เป็นสาเหตุส่วนใหญ่ที่ทำให้เกิดโรคหลอดเลือดสมอง พบได้ประมาณ 80% หลอดเลือดสมองอุดตัน เกิดได้จากลิ่มเลือดที่เกิดขึ้นในบริเวณอื่น ไหลไปตามกระแสเลือดจนไปอุดตันที่หลอดเลือดสมอง หรืออาจเกิดจากมีลิ่มเลือดก่อตัวในหลอดเลือดสมอง และขยายขนาดใหญ่ขึ้นจนอุดตันหลอดเลือดสมอง ส่วนสาเหตุของหลอดเลือดสมองตีบอาจเกิดจากการสะสมของไขมันในหลอดเลือด ทำให้หลอดเลือดตีบแคบ มีความยืดหยุ่นและมีประสิทธิภาพในการลำเลียงเลือดลดลง

2.1.2.2 หลอดเลือดสมองปริแตกหรือฉีกขาด (Hemorrhagic Stroke) พบได้ประมาณ 20% ของโรคหลอดเลือดสมอง เกิดจากหลอดเลือดมีความเปราะบางร่วมกับภาวะความดันโลหิตสูง ทำให้บริเวณที่เปราะบางนั้นโป่งพองและแตกออก หรืออาจเกิดจากหลอดเลือดเสียความยืดหยุ่นจากการสะสมของไขมันในหลอดเลือด ทำให้หลอดเลือดปริแตกได้ง่าย ซึ่งอันตรายมากเนื่องจากทำให้ปริมาณเลือดที่ไป เลี้ยงสมองลดลงอย่างฉับพลันและทำให้เกิดเลือดออกในสมอง ส่งผลให้ผู้ป่วยเสียชีวิตในเวลาอันรวดเร็วได้

2.1.3 ปัจจัยเสี่ยงของโรคหลอดเลือดสมอง

โรงพยาบาลบำรุงราษฎร์ (2558) ปัจจัยเสี่ยงของโรคหลอดเลือดสมองมีหลายสาเหตุ แบ่งเป็น ปัจจัยเสี่ยงที่ป้องกันไม่ได้ และปัจจัยเสี่ยงที่ป้องกันได้ ซึ่งปัจจัยเสี่ยงที่ป้องกันได้มักมีสาเหตุจากสุขภาพโดยรวมและรูปแบบการดำเนินชีวิต

โรคหลอดเลือดสมองตีบ หรือเรียกในชื่ออื่น คือโรคสมองขาดเลือด และโรคอัมพฤกษ์อัมพาต โดยเป็นโรคที่เกิดจากภาวะสมองขาดเลือดและออกซิเจนสำหรับหล่อเลี้ยงเนื้อเยื่อสมอง ที่มีสาเหตุมาจากภาวะหลอดเลือดในสมองตีบตันหรือมีการอุดตันของไขมันในเส้นเลือด รวมไปถึงการฝังตัวของลิ่มเลือดที่มีการแข็งตัวในหลอดเลือด ทำให้สมองขาดเลือดและออกซิเจนมีผลทำให้ร่างกายเป็นอัมพฤกษ์ อัมพาต ตามมา ซึ่งสาเหตุนี้เป็นสาเหตุที่พบมากที่สุดในบรรดาผู้ป่วยที่เป็นอัมพฤกษ์อัมพาต

2.1.3.1 ปัจจัยเสี่ยงที่ป้องกันไม่ได้

อายุ เมื่ออายุมากขึ้น หลอดเลือดก็จะเสื่อมตามไปด้วย โดยผิวชั้นในของหลอดเลือด จะหนาและแข็งขึ้นจากการที่มีไขมันและหินปูนมาเกาะ รูที่เลือดไหลผ่านจะแคบลงเรื่อยๆ

เพศ พบว่าเพศชายมีความเสี่ยงต่อโรคหลอดเลือดสมองสูงกว่าเพศหญิง

ภาวะการแข็งตัวของเลือดเร็วกว่าปกติ ส่งผลให้เกิดการจับตัวกันของเม็ดเลือดและ มีลิ่มเลือดเกิดขึ้นได้ง่ายกว่าคนปกติ

2.1.3.2 ปัจจัยเสี่ยงที่ป้องกันได้

- ความดันโลหิตสูง เป็นปัจจัยเสี่ยงที่สำคัญที่สุดของโรคหลอดเลือดสมอง ผู้ที่มีภาวะ ความดันโลหิตสูงจึงมีโอกาสเป็นโรคหลอดเลือดสมองได้มากกว่าคนปกติ

- เบาหวาน เป็นสาเหตุที่ทำให้หลอดเลือดแข็งทั่วร่างกาย หากเกิดที่สมองจะมีโอกาสเป็นโรคหลอดเลือดสมองมากกว่าคนปกติ 2-3 เท่า

- ไขมันในเลือดสูง เป็นความเสี่ยงของโรคหลอดเลือดสมองเช่นเดียวกับโรคหลอด เลือดหัวใจ คือภาวะไขมันสะสมอยู่ตามผนังหลอดเลือด ทำให้กีดขวางการลำเลียงเลือด

- โรคหัวใจ เช่น โรคลิ้นหัวใจผิดปกติ หัวใจเต้นผิดจังหวะ เป็นสาเหตุของการเกิดลิ่มเลือด ถ้าลิ่มเลือดไปอุดตันที่หลอดเลือดสมอง ก็จะทำให้สมองขาดเลือดได้

- การสูบบุหรี่ สารนิโคตินและคาร์บอนมอนอกไซด์ทำให้ปริมาณออกซิเจนลดลง และเป็นตัวทำลายผนังหลอดเลือดทำให้หลอดเลือดแข็งตัว พบว่าการสูบบุหรี่เพียงอย่างเดียวเพิ่มความ เสี่ยงต่อโรคหลอดเลือดสมองถึง 3.5%

- ยาคุมกำเนิด ในผู้หญิงที่ใช้ยาคุมกำเนิดที่มีฮอร์โมนเอสโตรเจนสูงจะมีความเสี่ยงต่อการเกิดโรคหลอดเลือดสมองสูง

- โรคซิฟิลิส เป็นสาเหตุของหลอดเลือดอักเสบและหลอดเลือดแข็ง

- การขาดการออกกำลังกาย

2.1.4 อาการของโรคหลอดเลือดสมอง

โรงพยาบาลบำรุงราษฎร์ (2558) เมื่อสมองขาดเลือดจะทำให้สมองไม่สามารถทำงานได้ตามปกติ ซึ่งอาการแสดงต่าง ๆ จะมากหรือน้อยขึ้นกับระดับความรุนแรงและตำแหน่งของสมองที่ถูกทำลาย เช่น

- ชาหรืออ่อนแรงที่ใบหน้าและ/หรือบริเวณแขนขาครึ่งซีกของร่างกาย
- พูดไม่ชัด ปากเบี้ยว มุมปากตก ฝ้าลายไหล กลืนลำบาก
- ปวดศีรษะ เวียนศีรษะทันทีทันใด
- ตามัว มองเห็นภาพซ้อนหรือเห็นครึ่งซีก หรือตาบอดข้างเดียวทันทีทันใด
- เดินเซ ทรงตัวลำบาก

อาการเหล่านี้มักเกิดขึ้นอย่างฉับพลัน ในรายที่มีภาวะสมองขาดเลือดแบบชั่วคราว (Transient Ischemic Attack: TIA) อาจมีอาการเตือนเหล่านี้เกิดขึ้นชั่วขณะแล้วหายไปเอง หรืออาจเกิดขึ้นได้หลายครั้ง ก่อนจะมีอาการสมองขาดเลือดแบบถาวร ดังนั้นหากมีอาการผิดปกติเกิดขึ้น ควรรีบพบแพทย์ทันที เนื่องจากอาการของโรคหลอดเลือดสมองจัดเป็นอาการร้ายแรงและอาจเป็นอันตรายถึงแก่ชีวิต หรือหากไม่ถึงชีวิต ก็อาจทำให้กลายเป็นโรคอัมพาต อัมพฤกษ์ ไม่สามารถช่วยเหลือตนเองและต้องใช้เวลาในการรักษาฟื้นฟูสุขภาพต่อไป

2.1.5 การตรวจวินิจฉัยโรคหลอดเลือดสมอง

โรงพยาบาลบำรุงราษฎร์ (2558) ในปัจจุบันมีวิธีการตรวจวินิจฉัยที่มีประสิทธิภาพและ สามารถบ่งชี้ถึงตำแหน่งของสมองและหลอดเลือดที่ผิดปกติ รวมถึงภาวะและสาเหตุที่เป็นปัจจัยเสี่ยงของ การเป็นโรคหลอดเลือดสมองได้ เช่น

2.1.5.1 การตรวจเลือดเพื่อดูความเข้มข้นและความสมบูรณ์ของเม็ดเลือด

2.1.5.2 การตรวจระดับน้ำตาลและระดับไขมันในเลือด

2.1.5.3 การตรวจหาการอักเสบของหลอดเลือด

2.1.5.4 การตรวจคลื่นไฟฟ้าหัวใจ (Electrocardiogram) เพื่อดูจังหวะการเต้นของหัวใจที่ ผิดปกติ

2.1.5.5 การตรวจสมองด้วยเครื่องเอกซเรย์คอมพิวเตอร์ (Computerized Tomography) เพื่อดูว่าสมองมีภาวะขาดเลือดหรือภาวะเลือดออกในสมองหรือไม่

2.1.5.6 การตรวจอัลตราซาวนด์หลอดเลือดบริเวณคอ (Carotid Duplex Scan) เพื่อดูขนาด และการไหลเวียนของหลอดเลือดแดงบริเวณคอที่ไปเลี้ยงสมองด้วยคลื่นความถี่สูง

2.1.5.7 การตรวจสมองด้วยคลื่นแม่เหล็กไฟฟ้า (Magnetic Resonance Imaging) เพื่อดูเนื้อสมอง หลอดเลือดสมอง หลอดเลือดที่คอ เป็นวิธีการที่ไม่เจ็บปวดและมีประสิทธิภาพสูง

2.1.6 การรักษา

โรงพยาบาลบำรุงราษฎร์ (2558) การรักษาขึ้นกับสาเหตุของโรคหลอดเลือดสมอง ว่าเป็นหลอดเลือดสมองตีบหรือหลอดเลือดสมองแตก โดยจะมีแนวทางการรักษาที่แตกต่างกัน

2.1.6.1 หลอดเลือดสมองตีบหรืออุดตัน เป้าหมายของการรักษาคือทำให้เลือดไหลเวียนได้อย่างปกติ โดยทางเลือกในการรักษามีหลายวิธี ในบางกรณีแพทย์อาจให้ยาละลายลิ่มเลือด ซึ่งพบว่าจะได้ผลดีกับผู้ที่มีอาการและอาการแสดงของโรคหลอดเลือดสมองและรีบมาโรงพยาบาลภายในระยะเวลา ไม่เกิน 4.5 ชั่วโมง

2.1.6.2 หลอดเลือดสมองปริแตกหรือฉีกขาด เป้าหมายของการรักษาคือการควบคุมปริมาณเลือดที่ออกด้วยการรักษาระดับความดันโลหิต ในกรณีที่เลือดออกมาก แพทย์อาจพิจารณาทำการผ่าตัด เพื่อป้องกันความเสียหายต่อสมองที่อาจเกิดขึ้นหากมีการเปลี่ยนแปลงความดันโลหิต

2.1.7 การป้องกันการกลับเป็นซ้ำ

โรงพยาบาลบำรุงราษฎร์ (2558) การป้องกันเป็นการรักษาโรคหลอดเลือดสมองที่ดีที่สุด และควรป้องกันก่อนการเกิดโรคหลอดเลือดสมอง คือ ต้องควบคุมปัจจัยเสี่ยงที่ส่งเสริมให้หลอดเลือดเกิดการ ตีบ อุดตัน หรือแตก เช่น ความดันโลหิตสูง โรคหัวใจ โรคเบาหวานไขมันในเลือดสูง การสูบบุหรี่ หรือขาด การออกกำลังกาย เป็นต้น

2.1.7.1 ตรวจเช็คสุขภาพประจำปีเพื่อค้นหาปัจจัยเสี่ยง ถ้าพบต้องรีบรักษาและพบแพทย์ อย่างสม่ำเสมอ

2.1.7.2 ในกรณีที่พบว่ามีปัจจัยเสี่ยงที่ทำให้หลอดเลือดตีบ อุดตัน หรือแตก ต้องรักษาและ รับประทานยาอย่างสม่ำเสมอตามแผนการรักษาของแพทย์ ห้ามหยุดยาเอง และควรรีบพบแพทย์ทันทีถ้า มีอาการผิดปกติ

2.1.7.3 ควบคุมระดับความดันโลหิต ไขมัน และน้ำตาลในเลือดให้อยู่ในเกณฑ์ปกติ

2.1.7.4 ควบคุมอาหารให้สมดุล หลีกเลี่ยงอาหารรสเค็ม หวาน มัน

2.1.7.5 ออกกำลังกายสม่ำเสมอ อย่างน้อย 30 นาทีต่อวัน 3 ครั้งต่อสัปดาห์

2.1.7.6 งดสูบบุหรี่ หลีกเลี่ยงเครื่องดื่มแอลกอฮอล์

2.1.7.7 ถ้ามีอาการเตือนที่แสดงว่าเลือดไปเลี้ยงสมองไม่พอชั่วคราว ควรรีบมาพบแพทย์ถึงแม้ว่าอาการเหล่านั้นจะหายได้เองเป็นปกติ

2.1.7.8 ผู้ที่เป็นหลอดเลือดสมองตีบหรืออุดตันแล้ว แพทย์จะให้การรักษาโดยใช้ยาเพื่อป้องกันการกลับเป็นซ้ำของโรคหลอดเลือดสมอง แต่การใช้ยาเหล่านี้จำเป็นต้องมีการติดตามผลและใช้ภายใต้ คำแนะนำของแพทย์อย่างเคร่งครัด เนื่องจากถ้ามีการใช้ยาผิด ประมาทเลินเล่อ หรือไม่มีการติดตามดูแล อย่างสม่ำเสมออาจเกิดภาวะแทรกซ้อนอย่างรุนแรง เป็นอันตรายถึงแก่ชีวิตได้

2.1.8 การประเมินความรุนแรงของโรคหลอดเลือดสมอง

ข้อมูลจากคณะพยาบาลศาสตร์ มหาวิทยาลัยมหิดล (2555) ระบุว่าการตรวจทางระบบประสาทที่มักเห็นใช้กันบ่อยๆ ก็คือการประเมินโดยใช้ Glasgow Coma Scale (GCS) แต่ปัจจุบันมีอีกวิธี เพิ่มเติมขึ้นมาคือ National Institute of Health Stroke Scale (NIHSS) ซึ่งการประเมินทั้งสองแบบมีความคล้ายคลึงกันและแตกต่างกันในบางประเด็น เช่น ทั้งสองวิธีมีการประเมิน Consciousness, Motor และ Perception เหมือนกัน แต่ NIHSS จะมีการประเมินในด้านของ Cognition ร่วมด้วย นอกจากนี้ จุดประสงค์ของการพัฒนาแบบประเมินทั้งสองก็ยังมีความต่างกันคือ NIHSS พัฒนาขึ้นมาเพื่อประเมิน ความรุนแรงของโรคหลอดเลือดสมอง ส่วน GCS พัฒนาขึ้นมาเพื่อประเมินผู้ป่วยที่ได้รับบาดเจ็บทางสมอง ดังนั้นในการประเมินผู้ป่วยที่มีปัญหาทางหลอดเลือดสมอง การประเมินโดยใช้ NIHSS จึงเป็นที่นิยมใช้กัน และแบบประเมินมีความเที่ยงอยู่ในระดับสูง ซึ่ง NIHSS จะมีการประเมินด้านต่าง ๆ ดังนี้

1a. ระดับความรู้สึกตัว (Level of Consciousness, LOC) โดยมีคะแนน 0-3 ดังนี้

0 = รู้สึกตัวดี ตอบสนองเป็นปกติ

1 = ง่วงซึม ปลุกตื่นได้ง่าย เมื่อตื่นถามตอบรู้เรื่อง และสามารถทำตามคำสั่งได้

2 = หลับตลอดเวลา ปลุกตื่นได้แต่ต้องใช้ตัวกระตุ้นแรงๆซ้ำๆกันหลายครั้งหรืออาจจำเป็นต้องใช้สิ่งเร้าที่ทำให้เกิดความเจ็บปวด

3 = ไม่ตอบสนอง แต่สามารถตรวจพบปฏิกิริยาอัตโนมัติ (Reflex) ได้

ระดับความรู้สึกตัว (Level of Consciousness, LOC) ผู้ตรวจจะต้องให้คะแนนเสมอ แม้ว่าจะทำการตรวจในกรณีที่มีข้อจำกัดที่ไม่สามารถประเมินได้อย่างครบถ้วน เช่นในผู้ป่วยที่ได้รับการใส่ท่อช่วย หายใจ หรือมีภยันตรายที่เกิดขึ้นกับหลอดลม หรือผู้ป่วยที่ใช้ภาษาอื่น จะให้คะแนน “3” เฉพาะกับผู้ที่ไม่ ตอบสนองต่อความเจ็บปวดเลย โดยในผู้ป่วยกลุ่มนี้อาจพบเพียงปฏิกิริยาการตอบสนองอัตโนมัติ เท่านั้น

1b. สามารถบอกเดือน และอายุได้ (LOC Questions) โดยมีคะแนน 0-2 ดังนี้

0 = ตอบได้ถูกต้องทั้ง 2 ข้อ

1 = ตอบได้ถูกเพียง 1 ข้อ

2 = ไม่สามารถตอบคำถามได้หรือตอบผิดทั้ง 2 ข้อ

ระดับความรู้สึกตัว (LOC Questions) ให้ถามชื่อเดือนและอายุในขณะปัจจุบันของผู้ป่วย ผู้ตรวจจะให้คะแนนตามคำตอบที่ใกล้เคียง ในรายที่มีความผิดปกติของความเข้าใจภาษา (Aphasia) หรือ ผู้ที่สูญเสียระดับความรู้สึกตัวและไม่เข้าใจคำถามจะได้คะแนน “2” ในผู้ป่วยที่พูดไม่ได้เนื่องจากใส่ท่อช่วย หายใจ หรือมีภยันตรายของหลอดลม ผู้ป่วยที่พูดไม่ชัดอย่างมากไม่ว่าจากสาเหตุใดๆก็ตาม ผู้ป่วยที่ไม่ เข้าใจในภาษาไทยหรือตอบคำถามไม่ได้ เนื่องจากปัญหาอื่นๆที่ไม่ได้เกิดจากความผิดปกติของความเข้าใจ ภาษา (Aphasia) จะได้คะแนน “1” คำตอบแรกมีความสำคัญมาก เนื่องจากเป็นคำตอบที่ใช้คะแนน ห้ามผู้ทดสอบกล่าวเป็นนัยให้ทราบหรือให้ท่าทางเพื่อนำไปสู่คำตอบที่ต้องการ

1C. หลับตา-ลืมตา และกำมือ คลายมือข้างที่ไม่เป็นอัมพาตได้หรือไม่ (LOC Commands) โดยมีคะแนน 0-2 ดังนี้

0 = ทำได้ถูกต้องทั้ง 2 อย่าง

1 = ทำได้ถูกต้องเพียงอย่างเดียว

2 = ไม่ทำตามคำสั่ง หรือทำไม่ถูกต้อง

ระดับความรู้สึกตัวหมวดคำสั่ง (LOC Commands) ให้ปฏิบัติตามคำบอก 2 ข้อ คือ

1) ลืมตาและหลับตา

2) กำมือและแบมือข้างที่ไม่อ่อนแรง

ในกรณีที่มือทั้งสองข้างไม่สามารถใช้งานได้ผู้ตรวจอาจใช้คำสั่งขั้นตอนเดียวอื่น ๆ ที่ไม่ยากนักในการประเมิน

ในกรณีที่กำหรือแบมือได้แต่ไม่เต็มที่เนื่องจากการอ่อนแรงให้ถือว่าสามารถทำได้ สำหรับผู้ที่ไม่ ทำตามคำสั่ง ให้ใช้ท่าทางช่วยให้เข้าใจหรือให้ทำตามได้ จากนั้นจึงให้คะแนนตามที่ปฏิบัติจริง (เช่น ไม่ทำ ตามคำสั่งเลย, สามารถทำตามคำสั่งได้หนึ่งข้อ, สามารถทำได้ทั้งสองข้อ)

ในกรณีที่มีบาดเจ็บถูกตัดแขนหรือมีปัญหาทางร่างกายของส่วนที่ใช้ในการปฏิบัติตาม คำสั่งข้างต้นให้ใช้คำสั่งขั้นตอนเดียวอื่นๆทดแทนได้ โดยจะให้คะแนนเฉพาะการตอบสนองแรกเท่านั้น

2. การเคลื่อนไหวของตา (Best Gaze) โดยมีคะแนน 0-2 ดังนี้

0 = มองตามได้เป็นปกติ

1 = ตาข้างใดข้างหนึ่งหรือทั้ง 2 ข้าง เหลือบมองไปด้านข้างได้แต่ไม่สุด

2 = เหลือบตามองไปด้านข้างไม่ได้เลย หรือมองไปด้านหนึ่งด้านใดจนสุด โดยไม่สามารถ แก้ไขได้ด้วย Oculocephalic Maneuver

การเคลื่อนไหวของตา (Best Gaze) ประเมินเฉพาะการเคลื่อนไหวของตาในแนวนอนเท่านั้น โดยให้มองตามหรือปฏิกิริยาตอบสนองชนิด Oculocephalic ในกรณีที่ตามองไปด้านใดด้านหนึ่งเพียง ด้านเดียว (Conjugate Gaze Deviation) ให้สั่งผู้ป่วยให้มองไปด้านตรงข้าม หรือตรวจปฏิกิริยาตอบสนอง Oculocephalic ถ้าทำได้จะได้คะแนนเท่ากับ “1” ในกรณีที่พบความผิดปกติของเส้นประสาทสมองคู่ที่ 3,4 หรือ 6 ผู้ป่วยจะได้รับคะแนนเท่ากับ “1” การตรวจนี้สามารถทำได้แม้ในผู้ป่วยที่มีปัญหาของความ เข้าใจภาษา (Aphasia) แนะนำให้ตรวจปฏิกิริยาตอบสนอง Oculocephalic ในผู้ป่วยที่มีความผิดปกติ ของการมองเห็น นอกจากนี้อาจใช้เทคนิคการมองตาผู้ป่วย (Eye Contact) จากนั้นสังเกตการณ์ เคลื่อนไหวของตาผู้ป่วยที่มีต่อการเคลื่อนไหวของผู้ตรวจ โดยวิธีนี้อาจช่วยบอกความผิดปกติได้

3. การมองเห็น (Visual Fields) การตรวจการมองเห็นนี้ผู้ตรวจจะทำการตรวจตาทีละข้าง โดย อาจใช้มือปิดตาอีกข้างหนึ่งก่อน มีการให้คะแนน 0-3 ดังนี้

0 = ลานสายตาปกติ

1 = ลานสายตาผิดปกติบางส่วน (Partial Hemianopia)

2 = ลานสายตาผิดปกติครึ่งซีก (Complete Hemianopia

3 = มองไม่เห็นทั้ง 2 ตา (ตาบอด)

การมองเห็น (Visual Fields) ให้ประเมินลานสายตาของผู้ป่วยโดยการตรวจวิธีเผชิญหน้า(Confrontation test) ทั้ง 4 ตำแหน่ง (Quadrant) โดยใช้ Visual Threat หรือให้ผู้ป่วยนับนิ้ว ถ้ามีการ ตอบสนองอย่างเหมาะสม ก็ถือว่าการตรวจเป็นปกติ ในกรณีที่ผู้ป่วยตาบอดหรือได้รับการผ่าตัดตาให้ ตรวจลานสายตาเฉพาะข้างที่เหลืออยู่ คะแนน “1” จะให้เฉพาะกรณีที่ตรวจพบความผิดปกติของลาน สายตาบางส่วน หรือเป็น Quadrantanopia เท่านั้น ผู้ป่วยที่ตาบอดทั้ง 2 ข้าง ไม่ว่าจากสาเหตุใดก็ตาม จะได้คะแนน “3” ผู้ตรวจควรใช้เทคนิค Double Simultaneous Stimulation ร่วมด้วย (โดย กระตุ้นให้มองที่ลานสายตาทั้งสองข้างพร้อมกัน) ซึ่งถ้าพบภาวะ Visual Extinction ผู้ป่วยจะได้คะแนน เท่ากับ “1” ในหัวข้อ 11

4. การเคลื่อนไหวของกล้ามเนื้อใบหน้า (Facial Palsy) โดยมีคะแนน 0-3 ดังนี้

0 = ไม่พบมีอาการอ่อนแรงของกล้ามเนื้อใบหน้า สามารถเคลื่อนไหวกล้ามเนื้อใบหน้าได้เป็นปกติ

1 = กล้ามเนื้อใบหน้าอ่อนแรงเล็กน้อย พอสังเกตเห็นมุมปากตกหรือไม่เท่ากันเมื่อผู้ป่วยยิ้ม

2 = กล้ามเนื้อใบหน้าอ่อนแรงมาก แต่ยังพอเคลื่อนไหวกล้ามเนื้อได้บ้าง

3 = ไม่สามารถเคลื่อนไหวกล้ามเนื้อใบหน้าในข้างใดหรือทั้ง 2 ข้างได้เลย

การเคลื่อนไหวของกล้ามเนื้อใบหน้า (Facial Palsy) ใช้คำสั่งหรือแสดงท่าทางให้ผู้ป่วยยิงฟัน หรือยิ้มและหลับตา จากนั้นจึงประเมินการอ่อนแรงของกล้ามเนื้อใบหน้า สำหรับผู้ที่ไม่ให้

ความร่วมมือ หรือมีระดับความรู้ตัวผิดปกติ ให้กระตุ้นโดยใช้สิ่งเร้าที่ทำให้เกิดความสมมาตรกันหรือไม่

ผู้ประเมินอาจต้องเคลื่อนย้ายสิ่งปกปิดใบหน้าออกเท่าที่ทำได้ในกรณีผู้ป่วยได้รับภยันตราย บริเวณใบหน้า ซึ่งจำเป็นต้องได้รับได้รับการทำแผลหรือผู้ป่วยที่ได้รับการใส่ท่อช่วยหายใจหรือมีสาเหตุ อื่นที่ต้องได้รับการปกปิดใบหน้า

5. กำลังของกล้ามเนื้อแขนและขา (Motor Arm and Leg) โดยมีคะแนน 0-4 ดังนี้

0 = ยกแขนสูง 90 องศาทำมุมกับลำตัวในท่านั่ง หรือ 45 องศาในท่านอนหงาย และสามารถคงไว้ในตำแหน่งที่ต้องการได้ตลอด 10 วินาที

1 = ยกแขนสูง 90 องศาทำมุมกับลำตัวในท่านั่ง หรือ 45 องศาในท่านอนหงายและ สามารถคงไว้ในตำแหน่งที่ต้องการได้เพียงครู่เดียวไม่ถึง 10 วินาที โดยที่แขนไม่ตกลงบนเตียง

2 = ยกแขนขึ้นได้บ้างแต่ไม่ถึงหรือไม่สามารถคงไว้ในตำแหน่งที่ต้องการได้จากนั้นแขน ตกลงบนเตียง

3 = ไม่สามารถยกแขนขึ้นได

4 = ไม่มีการเคลื่อนไหวของกล้ามเนื้อแขน

UN = แขนพิการหรือถูกตัด หรือพบมีปัญหาข้อติดยึดที่ไม่สามารถแปลผลการตรวจได้

กำลังของกล้ามเนื้อแขน (Motor Arm) A ข้างซ้าย B ข้างขวา ให้ผู้ป่วยเหยียดแขนออกไปด้านหน้าจนสุด ค้างไว้ 10 วินาทีในท่าฝ่ามือ โดยให้ทำมุม 90 องศา กับลำตัวในท่านั่ง หรือ 45 องศา ในท่านอน จากนั้นให้ตรวจกำลังของกล้ามเนื้อขา (ตรวจในท่านอนเสมอ) โดยให้ยกขาทีละข้างสูงทำมุม 30 องศา ในท่าเหยียด ค้างไว้ 5 วินาที โดยจะตัดสินว่าอ่อนแรงเมื่อแขนตกก่อน 10 วินาที หรือขาตกลงก่อน 5 วินาที สำหรับผู้ที่มีความผิดปกติของความเข้าใจภาษา (Aphasia) ให้ใช้ท่าทางหรือน้ำเสียงกระตุ้นให้ ผู้ป่วยปฏิบัติตามได้ ไม่แนะนำให้ใช้สิ่งเร้าที่ทำให้เกิดความเจ็บปวด การตรวจควรเริ่มจากแขนหรือขาข้าง ที่ปกติก่อนโดยตรวจทีละข้าง

จะให้คะแนน “UN” เฉพาะผู้ป่วยที่ถูกตัดแขนหรือขาหรือมีการยึดตัวของไหล่หรือข้อสะโพก ในกรณีนี้ผู้ตรวจควรเขียนเหตุผลดังกล่าวให้ชัดเจน

6. กำลังของกล้ามเนื้อแขนและขา (Motor Arm and Leg) โดยมีคะแนน 0-4 ดังนี้

0 = สามารถยกขาข้างที่อ่อนแรงขึ้นให้สะโพกทำมุม 30 องศากับพื้นในท่านอนหงาย และ คงตำแหน่งที่ต้องการได้ตลอด 5 วินาที

1 = สามารถยกขาข้างที่อ่อนแรงขึ้นให้สะโพกทำมุม 30 องศากับพื้นในท่านอนหงายได้ครู่ เดียว โดยไม่ถึง 5 วินาที ก็ต้องลดขาลงมา แต่ขาไม่ตกลงบนเตียง

2 = ยกขาขึ้นได้บ้างในท่านอนหงายแต่ไม่ถึงตำแหน่งที่ต้องการ ขาตกลงบนเตียงก่อน 5 วินาที

3 = ไม่สามารถยกขาขึ้นจากเตียงได้ในท่านอนหงาย

4 = ไม่มีการเคลื่อนไหวของกล้ามเนื้อขา

UN = ขาพิการหรือถูกตัด หรือพบมีปัญหาข้อติดยึดที่ไม่สามารถแปลผลการตรวจได้

กำลังของกล้ามเนื้อแขนและขา (Motor Arm and Leg) A ข้างซ้าย B ข้างขวา ให้ผู้ป่วยเหยียด แขนออกไปด้านหน้าจนสุด ค้างไว้ 10 วินาที ในท่าฝ่ามือ โดยให้ทำมุม 90 องศา กับลำตัวในท่านั่งหรือ 45 องศา ในท่านอน จากนั้นให้ตรวจกำลังของกล้ามเนื้อขา (ตรวจในท่านอนเสมอ) โดยให้ยกขาทีละข้าง สูงทำมุม 30 องศา ในท่าเหยียด ค้างไว้ 5 วินาที หรือขาตกลงก่อน 5 วินาที สำหรับผู้ที่มีความผิดปกติ ของความเข้าใจภาษา (Aphasia) ให้ใช้ท่าทางหรือฐ้าเสียงกระตุ้นให้ผู้ป่วยปฏิบัติตามได้ ไม่แนะนำให้ใช้ สิ่งเร้าที่ทำให้เกิดความเจ็บปวด การตรวจควรเริ่มจากแขนหรือขาข้างที่ปกติก่อนโดยตรวจทีละข้าง

จะให้คะแนน “UN” เฉพาะผู้ป่วยที่ถูกตัดแขนหรือขาหรือมีการยึดตัวของไหล่หรือข้อสะโพก ในกรณีนี้ผู้ตรวจควรเขียนเหตุผลดังกล่าวให้ชัดเจน

7. การประสานงานของแขนขา (Limb Ataxia) โดยมีคะแนน 0-2 ดังนี้

0 = การประสานงานของแขนขาทั้ง 2 ข้าง เป็นปกติ

1 = มีปัญหาในการประสานงานของแขนหรือขา 1 ข้าง

2 = มีปัญหาในการประสานงานของแขนหรือขา 2 ข้าง

UN = แขนหรือขาพิการหรือถูกตัดหรือพบปัญหาข้อติดยึดที่ไม่สามารถแปลผลการตรวจได้

การประสานงานของแขนขา (Limb Ataxia) ตรวจ Finger to Nose to Finger (โดยให้ผู้ป่วยใช้นิ้วชี้แตะปลายจมูกของตนเองจากนั้นให้เหยียดแขนจนสุดเพื่อแตะปลายนิ้วของผู้ตรวจ ทำสลับกันไปมา และดูการเคลื่อนไหวว่าแม่นยำหรือไม่) จากนั้นตรวจ Heel to Shin Test (ให้ผู้ป่วยยกส้นเท้ามาแตะที่หัวเข่าแล้วลากลงผ่านสันหน้าแข้งจนสุด และทำฐ้าอย่างต่อเนื่องกันไปเรื่อยๆ เพื่อดูการเคลื่อนไหวของเท้าว่าปกติหรือไม่) ให้ประเมินทั้ง 2 ข้าง ในกรณีที่ลานสายตาผิดปกติ ให้ทำการตรวจดังกล่าวเฉพาะในบริเวณที่ลานสายตาผิดปกติโดยจะถือว่ามี Ataxia ต่อเมื่อพบความผิดปกติของการเคลื่อนไหวที่ชัดเจนให้ถือว่าไม่ พบ Ataxia ในผู้ป่วยที่มีอาการแขนขาอ่อนแรงรุนแรงขยับไม่ได้ (Hemiplegia) หรือในผู้ป่วยที่ไม่เข้าใจ คำสั่ง

คะแนน “UN” จะให้เฉพาะกับผู้ป่วยที่ถูกตัดแขนขาหรือมีภาวะข้อยึดติดเท่านั้น โดยระบุ เหตุผลให้ชัดเจนในผู้ป่วยที่มีปัญหาด้านการมองเห็น ให้ตรวจเฉพาะ Finger to Nose Test ได้

8. การรับความรู้สึก (Sensory) โดยมีคะแนน 0-2 ดังนี้

0 = การรับความรู้สึกเป็นปกติ

1 = สูญเสียการรับความรู้สึกในระดับน้อยถึงปานกลาง การรับความรู้สึกจากวัตถุปลาย แหลมคมลดลง แต่ยังสามารถบอกได้ถึงความรู้สึกในบริเวณที่ถูกกระตุ้น

2 = สูญเสียการรับความรู้สึกในระดับรุนแรงหรือไม่รู้สึกว่าถูกสัมผัสที่บริเวณใบหน้าแขนและขา

ารรับความรู้สึก (Sensory) ประเมินการรับความรู้สึกตามคำบอกเล่าของผู้ป่วยในผู้ที่มีความ รู้สึกตัวลดลงหรือมีความผิดปกติของความเข้าใจภาษา (Aphasia) ให้ดูการตอบสนองสีหน้าหรือการชัก แขนหนีที่มีต่อวัตถุปลายแหลม ควรตรวจหลายๆตำแหน่งของร่างกาย เช่น แขน ขา ลำตัว ใบหน้า หรือ บริเวณอื่นเพื่อให้มั่นใจว่าผิดปกติจริง ผู้ที่มีความผิดปกติด้านการรับความรู้สึกที่รุนแรงจะได้คะแนน “2” โดยผู้ป่วยกลุ่มนี้จะไม่ทราบว่าถูกสัมผัส

ในผู้ที่สูญเสียความรู้สึกตัวหรือมีความผิดปกติของความเข้าใจภาษา อาจให้คะแนน “0” หรือ “1” สำหรับผู้ที่มีพยาธิสภาพในตำแหน่งของก้านสมองและสูญเสียการรับความรู้สึกของร่างกายทั้ง 2 ด้าน หรือผู้ที่ไม่ตอบสนองและมีอาการแขนขาอ่อนแรง จะได้รับคะแนน “2” เช่นเดียวกับผู้ที่หมดสติลึก (Coma, 1a=3) เฉพาะความผิดปกติของการรับความรู้สึกที่เกิดจากโรคหลอดเลือดสมองเท่านั้นที่จะถูกนำมาคิดคะแนน

9. ความสามารถด้านภาษา (Best Language) โดยมีคะแนน 0-3 ดังนี้

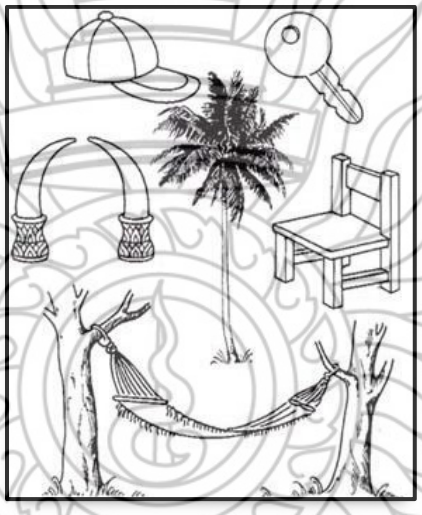
0 = การสื่อภาษาเป็นปกติ

1 = การสื่อภาษาสูญเสียไปในระดับน้อยถึงปานกลาง มีการสูญเสียความเข้าใจหรือความสามารถในการใช้ภาษา แต่ผู้ทดสอบยังพอที่จะเข้าใจว่าผู้ป่วยกำลังพูดถึงอะไรอยู่

2 = การสื่อภาษาสูญเสียอย่างรุนแรง ผู้ป่วยไม่สามารถสื่อสารให้เข้าใจได้และผู้ทดสอบ ไม่สามารถทราบได้ว่าผู้ป่วยกำลังพูดถึงอะไร

3 = ไม่พูดหรือไม่เข้าใจภาษาที่ผู้ตรวจพยายามสื่อและไม่สามารถแสดงท่าทางพูดหรือ เขียนให้ผู้อื่นเข้าใจได้ (Globalaphasia)

ภาพที่ 2.1 บรรยายรูปภาพตามภาพ
(รังสฤษฎ์ รังสรรค์, 2556)



ภาพที่ 2.2 บอกชื่อสิ่งของตามภาพ (Naming)
(รังสฤษฎ์ รังสรรค์, 2556)

ความสามารถด้านภาษา (Best Language) ความสามารถในการสื่อสารและความเข้าใจภาษา จะถูกประเมินจากการตรวจที่ผ่านมา สำหรับข้อนี้ผู้ป่วยจะถูกประเมินโดยให้บรรยายสิ่งที่เกิดขึ้นในภาพ บอกชื่อสิ่งของต่างๆ และให้อ่านประโยคที่ได้แนบมากับแบบทดสอบนี้ ความเข้าใจภาษาจะถูกประเมิน จากการตอบสนองต่อคำสั่งที่ใช้ในการตรวจร่างกายในกรณีที่การมองเห็นผิดปกติ ให้ตรวจโดยให้ผู้ป่วยกำสิ่งของที่คุ้นเคย จากนั้นบอกชื่อของสิ่งนั้นและพูดซ้ำๆ เพื่อความชัดเจนของการออกเสียง ในผู้ที่ได้รับการ ใส่ท่อช่วยหายใจ ให้ประเมินด้วยการเขียน ในผู้ที่ไม่รู้สึกตัว (Coma, 1a=3) จะได้รับคะแนน “3” ใน หัวข้อนี้ผู้ประเมินจะต้องให้คะแนนเสมอไม่เว้นแม้ในผู้ป่วยที่มีความรู้สึกตัวเปลี่ยนไป (ซึมหรือไม่ร่วมมือ) อย่างไรก็ตามคะแนน “3” จะให้เฉพาะกับผู้ที่ไม่พูดและไม่สามารถตอบสนองต่อคำสั่งอย่างเหมาะสม

10. การออกเสียง (Dysarthria) โดยมีคะแนน 0-2 ดังนี้

0 = เปล่งเสียงได้เป็นปกติ

1 = พูดไม่ชัดเล็กน้อยถึงปานกลาง ผู้ป่วยพูดไม่ชัดเป็นบางคำโดยผู้ตรวจพอเข้าใจได้

2 = พูดไม่ชัดอย่างมากหรือไม่พูด ไม่สามารถเข้าใจคำพูดของผู้ป่วยได้ โดยไม่มีความ ผิดปกติของความเข้าใจภาษา

UN = ผู้ที่ได้รับการใส่ท่อช่วยหายใจหรือมีปัญหาทางกายอื่นๆที่มีผลต่อการเปล่งเสียงเท่านั้น

ภาพที่ 2.3 การออกเสียงจากบัตรคำ

(รังสฤษฎ์ รังสรรค์, 2556)

ภาพที่ 2.4 การออกเสียงจากบัตรคำ

(รังสฤษฎ์ รังสรรค์, 2556)

การออกเสียง (Dysarthria) ประเมินความชัดเจนของการออกเสียงโดยให้อ่านคำหรือ ประโยคสั้นๆที่เตรียมไว้ในส่วนท้ายของแบบทดสอบในผู้ที่ผิดปกติของความเข้าใจภาษา (Aphasia) ให้ ประเมินความชัดเจนของการออกเสียงเมื่อผู้ป่วยพูด

เฉพาะผู้ที่ได้รับการใส่ท่อช่วยหายใจหรือมีปัญหาทางกานอื่นๆที่มีผลต่อการเปล่งเสียงเท่านั้น ที่จะได้คะแนนเท่ากับ “UN” โดยผู้ตรวจต้องระบุเหตุผลให้ชัดเจนของการออกเสียง

11. การขาดความสนใจในด้านหนึ่งด้านใดของร่างกาย (Extinction and Inattention, Formerly Neglect) โดยมีคะแนน 0-3 ดังนี้

0 = ไม่พบความผิดปกติ

1 = พบความผิดปกติของการรับรู้ชนิดใดชนิดหนึ่งดังต่อไปนี้คือ การมองเห็น การสัมผัส การได้ยิน เมื่อมีการกระตุ้นทั้ง 2 ข้างพร้อม ๆ กัน

2 = มีความผิดปกติของการรับรู้ มากกว่า 1 ชนิด หรือผู้ป่วยไม่รับรู้ว่าเป็นมือของตนเอง หรือสนใจต่อสิ่งเร้าเพียงด้านเดียว

การขาดความสนใจในด้านหนึ่งด้านใดของร่างกาย (Extinction and Inattention, Formerly Neglect) ข้อมูลจากการตรวจที่ผ่านมาอาจเพียงพอที่จะระบุว่าผู้ป่วยมีภาวะ inattention หรือไม่ ใน กรณีที่สูญเสียความสามารถในการมองเห็นแต่การรับรู้ความรู้สึกทางผิวหนังทั้งสองข้างปกติ ให้ถือว่าไม่ พบภาวะ inattention เช่นเดียวกับผู้ที่มีความผิดปกติของความเข้าใจภาษา (aphasia) แต่แสดงถึงความ สนใจในสิ่งเร้าทั้ง 2 ด้าน ก็ถือว่าผลการตรวจเป็นปกติ

การตรวจพบว่าผู้ป่วยขาดความสนใจต่อสิ่งเร้า (visuospatial neglect, anosognosia) ในด้าน ใดด้านหนึ่งของร่างกาย ให้ถือว่ามีภาวะ neglect โดยหัวข้อนี้จะถือว่าผิดปกติก็ต่อเมื่อตรวจพบเท่านั้น (แพทยสมาคมแห่งประเทศไทยในพระบรมราชูปถัมภ์, 2553)

การให้คะแนนจากการประเมินด้วย NIHSS จำแนกออกเป็น 4 ระดับ

คะแนน ≥ 25 = Very Severe Impairment

คะแนน 15–24 = Severe Impairment

คะแนน 5–14 = Mild to Moderately

คะแนน ≤ 4 = Mild Impairment

ข้อควรคำนึงในการประเมินระบบประสาทผู้ป่วย

เมื่อพยาบาลทำการประเมินหรือตรวจระบบประสาทผู้ป่วยแล้วพยาบาลจะต้องทำการบันทึก คะแนนทันทีเมื่อตรวจเสร็จห้ามย้อนกลับเพื่อไปเปลี่ยนแปลงคะแนน นอกจากนี้คะแนนที่ให้จะบอกถึง ความสามารถที่ผู้ป่วยทำได้จริง “ไม่ใช่” สิ่งที่ผู้ตรวจคิดว่าผู้ป่วยน่าจะทำได้ และการตรวจควรเป็นไปอย่างรวบรัด “ไม่ควร” กล่าวเป็นนัยหรือกล่าวชี้นำเพื่อให้ผู้ป่วยทำในสิ่งที่ต้องการแต่อาจทำได้ในบางกรณี ซึ่งจะระบุไว้ในแต่ละข้อย่อย

นอกจากนี้บทบาทของพยาบาลในการเป็นผู้ประเมินหรือตรวจระบบประสาทผู้ป่วยที่มีปัญหาทาง โรคหลอดเลือดสมองสามารถกระทำได้ และมีความน่าเชื่อถือ ซึ่งได้รับการยืนยันจากงานวิจัยที่พบว่าผลของการตรวจไม่แตกต่างจากแพทย์ ดังนั้นพยาบาลผู้ที่ต้องให้การ

ดูแลผู้ป่วยที่มีปัญหาทางโรคหลอดเลือดสมอง ควรเริ่มจากการฝึก ตรวจผู้ป่วยดังกล่าวร่วมกับผู้เชี่ยวชาญก่อนในระยะแรกๆ และเมื่อกระทำบ่อยๆ จะค่อยเกิดความชำนาญ ขึ้น ซึ่งจะทำให้เพิ่มคุณภาพในการดูแลผู้ป่วยปัญหาทางโรคหลอดเลือดสมองมากขึ้น (Goldstein และ Samsa ,1997 และยงชัย นิละนนท์, 2553)

2.1.9 ข้อบ่งชี้และข้อห้าม

ข้อบ่งชี้

- มีอาการของหลอดเลือดสมองตีบ ภายใน 4.5 ชั่วโมง (ในกรณีไม่ทราบเวลาที่เริ่มมีอาการอย่างชัดเจน หรือ มีอาการหลังตื่นนอน ให้นับเวลาล่าสุดที่มีพยานยืนยันว่ายังปกติ เป็นเวลาที่เริ่มมี อาการ)
- อายุ > 18 ปี
- ผล CT brain ไม่พบว่ามีเลือดออกในเนื้อสมองหรือชั้นใต้เยื่อหุ้มสมอง

ข้อห้าม

- ความดันโลหิตช่วงก่อนให้การรักษาสูง (SBP ≥ 185 มิลลิเมตรปรอท หรือ DBP ≥ 110 มิลลิเมตรปรอท
- CT brain พบมีสมองขาดเลือดมากกว่าขนาด 1/3 ของ Cerebral Hemisphere
- มีประวัติเลือดออกในสมองหรือกะโหลกศีรษะ ใน 3 เดือน
- มีอาการที่สงสัยว่ามีเลือดออกชั้นใต้เยื่อหุ้มสมอง (Subarachold Hemorrhage)
- มีประวัติเป็นโรคหลอดเลือดสมอง หรือบาดเจ็บที่ศีรษะรุนแรงภายใน 3 เดือน
- มีประวัติได้รับยาต้านการแข็งตัวของเลือดโดยมีค่า Prothrombin time ≥ 15 วินาที หรือมีค่า International Normalized Ratio (INR) ≥ 1.7
- ได้รับยา Heparin ภายใน 48 ชั่วโมง และมีค่า Partial-Thromboplastin Time (PTT) ผิดปกติ
- มีปริมาณเกร็ดเลือดน้อยกว่า 100,000/ลูกบาศก์มิลลิเมตร
- ตรวจร่างกายพบมีภาวะเลือดออก (Active Bleeding)
- มีการแทงหลอดเลือดแดงขนาดใหญ่ในตำแหน่งที่ไม่สามารถกดได้ภายใน 7 วัน
- มีระดับน้ำตาลในเลือด ≤ 50 มิลลิกรัม/เดซิลิตร (2.7 มิลลิโมล/ลิตร)

- อาการทางระบบประสาทดีขึ้นอย่างรวดเร็วจนเกือบเป็นปกติ หรือมีอาการอย่างเดียวไม่รุนแรง เช่น แขนขาอ่อนแรงเล็กน้อย โดย NIHSS น้อยกว่า 4 ยกเว้น Aphasia หรือมี Hemianopia

- มีประวัติผ่าตัดใหญ่ภายใน 14 วัน

- มีเลือดออกในทางเดินอาหารหรือทางเดินปัสสาวะภายใน 21 วัน

- อาการชักตอนเริ่มมีอาการร่วมกับอ่อนแรงหลังชัก (Todd Paralysis)

- มีประวัติ Recent Myocardial Infarction ภายใน 3 เดือน

ข้อห้ามเพิ่มเติมกรณีให้ยาในผู้ป่วยที่มีอาการระหว่าง 3 ถึง 4.5 ชั่วโมง

- มีประวัติได้รับยาละลายลิ่มเลือด (Warfarin) โดยไม่พิจารณาค่า INR

- อายุ > 80 ปี

- เป็นเบาหวานร่วมกับเคยมีโรคหลอดเลือดสมองอุดตันมาก่อน

- NIHSS > 25

2.1.10 ปริมาณยาและวิธีการบริหารยา

ยา Recombinant Tissue – Plasminogen Activator (rt-PA) ให้ยาในขนาด 0.9n มิลลิกรัม/กิโลกรัม ตามน้ำหนักผู้ป่วยแต่ไม่เกิน 90 มิลลิกรัม โดยแบ่งให้ 10% ทางหลอดเลือดดำใน 1 นาที ส่วนที่เหลือหยดทางหลอดเลือดดำต่อใน 60 นาที

2.2 การพัฒนาแอพพลิเคชั่นบนระบบปฏิบัติการไอโอเอส (iOS)

2.2.1 ระบบปฏิบัติการไอโอเอส (iOS)

ปัญญาพล พูลสวัสดิ์ (2558) ได้กล่าวถึง ไอโอเอส คือ ระบบปฏิบัติการบนอุปกรณ์พกพา (สมาร์ตโฟนหรือแท็บเล็ต) พัฒนาและจำหน่ายโดยบริษัทแอปเปิ้ล (Apple Inc.) เปิดตัวครั้งแรกในปี พ.ศ. 2550 เพื่อใช้บนไอโฟน (iPhone) และได้มีการพัฒนาเพิ่มเติมเพื่อใช้บน อุปกรณ์พกพาอื่น ๆ ของแอปเปิ้ล เช่น ไอพอดทัช (iPod touch) (ในเดือนกันยายน 2550) ไอแพด (iPad) (ในเดือนมกราคม 2553) ไอแพด มินิ (iPad mini) (พฤศจิกายน 2555) และ แอปเปิ้ลทีวี รุ่นที่ 2 (Apple TV gen 2) (ในเดือนกันยายน 2553) iOS แตกต่างจากวินโดวส์โฟนของไมโครซอฟท์และแอนดรอยด์ (ระบบปฏิบัติการ) ของกูเกิล โดยแอปเปิ้ลไม่อนุญาตให้นำ iOS ไปติดตั้งบนอุปกรณ์ที่ไม่ใช่อุปกรณ์ของแอปเปิ้ล ในเดือนสิงหาคม พ.ศ. 2556 แอปสโตร์ (App Store) ของแอปเปิ้ลมีแอพพลิเคชั่นมากกว่า 900,000 แอพพลิเคชั่นและมี 375,000 แอพพลิเคชั่น ที่ออกแบบมาเพื่อไอแพด (iPad) โดยเฉพาะ แอพพลิเคชั่นเหล่านี้มียอดดาวน์โหลดรวมกันมากกว่า 5 หมื่นล้านครั้ง ไอโอเอสมีส่วนแบ่ง 21% ของส่วนแบ่งระบบปฏิบัติการบนอุปกรณ์พกพาในไตรมาสที่ 4 ของปีพ.ศ. 2555 ในเดือนมิถุนายน 2555 ไอโอเอสมีส่วนแบ่งคิดเป็น 65% ของการบริโภคข้อมูลบนอุปกรณ์พกพา (ซึ่งรวม ไอพอดทัช และ ไอแพด) ในกลางปี 2555 มีอุปกรณ์ iOS มากกว่า 410 ล้านเครื่องที่เปิดใช้งาน จากการ

อ้างอิงจากงานแถลงเปิดตัวต่อสื่อโดยแอปเปิลใน วันที่ 12 กันยายน 2555 มีอุปกรณ์ไอโอเอส 400 ล้านตัวที่จำหน่ายไปแล้วในเดือนมิถุนายน 2555

ส่วนต่อประสานกับผู้ใช้ (User Interface) ของ iOS มีพื้นฐานแนวคิดมาจาก การควบคุมโดยตรง (Direct Manipulation) ด้วยการใช้มัลติทัช (Multitouch) องค์ประกอบของการควบคุม คือการใช้นิ้วเลื่อนสวิทช์ และปุ่ม เพื่อเป็นการควบคุมอุปกรณ์รวมถึงท่าทางอย่างอื่น เช่น การนำนิ้วมือปัดจากซ้ายไปขวา หรือขวาไปซ้าย (Pinch) แตะเบา ๆ (Tap) การนำนิ้วสองนิ้วบีบเขาหาศูนย์กลาง (Pinch), การนำนิ้วสองนิ้วกางออกจากศูนย์กลาง (Reverse pinch) ซึ่งทั้งหมดนี้มีความหมายที่เจาะจงในบริบทต่าง ๆ ของไอโอเอสและถือเป็นการใช้งานแบบส่วนต่อประสานกับผู้ใช้แบบมัลติทัช ภายในอุปกรณ์ที่ติดตั้งไอโอเอสจะมีเซนเซอร์ตรวจจับการเคลื่อนไหว เพื่อทำงานกับแอพพลิเคชั่นบางส่วนที่ตอบสนองการสั่นของอุปกรณ์ หรือ การหมุนอุปกรณ์ที่คำนวณในรูปแบบสามมิติ

2.2.2 ชุดพัฒนา iOS SDK

อิสริยะ ไพรีพ่ายฤทธิ์ (2552) ได้กล่าวถึงการพัฒนาโปรแกรมบน iOS ต้องมีชุดพัฒนาโปรแกรมสำหรับไอโอเอส (Apple’s iOS Software Development Kit: iOS SDK) ก่อน ซึ่งสมาชิกโครงการนักพัฒนาของแอปเปิ้ล จะสามารถดาวน์โหลด iOS SDK มาใช้พัฒนาได้ฟรีจาก https://developer.apple.com ซึ่งชุดพัฒนาทั้งหมดจะรวมอยู่ในซอฟต์แวร์ Xcode ที่เป็นสิ่งสำคัญที่สุดในการพัฒนาแอพพลิเคชั่นบนไอโอเอส โดยเครื่องมือนี้ มีความสามารถอย่างครบครัน มีระบบจัดการสภาพแวดล้อมในการพัฒนา รวมถึงตัวแก้ไขซอร์สโค้ดที่ชาญฉลาด ดีบั๊กเกอร์ (Debugger) แบบกราฟฟิก Xcode ใช้ส่วนประกอบหลายอย่างที่เป็น ซอฟต์แวร์โอเพนซอร์สของ GNN เช่น gcc (GNU Compiler Collection) และ gdb (GNU Preject Debugger) โดยสามารถใช้ Xcode พัฒนาแอพพลิเคชั่นได้ทั้งบนไอโอเอสและแมคโอเอส (macOS) ใน Xcode มีส่วนประกอบหลายอย่างซึ่งเป็นพื้นฐานสำคัญของการพัฒนาโปรแกรมบน iOS มีดังต่อไปนี้

2.2.2.1 Instruments เป็นเครื่องมือตรวจสอบการทำงานของโปรแกรมบน iPhone โดยจะสุ่มดูการใช้งานหน่วยความจำ และคอยตรวจสอบประสิทธิภาพของโปรแกรม เครื่องมือนี้ช่วยค้นหาต้นตอของปัญหาในโปรแกรมได้ง่าย Instruments สามารถแสดงการทำงานออกมาอยู่ในรูปกราฟ ตามลำดับเวลา ซึ่งจะบ่งชี้ว่าโปรแกรมของคุณส่วนไหนใช้ทรัพยากรของระบบมากที่สุด Instruments จะมีบทบาทสำคัญในการค้นหาจุดที่เรียกใช้หน่วยความจำอย่างสิ้นเปลือง และช่วยให้โปรแกรมทำงานบน iPhone ได้อย่างมีประสิทธิภาพ

2.2.2.2 Simulator จะจำลองการทำงานของ iPhone บนเครื่องแม็คเพื่อให้คุณสามารถพัฒนาและทดสอบโปรแกรมบนเครื่องคอมพิวเตอร์ได้โดยไม่ต้องใช้งาน iPhone โดย Simulator มี API แบบเดียวกับ iPhone และจะแสดงตัวอย่างของโปรแกรมที่สร้างขึ้นเพื่อให้ผู้พัฒนาเห็นหน้าจอและรูปแบบการทำงาน

2.2.2.3 Interface Builder (IB) เป็นตัวช่วยสร้างส่วนติดต่อกับผู้ใช้งานแบบกราฟฟิก และเชื่อมต่อเข้ากับโค้ดที่เขียนใน Xcode ซึ่งจะช่วยให้เราสร้างโปรแกรมต้นแบบได้รวดเร็วมากขึ้น ขั้นตอนการทำงานของ IB คือ ออกแบบ User Interface ของโปรแกรมด้วยเครื่องมือแบบกราฟฟิก จากนั้นก็ เชื่อมต่อส่วนประกอบที่เห็นบนจอเข้ากับอ็อบเจ็กต์และเมธอดทางโปรแกรมที่ได้ออกแบบไว้

2.2.2.4 Cocoa Touch คือ ไลบรารี่ที่รวมเอาคลาสต่าง ๆ ที่ Apple ได้เตรียมไว้ให้นักพัฒนา iOS เรียกใช้งาน ไลบรารี่ตัวนี้ประกอบไปด้วยเฟรมเวิร์คย่อยอีกหลายตัว ใช้สำหรับสร้างโปรแกรมแบบเชิงเหตุการณ์ (Event-driven) ที่เป็นกราฟฟิก ประกอบด้วยส่วนติดต่อผู้ใช้หลายชนิด เช่น หน้าต่างข้อความ หรือตาราง สามารถสร้างส่วนติดต่อผู้ใช้ที่นำกลับมาใช้ซ้ำใหม่ได้ (Reusable)

ตารางที่ 2.1 ตารางการพัฒนารุ่นของไอโอเอส (iOS)

รุ่น	เปิดตัวครั้งแรก	รายละเอียดหลัก
1	29 มิถุนายน พ.ศ. 2550	เปิดตัวพร้อมกับ iPhone 2G รุ่นแรก โดยใช้ชื่อว่า iPhone OS
2	11 กรกฎาคม พ.ศ. 2551	รองรับสำหรับการใช้งานใน iPhone 3G และ iPod Touch ซึ่งเปิดตัวเป็นครั้งแรก ทั้งยังรองรับ App store เป็นครั้งแรก
3	17 มิถุนายน พ.ศ. 2552	รองรับสำหรับการใช้งานใน iPhone 3GS สามารถคัดลอกและวางข้อความ และส่ง MMS ได้
4	21 มิถุนายน พ.ศ. 2553	รองรับสำหรับการใช้งานใน iPhone 4 เป็นรุ่นแรกที่ใช้ชื่อว่า iOS อย่างเป็นทางการ โดยใช้ชื่อว่า iOS 4 โดยเป็นเวอร์ชันแรกที่ iPhone รุ่นแรกไม่รองรับ ในรุ่นนี้รองรับฟังก์ชันมากมาย อาทิ Multitasking เป็นต้น และในรุ่น 4.2.1 เป็นรุ่นแรกที่เริ่มใช่งานใน ไอแพด ซึ่งเปิดตัวเป็นครั้งแรก

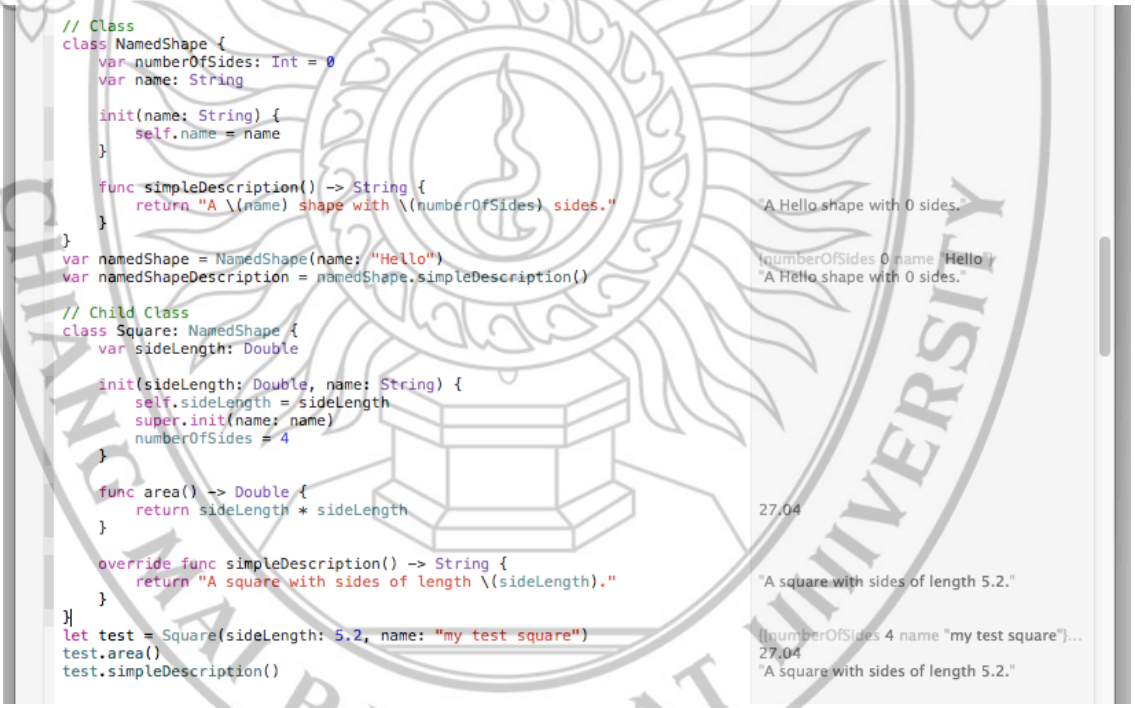
ตารางที่ 2.1 ตารางการพัฒนารุ่นของโอโอเอส (iOS) ต่อ

รุ่น	เปิดตัวครั้งแรก	รายละเอียดหลัก
5	6 มิถุนายน พ.ศ. 2554	รองรับสำหรับการใช้งานใน iPhone 4S รุ่นนี้มีการปรับเปลี่ยนรูปแบบหน้าตาของฟังก์ชันพื้นฐาน และรองรับระบบต่างๆมากมาย อาทิ ไอคลาวด์ และ สิริ เป็นต้น
6	19 กันยายน พ.ศ. 2555	รองรับสำหรับการใช้งานใน iPhone 5 และไอพอดทัช รุ่นที่ 5 เปลี่ยนไปใช้ระบบแผนที่ของ TomTom, สามารถ Facetime ผ่านระบบเซลลูล่าร์, การถ่ายภาพแบบพาโนรามา, คีย์บอร์ดภาษาไทยแบบ 4 แถว, แอปพลิเคชั่นนาฬิกาสำหรับ iPad
7	10 มิถุนายน พ.ศ. 2556	เปลี่ยนส่วนต่อประสานกับผู้ใช้ใหม่ทั้งหมดให้เป็นแบบเรียบง่าย เพิ่มสถานีวิทยุไอจูนส์ ศูนย์การตั้งค่าด่วน บริการส่งไฟล์ระหว่างอุปกรณ์ของแอปเปิลผ่านแอร์ดรอป และเสริมความสามารถของซีรี(อังกฤษ: Siri) หรือที่แผลงเป็น สิริ
8	2 มิถุนายน พ.ศ. 2557	เพิ่มไอคลาวด์ไดรฟ์ การแจ้งเตือนแบบอินเตอร์แอกทีฟ การสนับสนุนแป้นพิมพ์จากผู้พัฒนาอื่นนอกเหนือแอปเปิล การแบ่งปันข้อมูลในอุปกรณ์ของแอปเปิลภายในครอบครัว และระบบการค้นหาใหม่
9	9 มิถุนายน พ.ศ. 2558	ปรับปรุง Siri ให้มีความแม่นยำมากขึ้นกว่าเดิม , เปลี่ยนแอปพลิเคชัน จาก Passbook เป็น Wallet , เพิ่มแอปพลิเคชั่น News , ปรับปรุงอินเทอร์เฟส Multitasking และอื่นๆ - สำหรับ iPad โดยเฉพาะ : เพิ่มฟีเจอร์ QuickType keyboard , เพิ่มการรองรับ Slide Over , เพิ่มการรองรับรูปภาพข้างในรูปภาพ (Picture in Picture) และ เพิ่มการรองรับ Split View
10	7 กันยายน พ.ศ. 2559	หน้า Lock Screen แบบใหม่, เปิดให้นักพัฒนาเข้าถึง Siri, ระบบเดาคำที่ฉลาดยิ่งขึ้น, การจัดการรูปภาพอัจฉริยะ, แผนที่ปรับปรุงใหม่,Apple Music ออกแบบใหม่หมด, HomeKit พร้อมแอพใหม่ชื่อ Home

ที่มา https://th.wikipedia.org/wiki/IOS

2.2.3 ภาษาสวิฟต์ (Swift)

Swift คือ ภาษาคอมพิวเตอร์ใหม่ที่ถูกออกแบบและ พัฒนาโดยบริษัทแอปเปิ้ล (Apple Inc.) มีจุดประสงค์เพื่อใช้สร้างและพัฒนาแอพพลิเคชั่น บนระบบปฏิบัติการ iOS และ OS X Swift ถูกสร้าง และพัฒนามานานกว่า 6 ปี โดย Apple เปิดตัวสู่สาธารณะอย่างเป็นทางการในงาน WDDC 2014 (วันที่ 2 มิถุนายน 2557) Swift ปรับปรุงมาจากภาษา C และ Objective-C โดย Swift พัฒนารูปแบบการใช้งานต่าง ๆ ให้เป็นรูปแบบของตนเอง เช่น ใช้ Int แทน interger, ใช้ Double และ Float สำหรับเลขจุดทศนิยม, ใช้ String สำหรับข้อความและใช้ Array และ Dictionary สำหรับข้อมูลที่ถูกเก็บแบบ Collection เป็นต้น โดยภาษา Objective-C เป็นภาษาที่เก่าพอสมควร เป็นภาษาที่มั่นคงมีไลบรารี่มากมายคอยอำนวยความสะดวกให้กับนักพัฒนาที่ถูกสร้างและออกแบบจากนักพัฒนาทั่วโลก แต่มีโครงสร้างที่ซับซ้อนมากจนเกินไป และการศึกษาทำความเข้าใจกับภาษา Objective-C นั้นต้องใช้เวลาพอสมควรสำหรับนักพัฒนารุ่นใหม่ Swift จึงเกิดขึ้นมาเพื่อเป็นภาษาที่ใช้ทดแทน Objective-C โดยมีแนวคิดว่า เป็นภาษาที่ง่าย จึงถูกออกแบบมาเป็นภาษาสคริปต์ และมีโครงสร้างคล้ายกับโปรแกรมภาษาจาวากับภาษา C# โดยทาง Apple พยายามที่กำจัดความเป็นภาษา C ออกไป เช่น ลบ semicolon(;) ท้ายคำสั่งออกเนื่องจาก Apple มองเป็นอะไรที่ไม่สำคัญ และต้องการออกแบบให้อยู่ในรูปแบบของ Apple (บริษัทแอปเปิ้ล, 2558)



ภาพที่ 2.5 โครงสร้างภาษา Swift

ที่มา http://nuuneoi.com/blog/blog.php?read_id=684

2.3 การวิเคราะห์ระบบงานโดยใช้ยูเอ็มแอล (Unified Modeling Language: UML)

2.3.1 หลักการและเครื่องมือที่ใช้ในการวิเคราะห์เชิงวัตถุ (Object Oriented Analysis: OOA)

กิตติ ภักดีวัฒนะกุล และกิตติพงษ์ กลมกล่อม (2547 : 83) กล่าวว่า การวิเคราะห์เชิงวัตถุ (Object Oriented Analysis: OOA) คือการใช้กระบวนการการให้ความคิดรวบยอดกับวัตถุใด ๆ เพื่อ สร้าง class (Abstraction) เพื่อการวิเคราะห์ หรือการพิจารณาว่าอะไรคือปัญหาที่ต้องแก้ในขอบเขต ของสิ่งที่กำลังพิจารณา (Problem Domain) ของเรา

หลักการที่ใช้ในการวิเคราะห์เชิงวัตถุ คือ Abstraction ชนิดต่าง ๆ เพื่อการพิจารณาหา Object Class ความสัมพันธ์ และกิจกรรมของ Object แต่ ละตัวใน Problem Domain ที่เรากำหนด

เครื่องมือที่ใช้ใน การวิเคราะห์เชิงวัตถุ คือ แผนภาพ หรือ Diagram ต่าง ๆ โดย Diagram ในการออกแบบเชิงวัตถุ (Object Oriented Analysis and Design: OOAD) นั้น แบ่งออกเป็น 2 ประเภท คือ แผนภาพแบบคงที่ (Static Diagram) และ แผนภาพแบบไดนามิก (Dynamic Diagram) ไดนามิก (Dynamic) ที่ใช้ในการนี้ ใช้เพื่อการสื่อแนวคิดหรือส่งถึงการให้ แนวคิด (Concept) กับวัตถุ (Object) ต่างๆ แผนภาพ (Diagram) จะช่วยถ่ายทอดแนวคิดดังกล่าว ออกมาเป็นภาพที่เรา และผู้อื่นสามารถเข้าใจได้ และเพื่อใช้เป็นภาพที่เราจะใช้ต่อไปได้ในขั้นตอนการออกแบบเชิงวัตถุด้วย

การเขียน Diagram ตามแนวทางของยูเอ็มแอล (Unified Modeling Language : UML) เป็นภาษาเพื่อการวิเคราะห์และออกแบบ โดยส่วนประกอบของภาษาจะประกอบด้วย Diagram หลายๆ ชนิดด้วยกัน โดยจุดมุ่งหมายที่จะทำให้ผู้อ่านสามารถวิเคราะห์และออกแบบระบบคอมพิวเตอร์ตามหลักการ OOAD ได้อย่างมีประสิทธิภาพ จะไม่นำเสนอทุก ๆ Diagram ใน UML แต่จะใช้ Diagram บางตัว เพื่อสื่อแนวทางการแสดงภาพที่เกิดจากการวิเคราะห์และออกแบบเท่านั้น โดย Diagram ดังกล่าว ได้แก่

2.3.1.1 Static Diagram คือ Diagram ที่แสดงภาพในเชิงสถิต (Static) ของ Problem Domain นั่นคือการแสดงการมีอยู่ของ Class ต่างๆและความสัมพันธ์ของ Class เหล่านั้นในระบบ โดยไม่แสดงถึงกิจกรรมที่เกิดขึ้นแต่อย่างใด ซึ่ง Static Diagram ที่ใช้สำหรับ OOA ได้แก่

- แผนภาพที่แสดงการทำงานของผู้ใช้ระบบ (Use Case Diagram) ซึ่งแสดงถึง ส่วนประกอบต่างๆ ของ Problem Domain และความสัมพันธ์ของส่วนประกอบต่างๆ เหล่านั้น ซึ่งจะ เรียกส่วนประกอบเหล่านั้นว่า Use Case ซึ่งเปรียบเสมือนเป็น Class หนึ่ง Class เช่นกัน

- คลาส ไดอะแกรม (Class Diagram) เป็น Diagram ที่แสดงถึง Class ที่มีทั้งหมดใน Problem Domain หรือ Use Case หนึ่งๆ โดยแต่ละ Class จะมี ความสัมพันธ์ในเชิง Abstraction (Agregation, Association, Generalization) กับ Class อื่นๆ อย่างน้อย 1 ความสัมพันธ์เสมอ

2.3.1.2 ไดนามิก ไดอะแกรม (Dynamic Diagram) คือ Diagram ที่แสดงภาพ ในเชิงกิจกรรม (Dynamic) ของ Problem Domain นั่นคือการแสดงถึงสิ่งที่เกิดขึ้นจากกิจกรรมของ Class ต่างๆ ที่มีใน Problem Domain (ซึ่งแสดงโดย Class Diagram) จนทำให้เกิดเป็นกิจกรรมของ Problem Domain ในที่นี้ Dynamic Diagram ที่ใช้เพื่อ OOA มีอยู่ด้วยกัน 2 แผนภาพ Diagram คือ

- ซีเควนส์ ไดอะแกรม (Sequence Diagram) เป็นDiagramที่แสดงถึงกิจกรรมรวม ของระบบโดยกิจกรรมดังกล่าวนั้นเกิดจากการเรียกใช้งาน Function ที่มีอยู่ใน Class ต่างๆ นั่นเอง

- สเตทไดอะแกรม (State Diagram) เป็น Diagram ที่แสดงถึงกิจกรรมในภาพที่เจาะจงลงไปที่ฟังก์ชัน (Function) ต่างๆ ของ Class แต่ละตัว State Diagram จะอธิบายว่าในแต่ละฟังก์ชันของ Class หนึ่งๆนั้น จะทำให้ Class มีสถานะ (State) ใดบ้าง และจะเปลี่ยนสถานะของ Class ได้เมื่อใดและอย่างไรบ้าง

การวัดว่าการวิเคราะห์ระบบคอมพิวเตอร์ด้วยการวิเคราะห์เชิงวัตถุ นั้นมีประสิทธิภาพหรือสัมฤทธิ์ผลเพียงใดนั้น คือการวัดว่า Diagram ต่างๆ ที่ได้จากขั้นตอนนี้ สามารถเล่าเรื่องราวของปัญหา (Problem Domain) ทั้งในเชิง Static และ Dynamic ได้ครอบคลุมและชัดเจนเพียงใด ในขณะเดียวกันก็ต้องวัดว่าไดอะแกรม (Diagram) ที่ได้สามารถสื่อความ ได้ง่ายหรือยาก ใช้คำอธิบายเพิ่มเติมมากน้อยเพียงใด ประสิทธิภาพที่กล่าวมาทั้งหมดมีความจำเป็นอย่างยิ่ง เพราะในบางครั้งผู้วิเคราะห์ ผู้ออกแบบ และผู้พัฒนาอาจจะเป็นคนละกลุ่มกัน หาก Diagram ที่ได้ไม่ครอบคลุม เข้าใจยาก และไม่ชัดเจน ผู้ที่นำเอา Diagram ไปใช้ ย่อมเกิดปัญหาแน่นอน

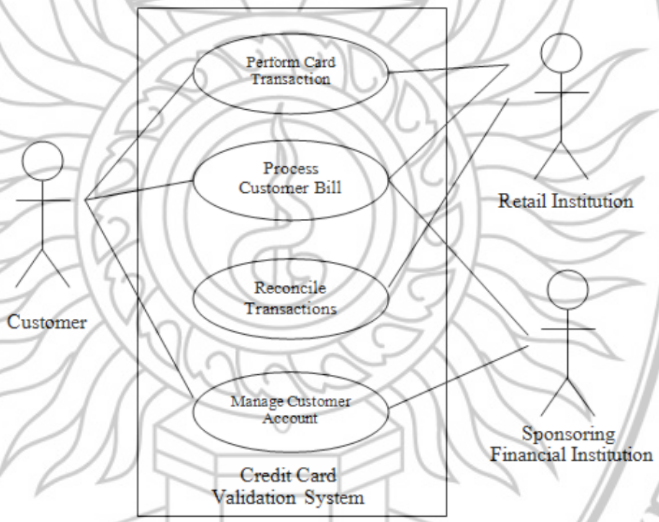
ด้วยเครื่องมือที่มี จะเห็นว่า สามารถจำลอง (Model) ปัญหาของโดเมน (Problem Domain) หรือเล่าเรื่องราว และด้วยสิ่งที่ได้จากขั้นตอนนี้เองจะเป็นวัตถุดิบเพื่อนำไปใช้ในการออกแบบ เพื่อแก้ปัญหาใน Problem Domain ด้วยคอมพิวเตอร์ ได้ต่อไป

2.3.2 ยูสเคส ไดอะแกรม (Use Case Diagram)

กิตติ ภักดีวัฒนะกุล และกิตติพงษ์ กลมกล่อม (2547 : 91) กล่าวว่าจุดประสงค์หลักของการ เขียน ยูสเคสไดอะแกรม (Use Case Diagram) ก็เพื่อเล่าเรื่องราวของปัญหาโดเมนทั้งหมดว่า มี ส่วนประกอบอะไรบ้างและเกี่ยวพันกันจนกลายเป็นระบบได้อย่างไร การเขียน Use Case Diagram จะช่วยให้ผู้พัฒนาระบบสามารถแยกแยะได้ว่าจะมีกิจกรรมอะไรที่น่าจะเกิดขึ้นในระบบบ้าง ยิ่งไปกว่านั้น Use Case Diagram ถือว่าเป็น Diagram ที่เป็นพื้นฐาน ซึ่งมีขีดความสามารถในการ

อธิบายสิ่งต่างๆ ด้วยรูปภาพที่ไม่ซับซ้อน ผู้เขียน Use Case Diagram ไม่จำเป็นต้องเป็นผู้ที่มีความรู้ทางคอมพิวเตอร์อย่างลึกซึ้ง แต่ในทางกลับกัน Use Case Diagram จะออกมาอย่างมีประสิทธิภาพหากถูกเขียนด้วยผู้ที่มีความเข้าใจในปัญหาโดเมน หรือเนื้องานตรงนั้นจริงๆ ดังนั้น ในทางปฏิบัติแล้ว เมื่อต้องการที่จะดึงความต้องการ (Requirements) หรือเรื่องราวต่างๆ ของระบบจากผู้ใช้งานหรือผู้ใช้หยุดแล้ว (End User) ผู้พัฒนาระบบมักจะมอบหมาย ผู้ใช้งานเป็นผู้เขียน Use Case Diagram หรือให้ผู้ใช้งานเข้ามาเล่าเรื่องราวและเขียน Use Case Diagram ควบคู่กันไปก็ได้ Use Case Diagram ถือ เป็นรากฐานในการเริ่มต้นการวิเคราะห์ระบบ ดังนั้น Use Case Diagram ที่สมบูรณ์และถูกต้อง ย่อมช่วยให้การวิเคราะห์ระบบมีความสมบูรณ์และถูกต้องด้วยเช่นกัน

สัญลักษณ์ที่ใช้ใน Use Case Diagram เราจะใช้รูปวงรีแทนแต่ละยูสเคส (Use Case) และใช้สัญลักษณ์รูปคนแทน แอ็คเตอร์ (Actor) หรือในบางครั้งแอ็คเตอร์ Actor อาจแทนด้วยรูปอื่น ๆ ก็ได้ เช่น รูปสิ่งก่อสร้าง เพื่อใช้แทน Actor ที่เป็นองค์กร เป็นต้น เราจะใช้เส้นตรงที่เชื่อมระหว่าง Use Case และ Actor เพื่อแสดงการใช้งาน Use Case ของ Actor นอกจากนั้น Use Case ทุก ๆ ตัว จะต้องอยู่ภายในสี่เหลี่ยมเดียวกัน ซึ่งมีชื่อระบบระบุอยู่ เพื่อแสดงความเป็น Use Case ของหัวข้อปัญหาเดียวกัน ดังภาพที่ 2.6



ภาพที่ 2.6 ยูสเคสไดอะแกรม (Use Case Diagram) ของระบบจัดการการใช้บัตรเครดิต
(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

ภาพที่ 2.6 แสดง Use Case Diagram ที่จำลองภาพการทำงานของระบบ จัดการการใช้บัตรเครดิต (Credit Card Validation System) ซึ่งผลจากการวิเคราะห์และแยกแยะระบบ ย่อย จะพบว่าระบบนี้ประกอบไปด้วย 4 ระบบย่อย

ระบบการรับและประมวลผลรายการใช้จ่ายด้วยบัตรเครดิตของลูกค้า แสดงด้วยUse Case ชื่อ Perform Card Transaction

ระบบการจัดทำใบเสร็จการชำระเงินด้วยบัตรเครดิต แสดงด้วย Use Case ชื่อ Process Customer Bill

ระบบการรวบรวมยอดการใช้จ่ายรายวัน แสดงด้วย Use Case ชื่อ Reconcile Transaction

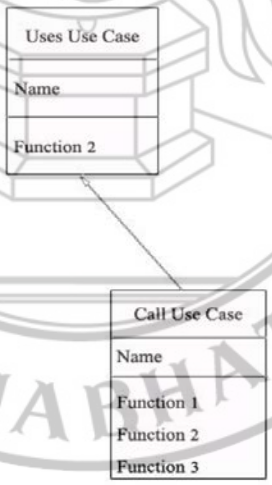
ระบบการจัดการบัญชีของลูกค้า แสดงด้วย Case ชื่อ Manage Customer Account

นอกจากนี้ ในระบบนี้ยังประกอบไปด้วย Actor คือ ลูกค้า (Customer) บริษัทที่ ออกบัตรเครดิต (Retail Institution) และสถาบันการเงินที่ให้การสนับสนุนการใช้จ่ายผ่านบัตรเครดิตซึ่ง เป็นสถาบันที่ถือบัญชีเพื่อการใช้จ่ายดังกล่าว (Sponsoring Financial Institution)

2.3.2.1 Uses และ Extends

ในหัวข้อที่แล้วเราได้ทราบมาแล้วว่า Use Case Diagram ประกอบไปด้วย Use Case ซึ่งเปรียบเสมือนระบบย่อย ๆ และมีแอ็คเตอร์ (Actor) เป็นผู้ใช้งานระบบย่อยเหล่านั้น แต่อย่างไรก็ตาม แต่ละ Use Case ภายในระบบเองก็สามารถมีความสามารถ ซึ่งกันและกันได้ โดยความสัมพันธ์ของ Use Case นั้น สามารถแบ่งออกได้เป็น 2 แบบ คือ Uses และ Extends

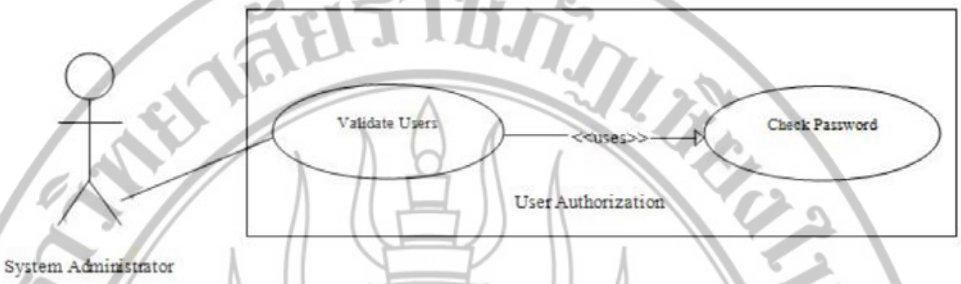
Uses หมายถึง การที่ Use Case หนึ่ง เรียกใช้งาน Use Case อีกอันหนึ่ง คล้าย ๆ กับการเรียกใช้งาน Program หลัก การเขียนสัญลักษณ์แทนการ Uses ของ Use Case นั้น ใช้สัญลักษณ์ลูกศรหัวสามเหลี่ยมใสชี้ไปยัง Use Case ที่ถูกเรียกใช้งาน และมีคำว่า <<Uses>> กำกับบนเส้นลูกศร ซึ่งสัญลักษณ์นี้เหมือนกับสัญลักษณ์ที่ใช้ในลักษณะทั่วไป (Generalization) ซึ่งเมื่อพิจารณาโดยละเอียดแล้ว จะพบว่า Uses ของ Use Case นั้นเหมือนกันกับลักษณะทั่วไป (Generalization) ทุกประการ เพราะการ Uses นั้น เราต้องรวม (include) เอาส่วนที่ถูกใช้ (Uses) ของ Use Case ที่ถูกเรียกใช้ มารวมไว้ใน Use Case ที่เรียกใช้งาน (Call Use Case) เปรียบได้กับ Specialized Class ขอให้พิจารณาภาพ 2.7 ประกอบไปด้วย



ภาพที่ 2.7 การเปรียบเทียบระหว่างการ Uses และ Generalization Abstraction

(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

การตรวจสอบ User ที่เข้ามาในระบบคอมพิวเตอร์ขององค์กรต่างๆ (แสดงด้วย User Case ชื่อ Validate Users) ต้องมีการตรวจสอบรหัสผ่าน (แสดงด้วย User Case ชื่อ Check Password) รวมอยู่ด้วย โดย Actor ของระบบนี้ก็คือผู้จัดการระบบ (System Administrator) เมื่อเขียนเป็น User Case Diagram จะได้ดังภาพที่ 2.8



ภาพที่ 2.8 ตัวอย่างของ Use Case Diagram ที่มี Uses
(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

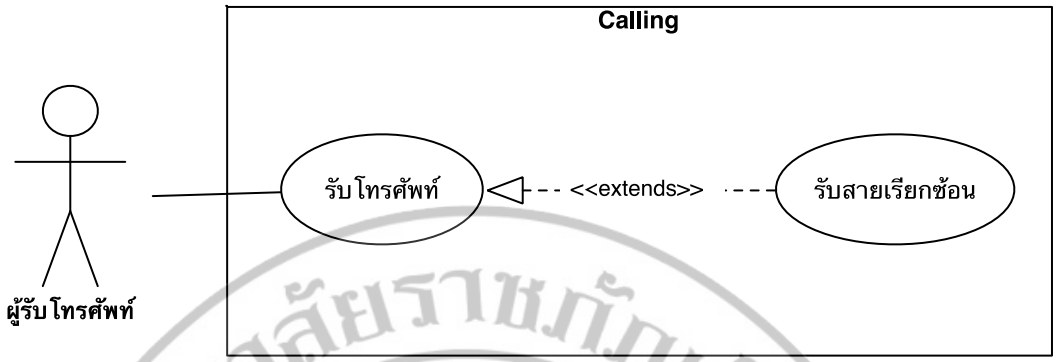
จากภาพที่ 2.8 จะเห็นว่าในการตรวจสอบผู้ที่จะเข้ามายังระบบคอมพิวเตอร์หนึ่งๆ นั้น (Validate Users) อาจจะมีหลายขั้นตอนด้วยกัน เช่น การตรวจสอบว่าผู้ใช้งานนั้นๆ เป็นผู้ใช้งานที่ได้สมัครไว้แล้วใช่หรือไม่ หรือการตรวจสอบว่า User Name ถูกต้องหรือไม่ เป็นต้น และที่แน่นอนที่สุดก็คือ การตรวจสอบรหัสผ่าน (Check Password) ซึ่ง Use Case Diagram ในภาพที่ 2.8 ได้แสดงการเรียกใช้ (Uses) Check Password โดย Validate Users สำหรับ Scenario ที่เกิดขึ้นใน Use Case Diagram นี้มีด้วยกันหลายกรณี เช่น

User ป้อน Password ที่ถูกต้อง การตรวจสอบ Password ใน Use Case ชื่อ Check Password ตรวจสอบได้ถูกต้อง ทำให้กิจกรรมใน Validate User ดำเนินต่อไปได้

User ป้อน Password ไม่ถูกต้องทำให้ Use Case ชื่อ Check Password ถูก เรียกใช้อีกหลายครั้ง จนกว่าจะถูกหรือจนกว่าจะครบ 3 ครั้ง จึงตัด User คนนั้นออกจากระบบเป็นต้น

Extends หมายถึงการที่ Use Case หนึ่งไปมีผลต่อการทำงานปกติของอีก Use Case หนึ่ง นั่นหมายถึงว่า Use Case ที่มา Extend นั้นจะมีผลทำให้การดำเนินการของ Use Case ที่ถูก Extend ถูกรบกวนหรือมีการสะดุด หรือมีการเปลี่ยนกิจกรรมไป เช่น กรณีที่บัตร ATM เสีย จะทำให้การ ถอนเงินตามปกติของ ATM ต้องล้มเหลว เป็นต้น สัญลักษณ์ที่ใช้แทน Extends ใน Use Case Diagram ก็คือ เส้นประ พร้อมหัวลูกศร โดยเริ่มจาก Use Case ที่ Extends ไปยัง Use Case ที่ถูก Extends

ขณะที่รับโทรศัพท์ปกติ หากมีสายเรียกเข้าซ้อนเข้ามา อาจต้องทำให้มีการรับสายเรียกซ้อนก่อน ซึ่งทำให้การรับสายโทรศัพท์ตามปกติต้องหยุดชะงักชั่วคราว ดังแสดงในภาพที่ 2.9



ภาพที่ 2.9 ตัวอย่างของ Use Case Diagram ที่มี Extend

ภาพที่ 2.9 เป็น Use Case Diagram ที่แสดงการรับโทรศัพท์ ซึ่งตามปกติ (ไม่มีสิ่งใดมารบกวน เช่น โทรศัพท์เสีย หรือมีสายเรียกซ้อน) กิจกรรมต่างๆ ของการรับโทรศัพท์และ สนทนา จะดำเนินไปใน Use Case รับโทรศัพท์ แต่เมื่อใดก็ตามเมื่อมีสายเรียกซ้อนเกิดขึ้น ดังที่แสดงใน รูปของ Use Case รับสายเรียกซ้อน จะทำให้กิจกรรมที่ดำเนินไปภายใน Use Case รับโทรศัพท์เกิดชะงัก (ซึ่งอาจจะหยุดสนทนาชั่วขณะ หรืออาจเปลี่ยนไปรับสายที่เรียกซ้อนมาแทน) ซึ่งในกรณีนี้ Use Case Diagram จะแสดงออกมาในรูป Use Case รับ โทรศัพท์ ถูก Extends โดย Use Case รับสาย เรียกซ้อน Scenario ที่เกิดขึ้นได้ มีอยู่ด้วยกัน 2 กรณีคือ เกิดสายเรียกซ้อนขึ้น หรือไม่เกิดสายเรียกซ้อน

2.3.3 คลาส ไดอะแกรม (Class Diagram)

2.3.3.1 หลักในการใช้สัญลักษณ์ต่าง ๆ ใน Class

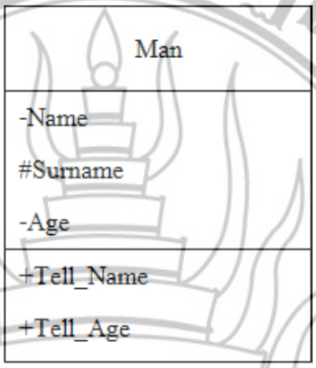
กิตติ ภักดีวัฒนะกุล และกิตติพงษ์ กลมกล่อม(2547: 103) Class Diagram คือ แผนภาพที่ใช้แสดง Class และความสัมพันธ์ในแง่ต่างๆ (Relationship) ระหว่างคลาส (Class) เหล่านั้น ซึ่งความสัมพันธ์ที่กล่าวถึงใน Class Diagram นี้ถือเป็นความสัมพันธ์เชิงสถิต (Static Relationship) หมายถึง ความสัมพันธ์ที่มีอยู่แล้วเป็นปกติในระหว่าง Class ต่าง ๆ ไม่ใช่ความสัมพันธ์ที่เกิดขึ้นเนื่องจากกิจกรรมต่าง ๆ ซึ่งเรียกว่า ความสัมพันธ์เชิงกิจกรรม (Dynamic Relationship) ความสัมพันธ์เชิงกิจกรรมนั้น เนื่องจากเราใช้ Class Diagram เพื่อ จำลองสภาพความเป็นจริงในเชิงสถิต ดังนั้นจึงจัดเอา Class Diagram ให้เป็นประเภทหนึ่งของ Static Diagram

2.3.3.2 หลักในการใช้สัญลักษณ์ต่างๆ ใน Class Diagram สิ่งที่กฎใน Class Diagram นั้น ประกอบไปด้วยกลุ่มของ Class และ กลุ่มของความสัมพันธ์ (Relationship) โดยสัญลักษณ์ที่ใช้ในการแสดง Class นั้นจะแทนด้วยสี่เหลี่ยมที่แบ่งออกเป็น 3 ส่วน โดยแต่ละส่วน (จากบนลงล่าง) จะใช้ในการแสดง ชื่อของ Class, แอททริบิวท์ (Attributes) ต่าง ๆ และฟังก์ชัน (Function) ต่าง ๆ ตามลำดับ

สัญลักษณ์ที่ใช้ในการแสดงความสัมพันธ์ (Relationship) นั้น ก็จะมีรูปแบบ แตกต่างกันออกไปตามประเภทของนามธรรม (Abstraction) ที่ทำให้เกิด Relationship นั้น ซึ่งได้แก่

Classification Abstraction, Aggregation Abstraction, Generalization Abstraction และ Association Abstraction

อนึ่ง ในการเขียนสัญลักษณ์แทน Class นั้น สิ่งที่เราต้องคำนึงถึงคือ สัญลักษณ์ที่ใช้แทน Visibility ซึ่ง Visibility นั้นสามารถแบ่งออกได้เป็น 3 ประเภทคือ Private, Protected, Public ซึ่งแทนด้วยสัญลักษณ์ -, # และ + ตามลำดับ โดยใส่ไว้ที่หน้า Attributes และ Function ที่ต้องการ (พิจารณาภาพที่ 2.10 ประกอบ)



ภาพที่ 2.10 สัญลักษณ์ Class ใน Class Diagram
(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

ความหมายของ Visibility แบบต่างๆ

- Attributes หรือ Function ที่มี Visibility แบบ Public หรือ Functions ที่ สามารถมองเห็นได้จากภายนอก สามารถเข้าไปเปลี่ยนค่า อ่านค่า หรือเรียกใช้งาน Attributes หรือ Functions นั้นได้ทันทีโดยอิสระจากภายนอก (โดยทั่วไปแล้ว Visibility แบบ Public มักจะใช้กับ Functions มากกว่า Attributes)

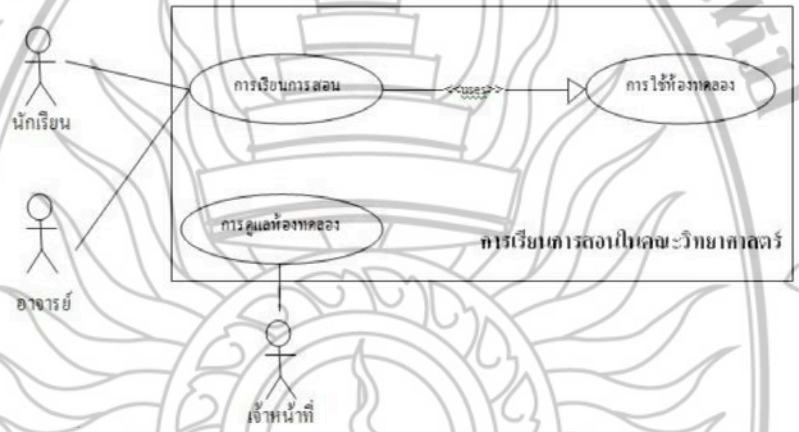
- Attributes หรือ Function ที่มี Visibility แบบ Private คือ Attributes หรือ Functions ที่ไม่สามารถมองเห็นได้จากภายนอกของ Class แต่สามารถมองเห็นได้จากภายในตัว Class เองเท่านั้น หากภายนอกต้องการที่จะเข้ามาเพื่อแก้ไขหรืออ่านค่าหรือเรียกใช้ Attributes และ Functions ที่เป็น Visibility นั้นจะทำได้เพียงวิธีเดียวคือทำผ่าน Function ที่เกี่ยวข้องสัมพันธ์กับ Attributes หรือ Functions ที่เป็น Private ดังกล่าว (โดยทั่วไปแล้ว Visibility แบบ Private มักจะใช้กับ Attributes มากกว่า Functions)

- Attributes หรือ Function ที่มี Visibility แบบ Protected คือ Attributes หรือ Function ที่สงวนไว้สำหรับการทำ Inheritance (Specialization) โดยเฉพาะ โดย Attributes หรือ Functions เหล่านี้จะเป็นเจ้าของ Superclass เมื่อทำ Inheritance แล้ว Attributes หรือ Functions ที่มี Visibility แบบ Protected จะกลายไปเป็น Private Attributes/Functions หรือ

Protected Attributes/Function ของ Subclass (ขึ้นอยู่กับข้อ กำหนด ของภาษา Programming ที่ใช้) แสดงการสร้าง Class Diagram จาก Problem Domain ต่อไปนี้

ในคณะวิทยาศาสตร์ของสถาบันการศึกษาแห่งหนึ่งมีบุคลากรหลายประเภทด้วยกัน ได้แก่ อาจารย์ นักศึกษาและเจ้าหน้าที่ โดยที่ อาจารย์แต่ละท่านมีหน้าที่ในการสอนวิชาใดวิชาหนึ่ง หรือ มากกว่า 1 วิชาก็ได้ และนักศึกษามีหน้าที่ในการศึกษาวิชาใดวิชาหนึ่ง หรือมากกว่า 1 วิชาก็ได้ในเวลาเดียวกันเจ้าหน้าที่ของภาควิชา คือเจ้าหน้าที่ประจำห้องทดลองต่าง ๆ โดยกำหนดว่าใน 1 ห้องทดลองจะต้องมีเจ้าหน้าที่ 1 คนเสมอ

จาก Problem Domain ที่กำหนด เราสามารถหากลุ่มของ Objects โดยเริ่มจาก Use Case Diagram ดังแสดงในภาพที่ 2.11 ต่อไปนี้



ภาพที่ 2.11 Use Case Diagram แสดงการเรียนการสอนในคณะวิทยาศาสตร์
(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

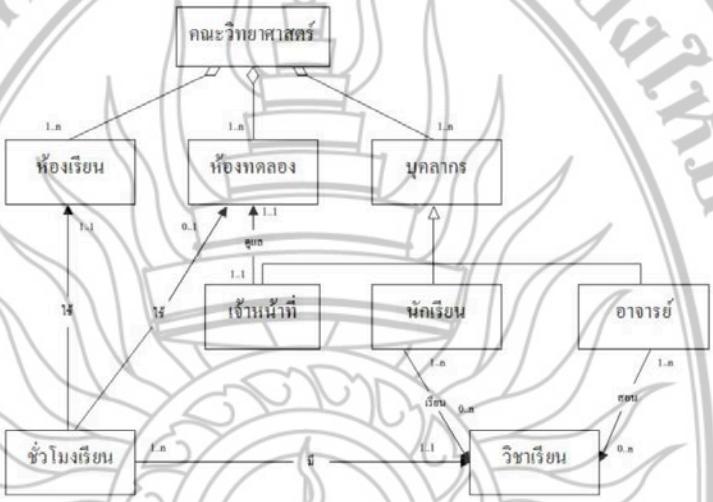
จาก Use Case ที่ได้เราสามารถหา Object หรือ Class ที่มีในแต่ละ Use Case ได้ดังนี้

ตารางที่ 2.2 แสดงการหา Object หรือ Class ที่มีในแต่ละ Use Case

Use Case	Object หรือ Class ที่หาได้จาก Use Case
การเรียนการสอน	นักเรียน อาจารย์ ห้องเรียน วิชาเรียน ชั่วโมงเรียน
การใช้ห้องทดลอง	นักเรียน อาจารย์ ห้องทดลอง
การดูแลห้องทดลอง	เจ้าหน้าที่ห้องทดลอง

(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

จากตารางจะเห็นว่า ในช่อง Object หรือ Class ที่หาได้ จะมี 2 บรรทัด โดยบรรทัดบนจะเป็น Object หรือ Class ที่เป็น Actor และในบรรทัดที่ 2 จะหมายถึง Object อื่น ๆ สามารถสร้าง Class Diagram เบื้องต้น (ยังไม่มี Attributes และ Functions) ดังภาพที่ 2.12 ที่แสดงไว้ด้านล่างนี้ โดยในภาพที่ 2.13 ได้เพิ่ม Class บาง Class ที่จำเป็น โดย Class ที่เพิ่มขึ้นได้แก่ คณะวิทยาศาสตร์ ซึ่งเกิดจาก Aggregation Abstraction (คณะวิทยาศาสตร์เกิดจาก Aggregate ของห้องเรียน ห้องทดลองและบุคลากร) และ Class บุคลากรที่เกิดจาก Generalization (บุคลากร จำแนกเป็น เจ้าหน้าที่ นักเรียนและอาจารย์)

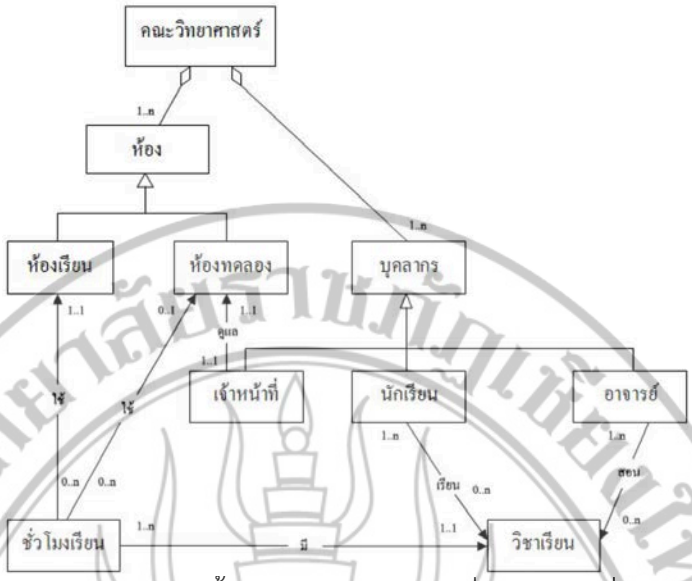


ภาพที่ 2.12 Class Diagram เบื้องต้นจาก Scenario ที่กำหนด
(กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

จากภาพที่ 2.12 จะพบว่าใน Class Diagram จะประกอบด้วย Class ที่จับต้องได้ (Tangible) และจับต้องไม่ได้ (Intangible) ซึ่ง Class ที่เป็น Tangible ได้แก่ บุคลากร (จำแนกเป็น เจ้าหน้าที่ นักเรียน และอาจารย์) ห้องเรียน และห้องทดลอง ส่วน Class ที่เป็น Intangible ได้แก่ คณะวิทยาศาสตร์ วิชาเรียน และชั่วโมงเรียน

นอกจากนี้ Problem Domain ที่กำหนดให้ เราจะพบว่า Class Diagram จะมีความสัมพันธ์กันอันเกิดจากการใช้ Abstraction แบบต่างๆ และมีการสร้าง Class เพิ่มขึ้นด้วย เพื่อเพิ่มความสมบูรณ์ให้กับ Class Diagram ได้แก่ Class ชั่วโมง เรียน เพื่อช่วยในการโยงความสัมพันธ์ระหว่างห้องเรียนกับวิชาเรียน

จาก Class Diagram ที่ได้ เราสามารถปรับเปลี่ยน Class Diagram เบื้องต้นให้สมบูรณ์ขึ้นได้อีก ดังภาพที่ 2.13



ภาพที่ 2.13 Class Diagram เบื้องต้นจาก Scenario ที่กำหนดให้ ที่ถูกทำให้สมบูรณ์ขึ้น (กิตติ ภักดีวัฒนะกุลและกิตติพงษ์ กลมกล่อม, 2547)

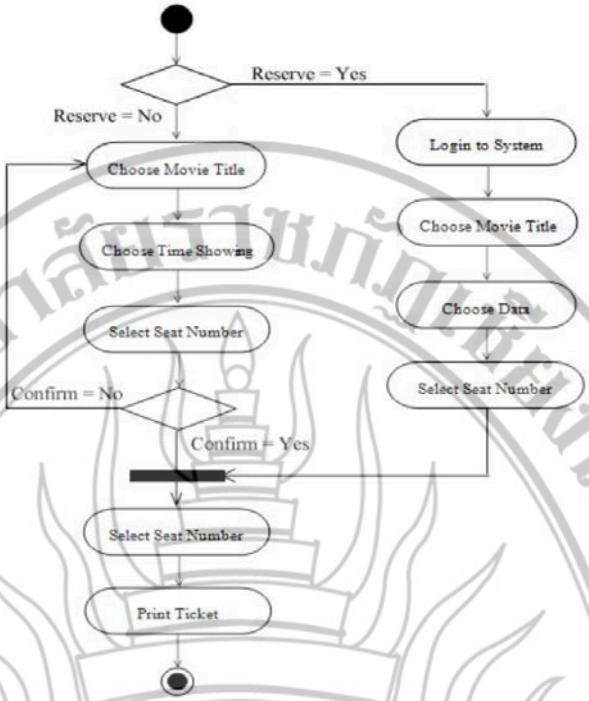
ภาพที่ 2.13 เป็นภาพแสดงการปรับปรุง Class Diagram จากภาพที่ 2.12 ซึ่งได้ใช้ Generalization Abstraction มาช่วยทำให้ Class Diagram สมบูรณ์ขึ้น (คำว่า สมบูรณ์ขึ้น คือทำให้ Class Diagram เข้าใจง่าย และครบถ้วนขึ้น อาจมีการเพิ่ม Class หรือความสัมพันธ์ขึ้นแต่ต้องไม่ทำให้เรื่องราวที่มีเปลี่ยนไป) โดยการเพิ่ม Class ห้อง ซึ่ง เป็น Generalization Class หรือ Superclass ของห้องเรียนกับห้องทดลองนั่นเอง

อย่างไรก็ตาม Class Diagram ที่ได้ จะถูกทำ Refinement ในขั้นตอนของ Object Oriented Design (OOD) เพื่อเพิ่มความสมบูรณ์ จนกระทั่งสามารถนำมาใช้เป็นต้นแบบในการพัฒนา ระบบงานในคอมพิวเตอร์ได้ในที่สุด

2.3.4 แอคทิวิตี้ไดอะแกรม (Activities Diagram)

ปรัชญา สุริวงษ์ และอดิธัญญ์ สุขประเสริฐ (2556 : 13) ใช้สำหรับแสดงลำดับการไหลของกิจกรรมต่าง ๆ โดยจะอธิบายในลักษณะของการกระทำจะมีเงื่อนไขและการตัดสินใจกำหนดไว้เพื่อควบคุมการไหลของกิจกรรมรวมถึง Message ที่รับส่งระหว่างแต่ละกิจกรรม

สัญลักษณ์แสดงด้วยสี่เหลี่ยมมนเหมือนแคปซูล เชื่อมโยงกันด้วยลูกศร เพื่อแสดงลำดับการ ทำ Activities ถัดไปได้โดยจะมีเส้นลูกศรชี้เข้ามารวมกันที่จุดเดียว (ตรงเส้นแนวนอน) นั่นคือ Activities ที่ชี้เข้ามาที่เส้นทึบดังกล่าวเสร็จหมดก่อนจึงทำ Activities ที่ชี้เข้ามาที่เส้นทึบดังกล่าวเสร็จหมดก่อน จึงทำ Activities ถัดไปได้ การแบ่งเป็น Swim Lanes เหมือนสระว่ายน้ำโดยแบ่งเป็นช่องในแนวดิ่งและ กำหนดแต่ละช่องด้วยชื่อของ Object ไว้แถวบนสุด

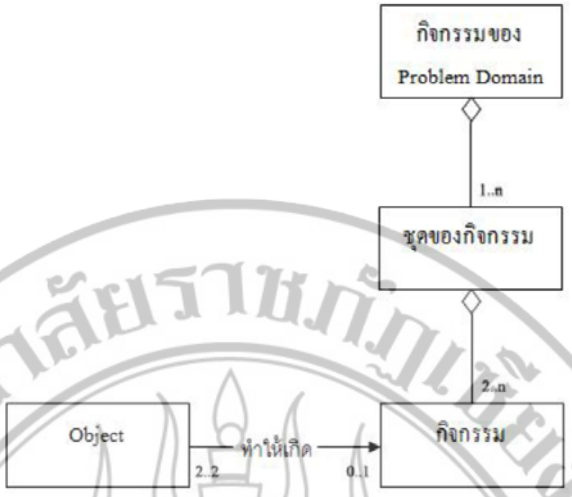


ภาพที่ 2.14 แสดงตัวอย่าง Activities Diagram
(ปรัชญา สุริวงษ์ และอดิธัญญ์ สุขประเสริฐ, 2556)

2.3.5 ซีเควนซ์ ไดอะแกรม (Sequence Diagram)

2.3.5.1 การจำลองภาพในเชิงกิจกรรม (Dynamic view) ด้วย Sequence Diagram

กิตติ ภักดีวัฒนะกุล และกิตติพงษ์ กลมกล่อม (2547 : 119) การสร้างแบบจำลอง เชิงกิจกรรม (Dynamic Model หรือ Behavioral Model) ของ Problem Domain ซึ่งก็คือการจำลอง กระบวนการที่ทำให้เกิดกิจกรรมของระบบ เกิดจากชุดของกิจกรรม ซึ่งกิจกรรมหนึ่ง ๆ นั้นเกิดจากการที่ Object หนึ่งโต้ตอบกับอัก Object หนึ่ง นั่นเอง เราสามารอธิบายการเกิดกิจกรรมของ Problem Domain ได้ด้วย Class Diagram ดังภาพที่ 2.15



ภาพที่ 2.15 Class Diagram แสดงกิจกรรมของ Problem Domain
(สุดา เธียรมนตรี, 2555)

ในการทำโอโอเอ (Object Oriented Analysis : OOA) นั้น จำเป็นต้องมีการจำลอง กิจกรรมของ Problem Domain ด้วยเช่นเดียวกัน ซึ่งการจำลองกิจกรรมของ Problem Domain เรียกว่าไดนามิกโมเดล (Dynamic Model)

ไดนามิกโมเดล (Dynamic Model) คือการจำลองกิจกรรมและลำดับของกิจกรรมที่ เกิดขึ้นหรืออาจเกิดขึ้นใน Problem Domain ให้อยู่ในรูปที่สามารถเข้าใจได้โดยง่าย เช่น การจำลองโดย ใช้คำ ประโยค หรือการจำลองโดยการเขียนภาพตามลำดับเวลาและเหตุการณ์ (เช่น ในหนังสือการ์ตูน) เป็นต้น แต่สำหรับการบรรยายกิจกรรมที่เกิดขึ้นตามหลักการของ OOAD นั้นสิ่งที่เหมาะสมที่สุดสำหรับการนี้คือ Sequence Diagram

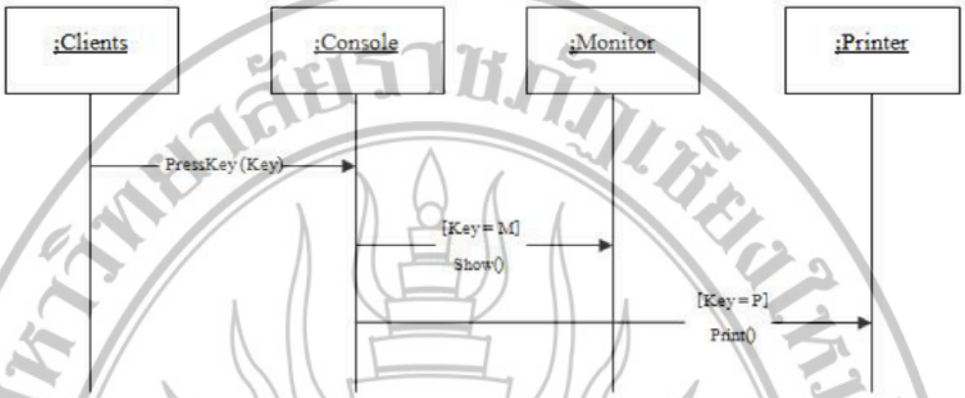
Sequence Diagram เป็น Diagram ที่ประกอบไป ด้วย Class หรือวัตถุ (Object) เส้นที่ใช้เพื่อแสดงลำดับเวลา และเส้นที่ใช้เพื่อแสดงกิจกรรมที่ เกิดขึ้นจาก Object หรือ Class ใน Diagram

ภายในซีเควนซ์ ไดอะแกรม (Sequence Diagram) จะใช้สี่เหลี่ยมแทน Class หรือ Object ซึ่งภายในกรอบสี่เหลี่ยมจะมีชื่อของ Object หรือ Class ประกอบอยู่ในรูปแบบ {Object}:Class

กิจกรรมที่เกิดขึ้นจะแทนด้วยลูกศรแนวนอนที่ชี้จาก Class หรือ Object หนึ่งไปยัง Class หรือ Object ตัวต่อไป การระบุชื่อกิจกรรมนั้นจะอยู่ในรูปแบบ { [Condition] } Function

ชื่อของกิจกรรมจะต้องเป็น Function ที่มีอยู่ใน Class หรือ Object ที่ลูกศรชี้ไป

เส้นแสดงเวลาจะแทนด้วยเส้นตรงประแนวตั้ง โดยเวลาจะเดินจากด้านบนมาสู่ด้านล่าง นั่นหมายถึงว่า ถ้าหากกิจกรรมที่เกิดขึ้นเกิดอยู่ด้านบนสุดนั่นหมายถึงกิจกรรมนั้น เป็นกิจกรรมแรก และกิจกรรมที่อยู่บริเวณต่ำลงมาจะเป็นกิจกรรมที่เกิดขึ้นต่อจากนั้น เพื่อความเข้าใจมากยิ่งขึ้น ขอให้พิจารณาภาพที่ 2.16 ต่อไปนี้



ภาพที่ 2.16 Sequence Diagram
(สุดา เธียรมนตรี, 2555)

จาก Sequence Diagram ในภาพที่ 2.16 เราสามารถอธิบายกิจกรรมที่เกิดขึ้นดังนี้

เมื่อพิจารณาลูกศรบนสุดจะพบว่า กิจกรรมแรกที่เกิดขึ้นคือ ผู้ใช้งาน (Client) กดปุ่ม (PressKey) ซึ่งอยู่บนแป้นพิมพ์ หรือ Console ซึ่ง PressKey จะต้องเป็น Function หนึ่งของ Console

เมื่อพิจารณาเส้นลูกศรที่อยู่ถัดจากเส้นแรก จะพบว่าหลังจากนั้นก็มีการตรวจสอบด้วยเงื่อนไขว่า ถ้าหาก Key ที่กด (ซึ่งเกิดจากกิจกรรมแรก) เป็น “M” ให้ Monitor แสดงผลลัพธ์ด้วย Function Show ซึ่งเป็น Function ของ Monitor

แต่ถ้า Key ที่กดเป็น P จึงให้พิมพ์ผลลัพธ์ออกทาง Printer ด้วย Function Print ซึ่งเป็น Function ที่อยู่ใน Printer (ให้พิจารณาเส้นล่างสุด)

จากภาพที่ 2.17 ที่แสดงให้เห็นนั้น เมื่อพิจารณาจะเห็นว่า Sequence Diagram จะ มีบทบาทและมีประโยชน์ในการช่วยพิจารณาว่าใน Class Diagram ที่เราสร้างขึ้น มี Function ใดของ Class ใดขาดหายไป หรือควรเพิ่มเติมเข้ามาหรือไม่ เพียงใด ซึ่งมีส่วนช่วยให้ Class ต่าง ๆ ที่เราสร้างขึ้น ใน Class Diagram มีความสมบูรณ์มากขึ้น ตัวอย่างเช่น ใน Sequence Diagram ถ้าหาก Monitor ไม่มี Function Show() มาก่อน เราก็สามารถเพิ่มกิจกรรมนี้ (ซึ่งก็คือการเพิ่ม Function Show() นั่นเอง) เข้า ไปใน Sequence Diagram ได้ ในขณะเดียวกันเราก็จะต้องย้อนกลับไปเพิ่ม Function Show() ของ Class Monitor ใน Class Diagram ด้วยเช่นกัน

การสร้าง Sequence Diagram จาก Use Case และ Class ที่มีอยู่

Sequence Diagram ของระบบ ATM

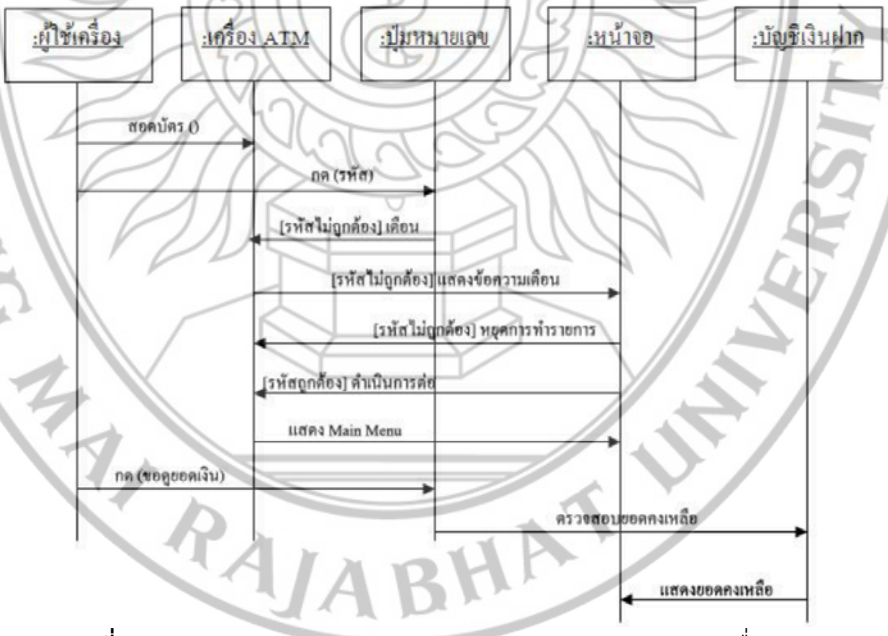
ในระบบ ATM จะประกอบด้วย Use Case ต่าง ๆ ดังนี้

- การถอนเงิน
- การดูยอดเงิน

ในระบบ ATM จะประกอบด้วย Class ต่าง ๆ ดังนี้

- เครื่อง ATM
- ปุ่มหมายเลข บนเครื่อง ATM (เป็น Aggregate ของเครื่อง ATM)
- หน้าจอบนเครื่อง ATM (เป็น Aggregate ของเครื่อง ATM)
- เครื่องจ่ายเงินหรือ Cash Dispenser (เป็น Aggregate ของเครื่อง ATM)
- เครื่องพิมพ์ Slip (เป็น Aggregate ของเครื่อง ATM)
- ผู้ใช้เครื่อง (ถือเป็น Class ที่เป็น Actor)
- เงินสด
- บัญชีเงินฝาก

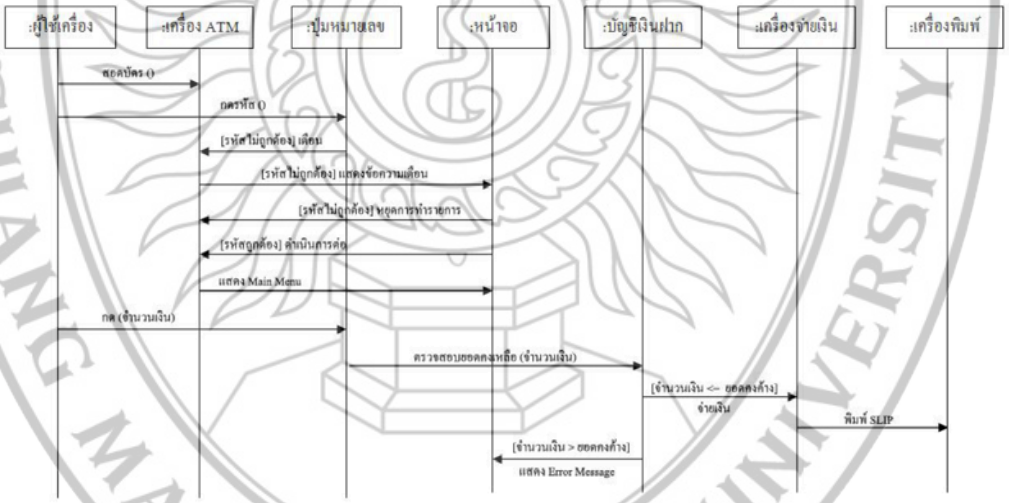
จาก Class และ Use Case (ซึ่งตามปกติควรจะแสดงด้วย Use Case Diagram และ Class Diagram ตามลำดับ แต่ไม่ได้เน้นที่การสร้าง Use Case Diagram และ Class Diagram จึงขอ ข้ามขั้นตอนนี้ไป) สามารถสร้าง Sequence Diagram สำหรับ Use Case ต่าง ๆ



ภาพที่ 2.17 Sequence Diagram ของการขอดูยอดเงินจากเครื่อง ATM
(สุดา เธียรมนตรี, 2555)

ภาพที่ 2.17 เป็นภาพของซีเควนซ์ (Sequence Diagram) ที่แสดงกิจกรรมใน Use Case การขอดูยอดเงินคงเหลือจากเครื่อง ATM โดยจะให้ผู้ใช้เครื่อง ซึ่งถือเป็น Actor เป็น Class ที่อยู่ทางซ้ายสุดของ Sequence Diagram ถัดมาคือเครื่อง ATM ซึ่ง เครื่อง ATM นั้นประกอบด้วย (Aggregate) ปุ่มหมายเลข และหน้าจอซึ่งเป็น Class ที่แสดงไว้ใน อันดับถัดมา และบัญชีเงินฝาก เป็น Class ที่อยู่ทางขวาสุดของ Sequence Diagram

กิจกรรมใน Use Case นี้เริ่มต้นที่ผู้ใช้เครื่องสอดบัตร ATM เข้าไปยังเครื่อง ATM (ซึ่งหมายความว่า ตู้ ATM ต้อง Function เพื่อการสอดบัตร ATM อยู่ในตัวเอง) ตามด้วยการกดรหัสของผู้ใช้เครื่องที่ปุ่มหมายเลข (Function กด เป็นของปุ่มหมายเลข) ซึ่งผลลัพธ์ที่ได้จากการกดนี้จะได้รหัสที่ผู้ใช้เครื่องกด ถ้ารหัสไม่ถูกต้อง เครื่องจะถูกสั่งให้เตือนผู้ใช้เครื่องว่ารหัสผิดพลาด แต่เครื่อง ATM จะเตือนได้นั้นต้องแสดงข้อความผ่านทางหน้าจอของเครื่อง ดังนั้นเครื่อง ATM จึงสั่งให้ หน้าจอแสดงข้อความเตือน หลังจากแสดงข้อความเตือนเสร็จสิ้นแล้ว เครื่องจะถูกสั่งให้หยุดทำรายการของผู้ใช้รายนี้ แต่ในทางกลับกันหากรหัสที่กดไว้นั้นถูกต้อง (พิจารณาเส้นที่ 6 จากด้านบน) จะเห็นว่าเครื่อง ATM ถูกสั่งให้ดำเนินการต่อไป โดยการดำเนินการแรกของเครื่องคือ การสั่งให้หน้าจอแสดงหน้าจอหลัก (Main Menu) หลังจากนั้นผู้ใช้จะกดปุ่มที่ปุ่มหมายเลขเพื่อระบุว่าตนต้องการขอดูยอดเงิน หลังจากนั้นบัญชีเงินฝากจะถูกสั่งให้ส่งค่ายอดคงเหลือ (ผ่านทาง Function ตรวจสอบยอดคงเหลือ) และผลที่ได้จะถูกแสดงออกทางหน้าจอซึ่งถือเป็นกิจกรรมสุดท้ายของ Use Case นี้



ภาพที่ 2.18 Sequence Diagram ของการถอนเงินด้วยเครื่อง ATM

(สุดา เธียรมนตรี, 2555)

ภาพที่ 2.18 เป็นภาพของ Sequence Diagram ที่แสดงภาพของ กิจกรรมใน Use Case การถอนเงินจากเครื่อง ATM ซึ่งเมื่อเปรียบเทียบกับภาพที่ 2.17 จะพบว่า มี Class ที่มีส่วน

ร่วมในซีเควนซ์ (Sequence Diagram) นี้เพิ่มขึ้น ซึ่งได้แก่เครื่องจ่ายเงิน (Cash Dispenser) และเครื่องพิมพ์

ถ้าสังเกตให้ดีเราจะพบว่า จากเส้นกิจกรรมเส้นบนสุดจนกระทั่งถึงเส้นที่ 7 จากด้านบน จะ เหมือนกับใน Sequence Diagram ของการดูยอดเงินฯ ทุกประการ แต่ความแตกต่างของกิจกรรมจะเกิดขึ้นหลังจากนั้น นั่นคือ

หลังจากที่ Main Menu แสดงให้เห็น ผู้ใช้จะกดจำนวนเงินที่ต้องการถอนที่ปุ่มหมายเลข (ซึ่ง Function กดอยู่ใน Class ปุ่มหมายเลขอยู่แล้ว) จากนั้นบัญชีจะถูกสั่งให้ตรวจสอบว่ามียอดคงเหลือ ในบัญชีเท่าใด ซึ่งถ้าหากว่าเงินที่มีอยู่ในบัญชีมีจำนวนมากกว่าจำนวนเงินที่ต้องการถอน เครื่องจ่ายเงินจะ ถูกสั่งให้จ่ายเงินต่อจากนั้นเครื่องพิมพ์จะถูกสั่งให้พิมพ์ SLIP แสดงการถอนเงินออกมา

ในทางกลับกัน ถ้าหากเงินในบัญชีมีจำนวนน้อยกว่าจำนวนที่ระบุว่าจะถอน หน้าจอจะถูกสั่ง ให้แสดงข้อความเตือนความผิดพลาด (Error Message)

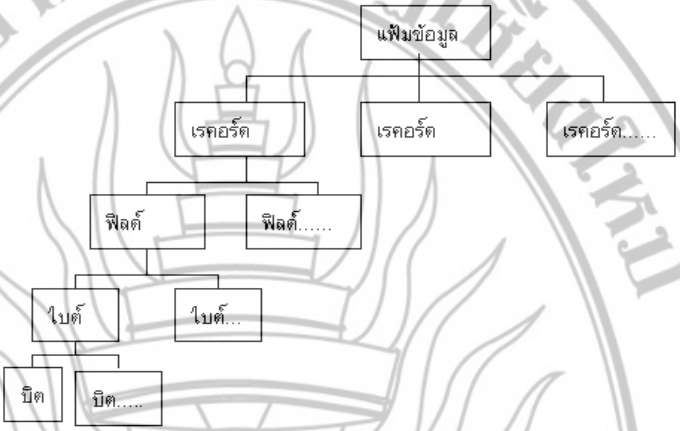
2.4 ทฤษฎีเกี่ยวกับระบบฐานข้อมูล

2.4.1 ความรู้เบื้องต้นเกี่ยวกับระบบฐานข้อมูล

ปัจจุบันระบบคอมพิวเตอร์เป็นสิ่งจำเป็นและมีความสำคัญต่อหน่วยงานทั้งภาครัฐและเอกชน จะเห็นได้จาก การประมวลผลข้อมูล (Data Processing) แบบ Manual เช่นการใช้เครื่องคิดเลข การทำงานบนกระดาษ การทำงานด้วย Worksheet ตลอดจนการเก็บเอกสารในรูปของกระดาษหรือแฟ้ม และรวบรวมเก็บไว้ในตู้หรือชั้น มีแนวโน้มลดลงอย่างรวดเร็ว ทั้งนี้เกิดจากหลายสาเหตุ ได้แก่ ปริมาณของข้อมูลมีมากขึ้นในลักษณะทวีคูณ (Exponential Growth) ราคาของเครื่องมือที่ใช้ในการประมวลผลซึ่งก็คือระบบคอมพิวเตอร์ถูกลงมาก รวมถึงการแข่งขันในเชิงของเวลาและคุณภาพในการนำข้อมูลที่ถูกประมวลผลแล้วไปใช้งานสูงขึ้น เป็นต้น

เนื่องจากข้อมูลเป็นส่วนที่สำคัญยิ่งของการประมวลผลเพราะถ้าปราศจากข้อมูลการประมวลผลก็ไม่อาจทำได้ ดังนั้นจะขอกล่าวถึงอย่างสั้นๆ ถึงความเป็นมาของการจัดการข้อมูลก่อนที่จะมาเป็นยุคของระบบฐานข้อมูล เริ่มต้นที่ระบบแฟ้มข้อมูล (File Systems) ซึ่งแนวคิดนี้พัฒนามาจากลักษณะการเก็บข้อมูลเอกสารจริงๆของหน่วยงาน นั่นคือแฟ้มเอกสารของงานหนึ่งก็คือแฟ้มกระดาษที่มีการเจาะรูตรงกลาง เช่นแฟ้มข้อมูลลูกค้า จะประกอบด้วยเอกสารที่มีรายละเอียดของลูกค้าทั้งหมดของบริษัท ได้แก่ ชื่อ ที่อยู่ เบอร์โทรศัพท์ ที่อยู่ที่สามารถติดต่อได้ ฯลฯ และบริษัทหนึ่งๆ อาจแฟ้มข้อมูลอื่นๆได้อีก เช่น แฟ้มข้อมูลเรื่องสินค้า แฟ้มข้อมูลเรื่องการขาย แฟ้มข้อมูลเรื่องเช็คธนาคาร เป็นต้น ซึ่งแต่ละแฟ้มข้อมูลก็จะประกอบด้วยข้อมูลในเรื่องเดียวกัน เมื่อนำแนวคิดดังกล่าวมาใช้ในระบบคอมพิวเตอร์ ข้อมูลที่ใช้ในการประมวลผลด้วยเครื่องคอมพิวเตอร์จะเป็นข้อมูลที่จัดเก็บ

ในรูปของแฟ้มข้อมูลในลักษณะคล้ายๆกัน แต่มีการเรียกหรือตั้งชื่อ (Terms) เฉพาะ เช่น ในแฟ้มข้อมูลลูกค้า รายละเอียดของลูกค้าแต่ละคน เรียกว่า ระเบียนหรือเรคอร์ด (Records) แฟ้มข้อมูลหนึ่งๆ จะประกอบด้วยระเบียนหลายๆ ระเบียน (มีลูกค้าหลายคน (และในแต่ละระเบียนประกอบด้วยหลายๆ ฟิลด์ (Fields) ซึ่งก็คือรายละเอียดของลูกค้านั่นเอง เราสามารถแสดงโครงสร้างของแฟ้มข้อมูลในลักษณะรูปภาพ ได้รูปที่ 1



ภาพที่ 2.19 โครงสร้างของแฟ้มข้อมูล
(นวลศรี เด่นวัฒนา, 2545)

เนื่องจากการประมวลผลกับแฟ้มข้อมูล (File Processing) เป็นเรื่องที่ค่อนข้างยุ่งยาก และเกิดความซ้ำซ้อนของข้อมูลระหว่างแฟ้มข้อมูลได้ง่าย เพราะบริษัทหนึ่ง ๆ มักมีแฟ้มข้อมูลมากกว่าหนึ่งแฟ้มและรายละเอียดข้อมูลบางแฟ้มอาจซ้ำกัน ทำให้เกิดแนวคิดในการพัฒนาระบบฐานข้อมูลขึ้นเพื่อลดปัญหาดังกล่าว

2.4.1.1 ความหมายของระบบฐานข้อมูล

ระบบฐานข้อมูล (Database Systems) หมายถึง กลุ่มของข้อมูลที่ถูกเก็บรวบรวมไว้ โดยมีความสัมพันธ์ซึ่งกันและกัน โดยไม่ได้บังคับว่าข้อมูลทั้งหมดนี้จะต้องเก็บไว้ในแฟ้มข้อมูลเดียวกันหรือแยกเก็บหลาย ๆ แฟ้มข้อมูล นั่นก็คือการเก็บข้อมูลในฐานข้อมูลนั้นเราอาจจะเก็บทั้งฐานข้อมูล โดยใช้แฟ้มข้อมูลเพียงแฟ้มข้อมูลเดียวกันได้ หรือจะเก็บไว้ในหลาย ๆ แฟ้มข้อมูล ที่สำคัญคือจะต้องสร้างความสัมพันธ์ระหว่างระเบียนและเรียกใช้ความสัมพันธ์นั้นได้ มีการกำจัดความซ้ำซ้อนของข้อมูลออกและเก็บแฟ้มข้อมูลเหล่านี้ไว้ที่ศูนย์กลาง เพื่อที่จะนำข้อมูลเหล่านี้มาใช้ร่วมกัน ควบคุมดูแลรักษาเมื่อผู้ต้องการใช้งานและผู้มีสิทธิ์จะใช้ข้อมูลนั้นสามารถดึงข้อมูลที่ต้องการออกไปใช้ได้ ข้อมูลบางส่วนอาจใช้ร่วมกับผู้อื่นได้ แต่บางส่วนผู้มีสิทธิ์เท่านั้นจึงจะสามารถใช้ได้ โดยทั่วไปองค์กรจะสร้างฐานข้อมูลไว้ เพื่อเก็บข้อมูลต่าง ๆ ขององค์กร โดยเฉพาะอย่างยิ่งข้อมูลในเชิงธุรกิจ เช่น ข้อมูลของลูกค้า ข้อมูลของสินค้า ข้อมูลของลูกจ้าง และการจ้างงาน เป็นต้น

เนื่องจากระบบฐานข้อมูลมีความซับซ้อน การควบคุมดูแลการใช้ฐานข้อมูลเป็นเรื่องที่ยุ่งยากกว่าการใช้แฟ้มข้อมูลมาก เพราะเราจะต้องตัดสินใจว่าโครงสร้างในการจัดเก็บข้อมูลควรจะเป็น เช่นไร การเขียนโปรแกรมเพื่อสร้างและเรียกใช้ข้อมูลจากโครงสร้างเหล่านี้ ถ้าโปรแกรมเหล่านี้เกิดทำงานผิดพลาดขึ้นมา ก็จะเกิดความเสียหายต่อโครงสร้างของข้อมูลทั้งหมดได้ เพื่อเป็นการลดภาวะการทำงานของผู้ใช้ จึงได้มีส่วนของฮาร์ดแวร์และโปรแกรมต่าง ๆ ที่สามารถเข้าถึงและจัดการข้อมูลในฐานข้อมูลนั้น เรียกว่า ระบบจัดการฐานข้อมูล หรือ DBMS (Database Management Systems) ระบบจัดการฐานข้อมูล คือ ซอฟต์แวร์ที่เปรียบเสมือนสื่อกลางระหว่างผู้ใช้และโปรแกรมต่าง ๆ ที่เกี่ยวข้องกับการใช้ฐานข้อมูล ซึ่งมีหน้าที่ช่วยให้ผู้ใช้เข้าถึงข้อมูลได้ง่ายสะดวกและมีประสิทธิภาพ การเข้าถึงข้อมูลของผู้ใช้อาจเป็นการสร้างฐานข้อมูล การแก้ไขฐานข้อมูล หรือการตั้งคำถามเพื่อให้ข้อมูลมา โดยผู้ใช้ไม่จำเป็นต้องรับรู้เกี่ยวกับรายละเอียดภายในโครงสร้างของฐานข้อมูล เปรียบเสมือนเป็นสื่อกลางระหว่างผู้ใช้และโปรแกรมต่าง ๆ ที่เกี่ยวข้องกับการใช้ฐานข้อมูล

2.4.1.2 ภาษาที่ใช้ในระบบฐานข้อมูล

ภาษาที่ใช้ในระบบฐานข้อมูลที่มีใช้กันในปัจจุบันจำแนกได้เป็น 3 ประเภท

1) ภาษานิยามข้อมูล (Data Definition Language: DDL) เป็นภาษาที่ใช้ในการกำหนดโครงสร้างการจัดเก็บข้อมูลและรายละเอียดฐานข้อมูลที่สร้างขึ้น

2) ภาษาควบคุม (Data Control Language: DCL) เป็นภาษาที่ใช้ในการควบคุมความถูกต้องของข้อมูล และควบคุมความปลอดภัยของข้อมูล ภาษาในส่วนนี้จะทำการป้องกันการเกิดเหตุการณ์ที่ผู้ใช้หลายคนเรียกใช้ข้อมูลพร้อมกัน โดยจะทำหน้าที่ควบคุมความถูกต้องของการใช้ข้อมูลและทำการลำดับการใช้ข้อมูลของผู้ใช้แต่ละคนและตรวจสอบสิทธิ์ในการใช้ข้อมูลนั้นๆ

3) ภาษาจัดการข้อมูล (Data Manipulation Language: DML) เป็นภาษาใช้สำหรับจัดการข้อมูลภายในฐานข้อมูล ได้แก่การเรียกค้น เพิ่ม ลบ และปรับปรุงฐานข้อมูล การเรียกดูข้อมูลจากฐานข้อมูลจะต้องผ่านคำสั่งหรือข้อความของภาษาจัดการข้อมูลหาข้อความ ซึ่งกลุ่มของข้อความเหล่านั้นมีลักษณะเป็นการถามระบบข้อมูลเพื่อให้ระบบจัดการฐานข้อมูลหาคำตอบจากข้อมูลที่เก็บไว้และตอบกลับมา กลุ่มของข้อความเหล่านั้นเรียกว่า ภาษาคำถาม (query language) แต่โดยทั่วไปแล้วคำว่า DML และ ภาษาคำถาม จะใช้แทนกันเสมอ เช่น

```
SELECT employee-name FROM employee-file
WHERE sex  =“female” AND salary GREATER THAN  5000
```

เป็นการไปเรียกดูข้อมูลชื่อของลูกจ้างที่เป็นผู้หญิงและมีเงินเดือนมากกว่า 5,000 จากฐานข้อมูลชื่อ employee-file ในบทเรียนนี้จะไม่กล่าวในรายละเอียดของภาษา DDL และ DCL แต่จะอธิบายเฉพาะวิธีการเขียนและใช้ ภาษา DML เท่านั้น

2.4.1.3 ประเภทของฐานข้อมูล

ประเภทของฐานข้อมูลโดยทั่วไปที่รู้จักกันในปัจจุบันมี 4 แบบ คือ ฐานข้อมูลแบบเครือข่าย (Network Data Model) ฐานข้อมูลแบบลำดับชั้น (Hierarchical Data Model) ฐานข้อมูลแบบสัมพันธ์ (Relational Data Model) และฐานข้อมูลแบบจำลองเชิงวัตถุ (Object Oriented Data Model) ฐานข้อมูลที่นิยมใช้กันแพร่หลายมากที่สุดในปัจจุบัน คือ ฐานข้อมูลแบบสัมพันธ์ ฐานข้อมูลแบบลำดับชั้น เป็นฐานข้อมูลแรกที่นำมาใช้ในวงการธุรกิจ ถัดมาคือฐานข้อมูลแบบเครือข่าย และปัจจุบันนี้มีฐานข้อมูลสมัยใหม่เกิดขึ้นคือฐานข้อมูลแบบจำลองเชิงวัตถุ นั่นคือฐานข้อมูลนั้นมีการนำเสนอข้อมูลและความสัมพันธ์ระหว่างข้อมูลที่แตกต่างกันไป ในบทเรียนนี้จะใช้ฐานข้อมูลแบบสัมพันธ์หรือรูปแบบตารางเป็นตัวอย่างประกอบการอธิบายการใช้ SQL

ฐานข้อมูลแบบสัมพันธ์จะนำเสนอข้อมูลและความสัมพันธ์ระหว่างข้อมูลในรูปรีเลชัน (Relation) หรือเรียกอีกอย่างหนึ่งว่าตาราง (Table) ตัวอย่างของ Relation แสดงในตารางที่ 2.3

ตารางที่ 2.3: ตารางแสดงข้อมูล Persons

LastName	FirstName	Address	City
ชิโนดม	เสรี	สามมุขธานี เมือง 745	ชลบุรี
เอี่ยมไพโรจน์	ธวัชชัย	ลงหาดบางแสน เมือง 134	ชลบุรี
เด่นวัฒนา	นวลศรี	339 พหลโยธิน 1	กทม.

(นวลศรี เด่นวัฒนา, 2545)

โครงสร้างของ Relation ประกอบด้วย

1) แถว (Row) ของข้อมูล โดยที่ ข้อมูล 1 แถว หมายถึงข้อมูล 1 รายการ ซึ่งแต่ละแถวของ Relation เรียกว่า “Tuple” โดยแต่ละแถวของข้อมูลจะประกอบไปด้วยหลาย Attributes หรือคอลัมน์ของข้อมูล

2) แต่ละคอลัมน์ (Column) ของ Relation ได้แก่คุณลักษณะของข้อมูลในแต่ละแถวซึ่งเราเรียกว่า Attribute เช่น ตาราง Persons ในตารางที่ 1 มี 5 tuples แต่ละ tuple ประกอบด้วย 4 attributes คือ นามสกุล (LastName) ชื่อ (Name) ที่อยู่ (Address) และเมือง (City)

คีย์หลัก (Primary key) เป็น Attribute หรือกลุ่มของ Attributes ที่บ่งบอกว่าข้อมูลจะต้องไม่ซ้ำกันในแต่ละแถวข้อมูลของตาราง ดังนั้นค่าของคีย์หลักจะต้องไม่ซ้ำกันในแต่ละ Tuple เช่น จาก ตาราง Persons สามารถใช้ LastName เป็นคีย์หลัก

สิ่งที่ควรทราบเกี่ยวกับข้อมูลในตาราง ของฐานข้อมูลแบบสัมพันธ์

1) ทุกตารางต้องมีคีย์หลัก

2) Tuple หรือข้อมูลในแต่ละแถวของตาราง มีข้อมูลไม่ซ้ำกัน เช่นเราจะไม่เก็บข้อมูลลูกค้าคนเดียวกันใน 2 Tuple

3) Tuple ไม่มีลำดับจากบนลงล่าง

4) Attribute ไม่มีลำดับจากซ้ายไปขวา

5) ทุก Attributes ในแต่ละตำแหน่งของ Tuple ในตารางจะมีค่าเพียงค่าเดียว จะไม่มีรายการของข้อมูล (เรียกว่า Repeating Group หรือ List of Value) ในตาราง Relation

2.4.2 รูปแบบบรรทัดฐาน (Normalization)

แนวคิดในการทำรีเลชั่นให้อยู่ในรูปแบบบรรทัดฐานถูกคิดค้นโดย อี.เอฟ.คอดด์ (E.F Codd) เป็นกระบวนการที่นำเค้าร่างของรีเลชั่นมาทำให้อยู่ในรูปแบบบรรทัดฐาน (Normal Form) เป็นทฤษฎีผู้ออกแบบฐานข้อมูลจะต้องนำมาใช้ในการแปลงข้อมูลที่อยู่ในรูปแบบที่ซับซ้อนให้อยู่ในรูปแบบที่ง่ายต่อการนำไปใช้งาน และก่อให้เกิดปัญหาน้อยที่สุดซึ่งความซ้ำซ้อนของข้อมูลในรีเลชั่นอาจทำ ให้เกิดความผิดปกติที่แบ่งออกเป็น 3 ลักษณะ คือ

- ความผิดปกติจากการเพิ่มข้อมูล (Insertion Anomaly)
- ความผิดปกติจากการลบข้อมูล (Delect Anomaly)
- ความผิดปกติจากการแก้ไขข้อมูล (Update Anomaly)

2.4.2.1 รูปแบบบรรทัดฐาน (Normal Form)

- รูปแบบบรรทัดฐานระดับที่ 1 (Fist Normal Form: 1NF) Table ใดที่อยู่ในรูปแบบ 1NF ได้ Table นั้นจะต้องไม่มีกลุ่มข้อมูลซ้ำ (Repeating Group)

- รูปแบบบรรทัดฐานระดับที่ 2 (Second Normal Form: 2NF) Table ใดที่อยู่ในรูปแบบ 2NF ได้ Table นั้นจะต้องเป็น 1NF และทุกฟิลด์ที่ไม่ใช้คีย์ (Non – key - field) ต้องขึ้นกับคีย์หลักอย่างแท้จริง ไม่ใช้ขึ้นกับเพียงบางส่วนของคีย์หลัก (ถ้าคีย์นั้นประกอบด้วยฟิลด์มากกว่าหนึ่งฟิลด์)

- รูปแบบบรรทัดฐานระดับที่ 3 (Third Normal Form: 3NF) Table ใดที่อยู่ในรูปแบบ 3NF ได้ Table นั้นจะต้องเป็น 2NF และทุกฟิลด์ที่ไม่ใช้คีย์ (Non – key - field) ต้องขึ้นกับคีย์หลักนั้น จะต้องไม่มีการขึ้นต่อกันระหว่างฟิลด์ที่ไม่ใช้คีย์ (Transitive Dependency)

- รูปแบบบรรทัดฐานบอยส์-คอดด์ (Boyce-Codd Normal Form: BCNF) คือ รีเลชั่นที่อยู่ในรูปแบบนอร์มัลระดับที่สองแล้ว และตัวกำหนดค่า (Determinant) ทุกตัวในรีเลชั่นนั้นเป็นคีย์คู่แข่ง (Candidate Key) ซึ่งในบางกรณีแม้รีเลชั่นจะอยู่ในรูปนอร์มัลระดับที่สาม ก็ยังมีโอกาสที่จะเกิดความผิดปกติจากการ จัดการข้อมูลได้ โดยที่ความผิดปกติจากการจัดการข้อมูลจะเกิดขึ้นได้ในกรณีที่เกิดเงื่อนไข 3 ประการคือ

1) รีเลชั่นมีคีย์คู่แข่งมากกว่าหนึ่งชุด

2) คีย์คู่แข่งเหล่านี้ประกอบด้วยแอททริบิวท์หลายตัวรวมกันคือเป็นคีย์ร่วม

3) คีย์คู่แข่งที่เป็นคีย์รวมเหล่านี้มีแอททริบิวท์บางตัวที่เหมือนกัน

การแปลงให้เป็นรีเลชั่นในรูปแบบของนอร์มัลบอยส์-คอด ทำโดยคัดลอกแอททริบิวท์ที่เป็นตัวที่กำหนดค่า ซึ่งไม่ใช่เป็นคีย์คู่แข่งออกมาเป็นรีเลชั่นใหม่อีกรีเลชั่นหนึ่ง โดยให้เป็นคีย์หลักของรีเลชั่นนั้นและดึงแอททริ บิวท์ที่ขึ้นกับแอททริบิวท์ที่เป็นตัวกำหนดค่านั้นออกมาอยู่ในรีเลชั่นใหม่ด้วย

- รูปแบบบรรทัดฐานระดับที่ 4 (Fourth Normal Form: 4 NF) Table ใดที่อยู่ในรูปแบบ 4NF ได้ Table นั้นจะต้องเป็น 3NF และต้องไม่ปรากฏความสัมพันธ์ระหว่าง Attribute ในรูปแบบ Multi – Value Dependency

- รูปแบบบรรทัดฐานระดับที่ 5 (Fifth Normal Form: 5NF) Table ใดที่อยู่ในรูปแบบ 5NF ได้ Table นั้นจะต้องเป็น 4NF และต้องมีคุณสมบัติ Join Dependency