การวิเคราะห์ระบบโดยใช้เทคนิคเชิงวัตถุ

จินดารัตน์ เบอรพันธุ์*

❄ ❄ ❄ ❄ ❄ ❄ ❄ ❄ ❄ ❄

บทนำ

เนื่องจากความจำเป็นและความสำคัญของคอมพิวเตอร์ทวีขึ้นเรื่อย ๆ โดยเฉพาะการใช้งานในลักษณะต่าง ๆ เช่น การใช้เป็นสถานีงาน (work station) การใช้เป็นเครื่องบริการข่ายงานบริเวณเฉพาะที่ (LAN Server) และการใช้ในลักษณะที่เป็นการประมวลผลแบบพร้อมกัน (concurrent processing)** เป็นต้น รวมทั้งการพัฒนาระบบสารนิเทศที่มีขนาดใหญ่และซับซ้อนยิ่งขึ้น ซึ่งมักส่งผลให้โครงการพัฒนาระบบสารนิเทศประสบปัญหางบประมาณไม่เพียงพอและเสร็จล่าช้ากว่ากำหนด และบางครั้งไม่สามารถตอบสนองความต้องการของผู้ใช้ได้ สาเหตุเหล่านี้ทำให้นักวิเคราะห์และออกแบบระบบสารนิเทศพยายามหาวิธีการและเทคนิคใหม่ ๆ ที่จะนำไปสู่การพัฒนาระบบสารนิเทศที่สอดคล้องกับความต้องการของผู้ใช้ และสามารถเอาชนะอุปสรรคต่าง ๆ ดังกล่าว เทคนิคหนึ่งที่กำลังได้รับความสนใจอย่างมาก และคาดว่าจะเป็นที่นิยมเพิ่มขึ้นเรื่อย ๆ คือ เทคนิคเชิงวัตถุ (object oriented technique)

จุดเริ่มต้นของเทคโนโลยีเชิงวัตถุอาจกล่าวได้ว่า เริ่มจากการที่นักวิเคราะห์และออกแบบระบบต้องการบรรยายและจำลองสิ่งต่าง ๆ ที่ปรากฏอยู่ตามสภาพที่เป็นจริง และแนวคิดในการใช้ภาษาเพื่อทำให้สามารถดำเนินกิจกรรมทั้ง 2 ลักษณะคู่กันไป เริ่มปรากฏเป็นความจริงเมื่อ Kristen Nygaard และ Ole John Dahl ได้พัฒนาภาษาการจำลองที่เรียกว่า Simula I ในปี ค.ศ. 1965 ต่อมาผู้ใช้ภาษา Simula พบว่านอกจากการจำลองแล้ว Simula ยังสามารถอำนวยความสะดวกอื่น ๆ ได้ เช่น การทำต้นแบบ เป็นต้น ดังนั้นในปี ค.ศ. 1966 จึงได้มีการปรับปรุงภาษาชุดคำสั่งดังกล่าวเป็น Simula 67 (Martin 1993: 261)

---

* ผู้ช่วยศาสตราจารย์ประจำภาควิชาบรรณารักษศาสตร์ คณะอักษรศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

** การประมวลผลได้หลาย ๆ งานพร้อม ๆ กันโดยอาศัยเทคนิคการแบ่งเวลาของโปรเซสเซอร์เพียงตัวเดียว

พัฒนาการทางเทคโนโลยีอีกด้านหนึ่งที่ส่งผลให้เทคนิคเชิงวัตถุปรากฏชัดเจนขึ้น คือ การพัฒนา Smalltalk ในช่วงต้นศตวรรษ 1970 โดย Alan Kay ได้ไปที่ Xerox ที่ Palo Alto Research Center และได้จัดตั้งกลุ่มวิจัยการศึกษา เพื่อพัฒนาแบบจำลองคอมพิวเตอร์ส่วนบุคคลชื่อ Dynabook และซอฟต์แวร์ชื่อ Smalltalk ผลจากการศึกษาพบว่า ปัญหาสำคัญปัญหาหนึ่งในการออกแบบ คือ การสื่อสารระหว่างระบบกับผู้ใช้ เพื่อให้ผู้ใช้เข้าใจโดยเฉพาะผู้ใช้ที่เป็นเด็ก Alan Kay คาดว่าในอนาคต การแสดงให้เห็นลักษณะที่เป็นอยู่ตามสภาพที่ปรากฏและการสื่อสารถึงแนวคิดต่าง ๆ ที่ประยุกต์ใช้ในการพัฒนาโปรแกรมคอมพิวเตอร์จะเป็นที่ต้องการและ Smalltalk ทำให้วิธีการเขียนโปรแกรมในแบบที่สามารถแสดงลักษณะที่เป็นจริงตามที่ปรากฏมีความเป็นไปได้ จึงกล่าวได้ว่า คำว่า “เชิงวัตถุ” (object-oriented) ได้เริ่มปรากฏขึ้นในช่วงการพัฒนา Smalltalk เช่นกัน (Martin 1993: 261-262 ; Stair: 128)

แนวคิดสำคัญของเทคนิคเชิงวัตถุ

แนวคิดและลักษณะสำคัญของเทคนิคเชิงวัตถุ ที่ควรทราบก่อนใช้เทคนิคดังกล่าวสรุปได้ดังนี้

1. การคิดของมนุษย์ (human thinking) เทคนิคเชิงวัตถุจะประยุกต์การเรียนรู้และวิธีคิดที่เป็นธรรมชาติของมนุษย์ในการจัดระเบียบความรู้และการแบ่งประเภทของมวลสารต่าง ๆ ที่ปรากฏอยู่ในโลกอย่างมีหลักเกณฑ์มาใช้กับการศึกษาระบบ ด้วยลักษณะเช่นนี้ การแสดงแผนภาพในลักษณะเชิงวัตถุจึงสามารถสื่อสารให้ผู้ใช้ระบบเข้าใจระบบที่ศึกษาได้ง่ายขึ้น (Martin 1993: 3 ; วราพร แสงเงิน 2544: 2)

2. วัตถุ (object) คำว่า “วัตถุ” ในที่นี้จะมีลักษณะเหมือนกับวัตถุทั่วไปที่พบเห็นในสภาพแวดล้อมประจำวัน คือ สามารถมองเห็น สัมผัส หรือ ให้ความรู้สึกในลักษณะต่าง ๆ ได้ รวมทั้งจะต้องมีข้อมูล (data) ที่มนุษย์ (ผู้ใช้ระบบ) ต้องการเก็บซึ่งประกอบด้วยสิ่งที่อธิบายคุณลักษณะ (attributes or properties) และพฤติกรรมที่วัตถุนั้นกระทำ (behavior) ดังนั้น “วัตถุ” จึงเป็นได้ทั้งรูปธรรม และนามธรรม ซึ่งอาจเป็นคน (พนักงาน บรรณารักษ์ ผู้ใช้) สถานที่ (ห้องสมุด บริษัท โรงงาน ร้านค้า) สิ่งของ (ทรัพยากรสารนิเทศ วารสาร สไลด์ ใบเสร็จรับเงิน เงิน) และการกระทำหรือเหตุการณ์ (การสืบค้นสารนิเทศ การยืมหนังสือ การจองตั๋วเครื่องบิน) เป็นต้น วัตถุเหล่านี้สามารถมีโครงสร้างที่สลับซับซ้อนได้ กล่าวคือ วัตถุหนึ่ง ๆ สามารถประกอบด้วยวัตถุย่อยได้ และในแต่ละวัตถุย่อยจะประกอบด้วยวัตถุย่อย ๆ ลงไปได้อีก จากคุณลักษณะเช่นนี้การสร้างแบบจำลองเชิงวัตถุในการวิเคราะห์เชิงวัตถุจึงมุ่งความสนใจไปที่ความเกี่ยวข้องและปฏิสัมพันธ์ของวัตถุตามที่ปรากฏในสภาพแวดล้อมของระบบ (Marakas 2001: 441)

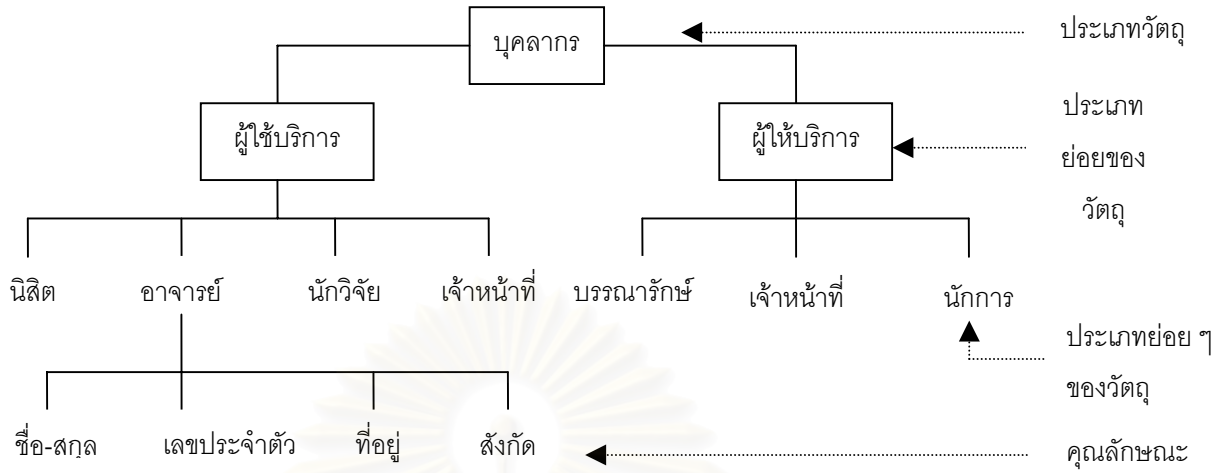
ลักษณะสำคัญของวัตถุที่จำเป็นต้องพิจารณาในการวิเคราะห์ ได้แก่ ประเภทของวัตถุ พฤติกรรม การถ่ายทอด การห่อหุ้ม และการขอร้อง

2.1 ประเภทของวัตถุ (object types) หมายถึง หมวดหมู่ของวัตถุ การวิเคราะห์วัตถุหนึ่ง ๆ ในเทคนิคเชิงวัตถุจะต้องสามารถระบุได้ว่าอยู่ในประเภทหรือหมวดหมู่ใด กล่าวอีกนัยหนึ่งได้ว่า 'วัตถุ' เป็นตัวอย่างหนึ่งในประเภทของวัตถุ (Martin 1993: 18) เช่น วารสารบรรณารักษศาสตร์ เป็นวัตถุและเป็นตัวอย่างหนึ่งของวัตถุที่จัดอยู่ในประเภท 'วารสาร' หรือ น.ส. วาสนา ทองทา และ นายวานิช ทำดี เป็นตัวอย่างหนึ่งของวัตถุประเภท 'ผู้ใช้' หรือ ใบเสร็จรับเงินเลขที่ 9454 เป็นวัตถุและเป็นตัวอย่างหนึ่งของวัตถุประเภท 'ใบเสร็จรับเงิน'

2.2 พฤติกรรม หรือ ปฏิบัติการ (behavior or operation) คือ พฤติกรรมที่ปรากฏกับวัตถุหนึ่ง ๆ หรือ การกระทำที่วัตถุหนึ่ง ๆ สามารถกระทำได้ วัตถุเป็นสิ่งที่มีคุณสมบัติตามประเภทของวัตถุนั้น ๆ และมีพฤติกรรมซึ่งแสดงให้เห็นโดยปฏิบัติการ ปฏิบัติการจึงมีความเกี่ยวเนื่องกับประเภทของวัตถุด้วย กล่าวคือ วัตถุแต่ละประเภทจะมีการกระทำที่เฉพาะของวัตถุนั้น ๆ การกระทำที่กำหนดให้กับวัตถุประเภทหนึ่งจะไม่สามารถกระทำโดยวัตถุประเภทอื่นได้ ตัวอย่าง เช่น วัตถุประเภท 'ใบเสร็จรับเงิน' จะมีพฤติกรรมหรือการกระทำต่าง ๆ ได้แก่ 1) คำนวณยอดรวมของใบเสร็จแต่ละฉบับ 2) ส่งต่อไปให้ลูกค้า และ 3) ตรวจสอบเป็นระยะ ๆ เพื่อดูว่าเป็นใบเสร็จที่ผ่านการจ่ายเงินแล้วหรือไม่ และเพิ่มอัตราดอกเบี้ย หากยังไม่มีการจ่ายเงิน เป็นต้น

พฤติกรรม หรือ ปฏิบัติการที่กำหนดให้วัตถุแต่ละประเภทนั้นเมื่อถูกนำไปเขียนเป็นโปรแกรมเชิงวัตถุ เพื่อให้สามารถกระทำการได้ตามที่กำหนดจะนิยมเรียกว่า วิธีการดำเนินการ (method)

2.3 การถ่ายทอด (inheritance) หมายถึง การถ่ายทอดคุณลักษณะและพฤติกรรมของวัตถุในระดับสูงกว่าไปยังระดับต่ำกว่า เหตุที่ต้องพิจารณาการถ่ายทอดคุณลักษณะเนื่องจากวัตถุสามารถแบ่งออกได้เป็นประเภทย่อย (subtypes) และประเภทย่อยก็สามารถแบ่งออกได้เป็นประเภทย่อย ๆ ลงไปอีก (sub-subtypes) เช่น วัตถุประเภท 'บุคคล' ซึ่งประกอบด้วยคุณลักษณะต่างๆ เช่น ชื่อ นามสกุล เลขประจำตัว ที่อยู่และสังกัดหน่วยงาน เป็นต้น สามารถแบ่งย่อยได้เป็น ผู้ใช้บริการ และ ผู้ให้บริการ และประเภทย่อยของผู้ใช้บริการ สามารถจำแนกเป็นประเภทย่อย ๆ คือ นิสิต อาจารย์ นักวิจัย และเจ้าหน้าที่ ส่วนประเภทย่อย ผู้ให้บริการ สามารถจำแนกเป็นประเภทย่อย ๆ ได้เช่นกัน คือ บรรณารักษ์ เจ้าหน้าที่ และนักการ ดังนั้นคุณลักษณะของวัตถุประเภท' บุคคล' จะปรากฏในวัตถุประเภทย่อย ๆ ของทั้งผู้ใช้บริการและผู้ให้บริการ ด้วย (แผนภาพ 1)



แผนภาพ 1

2.5 การห่อหุ้ม (encapsulation) การห่อหุ้มเป็นลักษณะการทำงานที่ผู้ใช้หรือวัตถุอื่นที่เกี่ยวข้องไม่จำเป็นต้องรู้ว่าวัตถุที่จะต้องมีปฏิสัมพันธ์ด้วยมีองค์ประกอบอะไรบ้าง หรือมีโครงสร้างซับซ้อนเพียงใด หรือมีการทำงานอย่างไร กล่าวคือ วัตถุจะซ่อนข้อมูลจากวัตถุอื่นและยอมให้ข้อมูลถูกเข้าถึงได้โดยผ่านปฏิบัติการของวัตถุที่เป็นเจ้าของข้อมูลนั้นเตรียมให้เท่านั้น ทั้งนี้เพื่อป้องกันการใช้ข้อมูลของวัตถุนั้น ๆ ตามอำเภอใจหรือใช้โดยไม่ได้ตั้งใจ (Martin 1993: 19-20)

แนวคิดเกี่ยวกับการห่อหุ้มมีผลดีต่อการพัฒนาโปรแกรมในด้านต่าง ๆ ได้แก่ 1) เกิดสภาพความเป็นส่วนจำเพาะ (modularity) ซึ่งหมายถึงการเขียนและการบำรุงรักษารหัสต้นฉบับ (source code) สามารถทำได้อย่างอิสระไม่ขึ้นอยู่กับรหัสต้นฉบับของวัตถุอื่น จึงมีลักษณะเหมือนสภาพที่เป็นอยู่จริงที่วัตถุในระบบหนึ่ง ๆ สามารถเปลี่ยนสภาพได้โดยไม่กระทบปฏิบัติการใด ๆ ของวัตถุอื่น และ 2) เกิดสภาพการซ่อนสารนิเทศ (information hiding) กล่าวคือ วัตถุหนึ่ง ๆ จะมีการเชื่อมประสานกับวัตถุอื่น ๆ โดยการสื่อสาร แต่ในขณะเดียวกันก็สามารถบำรุงรักษาสารนิเทศและวิธีดำเนินการของตนโดยการปรับเปลี่ยนได้โดยไม่มีผลกระทบต่อวัตถุที่พึ่งพามันอยู่ (Marakas 2001: 443)

2.6 การขอร้อง (request) หมายถึง การส่งข้อความบอกวัตถุให้ปฏิบัติการหรือดำเนินการใด ๆ โดยข้อความที่ส่งจะต้องประกอบด้วยชื่อวัตถุ ชื่อปฏิบัติการของวัตถุที่ต้องการให้ทำงาน และพารามิเตอร์ *ที่จำเป็นต่อการปฏิบัติงานนั้น ๆ

3. กระบวนการ และข้อมูล (process and data) ในขณะที่การวิเคราะห์และออกแบบเชิงโครงสร้างเน้นการศึกษาเกี่ยวกับกระบวนการและข้อมูลที่ปรากฏในระบบสารนิเทศโดยให้ความสำคัญกับการพิจารณาว่าระบบจะทำอะไร และจำเป็นต้องใช้ข้อมูลอะไรเพื่อดำเนินการดังกล่าวแล้วจึงพัฒนาแบบจำลองวิธีการประมวลข้อมูลเหล่านั้น แต่การวิเคราะห์และออกแบบเชิงวัตถุจะลดการให้ความสำคัญแก่กระบวนการ โดยเชื่อมโยงกระบวนการหรือวิธีการปฏิบัติกับข้อมูลเข้าเป็นส่วนเดียวกันภายในวัตถุ และวัตถุหนึ่ง ๆ จะมีลักษณะเหมือนกับสภาพที่ปรากฏอยู่จริงในระบบสารนิเทศที่ศึกษา (Laudon and Laudon 1998: 492 ; Marakas 2001: 25-26)

4.ความสามารถในการนำกลับมาใช้ (reusability) การวิเคราะห์และออกแบบเชิงวัตถุพยายามที่จะทำให้องค์ประกอบต่าง ๆ (วัตถุต่าง ๆ) ในระบบสารนิเทศสามารถนำกลับมาใช้ในระบบในลักษณะต่าง ๆ ได้อีกโดยไม่ต้องเปลี่ยนแปลง (Marakas 2001: 441) กล่าวคือ ใช้วิธีการแบ่งวัตถุเป็นพวก ๆ (class** or object type***) ในการออกแบบโปรแกรม ซึ่งจะทำให้สามารถดัดแปลงวัตถุให้เป็นไปตามความต้องการของผู้ใช้ได้ง่าย การใช้วัตถุเดิมที่มีการสร้างและเก็บไว้ในโปรแกรมห้องสมุดมาสร้างระบบใหม่เช่นนี้ทำให้ประหยัดงบประมาณ ประหยัดเวลา และเพิ่มความน่าเชื่อถือให้ระบบยิ่งขึ้น (Martin 1993: 32) เพราะวัตถุที่ผ่านการใช้จากระบบหนึ่ง ย่อมเชื่อได้ว่าผ่านการตรวจสอบและประเมินมาอย่างดีแล้ว

---

* ข้อมูลชนิดหนึ่งที่ใช้ประกอบกับคำสั่งเพื่อทำให้คำสั่งหรือโปรแกรมทำงานเฉพาะอย่าง

** การระบุประเภทของวัตถุในการพัฒนาโปรแกรม ซึ่งประกอบด้วย

1) โครงสร้างข้อมูล คือ ส่วนของข้อมูลที่เกี่ยวกับคุณลักษณะเฉพาะวัตถุ

2) วิธีดำเนินการ คือ ส่วนของปฏิบัติการที่วัตถุนั้นสามารถกระทำได้

ส่วนประกอบทั้ง 2 จะแยกออกจากกันอย่างชัดเจนทำให้การสร้าง การแก้ไข และการตรวจสอบความผิดพลาดสามารถทำได้ง่าย

*** ในขั้นตอนการวิเคราะห์ระบบ การระบุประเภทของวัตถุ นิยมเรียกว่า object type

การวิเคราะห์ในวงจรการพัฒนาระบบเชิงวัตถุ

วงจรการพัฒนาระบบเชิงวัตถุ (object-oriented system development) ประกอบด้วยขั้นตอนใหญ่ 4 ขั้นตอนดังนี้ (Stair 1996: 442-443)

1.การวิเคราะห์เชิงวัตถุ (object-orented analysis) หมายถึง การกำหนดขอบเขตของระบบที่ผู้ใช้ต้องการ โดยการให้นิยามวัตถุทั้งหมดที่มีอยู่ในสภาพแวดล้อมที่ผู้ใช้ทำงาน การวิเคราะห์จะเกี่ยวข้องกับการศึกษางานที่ปรากฏในขอบเขตของระบบที่ศึกษาและการสร้างแบบจำลองของวัตถุต่าง ๆ ที่เป็นส่วนหนึ่งของระบบดังกล่าว

2. การออกแบบเชิงวัตถุ (object-oriented design) หมายถึง การออกแบบระบบที่ได้จากการวิเคราะห์ โดยการอธิบายวัตถุในระบบทั้งหมดอย่างชัดเจน และกำหนดปฏิกิริยาที่มีต่อกันของวัตถุเหล่านั้น การออกแบบจะเกี่ยวข้องกับการพัฒนาแบบจำลองเชิงตรรก และแบบจำลองเชิงกายภาพของระบบใหม่ที่ผู้ใช้ต้องการโดยการเพิ่มรายละเอียดในแบบจำลองวัตถุที่ได้ในขั้นตอนการวิเคราะห์

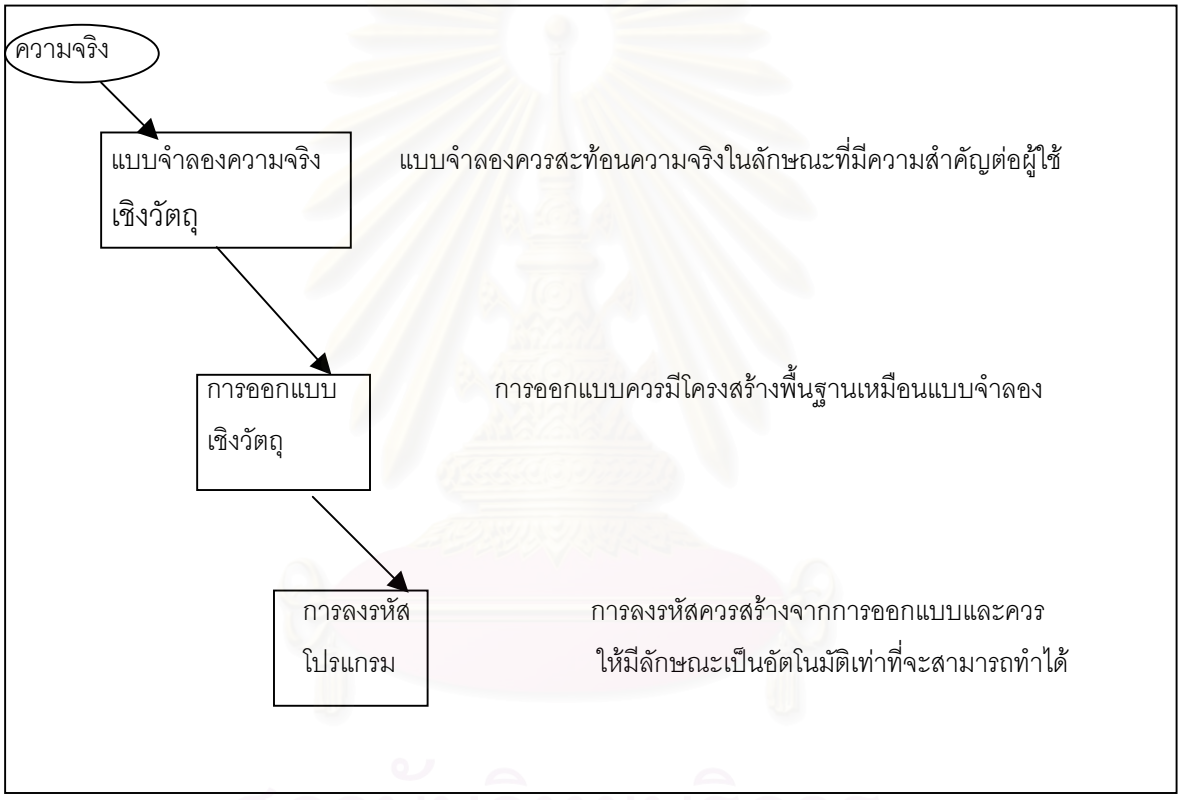
3. การเขียนโปรแกรมเชิงวัตถุ (object-oriented programming) หมายถึง วิธีการพัฒนาโปรแกรมที่ใช้การเชื่อมโยงข้อมูลกับขั้นตอนการปฏิบัติเข้าด้วยกันในวัตถุแต่ละชิ้น การเขียนโปรแกรมเชิงวัตถุจะเป็นการสร้างประเภทของวัตถุในระบบคอมพิวเตอร์ซึ่งมีลักษณะเช่นเดียวกับวัตถุในสภาพแวดล้อมจริงโดยใช้ภาษาเชิงวัตถุ หลังจากนั้นจึงสร้างเรื่องราวหรือรายละเอียดของเหตุการณ์ประกอบ เรื่องราวดังกล่าวจะเป็นเหมือนต้นฉบับซึ่งบันทึกวัตถุและพฤติกรรมที่ปรากฏกับวัตถุนั้น ๆ เมื่อมีการเรียกใช้ระบบเพื่อทำกิจกรรมต่าง ๆ โดยโปรแกรมเมอร์จะเขียนข้อความสั่งให้วัตถุแสดงพฤติกรรมดังกล่าว

4. การประเมิน (evaluation) หมายถึง การประเมินผลโดยผู้ใช้เมื่อโปรแกรมถูกนำมาใช้ในช่วงแรกเพื่อนำไปสู่การปรับปรุง ในขั้นตอนนี้ เรื่องราวหรือรายละเอียดของเหตุการณ์จะถูกเพิ่มเข้าไปในโปรแกรม เพื่อให้โปรแกรมที่พัฒนามีความสมบูรณ์ขึ้น หรืออาจต้องกลับไปวิเคราะห์ระบบใหม่ จนกว่าระบบใหม่ที่พัฒนาจะได้รับการทดสอบว่าสมบูรณ์และเป็นที่พอใจของผู้ใช้

อย่างไรก็ตามสำหรับบทความนี้จะนำเสนอรายละเอียดเฉพาะขั้นตอนการวิเคราะห์ระบบเท่านั้น

การวิเคราะห์ระบบเชิงวัตถุ

การวิเคราะห์ระบบเชิงวัตถุจะเริ่มด้วยขั้นตอนของการพิจารณาลักษณะของระบบที่ผู้ใช้ต้องการ และจำกัดขอบเขตวัตถุที่สนใจทั้งหมดในสภาพแวดล้อมของระบบที่ศึกษา แล้วสร้างแบบจำลองสภาพที่เป็นอยู่จริงโดยแบ่งประเภทวัตถุและเหตุการณ์ที่เกิดกับวัตถุเหล่านั้น ซึ่งจะนำไปสู่การออกแบบและพัฒนาโปรแกรม (แผนภาพ 2)



แผนภาพ 2

(จาก Martin 1993: 59)

ในการสร้างแบบจำลองสภาพที่ปรากฏอยู่ นักวิเคราะห์ระบบจะใช้วิธีการสร้างแผนภาพซึ่งสามารถแบ่งได้เป็น 2 ลักษณะ (Martin 1993: 61-66) ได้แก่

1. Object Structure Diagrams ใช้แสดงวัตถุ ประเภทวัตถุ ความสัมพันธ์ระหว่างวัตถุ และการถ่ายทอดคุณลักษณะของวัตถุในระบบที่วิเคราะห์ อาจเรียกว่า Object Structure Analysis (OSA) จัดเป็นแผนภาพที่เป็นตัวแทนของวัตถุและโครงสร้างของวัตถุในระบบ

2. Event Diagrams ใช้แสดงให้เห็นสิ่งที่เกิดขึ้นกับวัตถุและพฤติกรรมของวัตถุ อาจเรียกว่า Object Behavior Analysis (OBA) จัดเป็นแผนภาพที่เป็นตัวแทนของสิ่งที่เกิดขึ้นกับวัตถุ

Object Structure Analysis คือ การวิเคราะห์โครงสร้างของวัตถุ ซึ่งประกอบด้วยการพิจารณา 1)วิธีการแบ่งหมวดหมู่ของวัตถุในระบบเป็นประเภท และประเภทย่อยต่างๆ 2) ลักษณะความสัมพันธ์ระหว่างวัตถุประเภทต่าง ๆ และ 3)ส่วนประกอบของวัตถุโดยใช้แผนภาพต่าง ๆ ประกอบ ได้แก่

Generalization Diagrams

Object – Relationship Diagrams

Composed – of – Diagrams

Generalization Diagrams หมายถึง แผนภาพที่ใช้แสดงการแบ่งหมวดหมู่วัตถุในลักษณะลำดับชั้น จากลักษณะกว้าง ๆ สู่ลักษณะที่เฉพาะเจาะจงยิ่งขึ้น มี 2 ประเภท ได้แก่ Fern Diagrams และ Box Diagrams

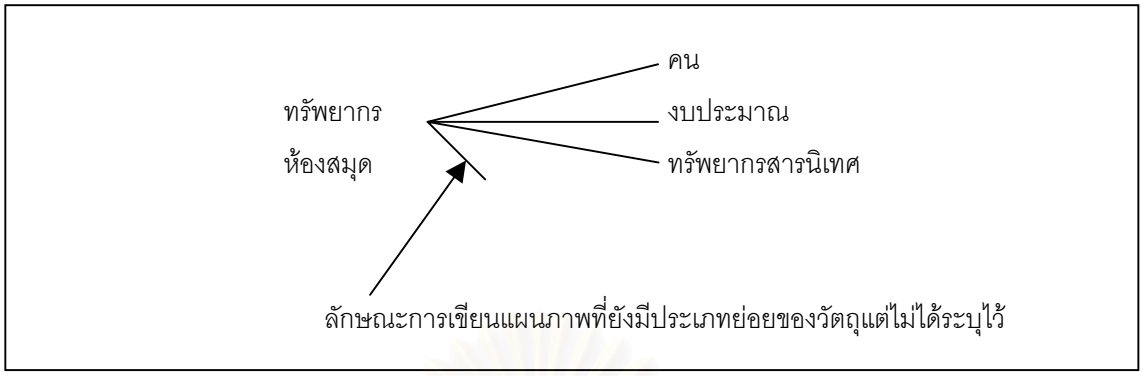
- Fern Diagrams อาจมีโครงสร้างแบบต้นไม้ (tree structure) หรือ แบบข่ายงาน (network structure) ก็ได้ สำหรับลักษณะของโครงสร้างแบบต้นไม้ วัตถุประเภทย่อยจะมีที่มาจากวัตถุประเภทเดียวกันเท่านั้น ซึ่งต่างจากลักษณะของโครงสร้างแบบข่ายงานที่วัตถุประเภทย่อยสามารถมีที่มาจากวัตถุประเภทต่าง ๆ ได้หลายประเภท (แผนภาพ 3, 4 และ 5)

แผนภาพ 3

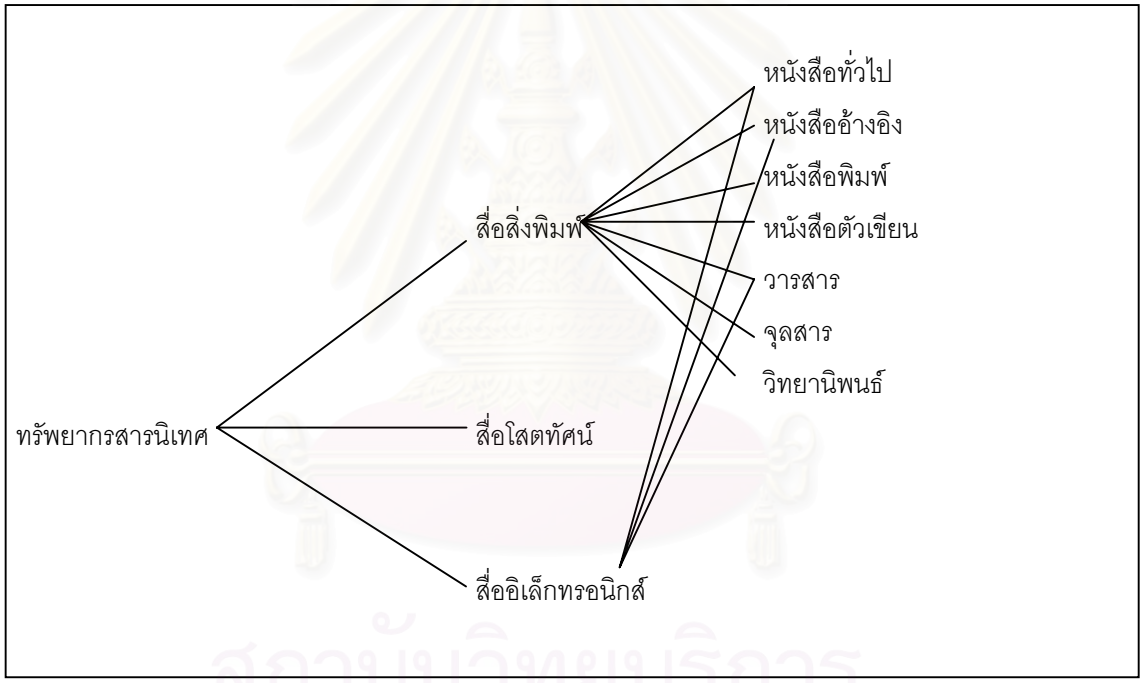
Fern Diagram ที่มีโครงสร้างแบบต้นไม้

---

* การใช้ Fern Diagram ลำดับขั้นสุดท้ายหากเป็นตัวอย่างของวัตถุประเภทนั้นจะใช้เส้นประเป็นตัวเชื่อมกับลำดับชั้นก่อนหน้า

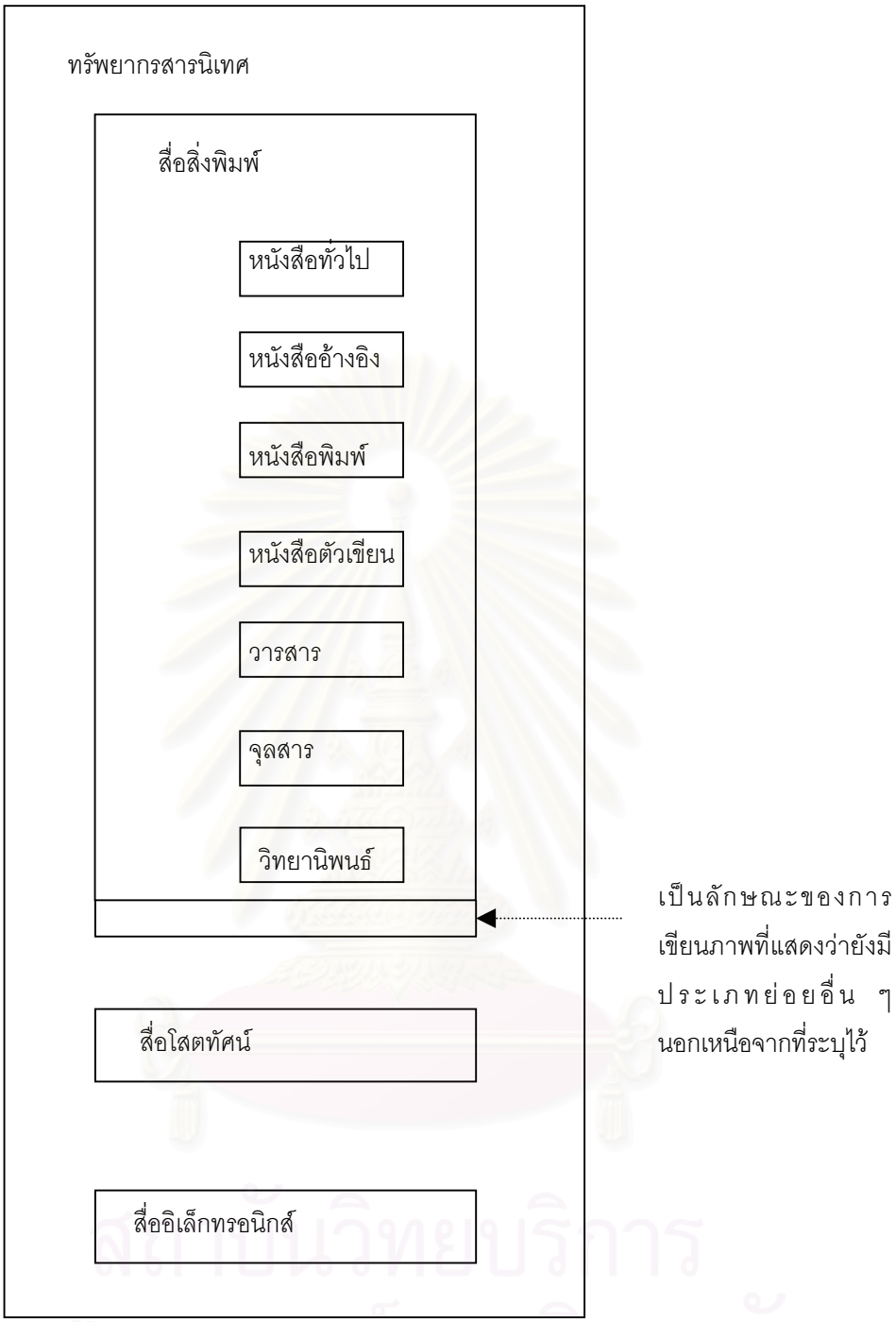


แผนภาพ 4
Fern Diagram ที่มีโครงสร้างแบบต้นไม้



แผนภาพ 5
Fern Diagram ที่มีโครงสร้างแบบข่ายงาน

- Box Diagrams คือ การใช้กล่องแทนประเภทของวัตถุ โดยซ้อนอยู่ภายในกล่องใหญ่เพื่อแทนลำดับชั้นที่ต่างกัน นิยมใช้เมื่อประเภทของวัตถุมีจำนวนไม่มาก เพราะมิฉะนั้นจะทำให้แผนภาพดูเทอะทะ (แผนภาพ 6)

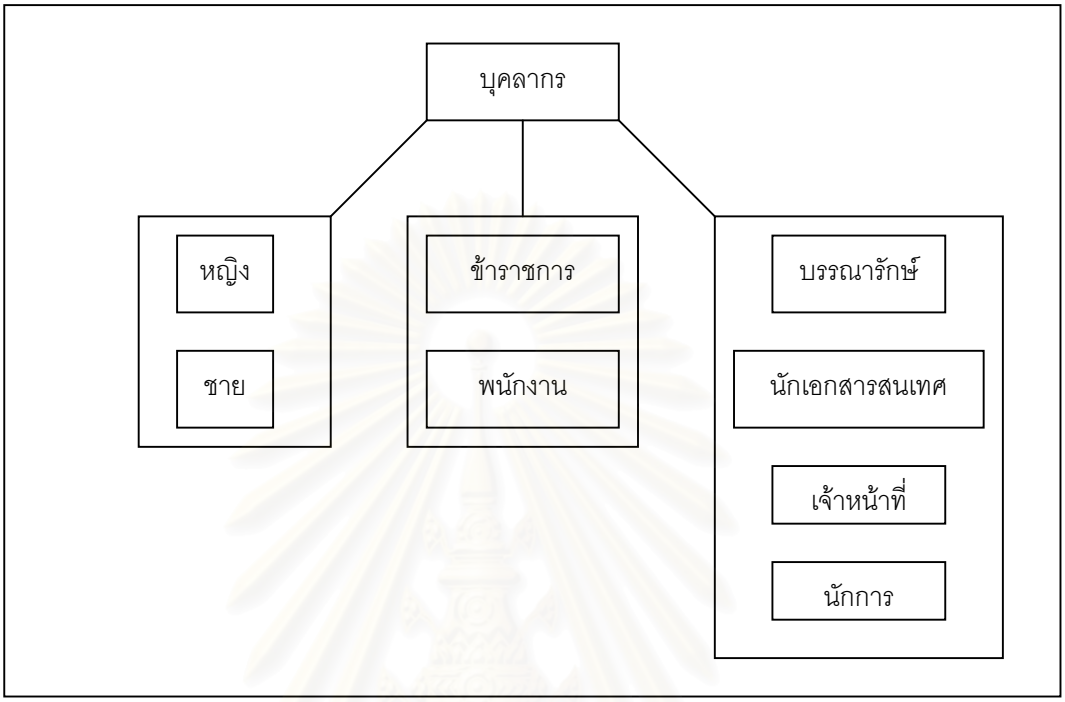


แผนภาพ 6

Box Diagram

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

นอกจากนั้น Box Diagrams ยังสามารถใช้แสดงประเภทของวัตถุที่เป็นประเภทย่อย ๆ ได้หลายประเภทในเวลาเดียวกัน (แผนภาพ 7)



แผนภาพ 7

Box Diagram แสดงประเภทของวัตถุที่เป็นประเภทย่อย ๆ

Object – Relationship Diagrams คือ แผนภาพที่แสดงความสัมพันธ์ระหว่างวัตถุแต่ละประเภท มีลักษณะคล้ายกับ Entity – Relationship Diagrams กล่าวคือ ใช้สัญลักษณ์ ▭ แทนประเภทของวัตถุและใช้สัญลักษณ์เส้นตรง ——— แสดงความสัมพันธ์ระหว่างวัตถุประเภทต่าง ๆ (Martin 1993: 82 - 88)

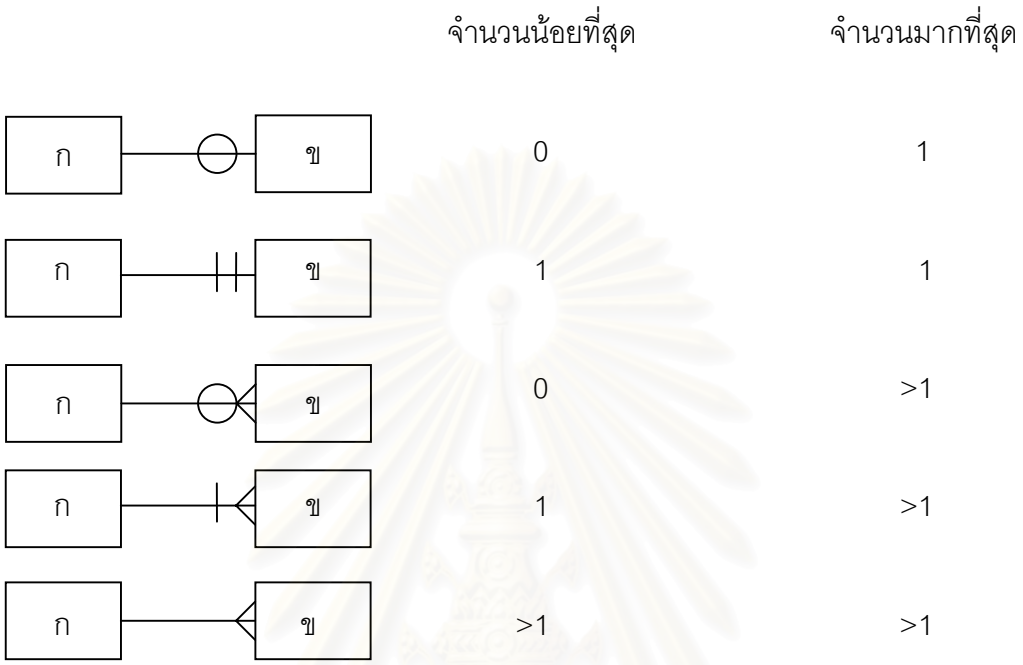
การแสดงความสัมพันธ์ระหว่างวัตถุประเภทต่าง ๆ สามารถสรุปได้เป็น 3 ลักษณะ ดังนี้

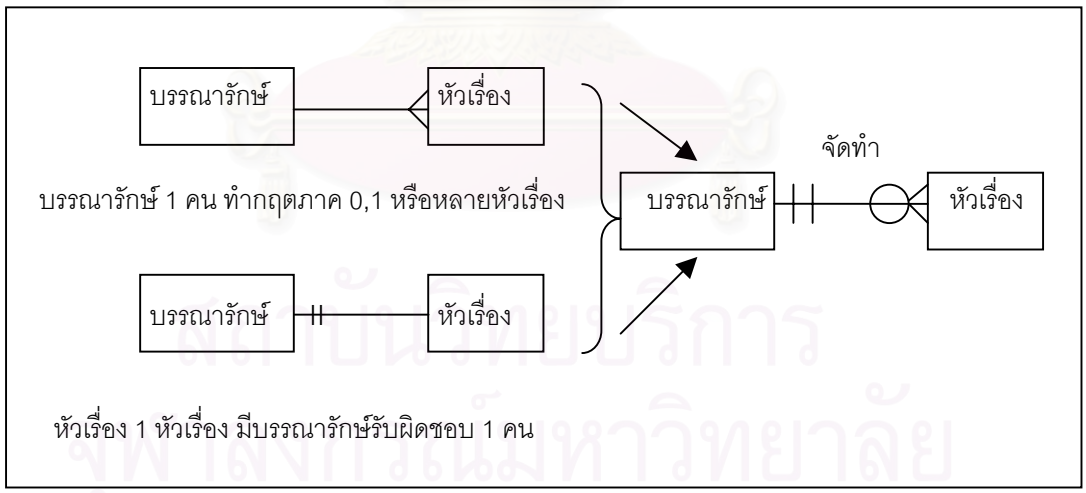
Cardinal Constraints

Instances

Subtypes and Inheritances

- Cardinal Constraints หมายถึง ข้อจำกัดเกี่ยวกับจำนวนของวัตถุประเภทหนึ่ง ๆ จำนวน 1 รายการ ที่มีความสัมพันธ์กับวัตถุประเภทอื่นจำนวน 1 รายการ สำหรับสัญลักษณ์ที่ใช้ระบุค่าของความสัมพันธ์มี 5 แบบ ดังนี้

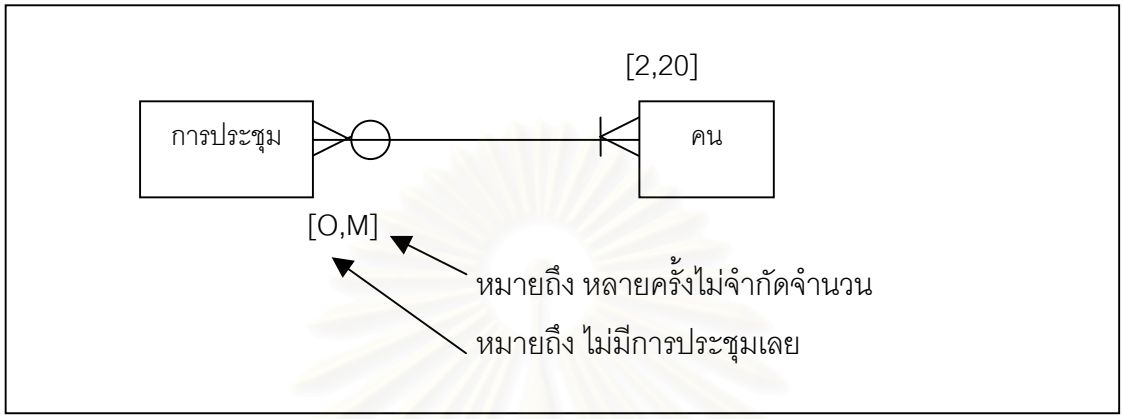




แผนภาพ 8

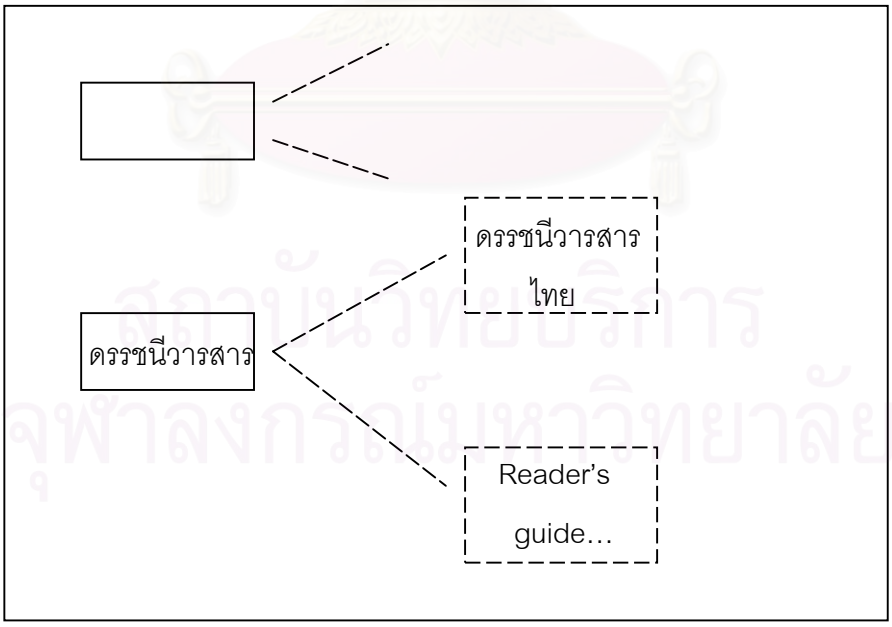
การแสดงความสัมพันธ์ระหว่างบรรณารักษ์กับหัวเรื่องกฤตภาค

ในบางครั้งการกำหนดจำนวน หรือ ค่าของความสัมพันธ์ไม่สามารถแสดงโดยใช้สัญลักษณ์ 0,1 หรือจำนวนมากกว่าได้ ดังนั้นให้ระบุค่าตัวเลขที่แน่นอนตรงตำแหน่งบนหรือใต้สัญลักษณ์ที่แสดงวัตถุประเภทนั้น เช่น การประชุมครั้งหนึ่งกำหนดว่าต้องมีบุคคลเข้าร่วมอย่างน้อย 2 คน แต่ไม่เกิน 20 คน สามารถแสดงสัญลักษณ์ได้ดังแผนภาพ 9



แผนภาพ 9

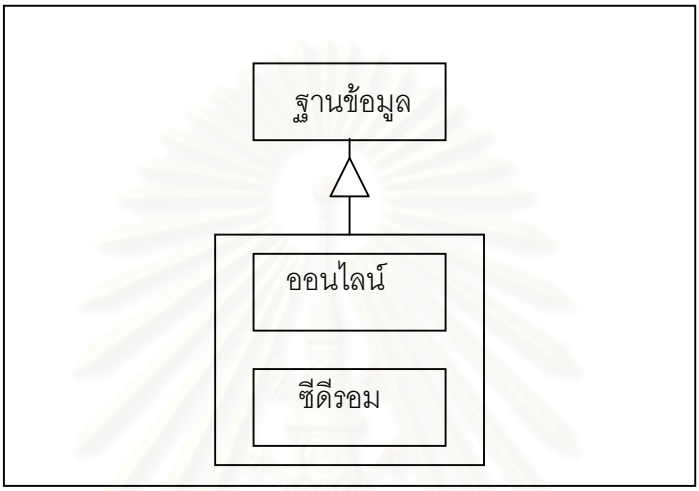
- Instances หมายถึง การแสดงความสัมพันธ์ในลักษณะการแสดงตัวอย่างของวัตถุประเภทนั้น ซึ่งจะใช้สัญลักษณ์เป็นเส้นประ และในส่วนของวัตถุที่เป็นตัวอย่าง อาจมีกรอบสี่เหลี่ยมหรือไม่มีก็ได้ ถ้ามีกรอบสี่เหลี่ยม กรอบนั้นจะต้องเป็นเส้นประเช่นกัน (แผนภาพ 10)



แผนภาพ 10

Instance แสดงตัวอย่างดรรชนีวารสาร

- Subtypes and Inheritances หมายถึง ความสัมพันธ์ในลักษณะที่เป็น subtypes และ supertypes ความสัมพันธ์ในลักษณะนี้จะมีการถ่ายทอดคุณลักษณะและพฤติกรรมของวัตถุที่เป็น supertypes ไปยัง subtype สัญลักษณ์ที่ใช้แสดง supertypes คือ กล่องสี่เหลี่ยมและสัญลักษณ์ที่ใช้แทน subtypes คือ กล่องสี่เหลี่ยมซ้อนอยู่ในกล่องสี่เหลี่ยม เหมือน Box Diagram ส่วนสัญลักษณ์ที่ใช้แทนความสัมพันธ์ คือ  ซึ่งหมายถึง ‘ เป็นประเภทย่อยของ ‘ (วราพร แสงเงิน 2544: 8) (แผนภาพ11)



แผนภาพ 11

- Composed – of – Diagrams คือ แผนภาพที่ใช้แสดงส่วนประกอบของวัตถุแต่ละประเภท โดยใช้สัญลักษณ์  (แผนภาพ 12)

แผนภาพ 12

Object Behavior Analysis คือ การวิเคราะห์พฤติกรรมวัตถุ ซึ่งประกอบด้วยการพิจารณา 1) สภาพที่วัตถุเป็นอยู่และสภาพที่เกิดความเปลี่ยนแปลง 2) เหตุการณ์ที่เกิดขึ้นและวิธีที่เหตุการณ์กระตุ้นให้เกิดพฤติกรรมหรือปฏิบัติการ และ 3) กฎเกณฑ์ที่ครอบคลุมวัตถุและพฤติกรรมของวัตถุ โดยใช้แผนภาพต่าง ๆ ได้แก่

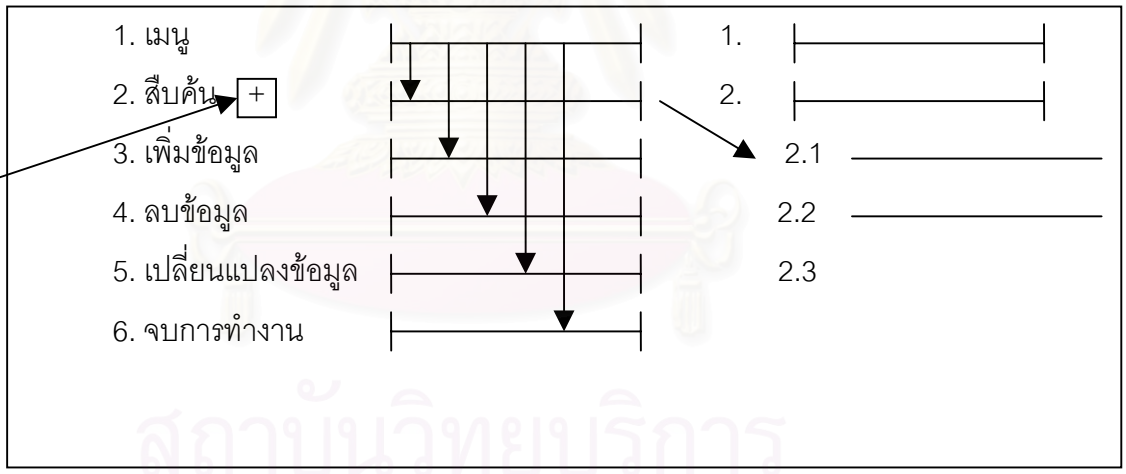
State – Transition Diagrams

Event Diagrams

Object – flow Diagrams

State – Transition Diagrams คือ แผนภาพที่แสดงให้เห็นลักษณะและลำดับของกระบวนการการเปลี่ยนแปลงของวัตถุจากสภาพหนึ่งไปเป็นอีกสภาพหนึ่ง โดยใช้สัญลักษณ์ ↓ แสดงถึงการเปลี่ยนแปลงและใช้สัญลักษณ์ ├──┤ แสดงสภาพแต่ละสภาพ (แผนภาพ 13)

การเปลี่ยนแปลงดังกล่าวอาจปรากฏขึ้นในลักษณะที่เป็นผลของสภาพที่เปลี่ยน หรือในทางกลับกันอาจเป็นตัวกระตุ้นให้เกิดการกระทำบางอย่างก็ได้



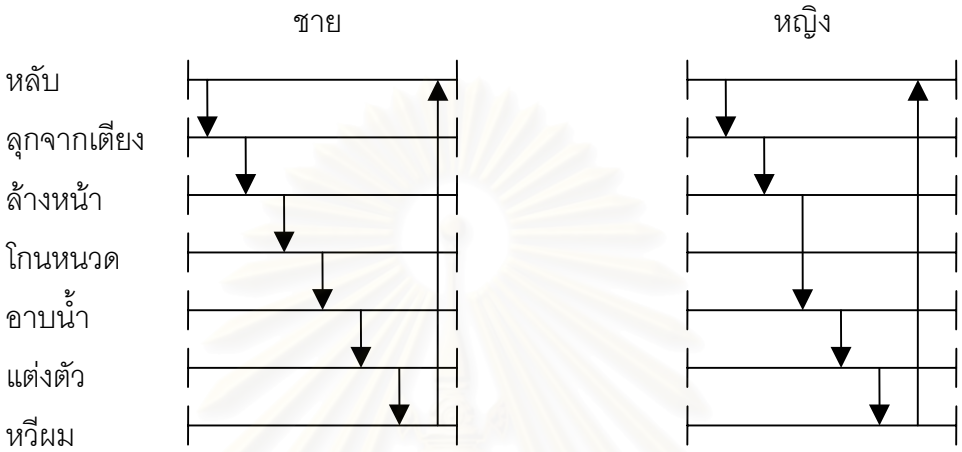
แผนภาพ 13

State-Transition Diagram

แสดงการเปลี่ยนสภาพของวัตถุ “เมนูหน้าจอคอมพิวเตอร์”

แสดงให้ทราบว่า สภาพนี้ประกอบด้วยสภาพย่อย

ดังที่ได้กล่าวมาข้างต้นว่า แนวคิดสำคัญอย่างหนึ่งของเทคนิคเชิงวัตถุ คือ ความสามารถในการนำกลับมาใช้ ดังนั้น ในขณะที่สร้างแผนภาพแสดงพฤติกรรมของวัตถุ นักวิเคราะห์ระบบจะต้องพิจารณาด้วยว่าปฏิบัติการใด และวัตถุประเภทใดสามารถนำกลับมาใช้และไม่สามารถนำกลับมาใช้ได้อีก โดยจะต้องระบุและบันทึกลักษณะที่สามารถนำกลับมาใช้ได้อีกลงในแผนภาพด้วย (แผนภาพ 14)



แผนภาพ 14

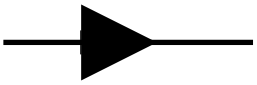
แสดงกิจวัตรในตอนเช้าของวัตถุ 'ผู้ชาย' และ 'ผู้หญิง'

(ดัดแปลงจาก Martin 1993: 106)

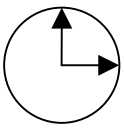
Event Diagrams คือ แผนภาพที่แสดงเหตุการณ์และการกระทำที่ถูกกระตุ้นให้เกิดขึ้นตามลำดับ โดยใช้สัญลักษณ์ต่าง ๆ ดังนี้

ใช้แทน การกระทำ (operation) เช่น การลงทะเบียนวารสาร การค้นหาสารนิเทศจากหนังสืออ้างอิง เป็นต้น

ใช้แทน เหตุการณ์ (event) เช่น วารสารที่ผ่านการลงทะเบียนแล้ว คำถามที่ผู้ใช้ถาม เป็นต้น

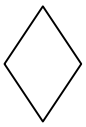


ใช้แทน ตัวกระตุ้นให้เกิดเหตุการณ์ (trigger)

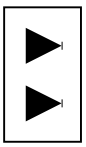


ใช้แทน เหตุการณ์ที่ระบุเวลาเพื่อกระตุ้นให้เกิดการกระทำ (clock events) เช่น เมื่อสิ้นสุดเวลาทำงานของทุกวัน ทุกๆ ต้นเดือน เป็นต้น

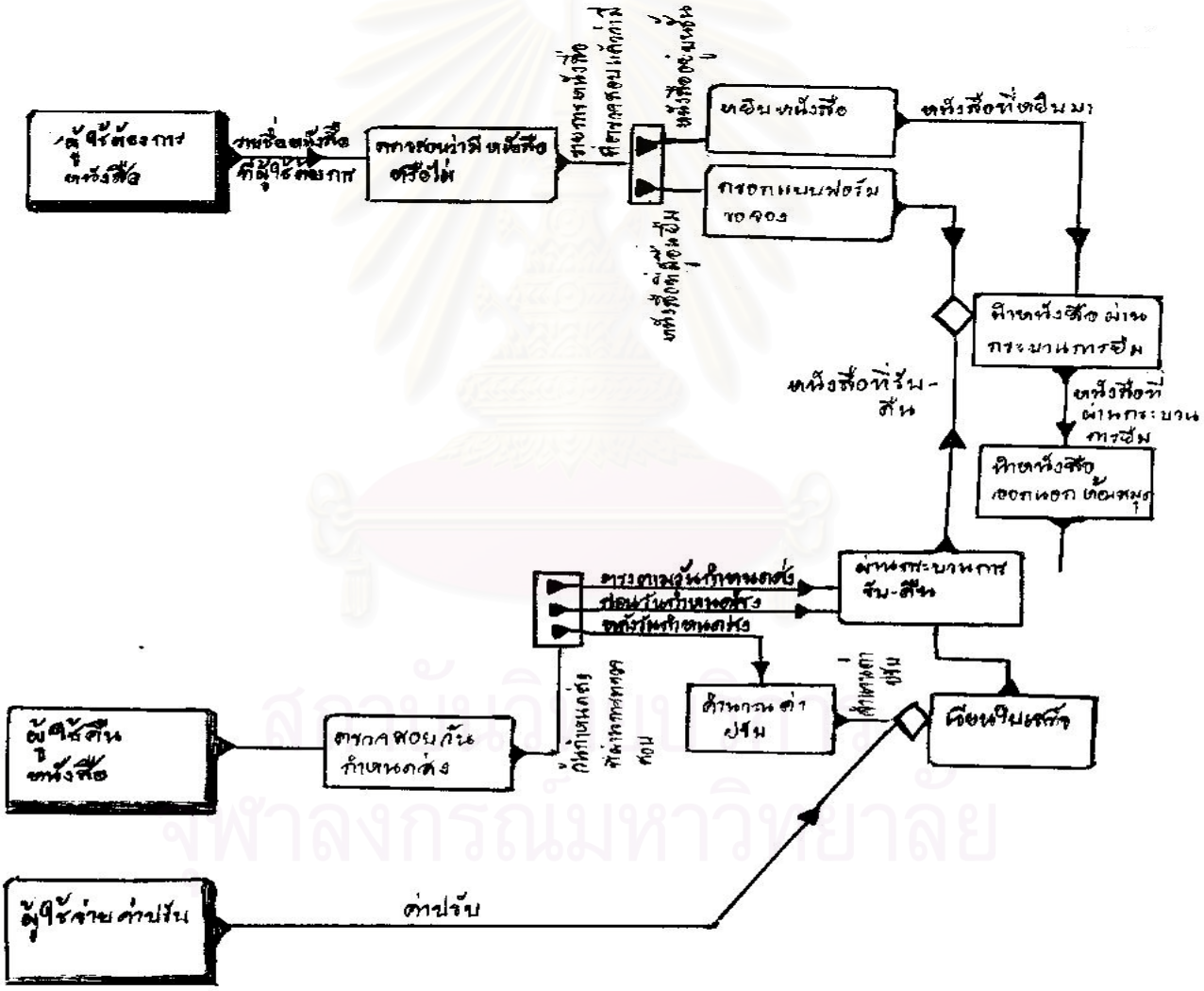
ใช้แทน การกระทำที่มาจากภายนอกระบบ (external operation) เช่น ผู้ใช้โทรศัพท์มาถามบรรณารักษ์บริการตอบคำถาม ผู้ใช้ส่งแบบฟอร์มขอใช้บริการยืมระหว่างห้องสมุด เป็นต้น



ใช้แทน การควบคุมเงื่อนไข (control condition) ซึ่งอาจเป็นเงื่อนไขเดียว หรือเงื่อนไขที่เป็นตรรกบูลีน (and , or)



ใช้แทน เหตุการณ์ย่อย (event subtype) ซึ่งหมายถึงเฉพาะเหตุการณ์ย่อยเหตุการณ์เดียวเท่านั้นที่จะเกิดขึ้น เมื่อเกิดการกระทำนั้น ๆ



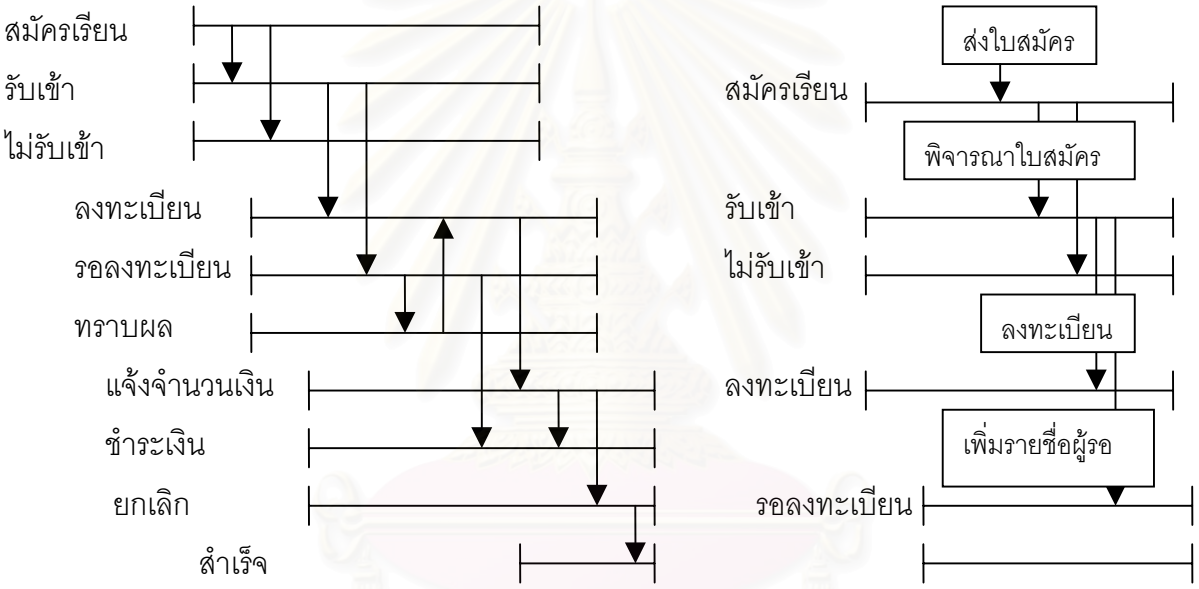
แผนภาพ 15

ตัวอย่าง Event Diagram ของการยืมหนังสือ

(ดัดแปลงจาก Martin 1993: 123)

เนื่องจากเหตุการณ์เป็นการเปลี่ยนแปลงสภาพของวัตถุที่มีความสำคัญต่อระบบที่ศึกษา Event Diagrams จึงมีการเชื่อมโยงอย่างเหนียวแน่นกับ State – Transition Diagrams (แผนภาพ 17)

นอกจากนั้นในสภาพความเป็นจริง การกระทำแต่ละครั้งจะดำเนินภารกิจที่ต้องกระทำจนเสร็จ โดยไม่เกี่ยวข้องกับส่วนอื่น ๆ และการกระทำจะถูกกระตุ้นโดยเหตุการณ์เดียว หรือหลายเหตุการณ์ ก็ได้ โดยที่การกระทำนั้น ๆ ไม่มีส่วนรับรู้ว่าเหตุการณ์ใดเป็นตัวกระตุ้น และกระตุ้นทำไม จึงกล่าวได้ว่า การกระทำไม่ได้ตระหนักถึงเหตุและผลของการกระทำ แนวคิดเช่นนี้ เป็นสิ่งจำเป็นสำหรับการจะนำการกระทำไปประยุกต์ใช้ในสภาพที่แตกต่างออกไป หรือใช้ในโปรแกรมอื่น (แผนภาพ 16)



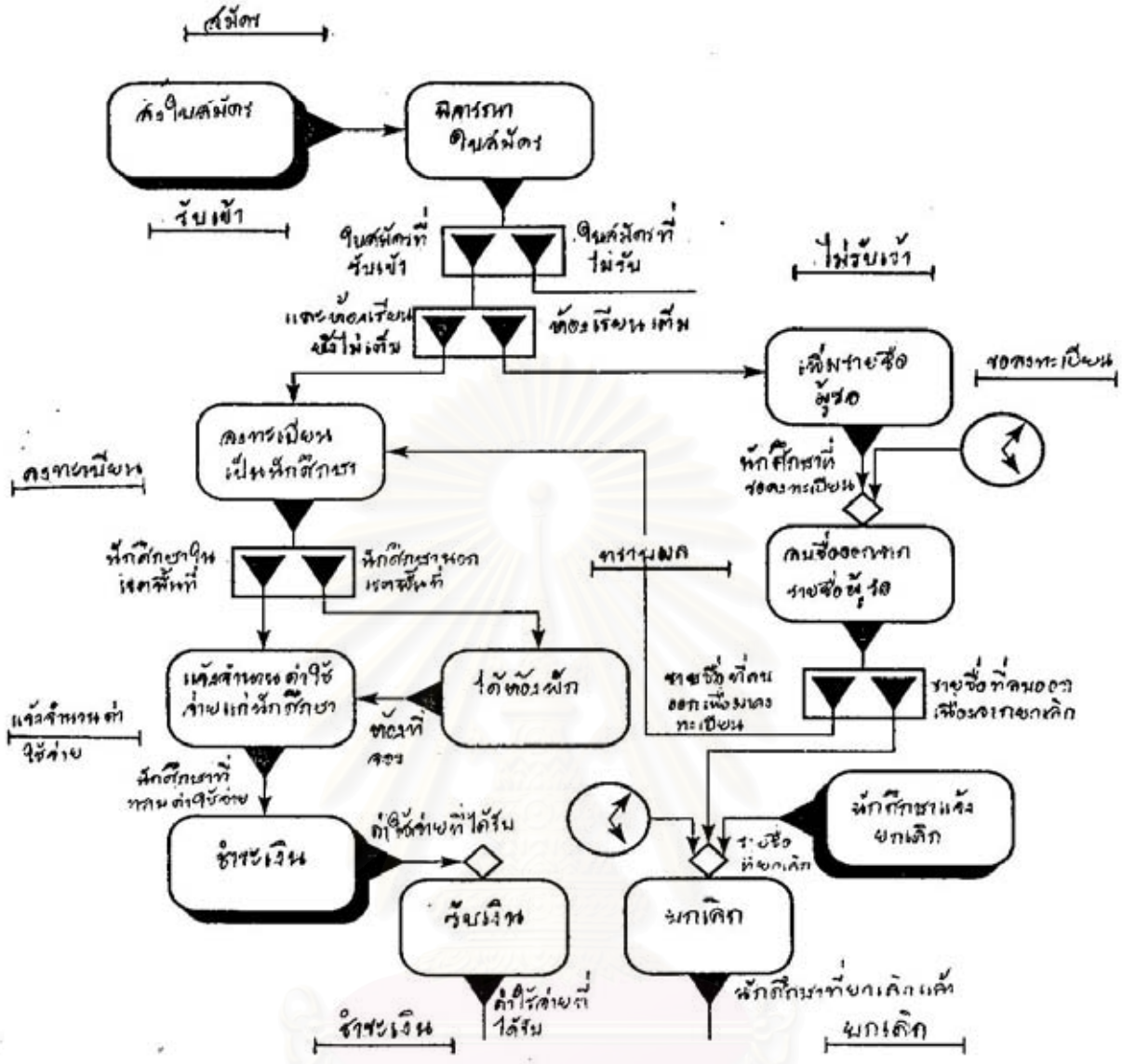
แผนภาพ 16

State-Transition Diagram

แสดงการเปลี่ยนสภาพภาพของวัตถุ "นักศึกษา" ระหว่างกระบวนการลงทะเบียน

(จาก Martin 1993: 162)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

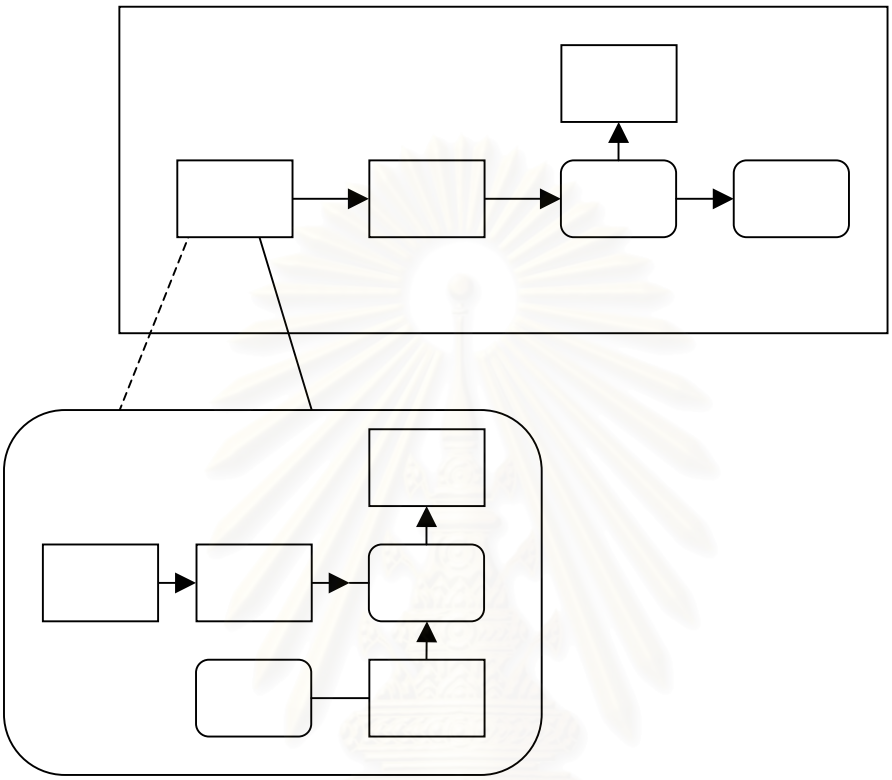


แผนภาพ 17

Event Diagram แสดงกระบวนการลงทะเบียนของนักศึกษาโดยนำ State – Transition Diagram จากแผนภาพ 16 มาบูรณาการเข้าด้วยกันในแผนภาพนี้ (จาก Martin 1993: 163)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Object-Flow Diagrams หมายถึง การเขียนแผนภาพแสดงกระบวนการของเหตุการณ์และการกระทำที่ซับซ้อนมาก ๆ โดยการแตกออกเป็นแผนภาพย่อยเพื่อแสดงกระบวนการหรือเหตุการณ์ย่อย (แผนภาพ 18)



แผนภาพ 18

Object - Flow Diagram

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทสรุป

การวิเคราะห์ระบบโดยใช้เทคนิคเชิงวัตถุ เป็นวิธีการนำไปสู่การพัฒนาโปรแกรมเชิงวัตถุ ซึ่งจะทำให้โปรแกรมมีคุณภาพมากขึ้น (Laudon and Laudon 1998: 492) เพราะนักวิเคราะห์ นักออกแบบ โปรแกรมเมอร์ และผู้ใช้ระบบใช้แบบจำลองแนวคิดเดียวกัน คือ ทุกคนนึกถึงประเภทของวัตถุ วัตถุ พฤติกรรมที่วัตถุแสดงหรือกระทำ และมองเห็นภาพของวัตถุประเภทต่าง ๆ ประเภทย่อยของวัตถุ และการถ่ายทอดคุณลักษณะ รวมทั้งองค์ประกอบของวัตถุ และการห่อหุ้ม และเหตุการณ์ที่กระตุ้นให้เกิดการเปลี่ยนแปลงสภาพของวัตถุ ด้วยเหตุนี้ช่วงต่อจากการวิเคราะห์สู่การ

ออกแบบของเทคนิคเชิงวัตถุจะเป็นธรรมชาติจนบางครั้งอาจยากที่จะแยกขั้นตอนการวิเคราะห์และการออกแบบออกจากกันได้อย่างชัดเจน ซึ่งแตกต่างจากวิธีการพัฒนาระบบแบบดั้งเดิมที่นักวิเคราะห์ นักออกแบบ และโปรแกรมเมอร์ ต่างใช้แนวคิดและมองภาพของระบบแตกต่างกัน คือ นักวิเคราะห์และนักออกแบบใช้แบบจำลองความสัมพันธ์ของเอนทิตี้ (entity-relationship model) แผนภาพการไหลของข้อมูล ผังงาน ส่วนโปรแกรมเมอร์ใช้การแปลความหมายของภาษาโคบอล ภาษาฟอร์แทน ภาษาซี หรือภาษาเอดา (Martin 1993: 62)

นอกจากนั้นผู้เสนอวิธีการพัฒนาระบบเชิงวัตถุเชื่อว่าการทำความเข้าใจ 'วัตถุ' ง่ายกว่าการพิจารณาระบบตามแนวคิดของแบบจำลองที่ประกอบด้วยส่วนนำเข้า กระบวนการ ผลลัพธ์ และการไหลของข้อมูล ซึ่งเป็นแนวคิดของวิธีการวิเคราะห์และออกแบบดั้งเดิม

อย่างไรก็ตามมีงานวิจัยบางเรื่องแสดงให้เห็นว่าการนำวิธีการพัฒนาซอฟแวร์เชิงวัตถุไปใช้กับการวิเคราะห์ความต้องการสารนิเทศเฉพาะด้านบางเรื่องอาจทำได้ยากกว่าการใช้วิธีการแบบดั้งเดิม (Vessay and Conger 1994 quoted in Laudon and Laudon 1998: 493) อาจกล่าวได้ว่าลักษณะของงานแบบเกมและเว็บจะเหมาะสมกับวิธีการพัฒนาซอฟแวร์เชิงวัตถุมากกว่า ส่วน Laudon and Laudon (1998: 494) ได้ตั้งข้อสังเกตว่าการใช้เทคนิคเชิงวัตถุอาจมีอุปสรรคอยู่บ้าง อันเนื่องมาจากบุคลากรที่เกี่ยวข้องไม่ได้รับการอบรมให้มีความรู้ความเข้าใจเกี่ยวกับวิธีการดังกล่าวอย่างเพียงพอ ดังนั้นจึงมักประสบปัญหาการปรับเปลี่ยนแนวคิดจากวิธีการแบบดั้งเดิมมาเป็นวิธีการเชิงวัตถุ รวมทั้งอาจมีปัญหาเกี่ยวกับการพัฒนาเทคโนโลยีใหม่เพื่อรองรับการใช้วิธีการเชิงวัตถุ

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

รายการอ้างอิง

Laudon, Kenneth and Laudon, Jane P. Management Information System: a New Approaches to Organization & Technology. 5th ed. Upper Saddle River: Prentice Hall, 1998.

Marakas, George M. System Analysis and Design: an Active Approach. Upper Saddle River, NJ: Prentice – Hall 2001.

Martin, James. Principles of Object – Oriented Analysis and Design. Englewood Cliffs, NJ: Prentice – Hall, 1993.

Stair, Ralph M. Principles of Information Systems: a Managerial Approach. 2nd ed. Cambridge: A Division of International Thomas Publishing,1996.

Vessey, Iris, and Conger, Sue A. "Requirement specification: Learning Object, Process, and Data Methodologies." Communications of the ACM 37,5(May 1994): 102-113. quoated in Kenneth C. Laudon, and Jane P. Laudon. Management Information System: a New Approaches to Organization & Technology. Upper Saddle: Prentice Hall, 1998.

วราพร แสงเงิน. "Object – Oriented Analysis and Design." [ม.ป.ท.], 2544. เอกสารอัดสำเนา

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย