บทที่ 2
ทฤษฎีและกรอบแนวคิดของโครงการวิจัย

ในบทนี้จะกล่าวถึงทฤษฎีต่างๆที่เกี่ยวข้อง และกรอบแนวคิดของโครงการวิจัยว่ามีลักษณะอย่างไร ซึ่งจะเป็นพื้นฐานในการดำเนินการวิจัยต่อไปดังนี้

ทฤษฎีและกรอบแนวคิดที่เกี่ยวข้องสามารถแบ่งเป็นส่วนหลักๆ ได้ 2 ส่วนสำคัญ คือ ส่วนคณิตศาสตร์ และส่วนคอมพิวเตอร์

ส่วนคณิตศาสตร์

1. **ดอกเบี้ย (interest)**

   คือ จำนวนเงินที่ผู้กู้จ่ายให้แก่ผู้ให้กู้ เพื่อตอบแทนการใช้ประโยชน์เงินต้นที่กู้ยืมไปตามอัตราร้อยละและตามระยะเวลาที่ตกลงกัน ดอกเบี้ยแบ่งออกเป็น 2 ประเภท

   ดอกเบี้ยเชิงเดียว(Simple interest)คือดอกเบี้ยที่คิดจากเงินต้นที่เท่ากันทุกครั้ง โดยคิดจากเงินต้นที่ได้รับในวันที่ผู้กู้

$$I = P \times r \times t$$

โดยที่ P คือ เงินต้นที่ลงทุนไป(กู้ไป)

r คือ อัตราดอกเบี้ยรายปี (เช่น ดอกเบี้ย 8% หมายความว่า $r = \frac{8}{100}$)

t คือ ระยะเวลาที่กู้ยืม (มีหน่วยเป็นปี)

$= P(1 + r \times t)$ เงินรวม(Accumulated amount) หรือ เงินลงทุน (หนี้) ก็คือ เงินต้นรวมกับดอกเบี้ย

$$S = P + I$$
$$= P + P \times r \times t$$

ตัวอย่าง นำเงิน 50,000 บาท ไปลงทุนเป็นเวลา 2 ปี ที่อัตราดอกเบี้ยเชิงเดี่ยว 6% ต่อปี จงหาจำนวนเงินค่าดอกเบี้ย และจำนวนยอดหนี้ทั้งหมดที่ต้องจ่าย

วิธีทำ $P = 50,000\ r = \frac{6}{100} = 0.06\ t = 2$

จาก $I = Prt$

แทนค่าจะได้ $I = 50,000 \times 0.06 \times 2 = 6,000$

จำนวนยอดหนี้ทั้งหมด(เงินรวม)

จาก $S = P + I$

จะได้ $50,000 + 6,000 = 56,000$

$\therefore$ ดอกเบี้ยทั้งหมดคือ $6,000$ บาท และ ยอดหนี้ทั้งหมดคือ $56,000$ บาท

ดอกเบี้ยทบต้น (Compound Interest)คือดอกเบี้ยที่คิดจากเงินต้นซึ่งระยะเวลาของการคิดดอกเบี้ยจะถูกแบ่งออกเป็นงวดๆวิธีการดอกเบี้ยทบต้นคือเมื่อถึงกำหนดเวลาคิดดอกเบี้ยก็จะมีการคิดดอกเบี้ยเชิงเดียวของงวดนั้นเป็นเท่าไร แล้วนำดอกเบี้ยที่ได้รับครั้งแรกรวมเข้ากับเงินต้นครั้งแรกเป็นเงินต้นครั้งที่ 2 แล้วนำดอกเบี้ยที่ได้รับครั้งที่ 2 รวมเข้ากับเงินต้นครั้งที่ 2 เป็นเงินต้นครั้งที่ 3 ทำเช่นนี้ไปเรื่อยจนครบกำหนด ในทางธุรกิจจะมีการคิดดอกเบี้ยทบต้นต่อปี ต่อครึ่งปี ต่อเดือน หรือต่อวันก็ได้ โดยระยะเวลาในการคิดดอกเบี้ยเรียกว่า **งวด**

อัตราดอกเบี้ยที่กำหนด คือ อัตราดอกเบี้ยต่องวด เช่น ถ้าอัตราดอกเบี้ย 12% ต่อปี คิดทบต้นทุก 6 เดือน หมายความว่า จะมีการคิดดอกเบี้ย 6% ต่อทุกๆ 6 เดือน คือปีละ 2 งวด หรือ ถ้าอัตราดอกเบี้ย 12% ต่อปี คิดทบต้นทุกไตรมาส หมายความว่า จะมีการคิดดอกเบี้ย 3% ต่อทุกๆ 3 เดือน คือปีละ 4 งวด ดังแสดงในตารางต่อไปนี้

อัตราดอกเบี้ยแต่ในนาม	จำนวนครั้งที่คิดดอกเบี้ยใน 1 ปี	อัตราดอกเบี้ยต่องวด (i)
12% ต่อปี คิดทบต้นต่อปี	1	0.12/1=0.12
12% ต่อปี คิดทบต้นทุก 6 เดือน	2	0.12/2=0.06
12% ต่อปี คิดทบต้นทุก 4 เดือน	3	0.12/3=0.04
12% ต่อปี คิดทบต้นทุกไตรมาส	4	0.12/4=0.03
12% ต่อปี คิดทบต้นทุกเดือน	12	0.12/12=0.01

ตารางต่อไปนี้แสดงการคิดดอกเบี้ยและเงินรวมของแต่ละงวด

งวดที่	เงินต้น	ดอกเบี้ย	เงินรวม
1	P	$P\times i$	$P(1+i)$
2	$P(1+i)$	$P(1+i)\times i$	$P(1+i)^2$
3	$P(1+i)^2$	$P(1+i)^2\times i$	$P(1+i)^3$
...	...	...	...
n	$P(1+i)^{n-1}$	$P(1+i)^{n-1}\times i$	$P(1+i)^n$

$$S_n = P(1+i)^n$$

2. ฟังก์ชัน (Function)

บทนิยาม ถ้า A และ B เป็นเซตที่ไม่ว่าง **ฟังก์ชัน** (Function) จาก A ไปยัง B คือเซตย่อย f ของ $A\times B$ โดยที่

(1) แต่ละ $a\in A$ จะมีคู่อันดับอย่างน้อยคู่อันดับหนึ่ง คือ $(a,b)\in f$ โดยที่ $b\in B$

(2) ถ้า $(a,b)\in f$ และ $(a,c)\in f$ แล้ว $b=c$

เซต A เรียกว่า **โดเมน** (Domain) ของฟังก์ชัน f โดเมนของฟังก์ชัน f จะเขียนแทนที่ด้วย D_f หรือ $\mathrm{dom}(f)$ นั่นคือ

$$\mathrm{dom}(f) = D_f = \{x : (x,y)\in f \text{ บางค่า } y\}$$

ถ้า $(x,y)\in f$ แล้ว y เรียกว่า **ค่าฟังก์ชัน**(Functional value) หรือ ภาพของ x ภายใต้ f (image of x under f) เขียนแทนด้วยสัญลักษณ์ $y=f(x)$

เซตย่อย $\{f(x) : x\in A\}$ ของ B เรียกว่า **เรนจ์** (range) ของ f เขียนแทนด้วย R_f หรือ $\mathrm{ran}(f)$ นั่นคือ

$$\mathrm{ran}(f) = R_f = \{y : (x,y)\in f \text{ บางค่า } x\}$$

นิยามเขียนฟังก์ชัน f จาก A ไปยัง B แทนด้วยสัญลักษณ์

$$f : A\to B$$

โดยกำหนดว่าโดเมนของ f เท่ากับ A เรนจ์ของ f เป็นเซตย่อยของ B กรณีเรนจ์ของ f เท่ากับ B จะเรียกชื่อพิเศษออกไป

โดยทั่วไป ถ้าให้ S เป็นเซต เราจะเรียกฟังก์ชัน $S\times S$ ไปยัง S ว่าเป็น**การดำเนินการทวิภาค**บน S (Binary operation on S)

2.1 ประเภทของฟังก์ชัน

2.1.1 ฟังก์ชันจาก A ไป B (Functions from A into B)

บทนิยาม f เป็นฟังก์ชันจาก A ไป B ก็ต่อเมื่อ f เป็นฟังก์ชันที่ A เป็นโดเมนและมีสับเซตของ B เป็นเรนจ์ f เป็นฟังก์ชันจาก A ไป B เขียนแทนด้วย $f : A \rightarrow B$

ตัวอย่าง 3 ถ้า $f = \{(1,3),(2,5),(4,7)\}$

จะได้ว่า $f(1) = 3$

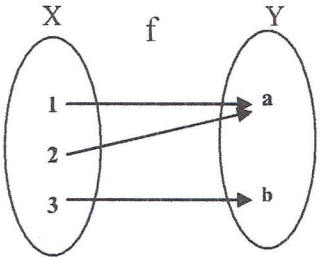
$f(2) = 5$

$f(4) = 7$

2.1.2 ฟังก์ชันจาก A ไปทั่วถึง B (Functions from A onto B)

บทนิยาม ฟังก์ชัน f จาก A ไป B จะเรียกว่า**ฟังก์ชันทั่วถึง** (Onto function) ถ้าเรนจ์ของ f เท่ากับ B ($ran(f) = B$) นั่นคือ ถ้ากำหนด $b \in B$ จะต้องมีสมาชิก $a \in A$ ที่ $f(a) = b$

ตัวอย่าง 4 จงพิจารณาว่า f เป็นฟังก์ชันไปทั่วถึงหรือไม่



จาก $X = \{1,2,3\}$ และ $Y = \{a,b\}$

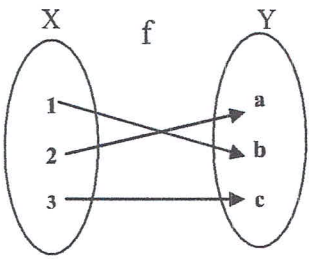
จะได้ $f = \{(1,a),(2,a),(3,b)\}$

ดังนั้น f เป็นฟังก์ชันไปทั่วถึงจาก X ไป Y

2.1.3 ฟังก์ชันหนึ่งต่อหนึ่ง (One – to – one Function)

บทนิยาม ฟังก์ชัน f จาก A ไป B จะเรียกว่า**ฟังก์ชันหนึ่งต่อหนึ่ง** (One – to – one function) ถ้า $x_1 \neq x_2$ แล้ว $f(x_1) \neq f(x_2)$ เขียนแทนด้วย $f : A \xrightarrow[1\text{–}1]{} B$

ตัวอย่าง 5 จงพิจารณาว่า f เป็นฟังก์ชันหนึ่งต่อหนึ่งหรือไม่



จาก $X=\{1,2,3\}$ และ $Y=\{a,b\}$

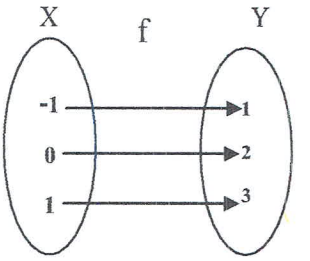
จะได้ $f=\{(1,b),(2,a),(3,c)\}$

ดังนั้น f เป็นฟังก์ชันหนึ่งต่อหนึ่งจาก X ไป Y

2.1.4 ฟังก์ชันหนึ่งต่อหนึ่งจาก A ไปทั่วถึง B (One–to–one Functions from A onto B or one–to–one correspondence)

บทนิยาม f เป็นฟังก์ชันหนึ่งต่อหนึ่งจาก A ไปทั่วถึง B ก็ต่อเมื่อ f เป็นฟังก์ชันหนึ่งต่อหนึ่งจาก A ไป B และ B เป็นเรจน์ของ f เขียนแทนด้วย $f: A \xrightarrow[\text{0nto}]{1\text{–}1} B$

ตัวอย่าง 6 จงพิจารณาว่า f เป็นฟังก์ชันหนึ่งต่อหนึ่งจาก X ไปทั่วถึง Y หรือไม่



จาก $X=\{-1,0,1\}$ และ $Y=\{1,2,3\}$ จะได้ $f=\{(-1,1),(0,2),(1,3)\}$

ดังนั้น f เป็นฟังก์ชันหนึ่งต่อหนึ่งจาก X ไปทั่วถึง Y

ส่วนคอมพิวเตอร์

พื้นฐานและทฤษฎีทางคอมพิวเตอร์

1. การเขียนโปรแกรมเบื้องต้น

แนวคิดการเขียนโปรแกรม

การเขียนโปรแกรม ทุกภาษานั้นเหมือนกัน สิ่งที่แตกต่างกัน ของแต่ละภาษาคือ syntax แต่สิ่งที่เหมือนกันของทุกภาษาคือ การใช้ประสบการณ์จากภาษาหนึ่ง ไปใช้ในอีกภาษาหนึ่งได้ ด้วยการซึมซับ เรื่องของ Structure Programming จนเข้าใจ เพื่อควบคุมในสิ่งที่คล้าย ๆ กันคือ input, process และ output ซึ่งหมายความว่า ถ้าท่านเขียนโปรแกรมอะไร ในภาษาหนึ่งได้แล้ว การเขียนโปรแกรมแบบนั้น ในภาษาอื่นย่อมไม่ใช่เรื่องยากอีกต่อไป เพียงแต่ต้องศึกษาถึง syntax หรือ

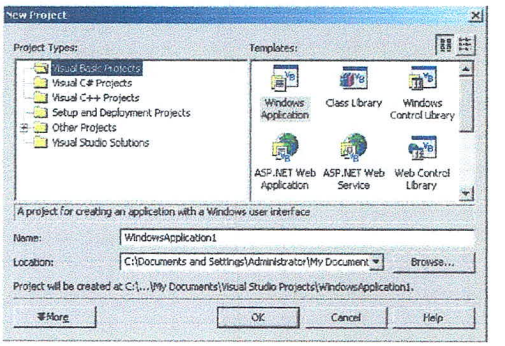
รูปแบบการเขียนของภาษาใหม่นั้นเพิ่มเติม แล้วนำประสบการณ์ที่เคยเขียน ไปสั่งให้ภาษาใหม่ทำงานตามต้องการ ผมจึงมักสนับสนุนให้นักเรียน ได้ศึกษาภาษาที่ไม่มีตัวช่วยมาก เพื่อให้เข้าใจในหลักการ และขั้นตอนการทำงาน อย่างละเอียดชัดเจน จากการทำงานของตัวแปรภาษาที่มีตัวช่วยน้อย ทำงานบน dos สามารถแปลเป็น exe และ นำไปใช้ได้โดยไม่ยุ่งยาก เช่น C, Pascal, Basic, Fox... หรือ Clipper เป็นต้น สำหรับการทำโครงงานปัญหาพิเศษนี้ในส่วนของการเขียนโปรแกรมผู้ทำได้เลือกภาษาที่ใช้คือ Visual Basic.NET

Visual Basic.NET

Visual Basic.NET เป็นภาษาที่พัฒนาต่อจาก Visual Basic 6.0 หรือพูดง่ายๆ ก็คือเป็นVisual Basic Version 7 ซึ่งขยายขีดความสามารถที่ Visual Basic เดิมไม่สามารถทำได้ โดยเฉพาะในเรื่องของการเป็นภาษาเชิงวัตถุอย่างแท้จริง (สนับสนุนโครงสร้างของภาษาที่เป็น OOP 100%) ทำให้โครงสร้างภาษาของ Visual Basic.NET นั้นมีความสมบูรณ์มากขึ้น แต่ก็ยังคงสนับสนุนรูปแบบการเขียนแบบเดิมไว้ ในบางส่วนเพื่อความสะดวกสำหรับผู้ที่ย้ายจาก Visualก่อนหน้านี้มาบน Visual Basic.NET Visual Basic.NET เป็นภาษาหนึ่งที่อยู่ในชุดเครื่องมือ Microsoft Studio.NET โดยจะใช้ IDE (Integrated Development Environment) รวมกับภาษาอื่นอีก 3 ภาษา ที่อยู่ในชุดเครื่องมือนี้ ซึ่งได้แก่ Visual C#, Visual C++ และ Visual J#

ส่วนประกอบของ Visual Basic.NET

จากรูปเลือก Visual Basic Projects ที่ด้านขวาจะแสดงรายการของ Project ที่สามารถพัฒนาได้ด้วย Visual Basic.NET ซึ่งมีดังนี้



ใน VB.NET มีชนิดของโปรเจ็กต์ดังต่อไปนี้

1.Windows Application ใช้พัฒนา Windows Application เราจะสร้างแอพพลิเคชั่นนี้ด้วยฟอร์มและคอรโทรล

2.Class Library ใช้พัฒนา Library ไว้ใช้งาน เป็นชนิดของโปรเจ็กต์ที่ให้เราสร้าง

คลาสที่จะใช้ในแอพพลิเคชั่นอื่นๆ ได้

3.Windows Control Library เป็นชนิดของโปรเจ็กต์ ซึ่งจะทำให้เราสร้างคอนโทรลใหม่ขึ้นมาได้ นอกเหนือจากคอนโทรลพื้นฐานที่ VB.NET ได้เตรียมไว้ให้

4.Mobile Web Application ใช้พัฒนา Application บน Mobile Device (ต้องติดตั้ง Mobile Toolkit เพิ่มเติมจึงจะมี Project นี้)

5.ASP.NET Web Application I ใช้พัฒนา Web Application หรือ Web Form โดยใช้ ASP.NET

6. ASP.NET Web Service ใช้พัฒนา Web Service ที่สามารถถูกเรียกใช้งานได้จากแอพพลิเคชั่นอื่นบนอินเตอร์เน็ต เช่นจาก ASP.NET

7.Web Control Library ใช้พัฒนา Web Control ไว้ใช้งาน ซึ่งจะช่วยให้เราสร้างคอนโทรลใหม่นอกเหนือจากคอนโทรลที่ VB.NET เตรียมไว้ให้ เพื่อใช้กับแอพพลิเคชั่น ASP.NET ได้

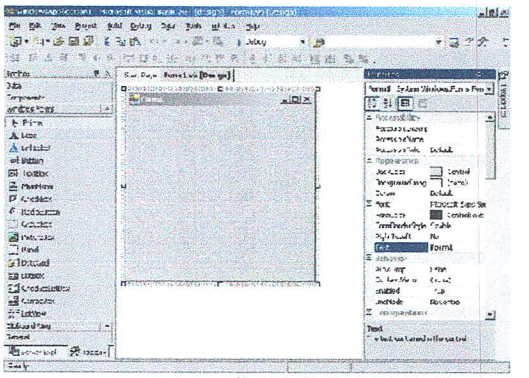
8. Console Application ใช้พัฒนา Console Application (Command Line) แอพพลิเคชั่นประเภทนี้จะต้องรันในแบบ command - line ซึ่งจะรับอินพุตจากคำสั่งที่เราพิมพ์ลงไป และแสดงผลลัพธ์ออกมาในหน้าต่าง Command Prompt

9.Windows Service ใช้พัฒนา Windows Service

10.Empty Project สร้าง Project ว่างๆ

11.Empty Web Project สร้าง Web Project ว่างๆ

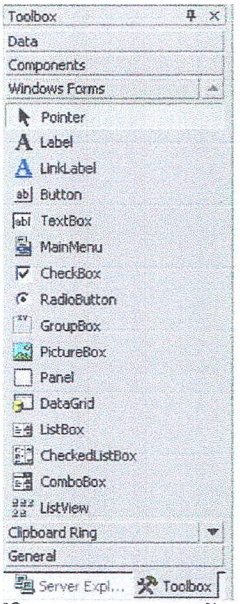
12.New Project in Existing Folder เป็น Project ใหม่ จาก Folder ที่มีอยู่แล้ว เมื่อทำการเลือก Project ที่ต้องการสร้าง ในที่นี้เลือก Windows Application จะได้หน้าต่างเพิ่ม ขึ้นมาพร้อมทั้งรายละเอียดต่างๆ ดังรูป



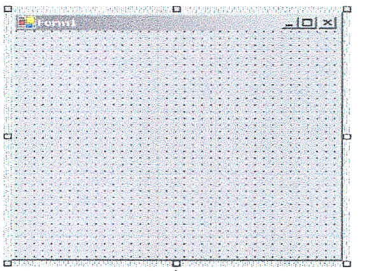
เมนูบาร์ : เก็บคำสั่งที่เราสามารถใช้งานได้ทั้งหมดใน VB.NET ประกอบไปด้วยเมนูทำงานกับ File, View และ Windows เป็นต้น

ทูลบาร์ : ประกอบด้วยปุ่มคำสั่งต่างๆ ที่ช่วยให้เราใช้งานคำสั่งของ VB.NET ได้รวดเร็วยิ่งขึ้น

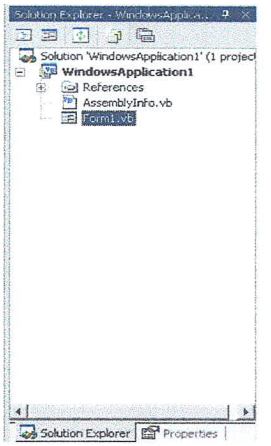
ทูลบอกซ์ : เป็นที่แสดงเครื่องมือต่างๆ ที่เราเรียกว่า คอนโทรล ซึ่งเป็นเครื่องมือที่เราสามารถเลือกไปวางลงบนฟอร์มได้ เพื่อออกแบบหน้าจอของโปรแกรม



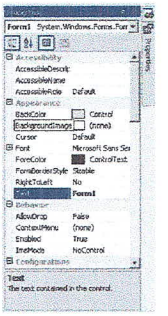
ฟอร์ม : เป็นหน้าต่างที่ใช้ในการออกแบบหน้าจอโปรแกรม



Solution Explorer : เป็นหน้าต่างที่แสดงโปรเจกต์ต่างๆ ที่มีใน Solution โดยแต่ละโปรเจกต์ก็จะมีโมดูล (Modules) ต่างๆ อยู่



หน้าต่าง Properties : เป็นหน้าต่างที่แสดงคุณสมบัติของคอนโทรลที่เลือกอยู่ในขณะนั้น

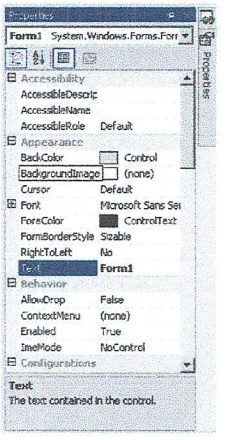


ทูลบอกซ์ (Toolbox) Visual Basic.NET

Toolbox : เป็นส่วนที่แสดง Component ทั้งหมดที่มากับ Visual Studio.NET ทั้งนี้ ทุกภาษา ที่ทำงานบน IDE นี้ สามารถใช้ Component ที่อยู่ใน Toolbox นี้ได้เหมือนกัน โดยจะมีการแบ่งออกเป็นหมวดหมู่ไว้เพื่อความสะดวกในการเรียกใช้งาน และยังสามารถเพิ่ม Component เพิ่มเติมได้อีกด้วยคอนโทรล (Controls) เป็นเครื่องมือในการออกแบบหน้าจอโปรแกรม ซึ่งเครื่องมือต่างๆ เหล่านี้ VB.NET ได้เตรียมไว้ให้ในทูลบอกซ์ โดยให้เราเลือกคอนโทรลที่ตรงกับจุดประสงค์ในการใช้งาน และนำมาวางบนฟอร์มว่างที่ปรากฏอยู่ ต่อไปนี้เป็นคอนโทรลทั้งหมดที่มีอยู่ในทูลบอกซ์

Properties Visual Basic.NET

คุณสมบัติ (Properties): คือลักษณะต่างๆ ของคอนโทรลที่ถูกนำมาวางบนฟอร์มที่เราสามารถกำหนดได้



คุณสมบัติ (Properties) พื้นฐานมีดังนี้

1. Name	ชื่อของวัตถุ VB จะตั้งชื่อให้วัตถุในตอนเริ่มต้น โดยใช้ชื่อคอนโทรลตามด้วยตัวเลข
2. Text	ข้อความที่ปรากฏในคอนโทรล
3. Backcolor	สีพื้น
4. Forecolor	มักจะหมายถึงสีของข้อความบนคอนโทรล
5. Cursor	รูปแบบเคอร์เซอร์ เมื่อนำเมาส์เคลื่อนที่อยู่เหนือคอนโทรลจะเปลี่ยนรูปไปตามที่เลือก
6. Enabled	สั่งให้คอนโทรลทำงานได้หรือไม่ได้
7. Font	รูปแบบของตัวอักษรที่ปรากฏบนคอนโทรล
8. Location	ตำแหน่งการวางคอนโทรลประกอบด้วยแนว x และ y
9. Locked	ถ้าถูกเลือกให้เป็น true คอนโทรลจะเคลื่อนไม่ได้ในตอนdesign
10. Size	ขนาดของคอนโทรล ประกอบด้วย width และ length
11. Visible	การมองเห็นคอนโทรล

คุณสมบัติ เมดธอด และอีเว็นต์

ในการที่จะออกแบบหน้าจอขึ้นมานั้นเราจำเป็นต้องทราบถึงหลักการพื้นฐานเกี่ยวกับฟอร์ม และคอนโทรล เพื่อที่จะได้สามารถนำมาออกแบบฟอร์มให้เหมาะสม

1. คุณสมบัติ (Properties) ให้เรากำหนดลักษณะต่างๆ ของฟอร์มและคอนโทรล เราสามารถกำหนดคุณสมบัติสำหรับคอนโทรลต่างๆ ผ่านทางหน้าต่าง Properties หรือโดยใช้คำสั่งที่มีรูปแบบดังต่อไปนี้

MyCmd.Text = "OK" MyCmd.Height = 1000 MyCmd.Width = 2000	เป็นการกำหนดค่าให้คุณสมบัติ Text ของ MyCmd เป็นการกำหนดค่าให้คุณสมบัติ Height ของ MyCmd เป็นการกำหนดค่าให้คุณสมบัติ Width ของ MyCmd

2. เมดธอด (Method) เป็นการสั่งให้ฟอร์ม และคอนโทรลทำงานตามที่เราร้องขอไป ในการสั่งให้ปุ่มคำสั่งทำงานตามที่เราร้องขอ จะใช้คำสั่งต่อไปนี้

```
Private Sub Form1_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles MyBase.Load
MyCmd.Focus()
End Sub
```

3. อีเว็นต์ (Events) เป็นเหตุการณ์ที่เกิดขึ้นกับฟอร์ม หรือคอนโทรลที่เราสามารถใส่คำสั่งเพื่อตอบสนองได้ ในการตอบสนองต่ออีเว็นต์ ใส่คำสั่งต่อไปPrivate Sub MyCmd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyCmd.Click

```
MyCmd.Text = "Click"
End Sub
```

คำสั่งที่ใช้ในการควบคุมการทำงาน

คำสั่งในการเลือกเส้นทางการทำงาน

คำสั่งที่ใช้ในการเลือกเส้นทางการทำงาน ทำให้เราสามารถเลือกเส้นทางการทำงานของโปรแกรมได้ตามเงื่อนไขที่อยู่ในโปรแกรม

if ...else...end if

1. แบบมีเงื่อนไขเดียว

```
If <เงื่อนไข> then
' คำสั่งเงื่อนไขที่มีค่าเป็น True
End if
```

2. หรือมีหลายเงื่อนไข

```
If <เงื่อนไข> then
' เงื่อนไขที่ 1 ที่มีค่าเป็น True
Elseif <เงื่อนไข> then
' เงื่อนไขที่ 2 ที่มีค่าเป็น True
Else
' เงื่อนไขทั้ง 2 เงื่อนไขที่มีค่าเป็น False
End if
```

2. การเขียนผังงาน (Flowchart)

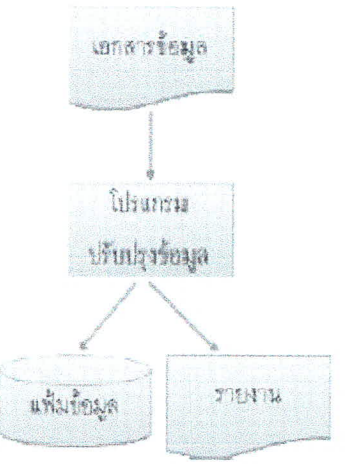
การเขียนผังงาน(Flowchart)เป็นเทคนิคหรือวิธีการอย่างหนึ่งสำหรับใช้เขียนแสดงอัลกอริทึม การเขียนผังงานจะใช้รูปภาพที่เป็นสัญลักษณ์มาตรฐานสากลนำไปเขียนแทนกิจกรรมต่าง ๆ ในแต่ละขั้นตอนของอัลกอริทึม ซึ่งกิจกรรมที่แตกต่างกันก็จะใช้สัญลักษณ์รูปภาพที่แตกต่างกัน โดยสัญลักษณ์ภาพนี้กำหนดตามมาตรฐานของ ANSI (American National Standards Institute) และ ISO (International Standard Organization) การเขียนแสดงอัลกอริทึมด้วยผังงานสามารถออกแบบได้ง่าย ผู้อื่นสามารถเข้าใจผังงานได้ง่าย เพราะเป็นมาตรฐานสากล และเมื่อนำผังงานไปเขียนโปรแกรมก็จะทำได้สะดวกรวดเร็ว

ประเภทของผังงาน

ผังงานทางคอมพิวเตอร์มี 2 ประเภทคือ ผังงานระบบ(System Flowchart) และผังงานโปรแกรม(Program Flowchart)

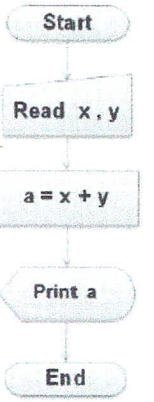
ผังงานระบบ(System Flowchart)

ผังงานระบบจะเป็นการแสดงให้เราเห็นว่า ในระบบหนึ่ง ๆ นั้นมีขั้นตอนในการทำงานอย่างไร ซึ่งจะมองเห็นในลักษณะภาพกว้าง ๆ ของระบบ แต่จะไม่เจาะลึกลงไปว่าในระบบว่าในแต่ละงานนั้นมีการทำงานอย่างไร คือ จะให้เห็นว่าจุดเริ่มต้นของงานเริ่มจากส่วนใด เป็นข้อมูลแบบใด มีการประมวลผลอย่างไร และจะได้ผลลัพธ์เป็นอย่างไรและเก็บอยู่ที่ใด ดังรูป



ผังงานโปรแกรม (Program Flowchart)

ผังงานโปรแกรม หรือ เรียกสั้น ๆ ว่า ผังงาน จะเป็นผังงานที่แสดงให้เห็นถึงลำดับขั้นตอนในการทำงานของโปรแกรม ตั้งแต่การรับข้อมูล การประมวลผล ตลอดจนผลลัพธ์ที่ได้ จะทำให้เขียนโปรแกรมได้สะดวกขึ้น ซึ่งผังงานชนิดนี้อาจสร้างมาจากผังงานระบบ โดยดึงเอาจุดที่เกี่ยวข้องกับคอมพิวเตอร์มาวิเคราะห์ว่าจะใช้ทำงานส่วนใดเพื่อที่จะให้ได้มาซึ่งผลลัพธ์ที่ต้องการ ดังรูป



ประโยชน์ของผังงาน

1. ทำให้มองเห็นรูปแบบของงานได้ทั้งหมด โดยใช้เวลาไม่มาก
2. การเขียนผังงานเป็นสากลสามารถนำไปเขียนโปรแกรมได้ทุกภาษา
3. สามารถตรวจสอบข้อผิดพลาดของโปรแกรมได้อย่างรวดเร็ว
4. หากมีการพัฒนาระบบงานในลำดับต่อไป สามารถทำได้อย่างรวดเร็ว โดยศึกษาจากผังงาน จะสามารถศึกษาได้อย่างรวดเร็ว และเข้าใจง่ายกว่าการศึกษาจากโปรแกรม

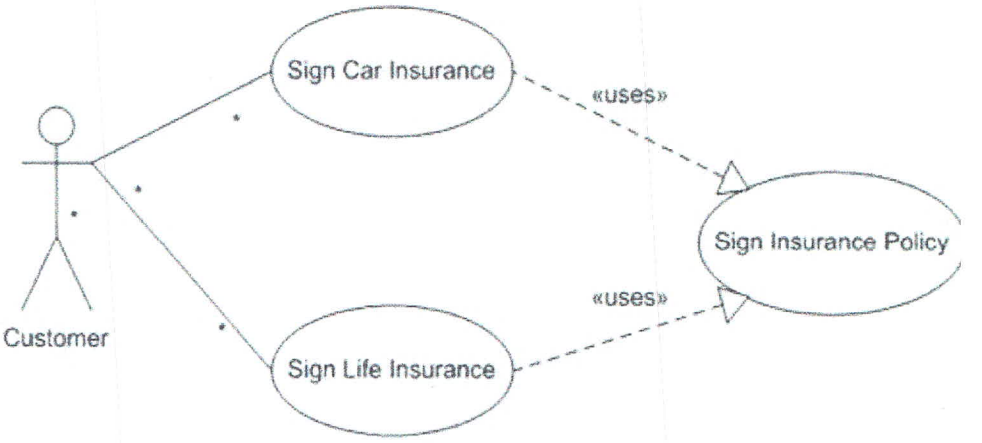
3. ความสัมพันธ์ระหว่าง Use Case

ความสัมพันธ์แบบรวม (Include Relationship)

หมายถึง ความสัมพันธ์ที่แต่ละ Use Case ภายในระบบเองมีความสัมพันธ์กัน โดยความสัมพันธ์ของ Use Case นั้น สามารถแบ่งออกได้ 2 แบบ คือ Include และ Extends

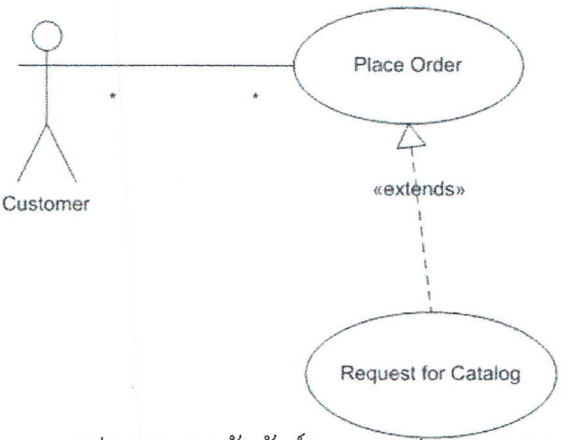
ความสัมพันธ์แบบ Include หมายถึง การที่ Use Case หนึ่ง เรียกใช้งาน Use Case อีกอันหนึ่ง คล้าย ๆ กับการเรียกใช้งาน Program ย่อยโดย Program หลัก การ

เขียนสัญลักษณ์แทนการ Include ของ Use Case นั้น ใช้สัญลักษณ์เส้นประพร้อมหัวลูกศรชี้ไปยัง Use Case ที่ถูกเรียกใช้งาน และมีคำว่า <<include>> กำกับอยู่บนเส้นลูกศร



ความสัมพันธ์แบบขยาย (Extend Relationship)

ความสัมพันธ์แบบ Extend หมายถึง การที่ Use Case หนึ่งไปมีผลต่อการทำงานตามปกติของอีก Use Case หนึ่ง นั่นหมายถึงว่า Use Case ที่มา Extend นั้นจะมีผลทำให้การทำงานของUse Case ที่ถูก Extend ถูกรบกวนหรือมีการสะดุดหรือมีกิจกรรมที่เปลี่ยนแปลงไป สัญลักษณ์ที่ใช้แทน Extend ใน Use Case Diagram ก็คือ ใช้สัญลักษณ์ลูกศร โดยเริ่มจาก Use Case ที่Extend ไปยัง Use Case ที่ถูก Extend และมีคำว่า << extend >> กำกับ

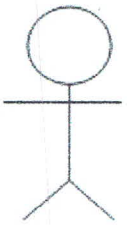


รูปแสดงความสัมพันธ์แบบขยาย(Extend Relationship)

ประโยชน์ของ Use case diagram

- ทราบความสามารถของระบบ
- ทราบผู้ใช้งานในแต่ละส่วนของระบบ
- ง่ายต่อการสื่อสารระหว่างลูกค้าและผู้พัฒนาระบบ
- ใช้ทดสอบระบบว่าตรงตามความต้องการของระบบหรือไม่
- ช่วยให้ผู้พัฒนาระบบสามารถแยกแยะกิจกรรมที่อาจจะเกิดขึ้นในระบบ
- เป็น diagram พื้นฐาน ที่สามารถอธิบายสิ่งต่าง ๆ ได้โดยใช้รูปภาพที่ไม่ซับซ้อน

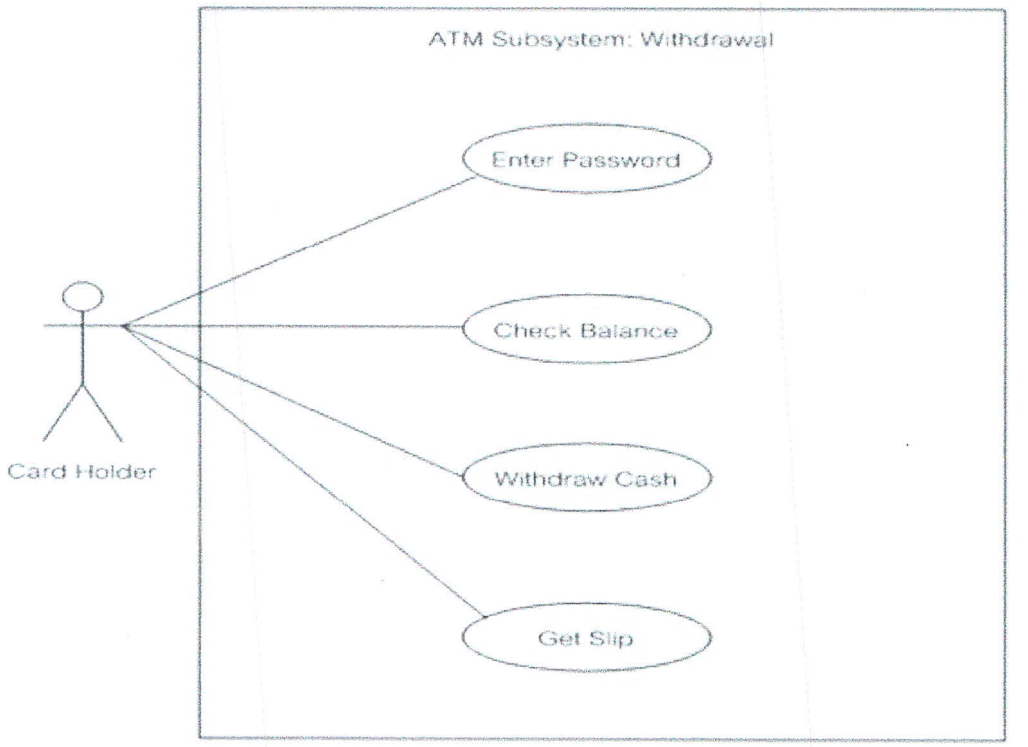
สัญลักษณ์ที่ใช้ใน Use case diagram

สัญลักษณ์	ความหมาย
Use caser name 	หน้าที่ของระบบที่จะต้องทำ
Actor name 	ทำหน้าที่ผลักดันให้เกิดกิจกรรมของระบบ หรือทำหน้าที่ควบคุมดูแลกิจกรรมของระบบ
System name 	เส้นแบ่งขอบเขตระหว่างระบบกับ Actor
Connection	เส้นเชื่อมระหว่าง Actor กับ Use case

Use Case Diagram ประกอบด้วย

- Actor คือ ผู้ที่กระทำกับระบบ อาจเป็นผู้ที่ทำการส่งข้อมูล, รับข้อมูล หรือ แลกเปลี่ยนข้อมูลกับระบบนั้นๆ เช่น ลูกค้ากับระบบสั่งซื้อสินค้าทางโทรศัพท์
- Use Case คือ หน้าที่หรืองานต่างๆในระบบ เช่น การเช็คสต็อก การสั่งซื้อสินค้า เป็นต้น
- Relationship คือ ความสัมพันธ์ระหว่าง Use Case กับ Actor

ตัวอย่าง Use Case Diagram การถอนเงิน



รูปแสดงตัวอย่าง Use Case Diagram การถอนเงิน

ในบทถัดไปจะอธิบายถึงการดำเนินงานการวิจัยที่ใช้แสดงรายละเอียดของการวิเคราะห์และออกแบบโปรแกรม