บทที่ 4

เทคนิคการเขียนโปรแกรมบนไมโครซอฟต์วินโดวส์

ความเป็นมา

ในปีค.ศ.1983บริษัทไมโครซอฟต์คอร์ปอเรชันได้เปิดตัวซอฟต์แวร์ไมโครซอฟต์วินโดวส์เป็นครั้งแรกซึ่งเป็นระบบการทำงานในสภาพแวดล้อมแบบกราฟิกทำหน้าที่เป็นตัวเชื่อมต่อระหว่างผู้ใช้งานกับอุปกรณ์ต่างๆของคอมพิวเตอร์ ทำให้เกิดความสะดวกในการใช้งาน โดยเริ่มวางจำหน่ายรุ่น 1.01 ในอีก 2 ปีถัดมา หลังจากนั้นบริษัทฯได้ทำการพัฒนาขีดความสามารถเพิ่มขึ้น ให้สามารถรองรับกับอุปกรณ์ต่างๆได้หลากหลายชนิดและออกรุ่น 2.0 ในปี ค.ศ.1987 ซึ่งกล่าวได้ว่ารุ่นนี้ได้มีการเปลี่ยนแปลงไปจากรุ่น 1.01 ค่อนข้างมาก โดยวินโดว์ที่เปิดขึ้นมาสามารถซ้อนทับกันได้หลายๆชั้น รวมทั้งยังสนับสนุนการใช้แผงแป้นอักขระร่วมกับเมาส์อีกด้วย

ภายหลังจากออกรุ่น 2.0 ไม่นานนัก บริษัทฯก็ได้ออกไมโครซอฟต์วินโดวส์รุ่น/386 เพื่อใช้กับเครื่องไมโครคอมพิวเตอร์ที่ใช้ซีพียู 80386 โดยตรง ซึ่งในรุ่นนี้ บริษัทฯได้เพิ่มขีดความสามารถของไมโครซอฟต์วินโดวส์ขึ้นมาก กล่าวคือได้พัฒนาให้ใช้ความสามารถพิเศษของซีพียู 80386 คือการทำงานในภาวะเสมือน-86 (virtual-86 mode) ทำให้สามารถนำโปรแกรมที่ทำงานภายใต้ระบบปฏิบัติการดอสโดยตรง มาทำงานบนไมโครซอฟต์วินโดวส์ได้พร้อมๆกันหลายๆโปรแกรม และได้เปลี่ยนชื่อของไมโครซอฟต์วินโดวส์รุ่น 2.0 เป็นไมโครซอฟต์วินโดวส์/286 เพื่อให้เกิดความสอดคล้องกัน

ในปี ค.ศ. 1990 บริษัทฯก็ได้ออกไมโครซอฟต์วินโดวส์รุ่น 3.0 ซึ่งได้มีการเปลี่ยนแปลงพัฒนาไปจากรุ่นเดิมมาก โดยได้รวมไมโครซอฟต์วินโดวส์ทั้งสองรุ่นเข้าด้วยกัน ปรับปรุงการทำงานของเปลือก(shell)ของไมโครซอฟต์วินโดวส์ให้ดีขึ้น สนับสนุนการทำงานกับระบบเครือข่าย(network)และรูปแบบของแผนที่บิตชนิดใหม่ที่ไม่ขึ้นกับอุปกรณ์ต่างๆ(Device Independent Bitmap : DIB) นอกจากนี้ส่วนที่มีการเปลี่ยนแปลงที่สำคัญที่สุดคือการสนับสนุนการใช้งานหน่วยความจำแบบยืดขยาย (extended memory) โดยสามารถอ้างอิงหน่วยความจำได้ถึง 16 เมกะไบต์ รวมทั้งถ้าหากทำงานบนซีพียู 80386 หรือสูงกว่าในภาวะ 386-Enhanced แล้วไมโครซอฟต์วินโดวส์จะใช้การจัดการหน่วยความจำแบบเสมือนของซีพียูซึ่งจะทำให้สามารถใช้หน่วยความจำได้มากกว่าที่มีอยู่จริงถึง 4 เท่า

อย่างไรก็ตามในรุ่น 3.0 นี้ก็ยังมีข้อจำกัดในเรื่องการแสดงผลของตัวอักษร เพราะตัวอักษรที่ใช้เป็นแบบแผนที่บิต ทำให้การแสดงผลตัวอักษรขนาดใหญ่ไม่สวยงามเนื่องจากนำแผนที่บิตมาขยาย จึงได้ร่วมมือกับ

บริษัทแอปเปิลคอมพิวเตอร์พัฒนารูปแบบตัวอักษรใหม่ขึ้นในรุ่น 3.1 ในปีคศ. 1992 เรียกว่ารูปแบบอักษรทรูไทป์ (True Type Font) ซึ่งเป็นตัวอักษรแบบโครงร่าง ทำให้สามารถย่อ-ขยายตัวอักษรได้ตามความต้องการ นอกจากนี้ก็ยังได้พัฒนาขีดความสามารถการแลกเปลี่ยนข้อมูลที่เรียกว่า OLE (Object Linked and Embedded) สนับสนุนระบบสื่อหลายแบบ(Multimedia) รวมทั้งปรับปรุงโปรแกรมต่างๆให้มีประสิทธิภาพดียิ่งขึ้น และในรุ่นนี้บริษัทฯก็ได้ตัดการสนับสนุนการใช้งานไมโครซอฟต์วินโดวส์บนเครื่องคอมพิวเตอร์ที่มีซีพียู 8088 หรือที่เรียกว่าภาวะจริง(real mode)

<u>จุดเด่นของไมโครซอฟต์วินโดวส์</u>

1. ระบบตัวประสานกับผู้ใช้แบบกราฟิก(Graphic User Interface : GUI)

เป็นการติดต่อกับผู้ใช้ผ่านทางรูปภาพกราฟิกที่สื่อความหมายถึงการทำงานต่างๆ เช่น สัญรูป(icon) แท่งเลื่อน(scroll bar) ทำให้เกิดความสวยงาม มีความน่าใช้ มีความสะดวกและง่ายต่อการเรียนรู้ นอกจากนี้ยังมีความเป็นมาตรฐาน โดยโปรแกรมต่างๆจะมีระบบตัวประสานกับผู้ใช้เหมือนๆกัน

2. การทำงานแบบหลายภารกิจ(Multitasking)

สภาพแวดล้อมของไมโครซอฟต์วินโดวส์ยินยอมให้ผู้ใช้เรียกใช้โปรแกรมมาทำงานได้หลายๆโปรแกรม หรือโปรแกรมเดียวกันพร้อมๆกัน โดยไมโครซอฟต์วินโดวส์จะเป็นผู้จัดสรรหน่วยความจำให้แต่ละโปรแกรมอย่างพอเพียงรวมทั้งคอยจัดการการใช้ทรัพยากรต่างๆร่วมกัน

3. ความเป็นอิสระต่ออุปกรณ์

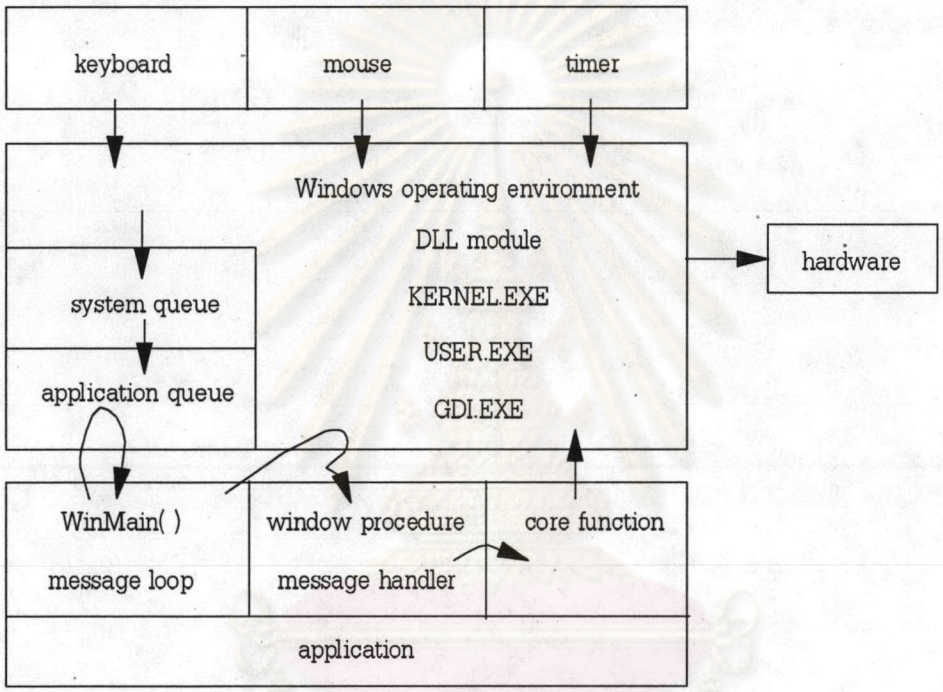
เนื่องจากไมโครซอฟต์วินโดวส์เป็นสภาพแวดล้อมที่เป็นตัวกลางระหว่างโปรแกรมที่เขียนขึ้นกับอุปกรณ์ต่างๆ โปรแกรมจึงไม่ต้องคอยจัดการกับอุปกรณ์ต่างๆโดยตรง เพียงแต่เรียกใช้ฟังก์ชันต่างๆที่ไมโครซอฟต์วินโดวส์จัดเตรียมไว้ให้ถูกต้องเท่านั้น ทำให้ผู้ใช้สามารถเปลี่ยนอุปกรณ์ในระบบได้เลย โดยไม่จำเป็นต้องเปลี่ยนแปลงโปรแกรมที่มีอยู่

4. การแลกเปลี่ยนข้อมูลระหว่างโปรแกรม

ไมโครซอฟต์วินโดวส์ได้สนับสนุนการโอนย้ายแลกเปลี่ยนข้อมูลระหว่างโปรแกรมได้ง่ายด้วยการคัดลอกข้อมูลจากโปรแกรมหนึ่งเข้าไปยังคลิปบอร์ด(clipboard) แล้วจึงคัดลอกจากคลิปบอร์ดไปยังอีกโปรแกรมหนึ่งได้ นอกจากนี้ยังมีขีดความสามารถในการเชื่อมโยงแฟ้มข้อมูลเข้าด้วยกันแบบ DDE (Dynamic Data Exchange) และ OLE(Object Linked and Embedded)

การทำงานของไมโครซอฟต์วินโดวส์

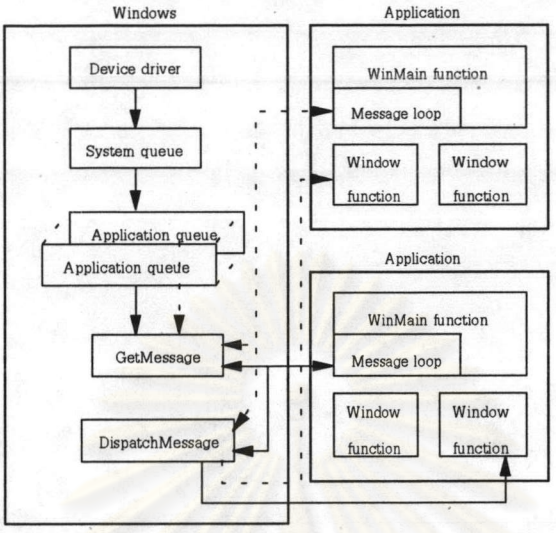
ดังที่กล่าวแล้วว่าซอฟต์แวร์ไมโครซอฟต์วินโดวส์เป็นซอฟต์แวร์ที่ช่วยให้การใช้งานเครื่องคอมพิวเตอร์เป็นไปอย่างมีประสิทธิภาพโดยไมโครซอฟต์วินโดวส์จะเป็นตัวเชื่อมต่อระหว่างโปรแกรมประยุกต์กับอุปกรณ์ฮาร์ดแวร์โดยจะมีฟังก์ชันการติดต่อให้โปรแกรมประยุกต์ต่างๆเรียกใช้ (Application Programming Interface)* ดังรูปที่ 4.1



รูปที่ 4.1 แสดงการทำงานของไมโครซอฟต์วินโดวส์และโปรแกรมประยุกต์

ไมโครซอฟต์วินโดวส์จะเป็นตัวคอยจับสัญญาณจากอุปกรณ์นำเข้าข้อมูล (Input Hardware) อาทิเช่น แผงแป้นอักขระ เมาส์ นาฬิกา เมื่ออุปกรณ์เหล่านี้ให้กำเนิดสัญญาณมา วินโดวส์ก็จะสร้างข้อความ(message)*ที่เหมาะสมกับสัญญาณนั้นๆเก็บไว้ในแถวคอยของระบบ (system queue) ตามหลักเข้าก่อนออกก่อน (first-in-first-out :FIFO)จากนั้นวินโดวส์จะพิจารณาว่าข้อความต่างๆนั้นเป็นของโปรแกรมประยุกต์ใดแล้วจึงส่งข้อความนั้นไปยังแถวคอยของโปรแกรมประยุกต์นั้นๆ โดยที่แต่ละโปรแกรมประยุกต์ก็จะมีแถวคอยของโปรแกรมประยุกต์(application queue) ของตนเองคอยรับข้อความที่ส่งมา จากนั้นโปรแกรมประยุกต์จึงนำข้อความที่ได้รับไปประมวลผลอีกทีหนึ่ง

*ดูรายละเอียดเพิ่มเติมได้จาก Microsoft Windows programmer's reference (Microsoft staff, 1990)



รูปที่ 4.2 แสดงการไหลของข้อความ

จากรูปที่ 4.2 เมื่อเกิดข้อความเช่นผู้ใช้กดแผงแป้นอักขระ วินโดวส์จะเก็บข้อความไว้ที่แถวคอยของระบบ และพิจารณาว่าเป็นข้อความของโปรแกรมประยุกต์ใด แล้วส่งไปเก็บไว้ที่แถวคอยของโปรแกรมประยุกต์นั้น จากนั้น โปรแกรมประยุกต์จะรับข้อความมาตรวจสอบว่าเป็นข้อความการหยุดโปรแกรมหรือไม่ ถ้าใช่ก็จะหยุดการทำงาน ของโปรแกรมนั้น แต่ถ้าไม่ใช่ก็จะจัดส่งกลับไปยังวินโดวส์เพื่อให้วินโดวส์จัดส่งไปยังวินโดว์ฟังก์ชันของโปรแกรม ประยุกต์ที่เหมาะสมเพื่อประมวลผลข้อความนั้นตามต้องการต่อไป

ลักษณะการทำงานของวินโดวส์โดยใช้ข้อความนี้เอง ทำให้วินโดวส์สามารถทำงานได้หลายงานพร้อมๆกัน โดยการสลับการทำงานของแต่ละโปรแกรมประยุกต์จะขึ้นอยู่กับข้อความกล่าวคือถ้าโปรแกรมประยุกต์ที่กำลังทำงานอยู่ ไม่มีข้อความในแถวคอย แต่มีอีกโปรแกรมประยุกต์ที่มีข้อความรออยู่ในแถวคอย วินโดวส์ก็จะสลับการทำงานไปยัง โปรแกรมประยุกต์หลังแทน ซึ่งการสลับการทำงานแบบนี้เรียกว่า jumpy multitasking หรือ nonpreemptive multitasking (Petzold, 1990)

<u>ลักษณะของโปรแกรม</u>

โปรแกรมที่เขียนขึ้นสำหรับการทำงานบนไมโครซอฟต์วินโดวส์นั้น มี 2 ลักษณะ ดังนี้

1. Executable Program (.EXE files)

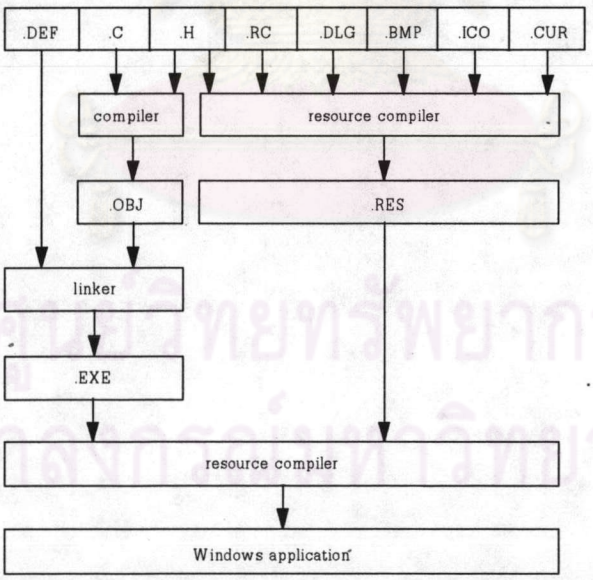
เป็นโปรแกรมที่สามารถทำงานได้ด้วยตนเองเช่นเดียวกับโปรแกรมต่างๆที่เขียนขึ้นบนระบบปฏิบัติการ ดอสทั่วๆไป

2. Dynamic Link Libraries (.DLL files)

เป็นโปรแกรมชนิดหนึ่งที่ไม่สามารถทำงานได้ด้วยตนเอง กล่าวคือโปรแกรมประเภทนี้จะเป็นการรวบรวมฟังก์ชันการทำงานต่างๆไว้ ให้โปรแกรมประเภทแรกเรียกใช้ ความแตกต่างที่สำคัญของโปรแกรมประเภทนี้ก็คือหน่วยความจำที่ใช้ เนื่องจากโปรแกรมที่ทำงานบนไมโครซอฟต์วินโดวส์สามารถทำงานได้พร้อมๆกัน วินโดวส์จะจัดสรรหน่วยความจำให้ทุกครั้งที่มีการเรียกใช้โปรแกรมประเภทแรก นั่นคือแต่ละโปรแกรมก็จะมีหน่วยความจำของตนเอง ในขณะที่โปรแกรมประเภทDLLนี้วินโดวส์จะจัดสรรหน่วยความจำให้เพียงชุดเดียวและถูกเรียกใช้จากโปรแกรมหลายๆโปรแกรมได้ ทำให้สามารถประหยัดหน่วยความจำ และขนาดของโปรแกรมแบบ .EXE นั้นก็จะมีขนาดเล็กลง อีกทั้งการแก้ไขฟังก์ชันการทำงานต่างๆก็ทำกับโปรแกรมประเภท .DLL นี้เพียงอย่างเดียว ไม่ต้องไปแก้ไขโปรแกรมแบบ .EXE อีก

<u>เทคนิคการเขียนโปรแกรม</u>

หลักการเขียนโปรแกรมเพื่อทำงานบนไมโครซอฟต์วินโดวส์อยู่บนพื้นฐานของการเขียนโปรแกรมแบบมัลติโมดูล (multimodule programming) กล่าวคือจะประกอบด้วยแฟ้มข้อมูลประเภทต่างๆที่ถูกแปลและนำมาเชื่อมต่อกันเป็นโปรแกรมประเภทที่สามารถทำงานได้ โดยแฟ้มข้อมูลต่างๆมีความสัมพันธ์กัน(ดูรูปที่ 4.3)ดังนี้



รูปที่ 4.3 แสดงส่วนประกอบและขั้นตอนการสร้างโปรแกรม

1. แฟ้มข้อมูลต้นฉบับ (source .C file)

เป็นแฟ้มข้อมูลที่เขียนขึ้นโดยใช้ภาษาคอมพิวเตอร์ เช่น ซี ปาสคาล(แต่ในที่นี้จะใช้ภาษาซีเป็นตัวอย่าง) ประกอบด้วยฟังก์ชันต่างๆตามขั้นตอนการทำงานที่กำหนดโดยโครงสร้างของแฟ้มข้อมูลจะต้องประกอบด้วยส่วนสำคัญดังนี้

1.1 การ include แฟ้มข้อมูล windows.h ซึ่งเป็นแฟ้มข้อมูลส่วนหัวที่จะขาดไม่ได้ เนื่องจากเป็นแฟ้มข้อมูลที่เก็บการประกาศฟังก์ชันต่างๆที่วินโดวส์เตรียมไว้ให้ใช้ ประเภทข้อมูล ค่าคงที่และโครงสร้างข้อมูลต่างๆ

1.2 ฟังก์ชัน WinMain เป็นฟังก์ชันที่เป็นจุดเริ่มต้นของการทำงานคล้ายกับฟังก์ชัน main ของโปรแกรมภาษาซีทั่วๆไป โดยจะต้องใช้ชื่อนี้เท่านั้นและการทำงานในฟังก์ชันนี้จะต้องประกอบด้วย

1.2.1 การลงทะเบียนวินโดว์คลาส (register window class) เป็นการกำหนดค่าให้กับตัวแปรโครงสร้างชนิด WNDCLASS เพื่อกำหนดรายละเอียดลักษณะพื้นฐานของวินโดว์ที่ต้องการสร้างในโปรแกรม เช่น ชื่อของ window procedure สัญรูปและตัวชี้ตำแหน่งของวินโดว์นั้น โดยการลงทะเบียนนี้จะกระทำเพียงครั้งแรกที่โปรแกรมถูกเรียกขึ้นมาทำงาน โดยถ้าก่อนหน้ามีโปรแกรมเดียวกันทำงานอยู่ก่อนแล้ววินโดวส์ก็จะไม่ทำการลงทะเบียนวินโดว์คลาสอีก ซึ่งเป็นผลดีทำให้ไม่เปลืองเนื้อที่หน่วยความจำ

1.2.2 การสร้างวินโดว์ (creating window) ในขั้นตอนนี้จะเป็นการสร้างวินโดว์หลักของโปรแกรม โดยจะมีการกำหนดชนิดของวินโดว์ที่ต้องการสร้าง ขนาดและตำแหน่งเริ่มต้นของวินโดว์ รายการเลือก การสร้างวินโดว์นี้จะเป็นการสร้างขึ้นภายในตัวไมโครซอฟต์วินโดวส์เท่านั้นยังไม่ถูกแสดงผลบนจอภาพจะต้องเรียกฟังก์ชัน ShowWindow และ UpdateWindow โดยฟังก์ชัน ShowWindow จะทำหน้าที่สร้างวินโดว์ตามลักษณะที่ต้องการบนหน้าจอ และฟังก์ชัน UpdateWindow จะทำหน้าที่ส่งข้อความ WM_PAINT ไปให้กับ window procedure เพื่อให้พื้นที่ภายในวินโดว์ถูกวาดตามที่โปรแกรมกำหนด

1.2.4 วังวนข้อความ (message loop) เป็นส่วนที่โปรแกรมจะเชื่อมต่อกับวินโดวส์ โดยจะเป็นการทำงานในลักษณะวนไปเรื่อยๆจนกว่าต้องการจะจบการทำงานของโปรแกรม ภายในวังวนจะมีฟังก์ชันการทำงานสองฟังก์ชันได้แก่ TranslateMessage ที่ทำหน้าที่แปลงข้อความเกี่ยวกับแผงแป้นอักขระให้มีค่าที่สามารถนำไปใช้งานได้ และ DispatchMessage ที่ทำหน้าที่ส่งข้อความกลับไปยังวินโดวส์เพื่อให้วินโดวส์จัดการส่งไปยัง window function ต่อไป

1.2.5 Window procedure ในส่วนนี้ถือว่าเป็นส่วนสำคัญของการเขียนโปรแกรมเพื่อทำงานบนไมโครซอฟต์วินโดวส์ เพราะว่าเป็นส่วนที่ทำหน้าที่ประมวลผลข้อความต่างๆที่ถูกส่งมา โดยชื่อของฟังก์ชันจะต้องกำหนดไว้ในขั้นตอนการลงทะเบียนวินโดว์คลาส เพื่อให้วินโดวส์รู้ว่าต้องส่งข้อความไปประมวลผลที่ฟังก์ชันใด โดยทั่วไปรูปแบบภายในของฟังก์ชันจะอยู่ในรูปของการใช้คำสั่ง switch - case statement เพื่อเลือกเอาข้อความไปประมวลผลเฉพาะตามต้องการ ส่วนข้อความที่ไม่ต้องการจะต้องถูกส่งไปที่ DefWindowProc ซึ่งเป็นฟังก์ชันภายในวินโดวส์ที่ถูกกำหนดให้ทำงานกับข้อความต่างๆที่ไม่ได้ถูกประมวลผล ซึ่งค่าที่ส่งกลับจาก DefWindowProc นี้จะถูกส่งกลับคืนสู่วินโดวส์อีกทีหนึ่ง

ตัวอย่างของแฟ้มข้อมูลแสดงได้ดังนี้

```
/* Example .C file */
#include <windows.h>

long FAR PASCAL WndProc(HWND, WORD, WORD, WORD);

int PASCAL WinMain(HANDLE hInstance, HANDLE hPrevInstance,
                   LPSTR lpszCmdParam, int nCmdShow)
{
HWND        hWnd;
MSG         msg;
WNDCLASS    wndclass;

if (!hPrevInstance)
    {
    wndclass.style          = CS_HREDRAW | CS_VREDRAW;
    wndclass.lpfnWndProc    = WndProc;
    wndclass.cbClsExtra  = 0;
    wndclass.cbWndExtra     = 0;
    wndclass.hInstance      = hInstance;
    wndclass.hIcon          = LoadIcon(NULL, IDI_APPLICATION);
    wndclass.hCursor      = LoadCursor(NULL, IDC_ARROW);
    wndclass.hbrBackground = GetStockObject(WHITE_BRUSH);
    wndclass.lpszMenuName = NULL;
    wndclass.lpszClassName = "Example";

    RegisterClass(&wndclass);
    }
hWnd = CreateWindow("Example", "Example source file",
                    WS_OVERLAPPEDWINDOW, CW_USEDEFAULT, CW_USEDEFAULT,
                    CW_USEDEFAULT, CW_USEDEFAULT,
                    NULL, NULL, hInstance, NULL);
ShowWindow(hWnd, nCmdShow);
UpdateWindow(hWnd);
while (GetMessage(&msg, NULL, 0, 0))
    {
    TranslateMessage(&msg);
    DispatchMessage(&msg);
    }
return msg.wParam;
}
long FAR PASCAL WndProc(HWND hWnd, WORD message, WORD wParam, LONG lParam)
{
HDC         hDC;
PAINTSTRUCT ps;

switch (message)
    {
    case WM_PAINT:
        hDC = BeginPaint(hWnd, &ps);
        EndPaint(hWnd, &ps);
        return 0;
    }
return DefWindowProc(hWnd, message, wParam, lParam);
```

}

2. Module definition file (.DEF file) เป็นแฟ้มข้อมูลที่ช่วยในขั้นตอนการเชื่อมต่อโปรแกรม โดยประกอบด้วยข้อมูลทางเทคนิคของโปรแกรมที่สร้างขึ้น เช่น ขนาดของสแตก ขนาดของหน่วยความจำขณะเริ่มต้นโปรแกรม ชื่อของโปรแกรมที่จะถูกเรียกให้ทำงาน คุณสมบัติของ code segment และ data segment รวมทั้งชื่อของ window procedure ทั้งหมด ตัวอย่างของ module definition file แสดงได้ดังนี้

```
/* example of .DEF file */
NAME          EXAMPLE
DESCRIPTION   'example of module definition file'
EXETYPE       WINDOWS
STUB          'WINSTUB.EXE'
CODE          PRELOAD MOVEABLE DISCARDABLE
DATA          PRELOAD MOVEABLE MULTIPLE
HEAPSIZE      1024
STACKSIZE     8192
EXPORTS       WndProc
```

3. แฟ้มข้อมูลส่วนหัว (.H file) เป็นแฟ้มข้อมูลส่วนหัวเช่นเดียวกับการเขียนโปรแกรมภาษาซีทั่วไป

4. Resource script file (.RC file) เป็นแฟ้มข้อมูลที่ใช้กำหนดการใช้ทรัพยากรต่างๆของโปรแกรม ประกอบด้วย การบรรยายการสร้างรายการเลือกและแป้นลัด ข้อความ ชื่อและชนิดของแฟ้มข้อมูลแบบแผนที่บิต แฟ้มข้อมูลสัญรูป แฟ้มข้อมูลตัวชี้ตำแหน่ง รวมทั้งแฟ้มข้อมูลกล่องคำโต้ตอบ(dialog box)ต่างๆ. ตัวอย่างของแฟ้มข้อมูลแสดงได้ดังนี้

```
/*  example of .RC file */
#include <windows.h>

cur1          CURSOR    cursor.cur
ico1          ICON      icon.ICO
bmp1          BITMAP    bmp.BMP
Example MENU
{
POPUP "&File"
              {
              MENUITEM "&New",    IDM_NEW
              MENUITEM "&Save",   IDM_SAVE
              MENUITEM "&Quit",   IDM_QUIT
              }
}
```

5. แฟ้มข้อมูลกล่องคำโต้ตอบ(.DLGfile)เป็นแฟ้มข้อมูลที่ใช้ในการบรรยายการสร้างกล่องคำโต้ตอบต่างๆ

6. แฟ้มข้อมูลแผนที่บิต (.BMP file) เป็นแฟ้มข้อมูลรูปภาพแบบแผนที่บิตที่ใช้ในโปรแกรม

7. แฟ้มข้อมูลสัญรูป (.ICO file) เป็นแฟ้มข้อมูลรูปภาพสัญรูปที่ใช้ในโปรแกรม

8. แฟ้มข้อมูลตัวชี้ตำแหน่ง (.CUR file) เป็นแฟ้มข้อมูลรูปภาพตัวชี้ตำแหน่งที่ใช้ในโปรแกรม

แฟ้มข้อมูลทั้งหมดนี้จะถูกแปลและนำมาเชื่อมต่อกัน จนกลายเป็นโปรแกรมที่สามารถทำงานบนไมโครซอฟต์วินโดวส์ได้ สำหรับรายละเอียดปลีกย่อยอื่นๆตลอดจนเทคนิคขั้นสูง สามารถศึกษาเพิ่มเติมได้จากคู่มือสำหรับการเขียนโปรแกรมเพื่อใช้งานบนไมโครซอฟต์วินโดวส์(Adam, 1992; Petzold, 1990; Microsoft Staff, 1990b, 1990c, 1990d; Norton and Yao, 1990; Rector, 1992; Richter, 1991)