โปรแกรมช่วยการเขียน Data Flow Diagram

KMIT'L Data Flow Diagram

โดย

วิชัย แซ่แต้

สุรินทร์ เสถียรศักดิ์พงศ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2535

ปริญญานิพนธ์ปีการศึกษา 2535

ภาควิชา คอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

เรื่อง โปรแกรมช่วยการเขียน data flow diagram

ผู้จัดทำ

1. นาย วิชัย แซ่แต้ 32.1298
2. นาย สุรินทร์ เสถียรศักดิ์พงศ์ 32.1395

.....อาจารย์ที่ปรึกษา

(อาจารย์ กฤดวัน เตรือตราชู)

บทคัดย่อ

เราได้ทำการศึกษาหลักการของออบเจ็กโอเรียนเท็ด โปรแกรมมิ่งซึ่งมีหลักการสำคัญ ๆ ดังนี้คือ เอนแคปซูลเลชั่น อินเฮอริแทน โพลีมอฟิสซึ่ม และ ไดนามิกไบดิ้ง จากนั้นก็ศึกษา C++ โปรแกรมมิ่งบนวินโดวด้วยออบเจ็บวินโดวไบบรารี่ ซึ่งเป็นการอิมพลีเมนต์ออบเจ็กโอเรียนเท็ด

จากซอฟแวร์เอนจิเนียร์ คือ ขบวนการที่จะเข้ามาเพื่อการพัฒนา,การปฏิบัติการ,การบำรุงรักษา และเลิกการใช้ซอฟแวร์นั้น และ Automated tool คือซอฟแวร์ทูล ที่ช่วยในการสังเคราะห์, วิเคราะห์ ซอฟแวร์ หรือ เอกสารประกอบซอฟแวร์ ซึ่งโปรแกรมเหล่านี้รู้จักกันในนามของ CASE (Computer Aided Software Engineering)

ตัวอย่างทูลที่ใช้ในซอฟแวร์เอนจิเนียร์ เช่น data flow diagram(DFD), entity - relationship diagram (ERD), data dictionary (DD), structure chart

ในปริญญานิพนธ์นี้ได้ทำการอิมพลีเมนต์ แอพพลิเคชั่นซึ่งช่วยในการเขียนดาต้าโฟลวไดอะแกรมขึ้นมา ซึ่งแอพพลิเคชั่นนี้เป็นทูลส่วนหนึ่งของเคสทูลดังกล่าวมาแล้วข้างต้น

ABSTRACT

We study OOP(Object Oriented Programming). It consists of the four important parts which are Encapsulation, Inheritance, Polymorphism and Dynamic binding. Later, we study C++ language programing on Window with OWL(Object Window Library) which implement OOP.

Software engineering is a systematic approach to the development, operation, maintenance and retirement of sofware.

Automated tools are software tools that aid in the synthesis , analysis, modelling or documentation of software. This program became know as CASE (Computer Aided Software Engineering).

Examples of software engineering tool are data flow diagram(DFD), entity relationship diagram (ERD), data dictionary (DD), structure chart.

In this project, we implement an application which helps writting DFD (Data Flow Diagram). It is a component of CASE tool

สารบัญ

บทที่1
บทนำ

SOFTWARE ENGINEER

Structured Techniques Structured techinques เป็นเทคนิคที่จะช่วย improved productivity ซึ่งจะทำ ให้ผู้พัฒนาระบบ หรือโปรแกรมที่สามารถตรวจสอบ ดูแลรักษาพัฒนาระบบได้ดีมากขึ้น ซึ่ง Structured Techniques นี้ ประกอบด้วย

*Structured analysis

*Structured designed

*Structured programming

*Top-down development

*Programming teams

*Structured walkthroughs

STRUCTURED ANALYSIS Structured analysis เป็นส่วนเริ่มต้น("front end") ในโปรเจคการพัฒนาระบบ ซึ่งในช่วงนี้เราจะต้องทำการหาความต้องการของผู้ใช้(user's requirement) และจะต้องจัด ทำเอกสาร(document) เพื่อใช้ในส่วนอื่นๆ อื่นๆต่อไป Structured analysis มีลักษณะเป็น graphic documentation tools ซึ่งประกอบด้วย tools ดังต่อไปนี้ -Data flow diagrams (DFDs) -Data dictionary (DD) -Entity relationship diagrams (ERDs) -Process Specifications -...

data flow diagram จะแสดงการไหลของข้อมูลตลอดทั้งระบบ ไม่ว่าระบบนั้นจะเป็น ระบบที่ไม่ใช้คอมพิวเตอร์(manaul), ระบบอัตโนมัติ(automatic) หรือผสมกันในรูปแบบทั้ง สองก็ตาม ส่วนประกอบของ data flow diagram ประกอบไปด้วย terminators(บางครั้งก็เรียก "sources" หรือ "sinks"), data flows, processes และ data stores (storage)

data flow diagram เราสามารถแตกเป็น data flow diagram ย่อยๆได้อีกหลายระดับ(หลายlevel) แต่ละprocessเราสามารถแทนด้วยรูปวงกลม

Windows Programming Approach with C++

Windows ถูกพัฒนาขึ้นโดยการใช้เทคนิคการเขียนโปรแกรมที่เรียกว่า Message Driven Programming กล่าวคือ Windows และ Windows Applications จะทำงานโดยขึ้นอยู่กับผู้ใช้ นั่นคือผู้ใช้จะเป็นผู้ทำให้เกิด messages ต่าง ๆ ขึ้น เช่น การกดคีย์ (WM_Keypressed) หรือ การกดปุ่มเมาส์(WM_LeftButtonDown/UP) เป็นต้น ซึ่งแต่ละโปรแกรมจะต้องมี Function ที่จะตอบสนองต่อ message ที่เกิดขึ้นเหล่านี้ จึงอาจกล่าวได้ว่า message ที่เกิดขึ้นเหล่านี้ เป็นการแทนการใช้ Function Call ในการเขียนโปรแกรมทั่วไปนั่นเอง

สำหรับการเขียนโปรแกรมด้วยภาษา C++ นั้น มีความคล้ายคลึงกันกับเทคนิคของ Windows มาก แต่จะต่างกันที่ message ใน C++ นั้น เป็นการเรียก Function Call โดยตรงเลย ในขณะที่ Window Message นั้นเป็นการแจ้ง event ที่เกิดขึ้นให้ Windows รับรู้เท่านั้น โดย Windows จะทำหน้าที่ตัดสินใจต่อไปว่าจะเรียกใช้ function ใดเป็นการตอบสนอง message นั้น ๆ ดังนั้นการประยุกต์ใช้ C++ เพื่อการเขียนโปรแกรมบน Windows จึงต้องมีการดัดแปลงไม่มากนัก ซึ่งก็ได้มี Compiler ที่สนับสนุนเทคนิคนี้ออกมาและเป็นที่นิยมคือ ชุด Borland C++ and Application Framework Version 3.1 ซึ่งเป็นรุ่นล่าสุดในขณะนี้ที่สนับสนุน การเขียนโปรแกรมบน Microsoft Window 3.1

ในการพัฒนา application ด้วย Borland C++ที่ทำนี้เราพัฒนาบน environment ดังนี้

1. IBM compatible computer with

-80386 to above CPU

-4 MB to above memory

-Hard disk with not less than 80 MB (for win3.1 and Borlandc++ 3.1 and Application Frameworks)

-VGA/SVGA controller and display monitor(color monitor is prefered)

-1.2/1.44 MB floppy disk drive

-printer

2 .Microsoft Window 3.1 (Thai Editon is prefered if you want to use Thai language)

3. Borlandc++ 3.1 and Application Frameworks (not essential for only execution)

สำหรับวัตถุประสงค์ของการพัฒนา project **นี้เป็นดังนี้**

1.**เรียนรู้** Object-Oriented Designe and Programming

2.**เรียนรู้** C++ language

3.**พัฒนา** Case Tool (data flow diagram)

บทที่2

OBJECT ORIENTED TECHNOLOGY

OBJECT ORIENTED PROGRAMING (OOP)

OOP เป็นวิธีการเขียนโปรแกรมที่ใช้แนวความคิดแบบ Object-Oriented โดยมองโปรแกรมเป็นระบบที่สนใจ และมีobjectเป็นสมาชิกในโปรแกรม ในobjectหนึ่งจะประกอบไปด้วยส่วนcodeคือรูทีนจัดการต่างๆ และส่วนdataซึ่งก็คือตัวแปรโปรแกรม ลักษณะการเขียนโปรแกรมแบบเดิมนั้น เราจะพยายามแบ่งงานทั้งหมดออกเป็นงานเล็กๆ แบ่งโปรแกรมออกเป็นprocedureหรือรูทีนย่อยๆ เพื่อจะจัดการส่วนต่างๆในโปรแกรม คำว่า โปรแกรมในแบบเดิมจึงหมายถึงเซ็ตของprocedure เมื่อมองโปรแกรมเป็นเซ็ตของ procedureแล้ว ขั้นแบบโปรแกรมจึงเป็นขั้นตอนของการจัดโครงสร้างของprocedu re ต่างๆในโปรแกรมตามลำดับก่อนหลังจากใหญ่ไปหาเล็ก เรียกว่าการออกแบบจากบนลงล่าง หรือ Top-down designed ในโปรแกรมแบบ OOP นั้น โปรแกรมมีนิยามเป็นเซ็ตของ OBJECT ออบเจกต์ถูกกำหนด ให้เป็นหน่วยใหม่ที่สร้างขึ้นมาเพื่อรวมเอาทั้งส่วนข้อมูล และส่วนของโค้ดที่จัดการกับข้อมูลนั้นไว้ด้วยกัน เมื่อนิยามของการ โปรแกรมเปลี่ยนไป การออกแบบโปรแกรมก็กลายเป็นการออกแบบโครงสร้างและความเกี่ยวข้องระหว่างออบเจกต์แต่ละตัวในโปรแกรม

OBJECT ออบเจกต์ คือปริมาณหนึ่งในระบบที่ประกอบด้วยองค์ประกอบสองส่วน คือ ข้อมูลและ โค้ดี โปรแกรม ส่วนข้อมูลใช้เก็บสถานะของตัวมันเองเรียกว่า data และส่วนโค้ดโปรแกรม ใช้ในการตอบสนองต่อออบเจกต์ตัวอื่นในระบบเดียวกันเรียกว่า method ออบเจกต์ใดๆในระบบจะสื่อสารกับออบเจกต์อื่นเพื่อให้บรรลุความต้องการของตน การสื่อสารนี้เป็นลักษณะ "ร้องขอ และ ตอบสนอง" เมื่อออบเจกต์ตัวหนึ่งขอความช่วยเหลือจากออบเจกต์หนึ่งเราเรียกว่า มันกำลังส่ง massage ไปยังออบเจกต์อื่น

CLASS คำว่า คลาส กับ ออบเจกต์ เป็นสิ่งคู่กัน เราจะดูว่าออบเจกต์มีลักษณะ characteristic อย่างไร เราสามารถดูได้จาก คลาส ลักษณะเฉพาะตัวของออบเจกต์เราเรียกว่า instance variable ซึ่งหมายถึงค่าตัวแปรเฉพาะของแต่ละออบเจกต์ในคลาส ขณะที่ class variable หมายถึง ค่าตัวแปรร่วมของคลาส ตัวอย่างเช่น สมมติว่าเราต้องการจะออกแบบเวิร์ดโปรเซสเซอร์โดยใช้ Object-oriented สักตัวหนึ่ง ขั้นแรกเราได้นิยามคลาสๆหนึ่งขึ้นมามีชื่อว่า paragraph คลาส paragraph นี้เป็นคลาสที่ใช้เก็บคุณสมบัติของข้อมูลหนึ่งย่อหน้าของเอกสาร ประกอบด้วยตัวแปรสองตัว คือ font และ text ตัวแปร font เป็นตัวแปรที่ใช้เก็บรูปแบบตัวอักษรที่จะปรากฏ ให้เห็นหน้าจอ และตัวแปร text เป็นอาร์เรย์ที่เก็บข้อความเอกสารในย่อหน้านั้น จากการพิจารณาจะพบว่า font นั้นเป็น class variable เพราะเป็นข้อมูลร่วมของคลาสparagrapgh ในขณะที่ text เป็น instance variable เพราะเป็นข้อมูลเฉพาะของ instance แต่ละตัว ในย่อหน้าหนึ่งอาจเก็บข้อความ "คุณรักผมหรือเปล่า" แต่อีกย่อหน้าอาจเก็บข้อความ "ผมรักคุณ" เนื่องจากหลักการ Object-oriented เป็นหลักการที่อาศัยการถ่ายทอดคุณสมบัติเป็นหลัก คลาสแต่ละคลาสในระบบจึงสามารถให้กำเนิดลูกหลานได้ เรียกลูกหลานของคลาสว่า subclass ซับคลาสจะรับเอาคุณสมบัติของคลาสที่ให้กำเนิดที่เรียกว่า parent class

หลักการโดยทั่วไปของ OOP

Encapsulation คือกระบวนการปกปิดความลับของ ออบเจกต์ การเข้าถึงค่าสถานะภายในออบเจกต์ จะกระทำได้โดยผ่านการเห็น ชอบจากออบเจกต์เจ้าของเสียก่อน นั่นก็ หมายความว่า การจะเข้าไปเปลี่ยนแปลงแก้ไขตัวแปรภายใน ออบเจกต์จะต้องกระทำผ่านmethod ของออบเจกต์ดังกล่าวเท่านั้น ตัวแปรใดๆภายในออบเจกต์จะมีสถานะ default เป็นข้อมูลส่วนตัว หรือค่า Private ข้อมูลแบบนี้จะไม่สามารถเข้าถึงได้โดยตรง การจะขอดูหรือเปลี่ยนแปลงค่า ต้องกระทำผ่าน procedure ที่เป็น methode ในออบเจกต์นั้นเท่านั้น ยกตัวอย่างโปรแกรมใน ภาษา C++ ต่อไปนี้

```
class  Point
{
        int  X,Y;
        public:
                int  GetX ()  { return X; }
                int  GetY ()  { return Y; }
};
```

จากโปรแกรมตัวอย่างเป็นคลาส Point ของออบเจกต์ที่เก็บข้อมูลจุดๆ หนึ่งบนจอภาพ การจะขอดูค่า X หรือ Y ในออบเจกต์นั้นไม่สามารถทำได้โดยตรง เช่น

```
printf("%d",X);
```

แต่ต้องผ่านการเรียกใช้เมดธอด int GetX () เสียก่อน ดังเช่น

```
printf("%d", APoint.GetX() );
```

เป็นต้น

ด้วยวิธีนี้ทำให้การทำความเข้าใจโปรแกรมค่อนข้างจะมีภาพที่ชัดเจนกว่ าเดิม อย่างน้อยก็ทำให้เรารู้ว่าออบเจกต์ตัวไหนในโปรแกรมกำลังต้องการทำอะไร แต่ก็ต้องแลกกับความเร็วในการทำ งานของโปรแกรมที่ต้องมีโอเวอร์เฮดสูงขึ้น

Inheritance เป็นการถ่ายยทอดคุณสมบัติของคลาส เป็นวิธีการสร้าง คลาสใหม่โดย การอาศัยรูปร่างของคลาสเดิม ถือเป็นคุณสมบัติที่ขาดไม่ได้ของ ภาษาแบบ Object oriented เพราะโปรแกรมในนิยามของ oop หมายถึงความ สัมพันธ์ของ object หากขาดซึ่งการถ่ายทอด คุณสมบัติ การสร้างชนิดข้อมูลหรือ คลาสใหม่ย่อมเป็นไปด้วยความยากลำบาก เราสามารถกำหนดชื่อเมดธอดใน คลา สลูกให้มีชื่อเดียวกับเมดธอดในคลาสพ่อได้ เพื่อลดความยุ่งยากในการกำหนดชื่อและ สามารถแสดงกริยาตอบสนองของ object ได้ชัดเจนยิ่งขึ้น ในภาษาแบบ object-oriented รุ่นใหม่มักจะมีคุณสมบัติการถ่ายทอดแบบหลายพ่อหลายี แม่ได้ด้วยซึ่งเรียก

ว่า Multiple inheritance คุณสมบัติที่ว่านี้หมายถึงการถ่ายทอดโดยรับเอา ลักษณะของคลาสมากกว่า 1 คลาส มาสร้างคลาสใหม่ ตัวอย่าง เช่น โปรแกรมเวิร์ดโปรเซสเซอร์ในปัจจุบัน มักจะรวมเอาความสามารถในการเก็บภาพกราฟฟิคไว้รวมกับเอกสารได้ หากจะ สร้างโปรแกรมที่มีคลาสชื่อ WordProcessor ขึ้นก็คงจะเป็นคลาสที่ถ่ายทอดมาจากคลาส 2 คลาส คือ TextEditor และ ImageProcessor เพราะมันมีลักษณะของคลาสทั้งสองรวมไว้ ด้วยกัน

Polymorphism ในระบบ object oriented การส่งเมสเสจแบบเดียวกันสามารถตอบสนองได้หลายแบบ และไม่จำเป็นต้องได้รับการตอบสนองเหมือนกัน ขึ้นอยู่กับ object ที่รับเมสเสจเป็นสำคัญ การยินยอมให้มีการตั้งชื่อโพรซีเจอร์ หรือธอดที่ซ้ำกันในโปรแกรม เป็นคุณสมบัติของ object oriented program ทุกตัวอยู่แล้ว เพียงแต่ในภาษาไม่เรียกส่วนนี้ว่าเป็น Polymorphism แต่รวมเข้ากับ คุณสมบัติการถ่ายทอดเพราะถือว่าในคลาสลูกที่รับเอาเมตธอด ของคลาสพ่อมานั้นสามารถดัดแปลงแก้ไขเมธธอดนั้นได้ตามใจ ใน c++ ก็มีการกำหนด overloading ซึ่งหมายถึงการประกาศชื่อฟังก์ชัน 2 ฟังก์ชั่นซ้ำกันแต่ทำงานคนละอย่าง เรียกเหล่าฟังก์ชั่นที่ใช้ชื่อชื่อเดียวกันนี้ว่า ฟังก์ชันโอเวอร์โหลด

Dynamic Binding คำว่า binding หมายถึง การที่ caller หรือ routine ที่เรียกใช้ได้มาซึ่ง address ของ routine ที่ถูกเรียกใช้ ในภาษาคอมพิวเตอร์แบบโพรซีเยอร์ การ binding จะเชื่อม addressของรูทีนที่ถูกเรียกใช้เข้ากับรูทีนที่เรียกใช้ แต่ในภาษาแบบ object oriented จะเชื่อมเอา address ของเมธธอด ของ object ที่ได้รับเมสเสจกับ ของเมสเสจ ที่ส่งไป การ binding สามารถเกิดขึ้นได้ทั้งขณะที่ทำการ compile program หรือขณะ run ถ้าการ binding ทำขณะทำการ compile จะเรียกว่า static binding ส่วนขณะ run program จะเรียกว่า dynamic binding ในภาษาแบบ procedural language การเกิด binding จะมี เฉพาะ static binding อย่างเดียว ในขณะที่ภาษาแบบ object oriented มีทั้งแบบ static และ dynamic

เหตุที่ต้องมี dynamic binding เนื่องจากในบางกรณี คอมไพเลอร์ไม่สามารถตรวจ สอบ สอบได้ว่า object ตัวใดกันแน่ที่จะได้รับเมสเสจจนกว่าจะมีการ run

Class Library หมายถึงชุดหรือโครงสร้างความสัมพันธ์ของคลาสต่าง ๆ ในระบบ object oriented มีลักษณะคล้ายกับ function library ของภาษา c class library เป็นที่เก็บนิยามทั้งหมดของคลาสทุกคลาสในสภาพแวดล้อมแบบ object oriented ถือเป็นหัวใจ หลักที่ขาดไม่ได้ของ object oriented โปรแกรมเมอร์สามารถลดเวลาในการพัฒนาโปรแกรม ไปได้โดยอาศัยคลาสที่เป็นเครื่องมือในการพัฒนาโปรแกรมที่ Build in ในภาษานั้น ไม่ว่าจะ เป็นคลาสของสแต็ก ลิงค์ลิส อาเรย์หลายมิติ รวมไปถึงคลาสของวินโดวส์ ข้อแตกต่างระหว่าง function library กับ class library คือ function library ไม่สามารถเปลี่ยนแปลงแก้ไขได้ แต่เราสามารถแก้ไข class library ได้ตามต้องการ

บทที่3

พื้นฐาน Windows Programming และ OWL technic

ความรู้เบื้องต้นเกี่ยวกับ Windows Programming technic

การจัดการเกี่ยวกับ Message, Command และ Control ต่าง ๆ ใน Window และ Dialog Box

หากเราต้องการจัดการเกี่ยวกับ Message, Command และ Control ต่าง ๆ ใน Window และ Dialog box เราจำเป็นต้องสร้าง Member function ของ Window หรือ Dialog Box นั้นเพื่อจัดการกับ Message, Command และ Control ต่างๆ โดยปกติที่ใช้กันบ่อย ๆ จะมี 3 ประเภทดังมีรายละเอียดต่อไปนี้

1.Windows Message (WM)

Windows Message หมายถึง System Message ที่มีอยู่เดิมใน Windows และถูกส่งมายัง Application

Identifier จะมีรูปแบบเป็น WM_XXXX เช่น WM_PAINT, WM_SIZE เป็นต้น Range ของ Windows Message มีดังนี้

Constant	Meaning
WM_FIRST	ค่าเริ่มต้นของ reserve range
WM_INTERNAL	ค่าเริ่มต้นของ Objectwindow reserverange
WM_COUNT	จำนวนของ reserve message
WM_RESERVED	WM_INTERNAL-WM_FIRST

ซึ่งค่าเหล่านี้จะเป็นเลขฐาน 16 ที่ Microsoft Window ได้ define ไว้แล้วในการอ้างถึง message ต่าง ๆ ใน Windows

2.Windows Command Message (CM)

เป็น Message ที่เกิดจาก Command ต่าง ๆ โดยสิ่งที่จะทำให้เกิด Command อาจจะเป็น Menu หรือ Accelerators ก็ได้

Identifier จะมีรูปแบบเป็น CM_XXXX เช่น CM_FILEOPEN, CM_EXIT เป็นต้น

Range ของ Windows Command มีดังนี้

Constant	Meaning	Values(0x)
CM_FIRST	ค่าเริ่มต้นของ user CM	0xA000
CM_INTERNAL	ค่าเริ่มต้นของ reserved internal CM	0xFF00
CM_COUNT	จำนวนของ user CM	0x6000
CM_RESERVED	N/A	CMINTERNAL-CMFIRST

ค่าของ CM_XXXX ผู้ใช้สามารถ define ขึ้นเพื่ออ้างถึง Command ต่าง ๆ ที่สร้างขึ้น โดย Command มักจะเกิดจาก 2 แหล่งคือ Menu หรือ Accelerator ซึ่ง CM_XXXX ที่ define ต้องมีค่าน้อยกว่า CM_RESERVED

3.Window Control Message(ID)

เป็น Message ที่เกิดจาก Control ต่าง ๆ ใน Windows เช่น Bottom, Edit Box, List Box เป็นต้น

Identifier จะมีรูปแบบเป็น ID_XXXX เช่น IDOK, IDYES, IDCANCEL, ID_EXIT เป็นต้น

Range ของ Windows Control มีดังนี้

Constant	Meaning	Value(0x)
ID_FIRST	ค่าเริ่มต้นของ child ID message	0x8000
ID_INTERNAL	สงวนไว้สำหรับ internal use	0x8F00
ID_COUNT	จำนวนของ child ID message	0x1000
ID_RESERVED	N/A	ID_INTERNAL-ID_FIRST

ค่าของ ID_XXXX ผู้ใช้สามารถ define ใหม่ได้เพื่ออ้างถึง Control ต่าง ๆ เช่น Button, Listbox โดยจะเป็น identifier ของ Control นั้น ๆ เมื่อ Control นั้นถูกเรียกก็จะเกิด message ที่มีค่าเท่ากับ ID_FIRST+ID_XXXX (ID ของ Control ที่ถูกเลือก)

หมายเหตุ

การ define ค่าของ CM_XXXX หรือ ID_XXXX ควรให้ค่าตั้งแต่ 100 ขึ้นไป และควรมีค่าน้อยกว่า CM_RESERVED และ ID_RESERVED เนื่องจาก ค่าที่ต่ำกว่า 100 นั้น Window และ Object Window นำไปใช้ในการ define Standard Command และ Standard Identifier Control เช่น IDOK, IDCANCEL เป็นต้น

การกำหนด Function ให้จัดการกับ Message ต่าง ๆ ที่ต้องการ

รูปแบบของการประกาศ member function ใน class ให้เป็นตัวจัดการเกี่ยวกับ Message ที่ต้องการ กำหนดดังนี้

FunctionName(TMessage& Msg) = [message value]

เช่น WMSize(TMessage& Msg) = [WM_FIRST + WM_SIZE];

CMExit(TMessage& Msg) = [CM_FIRST + CM_EXIT];

HandleShow(TMessage& Msg) = [ID_FIRST + ID_SHOW];

Windows Message ที่ใช้บ่อย ๆ

WM_PAINT

message นี้จะถูกส่งไปยัง window procedure เมื่อพบว่าเนื้อที่บน window บางส่วน หรือทั้งหมดจำเป็นต้องวาดใหม่ เช่น กรณีที่ถูก window อื่นทับ หรือต้องการที่จะเปลี่ยนการแสดงผลบน ใหม่ การคืนขนาดจาก minimize เป็นต้น

ในการเขียนโปรแกรมโดยใช้ object window จะมี member function ของ TWindow ซึ่งทำหน้าที่นี้ คือ WMPaint ซึ่ง WMPaint จะทำการ call ไปยัง Member function Paint เพื่อทำหน้าที่เขียนหน้าจอใหม่อีกต่อหนึ่ง

WM_SIZE

message นี้จะถูกส่งมาเมื่อมีการเปลี่ยนแปลงขนาดของ Window โดยจะมี parameter ที่ส่งมาดังนี้

WParam จะระบุชนิดของการเปลี่ยนขนาดที่ต้องการดังนี้

Value	Description
SIZE_MAXIMIZE	เมื่อ Window Maximize
SIZE_MINIMIZE	เมื่อ Window Minimize
SIZE_RESTORED	เมื่อ Window มีการ Resized แต่ไม่ Maximize หรือ Minimize
SIZE_MAXHIDE	Message จะถูกส่งไปทุก pop-up Window เมื่อมีบาง Window ที่ Maximize
SIZE_MAXSHOW	Message จะส่งไปทุก ๆ pop-up window เมื่อ Window ถูก restored เป็น former size

LOWORD(lParam) เป็นขนาดความกว้างใหม่ของ Client area

HIWORD(lParam) เป็นขนาดความสูงใหม่ของ Client area

หมายเหตุ function ที่จัดการ Message นี้ต้องตั้งชื่อว่า WMSize และควร return 0 หากทำงานเกี่ยวกับการ Resize เรียบร้อยแล้ว

WM_LBUTTONDBCLK, WM_LBUTTONDOWN, WM_LBUTTONUP, WM_MOUSEMOVE message ทั้ง 4 นี้จะเกิดขึ้นเมื่อ ผู้ใช้ double click, press, release left mouse button and move mouse ตามลำดับ โดยมี parameter ที่ส่งมาดังนี้

WParam parameter combination

Value	Description
MK_CONTROL	Set ถ้า CTRL key ถูกกด
MK_LBUTTONSet	ถ้า Left mouse button ถูกกด
MK_MBUTTON	Set ถ้า Middle mouse button ถูกกด
MK_RBUTTON	Set ถ้า Right mouse button ถูกกด
MK_SHIFT	Set ถ้า SHIFT key ถูกกด

LOWORD(lParam) จะเป็นค่า x-coordinate ของ cursor

HIWORD(lParam) จะเป็นค่า y-coordinate ของ cursor

หมายเหตุ x,y-coordinate เป็นค่าตำแหน่งเมื่อเทียบกับมุมซ้ายบนของ windows ไม่ใช่ใช่จอภาพ function ที่ทำการ process message นี้ควรจะ return 0

WM_INITMENU

message นี้จะถูกส่งเมื่อ menu เริ่ม active เรามักใช้ message นี้ ในการเปลี่ยนแปลงรูปแบบของ menu เช่น ใส่ Check mark หรือเอาออก ก่อนแสดง menu ซึ่งมี parameter ดังนี้

wParam เป็น handle ของ menu ที่จะ initialize

WM_SETCURSOR

message นี้จะถูกส่งเมื่อ cursor ของ mouse ที่เคลื่อนที่อยู่บน window ที่ไม่ใช่ caption มักจะใช้ในการ set cusor ให้อยู่ในลักษณะต่าง ๆ ตามที่ต้องการ หาก function ของเราทำการ set cursor แล้วให้ return TRUE มิฉะนั้น DefWindowProc จะทำการ set cursor ใหม่ ซึ่งมี parameter ดังนี้

wParam เป็น handle ของ window ที่ cursor อยู่

LOWORD (lParam) เป็น hit-test code เช่น HTCLIENT, HTHSCROLL, HTVSCROLL, HTBORDER

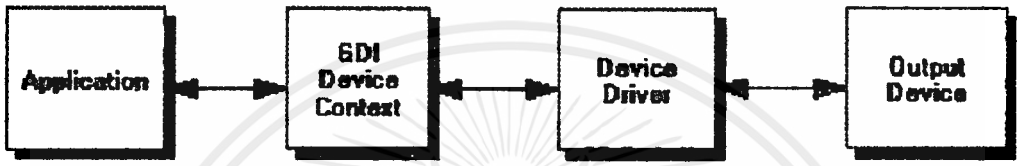
HIWORD(lParam) เป็นหมายเลขของ mouse-message

Device Context (DC)

ใน Windows environment จะมี Graphic Device Interface (GDI) ทำหน้าที่เป็นตัวเชื่อมโยงระหว่างตัว Application กับตัว Output Device ทำให้ผู้พัฒนา application ไม่จำเป็นต้องคำนึงถึงชนิด output device ที่ program ต้องติดต่อด้วย เช่น output เป็น printer หรือ screen หรือ printer เป็น printer รุ่นใด เพราะคำสั่งในการติดต่อจะเหมือนกันหมด เพียงแต่เลือก Device Context ให้ถูกชนิดตรงกับความต้องการ

Device Context เป็น device ที่เชื่อมระหว่างตัว Application กับ Device Driver และ Output device

ในการติดต่อกับ output device เราต้องติดต่อโดยผ่าน DC เท่านั้น ซึ่ง window แต่ละ window ก็จะมี DC เป็นของตัวเอง ซึ่งเราสามารถใช้ handle ของ DC นี้เป็นตัวชี้ถึง device context ที่ต้องการ



รูปที่ 3.1 แสดงความสัมพันธ์ระหว่าง Application , Device Context, Device Driver และ Output Device

การใช้ Object Windows Libraray ช่วยในการเขียนโปรแกรม

การสร้าง Window

ถ้าเราต้องการสร้าง window ที่ไม่ใช่ window เปล่า ๆ เราต้องทำการ derive class ของ Twindow ขึ้นมา โดย Constructor ต้องมีการผ่านค่าไปให้ Base Class คือ Twindow เราสามารถกำหนดลักษณะต่าง ๆ ของ window ใหม่ได้ที่ Constructor นี้เช่น การใช้คำสั่ง AssignMenu สร้าง Menu ขึ้นมา การกำหนดค่าในตัวแปร Attr ซึ่งเป็น data member ของ class เช่น ตำแหน่งของ Window(Attr.X, Attr.Y) ขนาด(Attr.W, Attr.H) รูปแบบของ Window(Attr.style) ซึ่ง style จะมี define ไว้ให้เลือกใช้ดังนี้

Style	Meaning
DS_LOCALEDIT	ระบุว่าการ edit control ใน dialog box จะใช้หน่วยความจำในส่วน data segment ของ application เนื่องจากใน defualt ที่มีอยู่ edit control ทั้งหมดใน dialog box จะใช้หน่วยความจำข้างนอก ในลักษณะนี้สามารถบีบโดยการเติม flag ของ DS_LOCALEDIT ไปที่ Style สำหรับ dialog box ซึ่งถ้า flag นี้ไม่ถูกใช้ ตัว message 2 ตัวคือ EM_GETHANDLE และ EM_SETHANDLE ก็ไม่ถูกใช้ เนื่องจากเนื้อที่ที่เก็บสำหรับ control จะไม่อยู่ใน data segment ของ application ในลักษณะนี้จะไม่มีผลต่อ edit control ที่ถูกสร้างภายนอก dialog box

DS_MODALFRAME จะสร้าง dialog box แบบ modal dialog box ทั้ง frame ซึ่ง ซึ่งสามารถรวมเอาทั้ง title bar และ System menu โดยระบุรูปแบบให้เป็น WS_CAPTION และ WS_SYSMENU

DS_NOIDLEMSG ยกเลิก WM_ENTERIDLE message ซึ่ง windows ควรจะสั่งไปที่ owner ของ dialog box ในขณะที่ dialog box ถูกแสดงขึ้นมา

DS_SYSMODAL สร้าง system-modal dialog box

WS_BORDER สร้าง window ที่มีขอบ (border)

WS_CAPTION สร้าง window ที่มี Title Bar (เป็น window ชนิด WS_BORDER style)

WS_CHILD สร้าง child window ซึ่งไม่สามารถกับ WS_POPUP style

WS_CHILDWINDOW สร้าง child window ซึ่งมีรูปแบบของ WS_CHILD

WS_CLIPCHILDREN แยกพื้นที่ที่ถูกจองโดย child window ขณะทำการวาดรูปภายใน parent window ซึ่งจะถูกใช้เมื่อกำลังทำการสร้าง parent window

WS_CLIPBLINGS clip child windows ที่ relative กัน หมายถึง เมื่อแต่ละ child window ได้รับ message WM_PAINT ตัว top level child window ก็จะถูก clip ให้ออกจากขอบเขตของ child window ที่ถูก update (ถ้าไม่กำหนด WS_CLIPSIBLING style และ child window มีการเหลื่อมกัน ก็เป็นไปได้ที่เวลาวาดรูปใน client area ของ child window หนึ่ง แล้ว client area ของ child window รอบข้างอาจถูกเขียนทับไปด้วย) สำหรับ WS_CLIBSIBLINGS style นี้จะใช้เฉพาะกับ WS_CHILD style เท่านั้น

WS_DISABLE สร้าง window ที่มีการ initial ให้เป็น disable

WS_DLGFRAME สร้าง window แบบ modal dialog box frame แต่ไม่มี title

WS_GROUP ระบุถึง control แรกของกลุ่ม controls ซึ่ง user สามารถเคลื่อนย้ายจาก control หนึ่งไปยังอันต่อไปโดยการใช้ arrow keys ซึ่ง control ทั้งหมดจะถูก define ไว้กลุ่มเดียวกัน และเมื่อมี control ตัวต่อไปเข้ามาก็จะเป็นการสิ้นสุดการ control กลุ่มนี้แล้วเริ่มกลุ่มใหม่

WS_HSCROLL สร้าง window ที่มี scroll bar ในแนวนอน

WS_ICONIC สร้าง window ซึ่งมีสภาพเริ่มต้นเป็น icon ใช้กับ WS_OVERLAPPED เท่านั้น

WS_MAXIMIZE สร้าง window ขนาด maximize

WS_MAXIMIZEBOX สร้าง window ที่มี box ขนาด maximize

WS_MINIMIZE สร้าง window ขนาด minimize

WS_MINIMIZEBOX สร้าง window ที่มี box ขนาด minimize

WS_OVERLAPPED สร้าง overlapped window ที่มี caption และ border

WS_OVERLAPPEDWINDOW สร้าง overlapped window ที่มี WS_OVERLAPPED, WS_CAPTION, WS_SYSMENU, WS_THICKFRAME, WS_MINIMIZEBOX, WS_MAXIMIZEBOX style

WS_POPUP สร้าง popup window (ไม่สามารถใช้ร่วมกับ WS_CHILD style ได้)

WS_POPUPWINDOWสร้าง popup window ที่มี WS_POPUP, WS_BORDER, WS_SYSMENU

WS_SIZEBOX สร้าง window ที่มี size box จะถูกใช้กับ window ที่มี title bar หรือมี vertical scroll bar หรือ horizontal scroll bar เท่านั้น

WS_SYSMENU สร้าง window ที่มี box ของ sysmenu ใน tilte bar ถ้าใช้กับ child window จะสร้าง close box แทนการสร้าง system box

WS_TABSTOP จะระบุว่าสามารถที่จะใช้ TAB key เพื่อเคลื่อนย้ายไปมาระหว่าง control ได้

WS_THICKFRAME สร้าง window ที่มี thick frame ซึ่งจะทำให้เปลี่ยนขนาด window ได้

WS_VISIBLE สร้าง window ซึ่งจะถูก initial ให้มองเห็นได้แต่แรก นำไปใช้กับ overlapped window และ popup window สำหรับ overlapped window นั้น parameter Y จะถูกใช้สำหรับ functoin ShowWindow

WS_VSCROLL สร้าง window ที่มี scroll bar ในแนวตั้ง

การลงทะเบียน Window Class ใน OWL

โดยปกติ OWL จะทำการ initialize ค่าต่างๆ (default) ของ window class ไว้แล้ว ถ้าเราต้องการจะเปลี่ยนแปลงเราสามารถทำได้โดยการสร้าง member function GetWindowClass ใน class ของ window ที่เราต้องการเปลี่ยนแปลง แทน function เดิมใน base class เช่นถ้าเราต้องการเปลี่ยนสีพื้นของ window และต้องการเปลี่ยน icon ของ application ให้เป็น icon ที่เราสร้าง

```
void  TExampleWindow::GetWindowClass (WNDCLASS& WndClass)
{
  TWindow::GetWindowClass (WndClass);
  WndClass.hbrBackground = (HBRUSH) COLOR_APPWORKSPACE+1;
  WndClass.hIcon = LoadIcon (GetApplication()->hInstance, "icon");
};
```

การเรียกใช้ Dialog Box

เมื่อเราสร้าง dialog box แล้ว เราสามารถที่จะเรียก dialog box ขึ้นมาใช้งานได้โดยการใช้คำสั่ง ExecDialog ซึ่งจะเป็น member function ของ class TModule ซึ่งเป็น base class ของ TApplication การเรียกใช้ทำได้โดย การใช้คำสั่ง

```
GetApplication()->ExecDialog (new TDialog (AParent, DlgName));
```

บทที่4

KMIT'L Data Flow Diagram Specification

Data Flow Diagram Editing Tool ซึ่งเป็นส่วนหนึ่งของ Case Tool ส่วนในด้าน Structure Techniques คือ Structure analysis :ซึ่งจะเป็นส่วน front end ของโปรเจคการพัฒนาระบบ ซึ่งต้องหาความต้องการของผู้ใช้ และจัดทำเอกสาร เพื่อใช้ในส่วนอื่น ๆ ต่อไป

ซึ่ง Tool ที่เราจะสร้างขึ้นนี้ (KMIT'L Data Flow Diagram) จะช่วยในการเขียนและออกแบบ Data Flow Diagram โดยจะเป็น Tool ที่ฉลาด รู้ประเภทและลักษณะของเอนทิตี้ที่แสดงด้วยภาพได้ เช่น data flow diagram รูปหนึ่งที่แสดงการส่งเอกสารจากงาน A ไปยังงาน B โดยใช้ลูกศรแสดงการส่งเอกสาร ถ้าหากเราลบงาน A ออกจากแผนภาพ ลูกศรแสดงการส่งเอกสารก็จะต้องถูกลบไปด้วยโดยอัตโนมัติ หรือ หากมีการย้าย object ที่เชื่อมกันด้วยลูกศร ลูกศรก็จะยืดขนาดตามด้วย นอกจากนี้ยังสามารถที่จะทำการ link ไป Sub Data Flow Diagram ได้ จากรูปของ process หนึ่งๆ

จากสิ่งที่เรากำหนดลักษณะ program ที่จะสร้างขึ้นข้างบนนี้ เราสามารถที่จะสรุปถึง specification ที่ต้องการของโปรแกรม KMIT'L Data Flow Diagram ที่สร้างขึ้นให้ทำงานบน Microsoft Windows เป็นข้อๆได้ดังนี้คือ

1.เป็นโปรแกรมช่วยวาดภาพ data flow diagram ตามแบบของ Yourdon

2.ใช้คุณลักษณะ MDI ของ Windows นั่นคือเราสามารถที่จะเปิดหน้าต่างที่จะใช้วาดรูป หรือใช้ดูได้หลายหน้าต่าง แต่แต่ละหน้าต่างที่เปิดจะต้องมีความสัมพันธ์กัน คืออยู่ใน project เดียวกัน หน้าต่างที่เปิดเพิ่มจะต้องเป็น Sub-data flow diagram ของ process ใด process หนึ่งใน chart (window) ที่เปิดมาแล้วเท่านั้น (นั่นคือไม่สามารถที่จะเปิด data flow diagram ที่ไม่เกี่ยวข้องกัน ขึ้นมาดูพร้อมกันได้)

3.รูปที่จะวาดบนโปรแกรมมีดังนี้คือ process, terminator, storage (data store) และ flowed data (data flow) ซึ่งรูปในแต่ละรูปสามารถที่จะพิมพ์ชื่อกำกับได้

4.ในส่วนของรูป process สามารถที่จะ link ไป Sub-data flow diagram file ได้ และสาทารถที่จะตัด link ที่เชื่อมอยู่ได้

5.chart ที่ถูกเปิดมาจากการ link ของ process (chart ที่เป็น Sub-data flow diagram) ที่ caption bar ของ window (chart) นั้นจะบอกด้วยว่ามาจาก process พ่อชื่ออะไร หมายเลขเท่าไร ยกตัวอย่างเช่น chart เป็น Sub-data flow diagram ที่มาจาก process พ่อหมายเลข 3 ชื่อ Tax Calculation ที่ caption bar ของ chart (window) จะบอกว่า SUB-DFD : 3 Tax Calculation

6.process แต่ละรูปจะมีหมายเลขกำกับ และรูป process ที่อยู่ใน Sub-data flow diagram ที่มาจาก process หนึ่งจะมีหมายเลขที่สัมพันธ์กันด้วย เช่น รูป process ที่อยู่ใน Sub-data flow diagram ที่มาจาก process พ่อหมายเลข 2 จะมีหมายเลข 2.1, 2.2, 2.3, ... และสำหรับรูป process ที่อยู่ใน Sub-data flow diagram ที่มาจาก process พ่อหมายเลข 2.1 จะมีหมายเลข 2.1.1, 2.1.2, 2.1.3, ... โดยหมายเลขนี้จะทำการเขียนให้อัตโนมัติโดยโปรแกรมในแต่ละครั้งที่เปิด

7.สามารถที่จะเคลื่อนย้ายภาพ (process, terminator,storage)ไปที่ใดในพื้นที่วาดภาพก็ได้ และสำหรับภาพที่มี flowed data (เส้นลูกศร) ติดอยู่ รูป flowed data จะเคลื่อนย้ายตามไปด้วย

8.สามารถที่จะลบภาพใดก็ได้ และสำหรับภาพที่มี flowed data (เส้นลูกศร) ติดอยู่ รูป flowed data จะถูกลบไปด้วย

9.รูป process, terminator, storage สามารถ set สีให้มีสีต่างกันได้ เช่น รูป process อาจจะมีสีหนึ่ง รูป terminator อาจจะมีอีกสีหนึ่ง และ รูป storage อาจจะมีอีกสีหนึ่งได้

10.เมื่อทำการเปิด project ขึ้นมาใหม่ project ที่มีอยู่เดิม (chart window ทั้งหมดที่เปิดขึ้น) จะถูกปิดทั้งหมด (จะมีการถามเพื่อความแน่ใจว่าต้องการปิด project เดิมจริง และสำหรับ file ที่ยังไม่ได้ save ก็จะมีการถามถึงความต้องการที่จะ save)

11.สามารถที่จะทำการพิมพ์ภาพที่วาดขึ้นออกทางเครื่องพิมพ์ได้

12.สามารถที่จะทำการจัดเก็บภาพที่สร้างขึ้นมาทั้งหมดลง file ได้

บทที่5

User Interface ของ Program

สำหรับส่วน user interface ของโปรแกรม KMIT'L Data Flow Diagram เรา เป็น 2 ส่วน ในส่วนแรกเป็น ฟังก์ชันบน Menu Bar และส่วนที่ 2 เป็นฟังก์ชัน Mode Button

ฟังก์ชันบน Menu Bar

บนส่วนของ Menu Bar ของโปรแกรม KMIT'L Data Flow Diagram ประกอบด้วยฟังก์ชันหลักๆคือ

-File

-Color

-Link

-Option

-Window

ฟังก์ชันบน popup menu File

+ New Project ฟังก์ชันทำหน้าที่สร้าง data flow diagram file (project file) ใหม่ (สร้างขึ้นใหม่)

+ Open Project ฟังก์ชันทำหน้าที่เปิด data flow diagram file (project file) ที่ต้องการ (ที่ถูกสร้างไว้แล้ว)

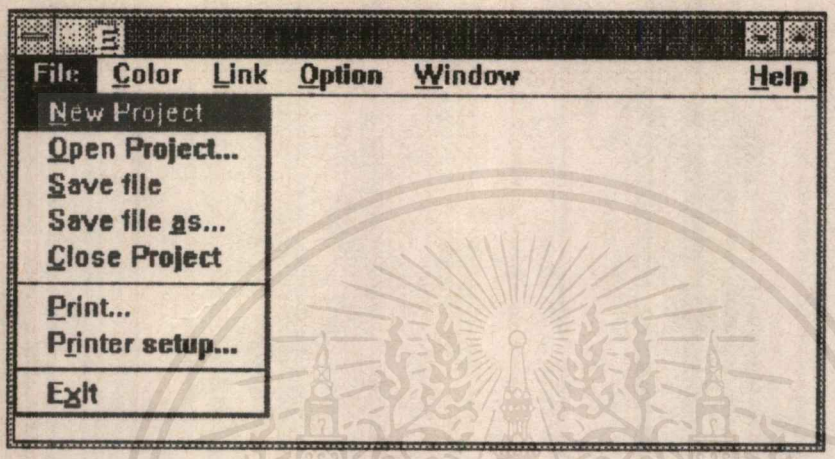
~ + Save file ฟังก์ชันทำหน้าที่ จัดเก็บ file ที่เขียนอยู่ขณะนั้นลงบน disk

+ Save file as ฟังก์ชันทำหน้าที่ จัดเก็บ file ที่เขียนอยู่ขณะนั้น ในชื่อใหม่

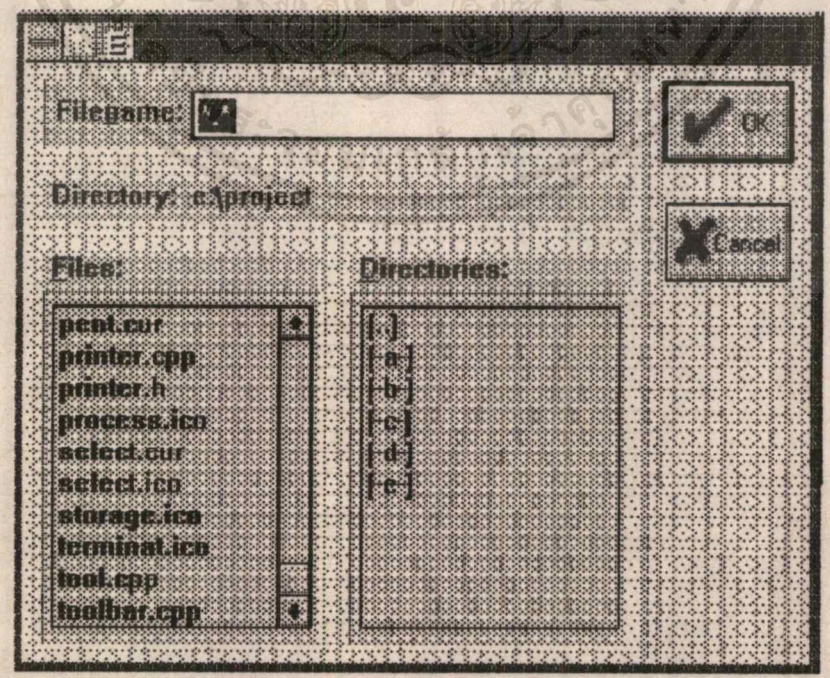
+ Print setup ฟังก์ชันทำหน้าที่ set up printer

+ Print ฟังก์ชันทำหน้าที่ พิมพ์ข้อมูลออกทางเครื่องพิพม์ (printer)

+ EXIT ฟังก์ชันทำหน้าที่ออกจาก program



รูปที่5.1 แสดง popup menu ของ File



รูปที่ 5.2 แสดง file dialog box

ฟังก์ชันบน popup menu Color

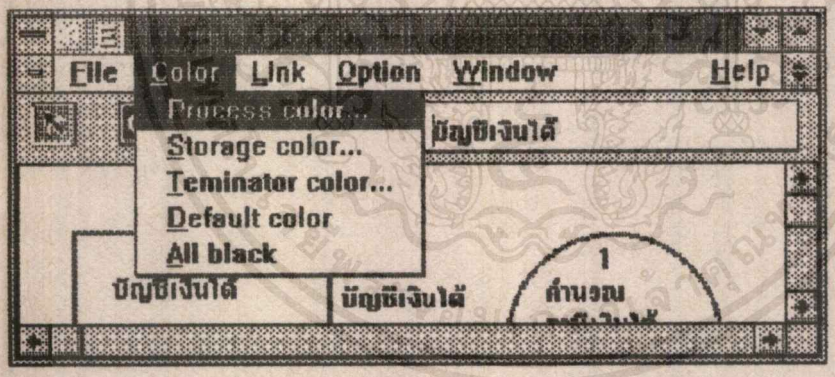
+ Process color ฟังก์ชันทำหน้าที่ set สีของรูป process ที่จะวาด

+ Storage color ฟังก์ชันทำหน้าที่ set สีของรูป storage ที่จะวาด

+ Terminate color ฟังก์ชันทำหน้าที่ set สีของรูป terminator ที่จะวาด

+ Default color ฟังก์ชันทำหน้าที่ set สีของรูป ต่างๆ ที่จะวาด โดยได้ทำการกำหนดสีของรูปต่างๆไว้แล้ว

+ All Black ฟังก์ชันทำหน้าที่ set สีของรูปต่างๆที่จะวาด ให้เป็นสีขาวดำหมด

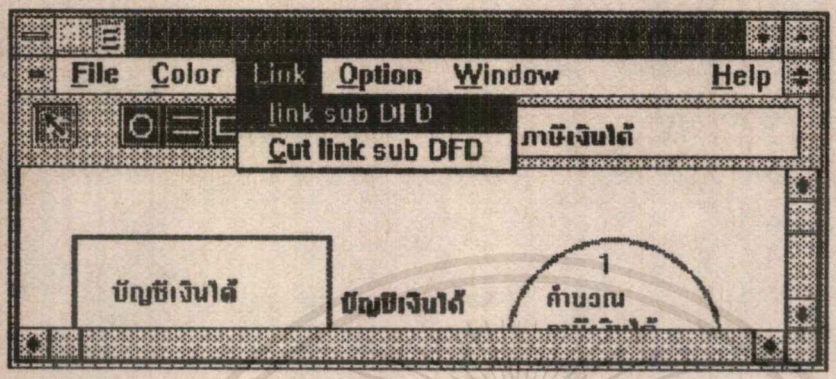


รูปที่ 5.3 แสดง popup menu ของ Color

ฟังก์ชันบน popup menu Link

+Link sub DFD ฟังก์ชันทำหน้าที่ link Process ที่ถูกเลือกขณะนั้นกับ sub data flow diagram ของ process นั้น ถ้า process นั้นยังไม่มีการ link ไป sub data flow diagram file ใด จะให้ทำการใส่ชื่อ file ที่ต้องการ link (ถ้า link กับชื่อ file ที่ไม่มีอยู่จริง จะเป็นการสร้าง file ใหม่ขึ้นมา) ถ้า process นั้นมีการ link อยู่แล้ว จะทำการเปิด file ที่ link ขึ้นมา

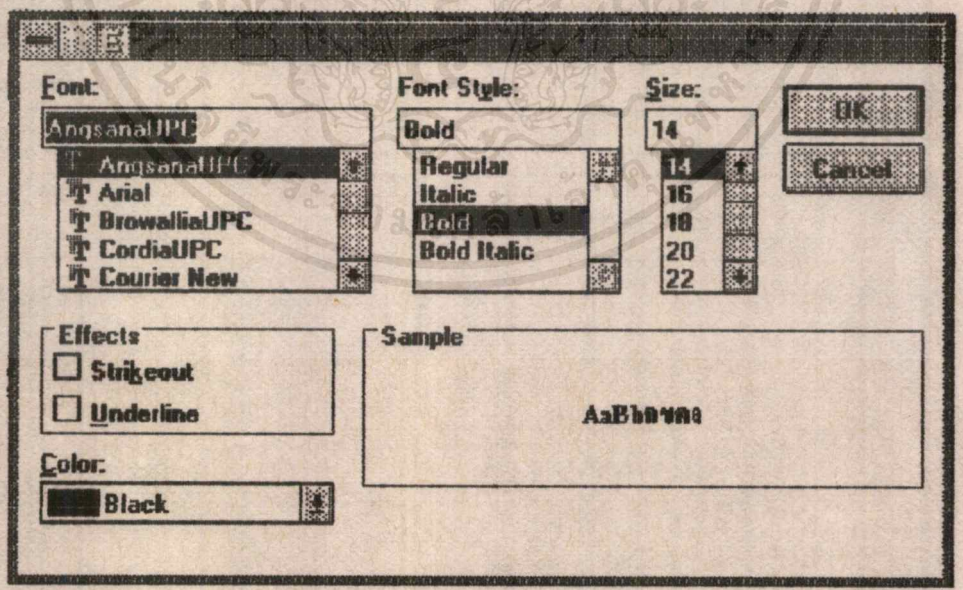
+Cut Link sub DFD ฟังก์ชันทำหน้าที่ตัดชื่อ file ที่ link กับ process ที่ถูกเลือกออก



รูปที่ 5.4 แสดง popup menu ของ Link

ฟังก์ชันบน popup menu Option

+ Fonts ฟังก์ชันทำหน้าที่ set text fonts



รูปที่ 5.5 แสดง fonts dialog box

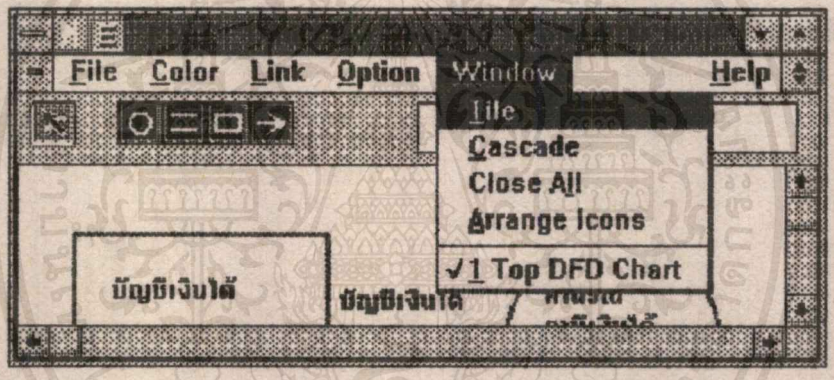
ฟังก์ชันบน popup menu Window

+Tile ฟังก์ชันทำหน้าที่จัดเรียง windows ที่เปิดขึ้นทั้งหมดใน program ในลักษณะ tile

+Cascade ฟังก์ชันทำหน้าที่จัดเรียง windows ที่เปิดขึ้นทั้งหมดใน program ในลักษณะ cascade

+Close All ฟังก์ชันทำหน้าที่ปิด windows ที่เปิดขึ้นมาทั้งหมดใน program

+Arrange Icons ฟังก์ชันทำหน้าที่จัดเรียง icons ของ windows ที่เปิดขึ้นมาทั้งหมดใน program



รูปที่ 5.6 แสดง popup menu ของ Window

ฟังก์ชันบนปุ่มต่าง ๆ (Mode Button)

MODE ในการทำงานของ DATA FLOW DIAGRAM EDITOR

SELECTION & MODIFY MODE

การ click mouse ที่ปุ่ม selection & modify mode button จะทำให้อยู่ในโหมดการทำงานนี้ ซึ่งในโหมดการทำงานนี้ มีฟังก์ชันในการทำงานอยู่ 3 ฟังก์ชันคือ

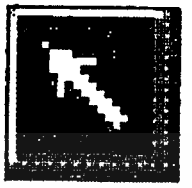
Move Object เมื่ออยู่ในโหมดการทำงานนี้ (selection and modify mode) เราสามารถที่จะเคลื่อนย้ายภาพต่างๆ (process, storage, terminator) ไปที่ใดก็ได้ในตำแหน่งวาดภาพ โดยทำการ click object ที่ต้องการจะเคลื่อนย้าย จากนั้นทำการ drag object นั้นไปยังตำแหน่งที่ต้องการ ในการ move object นี้ถ้า object นั้นมีมีเส้น(flowed data) ติดกับมันด้วย เส้นที่ติดกับ object นั้นก็จะ move ตามไปด้วย

Delete Object เมื่ออยู่ในโหมดการทำงานนี้ เราสามารถที่จะลบ ภาพต่างๆ (process, storage, terminator)ได้ โดยทำการ click object ที่ต้องการจะลบ จากนั้นทำการ ลบโดยการ click mouse button ทางขวา

Change Name เมื่ออยู่ในโหมดการทำงานนี้ เราสามารถที่จะเปลี่ยนชื่อของ object ได้ เมื่อเรา click object ที่ต้องการแล้ว ชื่อของ object จะแสดงที่ Name Window จากนั้นเราสามารถที่จะแก้ไขชื่อของ object นั้นได้โดยพิมพ์ชื่อที่ต้องการ

Link Sub Data Flow Diagram File เมื่ออยู่ในโหมดการทำงานนี้ เราสามารถที่จะ link sub data flow diagram ของ process object ที่ถูกเลือกได้ โดยการทำการ double click ที่ process object ที่ต้องการ link ไป sub data flow diagram ถ้า process ที่ถูกเลือกนั้น มีการ link กับ sub data flow diagram file อยู่แล้ว ก็จะ

ทำการเปิด sub data flow diagram file นั้น แต่ถ้า process ที่ถูกเลือกนั้นไม่ได้มีการ link กับ file ใด จะมีการให้ใส่ชื่อ file ที่ต้องการ link (ถ้าชื่อ file นั้นไม่มีอยู่ใน system disk จะทำการสร้าง file นั้นใหม่)

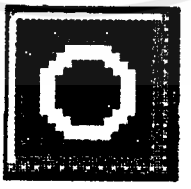


รูปที่ 5.7 แสดงปุ่ม selection and modify mode

DRAWING MODE

ในโหมดการทำงานนี้ เป็นโหมดเกี่ยวกับการวาดรูปต่างๆ ซึ่งประกอบไปด้วย

PROCESS DRAWING MODE การ click mouse ที่ปุ่ม process drawing mode จะทำให้อยู่ในโหมดการทำงานนี้ เมื่ออยู่ในโหมดการทำงานนี้เราสามารถวาดรูป process ได้โดยการ click mouse button ทางด้านซ้าย ลงบนตำแหน่งที่ต้องการวาด หลังจากที่วาดเสร็จแล้วเราอาจจะใส่ชื่อของรูป process นั้นได้เลยโดยการพิมพ์ชื่อ (ชื่อที่พิมพ์จะแสดงให้เห็นบน Name Window ด้วย)

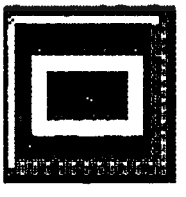


รูปที่ 5.8 แสดงปุ่ม PROCESS DRAWING MODE

STORAGE DRAWING MODE การ click mouse ที่ปุ่ม storage drawing mode จะทำให้อยู่ในโหมดการทำงานนี้ เมื่ออยู่ในโหมดการทำงานนี้เราสามารถวาดรูป storage ได้โดยการ click mouse button ทางด้านซ้าย ลงบนตำแหน่งแหน่งที่
ต้องการวาด หลังจากที่วาดเสร็จแล้วเราอาจจะใส่ชื่อของรูป storage นั้นได้เลยโดยการพิมพ์ชื่อ (ชื่อที่พิมพ์จะแสดงให้เห็นบน Name Window ด้วย)

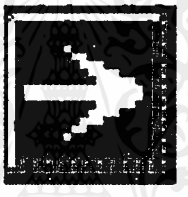
รูปที่ 5.9 แสดงปุ่ม STORAGE DRAWING MODE

TERMINATOR DRAWING MODE การ click mouse ที่ปุ่ม terminator drawing mode จะทำให้อยู่ในโหมดการทำงานนี้ เมื่ออยู่ในโหมดการทำงาน นี้เราสามารถวาดรูป terminator ได้โดยการ click mouse button ทางด้านซ้าย ลงบนตำแหน่งที่ต้องการวาด หลังจากที่วาดเสร็จแล้วเราอาจจะใส่ชื่อของรูป terminator นั้นได้เลยโดยการพิมพ์ชื่อ (ชื่อที่พิมพ์จะแสดงให้เห็นบน Name Window ด้วย)



รูปที่ 5.10 แสดงปุ่ม TERMINATOR DRAWING MODE

FLOWED DATA DRAWING MODE การ click mouse ที่ปุ่ม flowed data drawing mode จะทำให้อยู่ในโหมดการทำงานนี้ เมื่ออยู่ในโหมดการทำงานนี้ การวาดรูปของ flowed data จะแตกต่างจากรูปอื่นๆ คือเมื่อ click mouse แล้ว ครั้งแรกจุดนี้จะเป็นจุดเริ่มต้นของ flowed data เมื่อทำการ move mouse แล้วและ click mouse จุดที่ 2 ที่จุดที่ 2 นี้ถ้าเป็นตำแหน่งที่มี object อื่นอยู่ (process, storage, terminator) การวาดภาพก็จะสิ้นสุดลง แต่ถ้าไม่มีก็จะรอการ move mouse และ click mouse ต่อไป เราสามารถจะสิ้นสุดการวาดภาพที่ตำแหน่ง object อื่นอยู่ได้โดยการ double click mouse หลังจากที่วาดเสร็จแล้วเราอาจจะใส่ชื่อของรูป terminator นั้นได้เลยโดยการพิมพ์ชื่อ (ชื่อที่พิมพ์จะแสดงให้เห็นบน Name Window ด้วย)

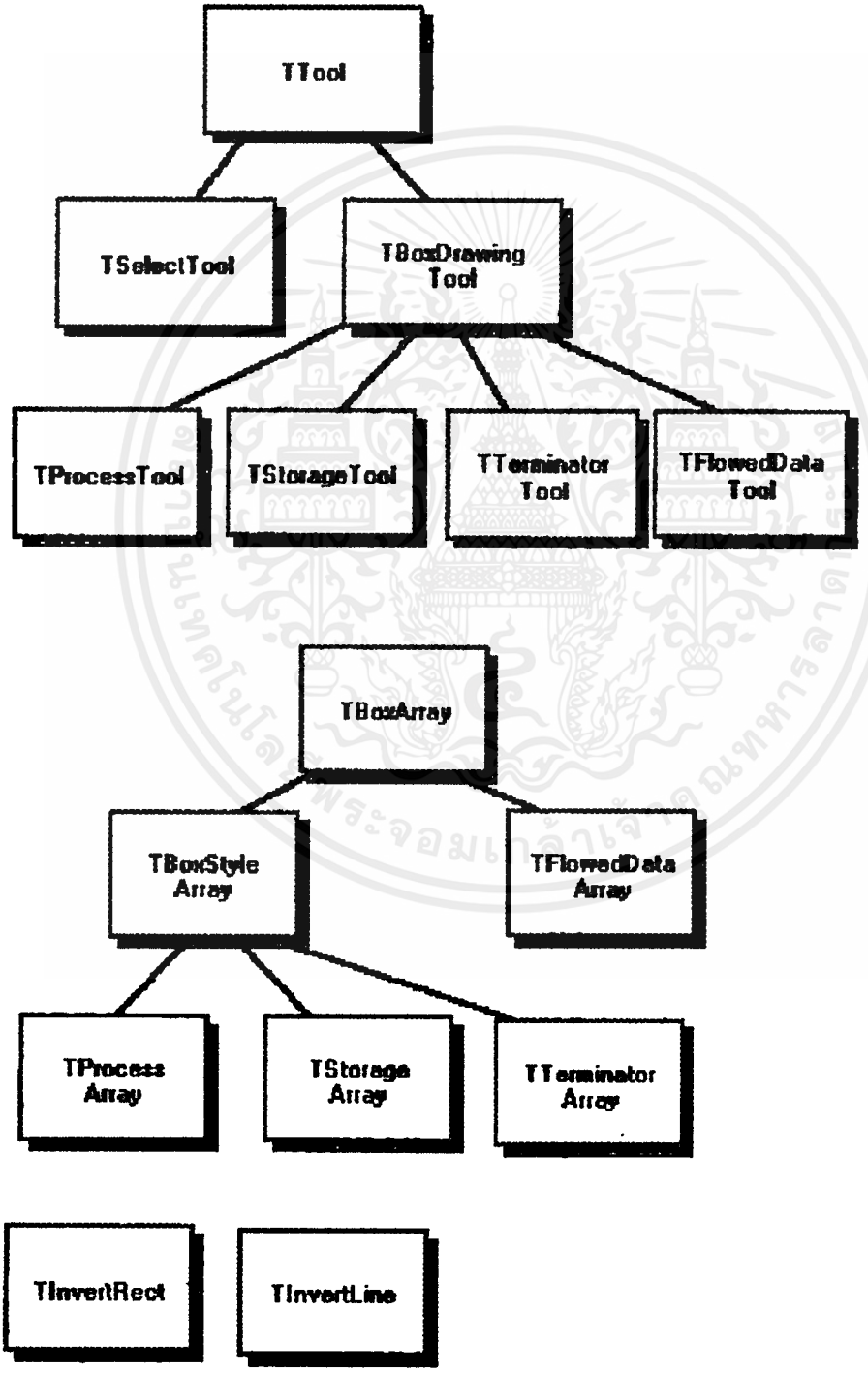


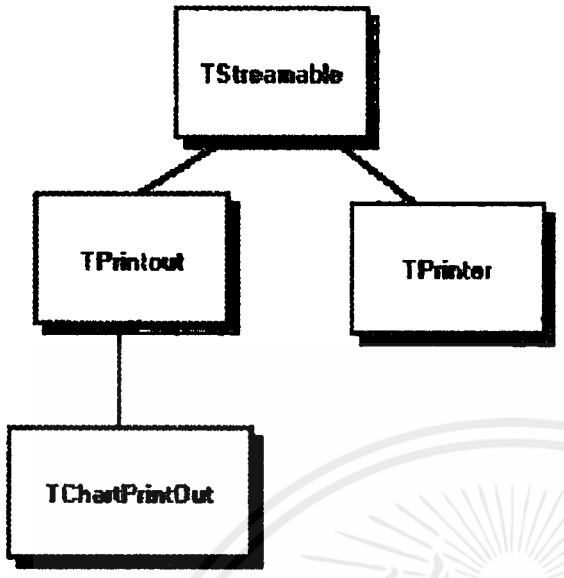
รูปที่ 5.11 แสดงปุ่ม FLOWED DATA DRAWING MODE

บทที่6

Objects Design ของโปรแกรม

สำหรับ objects ที่น่าสนใจทั้งหมดในโปรแกรมแสดงดังรูป





รายละเอียดของแต่ละ Object เป็นดังนี้

/* หมายเหตุ : คำว่า object (ตัวเล็กทั้งหมด) ข้างล่างที่จะเขียนในบทนี้ ให้หมายหมายถึง object ของรูปภาพ เช่น รูป process หนึ่งรูปจะถือเป็น 1 object

ส่วนคำว่า Object (โอตัวใหญ่) ให้หมายถึง Object (class) ที่สร้างขึ้นในโปรแกรมจริงๆ */

Class : TTool

Type : Abstract

Superclasses : none

Subclasses : TSelectTool, TBoxDrawingTool

Description : Classนี้จะเป็นตัว Load Tool Icon และ Load Tool Cursor จากนั้นจะจะเก็บ Tool Icon Handle และ Tool Cursor Handle ไว้เพื่อให้ Objects อื่นเรียกใช้ต่อไป

นอกจากนี้ยังจัดการ Mouse Message และ Key Message

(WM_CHAR, WM_KEYDOWN) ที่ส่งมาจาก Chart Window เพื่อให้ Sub Class ได้ใช้ต่อไป

Data Members :

HCURSOR	Cursor	handle ของ cursor ของ tool นั้น
HICON	Icon	handle ของ icon ของ tool นั้น
int	X, Y	ตำแหน่ง cursor x,y ใน window chart วาดรูป
WORD	wKey	key ตัวอักษร หรือ key board
HWND	Window	handle window ของ chart window ที่ใช้วาดรูป
PTScroller	PScroller	pointer ชี้ไป current scroller (scroller ใน chart window)
PTInformation	PInformation	pointer ชี้ไป TInformation

Member functions :

TTool (LPSTR Name, PTState PState)

Constructure ทำหน้าที่ Load Icon ที่ชื่อ "Name"+"Tool" และ Load Cursor ที่ชื่อ "Name"+"Cursor" เซตค่า member data State

HCURSOR GetCursor ( )

ฟังก์ชันจะ return ค่าของ handle ของ cursor ที่ load มา

HICON GetIcon ( )

ฟังก์ชันจะ return ค่าของ handle ของ icon ที่ load มา

void LMouseDown (HWND Hwnd, int x, int y, PTInformation PInform)

เซตค่า member data X,Y และเรียกฟังก์ชัน LeftDown ( ) ซึ่งฟังก์ชันนี้จะถูกเขียนให้ทำงานจริงใน Class ที่ derived ต่อไป

void MouseMove (int x, int y)

เซตค่า member data X,Y และเรียกฟังก์ชัน MMove ( ) ซึ่งฟังก์ชันนี้จะถูกเขียนให้ทำงานจริงใน Class ที่ derived ต่อไป

void MouseDoubleClick (int x, int y)

เซตค่า member data X,Y และเรียกฟังก์ชัน DoubleClick ( ) ซึ่งฟังก์ชันนี้จะถูกเขียนให้ทำงานจริงใน Class ที่ derived ต่อไป

void Char (WORD wKey)

เซตค่า member data wKey และเรียกฟังก์ชัน KChar ( ) ซึ่งฟังก์ชันนี้จะถูกเขียนให้ทำงานจริงใน Class ที่ derived ต่อไป

void KeyDown (WORD wKey)

เซตค่า member data wKey และเรียกฟังก์ชัน KKeyDown ( ) ซึ่งฟังก์ชันนี้จะถูกเขียนให้ทำงานจริงใน Class ที่ derived ต่อไป

void SetSoroller (PTScroller PScroll)

เซตค่า member data PScroller

virtual void LeftDown ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void LMouseUp ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void RMouseDown ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void MMove ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void DoubleClick ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void KKeyDown ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void KChar ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

Class : TSelectTool

Type : Concrete

Superclasses : TTool

Subclasses : none

Description : Classนี้จะจัดการเกี่ยวกับการเปลี่ยนแปลงแก้ไขข้อมูลใน instances ของ Class TProcessArray, TStorageArray, TTerminateArray ซึ่งกิจกรรมที่ทำให้เกิดการเปลี่ยนแปลงแก้ไขข้อมูลใน instances เหล่านั้นได้แก่

การ Move object (รูป และ ข้อมูล)

การ Delete object

การแก้ไขชื่อของ object

การ Link กับ Sub data flow diagram ของ instance ของ Class TProcessArray

Data Members :

BOOL	fDrag	flag check ว่ามีการ drag หรือไม่
BOOL	bCapture	flag check ว่าขณะนั้น mouse ถูก capture อยู่หรือไม่
BOOL	bWrite	flag check ว่ามี object ที่สามารถรับ key (จาก keyboard) หรือไม่
int	X1, Y1, X2, Y2	ตำแหน่ง x1, y1, x2, y2 ของ object รูป box หรือใช้เพื่อทำการ invalidate ตำแหน่งที่ต้องการวาดรูปใหม่ ใน chart window
PTBoxArray	PCurrentArray	pointer ชี้ไป object array ที่ถูกเลือก

int iSelect index ชี้ object ที่ถูกเลือกใน Current Array ขณะนั้น

RECT rScratch ตำแหน่งสี่เหลี่ยมของ object ที่กำลัง move

Member functions :

TSelectTool (LPSTR Name, PTState State)

constructer

void LeftDown ( )

function จะ check ว่าที่ตำแหน่ง x, y ที่ mouse down นั้น มี object หรือไม่ ถ้ามี object นั้นจะถูก set ให้เป็น object ที่ถูกเลือก

void MMove ( )

function จะ check ว่าขณะนั้นมี object ที่ถูกเลือกหรือไม่ และ mouse ถูก capture อยู่หรือไม่ ถ้าถูกต้องทั้ง 2 กรณีจะทำการ drag รูป object

void LMouseUp ( )

ทำการ set flag ต่างๆ เกี่ยวกับการ drag และการ capture mouse คืน

void RMouseDown ( )

จะทำการ delete object ออกจาก current array object ถ้าขณะนั้นมี object ที่ถูกเลือกอยู่

void KChar ( )

จะทำการเปลี่ยนแปลงชื่อ object ถ้ามี object ที่ถูกเลือกอยู่

void KKeyDown ( )

จะทำการเปลี่ยนแปลงชื่อ object ถ้ามี object ที่ถูกเลือกอยู่

void DoubleClick ( )

จะทำการ link sub data flow diagram ถ้ามี object ที่ถูกเลือกอยู่ และ object ที่ถูกเลือกนั้นเป็น object ใน PProcessArray

Class : TBoxDrawingTool

Type : Abstract

Superclasses : TTool

Subclasses : TProcessTool, TStorageTool, TTerminatorTool

Description : Classนี้จัดการเกี่ยวกับการสร้างข้อมูลให้กับ instance ของ Class ที่เป็น Sub class ของ TBoxArray (ซึ่งจะเป็น instance ของ Sub class ใดๆของ TBoxArray จะถูกระบุไว้ใน Sub class ของ Class นี้ (TBoxDrawingTool) อีกทีหนึ่ง)

นอกจากนี้ยังทำการการ Set ขนาด default ของ Box ที่จะวาดในรูปหนึ่งๆอีกด้วย

Data Members :

int iSelect index ชี้ object ที่ถูกเลือกใน array ขณะนั้น

int X1, Y1, X2, Y2 ตำแหน่ง box ของ object ที่ถูกเลือก หรือสร้างขึ้น

BOOL bWrite flag check ว่ามี object ที่ถูกเลือกเพื่อจะเขียนหรือเปลี่ยนแปลงชื่อหรือไม่

HDC DC handle Decvice Context ของ chart window ที่ใช้วาดภาพ

Member functions :

TBoxDrawingTool (LPSTR Name, PTState State)

Constructor

void LeftDown ( )

เรียกฟังก์ชัน SetBoxSize ( ) เพื่อทำการ set ขนาด box ของรูป จากนั้นก็เรียกฟังก์ชัน DrawObject ( ) เพื่อ update ข้อมูลใน Object array และ invalidate window เพื่อ show รูป

void KChar ( )

จะทำการเปลี่ยนแปลงชื่อ object ถ้ามี object ที่ถูกเลือกอยู่

void KKeyDown ( )

จะทำการเปลี่ยนแปลงชื่อ object ถ้ามี object ที่ถูกเลือกอยู่

virtual void SetBoxSize ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

virtual void DrawObject ( )

เป็น virtual function ซึ่งใน class นี้ไม่ได้ทำอะไรแต่จะถูกเขียนจริงใน class ที่ derived ต่อไป

Class : TProcessTool

Type : Concrete

Superclasses : TBoxDrawingTool

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการสร้างข้อมูลให้กับ instance ของ Class TProcessArray

Data Members :

none

Member functions :

TProcessTool (LPSTR Name, PTState State)

Constructor

void SetBoxSize ( )

ฟังก์ชันทำหน้าที่ set ขนาดของรูป box object

void DrawObject ( )

ฟังก์ชันทำหน้าที่ append ข้อมูลใหม่ เข้าไปใน ProcessArray class (instance) และทำการ invalidate chart window เพื่อที่จะแสดงผลรูปที่วาดขึ้น

Class : TStorageTool

Type : Concrete

Superclasses : TBoxDrawingTool

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการสร้างข้อมูลให้กับ instance ของ Class TStorageArray

Data Members :

none

Member functions :

TStorageTool (LPSTR Name, PTState State)

Constructor

void SetBoxSize ( )

ฟังก์ชันทำหน้าที่ set ขนาดของรูป box object

void DrawObject ( )

ฟังก์ชันทำหน้าที่ append ข้อมูลใหม่ เข้าไปใน StorageArray class (instance) และทำการ invalidate chart window เพื่อที่จะแสดงผลรูปที่วาดขึ้น

Class : TTerminatorTool

Type : Concrete

Superclasses : TBoxDrawingTool

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการสร้างข้อมูลให้กับ instance ของ Class TTerminatorArray

Data Members :

none

Member functions :

TTerminatorTool (LPSTR Name, PTState State)

Constructor

void SetBoxSize ( )

ฟังก์ชันทำหน้าที่ set ขนาดของรูป box object

void DrawObject ( )

ฟังก์ชันทำหน้าที่ append ข้อมูลใหม่ เข้าไปใน TerminatorArray class (instance) และทำการ invalidate chart window เพื่อที่จะแสดงผลรูปที่วาดขึ้น

Class : TFlowDataTool

Type : Concrete

Superclasses : TBoxDrawingTool

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการสร้างข้อมูลให้กับ instance ของ Class TFlowDataArray

Data Members :

BOOL bFirst flag check ว่าเป็นการ click mouse เริ่มต้นหรือไม่ (check ว่าเป็นจุดเริ่มต้นหรือไม่)

int iFirstType flag check ว่า มี Object class อะไรติดกับ ที่ point แรก

int iLastType flag check ว่า มี Object class อะไรติดกับที่ point สุดท้าย

int iFirst index ชี้ object ที่ติดกับ point แรก (ถ้ามี object ที่ติด)

int iLast index ชี้ object ที่ติดกับ point สุดท้าย (ถ้ามี object ที่ติด)

Member functions :

TFlowDataTool (LPSTR Name, PTState State)

Constructor

void LeftDown ( )

คอย update ข้อมูลใน FlowDataArray Object คอย check ว่าเป็นจุดแรกหรือสุดท้ายหรือไม่ เรียกฟังก์ชัน CheckInBox ( ) เพื่อคอยตรวจสอบว่าแต่ละจุดที่ mouse click ลงไปมี box object ที่ไม่ใช่ FlowData อยู่หรือไม่

void DoubleClick ( )

update ข้อมูลใน FlowDataArray Object ถือเป็นจุดสุดท้าย

void MMove ( )

ถ้าอยู่ในระหว่างการวาดรูปจะทำการเรียก Object InvertLine เพื่อแสดงให้เห็นการ move เส้นไปมา

BOOL CheckInBox ( );

ฟังก์ชัน check ที่ตำแหน่ง mouse click X,Y มี object ที่ไม่ใช่ FlowData อยู่หรือไม่

Class : TBoxArray

Type : Abstract

Superclasses : none

Subclasses : TBoxStyleArray, TFlowedDataArray

Description : Classนี้จัดการเกี่ยวกับการ update ข้อมูลที่เป็น box แสดงถึง (check) การอยู่หรือไม่อยู่ของ Object (รูป) ที่ตำแหน่งใดๆ ในพื้นที่วาดภาพ

Data Members :

RECTDATA rdValues[MAXOBJECTS] array ของข้อมูล box object

int cObjects count นับจำนวน objects

int iSelect index ชี้ object ที่ถูกเลือกในขณะนั้น

int iDrag index ชี้ object ที่ถูก drag ในขณะนั้น

int cCountNumber count Number ของ objects

PTState State pointer ชี้ไปที่ struct TState

Member functions :

TBoxArray (PTState AState)

constructer

int append (int x1, int y1, int x2, int y2)

append ข้อมูล box object เข้าไปใน array ฟังก์ชัน return ค่าเป็นหมายเลข index ของ object ที่ append นั้น ถ้าเกิดการ fail (ข้อมูลล้น array) จะ return -1

void deleteObject (int index)

ทำการลบข้อมูลตัวที่ index

int GetCount ( )

return จำนวน objects ใน array

int getX1 (int index)

return ค่าตำแหน่ง X1 ของรูป box object ตัวที่ index

int getY1 (int index)

return ค่าตำแหน่ง Y1 ของรูป box object ตัวที่ index

int getX2 (int index)

return ค่าตำแหน่ง X2 ของรูป box object ตัวที่ index

int getY2 (int index)

return ค่าตำแหน่ง y2 ของรูป box object ตัวที่ index

int getcName (int index)

return ความยาวของชื่อของ object ตัวที่ index

int getcFlowIn (int index)

return จำนวน Flow Data ที่ไหลเข้า box object

int getcFlowOut (int index)

return จำนวน Flow Data ที่ไหลออกจาก box object

int hittest (int x, int y)

return index ของ object ที่มีพื้นที่ครอบคลุมตำแหน่ง X, Y

int UpdatePos (int indez, int x1, int y1, int x2, int y2)

update ตำแหน่ง opject ตัวที่ index

int getSelect ( )

return index ของ object ที่ถูกเลือกอยู่ (iSelect)

void setSelect (int index)

set iSelect (object ที่ถูกเลือก) ให้เท่ากับ index

int getDrag ( )

return index ของ object ที่ถูก drag อยู่ (iDrag)

void setDrag (int index)

set iDrag (object ที่ถูก drag) ให้เท่ากับ index

int UpdateName (int index, LPSTR achName, int countName)

update ชื่อของ object ตัวที่ index โดย achName เป็น pointer ที่ชี้ไปยัง array ของชื่อที่ต้องการ update และ countName คือจำนวนตัวอักษรของชื่อ

void SetFlowIn (int index, int iFlowIn)

set ค่า Flow Data ที่ไหลเข้า object ตัวที่ index โดยที่ iFlowIn คือ index ชี้ Flow Data object ที่ไหลเข้า

void SetFlowOut (int index, int iFlowOut)

set ค่า Flow Data ที่ไหลออกจาก object ตัวที่ index โดยที่ iFlowOut คือ index ชี้ Flow Data object ที่ไหลออก

int GetFlowInCount (int index)

return จำนวน flow data ที่ไหลเข้า object ตัวที่ index

int GetFlowOutCount (int index)

return จำนวน flow data ที่ไหลออกจาก object ตัวที่ index

int GetFlowIn (int index, int number)

return หมายเลข (index) ของ flow data ที่ไหลเข้า object ตัวที่ index ลำดับที่ number

int GetFlowOut (int index, int number)

return หมายเลข (index) ของ flow data ที่ไหลออกจาก object ตัวที่ index ลำดับที่ number

Class : TFlowedDataArray

Type : concrete

Superclasses : TBoxArray

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการ update ข้อมูลที่เกี่ยวกับ objects ที่ flowed data ไหลเข้า และ ข้อมูลที่เกี่ยวกับ objects ที่ flowed data ไหลออก

การเขียนภาพ flowed data (พร้อมชื่อ)

Data Members :

LINEDATA kdData[MAXOBJECTS] array ข้อมูล object ที่มีลักษณะเป็นเส้น

Member functions :

TFlowDataArray (PTState AState)

constructure

int StartPoint (int x, int y)

ทำการ append ข้อมูล point (start point) ให้กับ kdData array

void NextPoint (int x, int y)

ทำการ update ข้อมูล point (next point) ให้กับ object ตัวสุดท้ายที่ได้ถูก append ไปแล้ว

void EndPoint (int x , int y)

ทำการ update ข้อมูล point (end point) ให้กับ object ตัวสุดท้ายที่ได้ถูก append ไปแล้ว และเป็นการสิ้นสุดการ update ข้อมูล point ของ object นั้น และทำการ append ข้อมูล box data ใน rdValues array ด้วย

void EndPointDoubleClick ( )

สิ้นสุดการ update ข้อมูล point โดยการ double click mouse และทำการ append ข้อมูล box data ใน rdValues array ด้วย

void SetBoxName (int index, int x, int y)

set ค่าตำแหน่งของ name box ของรูป flow data object ตัวที่ index

int CountPoints (int index)

return ค่าจำนวน point ของ flow data object ตัวที่ index

int FirstX (int index)

return ตำแหน่ง X ของจุดแรก (first point) ของ object ตัวที่ index

int FirstY (int index)

return ตำแหน่ง Y ของจุดแรก (first point) ของ object ตัวที่ index

int LastX (int index)

return ตำแหน่ง X ของจุดสุดท้าย (last point) ของ object ตัวที่ index

int LastY (int index)

return ตำแหน่ง Y ของจุดสุดท้าย (last point) ของ object ตัวที่ index

int AnyX (int index, int indexPt)

return ตำแน่ง X ลำดับที่ indexPt ของ object ตัวที่ index

int AnyY (int index, int indexPt)

return ตำแน่ง Y ลำดับที่ indexPt ของ object ตัวที่ index

void SetInType (int type, int typenumber)

set ชนิดของ object ที่ flow data (ที่ถูกเลือก select ในขณะนั้น) ไหลเข้าเป็น type และมี index ใน array ของ class type นั้น เป็น typenumber

type = 0 nothing

type = 1 Process

type = 2 Storage

type = 3 Terminator

void SetOutType (int type, int typenumber)

set ชนิดของ object ที่ flow data (ที่ถูกเลือก select ในขณะนั้น) ไหลออกเป็น type และมี index ใน array ของ class type นั้น เป็น typenumber

type = 0 nothing

type = 1 Process

type = 2 Storage

type = 3 Terminator

int GetInType (int index)

return ชนิดของ object ที่ flow data ตัวที่ index ไหลเข้า

int GetOutType (int index)

return ชนิดของ object ที่ flow data ตัวที่ index ไหลออก

int GetInTypeNumber (int index)

return (index) ตำแหน่งที่ของ object ใน array ของ object นั้น ที่ flow data ไหลเข้า

int GetOutTypeNumber (int index)

return (index) ตำแหน่งที่ของ object ใน array ของ object นั้น ที่ flow data ไหลออก

void SetFirst (int index, int x, int y)

ทำการเปลี่ยนแปลงตำแหน่งจุดแรก ของ flow data ตัวที่ index

void SetLasr (int index, int x, int y)

ทำการเปลี่ยนแปลงตำแหน่งจุดสุดท้าย ของ flow data ตัวที่ index

void deleteObject (int index)

ทำการลบข้อมูล flow data ตัวที่ index

int GetMaxX (int index)

return ค่าตำแหน่ง X ของจุดที่อยู่ล่างสุด (ค่า X มีค่ามากสุด) ของ flow data ตัวที่ index

int GetMaxY (int index)

return ค่าตำแหน่ง Y ของจุดที่อยู่ล่างสุด (ค่า Y มีค่ามากสุด) ของ flow data ตัวที่ index

int GetMinX (int index)

return ค่าตำแหน่ง X ของจุดที่อยู่บนสุด (ค่า X มีค่าน้อยสุด) ของ flow data ตัวที่ index

int GetMinY (int index)

return ค่าตำแหน่ง Y ของจุดที่อยู่บนสุด (ค่า X มีค่าน้อยสุด) ของ flow data ตัวที่ index

void FindMaxMin (int index)

ทำการ search หาค่าจุดที่อยู่ต่ำสุด และจุดที่อยู่สูงสุดใหม่

void draw (HDC hdc, int index)

ทำการวาดภาพ flow data ตัวที่ index

void draw2 (HDC hdc, int x1, int y1, int x2, int y2)

ทำการวาดภาพ box name ของ flow data ที่ใช้สำหรับการ move

Class : TBoxStyleArray

Type : Abstract

Superclasses : TBoxArray

Subclasses : TProcessArray, TStorageArray, TFlowedDataArray

Description : Classนี้จัดการเกี่ยวกับการ update ข้อมูลที่เกี่ยวกับ flowed data ที่ไหลเข้า หรือไหลออกจาก box object

Data Members :

none

Member functions :

TBoxStyleArray (PTState AState)

constructor

void DeleteLine (int iDeleteNumber)

delete ข้อมูล index ของ flow data ที่มี index หมายเลข iDeleteNumber ทั้งที่เป็น flow in และ flow out ของ box objects ทุกๆตัว

Class : TProcessArray

Type : concrete

Superclasses : TBoxStyleArray

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการ update ข้อมูลที่เกี่ยวกับการ link ไป sub data flowed diagram file

การเขียนรูป process (พร้อมหมายเลขและชื่อ)

Data Members :

PROCESSLINK plLinkFile[MAXOBJECTS] array ข้อมูลเกี่ยวกับการ link file ของ process object

Member functions :

TProcessArray (PTState AState)

constructor

int append (int x1, int y1, int x2, int y2)

append ข้อมูล process object เข้าไปใน array ฟังก์ชัน return ค่าเป็นหมายเลข index ของ object ที่ append นั้น ถ้าเกิดการ fail (ข้อมูลต้น array) จะ return -1

void deleteObject (int index)

ทำการลบข้อมูลตัวที่ index

BOOL IsLink (int index)

return TRUE ถ้า process ตัวที่ index มีการ link กับ file ใดอยู่ และ return FAULT ถ้าไม่ได้ link กับ file ใด

void SetLinkFile (int index, LPSTR FileName)

Set link file ของ process ตัวที่ index โดยที่ FileName เป็น pointer ชี้ไปที่ชื่อ file

void CutLink (int index)

ทำการลบ link file ของ process ตัวที่ index

void draw (HDC hdc, int index)

ทำการวาดภาพ process ตัวที่ index

void draw2 (HDC hdc, int x1, int y1, int x2, int y2)

ทำการวาดภาพ process ที่ใช้สำหรับการ move

Class : TStorageArray

Type : concrete

Superclasses : TBoxStyleArray

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการเขียนรูป storage (พร้อมชื่อ)

Data Members :

none

Member functions :

TStorageArray (PTState AState)

constructor

void draw (HDC hdc, int index)

ทำการวาดภาพ storage ตัวที่ index

void draw2 (HDC hdc, int x1, int y1, int x2, int y2)

ทำการวาดภาพ storage ที่ใช้สำหรับการ move

Class : TTerminatorArray

Type : concrete

Superclasses : TBoxStyleArray

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการเขียนรูป Terminator (พร้อมชื่อ)

Data Members :

none

Member functions :

TTerminatorArray (PTState AState)

constructor

void draw (HDC hdc, int index)

ทำการวาดภาพ terminator ตัวที่ index

void draw2 (HDC hdc, int x1, int y1, int x2, int y2)

ทำการวาดภาพ terminator ที่ใช้สำหรับการ move

Class : TInvertRect

Type : concrete

Superclasses : none

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการเขียนรูปที่มีลักษณะเป็น box ในขณะทำการ move รูป โดยใช้ raster operation mode ของ Windows ในการวาด

Data Members :

PTBoxArray	PCurrentArray	pointer ชี้ไป box array Object (class instance) ที่ต้องการ move
PTScroller	PScroller	pointer ชี้ไป scroller ของ chart window ที่ใช้ในการวาดภาพ
RECT	rLocation	ตำแหน่งพื้นที่สี่เหลี่ยมของ box object ขณะ move
HANDLE	hbrNull	handle Null pen ที่ใช้ในการวาดภาพขณะ move

Member functions :

TInvertRect (PTBoxArray PCurrentArray, PTScroller PScroller,

int x1, int y1, int x2, int y2)

constructor โดยที่ x1, y1, x2, y2 เป็นตำแหน่งจุดพื้นที่สี่เหลี่ยมเริ่มต้นของ object ที่ต้องการ move

void Invert (HDC hdc)

ทำการ invert ภาพที่ตำแหน่ง rLocation (จากมีภาพ object เป็นไม่มี หรือจากไม่มี ไม่เขียนออก เป็นมี เขียนออกแสดงให้เห็น)

void Move (HDC hdc, int x1, int y1, int x2, int y2)

ทำการลบรูปที่ตำแหน่ง rLocation เดิม (invert) และทำการวาดภาพที่ตำแหน่งใหม่

Class : TInvertLine

Type : concrete

Superclasses : none

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการเขียนรูปของเส้นที่มีการเคลื่อนที่ไปมา โดยใช้ raster operation เช่นกัน

Data Members :

PTScroller PScroller pointer ชี้ไป scroller ของ chart window ที่ใช้ในการวาดภาพ

int x1, y1, x2, y2 ตำแหน่งจุดเริ่มต้นและจุดสุดท้ายของ line ที่ต้องการแสดงภาพ move

HANDLE hbrNull handle Null pen ที่ใช้ในการวาดภาพขณะ move

Member functions :

TInvertLine (PTScroller PScroller, int x1, int y1, int x2, int y2)

constructor โดยที่ x1, y1 เป็นตำแหน่งจุดเริ่มต้น และ x2, y2 เป็นตำแหน่งจุดสุดท้ายของ line ที่ต้องการแสดงภาพ move

void Invert (HDC hdc)

ทำการ invert ภาพ line ที่ x1, y1, x2, y2 (จากมีภาพ object เป็นไม่มี หรือจากไม่มี ไม่เขียนออก เป็นมี เขียนออกแสดงให้เห็น)

void Move (HDC hdc, int x, int y)

ทำการลบรูป line ที่ตำแหน่งเดิม (invert) และทำการวาดภาพที่ตำแหน่งใหม่ (ตำแหน่ง x1,y1 เดิม x2,y1 ใหม่)

void SetNewPoint (int x, int y)

set ตำแหน่ง x1, y1 ใหม่

Class : TPrintout

Type : abstract

Superclasses : TStreamable

Subclasses : TChartPrintOut

Description : Classนี้จัดการเกี่ยวกับการพิมพ์ เป็น friend กับ TPrinter

Data Members :

Pchar	TiTle;	pointer ที่ชี้ไปยังชื่อหัวข้อ ของการพิมพ์
BOOL	Banding;	flag สำหรับจะพิมพ์ Text or graphic
BOOL	ForceAllBands	flag สำหรับจะพิมพ์ ทั้ง text และ graphic

Member functions :

TPrintout (Pchar ATitle)

โดยที่ ATitle เป็น pointer ที่ชี้ไปยังชื่อหัวข้อของการพิมพ์

~TPrintout()

ทำหน้าที่ deallocate TiTle

void PrintPage(HDC DC, WORD Page, POINT Size, LPRECT Rect, WORD flag)

ทำหน้าที่ print สิ่งที่ต้องการพิมพ์ออกทาง printer

BOOL IsNextPage()

ทำหน้าที่ตรวจว่ามี page ต่อไปต้องพิมพ์หรือไม่

Class : TChartPrintOut

Type : concrete

Superclasses : TPrintout

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับสิ่งที่ต้องการพิมพ์ออกทาง printer (ภาพที่วาด)

Data Members : none

Member functions :

TChartPrintOut(Pchar ATitle)

contructor ทำหน้าที่ส่งหัวข้อที่จะทำการพิมพ์ให้แก่ TPrintout

void PrintPage(HDC DC, WORD Page, POINT Size, LPRECT Rect, WORD flag)

ทำการพิมพ์สิ่งที่ต้องการพิมพ์ออก printer ด้วย handle ของ printer ที่ได้มา

void SetBanding(BOOL b)

ทำหน้าที่ set ค่า flag ในการพิมพ์ว่าเป็น Text หรือ graphic

Class : TPrinter

Type : concrete

Superclasses : TStreamable

Subclasses : none

Description : Classนี้จัดการเกี่ยวกับการ get ค่าของ printer โดยการ get ค่าจาก WIN.INI และ นำมาหา handle ของ printer เพื่อที่จะใช้ในการพิมพ์ ของ TPrintout

Data Members :

Pchar Device, Driver, Port; รายละเอียดเกี่ยวกับ printer device

int Status; สถานะของ printer ถ้าเกิดข้อผิดพลาดจะไม่เท่ากับ PS_OK

int Error; จะมีค่าน้อยกว่า 0 ถ้าเป็นการเกิด error ระหว่างการพิมพ์

HINSTANCE DeviceModule; แฮนเดิลสำหรับ printer device module

PTDeviceModeFon DeviceMode; pointer ที่ชี้ไปยัง function DevMode

LPFNDEVMODE ExtDeviceMode; pointer ที่ชี้ไปยัง function ExtDevMode

PDEVMODE DevSettings; Local copy ของ printer settings

int DevSettingSize; ขนาดของ printer settings

Member functions :

TPrinter()

constructor ทำหน้าที่ initial TPrinter Object ให้ใช้ได้กับ default printer

~TPrinter()

destructor ทำหน้าที่ de-allocate resource โดยผ่าน cleardevice()

void ClearDevice()

ทำหน้าที่ clear object ที่เกี่ยวข้องกับ device นี้

void SetDevice()

ทำหน้าที่ set ค่า device ให้กับ printer object โดยผ่าน GetDefaultPrinter()

void GetDefaultPrinter()

ทำหน้าที่ get ค่า default printer จาก WIN.INI

void Configure()

ทำหน้าที่ นำ dialog ซึ่งเป็น child window มา configure กับ printer driver

void GetDC()

ทำหน้าที่สร้าง Device Context ของ printer จากรายละเอียดของ printer ที่มีอยู่

void CalcBandingFlags()

ทำหน้าที่คำนวณหา banding flag ว่าตอนนี้จะพิมพ์ text ,graphic หรือทั้งสองอย่าง

void Print(PTWindowsObject ParentWin, PTPrintout Printout)

ทำหน้าที่จัดการ physical printing สิ่งที่ Printout จะพิมพ์ ,สร้างและจัดการ Abort Dialog ระหว่างการพิมพ์

Limitation

มีข้อจำกัดต่าง ๆ ดังนี้คือ

1.-ขนาดรูปยังไม่สามารถเปลี่ยนแปลงได้

2. จำนวนรูปในแต่ละ object มีขนาดจำกัด คือไม่เกินที่กำหนดไว้คือ 50 ตัว

3. ไม่สามารถใช้การ input/output ข้อมูลกับ clipboard ได้

4. ไม่สามารถ zoom in/out ได้

Expansion

สิ่งที่จะสามารถขยายต่อไปในอนาคต มีดังนี้คือ

1.ให้สามารถเปลี่ยนขนาดรูปได้

2.ให้สามารถใช้การ input/output ข้อมูลกับ clipboard ได้

3.ให้สามารถ zoom in/out ได้

ภาคผนวก

KMIT'L data flow diagram Mannual

<u>การใช้ KMIT 'L DFD มีรายละเอียดดังนี้คือ</u>

การสร้างชุดของแผนภาพใหม่

การเรียกชุดของแผนภาพที่เก็บไว้ในไฟล์

การเก็บแผนภาพ

การเปลี่ยนชื่อแผนภาพ

การปิดชุดแผนภาพเพื่อเปลี่ยนชุดแผนภาพใหม่

การพิมพ์

การเตรียมเครื่องพิมพ์

การออกจากโปรแกรม

การเปลี่ยนสี Process

การเปลี่ยนสี Terminator

การเปลี่ยนสี Storage

การเปลี่ยนสีทุก object ให้กลับไปสู่ค่าปกติ

การเปลี่ยนสีทุก object ให้เป็นสีดำ

การเปลี่ยนระดับของ DFD หรือการเชื่อมสู่ DFD ระดับล่าง ๆ

การตัดการเชื่อมสู่ DFD ระดับล่าง

การเปลี่ยนฟอนต์และสีของฟอนต์

การจัดให้ DFD แต่ละรูปมีขนาดเท่า ๆ กันเฉลี่ยเต็ม Main Window

การจัดให้ DFD แต่ละรูปซ้อน ๆ กันเป็นแผ่นอย่างมีระเบียบ

การทำให้ DFD แต่ละรูปเป็น icon

การวาดรูป object ลงบนแผนภาพ

Process

Storage

Terminator

Flow Data

การเปลี่ยนแปลงชื่อหรือให้ชื่อแก่ object

Process

Storage

Terminator

Flow Data

หมายเหตุ

การย้ายตำแหน่ง object

Process

Storage

Terminator

การลบ object จากแผนภาพ

Process

Storage

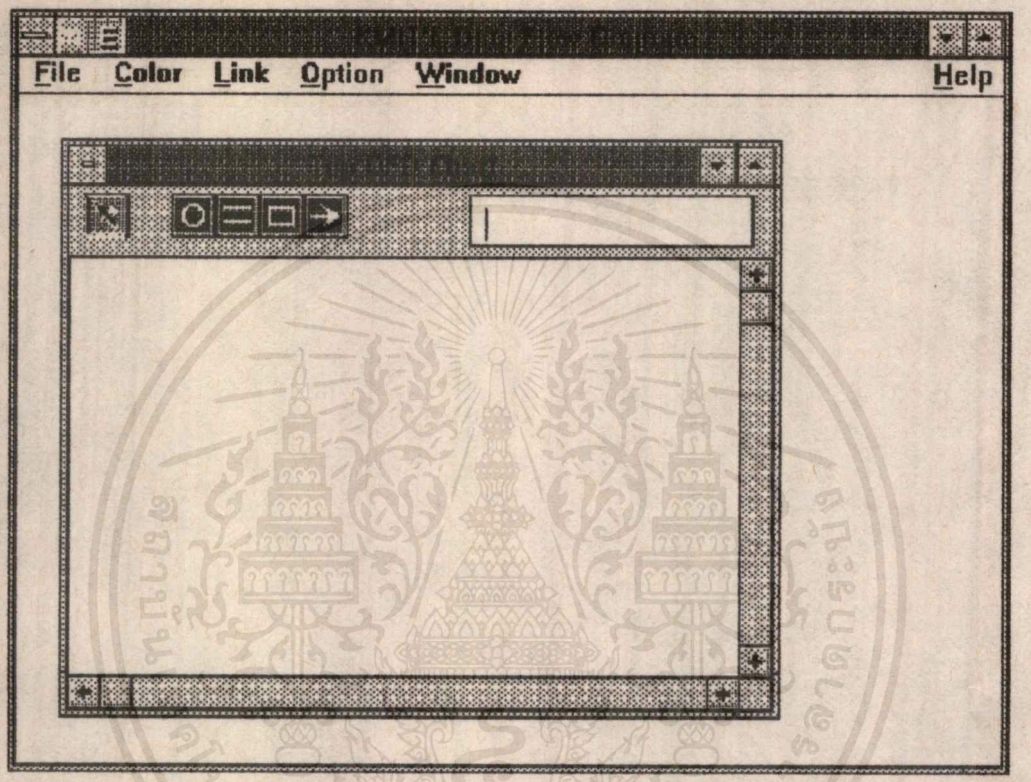
Terminator

Flow Data

การเลื่อนแผนภาพบนหน้าจอขึ้นหรือลง

การสร้างชุดของแผนภาพใหม่

ทำได้โดยการเลือกเมนู File จากนั้นเลือกเมนู New Project ซึ่งจะทำให้มี Top level DFD ขึ้นมาบนจอดังนี้คือ

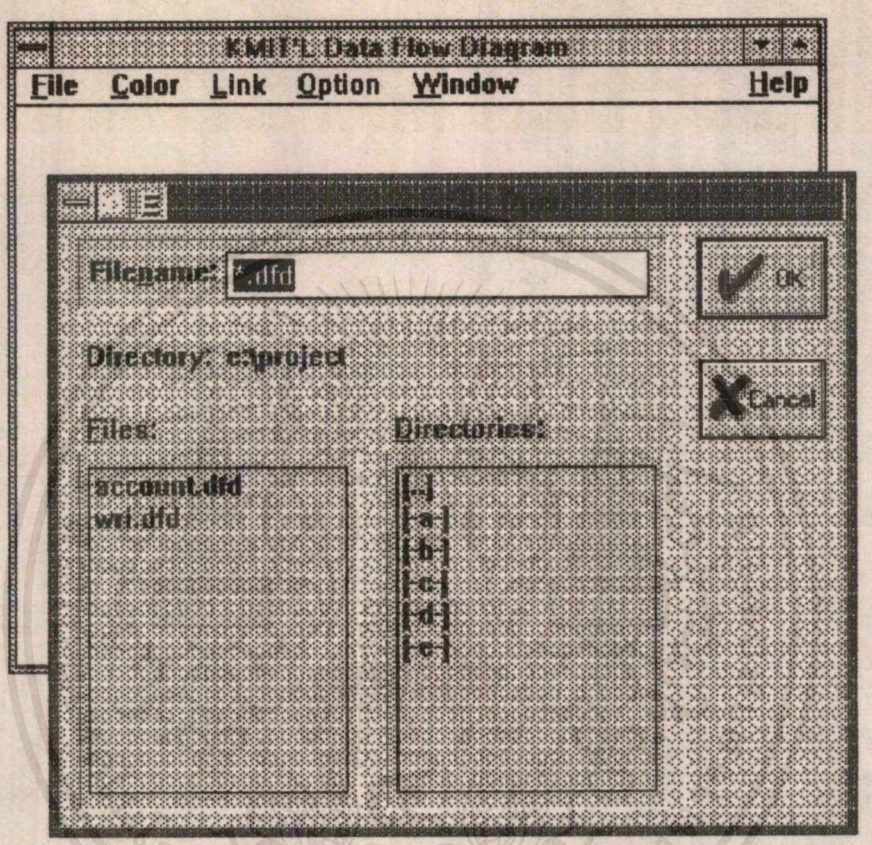


รูปแสดง window ที่เปิดขึ้นใหม่หลังจากเลือก new project แล้ว

จากนั้นก็ทำการสร้างแผนภาพได้เลย

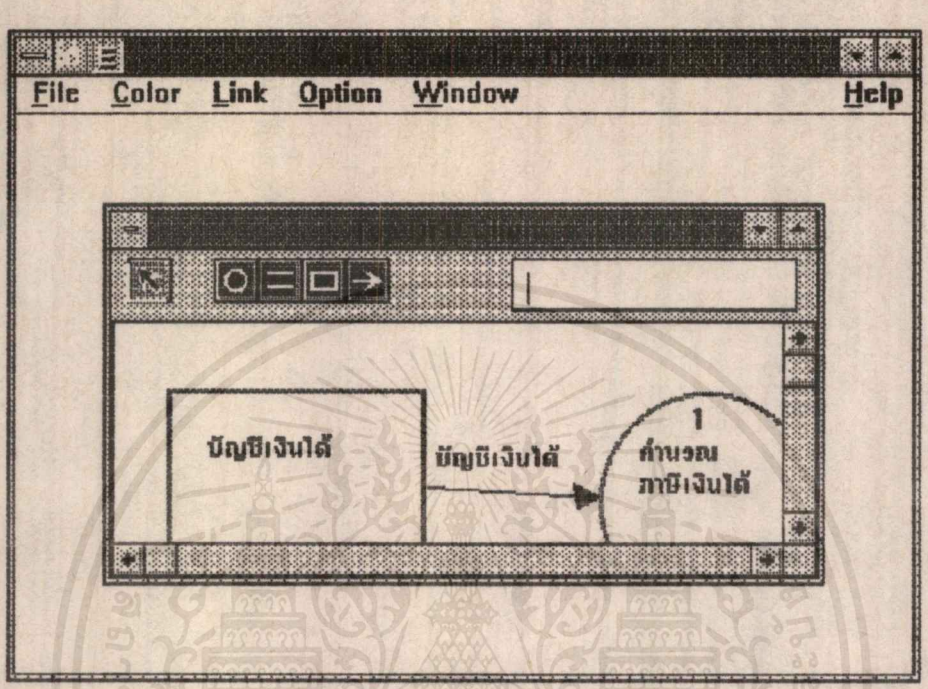
การเรียกชุดของแผนภาพที่เก็บไว้ในไฟล์มาประมวลผล

ทำได้โดยการเลือกเมนู File จากนั้นเลือกเมนู Open Project ซึ่งจะทำให้มี File Dialog box ขึ้นมาเพื่อถามชื่อไฟล์ บนจอดังนี้คือ



รูปแสดง file dialog ในกรณีต้องการเปิด project file ที่สร้างไว้แล้ว

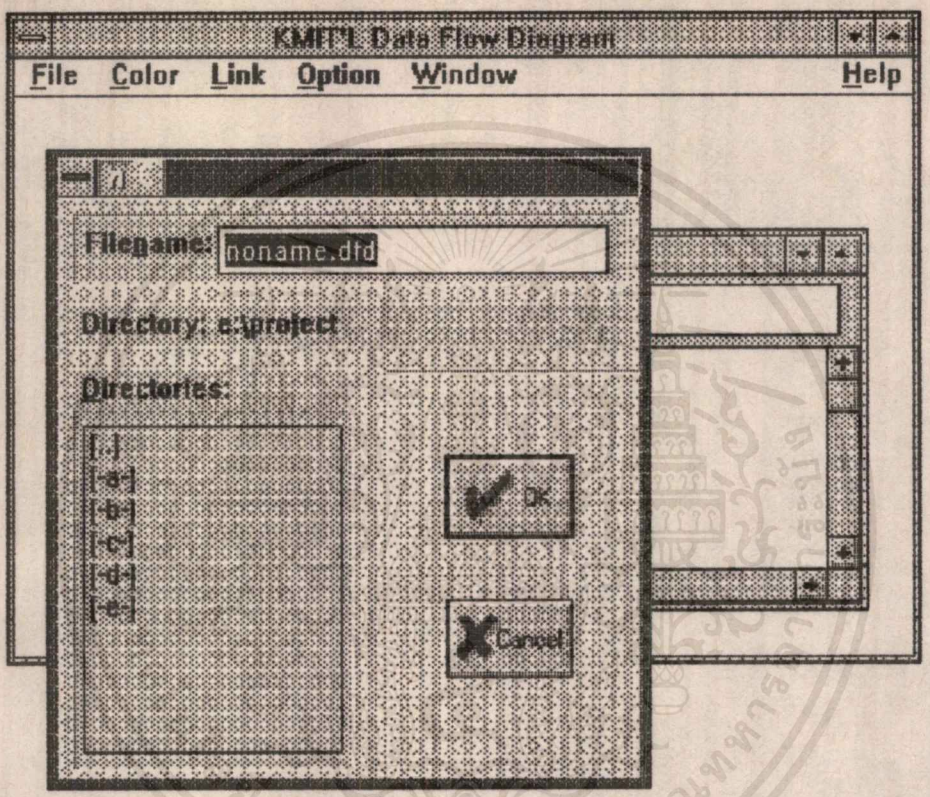
ซึ่งเมื่อใส่ชื่อไฟล์แล้วกด Enter key หรือ click mouse ที่ปุ่ม OK รูปด้านล่างแสดงรูปหลังจากเปิด file WRI.DFD แล้ว



รูปแสดงตัวอย่างหลังจากเปิด project file WRI.DFD

การเก็บแผนภาพ

ทำได้โดยการเลือกเมนู File จากนั้นเลือกเมนู Save file ถ้าหากมีการให้ชื่อไฟล์ไว้แล้ว จะทำให้โปรแกรม เก็บ DFD ลงไฟล์นั้นทันที แต่หากยังไม่มีการให้ชื่อไฟล์ ก็จะมี File Dialog box ขึ้นมาเพื่อถามชื่อไฟล์ ซึ่งมี default ชื่อ file เป็น NONAME.DFD (ซึ่งเราสามารถพิมพ์ชื่อใหม่ได้เลย) บนจอดังนี้คือ

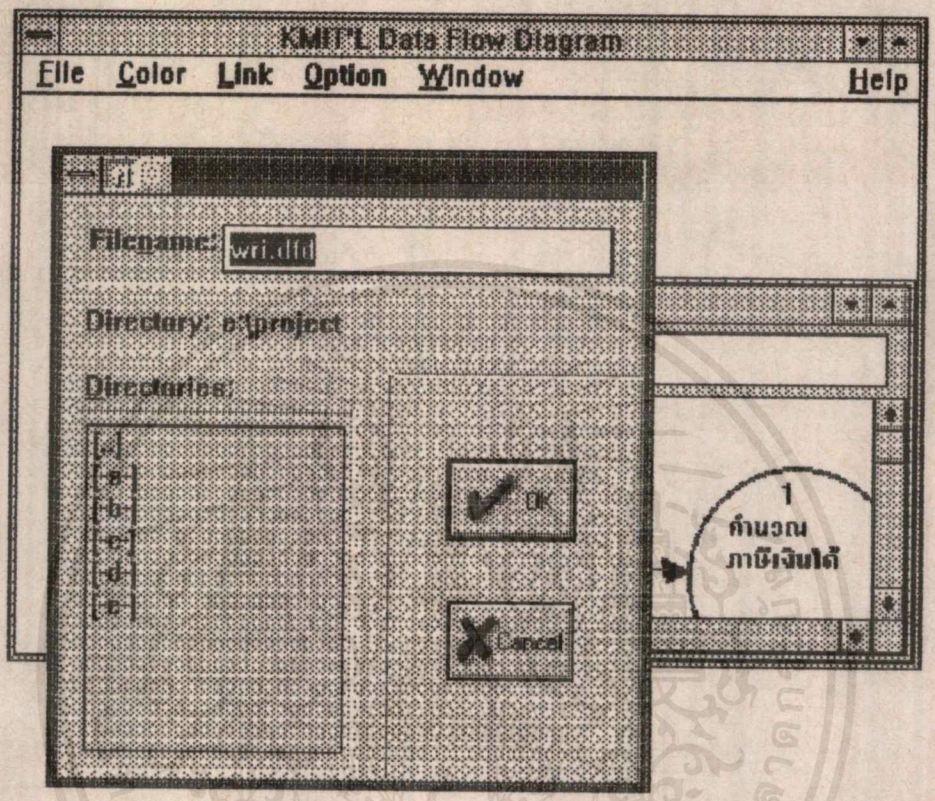


รูปแสดง dialog box ที่จะเกิดขึ้นในกรณี file ที่ต้องการจะ save เป็น file ใหม่

ซึ่งเมื่อให้ชื่อไฟล์แล้ว โปรแกรมก็จะทำการ Save เก็บไฟล์นั้นไว้

การเปลี่ยนชื่อแผนภาพและเก็บลงไฟล์

ทำได้โดยการเลือกเมนู File จากนั้นเลือกเมนู Save file as ก็จะมี File Dialog box ขึ้นมาเพื่อถามชื่อไฟล์ ซึ่งมี default เป็นชื่อ file เดิม บนจอดังนี้คือ



รูปแสดง dialog box ในกรณีต้องการ save file ในชื่อใหม่

เมื่อให้ชื่อไฟล์เสร็จแล้ว โปรแกรมก็จะทำการ Save เก็บไฟล์นั้นไว้

การปิดชุดแผนภาพเพื่อเปลี่ยนชุดแผนภาพใหม่

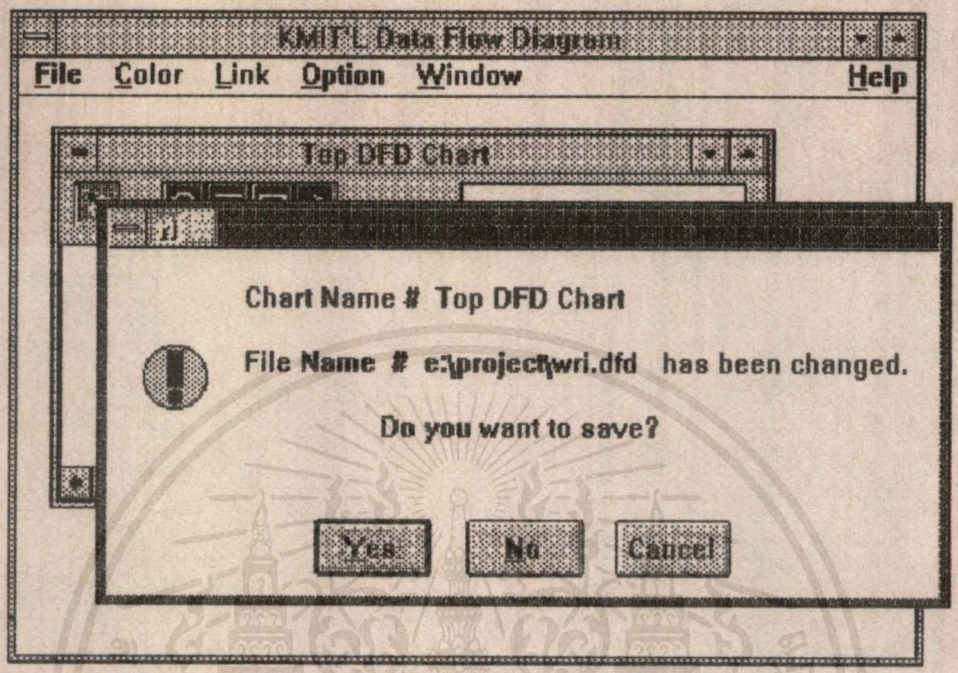
ทำได้โดย 2 วิธีคือ

1.การเลือกเมนู File จากนั้นเลือกเมนู Close Project

2.การเลือกเมนู Window จากนั้นเลือกเมนู Close All

ซึ่งถ้าหากมีการเก็บไฟล์ไว้แล้ว และไม่มีการเปลี่ยนแปลง diagram ก็จะทำให้โปรแกรมปิดชุดแผนภาพนั้นเลย แต่หากไม่มีการเก็บไฟล์ไว้ หรือเก็บไว้แล้ว มีการ

เปลี่ยนแปลงแผนภาพอีก ก็จะทำให้โปรแกรม ถามว่าจะให้เก็บ DFD ลงไฟล์นั้น หรือไม่ หากไม่เก็บโปรแกรมก็จะปิด บนจอดังนี้คือ

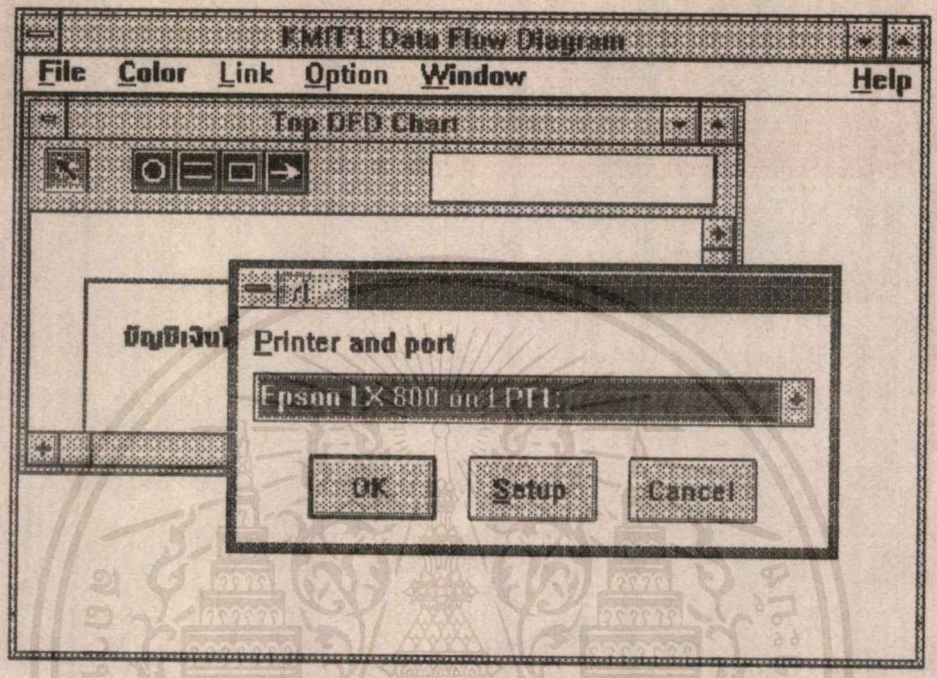


รูปแสดงตัวอย่าง dialog box เตือนว่า file มีการเปลี่ยนแปลงเกิดขึ้น จะให้ save ไม่

ถ้า file นั้นเป็น file ใหม่ หากจะเก็บ โปรแกรมก็จะทำการถามชื่อไฟล์ที่ต้องการจะจัดเก็บ ก็จะมี File Dialog box ขึ้นมาเพื่อถามชื่อไฟล์ เมื่อให้ชื่อไฟล์เสร็จแล้ว โปรแกรมก็จะทำการ Save เก็บไฟล์นั้นไว้ต่อไป

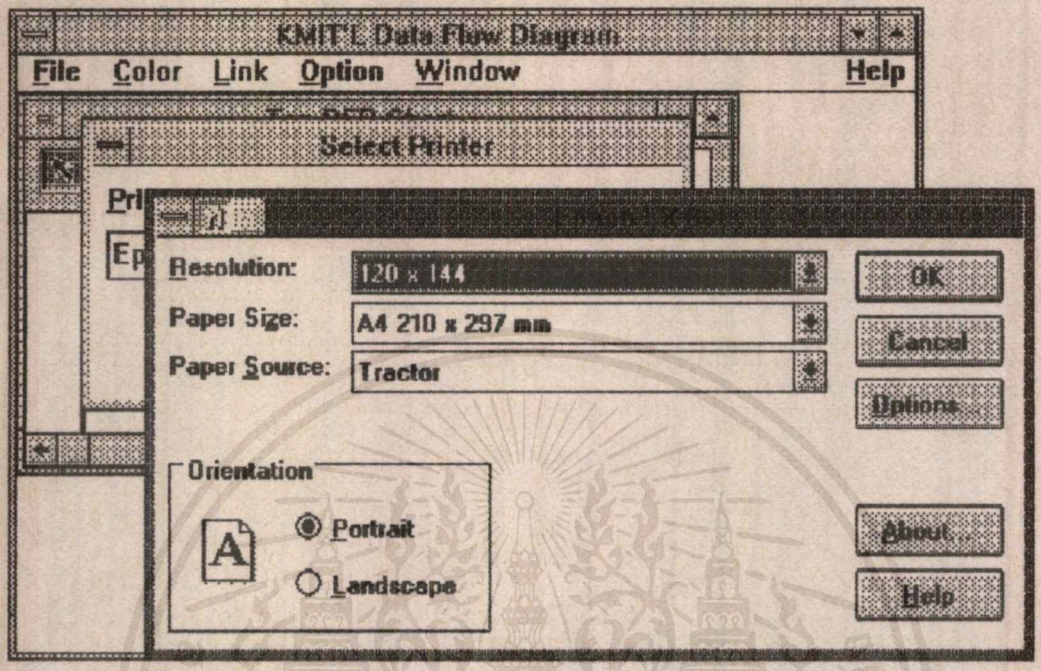
การเตรียมเครื่องพิมพ์

ทำได้โดยการเลือกเมนู File จากนั้นเลือกเมนู Printer setup จากนั้นจะมี printer setup Dialog box ขึ้นมาให้ set option ต่าง ๆ ดังรูป



รูปแสดง dialog box สำหรับเลือก printer

ซึ่งถ้าเราเลือก printer แล้ว เราสามารถที่จะ set up ลักษณะการพิมพ์ตามที่ต้องการโดย click ที่ปุ่ม Setup หลังจาก click แล้ว จะเกิด dialog setup ดังรูป



รูปแสดง setup dialog box หลังจาก click ปุ่ม setup

การพิมพ์

ทำได้โดยการเลือกเมนู File จากนั้นเลือกเมนู print โปรแกรมก็จะเริ่มส่งข้อมูลไปพิมพ์ยัง printer โดยผ่าน Print Manager ของ Window แต่หาก printer ยังไม่พร้อมหรือยังไม่ได้ต่อ ก็จะหยุดการพิมพ์ไว้ โดยจะสามารถพิมพ์ต่อหลังจากเตรียม printer ให้พร้อมหรือใส่กระดาษ ที่ Print Manager

การออกจากโปรแกรม

สามารถทำได้ 2 วิธีคือ

1.ออกโดยผ่าน system menu โดยการ double click mouse ที่ system menu หรือ click mouse ที่ system menu แล้ว เลือก Close หรือ กด Alt + F4

2.ออกโดยผ่าน Application menu โดยการ เลือกเมนู File แล้วเลือก Exit

การเปลี่ยนสี Process

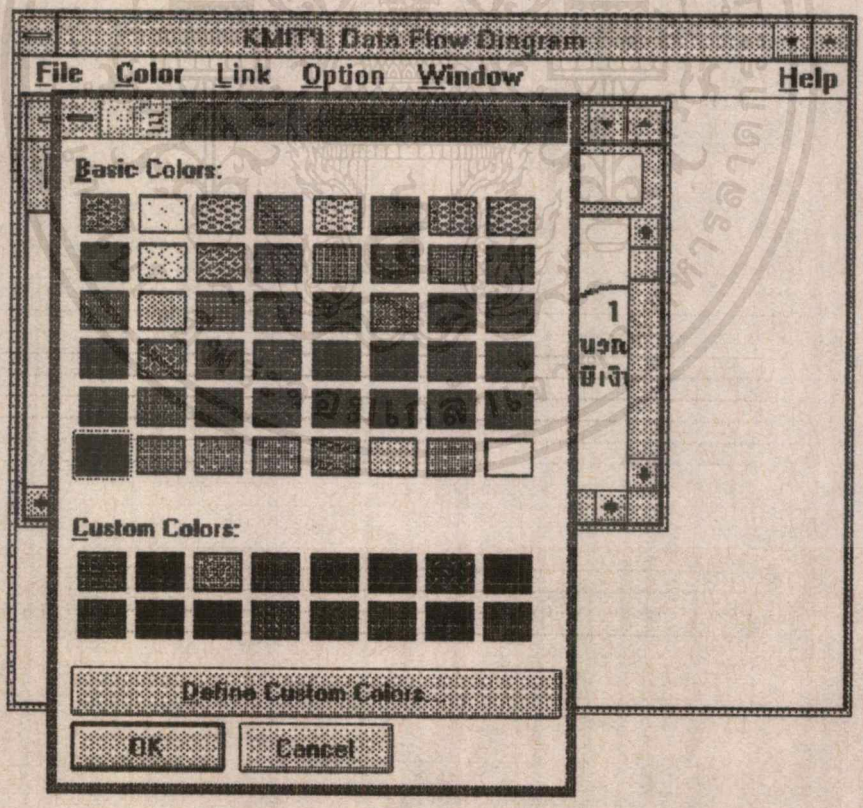
สามารถทำได้โดยการเลือกเมนู Color แล้วเลือก Process color จะมี dialog box ตารางสีขึ้นมาเพื่อให้เลือกสีดังรูป

การเปลี่ยนสี Terminator

สามารถทำได้โดยการเลือกเมนู Color แล้วเลือก Terminator color จะมี dialog box ตารางสีขึ้นมาเพื่อให้เลือกสีดังรูป

การเปลี่ยนสี Storage

สามารถทำได้โดยการเลือกเมนู Color แล้วเลือก Storage color จะมี dialog box ตารางสีขึ้นมาเพื่อให้เลือกสีดังรูป



รูปแสดง dialog box ตารางสี

การเปลี่ยนสีทุก object ให้กลับไปสู่ค่าปกติ

สามารถทำได้โดยการเลือกเมนู Color แล้วเลือก Default color ก็จะทำให้สีของ object ต่าง ๆ กลับไปสู่ค่าที่ตั้งไว้เดิมคือ

Object	Color
Process	red
Terminator	blue
Storage	black green
Flow Data	black

การเปลี่ยนสีทุก object ให้เป็นสีดำ

สามารถทำได้โดยการเลือกเมนู Color แล้วเลือก All black จะทำให้ทุก object เปลี่ยนเป็นสีดำ

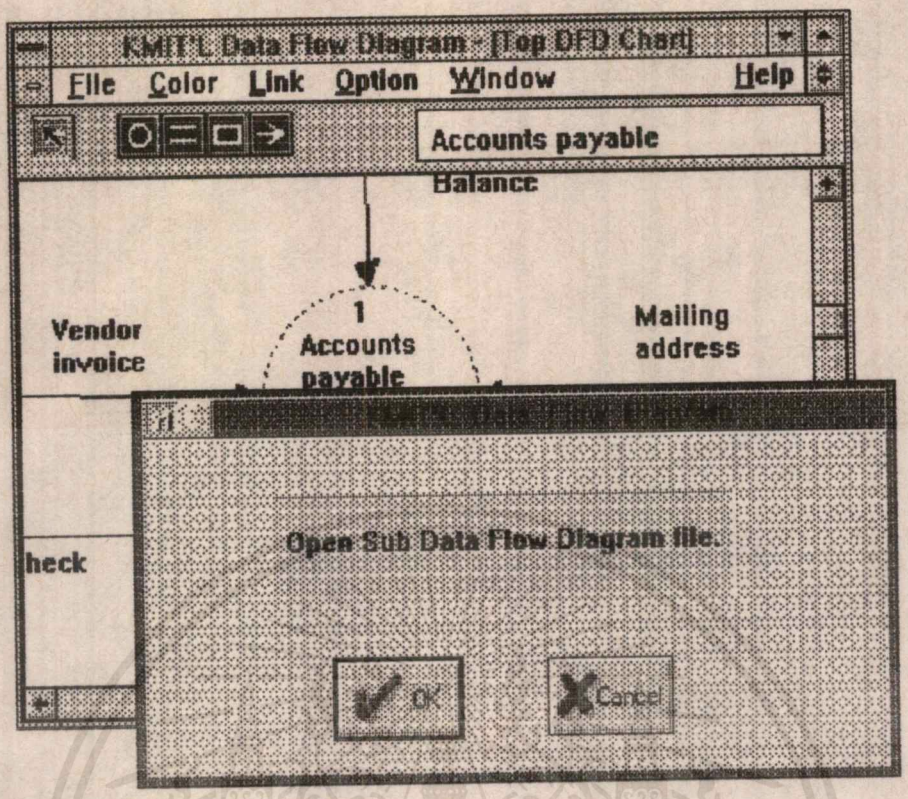
การเปลี่ยนระดับของ DFD หรือการเชื่อมสู่ DFD ระดับล่าง ๆ

สามารถทำได้ 2 วิธีคือ

1.โดยการกดปุ่ม Select and Modify เพื่อเข้าไปอยู่ใน Select and Modify mode แล้วไป click left mouse button ที่ process ที่ต้องการ link สู่ DFD ย่อย จากนั้นเลือกเมนู Link เลือก Link sub DFD

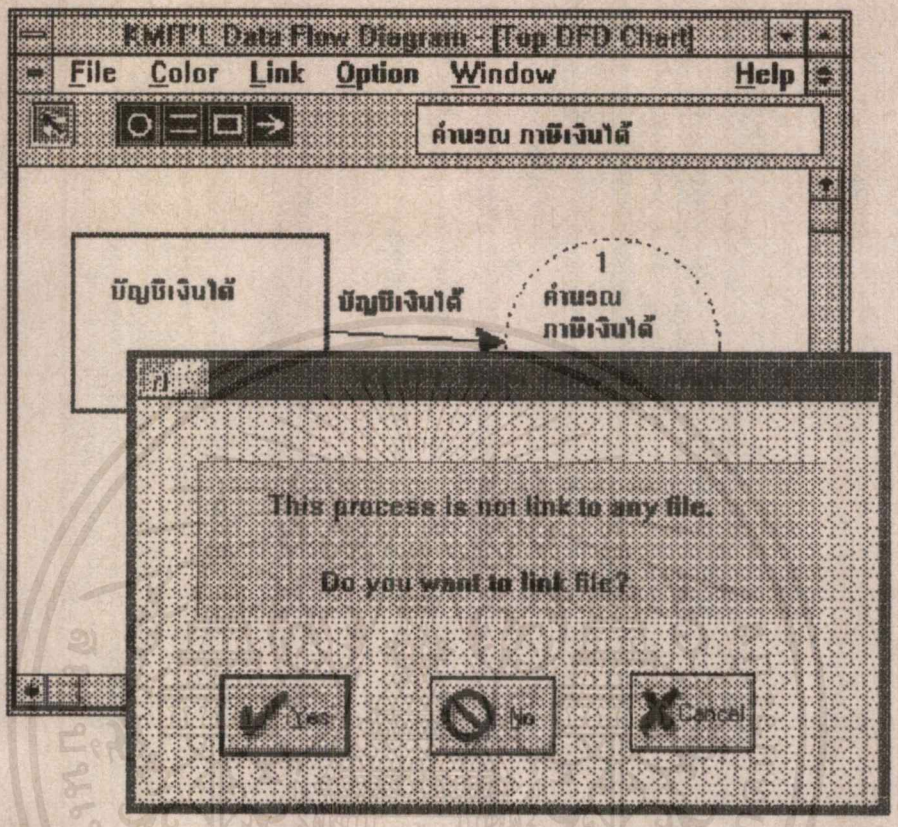
2.โดยการกดปุ่ม Select and Modify เพื่อเข้าไปอยู่ใน Select and Modify mode แล้วไป double click left mouse button ที่ process ที่ต้องการ link สู่ DFD ย่อย

จากนั้นก็จะมี Dialog box ขึ้นมาถามว่าจะเปลี่ยนไปสู่ DFD ระดับล่างหรือไม่ (กรณีมีการสร้าง link สู่ sub DFD ไว้แล้ว) ดังรูป



รูปแสดง dialog box ถามเพื่อความแน่ใจว่าต้องการเปิด sub data flow diagram file จริงหรือไม่

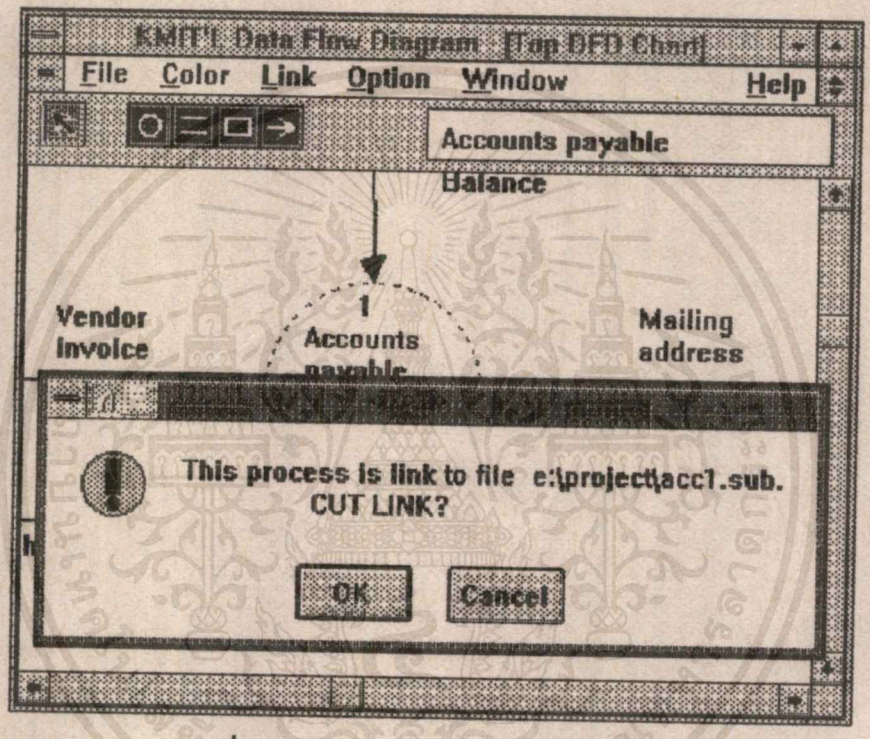
หรือไม่ก็จะถามว่าจะสร้าง link สู่ sub DFD ไฟล์ไหน(กรณีไม่เคยมีการสร้าง link สู่ sub DFD ของ process นั้น ๆ) ดังรูป



รูป dialog box เตือนว่า process ที่ถูกเลือกไม่ได้ link กับ file ใด ต้องการ link file หรือไม่

การตัดการเชื่อมสู่ DFD ระดับล่าง

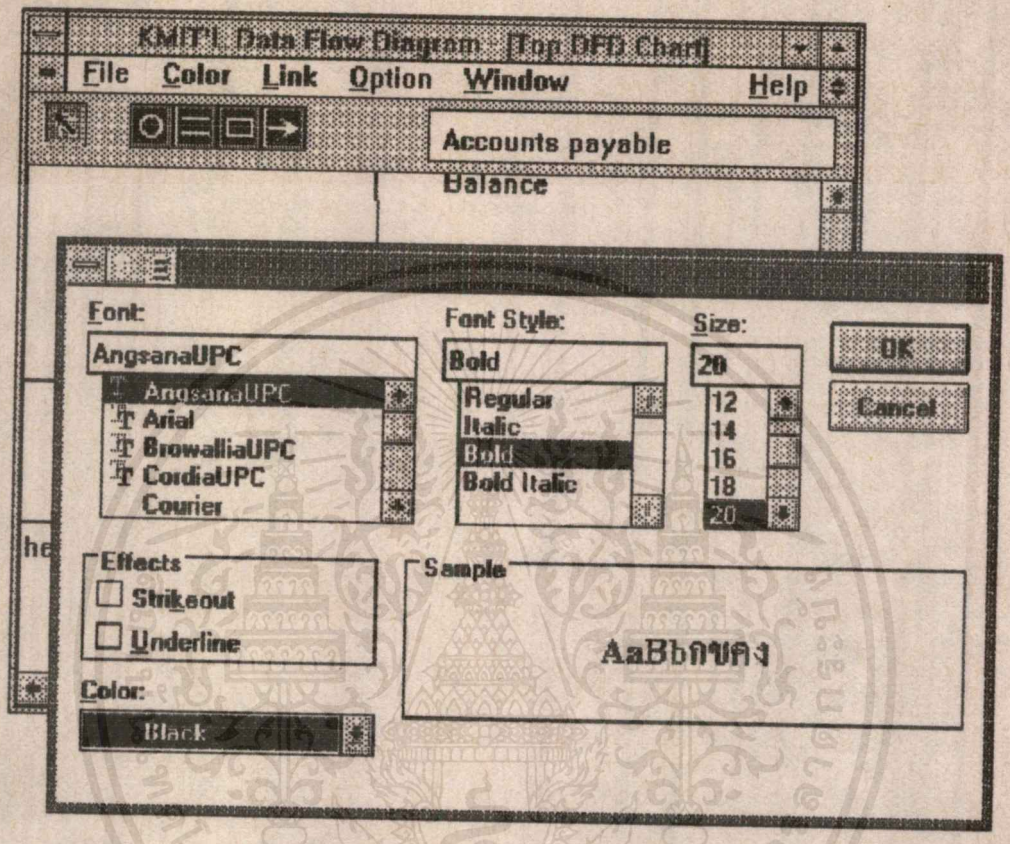
หากมีการสร้าง link สู่ sub DFD ของ process ไว้แล้ว ต้องการตัดการ link นี้ออกก็จะทำได้โดยการการกดปุ่ม Select and Modify เพื่อเข้าไปอยู่ใน Select and Modify mode แล้วไป click left mouse button ที่ process ที่ต้องการตัด link สู่ DFD ย่อย จากนั้นก็เลือกเมนู link เลือก cut link sub ก็จะมี dialog box ขึ้นมาถามว่าจะตัด link หรือเปล่าดังรูป



รูปแสดง dialog box ถามเพื่อความแน่ใจในกรณีต้องการจะตัด file ที่ link กับ process ที่ถูกเลือก

การเปลี่ยนฟอนต์และสีของฟอนต์

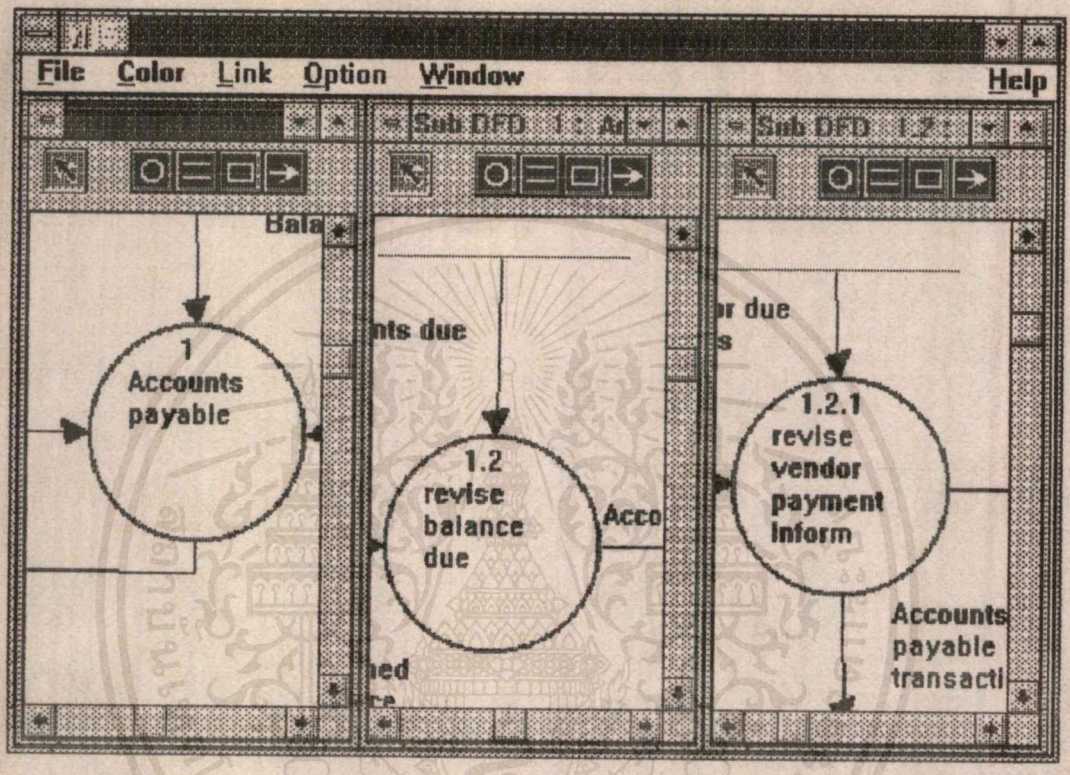
ทำโดยการเลือกเมนู option เลือก font จากนั้นก็จะมี font dialog box ขึ้นมาให้ set ดังรูป



รูปแสดง dialog box การเปลี่ยน fonts

การจัดให้ DFD แต่ละรูปมีขนาดเท่า ๆ กันเฉลี่ยเต็ม Main Window

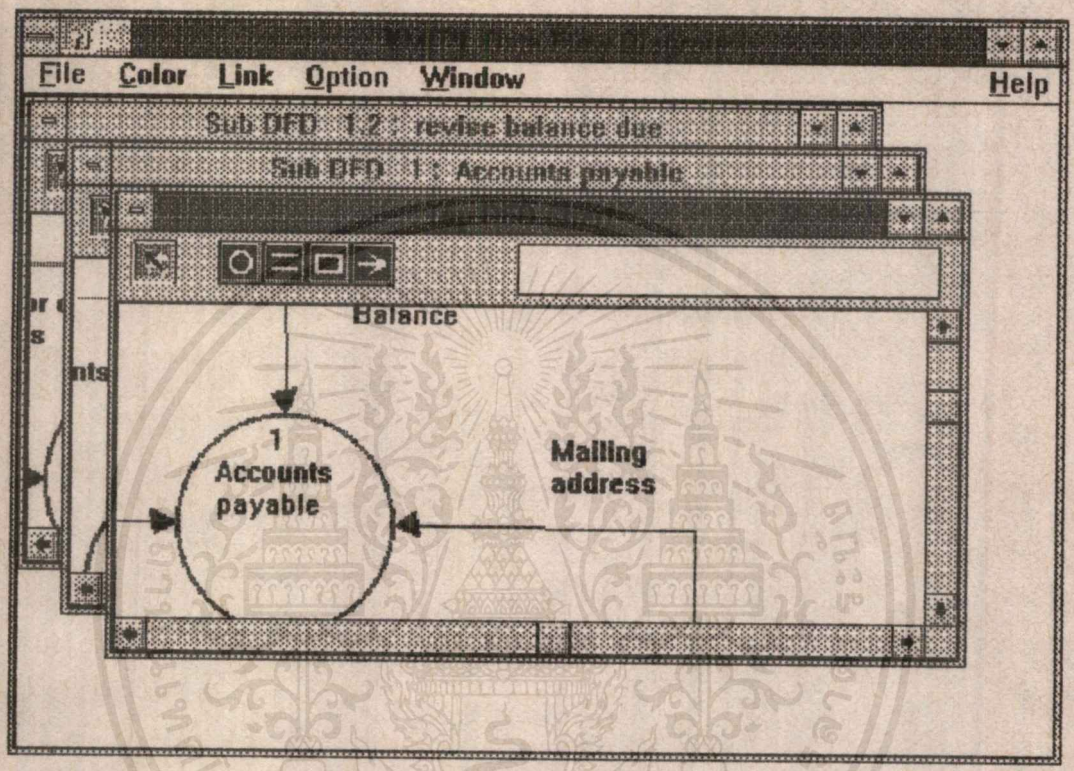
หลังจากเปิด DFD ขึ้นมาหลาย ๆ แผ่น ก็สามารถจัดให้ดูง่าย ๆ โดยให้มีขนาดเฉลี่ย เท่า ๆ กันเต็ม Main Window ได้โดยเลือกเมนู Window แล้วเลือก Tile จะเกิดผลดังรูป



รูปแสดง windows ในลักษณะ tile

การจัดให้ DFD แต่ละรูปซ้อน ๆ กันเป็นแผ่นอย่างมีระเบียบ

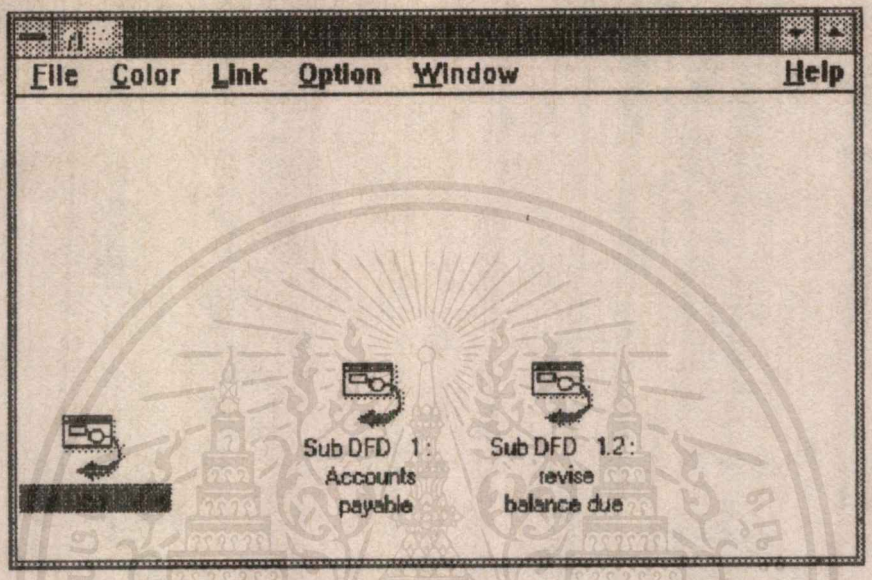
หลังจากเปิด DFD ขึ้นมาหลาย ๆ แผ่น ก็สามารถจัดให้ดูง่าย ๆ โดยให้มีขนาดเรียงซ้อน ๆ กันอย่างเป็นระเบียบใน Main Window ได้โดยเลือกเมนู Window แล้วเลือก Cascade จะเกิดผลดังรูป



รูปแสดง windows ในลักษณะ cascade

การทำให้ DFD แต่ละรูปเป็น icon

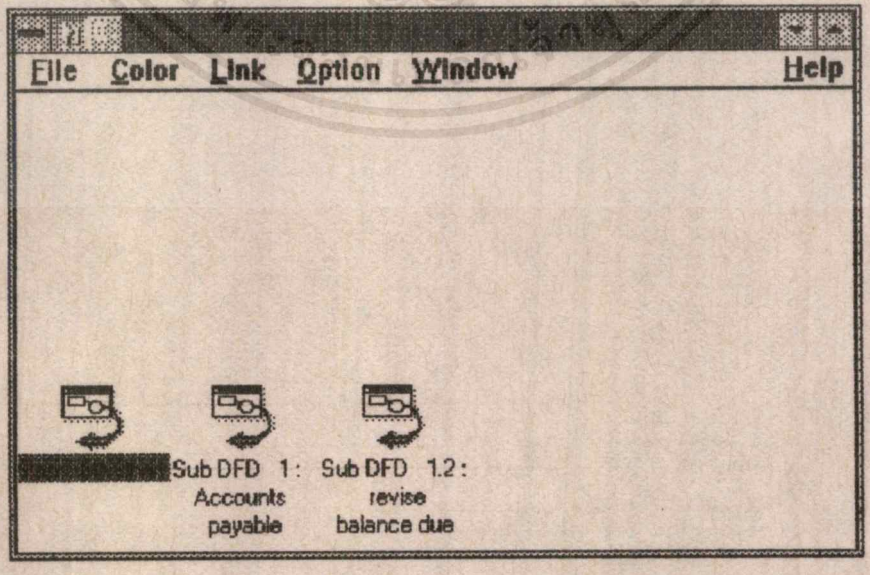
หลังจากเปิด DFD ขึ้นมาหลาย ๆ แผ่น ก็สามารถย่อเก็บเป็น icon เผื่อเรียกมาดูดูอีก โดยเลือก system menu minimize ที่แต่ละ Child Window แล้วเลือก minimize จะเกิดผลดังรูป



รูปแสดง windows ที่ถูกย่อเป็น icon แล้ว

การทำให้ DFD ที่ยุบเป็น icon เรียงกันอย่างเป็นระเบียบ

ทำได้โดยการเลือกเมนู Window แล้วเลือก Arrange icon จะเกิดผลดังรูป



รูปแสดง icons ที่ถูกจัดเรียงแล้ว

การวาดรูป object ลงบนแผนภาพ

หลังจากเปิดไฟล์ DFD มาแล้วโดยการเลือก File->New Project หรือ File->Open Project แล้วก็สามารถทำการวาดแผนภาพได้โดย

หากจะวาด

Process

Storage

Terminator

ให้ไป click left mouse button ที่ปุ่มเหล่านั้น จากนั้นก็ไป click left mouse button ในตำแหน่งที่ต้องการวาด ก็จะปรากฏรูป object ชนิดนั้น ๆ ขึ้นมา จากนั้นก็จะสามารถใส่ชื่อ object นั้นลงไปได้เลยโดยผ่าน keyboard หรือจะไม่ใส่ก็ได้

Flow Data มีพิเศษ คือหากจะทำการวาดรูป object นี้ก็ไป click left mouse button ที่ปุ่ม Flow Data จากนั้น

-หากวาด Flow Data ที่ประกอบด้วยเส้นตรงเส้นเดียว ก็ให้ click left mouse button ที่ตำแหน่งเริ่มต้น

-หากวาด Flow Data ที่ประกอบด้วยเส้นตรงหลายเส้น ก็ให้ click left mouse button ที่ตำแหน่งเริ่มต้น แล้วก็ click left mouse button ที่จุดหักเห ไปเรื่อย ๆ

แล้วไป double click left mouse button ที่ตำแหน่งสิ้นสุด หรือหาก ตำแหน่งสิ้นสุดเป็น object จะใช้แค่ click แทน double click ก็ได้ จากนี้ก็จะสามารถใส่ชื่อลงไปโดยผ่าน ทาง keyboard หรือ ไม่ใส่ก็ได้ (หากไม่ใส่ จะให้ชื่อเป็น noname)

การเปลี่ยนแปลงชื่อหรือให้ชื่อแก่ object

ทำหลังจาก วาดรูป object นั้นเสร็จใหม่ ๆ โดยผ่าน keyboard หรือทำโดยการ click left mouse button (H/W) ที่ Select and Modify button (S/W) แล้วไป click left mouse button ที่ object ที่ต้องการเปลี่ยนชื่อ

Process

Storage

Terminator

จากนั้นก็ทำการแก้ไขหรือให้ชื่อ โดยผ่าน keyboard

Flow Data จะมีพิเศษคือแทนที่ จะ click left mouse button ที่ object ก็ให้ไป click ที่ สี่เหลี่ยมของชื่อ object แทน

การย้ายตำแหน่ง object

หลังจากวาดรูปแต่ละ object ก็จะสามารถย้าย object ได้โดยทำโดยการ click left mouse button (H/W) ที่ Select and Modify button (S/W) แล้วไป click left mouse button ที่ object

Process

Storage

Terminator

ที่ต้องการย้ายแล้ว drag mouse ไป drop object ลงบนตำแหน่งใหม่ที่ต้องการ

หมายเหตุ Flow Data จะไม่สามารถย้ายได้แต่ จะยึดตาม object ที่เชื่อมติดอยู่กับมัน แต่จะสามารถย้ายตำแหน่งชื่อของมันได้โดยวิธีการข้างต้นเช่นกัน

การลบ object จากแผนภาพ

หลังจากวาดรูปแต่ละ object ก็จะสามารถลบ object ได้โดยทำโดยการ click left mouse button (H/W) ที่ Select and Modify button (S/W) แล้วไป click left mouse button ที่ object

Process

Storage

Terminator

Name of Flow Data

จากนั้นก็ click right mouse button เป็นการลบ object นั้น ๆ

หมายเหตุ หากมีการลบ object ที่เชื่อมกันด้วย Flow Data จะทำให้ Flow Data ที่เชื่อมอยู่ถูกลบไปโดยอัตโนมัติ

การเลื่อนแผนภาพบนหน้าจอขึ้นหรือลง

ทำโดยการ click left mouse button ที่ scroll bar หรือที่ปุ่มประจำ scroll bar

~

เอกสารอ้างอิง

Borland C++ Programing for Windows

Peter Norton, Paul L. Yao

A Bantam Book / January 1992

The Waite Group C++ Programing

John Thomas Berry

1998 by The Waite Group, Inc.

Designing Object-Oriented Software

Rebecca Wirfs-Brock, Brain Wilkerson, Laura Wiener

1990 by Prentice-Hall, Inc.

Analysis & Design of Information System

James A. Senn

1989 by McGraw-Hill Co.