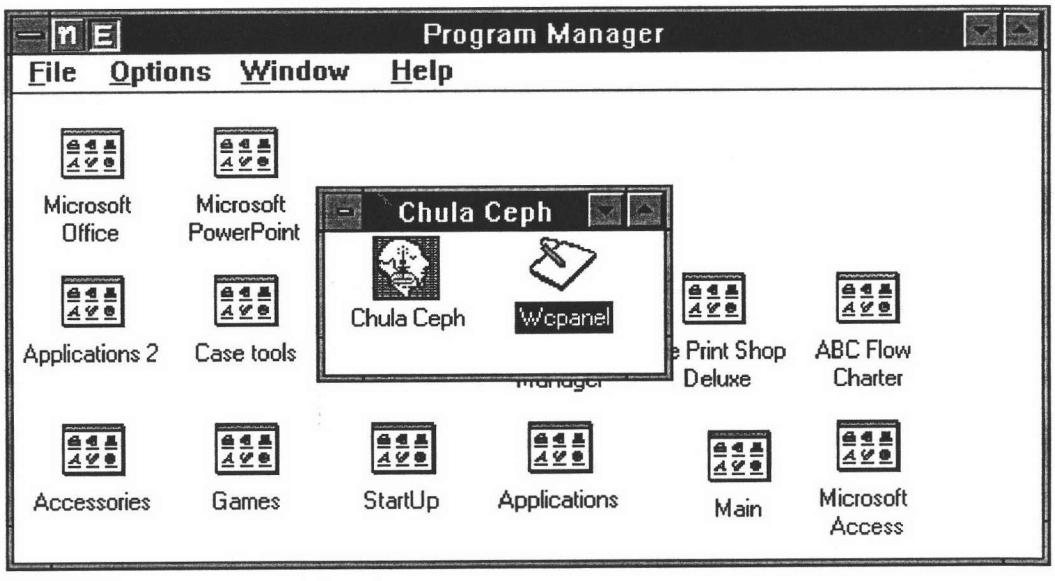
บทที่ 3

การเขียนโปรแกรมบนวินโดว์

ปัจจุบันการพัฒนาโปรแกรมภายใต้สภาวะแวดล้อมแบบวินโดว์ได้เป็นที่นิยมอย่างมาก ดังเห็นได้จากโปรแกรมต่าง ๆ ที่เป็นที่นิยมใช้กันอย่างแพร่หลาย ดังเช่น โปรแกรม Microsoft Word, Microsoft Excel, Microsoft Powerpoint, Corel Draw ฯลฯ เนื่องจากความสะดวก ความมีประสิทธิภาพ และมีการประสานกับผู้ใช้อย่างเป็นรูปแบบที่แน่นอน ทำให้ผู้ใช้สามารถเรียนรู้และใช้งานโปรแกรมได้อย่างรวดเร็วและมีประสิทธิภาพ ผู้ใช้ที่สามารถใช้โปรแกรมชนิดใดชนิดหนึ่งบนวินโดว์ได้ ก็จะสามารถเข้าใจและใช้งาน โปรแกรมอื่น ๆ ได้เช่นกัน



รูปที่ 3.1 แสดงจอภาพของโปรแกรม Microsoft Windows

รูปแบบการเขียนโปรแกรมบนวินโดว์

Kruglinski (1993) ได้อธิบายถึงการเขียนโปรแกรมบนวินโดว์ดังนี้ ในการเขียนโปรแกรมภายใต้ระบบปฏิบัติการ MS-DOS โดยใช้ภาษา C นั้น จะต้องมีส่วนของฟังก์ชันที่ชื่อว่า main ( ) เมื่อผู้ใช้เริ่มต้นโปรแกรม ระบบปฏิบัติการจะเรียกฟังก์ชัน main ( ) หลังจากนั้นฟังก์ชัน main ( ) จะทำการเรียกฟังก์ชันย่อยต่าง ๆ ตามแต่ที่ผู้เขียนโปรแกรมได้กำหนดไว้ ดังแสดงตัวอย่างรูปที่ 3.2

ส่วนการเขียนโปรแกรมภายใต้สภาวะแวดล้อมแบบวินโดว์นั้น โปรแกรมจะมีฟังก์ชันที่ชื่อว่า Winmain ( ) ใช้แทนที่ฟังก์ชัน main ( ) เดิม และฟังก์ชัน Winmain ( ) นี้จะมีหน้าที่สำคัญในการสร้างวินโดว์หลักของโปรแกรม ซึ่งจะมีส่วนประมวลผล message ของตัวเอง ซึ่งจะคอยรับ message ที่ส่งมาจากวินโดว์อื่น และทำการประมวลผลต่อไป ดังแสดงตัวอย่างในรูปที่ 3.3

```
/* Simple  C  Program  */
main ( ) {
                Printf ("Hello  World") ;
        {
```

รูปที่ 3.2 ตัวอย่างโปรแกรมภาษา C ซึ่งใช้ฟังก์ชัน main ( )

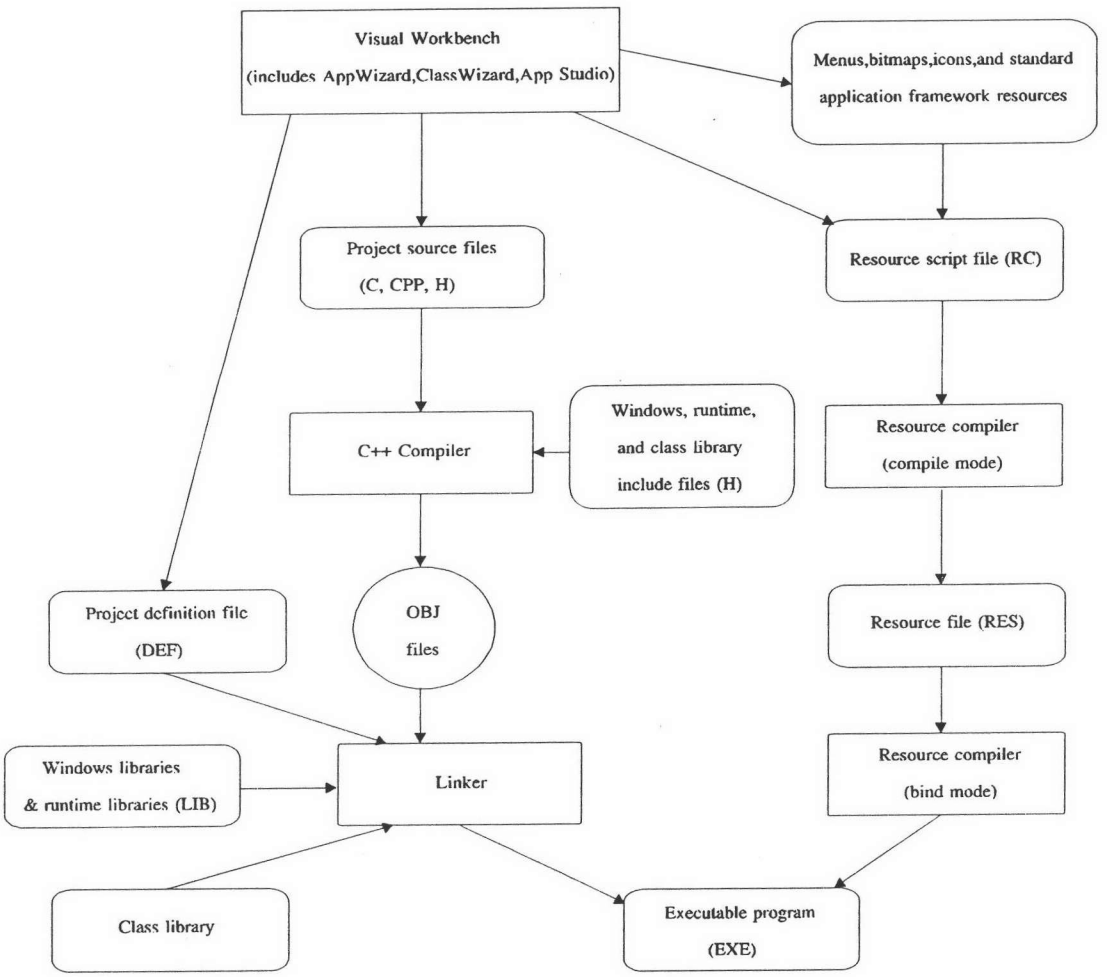
```
Winmain ( ) {
int PASCAL WinMain(hInstance, hPrevInstance, lpCmdLine, nCmdShow)
HANDLE hInstance;              /* current instance */
HANDLE hPrevInstance;          /* previous instance */
LPSTR lpCmdLine;               /* command line */
int nCmdShow;                  /* show-window type (open/icon) */
{
   MSG msg;                    /* message */
   if (!hPrevInstance)         /* Other instances of app running? */
   if (!InitApplication(hInstance)) /* Initialize shared things */
        return (FALSE);        /* Exits if unable to initialize */
   /* Perform initializations that apply to a specific instance */
   if (!InitInstance(hInstance, nCmdShow))
        return (FALSE);
   /* Acquire and dispatch messages until a WM_QUIT message is received.*/
   while (GetMessage(&msg,  /* message structure  */
        NULL,          /* handle of window receiving the message */
        NULL,          /* lowest message to examine */
        NULL))         /* highest message to examine */
   {
   TranslateMessage(&msg);  /* Translates virtual key codes */
   DispatchMessage(&msg);   /* Dispatches message to window */
   }
   return (msg.wParam);        /* Returns the value from PostQuitMessage */
   }
```

รูปที่ 3.3 ตัวอย่างโปรแกรมภาษา C ที่เขียนภายใต้สภาวะแวดล้อมแบบวินโดว์ โดยใช้ฟังก์ชัน Winmain ( ) เป็นฟังก์ชันหลัก

เครื่องมือในการพัฒนาโปรแกรมบนวินโดว์

ในการพัฒนาโปรแกรมบนวินโดว์นั้น สามารถเลือกใช้เครื่องมือในการพัฒนาได้หลายอย่างแล้วแต่ความถนัด โดยมีหลายบริษัทได้ผลิตออกจำหน่าย ดังเช่น Microsoft Visual C++, Borland C++, Microsoft Window SDK แต่ในการวิจัยนี้ ผู้ทำการวิจัยเลือกใช้โปรแกรม Microsoft Visual C++ Version 1.0 ซึ่งจะอธิบายขั้นตอนการพัฒนาโปรแกรมโดยสังเขปดังนี้

1. สร้างแฟ้มข้อมูลส่วนที่เป็นภาษา C หรือ C++ โดยใช้นามสกุล .C หรือ .CPP
2. สร้างแฟ้มข้อมูลทรัพยากร โดยใช้นามสกุล .RC
3. สร้างแฟ้มข้อมูลกำหนดมอดูล โดยใช้นามสกุล .DEF
4. เมื่อสั่งให้โปรแกรม Microsoft Visual C++ สร้างโปรแกรมผลลัพท์ โปรแกรม Microsoft Visual C++ จะทำการแปลโปรแกรมภาษา C หรือ C++ ให้เป็นโปรแกรมภาษาจุดหมาย หลังจากนั้นจะทำการเชื่อมโปรแกรม ภาษาจุดหมายที่ได้เข้าด้วยกันและเชื่อมกับคลังโปรแกรมวินโดว์ ตามข้อมูล ที่ได้กำหนดไว้ในแฟ้มข้อมูลกำหนดมอดูล และสุดท้ายจะทำการเชื่อมกับ ส่วนทรัพยากรที่ได้ทำการแปลแล้ว ก็จะได้โปรแกรมผลลัพท์ ซึ่งสามารถ ทำงานได้ ภายใต้สภาวะแวดล้อมแบบวินโดว์



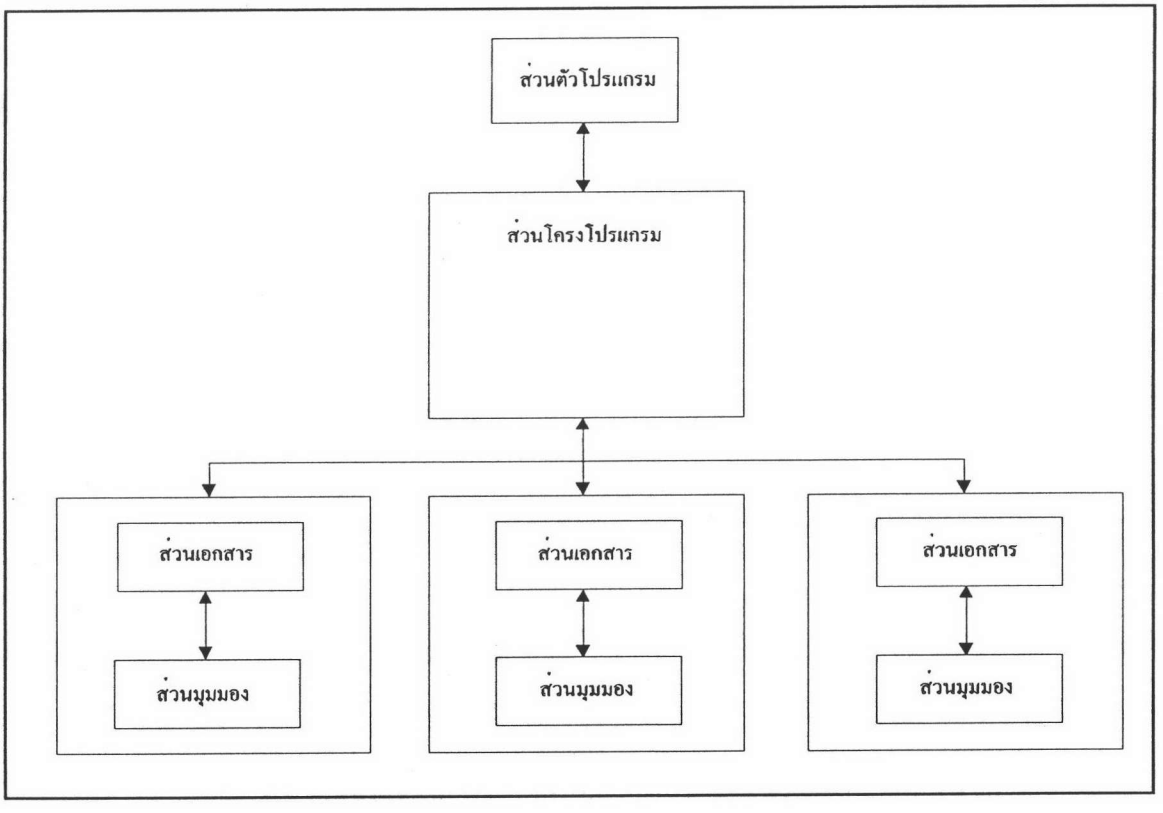
รูปที่ 3.4 แสดงขั้นตอนการพัฒนาโปรแกรมบนวินโดว์

โครงสร้างมาตรฐานของโปรแกรม

โปรแกรม Microsoft Visual C++ มีการกำหนดโครงสร้างของโปรแกรมให้เป็นมาตรฐานเดียวกัน (Microsoft Corporation , 1993) โดยเรียกว่า "Application Framework" ซึ่งประกอบด้วย 4 ส่วน ที่สำคัญดังนี้

1. ส่วนตัวโปรแกรม
2. ส่วนโครงโปรแกรม
3. ส่วนเอกสาร
4. ส่วนมุมมอง

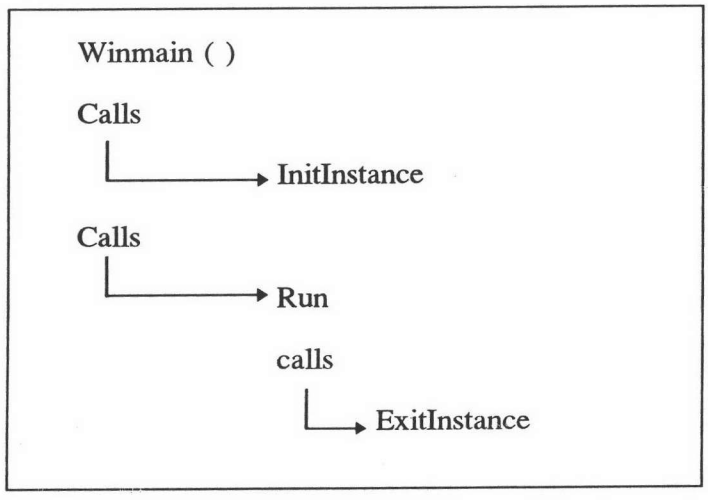
โดยแต่ละส่วนจะถูกแยกออกเป็นแฟ้มข้อมูลต่างหาก และถูกเชื่อมเข้าด้วยกันด้วยส่วนจัดการโครงการของ Microsoft Visual C++ นอกจากนี้แล้วส่วนประกอบทั้ง 4 นี้ยังมีลักษณะเป็นคลาส ซึ่งถ่ายทอดมาจากคลาสพื้นฐาน (Microsoft foundation Class) โดยส่วนตัวโปรแกรมถ่ายทอดมาจากคลาส CWinApp ส่วนโครงโปรแกรมถ่ายทอดมาจาก คลาส CFrameWnd ส่วนมุมมองถ่ายทอดมาจากคลาส CView และส่วนเอกสารถ่ายทอดมาจากคลาส CDocument



รูปที่ 3.5 แสดงส่วนประกอบของโครงสร้างมาตรฐาน

1. <u>ส่วนตัวโปรแกรม</u>

ดังที่กล่าวแล้วข้างต้นว่าส่วนตัวโปรแกรมนั้นถ่ายทอดมาจากคลาส CWinApp โดย คลาส CWinApp นี้จะมีฟังก์ชัน Winmain ( ) ซึ่งเป็นฟังก์ชันหลักของโปรแกรมบนวินโดว์ โดยเมื่อผู้ใช้เริ่มต้นโปรแกรม คลาส CWinApp จะเรียกฟังก์ชัน Winmain ( ) ทำงานโดยอัตโนมัติ ต่อจากนั้นฟังก์ชัน WinMain ( ) จะเรียกฟังก์ชัน InitInstance ( ) และฟังก์ชัน Run ( ) เพื่อกำหนดค่าเริ่มต้นและสร้างส่วนจัดการข่าวสารตามลำดับ ดังรูปที่ 3.6

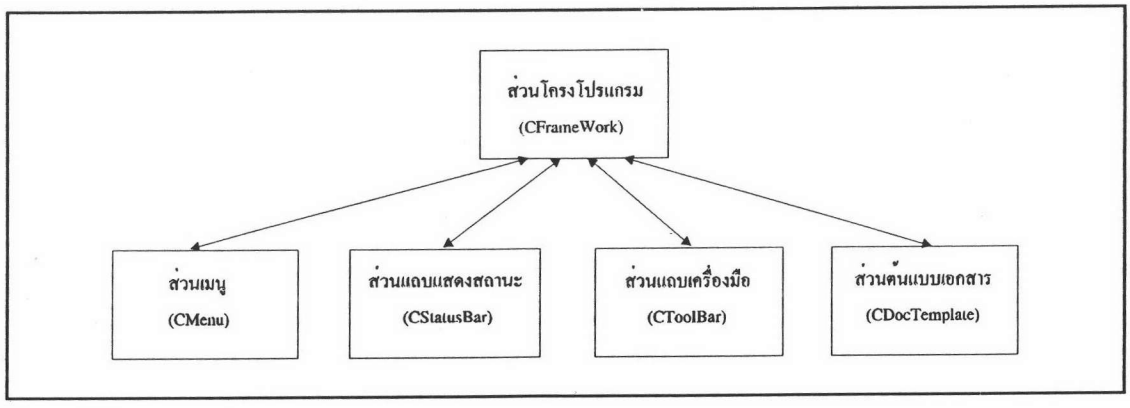


รูปที่ 3.6 แสดงขั้นตอนการทำงานของส่วนตัวโปรแกรม

2. ส่วนโครงโปรแกรม

ส่วนโครงโปรแกรม เป็นส่วนโครงสร้างหลักของโปรแกรมโดยจะประกอบด้วย ส่วนเมนู, ส่วนแถบแสดงสถานะ, ส่วนแถบเครื่องมือ และส่วนต้นแบบเอกสาร โดยมีรายละเอียดดังนี้

ส่วนเมนูจะแสดงรายการเพื่อให้ผู้ใช้เลือก ส่วนแถบสถานะจะเป็นส่วนสำหรับแสดงข้อความเพื่อติดต่อกับผู้ใช้, ส่วนแถบเครื่องมือจะเป็นส่วนให้ผู้ใช้เลือก โดยมีลักษณะเหมือนปุ่มกดหรือแถบเลือก และส่วนต้นแบบเอกสาร ซึ่งถือเป็นส่วนหลักในการจัดการส่วนเอกสาร และส่วนมุมมอง ดังแสดงในรูปที่ 3.7



รูปที่ 3.7 แสดงส่วนประกอบของส่วนโครงโปรแกรม

3. ส่วนเอกสาร

ส่วนเอกสารจะเป็นส่วนที่จัดการกับข้อมูลโดยตรง จะถูกถ่ายทอดจากคลาส CDocument โดยจะมีฟังก์ชันที่ใช้ในการเรียกข้อมูล จัดเก็บข้อมูล และสร้างข้อมูลชุดใหม่ แต่ส่วนเอกสารไม่มีส่วนแสดงผลอยู่ด้วย ต้องอาศัยส่วนมุมมองในการติดต่อกับผู้ใช้ โดยส่วนโครงโปรแกรมจะสร้างส่วนต้นแบบเอกสาร และประกอบส่วนเอกสารเข้ากับส่วนมุมมอง เป็นชุด ๆ เพื่อสามารถเรียกใช้ได้

4. ส่วนมุมมอง

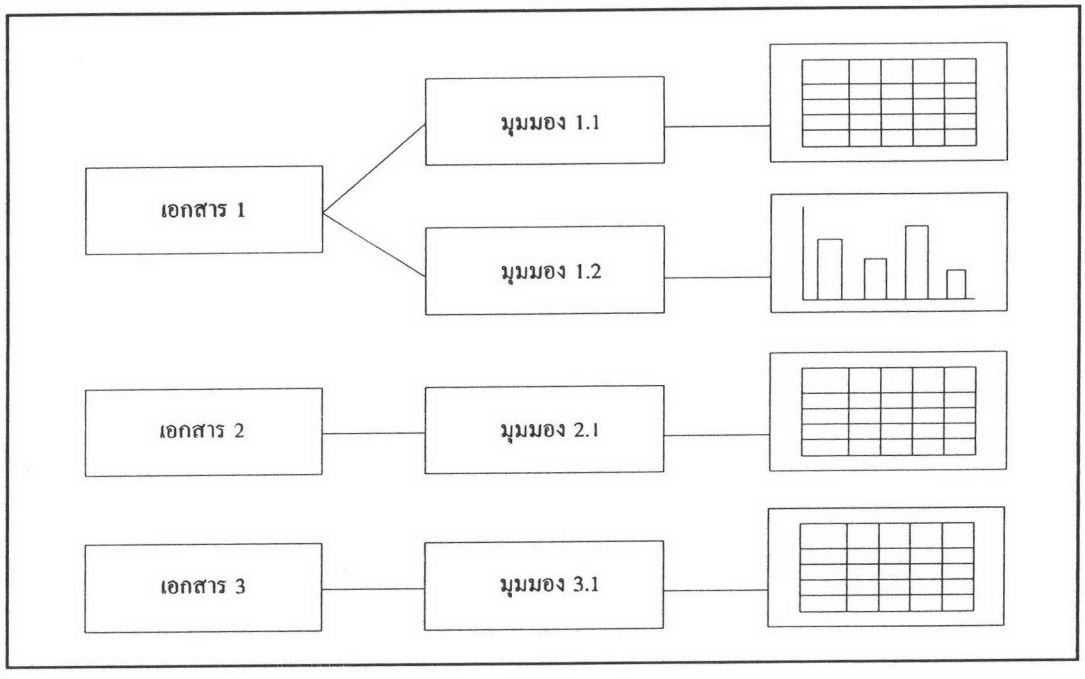
ส่วนมุมมองจะถูกถ่ายทอดมาจากคลาส CView โดยคลาส CView ยังมีคลาสย่อยอีกสองคลาสคือ CScrollView และ CFormView

โดยส่วนมุมมองจะเป็นส่วนแสดงผลต่อผู้ใช้ และส่วนเอกสารหนึ่งชุดสามารถมีส่วนมุมมองได้หลายมุมมอง ดังจะกล่าวรายละเอียดในหัวข้อ ความสัมพันธ์ระหว่างส่วนเอกสารและส่วนมุมมอง

ในโปรแกรมหนึ่งโปรแกรมจะมีส่วนของตัวโปรแกรมและโครงโปรแกรมได้เพียงครั้งเดียว แต่มีส่วนของเอกสารและมุมมองได้มากกว่าหนึ่งครั้ง

ความสัมพันธ์ระหว่างส่วนเอกสารและส่วนมุมมอง

ในการจัดการข้อมูลแบบโครงสร้างมาตรฐานนั้น จะมองข้อมูลออกเป็น 2 ส่วน คือ ส่วนตัวข้อมูลโดยเรียกว่าส่วนเอกสาร และส่วนมุมมองของข้อมูลซึ่งเรียกว่าส่วนมุมมอง ในข้อมูลหนึ่งชุดนั้นจะประกอบส่วนเอกสารเพียงครั้งเดียว แต่อาจประกอบด้วยส่วนมุมมองได้มากกว่าหนึ่งครั้ง ดังแสดงในรูปที่ 3.8



รูปที่ 3.8 แสดงความสัมพันธ์ระหว่างส่วนเอกสารและส่วนมุมมอง

โปรแกรมที่ถูกพัฒนาขึ้นโดยใช้ Microsoft Visual C++ นั้น สามารถแบ่งได้ 2 รูปแบบคือ โปรแกรมชนิดเอกสารเดียว (SDI) และโปรแกรมชนิดหลายเอกสาร (MDI) โดยโปรแกรมชนิดเอกสารเดียว จะสามารถเปิดแฟ้มข้อมูลได้ครั้งละหนึ่งแฟ้มเท่านั้น แต่โปรแกรมชนิดหลายเอกสารจะสามารถเปิดแฟ้มข้อมูลได้ครั้งละหลายแฟ้มพร้อม ๆ กัน

<u>วิธีการแปลงส่งจุดในวินโดว์</u>

Holzner (1994) ได้อธิบายวิธีการแปลงส่งจุดในวินโดว์ดังนี้ ในการเขียนโปรแกรม ภายใต้สภาวะแวดล้อมแบบวินโดว์นั้น จะใช้ระนาบ 2 ชนิด คือ ระนาบเชิงตรรกะและระนาบเชิงอุปกรณ์ ระนาบเชิงอุปกรณ์จะมีหน่วยวัดหนึ่งหน่วยเท่ากับหนึ่งจุดภาพบนจอภาพ โดยค่าทางแกน x จะเพิ่มขึ้นจากซ้ายไปขวา และค่าทางแกน y จะเพิ่มจากบนลงล่าง ส่วนระนาบเชิงตรรกะนั้นหน่วยวัดจะขึ้นกับวิธีการแปลงส่งจุด ซึ่งมีทั้งหมด 8 วิธี ดังนี้

1. MM_TEXT

เป็นวิธีการแปลงส่งจุดแบบหนึ่งต่อหนึ่ง โดยหนึ่งจุดบนระนาบเชิงตรรกะจะมีขนาดเท่ากับหนึ่งจุดบนระนาบเชิงอุปกรณ์ มีค่าทางแกน x เพิ่มขึ้นจากซ้ายไปขวา ทางแกน y เพิ่มจากบนลงล่าง

2. MM_LOENGLISH

เป็นวิธีการแปลงส่งจุด โดยหนึ่งจุดบนระนาบเชิงตรรกะจะมีขนาดเท่ากับ 0.01 นิ้ว มีค่าทางแกน x เพิ่มขึ้นจากซ้ายไปขวา และมีค่าทางแกน y เพิ่มขึ้นจากล่างขึ้นบน

3. MM_HIENGLISH

เป็นวิธีการแปลงส่งจุด โดยหนึ่งจุดบนระนาบเชิงตรรกะมีขนาดเท่ากับ 0.001 นิ้ว มีค่าทางแกน x เพิ่มขึ้นจากซ้ายไปขวา และมีค่าทางแกน y เพิ่มขึ้นจากล่างขึ้นบน

4. MM_LOMETRIC

เป็นวิธีการแปลงส่งจุด โดยหนึ่งจุดบนระนาบเชิงตรรกะมีขนาดเท่ากับ 0.1 มิลลิเมตร มีค่าทางแกน x เพิ่มขึ้นจากซ้ายไปขวา และมีค่าทางแกน y เพิ่มขึ้นจากล่างขึ้นบน

5. MM_HIMCTRIC

เป็นวิธีการแปลงส่งจุด โดยหนึ่งจุดบนระนาบเชิงตรรกะ มีขนาดเท่ากับ 0.01 มิลลิเมตร มีค่าทางแกน x เพิ่มขึ้นจากซ้ายไปขวา และมีค่าทางแกน y เพิ่มขึ้นจากล่างขึ้นบน

6. MM_TWIPS

เป็นวิธีการแปลงส่งจุด โดยหนึ่งจุดบนระนาบเชิงตรรกะ มีขนาดเท่ากับ 1/1440 นิ้ว มีค่าทางแกน x เพิ่มขึ้นจากซ้ายไปขวา และมีค่าทางแกน y เพิ่มขึ้นจากล่างขึ้นบน

7. MM_ISOTROPIC

เป็นวิธีการแปลงส่งจุด โดยไม่กำหนดค่าการแปลงจากระนาบเชิงตรรกะไปยังค่าบนระนาบเชิงอุปกรณ์เป็นค่าคงที่ แต่จะกำหนดเป็นอัตราส่วนซึ่งสามารถเปลี่ยนแปลงได้ และอัตราส่วนการแปลงในแกน x และ y จะต้องเท่ากัน โดยมีสมการดังนี้

$$Dx = (Lx - xWO) * xVE / xWE + xVo$$

$$Dy = (Ly - yWO) * yVE / yWE + yVO$$

$$Lx = (Dx - xVO) * xWE / xVE + xWO$$

$$Ly = (Dy - yVO) * yWE / yVE + yWO$$

เมื่อ

xWO คือ ค่าจุดเริ่มต้นของวินโดว์ ในแกน x บนระนาบเชิงตรรกะ

yWO คือ ค่าจุดเริ่มต้นของวินโดว์ ในแกน y บนระนาบเชิงตรรกะ

xWE คือ ค่าความกว้างของวินโดว์ ในแกน x บนระนาบเชิงตรรกะ

yWE คือ ค่าความกว้างของวินโดว์ ในแกน y บนระนาบเชิงตรรกะ

xVO คือ ค่าจุดเริ่มต้นของวิวพอร์ต ในแกน x บนระนาบเชิงอุปกรณ์

yVO คือ ค่าจุดเริ่มต้นของวิวพอร์ต ในแกน y บนระนาบเชิงอุปกรณ์

xVE คือ ค่าความกว้างของวิวพอร์ต ในแกน x บนระนาบเชิงอุปกรณ์

yVE คือ ค่าความกว้างของวิวพอร์ต ในแกน y บนระนาบเชิงอุปกรณ์

Lx คือ ค่าจุดบนระนาบเชิงตรรกะ ในแกน x

Ly คือ ค่าจุดบนระนาบเชิงตรรกะ ในแกน y

Dx คือ ค่าจุดบนระนาบเชิงอุปกรณ์ ในแกน x

Dy คือ ค่าจุดบนระนาบเชิงอุปกรณ์ ในแกน y

8. MM_ANISOTROPIC

เป็นวิธีการแปลงส่งจุด โดยไม่กำหนดค่าการแปลงจากระนาบเชิงตรรกะไปยังค่าบนระนาบเชิงอุปกรณ์เป็นค่าคงที่ แต่จะกำหนดเป็นอัตราส่วนซึ่งสามารถเปลี่ยนแปลงได้ และอัตราส่วนการแปลงในแกน x และ y อาจไม่เท่ากันก็ได้ และใช้สมการในการแปลงค่าจากระนาบเชิงตรรกะไปยังระนาบเชิงอุปกรณ์ เหมือนกับวิธีการส่งจุดแบบ MM_ISOTROPIC

วิธีการแปลงส่งจุด	จำนวนหน่วยบนระนาบเชิงตรรกะ	ขนาด
MM_LOENGLISH	100	1 นิ้ว
MM_HIENGLISH	1000	1 นิ้ว
MM_LOMETRIC	10	1 มิลลิเมตร
MM_HIMETRIC	100	1 มิลลิเมตร
MM_TEXT	1	1 จุดภาพ
MM_TWIPS	1440	1 นิ้ว

ตารางที่ 3.1 วิธีการแปลงส่งจุดในวินโดว์