รายการอ้างอิง

1 James Martin with Kathleen Kavanagh Chapman, Joe Leben. Enterprise Networking Strategies and Transport Protocols. New Jersey: Prentice Hall, 1996.

2 Heinz-Gerd Hegering. Integrated Network and System Management. Cambridge: Addison-Wesley Publishing, 1994.

3 Ward Rosenberry, David Kenneg, Gerry Fisher. Understanding DCE. Sebastopol, California: O'Reilly & Associates , 1992.

4 Karen Watterson. Client/Server technology for managers. California: Addison-Wesley Publishing, 1995.

5 Chris Corry, Vincent Mayfield, John Cadman and Randy Morin. COM/DCOM Primer Plus. Indianapolis, Indiana: Sams Publishing, 1999.

6 Lyn Robinson and K.David White. Database Programming with Visual C++6 in 21 Days. Indianapolis, Indiana: Sams Publishing, 1999.

7 Viktor Toth. Programming Window 98/NT Unleashed. Indianapolis, Indiana: Sams Publishing, 1998.

8 Ron Soukup and Kalen Delaney. Inside Microsoft SQL Server 7.0. Redmond, Washington: Microsoft Press, 1999.

9 Tim Darby with Lee Hadfield and Noelani Rodriguez. Special Edition Using Microsoft System Management Server. Indianapolis, Indiana: Que Corporation, 1996.

10 Intel Corporation. Intel LANDesk Management Suite. Hillsboro, Oragon: Intel Corporation, 1996.

11 Computer Associates Inc, CA-Unicenter for Window NT. New York: Computer Associates, 1995.

ภาคผนวก ก.
โครงสร้างตารางข้อมูล

ลำดับที่	ชื่อเขตข้อมูล	ประเภท	ขนาด	คำอธิบาย
1	MachineID	Number	4	หมายเลขเครื่องคอมพิวเตอร์ (PK)
2	ComputerName	Char	40	ชื่อเครื่องคอมพิวเตอร์
3	Domain	Char	40	ชื่อโดเมน (Domain)หรือกลุ่ม
4	OperatingSystemType	Char	4	ชนิดของระบบปฏิบัติการ
5	OSVersion	Char	20	เวอร์ชั่นของระบบปฏิบัติการ
6	SystemDirectory	Char	50	สารบบของระบบที่ใช้เก็บโปรแกรม
7	HostName	Char	40	ชื่อเครื่องแม่ข่าย
8	IPAddress	Char	20	หมายเลขไอพี (IP Address)
9	MacAddress	Char	40	หมายเลขของการ์ดอีเทอร์เน็ต
10	SWDAgentFlag	Number	4	Flag = 0 , ยังไม่ได้ติดตั้ง Agent แล้ว Flag = 1 , ติดตั้ง Agent แล้ว
11	DriveCSize	Number	4	ขนาดเนื้อที่ของพาร์ติชั่น (patition) Drive C
12	DriveCFSpace	Number	4	ขนาดเนื้อที่เหลือของพาร์ติชั่น Drive C
13	DriveDSize	Number	4	ขนาดเนื้อที่ของพาร์ติชั่น Drive D
14	DriveDFSpace	Number	4	ขนาดเนื้อที่เหลือของพาร์ติชั่น Drive D
15	DriveESize	Number	4	ขนาดเนื้อที่ของพาร์ติชั่น Drive E
16	DriveEFSpace	Number	4	ขนาดเนื้อที่เหลือของพาร์ติชั่น Drive E
17	MemSize	Number	4	ขนาดของหน่วยความจำหลัก
18	UserName	Char	20	ชื่อผู้ใช้ ที่เป็นคนสร้างระเบียน
19	CreateDate	Date	8	วันที่สร้างระเบียน
20	Remark	Char	255	คำอธิบายเพิ่ม

ตารางที่ ก-1 แสดงโครงสร้างข้อมูลรายละเอียดเครื่องคอมพิวเตอร์ (Inventory)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	OperatingSystemType	Number	4	หมายเลขประเภทระบบปฏิบัติการ (PK)
2	OperatingSystemName	Char	50	ชื่อระบบปฏิบัติการ

ตารางที่ ก-2 แสดงโครงสร้างข้อมูลรายละเอียดระบบปฏิบัติการ (Operating System)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	InstallType	Number	4	หมายเลขประเภทการติดตั้ง (PK)
2	InstallTypeName	Char	50	ชื่อประเภทโปรแกรมสำเร็จรูปแบบพีดีเอฟ (PDF)

ตารางที่ ก-3 แสดงโครงสร้างข้อมูลรายละเอียดประเภทโปรแกรมสำเร็จรูปแบบพีดีเอฟ (Package PDF Type)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	PdfID	Number	4	หมายเลขพีดีเอฟ (PK)
2	ArchitectureID	Number	4	หมายเลขสถาปัตยกรรม (PK)
3	InstallType	Number	4	หมายเลขประเภทการติดตั้ง (PK)
4	PdfDescription	Char	100	ชื่อโปรแกรมสำเร็จรูปแบบพีดีเอฟ
5	PdfVersion	Char	20	เวอร์ชั่นของโปรแกรมสำเร็จรูป
6	Vender	Char	40	ชื่อบริษัทที่สร้างโปรแกรมสำเร็จรูป
7	CommandExec	Char	50	ชื่อคำสั่งที่ใช้ในการติดตั้ง
8	WorkDirectory	Char	100	สถานที่เก็บคำสั่งที่ใช้ในการติดตั้ง
9	PdfFileName	Char	20	ชื่อแฟ้มข้อมูลโปรแกรมสำเร็จรูปแบบพีดีเอฟ
10	DiskSpace	Number	4	เนื้อที่ที่ใช้เก็บโปรแกรมสำเร็จรูป
11	UserName	Char	20	ชื่อผู้ใช้ที่สร้างระเบียน
12	CreateDate	Date	8	วันที่สร้างระเบียน
13	Remark	Char	255	คำอธิบายเพิ่มเติม

ตารางที่ ก-4 แสดงโครงสร้างข้อมูลรายละเอียดโปรแกรมสำเร็จรูปแบบพีดีเอฟ (Package PDF)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	ArchitectureID	Number	4	หมายเลขสถาปัตยกรรม (PK)
2	ArchitectureName	Char	255	ชื่อประเภทของสถาปัตยกรรม

ตารางที่ ก-5 แสดงโครงสร้างข้อมูลรายละเอียดสถาปัตยกรรมของคอมพิวเตอร์ (Architecture)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	PdfID	Number	4	หมายเลขพีดีเอฟ (PK)
2	ProductName	Char	50	ชื่อโปรแกรมสำเร็จรูปแบบพีดีเอฟ
3	Vender	Char	50	ชื่อบริษัทที่ผลิตโปรแกรมสำเร็จรูป

ตารางที่ ก-6 แสดงโครงสร้างข้อมูลรายละเอียดกลุ่มโปรแกรมสำเร็จรูปแบบพีดีเอฟ (PDF Group)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	PkgID	Number	4	หมายเลขโปรแกรมสำเร็จรูป (PK)
2	PkgName	Char	255	ชื่อโปรแกรมสำเร็จรูปที่จะใช้ในการกระจายการติดตั้ง
3	ProductName	Char	100	ชื่อโปรแกรมสำเร็จรูป
4	Version	Char	20	เวอร์ชั่นของโปรแกรมสำเร็จรูป
5	Vender	Char	40	ชื่อบริษัทที่ผลิตโปรแกรมสำเร็จรูป
6	PdfID	Number	4	หมายเลขพีดีเอฟ (FK)
7	ArchitectureID	Number	4	หมายเลขสถาปัตยกรรม (FK)
8	InstallType	Number	4	ประเภทการติดตั้ง (FK)
9	InstallMode	Number	4	โหมดที่ใช้ในการติดตั้ง 0 = โหมดปกติจะถามก่อนติดตั้ง 1 = โหมดการติดตั้งแบบทันทีทันใด
9	SourceDirectory	Char	255	ที่เก็บโปรแกรมสำเร็จรูป
10	UserName	Char	20	ชื่อผู้ใช้ที่สร้างระเบียน
11	CreateDate	Char	8	วันที่สร้างระเบียน
12	Remark	Char	255	คำอธิบายเพิ่มเติม

ตารางที่ ก-7 แสดงโครงสร้างข้อมูลรายละเอียดโปรแกรมสำเร็จรูป (Package)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	AdvertiseID	Number	4	หมายเลขการกระจายโปรแกรมสำเร็จรูป (PK)
2	PkgID	Number	4	หมายเลขโปรแกรมสำเร็จรูป (PK)
3	MachineID	Number	4	หมายเลขเครื่องคอมพิวเตอร์ (PK)
4	ServerTranFlag	Number	4	Flag ของการสร้างโปรแกรมสำเร็จรูปที่ฝั่งเครื่องให้บริการ 0 = ยังไม่ได้สร้าง 1 = สร้างแล้ว
5	ClientTranFlag	Number	4	Flag ของการส่งและจัดเตรียมโปรแกรมสำเร็จรูปที่ฝั่งรับบริการ 0 = ยังไม่ได้สร้าง 1 = สร้างแล้ว
6	PcmFlag	Number	4	Flag ของ การติดตั้งโปรแกรมที่ฝั่งรับบริการ 0 = ยังไม่ได้สร้าง 1 = สร้างแล้ว
7	PcmRetry	Number	4	จำนวนครั้งที่พยายามติดตั้งโปรแกรมสำเร็จรูป
8	UserName	Char	20	ชื่อผู้ใช้ ที่สร้างระเบียน
9	CreateDate	Char	8	วันที่สร้างระเบียน
10	LastAccess	Date	8	วันที่ใช้งานล่าสุด
11	StartSchedual	Date	8	วันที่เริ่มต้นในการกระจายโปรแกรม
12	EndSchedual	Date	8	วันที่สิ้นสุดในการกระจายโปรแกรม
13	SchedualFlag	Number	4	Flag ของการกระจายโปรแกรม 0 = อยู่ในช่วงของการกระจาย 1 = ไม่อยู่ในช่วงของการกระจาย
14	SourcePackage	Char	255	ที่เก็บโปรแกรมสำเร็จรูป
15	Remark	Char	255	คำอธิบายเพิ่มเติม

ตารางที่ ก-8 แสดงโครงสร้างข้อมูลการกระจายโปรแกรมสำเร็จรูปไปยังเครื่องคอมพิวเตอร์ที่ระบุ (AdvertisePackage Table)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	AdvertiseID	Number	4	หมายเลขการกระจายโปรแกรมสำเร็จรูป (PK)
2	AdvertiseName	Char	255	ชื่อการกระจายโปรแกรมสำเร็จรูป ที่ใช้ในการติดตั้ง

ตารางที่ ก-9 แสดงโครงสร้างข้อมูลการกระจายโปรแกรมสำเร็จรูป (Advertise)

ลำดับที่	ชื่อ Field	ประเภท	ขนาด	คำอธิบาย
1	MachineID	Number	4	หมายเลขเครื่องคอมพิวเตอร์ (PK)
2	PkgID	Number	4	หมายเลขโปรแกรมสำเร็จรูป (PK)
3	PkgName	Char	40	ชื่อโปรแกรมสำเร็จรูป
4	Version	Char	20	เวอร์ชั่นของโปรแกรมสำเร็จรูป
5	Vender	Char	40	ชื่อผู้ผลิตโปรแกรมสำเร็จรูป
6	UserName	Char	40	ชื่อผู้ใช้ที่สร้างระเบียน
7	InstallDate	Date	8	วันที่สร้างระเบียน
8	Remark	Char	255	คำอธิบายเพิ่มเติม

ตารางที่ ก-10 แสดงโครงสร้างข้อมูลการเก็บรายละเอียดโปรแกรมสำเร็จรูป (Software Inventory)

ภาคผนวก ข.
การใช้งานโปรแกรม

ก่อนที่จะใช้งานโปรแกรมระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย ผู้ควบคุมระบบจะต้องมีการติดตั้งส่วนประกอบโปรแกรมต่างๆ เพื่อที่จะทำให้โปรแกรมสามารถทำงานผ่านระบบเครือข่ายได้ โดยจะต้องมีการกำหนดระบบรักษาความปลอดภัยของข้อมูลผ่านเครือข่าย เพื่อระบุความสามารถการทำงานของออบเจ็กต์และซีโอเอ็มออบเจ็กต์ผ่านระบบเครือข่ายว่าอนุญาตให้ออบเจ็กต์ใดบ้างสามารถทำงานหรือถูกสั่งให้ทำงานผ่านเครือข่ายได้

สิ่งที่ต้องเตรียมก่อนการใช้งานคือ การจัดหาโปรแกรมไมโครซอฟท์ *ซีโอเอ็ม/ดีซีโอเอ็ม* (COM/DCOM ย่อมาจาก Component Object Model/ Distribute Component Obejct Model) สำหรับเครื่องลูกข่ายหรือเครื่องรับบริการ ซึ่งจะทำให้ระบบสามารถเรียกใช้หรือสั่งงานซีโอเอ็มออบเจ็กต์ผ่านเครือข่ายได้

การติดตั้งโปรแกรม ผู้ใช้ต้องนำคลาสต่างๆ ของโปรแกรมมาเก็บไว้ที่เครื่องรับบริการ จากนั้นให้สั่งคำสั่งติดตั้งซีโอเอ็มออบเจ็กต์ในค่ารีจิสตรี (Registry) เป็นการลงทะเบียนของซีโอเอ็มออบเจ็กต์ เพื่อประกาศให้ระบบปฏิบัติการวินโดวส์ รู้จักและสามารถเรียกใช้งานส่วนประกอบของซีโอเอ็มออบเจ็กต์ได้อย่างถูกต้อง เมื่อติดตั้งโปรแกรมเรียบร้อยผู้ใช้หรือผู้ควบคุมระบบจะสามารถใช้งานได้อย่างมีประสิทธิภาพ

สิ่งที่ต้องเตรียมก่อนใช้งานโปรแกรม

ระบบที่จะใช้กับโปรแกรมควรเป็นดังนี้

1. ระบบปฏิบัติการไมโครซอฟท์วินโดวส์ 95/98 หรือเอ็นที (MS Windows 95/98 or NT)
2. ติดตั้งโปรแกรมไมโครซอฟท์เอ็กโพลเลอร์ เวอร์ชั่น 4 ขึ้นไป หรือติดตั้งโปรแกรมไมโครซอฟท์ *ซีโอเอ็ม / ดีซีโอเอ็ม* สำหรับระบบปฏิบัติการวินโดวส์ 95/98 หรือเอ็นที
3. ติดตั้งโปรแกรมไลบรารี่ (Library) ต่างๆที่จะถูกเรียกใช้ในภาษาซีพลัสพลัส ในสารบบหลักของระบบปฏิบัติการวินโดวส์เช่น c:\windows\system เป็นต้น
4. เฉพาะเครื่องแม่ข่ายจะต้องติดตั้งโปรแกรมไมโครซอฟท์ โอดีบีซี ไดร์เวอร์ (OBDC Driver ย่อมาจาก Open Database Connectivity Driver)
5. เพื่อใช้ในการติดต่อกับระบบฐานข้อมูลเชิงสัมพันธ์

การติดตั้งโปรแกรมระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย

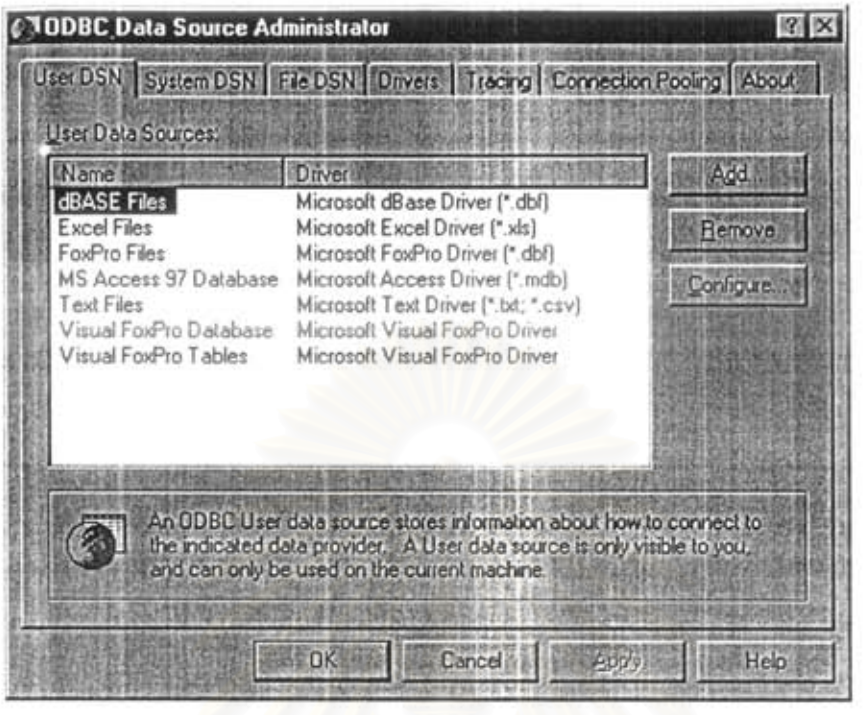
การติดตั้งโปรแกรมนั้นเนื่องจากการใช้งานจะเป็นการสั่งงานซีโอเอ็มออบเจ็กต์ผ่านเครือข่าย ต้องอาศัยการทำงานของซีโอเอ็มและดีซีโอเอ็ม ซึ่งจะต้องมีการกำหนดระบบรักษาความปลอดภัยเพื่อให้สามารถทำงานผ่านเครือข่ายและต้องมีการติดต่อและใช้งานระบบฐานข้อมูล ดังนั้นจะสามารถแบ่งการติดตั้งโปรแกรมออกเป็นส่วนๆ ดังต่อไปนี้

1. การติดตั้งส่วนของโปรแกรมในการติดตั้งระบบฐานข้อมูลที่เครื่องให้บริการ
2. การติดตั้งโปรแกรมระบบบริหารข้อมูลที่เครื่องให้บริการและเครื่องรับบริการ
3. การติดตั้งระบบการรักษาความปลอดภัยของซีโอเอ็มออบเจ็กต์ผ่ายเครือข่าย

1. การติดตั้งส่วนของโปรแกรมในการติดตั้งระบบฐานข้อมูลที่เครื่องให้บริการ

ในการติดตั้งส่วนนี้จะกระทำที่เครื่องแม่ข่ายหรือเครื่องให้บริการเท่านั้น ซึ่งจะรับผิดชอบในการติดต่อกับระบบฐานข้อมูลทั้งหมดที่เครื่องแม่ข่าย โดยระบบจะต้องติดตั้งส่วนนี้ก่อนเพื่อที่จะกำหนดสิทธิการใช้งานของผู้ใช้ในฐานข้อมูล และกำหนดชื่อฐานข้อมูลที่ใช้และเครื่องที่ติดตั้งระบบฐานข้อมูล ซึ่งจะสามารถทำได้โดยเรียกใช้โปรแกรมแอบเพล็ตโอดีบีซี (ODBC) 32 จากคอนโทรลพาเนล (Control Panel) ของระบบปฏิบัติการวินโดวส์เอ็นที ดังรูปที่ ข-1

รูปที่ ข-1 แสดงหน้าจอของคอนโทรลพาเนล

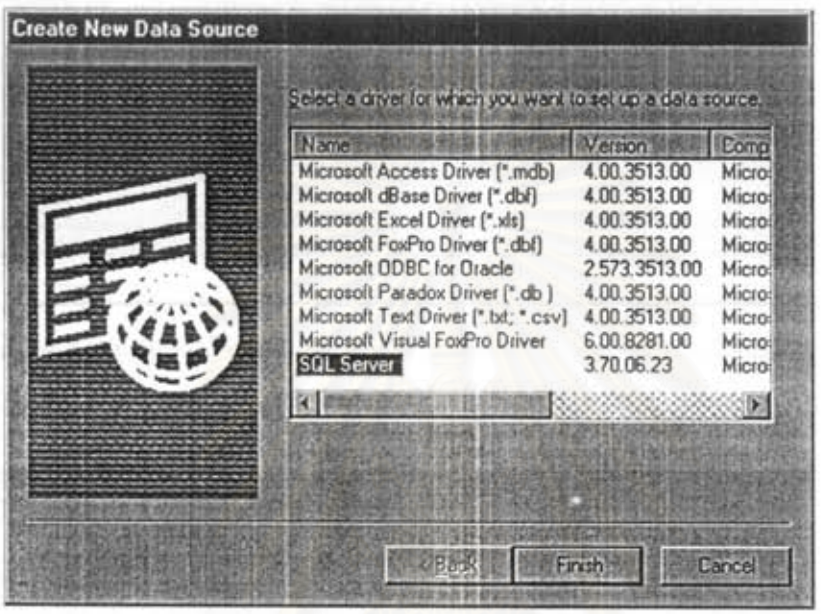


รูปที่ ข-2 แสดงหน้าจอโปรแกรมโอดีบีซี 32

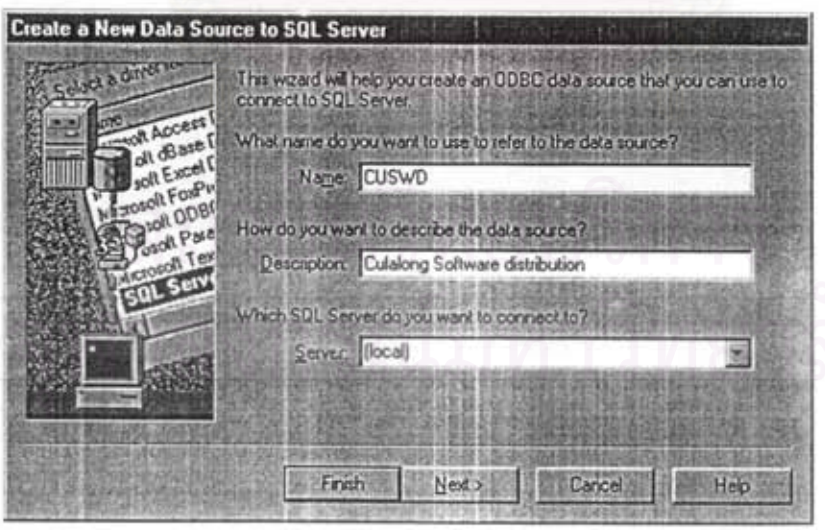
เมื่อสั่งให้โปรแกรมโอดีบีซี 32 ทำงานจะแสดงหน้าจอ ซึ่งผู้ใช้สามารถสร้าง แก้ไข และลบแหล่งกำเนิดฐานข้อมูลโอดีบีซี ดังรูปที่ ข-2 โดยแหล่งกำเนิดฐานข้อมูลโอดีบีซี จะถูกเรียกว่าดีเอสเอ็น (DSN ย่อมาจาก Data Source Name) ซึ่งในกรณีนี้จะมีดีเอสเอ็น 3 แบบคือ ดีเอสเอ็นของผู้ใช้ (User DSN) ซึ่งจะถูกใช้ ณ เครื่องคอมพิวเตอร์เท่านั้นและสามารถใช้งานได้เฉพาะผู้ใช้ที่ใช้งานอยู่ในปัจจุบันเท่านั้น ส่วนดีเอสเอ็นแบบระบบ (System DSN) จะเป็นการใช้งาน ณ เครื่องคอมพิวเตอร์ปัจจุบันซึ่งจะถูกใช้จากหลายผู้ใช้ ส่วนดีเอสเอ็นแบบสุดท้ายคือแบบแฟ้มข้อมูล (File DSN) ซึ่งสามารถเข้า ณ เครื่องคอมพิวเตอร์ปัจจุบันหรือแบบระยะไกลได้ ซึ่งจะถูกใช้จากหลายผู้ใช้ในที่นี้จะใช้ดีเอ็นเอสแบบระบบ โดยให้เลือกดีเอ็นเอสแบบระบบ จากนั้นให้กดปุ่ม Add เพื่อสร้างดีเอ็นเอสแบบระบบดังรูปที่ ข-3

ให้พิมพ์ชื่อของแหล่งข้อมูลเช่น CUSWD ดังรูปที่ ข-4 ซึ่งผู้ใช้สามารถเขียนคำอธิบายความหมายของแหล่งกำเนิดฐานข้อมูลที่ใช้ ให้กำหนดที่เครื่องให้บริการที่ติดตั้งโปรแกรมเอสคิวเอล เป็นแบบโลคอล(local) จากนั้นกดปุ่ม Next จะแสดงดังรูปที่ ข-5 จะเป็นการกำหนดการเข้าถึงฐานข้อมูล โดยให้เลือกเป็นการเข้าถึงแบบผู้ใช้ของโปรแกรมเอสคิวเอล โดยป้อนข้อมูลผู้ใช้เท่ากับ cusa และป้อนรหัสผ่านเท่ากับ cusaa จากนั้นกดปุ่ม Next เครื่องจะทำการติดต่อกับระบบฐานข้อมูล และแสดง

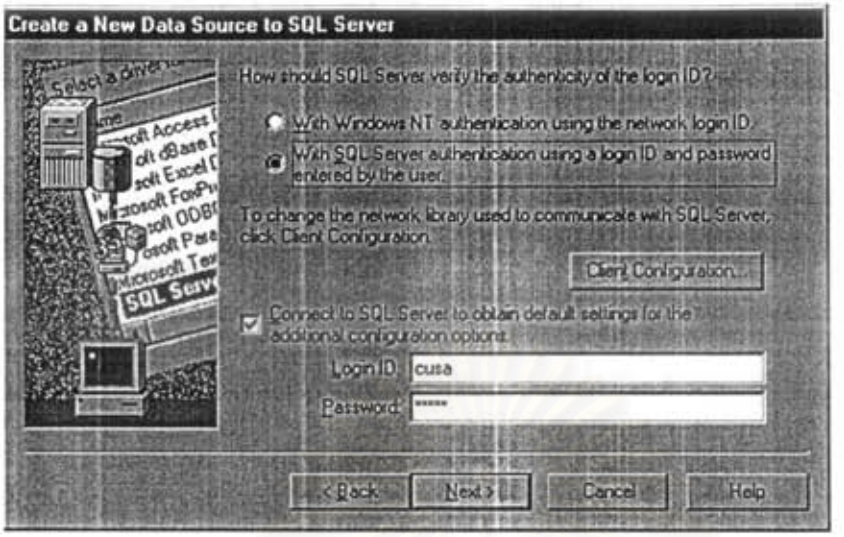
ดังรูปที่ ข-6 ให้ระบุฐานข้อมูลที่ใช้ในการติดต่อชื่อ cuswd จากนั้นให้กดปุ่ม Next และกดปุ่ม Finish และกดปุ่ม OK เพื่อสร้างแหล่งกำเนิดฐานข้อมูลโอดีบีซี ซึ่งหลังจากที่สร้างเรียบร้อยแล้วแล้ว เราสามารถใช้ดีเอสเอ็นระบบเป็นตัวติดต่อกับฐานข้อมูลได้จากภายในโปรแกรมระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย



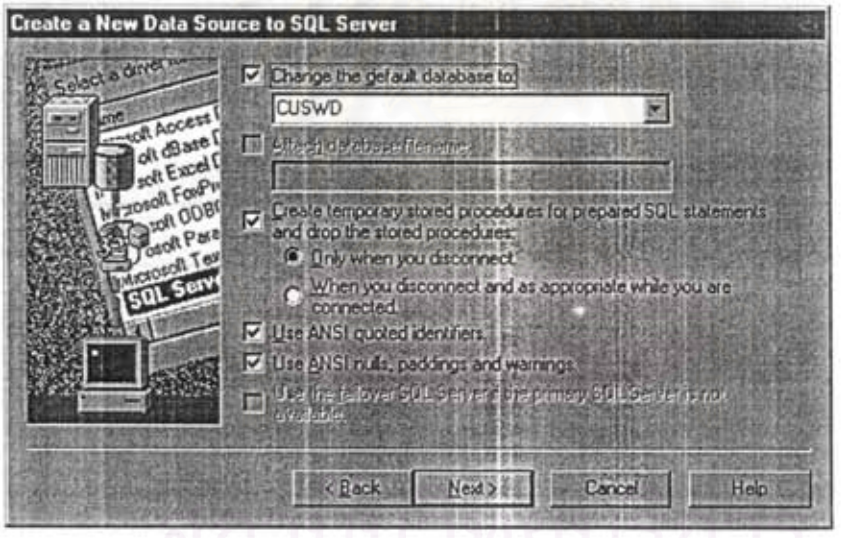
รูปที่ ข-3 แสดงการเลือกโอดีบีซี ไดร์เวอร์



รูปที่ ข-4 แสดงการติดตั้งดีเอสเอ็นจากโปรแกรมไมโครซอฟท์ เอสคิวเอล



รูปที่ ข-5 แสดงการกำหนดการเข้าถึงฐานข้อมูลไมโครซอฟท์ เอสคิวเอล



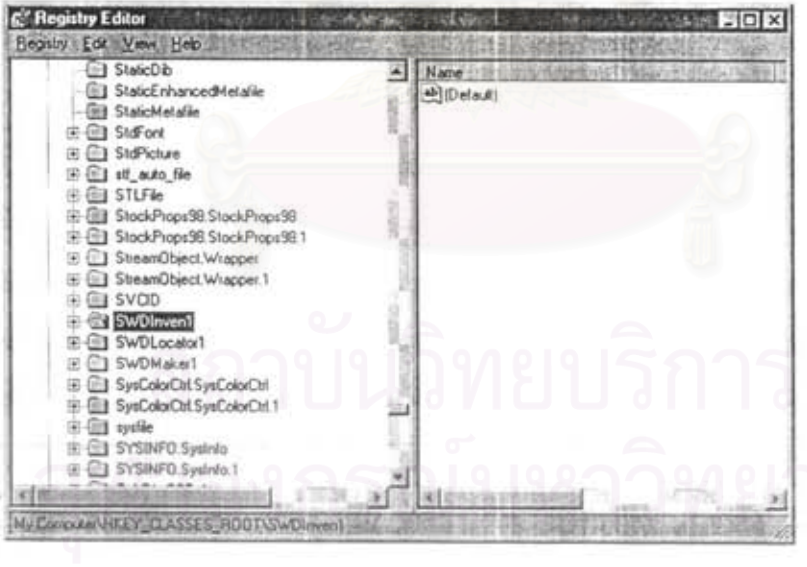
รูปที่ ข-6 แสดงการเลือกฐานข้อมูลที่ใช้ในการติดต่อฐานข้อมูล ไมโครซอฟท์ เอสคิวเอล

2. การติดตั้งโปรแกรมระบบบริหารข้อมูลที่เครื่องให้บริการและเครื่องรับบริการ

การติดตั้งโปรแกรมระบบไม่ว่าจะเป็นการเรียกโปรแกรมผ่านเครื่องให้บริการและรับบริการสามารถทำได้โดยการติดตั้งแฟ้มข้อมูลภายในสารบบเดียวกัน เช่น c:\windows\cuswd โดยจะแบ่งสารบบย่อยๆดังต่อไปนี้

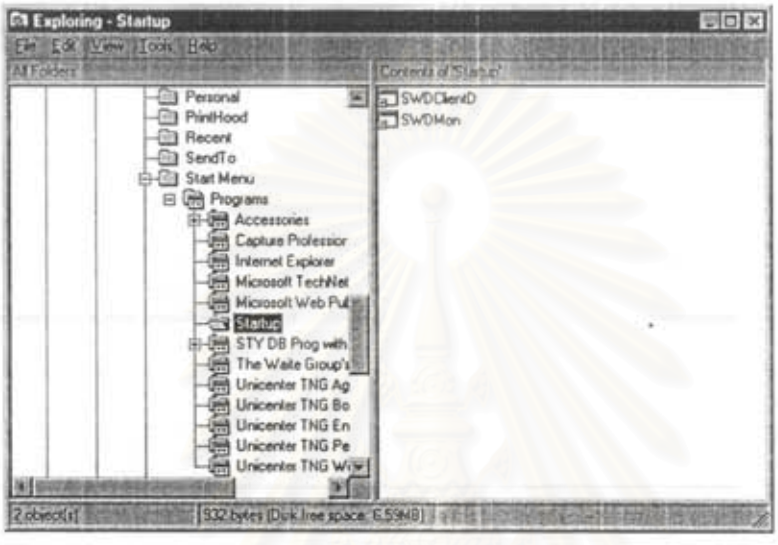
- Log เป็นสารบบที่เก็บแฟ้มข้อมูลที่เก็บรายละเอียดการทำงานต่างๆ ที่ทำเสร็จสิ้นไปแล้ว
- Request เป็นสารบบที่เก็บคำสั่งในการติดตั้งโปรแกรมสำเร็จรูปที่ต้องการทำงาน ณ เครื่องคอมพิวเตอร์ที่ใช้งาน
- Bin เป็นสารบบที่เก็บคำสั่งต่างๆที่ใช้งานในระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย
- Tmp เป็นสารบบที่ใช้งานชั่วคราวในขณะทำงาน

หลังจากที่โปรแกรมเสร็จแล้วขั้นตอนต่อไปคือการติดตั้งซีโอเอ็มออบเจ็กต์เข้าไปในรีจิสตรี เพื่อให้ระบบปฏิบัติการวินโดวส์ รู้จักและสามารถเรียกใช้งานได้อย่างถูกต้อง โดยสามารถติดตั้งโดยใช้คำสั่ง expdll9x.bat สำหรับระบบปฏิบัติการวินโดวส์ 95/98 หรือคำสั่ง expdllnt.bat สำหรับระบบปฏิบัติการวินโดวส์เอ็นที หลังจากเรียกใช้โปรแกรมเสร็จเรียบร้อยแล้ว เราสามารถตรวจสอบการติดตั้งซีโอเอ็มออบเจ็กต์ของระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่ายโดยใช้คำสั่ง REGEDIT.EXE แสดงดังรูปที่ ข-7



รูปที่ ข–7 แสดงการตรวจสอบซีโอเอ็มออบเจ็กต์โดยใช้โปรแกรม REGEDIT.EXE

หลังจากที่ติดตั้งซีโอเอ็มออบเจ็กต์ในรีจิสตรีของระบบปฏิบัติการวินโดวส์เรียบร้อยแล้วเพื่อให้ระบบทำงานทุกครั้งที่เปิดเครื่องคอมพิวเตอร์ให้ติดตั้งโปรแกรม SWDMON.EXE และ PCMDAEMON.EXE ที่ startup ดังรูปที่ ข-8

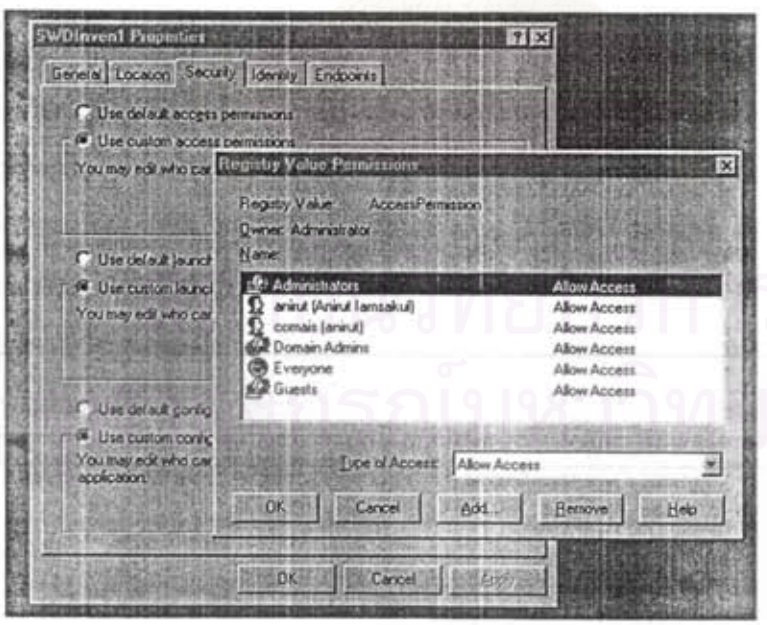


รูปที่ ข-8 แสดงหน้าจอการติดตั้งโปรแกรมทำงานแบบอัตโนมัติ

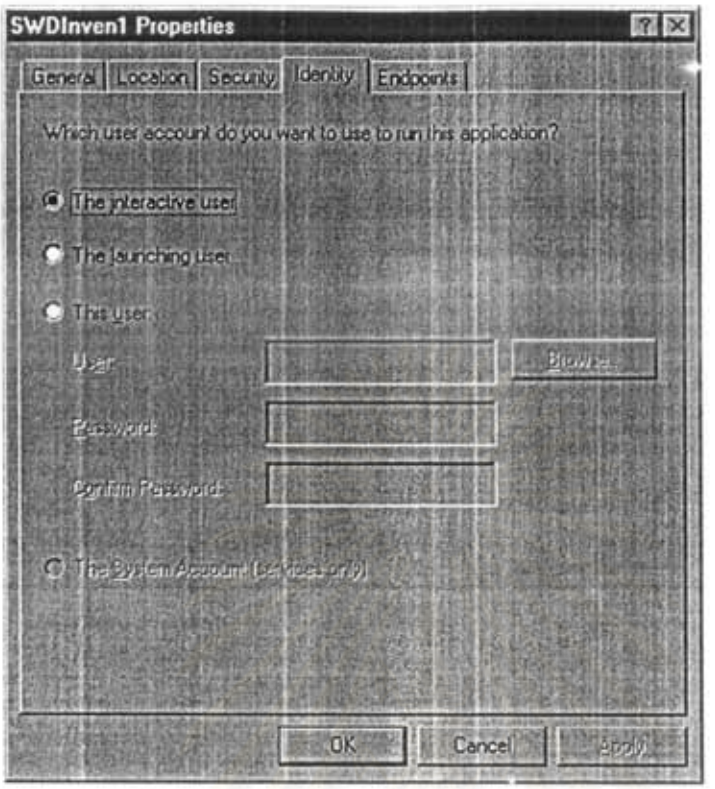
3. การติดตั้งระบบการรักษาความปลอดภัยของซีโอเอ็มออบเจ็กต์ผ่ายเครือข่าย

หลังจากที่ติดตั้งโปรแกรมทุกอย่างเรียบร้อยแล้ว ถ้าผู้ใช้ทดลองใช้งานโปรแกรมจะพบปัญหาว่า ผู้ใช้ไม่สามารถใช้งานระบบบริหารข้อมูลผ่านเครือข่ายไปยังเครื่องรับบริการได้ จะสามารถทำงานได้เฉพาะเครื่องให้บริการเท่านั้น เพราะผู้ใช้ยังไม่ได้กำหนดการติดตั้งระบบรักษาความปลอดภัยของ ซีโอเอ็มออบเจ็กต์ผ่านเครือข่าย ซึ่งสามารถติดตั้งโดยใช้คำสั่ง DCOMCNFG.EXE แสดงดังรูปที่ ข-9 ซึ่งจะแสดงซีโอเอ็มออบเจ็กต์ที่ติดตั้งไปแล้วที่ระบบปฏิบัติการวินโดวส์ ในระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่ายจะมีซีโอเอ็มออบเจ็กต์ที่ต้องกำหนดการติดตั้งระบบรักษาความปลอดภัยอยู่ 3 ซีโอเอ็มออบเจ็กต์คือ SWDMAKE SWDINVENT และ SWDLOCATOR โดยจะติดตั้งทีละซีโอเอ็มออบเจ็กต์ โดยการติดตั้งต้องกำหนดส่วนที่เกี่ยวข้องคือการรักษาความปลอดภัยและลักษณะการตรวจสอบบัญชีผู้ใช้ แสดงดังรูปที่ ข-10 และรูปที่ ข-11 ซึ่งเมื่อมีการกำหนดระบบรักษาความปลอดภัยเรียบร้อยแล้วจะต้องทำทุกเครื่องที่จะใช้ระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่ายก็จะสามารถทำงานได้ถูกต้อง

รูปที่ ข-9 แสดงการทำงานของโปรแกรม DCOMCNFG.EXE



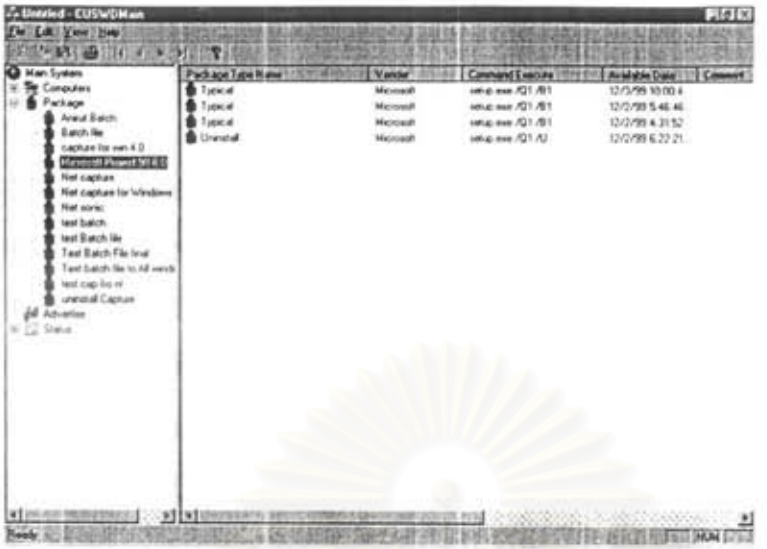
รูปที่ ข-10 แสดงการกำหนดการติดตั้งรักษาความปลอดภัย



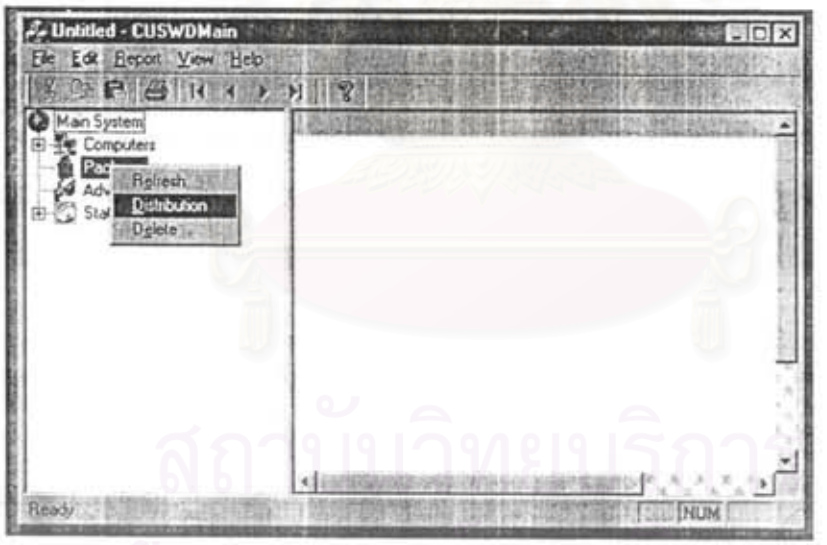
รูปที่ ข-11 แสดงการกำหนดลักษณะการตรวจสอบบัญชีผู้ใช้

การใช้งานโปรแกรม

เครื่องคอมพิวเตอร์ฝ่ายรับบริการ การทำงานส่วนใหญ่จะไม่มีหน้าจอในการใช้งานแต่จะเก็บการทำงานเบื้องหลัง จะทำงานตั้งแต่ผู้ใช้เข้าสู่ระบบ (Login) โดยจะทำการติดต่อกับเครื่องคอมพิวเตอร์ฝั่งให้บริการเพื่อเก็บรายละเอียดของคอมพิวเตอร์ไปเก็บยังฐานข้อมูลที่ฝั่งเครื่องให้บริการ ซึ่งการทำงานจะแบ่งออกเป็น 2 ส่วนคือ ส่วนที่อยู่เบื้องหลัง ได้แก่คอมออบเจ็กต์ต่างๆที่กล่าวไปแล้วในบทที่ 4 และส่วนที่ใช้ในการเตรียมและติดตั้งโปรแกรมสำเร็จรูปและสอบถามสถานะการติดตั้งโปรแกรมสำเร็จรูปซึ่งได้แสดงไปแล้วในบทที่ 5 โดยสามารถเรียกคำสั่ง SWDMAIN.EXE ในการเรียกใช้คำสั่งซึ่งจะเป็นการตรวจสอบรายละเอียดคอมพิวเตอร์ตามระบบปฏิบัติการ การตรวจสอบรายละเอียดโปรแกรมสำเร็จรูป การตรวจสอบสถานะการกระจายและการติดตั้งโปรแกรมสำเร็จรูป ดังรูป ข-12 และเราสามารถติดตั้งโปรแกรมสำเร็จรูปได้โดยการกดปุ่มเมาส์ขวาและเลือก Distribution เพื่อเป็นการเตรียมการกระจายการติดตั้งโปรแกรมสำเร็จรูปแสดงดังรูป ข-13 และหน้าจอผลลัพธ์ได้กล่าวไว้แล้วในบทที่ 4 ดังรูปที่ 4.3 ถึง 4.14 และหน้าจอในการติดตั้งแสดงดังรูปที่ 4.15 และ 4.16 ส่วนหน้าจอการสอบถามการติดตั้งโปรแกรมสำเร็จรูปได้กล่าวไว้ในบทที่ 5 แสดงดังรูป 5-1 ถึง 5-6



รูปที่ ข-12 แสดงหน้าจอโปรแกรมระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย



รูปที่ ข-13 แสดงหน้าจอการเตรียมการกระจายโปรแกรมสำเร็จรูป

ภาคผนวก ค
การเปรียบเทียบโปรแกรม

การเปรียบเทียบโปรแกรมทั้งสามโปรแกรมคือโปรแกรมอินเทล แลนเดส โปรแกรมไมโครซอฟท์ เอสเอ็มเอส และโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ ซึ่งทำงานร่วมกับโปรแกรมเอเจ็น ระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย ซึ่งจะแสดงให้เห็นความสามารถของโปรแกรมทั้งสามโปรแกรมที่จัดเป็นโปรแกรมที่ใช้ในการบริหารระบบที่นิยมใช้ในแวดวงธุรกิจ ซึ่งมีความสามารถในการตรวจสอบ ควบคุม และรายงานผลการทำงานของระบบเครือข่าย ตลอดจนใช้เป็นเครื่องมือในการแก้ไขปัญหาเมื่อมีสิ่งผิดปกติเกิดขึ้นในเครือข่าย

เนื่องจากผลการทดสอบการเปรียบเทียบที่กล่าวไว้ในบทที่ 5 และ 6 ซึ่งสรุปผลการเปรียบเทียบของโปรแกรมทั้งสามโปรแกรมไม่ได้ เพราะโปรแกรมเอเจ็นที่พัฒนาขึ้นในวิทยานิพนธ์นี้คือระบบบริหารข้อมูลในฮาร์ดิสก์ผ่านเครือข่ายทำงานได้โดยสมบูรณ์ โดยไม่ต้องอาศัยการทำงานร่วมกับโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ แต่จะช่วยให้การทำงานของโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ทำงานได้สมบูรณ์ครบถ้วนขึ้นในส่วนที่ใช้ในการบริหารระบบ แต่ไม่ได้ให้การทำงานของระบบเร็วขึ้น ทำให้ไม่สามารถเปรียบเทียบเวลาการทำงานกับอีกสองโปรแกรมได้ เพราะถ้าจะเปรียบเทียบโปรแกรมเอเจ็นกับอีกสองโปรแกรม ความสามารถของโปรแกรมก็ไม่เท่ากัน ดังนั้นก็ไม่สามารถสรุปการเปรียบเทียบได้

การเปรียบเทียบในบทนี้จะแสดงการเปรียบเทียบความสามารถทำงานทั่วๆไป โดยจะเปรียบเทียบโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ ซึ่งจะมีการทำงานร่วมกับโปรแกรมเอเจ็นที่พัฒนาขึ้นเอง กับโปรแกรมอินเทลแลนเดส และโปรแกรมไมโครซอฟท์เอสเอ็มเอส แสดงในตารางที่ ค-1 และ ค-2

ชื่อโปรแกรม	หน่วยประมวลผล	หน่วยความจำหลัก	ฮาร์ดดิสก์	ระบบปฏิบัติการเครือข่าย
TNG	Pentium Up	64	300 MB	Window NT, UNIX
SMS	Pentium Up	64	400 MB	Window NT, NetWare, Lan Manager, Lan Server
Intel LANDesk	Pentium Up	32	350 MB	Window NT, NetWare

ตารางที่ ค-1 แสดงการเปรียบเทียบสิ่งแวดล้อมของซอฟต์แวร์ที่ใช้

ความสามารถ	LANDesk Management Suite Version 6.1	Microsoft SMS Version 2.0	TNG Unicenter Version 2.1 (Standard) + (Agent ที่พัฒนาขึ้นเอง)
1. HW & SW Inventory	/	/	/ (limited)
2. SW Distribution - วิธีการ	/ (limited) PDF	/ Snapshot	/ (limited) PDF
3. Desktop Configuration	/	/	/
4. Remote Control	/	/	(option)
5. WAN Enable	/	/	/
6. Alarm & Event Management	/ (alarm only)	/ (alarm only)	/
7. Multi – Site	/	/	/
8. Remote Management	/	/	(option)
9. Web Interface	/	/	(option)
10. Performance management	/ (limited)	/	/
11. Network management	/	/	/
12. Anti Virus	/	/	(option)
13. Report Management	/	/	/
14. SDK (Agent)	-	-	/
15. Workload Management	/ (limited)	/ (limited)	/
16. Database Support	/	/	/
17. Software Metering	/	/	(option)
18. Full Object GUI (รวมถึงอุปกรณ์เครือข่าย)	-	-	/

ตารางที่ ค-2 แสดงการเปรียบเทียบความสามารถของโปรแกรมที่ทดสอบ

จากตารางที่ ค-2 สามารถเปรียบเทียบความสามารถและการใช้งานของโปรแกรมที่ทดสอบ โดยอาศัยส่วนต่าง ๆ ประกอบดังต่อไปนี้

1. การเก็บรายละเอียดของฮาร์ดแวร์และซอฟต์แวร์ (Hardware and Software Inventory) เพื่อให้ผู้บริหารระบบสามารถติดตามโครงแบบปัจจุบันของเครื่องให้บริการและเครื่องรับบริการ สภาวะของระบบปฏิบัติการและค่าพารามิเตอร์ต่าง ๆ ถ้ามีพารามิเตอร์มากจะสามารถส่งข้อมูลได้มาก ค้นหาข้อมูลได้แม่นยำและแสดงรายงานผลออกมาในรูปแบบที่อ่านเข้าใจง่าย
2. การกระจายและปรับปรุงซอฟต์แวร์ (Software distribution) เพื่อให้ผู้บริหารระบบลดเวลาการดูแลระบบโดยผ่านซอฟต์แวร์จัดการระบบ ในส่วนของการจัดการระบบนั้นซอฟต์แวร์จะต้องมีความสามารถในการกระจายการติดตั้งได้ง่าย มีความยืดหยุ่นในการจัดการซอฟต์แวร์ และจัดตารางการทำงาน (Schedule) ได้อย่างมีประสิทธิภาพ
3. การกำหนดโครงแบบของเครื่องคอมพิวเตอร์ (Desktop Configuration) เป็นการกำหนดความสามารถในการกำหนดโครงแบบและการใช้งานบนเครื่องคอมพิวเตอร์
4. การควบคุมการทำงานจากเครือข่ายระยะไกล (Remote Control) เป็นความสามารถในการควบคุมการทำงานของเครื่องรับบริการโดยสามารถถูกกระทำที่เครื่องให้บริการ แต่สามารถควบคุมการใช้งานต่างๆผ่านเครือข่ายระยะไกลได้
5. การทำงานผ่านเครือข่ายระยะไกล (WAN Enable) เป็นความสามารถในการสนับสนุนการใช้งานของโปรแกรมผ่านอุปกรณ์เครือข่ายระยะไกล
6. การแจ้งเหตุการณ์และการบริหารเหตุการณ์ (Alarm and Event management) เป็นการนำเสนอความสามารถในการแจ้งเหตุการณ์เมื่อมีสิ่งปกติเกิดขึ้น และความสามารถในการตอบสนองเหตุการณ์ที่เกิดขึ้นได้
7. การทำงานแบบหลายศูนย์รวม (Multi – Site) เป็นความสามารถในการทำงานเมื่อมีระบบการทำงานแบบหลายศูนย์
8. การควบคุมจากเครือข่ายใยแมงมุม (Web interface) เป็นความสามารถในการทำงานผ่านเครือข่ายใยแมงมุม
9. การบริหารประสิทธิภาพ (Performance Management) เป็นความสามารถในการบริหารประสิทธิภาพการทำงานของเครื่องคอมพิวเตอร์ต่างๆ

10. การบริหารเครือข่าย (Network Management) เป็นความสามารถในการควบคุมและดูแลการทำงานของเครือข่ายรวมถึงส่วนประกอบของเครือข่ายให้สามารถทำงานได้อย่างมีประสิทธิภาพ
11. การตรวจสอบไวรัส (Anti Virus) เป็นความสามารถในการตรวจสอบไวรัสและแก้ไขเมื่อมีการติดไวรัสในเครือข่าย
12. การบริหารรายงาน (Report Management) เป็นความสามารถในการจัดทำรายงานเพื่อสนับสนุนการทำงานให้มีประสิทธิภาพ
13. การสนับสนุนการพัฒนาซอฟต์แวร์ (Software Development Kit) เป็นการนำเสนอความสามารถในการพัฒนาโปรแกรม เพื่อเชื่อมต่อกับโปรแกรมหรือส่วนของโปรแกรมให้มีประสิทธิภาพมากขึ้น
14. การบริหารการทำงานของโปรแกรม (Workload Management) เป็นความสามารถในการบริหารการทำงานของโปรแกรมต่างๆของเครื่องคอมพิวเตอร์ให้สามารถทำงานได้อย่างมีประสิทธิภาพ
15. การสนับสนุนการติดต่อระบบฐานข้อมูล (Database Support) เป็นความสามารถในการติดต่อกับระบบฐานข้อมูลต่างๆ
16. การเชื่อมต่อวัตถุแบบจียูไอ (Object GUI / GUI ย่อมาจาก Graphic User Interface) เป็นความสามารถในการติดต่อและแสดงผลเป็นรูปภาพ
17. การจำกัดการใช้งานร่วมกัน (Software Metering) เป็นความสามารถในการให้ผู้บริหารระบบติดตามหาจำนวนโปรแกรมที่ติดตั้งในเครือข่ายรวมถึงการจำกัดจำนวนการใช้งานร่วมกัน ซึ่งจะมีประโยชน์มากในกรณีที่ ผู้บริหารระบบสนใจปัญหาการละเมิดข้อตกลงในการใช้ซอฟต์แวร์

จากการทดสอบและเปรียบเทียบดังตารางที่ ค-1 และ ค-2 แสดงให้เห็นเพื่อเป็นการเปรียบเทียบให้เห็นประสิทธิภาพเท่านั้นซึ่งพอจะสรุปได้ดังนี้

1. การติดตั้งโปรแกรมสำเร็จรูปโดยผ่านโปรแกรมอินเทล แลนเดสจะสามารถทำงานได้ดีที่สุด มีความยืดหยุ่นในการติดตั้งมากกว่าเพราะใช้เทคนิคสแนบช็อต (Snap Shot) ซึ่งเป็นการจดจำการติดตั้งทั้งหมดในการติดตั้ง ซึ่งไม่ต้องอาศัยการติดตั้งแบบแฟ้ม ข้อมูลพีดีเอฟแบบโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์และโปรแกรมไมโครซอฟท์ เอสเอ็มเอส

2. การทำงานในส่วนของการบริหารเหตุการณ์และรายงานเหตุการณ์ โปรแกรมทีเอ็นจี สามารถทำงานได้ครบถ้วนและมีประสิทธิภาพมากกว่าโปรแกรมไมโครซอฟท์ เอสเอ็มเอส และโปรแกรมอินเทล แลนเดส
3. การบริหารการทำงาน ของโปรแกรม (Workload Management) โปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ สามารถทำงานได้ชัดเจนกว่าโปรแกรมไมโครซอฟท์ เอสเอ็มเอสและโปรแกรมอินเทล แลนเดส ซึ่งจะสามารถแบ่งการทำงานเป็นส่วนๆและกำหนดเวลาการทำงานเป็นช่วงเวลา ทำให้สามารถใช้งานเครื่องคอมพิวเตอร์ได้อย่างมีประสิทธิภาพสูงสุด
4. โปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์เท่านั้นที่เปิดให้มีการเชื่อมต่อโปรแกรมระหว่างโปรแกรมที่พัฒนาขึ้นมาเอง และโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ ทำให้สามารถติดต่อและเรียกใช้งานเพื่อทำงานที่ต้องการได้อย่างมีประสิทธิภาพ
5. การแสดงผลในแบบรูปภาพหรือจียูไอ ซึ่งมีเพียงโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์เท่านั้นที่จะสามารถแสดงผลเป็นรูปภาพและสามารถบริหารข้อมูลได้ไปพร้อมๆกัน บนฐานข้อมูลเดียวกันได้อย่างมีประสิทธิภาพและสามารถกำหนดมุมมองที่สนใจเพื่อวิเคราะห์ปัญหาต่างๆได้สะดวกยิ่งขึ้น

สรุปได้ว่าความสามารถต่างๆของโปรแกรมทั้งสามโปรแกรมที่ใช้ในการทดสอบจะพอๆกัน แต่โปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์ จะมีความสามารถในการทำงานที่มีรายละเอียดสูงกว่า ถึงแม้ว่าการทำงานบางตัวจะต้องซื้อเพิ่ม แต่การใช้งานโปรแกรมจะง่ายกว่าและเปิดให้ผู้ใช้สามารถเขียนโปรแกรม (Software Development Kit) เพื่อใช้ในการติดต่อกับโปรแกรมทีเอ็นจี ยูนิเซ็นเตอร์เองหรืออ่านข้อมูลเพื่อใช้ในการบริหารข้อมูลบนเครือข่ายได้ ซึ่งจะเห็นว่าจะเป็นโปรแกรมที่มีประโยชน์และมีประสิทธิภาพสูง

ภาคผนวก ง
รายละเอียดแฟ้มรหัสต้นฉบับ
ระบบบริหารข้อมูลในฮาร์ดดิสก์ผ่านเครือข่าย

1. โปรแกรมส่วนจัดเก็บข้อมูลรายละเอียดเครื่องคอมพิวเตอร์ต่างๆ

1.1 โปรแกรมตัวกลางในการติดต่อซีโอเอ็มออบเจ็กต์ต่างๆ (SWDLocator.exe)

1.1.1 SWDloc.idl กำหนดต้นแบบของซีโอเอ็มออบเจ็กต์ SWDLocator

```
// Bring in needed system IDL files
import "wtypes.idl";
import "unknwn.idl";
//
// Interface information for ISWDLocator
//
[ object, uuid(65EC0A46-86D5-11d3-9E6E-00AA00885006) ]
interface ISWDLocator : IUnknown
{
  HRESULT Locate([in] LPOLESTR          pszItemName,
          [in] REFIID           riidResult,
          [out, iid_is(riidResult)] void **ppvResult);
};
//
// Class information for SWDLocator1
//
[ uuid(96DB56D0-86D6-11d3-9E6E-00AA00885006) ]
coclass SWDLocator1
{
  interface ISWDLocator;
};
```

1.1.2 SWDloc_dll.c กำหนดต้นแบบของซีโอเอ็มออบเจ็กต์ SWDLocator

```
/*********************************************************
  DllData file -- generated by MIDL compiler
  This file is regenerated by MIDL on every IDL file compile.
  To completely reconstruct this file, delete it and rerun MIDL
  on all the IDL files in this DLL, specifying this file for the
  /dlldata command line option
*********************************************************/
#include <rpcproxy.h>
```

```
#ifdef __cplusplus
extern "C"  {
#endif
EXTERN_PROXY_FILE( SWDloc )

PROXYFILE_LIST_START
/* Start of list */
  REFERENCE_PROXY_FILE( SWDloc ),
/* End of list */
PROXYFILE_LIST_END
DLLDATA_ROUTINES( aProxyFileList, GET_DLL_CLSID )
#ifdef __cplusplus
} /*extern "C" */
#endif
/* end of generated dlldata file */
```

1.1.3 PSSWDloc1.def กำหนดต้นแบบของซีโอเอ็มออบเจ็กต์

```
LIBRARY   PSSWDloc1
EXPORTS
        DllGetClassObject     PRIVATE
        DllCanUnloadNow       PRIVATE
        DllRegisterServer     PRIVATE
        DllUnregisterServer   PRIVATE
        GetProxyDllInfo       PRIVATE
```

1.1.4 SWDloc_i.c ฟังก์ชั่นต้นแบบของซีโอเอ็มออบเจ็กต์ SWDLocator

```
/* this file contains the actual definitions of */
/* the IIDs and CLSIDs */
/* link this file in with the server and any clients */
/* File created by MIDL compiler version 5.01.0164 */
/* at Tue Nov 02 20:12:13 1999
 */
/* Compiler settings for ..\Common\SWDloc.idl:
    Os (OptLev=s), W1, Zp8, env=Win32, ms_ext, app_config, c_ext
    error checks: allocation ref bounds_check enum stub_data
*/
//@@MIDL_FILE_HEADING( )
#ifdef __cplusplus
extern "C"{
#endif

#ifndef __IID_DEFINED__
#define __IID_DEFINED__
typedef struct _IID
{
```

```
  unsigned long x;
  unsigned short s1;
  unsigned short s2;
  unsigned char  c[8];
} IID;
#endif // __IID_DEFINED__
#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED
const IID IID_ISWDLocator = {0x66EC0A46,0x86D5,0x11d3,{0x9E,0x6E,0x00,0xAA,0x00,0xB8,0x50,0x06}};
const CLSID CLSID_SWDLocator1 = {0x96DB56D0,0x86D6,0x11d3,{0x9E,0x6E,0x00,0xAA,0x00,0xB8,0x50,0x06}};
#ifdef __cplusplus
}
#endif
```

1.1.6 Atlline.cpp ฟังก์ชั่นต้นแบบของซีโอเอ็มออบเจ็กต์

```
#include "ATLLine.h"
#include "Utility.h"
CATLCommandLine::CATLCommandLine(PCHAR pszCommandLine)
{
  // Initialize our temporary and blast into uppercase
  CHAR  achCommLine[256];
  _ASSERTE(strlen(pszCommandLine) < 256);
  strcpy(achCommLine, pszCommandLine);
  _strupr(achCommLine);
  // Now look for a RegServer or UnregServer argument
  m_bRegister   = FindArg(achCommLine, "REGSERVER")   ? TRUE : FALSE;
  m_bUnregister = FindArg(achCommLine, "UNREGSERVER") ? TRUE : FALSE;
}

BOOL CATLCommandLine::FindArg(PCHAR pszBaseString,
                              PCHAR pszSearchArg)
{
  // Now look for the search string
  PCHAR pszArg = strstr(pszBaseString, pszSearchArg);
  if (pszArg && pszArg != pszBaseString)
  {
    // Is this a command-line argument?
    if (*(--pszArg) == '-' || *pszArg == '/')
      return TRUE;
  }
  return FALSE;
}
```

1.1.7 ATLLocalSrv.cpp ฟังก์ชั่นซีโอเอ็มออบเจ็กต์แบบโลคอลเซิร์ฟเวอร์

```
// Get needed include files
#include "SWDLocImp.h"
#include "ATLLine.h"
// Object map, indicates which classes are serviced by this server
BEGIN_OBJECT_MAP(ObjectMap)
  OBJECT_ENTRY(CLSID_SWDLocator1, ComSWDLocator)
END_OBJECT_MAP()
// The global module object
QueATLModule _Module;
//
// Standard start function for a Windows executable
//
extern "C"
int WINAPI WinMain (HINSTANCE   hInstance,
            HINSTANCE   /*hPrevInstance*/,
            LPSTR       pszCommLine,
            int         /*nShowCmd*/)
{
  // Process our command-line
  CATLCommandLine CommLine(pszCommLine);
  // Crank up COM
  HRESULT hResult = CoInitialize(NULL);
  if (FAILED(hResult)) {
    ReportError("Could not initialize OLE subsystem.", hResult);
    return FALSE;
  }
// Setup the module object
  _Module.Init(ObjectMap, hInstance);
// Do we need to self-register?
  if (CommLine.Register())
    hResult = _Module.RegisterServer();
  // Modify by anirut lamsakul
hResult = _Module.UpdateRegistryFromResource(IDR_REGSCRIPT,TRUE);
// Do we need to self-unregister?
  if (CommLine.Unregister())
    hResult = _Module.UnregisterServer();
// Everything's looking good -- startup the server?
  if (CommLine.ShouldRun())
  {
// Register the class factories with COM
    hResult = _Module.RegisterClassObjects(CLSCTX_LOCAL_SERVER,
                        REGCLS_MULTIPLEUSE);
    if (FAILED(hResult)) {
      ReportError("Could not register class factories.", hResult);
      return FALSE;
```

```
    }

// Main server message loop
    MSG msg;
    while (GetMessage(&msg, 0, 0, 0))
      DispatchMessage(&msg);
// Unregister class factories
    _Module.RevokeClassObjects();
  }
// Bring down COM
  CoUninitialize();
return hResult;
}
```

1.1.8 StdALT.cpp ฟังก์ชั่นมาตรฐานของซีโอเอ็มออบเจ็กต์

```
// Get needed include files
#include "StdInc.h"
#ifdef _ATL_STATIC_REGISTRY
#include <statreg.h>
#include <statreg.cpp>
#endif
#include <atlimpl.cpp>
```

1.1.9 Utility.cpp ฟังก์ชั่นช่วยเหลือ

```
// Get needed system and custom include files
#include "Utility.h"
#include <stdlib.h>
#include <time.h>
#include <sys/types.h>
#include <sys/timeb.h>
#include <strstream>
// Generic error reporting function
// Display error text and HRESULT
void ReportError(const std::string& strErrorText, HRESULT hResult)
{
  #ifdef NO_CONSOLE
    std::strstream strbuff;
    strbuff << "Error: \"" << strErrorText << "\"\n";
    strbuff << "HRESULT: 0x" << std::hex << (ULONG) hResult << std::dec << std::ends;
    MessageBox(NULL, strbuff.str(), "Error", MB_OK | MB_ICONWARNING);
  #else
    std::cout << "Error: \"" << strErrorText << "\"\n";
    std::cout << "HRESULT: 0x" << std::hex << (ULONG) hResult << std::dec << std::endl;
  #endif
}
```

```
// Convert a ULONG to a string or CString object
UTILSTR ULongToStr(const ULONG ulNumber)
{
  CHAR chBuffer[33];
  _ultoa(ulNumber, chBuffer, 10);
  return UTILSTR(chBuffer);
}
/*
 * AnsiToUnicode converts the ANSI string pszA to a Unicode string
 * and returns the Unicode string through ppszW. Space for the
 * the converted string is allocated by AnsiToUnicode.
 */
HRESULT __fastcall AnsiToUnicode(LPCSTR pszA, LPOLESTR* ppszW)
{
  ULONG cCharacters;
  DWORD dwError;
// If input is null then just return the same.
  if (NULL == pszA)
  {
    *ppszW = NULL;
    return NOERROR;
  }
// Determine number of wide characters to be allocated for the
  // Unicode string.
  cCharacters =  strlen(pszA)+1;
  // Use of the OLE allocator is required if the resultant Unicode
  // string will be passed to another COM component and if that
  // component will free it. Otherwise you can use your own allocator.
  *ppszW = (LPOLESTR) CoTaskMemAlloc(cCharacters*2);
  if (NULL == *ppszW)
    return E_OUTOFMEMORY;

  // Covert to Unicode.
  if (0 == MultiByteToWideChar(CP_ACP, 0, pszA, cCharacters,
                  *ppszW, cCharacters))
  {
    dwError = GetLastError();
    CoTaskMemFree(*ppszW);
    *ppszW = NULL;
    return HRESULT_FROM_WIN32(dwError);
  }

  return NOERROR;
}
/*
 * UnicodeToAnsi converts the Unicode string pszW to an ANSI string
```

```
* and returns the ANSI string through ppszA. Space for the
* the converted string is allocated by UnicodeToAnsi.
*/
HRESULT __fastcall UnicodeToAnsi(LPCOLESTR pszW, LPSTR* ppszA)
{
  ULONG cbAnsi, cCharacters;
  DWORD dwError;

  // If input is null then just return the same.
  if (pszW == NULL)
  {
    *ppszA = NULL;
    return NOERROR;
  }

  cCharacters = wcslen(pszW)+1;
  // Determine number of bytes to be allocated for ANSI string. An
  // ANSI string can have at most 2 bytes per character (for Double
  // Byte Character Strings.)
  cbAnsi = cCharacters*2;

  // Use of the OLE allocator is not required because the resultant
  // ANSI string will never be passed to another COM component. You
  // can use your own allocator.
  *ppszA = (LPSTR) CoTaskMemAlloc(cbAnsi);
  if (NULL == *ppszA)
    return E_OUTOFMEMORY;

  // Convert to ANSI.
  if (0 == WideCharToMultiByte(CP_ACP, 0, pszW, cCharacters, *ppszA,
                  cbAnsi, NULL, NULL))
  {
    dwError = GetLastError();
    CoTaskMemFree(*ppszA);
    *ppszA = NULL;
    return HRESULT_FROM_WIN32(dwError);
  }
return NOERROR;
}
// Function to seed the random number generator based on the system time
void SeedRandomGenerator()
{
  struct _timeb timebuffer;
  _ftime(&timebuffer);
  srand(timebuffer.millitm);
}
```

1.1.10 SWDLocImp.cpp ฟังก์ชั่นการติดต่อของซีโอเอ็มออบเจ็กต์ SWDLocator

```
// Get needed include files
//#define INITGUID
#include "SWDLocImp.h"
#include "Utility.h"
// Constructor and destructor
ComSWDLocator::ComSWDLocator()
{
  VerboseMsg("In SWDLocator constructor.\n");
}
ComSWDLocator::~ComSWDLocator()
{
  VerboseMsg("In SWDLocator destructor.\n");
}
// ISWDLocator interface members
STDMETHODIMP
ComSWDLocator::Locate(LPOLESTR     pszItemName,
                      REFIID       riidResult,
                      PPVOID       ppvResult)
{
  HRESULT hResult;
  VerboseMsg("In Locate.\n");
  USES_CONVERSION;
  // Create a moniker for the SWD server
  IMoniker* pIMoniker = NULL;
  hResult = CreateFileMoniker(pszItemName,
                              &pIMoniker);
  if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not create file moniker.", hResult);
    return hResult;
  }
  // Check to see if there is a registered object
  hResult = BindMoniker(pIMoniker, 0, riidResult, ppvResult);
  if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not bind to moniker.", hResult);
    return hResult;
  }
  // Clean up our various interface pointers
  pIMoniker->Release();
  pIMoniker = NULL;
return NOERROR;
}
```

1.3 โปรแกรมเอเจ็นจัดเก็บข้อมูลรายละเอียดเครื่องคอมพิวเตอร์ที่เครื่องรับบริการ (inventAgent.exe)

1.3.1 GUIDS.h กำหนดค่าเริ่มต้นของซีโอเอ็มออบเจ็กต์

```
#ifndef GUIDS_H
#define GUIDS_H
// Get needed include files
#include "StdInc.h"
// We can get the IID for ISWD"Class" from the IDL-generated header file
#include "SWDMake.h"
#include "SWDLoc.h"
#include "SWDInven.h"
DEFINE_GUID(CLSID_SWDLocator1,
0x96DB56D0, 0x86D6, 0x11d3, 0x9E, 0x6E, 0x00,0xAA, 0x00,0xB8, 0x50, 0x06);
DEFINE_GUID(CLSID_SWDMaker1,
0xB17438C0, 0x882E, 0x11d3, 0x9E, 0x71, 0x00,0xAA, 0x00,0xB8, 0x50, 0x06);
DEFINE_GUID(CLSID_SWDInven1,
0x716A19B0, 0x9135, 0x11d3, 0x9B, 0xCA, 0x00,0xAA, 0x00,0xB8, 0x50, 0x06);
#endif
```

1.3.2 StdAfx.h กำหนดฟังก์ชั่นต้นแบบ

```
// stdafx.h : include file for standard system include files,
//  or project specific include files that are used frequently, but
//      are changed infrequently
//
#if !defined(AFX_STDAFX_H__22D7E763_8850_11D3_9E72_00AA00B85006__INCLUDED_)
#define AFX_STDAFX_H__22D7E763_8850_11D3_9E72_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#define VC_EXTRALEAN            // Exclude rarely-used stuff from Windows headers
#include <afxwin.h>             // MFC core and standard components
#include <afxext.h>             // MFC extensions
#include <afxdtctl.h>           // MFC support for Internet Explorer 4 Common Controls
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>             // MFC support for Windows Common Controls
#endif // _AFX_NO_AFXCMN_SUPPORT
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_STDAFX_H__22D7E763_8850_11D3_9E72_00AA00B85006__INCLUDED_)
```

1.3.3 inventAgent.h กำหนดค่าเริ่มต้นของฟังก์ชั่นจัดเก็บรายละเอียดเครื่องคอมพิวเตอร์

```
// InvenAgent.h : main header file for the INVENAGENT application
//

#if !defined(AFX_INVENAGENT_H__22D7E75F_8850_11D3_9E72_00AA00B85006__INCLUDED_)
#define AFX_INVENTAGENT_H__22D7E75F_8850_11D3_9E72_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#ifndef __AFXWIN_H__
        #error include 'stdafx.h' before including this file for PCH
#endif
//#include "resource.h"                 // main symbols
////////////////////////////////////////////////////////////////////////
// CInvenAgentApp:
// See InvenAgent.cpp for the implementation of this class
class CInvenAgentApp : public CWinApp
{
public:
        CInvenAgentApp();
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CClientwizApp)
        public:
        virtual BOOL InitInstance();
        //}}AFX_VIRTUAL
// Implementation
        //{{AFX_MSG(CInvenAgentApp)
                // NOTE - the ClassWizard will add and remove member functions here.
                //    DO NOT EDIT what you see in these blocks of generated code !
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_INVENAGENT_H__22D7E75F_8850_11D3_9E72_00AA00B85006__INCLUDED_)
```

1.3.7 SWDInvent_i.c ฟังก์ชั่นต้นแบบของซีโอเอ็มออบเจ็กต์ SWDInvent

```
/* this file contains the actual definitions of */
/* the IIDs and CLSIDs */
/* link this file in with the server and any clients */
/* File created by MIDL compiler version 5.01.0164 */
/* at Thu Jan 20 01:07:12 2000
 */
```

```
/* Compiler settings for ..\Common\SWDInven.idl:
    Os (OptLev=s), W1, Zp8, env=Win32, ms_ext, app_config, c_ext
    error checks: allocation ref bounds_check enum stub_data
*/
//@@MIDL_FILE_HEADING( )
#ifdef __cplusplus
extern "C"{
#endif
#ifndef __IID_DEFINED__
#define __IID_DEFINED__
typedef struct _IID
{
    unsigned long x;
    unsigned short s1;
    unsigned short s2;
    unsigned char  c[8];
} IID;
#endif // __IID_DEFINED__
#ifndef CLSID_DEFINED
#define CLSID_DEFINED
typedef IID CLSID;
#endif // CLSID_DEFINED
const IID IID_ISWDInven = {0x6B57EA80,0x9134,0x11d3,{0x9B,0xCA,0x00,0xAA,0x00,0xB8,0x50,0x06}};
const CLSID CLSID_SWDInven1 = {0x716A19B0,0x9135,0x11d3,{0x9B,0xCA,0x00,0xAA,0x00,0xB8,0x50,0x06}};
#ifdef __cplusplus
}
#endif
```

1.3.9 inventAgent.cpp ฟังก์ชั่นการเก็บรายละเอียดเครื่องคอมพิวเตอร์

```
// InvenAgent.cpp : Defines the class behaviors for the application.
//
//#include <atlbase.h>
//#include <atlcom.h>
#include "stdafx.h"
#include "InvenAgent.h"
//#include "resource.h"
#include "utility.h"
#include "GUIDs.h"
/////////////////////////////////////////////////////////////////////////////
// CInvenAgentApp
BEGIN_MESSAGE_MAP(CInvenAgentApp, CWinApp)
        //{{AFX_MSG_MAP(CClientwizApp)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG
```

```
        ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CClientwizApp construction
CInvenAgentApp::CInvenAgentApp()
{
        // TODO: add construction code here,
        // Place all significant initialization in InitInstance
}
/////////////////////////////////////////////////////////////////////////////
// The one and only CClientwizApp object
CInvenAgentApp theApp;
/////////////////////////////////////////////////////////////////////////////
// CClientwizApp initialization
BOOL CInvenAgentApp::InitInstance()
{ HRESULT hResult;
        // Standard initialization
        // If you are not using these features and wish to reduce the size
        //  of your final executable, you should remove from the following
        //  the specific initialization routines you do not need.
#ifdef _AFXDLL
        Enable3dControls();                     // Call this when using MFC in a shared DLL
#else
        Enable3dControlsStatic();       // Call this when linking to MFC statically
#endif
        if (!AfxOleInit()) {
                AfxMessageBox("Could not initialize OLE subsystem.");
                return FALSE;
        }
  USES_CONVERSION;
  // Now create the appropriate server object
  ISWDLocator*  pISWDLocator = NULL;
  LPOLESTR    pszHostname  = NULL;
  ISWDInven*   pISWDInven = NULL;
  char       achHostname[20];
  CString     sComputerName;
  CString     sDomain;
  CString     sOperatingSystem;
  CString     sOSVersion;
  CString     sSystemDirectory;
  CString     sHostName;
  CString     sIPAddress;
  CString     sMacAddress;
  CString     sUserName;
  CString     PathName;
  CString     buf1;
```

```
CString     stemp;
ULONG       ulDriveCSize=0;
ULONG       ulDriveCFSpace=0;
ULONG       ulDriveDSize=0;
ULONG       ulDriveDFSpace=0;
ULONG       ulDriveESize=0;
ULONG       ulDriveEFSpace=0;
ULONG       ulMemSize=0;
char        num[20];
char        str[50];
int         l,i;
CStdioFile  f1;
GetWindowsDirectory(str,50);
PathName = str;
PathName += "\\CUSWD\\bin\\Inven.exe";
system(PathName);
PathName = str;
PathName += "\\CUSWD\\Log\\Inven.cfg";
f1.Open(PathName,CFile::modeRead);
while (!feof(f1.m_pStream))
{
                f1.ReadString(buf1);
                if ((i=buf1.Find("omputer")) > 0)
                {
                        l = buf1.GetLength();
                        i = buf1.Find("=");
                        sComputerName = buf1.Right(l-i-2);
                        sComputerName.TrimRight();
                }
                if ((i=buf1.Find("ost")) > 0)
                {
                        l = buf1.GetLength();
                        i = buf1.Find("=");
                        sHostName = buf1.Right(l-i-2);
                        sHostName.TrimRight();
                }
                if ((i=buf1.Find("omain")) > 0)
                {
                        l = buf1.GetLength();
                        i = buf1.Find("=");
                        sDomain = buf1.Right(l-i-2);
                        sDomain.TrimRight();
                }
                if ((i=buf1.Find("perating")) > 0)
                {
                        l = buf1.GetLength();
```

```
            i = buf1.Find("=");
            sOperatingSystem = buf1.Right(l-i-2);
            sOperatingSystem.TrimRight();
}
if ((i=buf1.Find("S Version")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            sOSVersion = buf1.Right(l-i-2);
            sOSVersion.TrimRight();
}
if ((i=buf1.Find("P Address")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            sIPAddress = buf1.Right(l-i-2);
            sIPAddress.TrimRight();
}
if ((i=buf1.Find("ac Address")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            sMacAddress = buf1.Right(l-i-2);
            sMacAddress.TrimRight();
}
if ((i=buf1.Find("Directory")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            sSystemDirectory = buf1.Right(l-i-2);
            sSystemDirectory.TrimRight();
}
if ((i=buf1.Find("ser name")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            sUserName = buf1.Right(l-i-2);
            sUserName.TrimRight();
}
if ((i=buf1.Find("C: Size")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            stemp = buf1.Right(l-i-2);
            stemp.TrimRight();
            strcpy(num,stemp);
```

```
            ulDriveCSize = atol(num);
}
if ((i=buf1.Find("C: Free")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            stemp = buf1.Right(l-i-2);
            stemp.TrimRight();
            strcpy(num,stemp);
            ulDriveCFSpace = atol(num);
}
if ((i=buf1.Find("D: Size")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            stemp = buf1.Right(l-i-2);
            stemp.TrimRight();
            strcpy(num,stemp);
            ulDriveDSize = atol(num);
}
if ((i=buf1.Find("D: Free")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            stemp = buf1.Right(l-i-2);
            stemp.TrimRight();
            strcpy(num,stemp);
            ulDriveDFSpace = atol(num);
}
if ((i=buf1.Find("E: Size")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            stemp = buf1.Right(l-i-2);
            stemp.TrimRight();
            strcpy(num,stemp);
            ulDriveESize = atol(num);
}
if ((i=buf1.Find("E: Free")) > 0)
{
            l = buf1.GetLength();
            i = buf1.Find("=");
            stemp = buf1.Right(l-i-2);
            stemp.TrimRight();
            strcpy(num,stemp);
            ulDriveEFSpace = atol(num);
```

```
                }
                if ((i=buf1.Find("Memory Size")) > 0)
                {
                        i = buf1.GetLength();
                        i = buf1.Find("=");
                        stemp = buf1.Right(i-i-2);
                        stemp.TrimRight();
                        strcpy(num,stemp);
                        ulMemSize = atol(num);
                }
}
f1.Close();
// assign values into Parameters
BSTR            bstrComputerName = sComputerName.AllocSysString();
BSTR            bstrDomain = sDomain.AllocSysString();
BSTR            bstrOperationgSystem = sOperatingSystem.AllocSysString();
BSTR            bstrOSVersion = sOSVersion.AllocSysString();
BSTR            bstrSystemDirectory = sSystemDirectory.AllocSysString();
BSTR            bstrHostName = sHostName.AllocSysString();
BSTR            bstrIPAddress = sIPAddress.AllocSysString();
BSTR            bstrMacAddress = sMacAddress.AllocSysString();
BSTR            bstrUserName = sUserName.AllocSysString();
// define Server to connect SWD Maker COM
strcpy(achHostname,"ANT-SERVER");
strcpy(achHostname,sHostName);
AnsiToUnicode((PCHAR) achHostname, &pszHostname);
// Must change by SQL DBMS
COSERVERINFO  ServerInfo = { 0, pszHostname, NULL, 0 };
MULTI_QI    MultiQI   = { &IID_ISWDLocator, NULL, NOERROR };
hResult = CoCreateInstanceEx(CLSID_SWDLocator1,
                NULL, CLSCTX_SERVER,
                &ServerInfo,
                1, &MultiQI);
if (FAILED(hResult)) {
  // We'll let the client report the error
  // ReportError("Could not create a new PizzaLocator object.", hResult);
        AfxMessageBox("Can not create SWDLocator object.");
        return hResult;
}
// Free up our hostname string
CoTaskMemFree(pszHostname);
// Just for convenience
piSWDLocator = (ISWDLocator*) MultiQI.pItf;
// Look for any PizzaMaker objects that might be currently running
// on the same machine as the locator (PPVOID) &piSWDInven
hResult = piSWDLocator->Locate(A2OLE("SWDInven"),
```

```
                IID_ISWDInven,
                (PPVOID) &pISWDInven);
// Clean up some interface pointers...
pISWDLocator->Release();
pISWDLocator = NULL;
if (FAILED(hResult)) {
  // We'll let the client report the error
            AfxMessageBox("Can not connect SWDInven");
            return hResult;
}
hResult=pISWDInven->MakeInventory(bstrComputerName,
              bstrDomain,bstrOperationgSystem,bstrOSVersion,
                    bstrSystemDirectory,bstrHostName,bstrIPAddress,
                    bstrMacAddress,bstrUserName,ulDriveCSize,ulDriveCFSpace,
                    ulDriveDSize,ulDriveDFSpace,ulDriveESize,ulDriveEFSpace,
                    ulMemSize);
if (FAILED(hResult))
{ AfxMessageBox("Error Make Package");}
AfxMessageBox("Maked Inventory Complete..");
pISWDInven->Release();
            return FALSE;
}
```

1.4 โปรแกรมจัดเก็บข้อมูลรายละเอียดเครื่องคอมพิวเตอร์ที่เครื่องให้บริการ (SWDInvent.exe)

1.4.5 SWDInvenImp.h กำหนดค่าฟังก์ชั่นเริ่มต้นการจัดเก็บข้อมูลบน เครื่องให้บริการ

```
#ifndef SWDINVENIMP_H
#define SWDINVENIMP_H
// Get needed include files
#include "StdAfx.h"
#include "resource.h"
#include "GUIDs.h"
class ATL_NO_VTABLE ComSWDInven :
  public CComObjectRoot,
  public CComCoClass<ComSWDInven, &CLSID_SWDInven1>,
  public ISWDInven
{
public:
// Ctor and dtors
  ComSWDInven();
  virtual ~ComSWDInven();
// Overriden public methods
  HRESULT FinalConstruct();
  void   FinalRelease();
BOOL m_IsConnectionOpen;
  _ConnectionPtr m_pConnection;
```

```
STDMETHOD(MakeInventory)( BSTR      bstrComputerName,
                          BSTR      bstrDomain,
                          BSTR      bstrOperatingSystem,
                          BSTR      bstrOSVersion,
                          BSTR      bstrSystemDirectory,
                          BSTR      bstrHostName,
                          BSTR      bstrIPAddress,
                          BSTR      bstrMaxAddress,
                          BSTR      bstrUserName,
                          ULONG     ulDriveCSize,
                          ULONG     ulDriveCFSpace,
                          ULONG     ulDriveDSize,
                          ULONG     ulDriveDFSpace,
                          ULONG     ulDriveESize,
                          ULONG     ulDriveEFSpace,
                          ULONG     ulMemSize);

  STDMETHOD(UpdateClient)( ULONG   ulAdvertiseID,
                           ULONG   ulPkgID,
                           BSTR    bstrComputerName);

  STDMETHOD(UpdatePcmComplete)( ULONG   ulAdvertiseID,
                                ULONG   ulPkgID,
                                BSTR    bstrComputerName);

  STDMETHOD(UpdatePcmRetry)(   ULONG   ulAdvertiseID,
                               ULONG   ulPkgID,
                               BSTR    bstrComputerName);
  BEGIN_COM_MAP(ComSWDInven)
    COM_INTERFACE_ENTRY(ISWDInven)
  END_COM_MAP()
DECLARE_REGISTRY_RESOURCEID(IDR_REGSCRIPT)
  DECLARE_PROTECT_FINAL_CONSTRUCT()
// Private data members
  IMoniker* m_pIMoniker;
  DWORD     m_dwROTCookie;
};
typedef ComSWDInven* PComSWDInven;
#endif
```

1.4.9 SWDInven.idl กำหนดต้นแบบของซีโอเอ็มออบเจ็กต์ SWDInven

```
// Bring in needed system IDL files
import "wtypes.idl";
Import "unknwn.idl";
//
```

```
// Interface information for ISWDInven
//
//
[ object, uuid(6B57EA80-9134-11d3-9BCA-00AA00B85006) ]
interface ISWDInven : IUnknown
{
  HRESULT MakeInventory(   [in] BSTR    bstrComputerName,
                           [in] BSTR    bstrDomain,
                           [in] BSTR    bstrOperatingSystem,
                           [in] BSTR    bstrOSVersion,
                           [in] BSTR    bstrSystemDirectory,
                           [in] BSTR    bstrHostName,
                           [in] BSTR    bstrIPAddress,
                           [in] BSTR    bstrMacAddress,
                           [in] BSTR    bstrUserName,
                           [in] ULONG   ulDriveCSize,
                           [in] ULONG   ulDriveCFSpace,
                           [in] ULONG   ulDriveDSize,
                           [in] ULONG   ulDriveDFSpace,
                           [in] ULONG   ulDriveESize,
                           [in] ULONG   ulDriveEFSpace,
                           [in] ULONG   ulMemSize);
  HRESULT UpdatePcmComplete([in] ULONG   ulAdvertiseID,
                            [in] ULONG   ulPkgID,
                            [in] BSTR    bstrComputerName);

  HRESULT UpdateClient(    [in] ULONG   ulAdvertiseID,
                           [in] ULONG   ulPkgID,
                           [in] BSTR    bstrComputerName);
  HRESULT UpdatePcmRetry(  [in] ULONG   ulAdvertiseID,
                           [in] ULONG   ulPkgID,
                           [in] BSTR    bstrComputerName);
};
//
// Class information for SWDInven1
//
[ uuid(716A19B0-9135-11d3-9BCA-00AA00B85006) ]
coclass SWDInven1
{
  interface ISWDInven;
};
```

1.4.12 SWDInvenImp.cpp ฟังก์ชั่นการติดต่อของซีโอเอ็มออบเจ็กต์ SWDInven ในการจัดเก็บข้อมูลลงฐานข้อมูลที่เครื่องให้บริการ

```
// Get needed include files
#include "stdafx.h"
#include "SWDInvenImp.h"
#include "Utility.h"
#include "StdInc.h"
//
// Constructor and destructor
//
ComSWDInven::ComSWDInven()
    : m_pIMoniker(NULL),
      m_dwROTCookie(0)
{

  VerboseMsg("In SWDInven constructor.\n");
}

ComSWDInven::~ComSWDInven()
{
  VerboseMsg("In SWDInven destructor.\n");
}
//
// Overridden public methods
//
HRESULT ComSWDInven::FinalConstruct()
{
  USES_CONVERSION;
  VerboseMsg("In SWDInven FinalConstruct method.\n");
  // Create the file moniker
  HRESULT hResult = CreateFileMoniker(A2OLE("SWDInven"),
                        &m_pIMoniker);
  if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not create file moniker.", hResult);
    return hResult;
  }
 // Get an interface pointer for the ROT
  IRunningObjectTable* pIROT = NULL;
  hResult = GetRunningObjectTable(0, &pIROT);
  if (FAILED(hResult)) {
    m_pIMoniker->Release();
    m_pIMoniker = NULL;
    return hResult;
  }
```

```
// Now place the moniker into the ROT
hResult = pIROT->Register(0, GetUnknown(), m_pIMoniker, &m_dwROTCookie);
if (FAILED(hResult)) {
  m_pIMoniker->Release();
  m_pIMoniker = NULL;
  pIROT->Release();
  return hResult;
}
// Release our IROT pointer
pIROT->Release();
// Start connection database SQL SERVER CUSWD
HRESULT hr;
try
{
        hr = m_pConnection.CreateInstance( __uuidof( Connection ) );
        if (SUCCEEDED(hr))
        {
                hr = m_pConnection->Open(
                _bstr_t(L"CUSWD"),    // System DSN of SQL Server
                _bstr_t(L"cusa"),     // User name
                _bstr_t(L"cusaa"),    // Password
                adModeUnknown);
                if (SUCCEEDED(hr))   m_IsConnectionOpen = TRUE;
        }
}
catch( _com_error &e )
{
        // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for clases generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
        return hr;
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
        return hr;
}
return NOERROR;
}
void ComSWDInven::FinalRelease()
{
```

```
// Remove our moniker from the ROT
if (m_dwROTCookie) {
// Get an interface pointer for the ROT
  IRunningObjectTable* pIROT = NULL;
  HRESULT hResult = GetRunningObjectTable(0, &pIROT);
 // Now remove the moniker
  if (SUCCEEDED(hResult)) {
    hResult = pIROT->Revoke(m_dwROTCookie);
    pIROT->Release();
  }
  m_dwROTCookie = 0;
 }
// And get rid of our Moniker
 if (m_pIMoniker) {
  m_pIMoniker->Release();
  m_pIMoniker = NULL;
 }
 // Close Database Connection
 if (m_IsConnectionOpen)
 {
            m_IsConnectionOpen = FALSE;
            m_pConnection->Close();
 }
}
//
// ISWDMaker interface members
//
STDMETHODIMP
ComSWDInven::MakeInventory ( BSTR    bstrComputerName,
                             BSTR    bstrDomain,
                             BSTR    bstrOperatingSystem,
                             BSTR    bstrOSVersion,
                             BSTR    bstrSystemDirectory,
                             BSTR    bstrHostName,
                             BSTR    bstrIPAddress,
                             BSTR    bstrMacAddress,
                             BSTR    bstrUserName,
                             ULONG   ulDriveCSize,
                             ULONG   ulDriveCFSpace,
                             ULONG   ulDriveDSize,
                             ULONG   ulDriveDFSpace,
                             ULONG   ulDriveESize,
                             ULONG   ulDriveEFSpace,
                             ULONG   ulMemSize)
{
  VerboseMsg("In MakeInventory.\n");
```

```
LPSTR pszComputerName,pszDomain,pszOperatingSystem,pszOSVersion;
LPSTR pszSystemDirectory,pszHostName,pszIPAddress,pszMacAddress,pszUserName;
// Convert value to  used
UnicodeToAnsi(bstrComputerName,&pszComputerName);
UnicodeToAnsi(bstrDomain,&pszDomain);
UnicodeToAnsi(bstrOperatingSystem,&pszOperatingSystem);
UnicodeToAnsi(bstrOSVersion,&pszOSVersion);
UnicodeToAnsi(bstrSystemDirectory,&pszSystemDirectory);
UnicodeToAnsi(bstrHostName,&pszHostName);
UnicodeToAnsi(bstrIPAddress,&pszIPAddress);
UnicodeToAnsi(bstrMacAddress,&pszMacAddress);
UnicodeToAnsi(bstrUserName,&pszUserName);
   _RecordsetPtr pRecordSet;
_bstr_t bstrQuery1("SELECT * FROM Inventory");
CString ssql1="UPDATE Inventory ";
CString strQueryOS="SELECT OperatingSystemType FROM OperatingSystem \
                WHERE OperatingSystemName = '";
strQueryOS += pszOperatingSystem;
strQueryOS += "'";
_bstr_t bstrQueryMax("SELECT MAX(MachineID) AS KY FROM Inventory");
CString ssql="INSERT INTO Inventory VALUES (";
_variant_t vRecsAffected(0L);
long cOperatingSystemType;
LONG cNumRow=1;
char  str[300];
char msg[80];
try
{
        pRecordSet = m_pConnection->Execute(bstrQueryMax,&vRecsAffected,adOptionUnspecified);
        _variant_t vNum;
        while (!pRecordSet->GetadoEOF())
        {
                vNum = pRecordSet->GetCollect(L"KY");
                cNumRow = vNum;
                cNumRow ++;
                pRecordSet->MoveNext();
        }
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
```

```
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
            TRACE( " *** Unhandled Exception *** ");
}
// Search Operating System Type
try
{
            pRecordSet = m_pConnection->Execute( _bstr_t(strQueryOS),&vRecsAffected,adOptionUnspecified);
            _variant_t vNum;
            while (!pRecordSet->GetadoEOF())
            {
                        vNum = pRecordSet->GetCollect(L"OperatingSystemType");
                        cOperatingSystemType = vNum;
                        pRecordSet->MoveNext();
            }
            pRecordSet->Close();
}
catch( _com_error &e )
{           // Get info from _com_error
            _bstr_t bstrSource(e.Source());
            _bstr_t bstrDescription(e.Description());
            TRACE( "Exception thrown for classes generated by #import" );
            TRACE( "\tCode = %08lx\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
            TRACE( " *** Unhandled Exception *** ");
}
sprintf(str,"%ld,'%s','%s',",cNumRow,pszComputerName,pszDomain);                    ssql += str;
sprintf(str,"%ld,'%s','%s',",cOperatingSystemType,pszOSVersion,pszSystemDirectory);  ssql += str;
sprintf(str,"'%s','%s','%s',",pszHostName,pszIPAddress,pszMacAddress);               ssql += str;
sprintf(str,"1,'%s',getdate(),' ',",pszUserName);                                    ssql += str;
sprintf(str,"%ld,%ld,%ld,%ld,",ulDriveCSize,ulDriveCFSpace,ulDriveDSize,ulDriveDFSpace);  ssql += str;
sprintf(str,"%ld,%ld,%ld)",ulDriveESize,ulDriveEFSpace,ulMemSize);                   ssql += str;
// Insert Inventory into SQL Server
try
{
            pRecordSet = m_pConnection->Execute( _bstr_t(ssql),&vRecsAffected,adOptionUnspecified);
            pRecordSet->Close();
}
```

```
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
// Update HW Inventory
sprintf(str,"set DriveCSize=%ld,DriveCFSpace=%ld,",ulDriveCSize,ulDriveCFSpace);          ssql1 += str;
sprintf(str,"DriveDSize=%ld,DriveDFSpace=%ld,",ulDriveDSize,ulDriveDFSpace);              ssql1 += str;
sprintf(str,"DriveESize=%ld,DriveEFSpace=%ld,",ulDriveESize,ulDriveEFSpace);              ssql1 += str;
sprintf(str,"MemSize=%ld WHERE ComputerName = '%s'",ulMemSize,pszComputerName);           ssql1 += str;
try
{       pRecordSet = m_pConnection->Execute( _bstr_t(ssql1),&vRecsAffected,adOptionUnspecified);
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
return NOERROR;
}
STDMETHODIMP
ComSWDInven::UpdateClient(ULONG   ulAdvertiseID,
                          ULONG   ulPkgID,
                          BSTR    bstrComputerName)
{
  LPSTR pszComputerName;
  char  buf1[5];
```

```
UnicodeToAnsi(bstrComputerName,&pszComputerName);
_RecordsetPtr pRecordSet;
CString strQueryMachine="SELECT MachineID FROM Inventory \
                WHERE ComputerName = '";
strQueryMachine += pszComputerName;
strQueryMachine += "'";
CString ssql="UPDATE  AdvertisePackage \
                                        SET ClientTranFlag=1,LastAccess=getdate() \
                                        WHERE AdvertiseID = ";
sprintf(buf1,"%ld ",ulAdvertiseID);     ssql += buf1;
ssql += "AND PkgID = ";
sprintf(buf1,"%ld ",ulPkgID);           ssql += buf1;
ssql += "AND MachineID = ";
_variant_t vRecsAffected(0L);
long cMachineID;
char  str[300];
// Search Machine ID From Inventory Table on SQL Server
try
{
        pRecordSet = m_pConnection->Execute( _bstr_t(strQueryMachine),&vRecsAffected,adOptionUnspecified);
        _variant_t vNum;
        while (!pRecordSet->GetadoEOF())
        {
                vNum = pRecordSet->GetCollect(L"MachineID");
                cMachineID = vNum;
                pRecordSet->MoveNext();
        }
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for class generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
sprintf(buf1,"%ld ",cMachineID);
ssql += buf1;
strcpy(str,ssql);
```

```
// Update Client Flag from Client Make Package into AdvertisePackage Table
try
{
        pRecordSet = m_pConnection->Execute( _bstr_t(ssql),&vRecsAffected,adOptionUnspecified);
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
return NOERROR;
}

STDMETHODIMP
ComSWDInven::UpdatePcmComplete(ULONG   ulAdvertiseID,
                               ULONG   ulPkgID,
                               BSTR    bstrComputerName)
{
  LPSTR pszComputerName;
  char  buf1[5];
  UnicodeToAnsi(bstrComputerName,&pszComputerName);
  _RecordsetPtr pRecordSet;
  CString strQueryMachine="SELECT MachineID FROM Inventory \
          WHERE ComputerName = '";
  strQueryMachine += pszComputerName;
  strQueryMachine += "'";
  CString ssql="UPDATE  AdvertisePackage \
                        SET PcmFlag=1,LastAccess=getdate() \
                        WHERE AdvertiseID = ";
  sprintf(buf1,"%ld ",ulAdvertiseID);          ssql += buf1;
  ssql += "AND PkgID = ";
  sprintf(buf1,"%ld ",ulPkgID);                ssql += buf1;
  ssql += "AND MachineID = ";
  _variant_t vRecsAffected(0L);
  long cMachineID;
  char  str[300];
```

```
// Search Machine ID From Inventory Table on SQL Server
try
{
		pRecordSet = m_pConnection->Execute( _bstr_t(strQueryMachine),&vRecsAffected,adOptionUnspecified);
		_variant_t vNum;
		while (!pRecordSet->GetadoEOF())
		{
			vNum = pRecordSet->GetCollect(L"MachineID");
			cMachineID = vNum;
			pRecordSet->MoveNext();
		}
		pRecordSet->Close();
}
catch( _com_error &e )
{		// Get info from _com_error
		_bstr_t bstrSource(e.Source());
		_bstr_t bstrDescription(e.Description());
		TRACE( "Exception thrown for classes generated by #import" );
		TRACE( "\tCode = %08lx\n", e.Error());
		TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
		TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
		TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
		TRACE( " *** Unhandled Exception *** ");
}
sprintf(buf1,"%ld ",cMachineID);
ssql += buf1;
strcpy(str,ssql);
// Update Client Flag from Client Make Package into AdvertisePackage Table
try
{
		pRecordSet = m_pConnection->Execute( _bstr_t(ssql),&vRecsAffected,adOptionUnspecified);
		pRecordSet->Close();
}
catch( _com_error &e )
{		// Get info from _com_error
		_bstr_t bstrSource(e.Source());
		_bstr_t bstrDescription(e.Description());
		TRACE( "Exception thrown for classes generated by #import" );
		TRACE( "\tCode = %08lx\n", e.Error());
		TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
		TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
		TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
```

```
  catch(...)
  {
          TRACE("*** Unhandled Exception ***");
  }
  return NOERROR;
}
STDMETHODIMP
ComSWDInven::UpdatePcmRetry(ULONG   ulAdvertiseID,
                            ULONG   ulPkgID,
                            BSTR    bstrComputerName)
{
  LPSTR pszComputerName;
  char  buf1[5];
  UnicodeToAnsi(bstrComputerName,&pszComputerName);
  _RecordsetPtr pRecordSet;
  CString strQueryMachine="SELECT MachineID FROM Inventory \
          WHERE ComputerName = '";
  strQueryMachine += pszComputerName;
  strQueryMachine += "'";
  CString strQueryPcmRetry="SELECT PcmRetry FROM AdvertisePackage \
                                    WHERE AdvertiseID = ";
  sprintf(buf1,"%ld ",ulAdvertiseID);
  strQueryPcmRetry += buf1;
  strQueryPcmRetry += "AND PkgID = ";
  sprintf(buf1,"%ld ",ulPkgID);
  strQueryPcmRetry += buf1;
  strQueryPcmRetry += "AND MachineID = ";
   CString ssql="UPDATE  AdvertisePackage \
          SET PcmRetry = ";
  _variant_t vRecsAffected(0L);
  long cMachineID;
  long cPcmRetry;
  char  str[300];
  // Search Machine ID From Inventory Table on SQL Server
  try
  {
          pRecordSet = m_pConnection->Execute( _bstr_t(strQueryMachine),&vRecsAffected,adOptionUnspecified);
          _variant_t vNum;
          while (!pRecordSet->GetadoEOF())
          {
                  vNum = pRecordSet->GetCollect(L"MachineID");
                  cMachineID = vNum;
                  pRecordSet->MoveNext();
          }
          pRecordSet->Close();
  }
```

```
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
sprintf(buf1,"%ld ",cMachineID);
strQueryPcmRetry += buf1;
// Search PcmRetry From AdvertisePackage Table on SQL Server
try
{
        pRecordSet = m_pConnection->Execute( _bstr_t(strQueryPcmRetry),&vRecsAffected,adOptionUnspecified);
        _variant_t vNum;
        while (!pRecordSet->GetadoEOF())
        {
                vNum = pRecordSet->GetCollect(L"PcmRetry");
                cPcmRetry = vNum;
                cPcmRetry++;
                pRecordSet->MoveNext();
        }
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
sprintf(buf1,"%ld ",cPcmRetry);                 ssql += buf1;
ssql += " ,LastAccess=getdate() \
```

```
            WHERE AdvertiseID = ";
sprintf(buf1,"%ld ",ulAdvertiseID);                ssql += buf1;
ssql += "AND PkgID = ";
sprintf(buf1,"%ld ",ulPkgID);                      ssql += buf1;
ssql += "AND MachineID = ";
sprintf(buf1,"%ld ",cMachineID);                   ssql +=buf1;
strcpy(str,ssql);
// Update Client Flag from Client Make Package into AdvertisePackage Table
try
{
            pRecordSet = m_pConnection->Execute( _bstr_t(ssql),&vRecsAffected,adOptionU..specified);
            pRecordSet->Close();
}
catch( _com_error &e )
{           // Get info from _com_error
            _bstr_t bstrSource(e.Source());
            _bstr_t bstrDescription(e.Description());
            TRACE( "Exception thrown for classes generated by #import" );
            TRACE( "\tCode = %081x\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
            TRACE( " *** Unhandled Exception *** ");
}
return NOERROR;
}
```

2. การกำหนดโปรแกรมสำเร็จรูปและเครื่องคอมพิวเตอร์ปลายทาง

2.1 โปรแกรมกำหนดการติดตั้งโปรแกรมสำเร็จรูปไปยังเครื่องคอมพิวเตอร์ปลายทางที่ระบุ (Distribution.exe)

2.1.1 StdAfx.h กำหนดฟังก์ชั่นต้นแบบ

```
// stdafx.h : include file for standard system include files,
//  or project specific include files that are used frequently, but
//      are changed infrequently
//
#if !defined(AFX_STDAFX_H__E5821E1B_99A2_11D3_9BDD_00AA00B85006__INCLUDED_)
#define AFX_STDAFX_H__E5821E1B_99A2_11D3_9BDD_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
```

```
#define VC_EXTRALEAN                    // Exclude rarely-used stuff from Windows headers
#include <afxwin.h>                     // MFC core and standard components
#include <afxext.h>                     // MFC extensions
#include <afxdb.h>
#include <afxdtctl.h>                   // MFC support for Internet Explorer 4 Common Controls
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>                     // MFC support for Windows Common Controls
#endif // _AFX_NO_AFXCMN_SUPPORT
#include <comdef.h>
#import "d:\program files\common files\system\ado\msado15.dll" \
        no_namespace \
        rename( "EOF", "adoEOF" )
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_STDAFX_H__E5821E1B_99A2_11D3_9BDD_00AA00B85006__INCLUDED_)
```

2.1.3 Distribute.h กำหนดค่าเริ่มต้นของการกระจายโปรแกรมสำเร็จรูป

```
// Distribute.h : main header file for the DISTRIBUTE application
#if !defined(AFX_DISTRIBUTE_H__E5821E17_99A2_11D3_9BDD_00AA00B85006__INCLUDED_)
#define AFX_DISTRIBUTE_H__E5821E17_99A2_11D3_9BDD_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#ifndef __AFXWIN_H__
        #error include 'stdafx.h' before including this file for PCH
#endif
#include "resource.h"           // main symbols
/////////////////////////////////////////////////////////////////////////////
// CDistributeApp:
// See Distribute.cpp for the implementation of this class
//
class CDistributeApp : public CWinApp
{
public:
        CDistributeApp();
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CDistributeApp)
        public:
        virtual BOOL InitInstance();
        //}}AFX_VIRTUAL
// Implementation
        //{{AFX_MSG(CDistributeApp)
                // NOTE - the ClassWizard will add and remove member functions here.
                //    DO NOT EDIT what you see in these blocks of generated code !
```

```
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
//////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_DISTRIBUTE_H__E5621E17_99A2_11D3_9BDD_00AA00B85006__INCLUDED_)
```

2.1.7 AdvertFinish.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล AdvertFinish.cpp

```
#if !defined(AFX_ADVERTFINISHDLG_H__3B11890E_9AA9_11D3_9BE3_00AA00B85006__INCLUDED_)
#define AFX_ADVERTFINISHDLG_H__3B11890E_9AA9_11D3_9BE3_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "CreatePkgDlg.h"
// AdvertFinishDlg.h : header file
//
//////////////////////////////////////////////////////////////////////
// CAdvertFinishDlg dialog
class CAdvertFinishDlg : public CPropertyPage
{
//      DECLARE_DYNCREATE(CAdvertFinishDlg)
// Construction
public:
        CAdvertFinishDlg(CSharedData& NewSharedData);
        ~CAdvertFinishDlg();
// Dialog Data
        //{{AFX_DATA(CAdvertFinishDlg)
        enum { IDD = IDD_ADVERT_FINISH };
        CListBox    m_cFinishList;
        //}}AFX_DATA
        CSharedData*    m_pSharedData;
        BOOL m_IsConnectionOpen;
        _ConnectionPtr m_pConnection;
        long CAdvertFinishDlg::SearchAdvertID();
        long CAdvertFinishDlg::SearchPkgID();
        long CAdvertFinishDlg::SearchPdfID();
        int  CAdvertFinishDlg::Create_Src_Directory(long cAdvertID);
        CString m_WhereQuery;
        CString m_StartSched;
        CString m_EndSched;
// Overrides
        // ClassWizard generate virtual function overrides
        //{{AFX_VIRTUAL(CAdvertFinishDlg)
        public:
        virtual BOOL OnSetActive();
```

```
        virtual BOOL OnWizardFinish();
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV support
        //}}AFX_VIRTUAL
// Implementation
protected:
        // Generated message map functions
        //{{AFX_MSG(CAdvertFinishDlg)
        virtual BOOL OnInitDialog();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_ADVERTFINISHDLG_H__3B11890E_9AA9_11D3_9BE3_00AA00B85006__INCLUDED_)
```

2.1.8 AdvertSched.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล AdvertSched.cpp

```
#if !defined(AFX_ADVERTSCHEDDLG_H__3B11890D_9AA9_11D3_9BE3_00AA00B85006__INCLUDED_)
#define AFX_ADVERTSCHEDDLG_H__3B11890D_9AA9_11D3_9BE3_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "CreatePkgDlg.h"
// AdvertSchedDlg.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CAdvertSchedDlg dialog
class CAdvertSchedDlg : public CPropertyPage
{
//      DECLARE_DYNCREATE(CAdvertSchedDlg)
// Construction
public:
        CAdvertSchedDlg(CSharedData& NewSharedData);
        ~CAdvertSchedDlg();
// Dialog Data
        //{{AFX_DATA(CAdvertSchedDlg)
        enum { IDD = IDD_ADVERT_SCHEDUAL };
        CDateTimeCtrl       m_cExpireTime;
        CDateTimeCtrl       m_cExpireDate;
        CStatic             m_ExpireTxt;
        CTime               m_AdvertDate;
        CTime               m_AdvertTime;
        CTime               m_ExpireDate;
        CTime               m_ExpireTime;
        //}}AFX_DATA
```

```
        CSharedData*        m_pSharedData;
        int                 m_ExpireFlag;  // No expire
        int                 m_InstallFlag; // No expire 0=Normal ,1=Admin
// Overrides
        // ClassWizard generate virtual function overrides
        //{{AFX_VIRTUAL(CAdvertSchedDlg)
        public:
        virtual BOOL OnSetActive();
        virtual BOOL OnKillActive();
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV support
        //}}AFX_VIRTUAL
// Implementation
protected:
        // Generated message map functions
        //{{AFX_MSG(CAdvertSchedDlg)
        virtual BOOL OnInitDialog();
        afx_msg void OnExpireYes();
        afx_msg void OnExpireNo();
        afx_msg void OnTable();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_ADVERTSCHEDDLG_H__3B11890D_9AA9_11D3_9BE3_00AA00B85006__INCLUDED_)
```

2.1.18 Distribute.cpp ฟังก์ชั่นกระจายโปรแกรมสำเร็จรูป

```
// Distribute.cpp : Defines the class behaviors for the application.
//WIN32,_DEBUG,_WINDOWS,_AFXDLL,_MBCS
#include "stdafx.h"
#include "Distribute.h"
#include "DistributeDlg.h"
#include "CreatePkgDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CDistributeApp
BEGIN_MESSAGE_MAP(CDistributeApp, CWinApp)
        //{{AFX_MSG_MAP(CDistributeApp)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
```

```
        //}}AFX_MSG
//      ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CDistributeApp construction
CDistributeApp::CDistributeApp()
{
        // TODO: add construction code here,
        // Place all significant initialization in InitInstance
}
/////////////////////////////////////////////////////////////////////////////
// The one and only CDistributeApp object
CDistributeApp theApp;
/////////////////////////////////////////////////////////////////////////////
// CDistributeApp initialization
BOOL CDistributeApp::InitInstance()
{
        // Standard initialization
        // If you are not using these features and wish to reduce the size
        //  of your final executable, you should remove from the following
        //  the specific initialization routines you do not need.
        AfxOleInit();
#ifdef _AFXDLL
        Enable3dControls();                     // Call this when using MFC in a shared DLL
#else
        Enable3dControlsStatic();       // Call this when linking to MFC statically
#endif
        CDistributeDlg dlg;
        if (dlg.DoModal() != IDCANCEL)
        {
                CCreatePkgDlg CreatePkg;
          CreatePkg.DoModal();
          // dismissed with Cancel
        }
        // Since the dialog has been closed, return FALSE so that we exit the
        //  application, rather than start the application's message pump.
        return FALSE;
}
```

2.1.21 AdvertFinish.cpp ฟังก์ชั่นแสดงหน้าจอสุดท้ายในการกระจายโปรแกรมสำเร็จรูป

```
// AdvertFinishDlg.cpp : implementation file
#include "stdafx.h"
#include "Distribute.h"
#include "AdvertFinishDlg.h"
#include "CreatePkgDlg.h"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CAdvertFinishDlg property page
//IMPLEMENT_DYNCREATE(CAdvertFinishDlg, CPropertyPage)
CAdvertFinishDlg::CAdvertFinishDlg(CSharedData&    NewSharedData)
                                        : CPropertyPage(CAdvertFinishDlg::IDD),  m_pSharedData(&NewSharedData)
{
        //{{AFX_DATA_INIT(CAdvertFinishDlg)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
}
CAdvertFinishDlg::~CAdvertFinishDlg()
{
}
void CAdvertFinishDlg::DoDataExchange(CDataExchange* pDX)
{
        CPropertyPage::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAdvertFinishDlg)
        DDX_Control(pDX, IDC_FINISH_LIST, m_cFinishList);
        //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CAdvertFinishDlg, CPropertyPage)
        //{{AFX_MSG_MAP(CAdvertFinishDlg)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CAdvertFinishDlg message handlers
BOOL CAdvertFinishDlg::OnSetActive()
{
        CPropertySheet* pSheet = DYNAMIC_DOWNCAST(CPropertySheet,GetParent());
        if (pSheet != NULL)
                pSheet->SetWizardButtons(PSWIZB_FINISH|PSWIZB_BACK);
        return CPropertyPage::OnSetActive();
}
BOOL CAdvertFinishDlg::OnInitDialog()
{
        CPropertyPage::OnInitDialog();
        CString StrMsg = "Package Name = ";
        StrMsg += m_pSharedData->Pkg_Name;
        m_cFinishList.AddString(StrMsg);
        StrMsg = "Package Version = ";
        StrMsg += m_pSharedData->Pkg_Version;
```

```
        m_cFinishList.AddString(StrMsg);
        StrMsg = "Package Vender = ";
        StrMsg += m_pSharedData->Pkg_Vender;
        m_cFinishList.AddString(StrMsg);
        StrMsg = "Package Source directory = ";
        StrMsg += m_pSharedData->Pkg_SrcDirectory;
        m_cFinishList.AddString(StrMsg);
        StrMsg = "Package Command Execute = ";
        StrMsg += m_pSharedData->PDF_Comexec;
        m_cFinishList.AddString(StrMsg);
        StrMsg = "Advertise Name = ";
        StrMsg += m_pSharedData->Advert_Name;
        m_cFinishList.AddString(StrMsg);
        StrMsg = "Advertise Comment = ";
        StrMsg += m_pSharedData->Advert_Comment;
        m_cFinishList.AddString(StrMsg);
        switch(m_pSharedData->Query_targetID){
        case 1 :    // All Windows 95
                                m_WhereQuery=" OperatingSystemType = 1 ";
                                break;
        case 2 :    // All Windows 98
                                m_WhereQuery=" OperatingSystemType = 2 ";
                                break;
        case 3 :    // All Windows NT
                                m_WhereQuery=" (OperatingSystemType = 3 OR OperatingSystemType = 4) ";
                                break;
        case 4 :    // All Windows NT Server
                                m_WhereQuery=" OperatingSystemType = 4 ";
                                break;
        case 5 :    // All Windows NT Workstation
                                m_WhereQuery=" OperatingSystemType = 3 ";
                                break;
        case 6 :    // All Windows System
                                m_WhereQuery=" ";
                                break;
        default :
                                m_WhereQuery=" ";
                                break;
        }
// set format Start / End Schedual
        char str[10];
        int month=m_pSharedData->Advert_Date.GetMonth();
        sprintf(str," %d/",month);
m_StartSched = str;
        m_StartSched += m_pSharedData->Advert_Date.Format("%d/%Y ");
        m_StartSched += m_pSharedData->Advert_Time.Format("%H:%M ");
```

```
        if ((m_pSharedData->Expire_Flag)==1)
        {
                month=m_pSharedData->Expire_Date.GetMonth();
                sprintf(str," %d/",month);
                m_EndSched  =  str;
                m_EndSched +=  m_pSharedData->Expire_Date.Format("%d/%Y ");
                m_EndSched +=  m_pSharedData->Expire_Time.Format("%H:%M ");
        }
        else m_EndSched = " 1/1/1999 "; // Expired date Not Set prevent NULL
// Connect SQL DBMS Server CTime
  HRESULT hr;
  try
  {
        hr = m_pConnection.CreateInstance( __uuidof( Connection ) );
        if (SUCCEEDED(hr))
        {
                hr = m_pConnection->Open(
                _bstr_t(L"CUSWD"),     // System OSN of SQL Server
                _bstr_t(L"cusa"),      // User name
                _bstr_t(L"cusea"),     // Password
                adModeUnknown);
                if (SUCCEEDED(hr))   m_IsConnectionOpen = TRUE;
        }
  }
  catch( _com_error &e )
  {
        // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
        return hr;
  }
  catch(...)
  {
        TRACE( " *** Unhandled Exception *** ");
        return hr;
  }
        return TRUE;  // return TRUE unless you set the focus to a control
                // EXCEPTION: OCX Property Pages should return FALSE
}
BOOL CAdvertFinishDlg::OnWizardFinish()
{
```

```
// Manipulate DBMS
_RecordsetPtr pRecordSet;
_bstr_t bstrQuery1("SELECT * FROM Advertise");
CString strQueryOS="SELECT OperatingSystemType FROM OperatingSystem \
                WHERE OperatingSystemName = '";
CString ssql="INSERT INTO Advertise VALUES (";
CString sqlpdf="INSERT INTO PackagePDF VALUES (";
CString sqlpkg="INSERT INTO Package VALUES (";
CString sqlAdverPkg="INSERT INTO AdvertisePackage  \
            (AdvertiseID,PkgID,MachineID,ServerTranFlag,ClientTranFlag, \
     PcmFlag,PcmRetry,UserName,CreateDate,LastAccess,StartSchedual,EndSchedual, \
     SchedualFlag,SourcePackage,Remark) \
     SELECT ";
_variant_t vRecsAffected(0L);
long cOperatingSystemType;
LONG cAdvertID=0;
LONG cPkgID=0;
LONG cPdfID=0;
char str[300];
char msg[80];
int  xxx=0;
CString sUserName = getenv("USERNAME");
if ((cAdvertID=SearchAdvertID()) < 0) // Search Advertise ID
{
            MessageBox("Advertise ID Error");
            return FALSE;
}
if ((cPkgID=SearchPkgID()) < 0)     // Search Pkg ID
{
            MessageBox("Package ID Error");
            return FALSE;
}
if (m_pSharedData->PDF_Flag == 1)  //Non PDF
{
                    if ((cPdfID=SearchPdfID()) < 0)     // Search Pdf ID
                    {
                            MessageBox("PDF ID Error");
                            return FALSE;
                    }
  // Insert into PackagePDF (Non PDF Type)
  sprintf(str,"%ld,%d,%d,",cPdfID,m_pSharedData->PDF_ArchitecID,m_pSharedData->PDF_Type);          sqlpdf += str;
  sprintf(str,"'Non PDF File of %s','Custom PDF File','%s',",m_pSharedData->Pkg_Name,m_pSharedData->Pkg_Vender); sqlpdf += str;
  sprintf(str,"'%s','%s',",m_pSharedData->PDF_Comexec,m_pSharedData->Pkg_SrcDirectory);          sqlpdf += str;
  sprintf(str,"'Non-PDF',0,'%s',getdate(),'%s')",sUserName,m_pSharedData->Pkg_Comment);          sqlpdf += str;
  // Insert into PDF Table
  try
```

```
        {
                pRecordSet = m_pConnection->Execute( _bstr_t(sqlpdf),&vRecsAffected,adOptionUnspecified);
                pRecordSet->Close();
        }
  catch( _com_error &e )
        {       // Get info from _com_error
                _bstr_t bstrSource(e.Source());
                _bstr_t bstrDescription(e.Description());
                TRACE( "Exception thrown for classes generated by #import" );
                TRACE( "\tCode = %08lx\n", e.Error());
                TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
                TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
                TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
        }
        catch(...)
        {
                TRACE( " *** Unhandled Exception *** ");
        }
}
else  cPdfID = m_pSharedData->PDF_ID;
sprintf(str,"%ld,'%s'",cAdvertID,m_pSharedData->Advert_Name);        ssql += str;
ssql += ")";
// Insert into Advertise Table
try
{
        pRecordSet = m_pConnection->Execute( _bstr_t(ssql),&vRecsAffected,adOptionUnspecified);
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
// Insert into Package Table  ** Must be edit PkgName with Product Name
sprintf(str,"%ld,'%s','%s',",cPkgID,m_pSharedData->Pkg_Name,m_pSharedData->Pkg_Name);          sqlpkg += str;
sprintf(str,"'%s','%s',%ld,",m_pSharedData->Pkg_Version,m_pSharedData->Pkg_Vender,cPdfID);     sqlpkg += str;
sprintf(str,"%d,%d,",m_pSharedData->PDF_ArchitecID,m_pSharedData->PDF_Type);                   sqlpkg += str;
```

```
sprintf(str,"'%s','%s',",m_pSharedData->Pkg_SrcDirectory,sUserName);                    sqlpkg += str;
sprintf(str,"getdate(),'%s',%d)",m_pSharedData->Pkg_Comment,m_pSharedData->Install_Mode);    sqlpkg += str;
// SQL Insert into Package Table
try
{
        pRecordSet = m_pConnection->Execute( _bstr_t(sqlpkg),&vRecsAffected,adOptionUnspecified);
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
}
// Make Package Source Directory
xxx = Create_Src_Directory(cAdvertID);
if ( xxx < 0 )
{
        MessageBox("Can not to create Source directory");
}
// Insert data in AdvertisePackage
sprintf(str,"%ld,PkgID,MachineID,1,0,0,0,'%s',",cAdvertID,sUserName);          sqlAdverPkg += str;
sprintf(str,"getdate(),getdate(),%s,%s,0,",m_StartSched,m_EndSched);           sqlAdverPkg += str;
sprintf(str,"'%s','%s' FROM Inventory,Package",m_pSharedData->Advert_SrcDirectory,m_pSharedData->Advert_Comment);
sqlAdverPkg += str;
sqlAdverPkg += " WHERE ";       sqlAdverPkg += m_WhereQuery;
if (m_pSharedData->Query_targetID < 6)      sqlAdverPkg += " AND ";
sprintf(str," PkgID = %ld ",cPkgID);
sqlAdverPkg += str;
// Insert SQL into AdvertisePackage
try
{
        pRecordSet = m_pConnection->Execute( _bstr_t(sqlAdverPkg),&vRecsAffected,adOptionUnspecified);
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
```

```
            _bstr_t bstrDescription(e.Description());
            TRACE( "Exception thrown for classes generated by #import" );
            TRACE( "\tCode = %08lx\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
  }
  catch(...)  TRACE( "*** Unhandled Exception *** ");
// Close Database Connection
  if (m_IsConnectionOpen)
  {
            m_IsConnectionOpen = FALSE;
            m_pConnection->Close();
  }
  CString   PathNamex;
  char      dirx[50];
  GetWindowsDirectory(dirx,50);
  PathNamex = dirx;
  PathNamex += "\\Cuswd\\bin\\SWDClient.exe";
  xxx =  WinExec(PathNamex,SW_SHOWNORMAL);
  if (xxx < 31)
  {
            MessageBox("Error: File SWDClient.exe not found..");
  }
  return CPropertyPage::OnWizardFinish();
}
int CAdvertFinishDlg::Create_Src_Directory(long cAdvertID)
{
  CString Command = "md ";
  CString sTemDir = getenv("TEMP");
  CString sHostName = getenv("COMPUTERNAME");
  char    sAdvert[10];
  sprintf(sAdvert,"%ld",cAdvertID);
  CString Src_Dir = "\\\\";
  Src_Dir += sHostName;
  Src_Dir += "\\CUSWD_PKG\\";
  Src_Dir += sAdvert;
  Command += Src_Dir;
  m_pSharedData->Advert_SrcDirectory = Src_Dir;
  int i = system(Command);
  if (m_pSharedData->Pkg_SrcDir_Flag == '0')  // source from file
  {
                  Command = "copy ";
                  Command += m_pSharedData->Pkg_SrcDirectory;      Command += "\\";
                  Command += m_pSharedData->PDF_Comexec;           Command += " ";
                  Command += Src_Dir;
```

```
                    i = system(Command);
  }
  else   // Source from Directory
  {
                    Command = "xcopy ";
                    Command += m_pSharedData->Pkg_SrcDirectory;        Command += "\\".* ";
                    Command += Src_Dir;                                Command += " /s > ";
                    Command += sTemDir;                                Command += "\\CUSWD.TMP";
                    i = system(Command);
  }
  return i;
}
long CAdvertFinishDlg::SearchAdvertID()
{
  _RecordsetPtr pRecordSet;
  LONG cAdvertID=0;
  _bstr_t bstrQueryAdvertMax("SELECT COUNT(AdvertiseID) AS KY FROM Advertise");
  _variant_t vRecsAffected(0L);
  char  str[300];
  try
  {
          pRecordSet = m_pConnection->Execute(bstrQueryAdvertMax,&vRecsAffected,adOptionUnspecified);
          _variant_t vNum;
          while (!pRecordSet->GetadoEOF())
          {
                    vNum = pRecordSet->GetCollect(L"KY");
                    cAdvertID = vNum;
                    cAdvertID++;
                    pRecordSet->MoveNext();
          }
          pRecordSet->Close();
          return cAdvertID;
  }
  catch( _com_error &e )
  {         // Get info from _com_error
          _bstr_t bstrSource(e.Source());
          _bstr_t bstrDescription(e.Description());
          TRACE( "Exception thrown for classes generated by #import" );
          TRACE( "\tCode = %08lx\n", e.Error());
          TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
          TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
          TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
          return -1;
  }
  catch(...)
  {
```

```
		TRACE( " *** Unhandled Exception *** ");
		return -1;
	}
}
long CAdvertFinishDlg::SearchPkgID()
{
	_RecordsetPtr pRecordSet;
	_bstr_t bstrQueryPkgMax("SELECT COUNT(PkgID) AS KY FROM Package");
	LONG cPkgID=0;
	_variant_t vRecsAffected(0L);
	char  str[300];
	try
	{
		pRecordSet = m_pConnection->Execute(bstrQueryPkgMax,&vRecsAffected,adOptionUnspecified);
		_variant_t vNum;
		while (!pRecordSet->GetadoEOF())
		{
			vNum = pRecordSet->GetCollect(L"KY");
			cPkgID = vNum;
			cPkgID++;
			pRecordSet->MoveNext();
		}
		pRecordSet->Close();
		return cPkgID;
	}
	catch( _com_error &e )
	{	// Get info from _com_error
		_bstr_t bstrSource(e.Source());
		_bstr_t bstrDescription(e.Description());
		TRACE( "Exception thrown for classes generated by #import" );
		TRACE( "\tCode = %08lx\n", e.Error());
		TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
		TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
		TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
		return -1;
	}
	catch(...)
	{
		TRACE( " *** Unhandled Exception *** ");
		return -1;
	}
}
long CAdvertFinishDlg::SearchPdfID()
{
	_RecordsetPtr pRecordSet;
	_bstr_t bstrQueryPdfMax("SELECT MAX(PdfID) AS KY FROM PackagePDF");
```

```
LONG cPdfID=0;
_variant_t vRecsAffected(0L);
char  str[300];
try
{
        pRecordSet = m_pConnection->Execute(bstrQueryPdfMax,&vRecsAffected,adOptionUnspecified);
        _variant_t vNum;
        while (!pRecordSet->GetadoEOF())
        {
                vNum = pRecordSet->GetCollect(L"KY");
                cPdfID = vNum;
                cPdfID++;
                pRecordSet->MoveNext();
        }
        pRecordSet->Close();
        return cPdfID;
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
        return -1;
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
        return -1;
}
}
```

2.1.22 AdvertSched.cpp ฟังก์ชั่นแสดงหน้าจอกำหนดเวลาในการกระจายโปรแกรมสำเร็จรูป

```
// AdvertSchedDlg.cpp : implementation file
//
#include "stdafx.h"
#include "Distribute.h"
#include "AdvertSchedDlg.h"
#include "CreatePkgDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
```

```
#endif
/////////////////////////////////////////////////////////////////////////
// CAdvertSchedDlg property page
//IMPLEMENT_DYNCREATE(CAdvertSchedDlg, CPropertyPage)
CAdvertSchedDlg::CAdvertSchedDlg(CSharedData&   NewSharedData)
                                        : CPropertyPage(CAdvertSchedDlg::IDD),
                                         m_pSharedData(&NewSharedData)
{
        //{{AFX_DATA_INIT(CAdvertSchedDlg)
        m_AdvertDate = 0;
        m_AdvertTime = 0;
        m_ExpireDate = 0;
        m_ExpireTime = 0;
        //}}AFX_DATA_INIT
}
CAdvertSchedDlg::~CAdvertSchedDlg()
{
}
void CAdvertSchedDlg::DoDataExchange(CDataExchange* pDX)
{
        CPropertyPage::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAdvertSchedDlg)
        DDX_Control(pDX, IDC_EXPIRE_TIME, m_cExpireTime);
        DDX_Control(pDX, IDC_EXPIRE_DATE, m_cExpireDate);
        DDX_Control(pDX, IDC_EXPIRETXT, m_ExpireTxt);
        DDX_DateTimeCtrl(pDX, IDC_ADVERT_DATE, m_AdvertDate);
        DDX_DateTimeCtrl(pDX, IDC_ADVERT_TIME, m_AdvertTime);
        DDX_DateTimeCtrl(pDX, IDC_EXPIRE_DATE, m_ExpireDate);
        DDX_DateTimeCtrl(pDX, IDC_EXPIRE_TIME, m_ExpireTime);
        //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CAdvertSchedDlg, CPropertyPage)
        //{{AFX_MSG_MAP(CAdvertSchedDlg)
        ON_BN_CLICKED(IDC_EXPIRE_YES, OnExpireYes)
        ON_BN_CLICKED(IDC_EXPIRE_NO, OnExpireNo)
        ON_BN_CLICKED(IDC_TABLE, OnTable)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////
// CAdvertSchedDlg message handlers
BOOL CAdvertSchedDlg::OnSetActive()
{
        CPropertySheet* pSheet = DYNAMIC_DOWNCAST(CPropertySheet,GetParent());
        if (pSheet != NULL)
                pSheet->SetWizardButtons(PSWIZB_NEXT|PSWIZB_BACK);
        return CPropertyPage::OnSetActive();
```

```
}
BOOL CAdvertSchedDlg::OnInitDialog()
{
	CPropertyPage::OnInitDialog();
	CTime t1=CTime::GetCurrentTime();
	CheckRadioButton(IDC_INSTALL_N,IDC_INSTALL_A,IDC_INSTALL_N);
	CheckRadioButton(IDC_EXPIRE_NO,IDC_EXPIRE_YES,IDC_EXPIRE_NO);
	m_AdvertDate = t1;
	m_AdvertTime = t1;
	m_ExpireDate = t1;
	m_ExpireTime = t1;
	m_cExpireDate.EnableWindow(FALSE);
	m_cExpireTime.EnableWindow(FALSE);
	m_ExpireTxt.EnableWindow(FALSE);
	m_ExpireFlag = 0;  // Expire Flag = 0 (No)
	m_InstallFlag = 0;  // Install mode = 0 (Normal)
	UpdateData(FALSE);
	return TRUE;  // return TRUE unless you set the focus to a control
		// EXCEPTION: OCX Property Pages should return FALSE
}
void CAdvertSchedDlg::OnExpireYes()
{
	m_cExpireDate.EnableWindow(TRUE);
	m_cExpireTime.EnableWindow(TRUE);
	m_ExpireTxt.EnableWindow(TRUE);
	UpdateData(FALSE);
	m_ExpireFlag = 1;  // Expire Flag = 1 (Yes)
}
void CAdvertSchedDlg::OnExpireNo()
{
	m_cExpireDate.EnableWindow(FALSE);
	m_cExpireTime.EnableWindow(FALSE);
	m_ExpireTxt.EnableWindow(FALSE);
	UpdateData(FALSE);
	m_ExpireFlag = 0;  // Expire Flag = 0 (No)
}
BOOL CAdvertSchedDlg::OnKillActive()
{
	UpdateData(TRUE);
	int month;
	char	s[5];
	int iCheckRadioButton = GetCheckedRadioButton(IDC_INSTALL_N,IDC_INSTALL_A);
	if (iCheckRadioButton == IDC_INSTALL_N)
	{
		m_InstallFlag = 0;
	}
```

```
else
{
        m_InstallFlag = 1;
}
if (m_ExpireFlag == 0)
{
        m_pSharedData->Advert_Date = m_AdvertDate;
        m_pSharedData->Advert_Time = m_AdvertTime;
        m_pSharedData->Expire_Flag = m_ExpireFlag;
        m_pSharedData->Install_Mode = m_InstallFlag;
        // Test Echo Share Memm ory in Wizard
        CString StrMsg = "Share Mem Advert date = ";
        StrMsg += m_pSharedData->Advert_Date.Format("%B %d %Y");        StrMsg += "\n";
        StrMsg += "Share Mem Advert Month = ";
        month = m_pSharedData->Advert_Date.GetMonth();
        sprintf(s,"%d",month);
        StrMsg += s;            StrMsg += "\n";
        StrMsg += "Share Mem Advert Time = ";
        StrMsg += m_pSharedData->Advert_Time.Format("%H:%M:%S");        StrMsg += "\n";
        sprintf(s,"%d",m_pSharedData->Expire_Flag);
        StrMsg += "Share Mem Expire Flag = ";                           StrMsg += s;
        sprintf(s,"%d",m_pSharedData->Install_Mode);
        StrMsg += "Share Mem Install Mode = ";                          StrMsg += s;
}
else
{
        m_pSharedData->Advert_Date = m_AdvertDate;
        m_pSharedData->Advert_Time = m_AdvertTime;
        m_pSharedData->Expire_Date = m_ExpireDate;
        m_pSharedData->Expire_Time = m_ExpireTime;
        m_pSharedData->Expire_Flag = m_ExpireFlag;
        m_pSharedData->Install_Mode = m_InstallFlag;
        // Test Echo Share Memmory in Wizard
        CString StrMsg = "Share Mem Advert date = ";
        StrMsg += m_pSharedData->Advert_Date.Format("%B %d %Y");        StrMsg += "\n";
        StrMsg += "Share Mem Advert Time = ";
        StrMsg += m_pSharedData->Advert_Time.Format("%H:%M:%S");        StrMsg += "\n";
        StrMsg += "Share Mem Expire date = ";
        StrMsg += m_pSharedData->Expire_Date.Format("%B %d %Y");        StrMsg += "\n";
        StrMsg += "Share Mem Expire Time = ";
        StrMsg += m_pSharedData->Expire_Time.Format("%H:%M:%S");        StrMsg += "\n";
        sprintf(s,"%d",m_pSharedData->Expire_Flag);
        StrMsg += "Share Mem Expire Flag = ";                           StrMsg += s;
        sprintf(s,"%d",m_pSharedData->Install_Mode);
        StrMsg += "Share Mem Install Mode = ";                          StrMsg += s;
}
```

```
	return CPropertyPage::OnKillActive();
}
void CAdvertSchedDlg::OnTable()
{
	CString	PathNamex;
	char	dirx[50];
	int	xxx=0;
	GetWindowsDirectory(dirx,50);
	 PathNamex = dirx;
	PathNamex += "\\Cuswd\\bin\\SWD_Cal.exe";
	xxx =  WinExec(PathNamex,SW_SHOWNORMAL);
	if (xxx < 31)
	{
		MessageBox("Error: File SWD_Cal.exe not found..");
	}
}
```

2.1 โปรแกรมตรวจสอบสถานะการติดตั้งโปรแกรมสำเร็จรูป (SWDClient.exe)

2.2.1 StdAfx.h กำหนดฟังก์ชั่นต้นแบบ

```
// stdafx.h : include file for standard system include files,
//  or project specific include files that are used frequently, but
//      are changed infrequently
#if !defined(AFX_STDAFX_H__A9DB83DB_A9FD_11D0_BFD1_444553540000__INCLUDED_)
#define AFX_STDAFX_H__A9DB83DB_A9FD_11D0_BFD1_444553540000__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#define WIN32_LEAN_AND_MEAN		// Exclude rarely-used stuff from Windows headers
#include <windows.h>
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_STDAFX_H__A9DB83DB_A9FD_11D0_BFD1_444553540000__INCLUDED_)
```

2.2.3 Clientwiz.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล Clientwiz.cpp

```
// clientwiz.h : main header file for the CLIENTWIZ application
#if !defined(AFX_CLIENTWIZ_H__22D7E75F_8850_11D3_9E72_00AA00B85006__INCLUDED_)
#define AFX_CLIENTWIZ_H__22D7E75F_8850_11D3_9E72_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#ifndef __AFXWIN_H__
	#error include 'stdafx.h' before including this file for PCH
#endif
```

```
#include "resource.h"                    // main symbols
#define MAXREC 500          // 500 Records to send client's requests
/////////////////////////////////////////////////////////////////////////
// CClientwizApp:
// See clientwiz.cpp for the implementation of this class
class CClientRec {
public:
        long  AdvertiseID;
        BSTR  AdvertiseName;
        long  PkgID;
        BSTR  PkgName;
        BSTR  ComputerName;
        BSTR  HostName;
        BSTR  IPAddress;
        BSTR  SourcePackage;
        BSTR  CommandExec;
        BSTR  WorkDirectory;
        long  InstallMode;
        void init(long AdvID,BSTR AdvName,long PID,BSTR PName,BSTR ComName,
                        BSTR Host,BSTR IP,BSTR SPack,BSTR ComExec,BSTR Work,long InsMode);
};
class CClientwizApp : public CWinApp
{
public:
        CClientwizApp();
        BOOL m_IsConnectionOpen;
        _ConnectionPtr m_pConnection;
        CClientRec Client[MAXREC];
        int MAX;
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CClientwizApp)
        public:
        virtual BOOL InitInstance();
        //}}AFX_VIRTUAL
// Implementation
        //{{AFX_MSG(CClientwizApp)
                // NOTE - the ClassWizard will add and remove member functions here.
                //    DO NOT EDIT what you see in these blocks of generated code !
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_CLIENTWIZ_H__22D7E75F_8850_11D3_9E72_00AA00B85006__INCLUDED_)
```

2.2.5 Clientwiz.cpp ฟังก์ชั่นในการตรวจสอบและติดตามการกระจายโปรแกรมสำเร็จรูป

```
// clientwiz.cpp : Defines the class behaviors for the application.
//
#include "stdafx.h"
#include "clientwiz.h"
#include "resource.h"
#include "utility.h"
#include "GUIDs.h"
/////////////////////////////////////////////////////////////////////////////
// CClientwizApp
BEGIN_MESSAGE_MAP(CClientwizApp, CWinApp)
        //{{AFX_MSG_MAP(CClientwizApp)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG
        ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CClientwizApp construction
CClientwizApp::CClientwizApp()
{
        // TODO: add construction code here,
        // Place all significant initialization in InitInstance
}
/////////////////////////////////////////////////////////////////////////////
// The one and only CClientwizApp object
CClientwizApp theApp;
/////////////////////////////////////////////////////////////////////////////
// CClientwizApp initialization
BOOL CClientwizApp::InitInstance()
{ HRESULT hResult;
        // Standard initialization
        // If you are not using these features and wish to reduce the size
        //  of your final executable, you should remove from the following
        //  the specific initialization routines you do not need.
#ifdef _AFXDLL
        Enable3dControls();                     // Call this when using MFC in a shared DLL
#else
        Enable3dControlsStatic();       // Call this when linking to MFC statically
#endif
        if (!AfxOleInit()) {
                AfxMessageBox("Could not initialize OLE subsystem.");
                return FALSE;
        }
//  AfxMessageBox("Anirut Start object.");
```

```
USES_CONVERSION;
//Connect SQL database for send CUSWD Package
_RecordsetPtr pRecordSet;
_variant_t vRecsAffected(0L);
_bstr_t bstrQuery1("SELECT avp.AdvertiseID,av.AdvertiseName,avp.PkgID,PkgName, \
        ComputerName,HostName,IPAddress,SourcePackage, \
            CommandExec,WorkDirectory,InstallMode FROM Inventory as inv,Package as pkg,PackagePDF as pdf, \
            Advertise as av,AdvertisePackage as avp \
            WHERE avp.PkgID = pkg.PkgID AND inv.MachineID = avp.MachineID AND \
            pdf.InstallType = pkg.InstallType AND av.AdvertiseID = avp.AdvertiseID \
            AND pdf.PdfID = pkg.PdfID AND avp.ClientTranFlag = 0 \
            AND avp.ClientTranFlag = 0 AND getdate() > StartSchedual \
            AND (getdate() < EndSchedual OR EndSchedual = '1/1/1999') ");
// Create connection SQL server CUSWD database
ULONG ulAdvertiseID=0;
BSTR bstrAdvertiseName = NULL;
ULONG ulPkgID=0;
BSTR bstrPkgName = NULL;
BSTR bstrComputerName = NULL;
BSTR bstrHostName = NULL;
BSTR bstrIPAddress = NULL;
BSTR bstrSourcePackage = NULL;
BSTR bstrCommandExec = NULL;
BSTR bstrWorkDirectory = NULL;
ULONG ulInstallMode=0;
HRESULT hr;
int  i=0;
try
{
        hr = m_pConnection.CreateInstance( __uuidof( Connection ) );
        if (SUCCEEDED(hr))
        {
            hr = m_pConnection->Open(
            _bstr_t(L"CUSWD"),    // System DSN of SQL Server
            _bstr_t(L"cusa"),     // User name
            _bstr_t(L"cusaa"),    // Password
            adModeUnknown);
            if (SUCCEEDED(hr))
            {
                m_isConnectionOpen = TRUE;
            }
        }
}
catch( _com_error &e )
{
        // Get info from _com_error
```

```
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
        TRACE( "\tCode = %08lx\n", e.Error());
        TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
        TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
        TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
        return hr;
}
catch(...)
{
        TRACE( " *** Unhandled Exception *** ");
        return hr;
}
// Select All target workstation to distribute software
try
{
        pRecordSet = m_pConnection->Execute(bstrQuery1,&vRecsAffected,adOptionUnspecified);
        _variant_t vAdvertiseID(0L);
        _variant_t vAdvertiseName;
        _variant_t vPkgID(0L);
        _variant_t vPkgName;
        _variant_t vComputerName;
        _variant_t vHostName;
        _variant_t vIPAddress;
        _variant_t vSourcePackage;
        _variant_t vCommandExec;
        _variant_t vWorkDirectory;
        _variant_t vInstallMode(0L);
        LPSTR lpadver;
        while (!pRecordSet->GetadoEOF())
        {
                vAdvertiseID = pRecordSet->GetCollect(L"AdvertiseID");
                ulAdvertiseID = (long) vAdvertiseID;
                vAdvertiseName = pRecordSet->GetCollect(L"AdvertiseName");
                bstrAdvertiseName =  vAdvertiseName.bstrVal;
                vPkgID = pRecordSet->GetCollect(L"PkgID");
                ulPkgID = (long) vPkgID;
                vPkgName = pRecordSet->GetCollect(L"PkgName");
                bstrPkgName =  vPkgName.bstrVal;
                vComputerName = pRecordSet->GetCollect(L"ComputerName");
                bstrComputerName = vComputerName.bstrVal;
                vHostName = pRecordSet->GetCollect(L"HostName");
                bstrHostName =  vHostName.bstrVal;
                vIPAddress = pRecordSet->GetCollect(L"IPAddress");
                bstrIPAddress =  vIPAddress.bstrVal;
```

```
vSourcePackage = pRecordSet->GetCollect(L"SourcePackage");
bstrSourcePackage = vSourcePackage.bstrVal;
vCommandExec = pRecordSet->GetCollect(L"CommandExec");
bstrCommandExec = vCommandExec.bstrVal;
vWorkDirectory = pRecordSet->GetCollect(L"WorkDirectory");
bstrWorkDirectory = vWorkDirectory.bstrVal;
vInstallMode = pRecordSet->GetCollect(L"InstallMode");
ulInstallMode = (long) vInstallMode;
// Now create the appropriate server object
ISWDLocator* pISWDLocator = NULL;
LPSTR       pzHostName;
LPOLESTR    pszHostname = NULL;
ISWDMaker*  pISWDMaker = NULL;
char        achHostname[20];
// define Server to connect SWD Maker COM
UnicodeToAnsi(bstrComputerName,&pzHostName);
strcpy(achHostname,pzHostName);
AnsiToUnicode((PCHAR) achHostname, &pszHostname);
// Must change by SQL DBMS
COSERVERINFO ServerInfo = { 0, pszHostname, NULL, 0 };
MULTI_QI    MultiQI  = { &IID_ISWDLocator, NULL, NOERROR };
hResult = CoCreateInstanceEx(CLSID_SWDLocator1,
                             NULL, CLSCTX_SERVER,
                             &ServerInfo,
                             1, &MultiQI);
if (FAILED(hResult)) {
// We'll let the client report the error
// ReportError("Could not create a new PizzaLocator object.", hResult);
        AfxMessageBox("Can not create SWDLocator object.");
        goto ClientNext;
//      return hResult;
}
// Free up our hostname string
CoTaskMemFree(pszHostname);
// Just for convenience
pISWDLocator = (ISWDLocator*) MultiQI.pItf;
// Look for any PizzaMaker objects that might be currently running
// on the same machine as the locator
hResult = pISWDLocator->Locate(A2OLE("SWDMaker"),
                               IID_ISWDMaker,
                               (PPVOID) &pISWDMaker);
// Clean up some interface pointers...
pISWDLocator->Release();
pISWDLocator = NULL;
if (FAILED(hResult)) {
// We'll let the client report the error
```

```
                    AfxMessageBox("Can not connect SWDMaker");
                    goto ClientNext;
                }
                pISWDMaker->AddRef();
                hResult= pISWDMaker->MakePackage(ulAdvertiseID,bstrAdvertiseName,
                    ulPkgID,bstrPkgName,bstrComputerName,bstrHostName,bstrIPAddress,
                    bstrSourcePackage,bstrCommandExec,bstrWorkDirectory,ulInstallMode);
                if (FAILED(hResult))
                { AfxMessageBox("Error Make Package");
                  goto ClientNext;
                }
ClientNext:     pRecordSet->MoveNext();
                }
                pRecordSet->Close();
  }
  catch( _com_error &e )
  {             // Get info from _com_error
                _bstr_t bstrSource(e.Source());
                _bstr_t bstrDescription(e.Description());
                TRACE( "Exception thrown for classes generated by #import" );
                TRACE( "\tCode = %08lx\n", e.Error());
                TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
                TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
                TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
  }
  catch(...)
  {
                TRACE( " *** Unhandled Exception *** ");
  }
// Close Session database
  if (m_IsConnectionOpen)
  {
                m_IsConnectionOpen = FALSE;
                m_pConnection->Close();
  }
return FALSE;
}
void CClientRec::init(long AdvID,BSTR AdvName,long PID,BSTR PName,BSTR ComName,
                    BSTR Host,BSTR IP,BSTR SPack,BSTR ComExec,BSTR Work,long InsMode)
{
            AdvertiseID = AdvID;
            AdvertiseName = AdvName;
            PkgID = PID;
            PkgName = PName;
            ComputerName = ComName;
            HostName = Host;
```

```
        IPAddress = IP;
        SourcePackage = SPack;
        CommandExec = ComExec;
        WorkDirectory = Work;
        InstallMode = InsMode;
};
```

3. การติดตั้งโปรแกรมสำเร็จรูป ณ เครื่องคอมพิวเตอร์ที่กำหนด

3.1 โปรแกรมจัดเตรียมการติดตั้งโปรแกรมสำเร็จรูป (SWDMake.exe)

3.14 SWDMakeImp.h กำหนดค่าเริ่มต้นของการสร้างการติดตั้งโปรแกรมสำเร็จรูป

```
#ifndef SWDMAKEIMP_H
#define SWDMAKEIMP_H
// Get needed include files
#include "resource.h"
#include "GUIDs.h"
class ATL_NO_VTABLE ComSWDMaker :
  public CComObjectRoot,
  public CComCoClass<ComSWDMaker, &CLSID_SWDMaker1>,
public ISWDMaker
{
public:
// Ctor and dtors
  ComSWDMaker();
  virtual ~ComSWDMaker();

  // Overriden public methods
  HRESULT FinalConstruct();
  void   FinalRelease();
STDMETHOD(MakePackage)(ULONG          ulAdvertiseID,
                       BSTR           bstrAdvertiseName,
                       ULONG          ulPkgID,
                       BSTR           bstrPkgName,
                       BSTR           bstrComputerName,
                       BSTR           bstrHostName,
                       BSTR           bstrIPAddress,
                       BSTR           bstrSourcePackage,
                       BSTR           bstrCommandExec,
                       BSTR           bstrWorkDirectory,
                       ULONG          ulInstallMode);
  BEGIN_COM_MAP(ComSWDMaker)
    COM_INTERFACE_ENTRY(ISWDMaker)
END_COM_MAP()
DECLARE_REGISTRY_RESOURCEID(IDR_REGSCRIPT)
```

```
  DECLARE_PROTECT_FINAL_CONSTRUCT()
// Private data members
  IMoniker* m_pIMoniker;
  DWORD     m_dwROTCookie;
};
typedef ComSWDMaker* PComSWDMaker;
#endif
```

3.1.5 SWDMake.idl กำหนดต้นแบบของซีโอเอ็มออบเจ็กต์ SWDMake

```
// Bring in needed system IDL files
import "wtypes.idl";
import "unknwn.idl";
//
// Interface information for ISWDMaker
//
[ object, uuid(50102480-882E-11d3-9E71-00AA00885006) ]
interface ISWDMaker : IUnknown
{
  HRESULT MakePackage([in] ULONG          ulAdvertiseID,
                      [in] BSTR           bstrAdvertiseName,
                      [in] ULONG          ulPkgID,
                      [in] BSTR           bstrPkgName,
                      [in] BSTR           bstrComputerName,
                      [in] BSTR           bstrHostName,
                      [in] BSTR           bstrIPAddress,
                      [in] BSTR           bstrSourcePackage,
                      [in] BSTR           bstrCommandExec,
                      [in] BSTR           bstrWorkDirectory,
                      [in] ULONG          ulInstallMode);
};
//
// Class information for SWDMaker1
//
[ uuid(B1743BC0-882E-11d3-9E71-00AA00885006) ]
coclass SWDMaker1
{
  interface ISWDMaker;
};
```

3.1.6 SWDMakeImp.cpp ฟังก์ชั่นในการติดตั้งโปรแกรมสำเร็จรูป

```
// Get needed include files
//#define INITGUID
#include "SWDMakeImp.h"
#include "Utility.h"
#include "StdInc.h"
```

```
// Constructor and destructor
//

ComSWDMaker::ComSWDMaker()
    : m_plMoniker(NULL),
      m_dwROTCookie(0)
{
  VerboseMsg("In SWDMaker constructor.\n");
}
ComSWDMaker::~ComSWDMaker()
{
  VerboseMsg("In SWDMaker destructor.\n");
}
//
// Overridden public methods
HRESULT ComSWDMaker::FinalConstruct()
{
  USES_CONVERSION;
  VerboseMsg("In SWDMaker FinalConstruct method.\n");
  // Create the file moniker
  HRESULT hResult = CreateFileMoniker(A2OLE("SWDMaker"),
                                      &m_plMoniker);
  if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not create file moniker.", hResult);
    return hResult;
  }
 // Get an interface pointer for the ROT
  IRunningObjectTable* pIROT = NULL;
  hResult = GetRunningObjectTable(0, &pIROT);
  if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not obtain ROT interface pointer.", hResult);
    m_plMoniker->Release();
    m_plMoniker = NULL;
    return hResult;
  }
 // Now place the moniker into the ROT
  hResult = pIROT->Register(0, GetUnknown(), m_plMoniker, &m_dwROTCookie);
   if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not register Moniker in the ROT.", hResult);
    m_plMoniker->Release();
    m_plMoniker = NULL;
    pIROT->Release();
    return hResult;
```

```
  }
// Release our IROT pointer
  pIROT->Release();
return NOERROR;
}
void ComSWDMaker::FinalRelease()
{
  // Remove our moniker from the ROT
  if (m_dwROTCookie) {
// Get an interface pointer for the ROT
    IRunningObjectTable* pIROT = NULL;
    HRESULT hResult = GetRunningObjectTable(0, &pIROT);
    // Now remove the moniker
    if (SUCCEEDED(hResult)) {
      hResult = pIROT->Revoke(m_dwROTCookie);
      pIROT->Release();
    }
    m_dwROTCookie = 0;
  }
 // And get rid of our Moniker
  if (m_pIMoniker) {
    m_pIMoniker->Release();
    m_pIMoniker = NULL;
  }
}
//
// ComSWDMaker interface members
//
STDMETHODIMP
ComSWDMaker::MakePackage(ULONG      ulAdvertiseID,
                         BSTR       bstrAdvertiseName,
                         ULONG      ulPkgID,
                         BSTR       bstrPkgName,
                         BSTR       bstrComputerName,
                         BSTR       bstrHostName,
                         BSTR       bstrIPAddress,
                         BSTR       bstrSourcePackage,
                         BSTR       bstrCommandExec,
                         BSTR       bstrWorkDirectory,
                         ULONG      ulInstallMode)
{
  VerboseMsg("In MakePackage.\n");
  LPSTR pszAdvertiseName,pszPkgName,pszComputerName,pszHostName;
  LPSTR pszIPAddress,pszSourcePackage,pszCommandExec,pszWorkDirectory;
  char  str[500];
  char msg[500];
```

```
char buf[50];;
ISWDLocator*  piSWDLocator = NULL;
LPOLESTR     pszHostname  = NULL;
ISWDInven*   piSWDInven = NULL;
HRESULT hResult;
char        achHostname[20];
// Save Package receive in Drive:\windows\cuswd\requests
char dir[50];
int  i=50,size,j=0;
size = GetWindowsDirectory(dir,i);
if ( size == 0)
{ return ERROR; }
if ( ulInstallMode == 0 )          sprintf(buf,"%s\\cuswd\\request\\%ld.pkg",dir,ulPkgID);
else                               sprintf(buf,"%s\\cuswd\\request\\%ld.pka",dir,ulPkgID);
UnicodeToAnsi(bstrAdvertiseName,&pszAdvertiseName);
UnicodeToAnsi(bstrPkgName,&pszPkgName);
UnicodeToAnsi(bstrComputerName,&pszComputerName);
UnicodeToAnsi(bstrHostName,&pszHostName);
UnicodeToAnsi(bstrIPAddress,&pszIPAddress);
UnicodeToAnsi(bstrSourcePackage,&pszSourcePackage);
UnicodeToAnsi(bstrCommandExec,&pszCommandExec);
UnicodeToAnsi(bstrWorkDirectory,&pszWorkDirectory);
try
{
                    CStdioFile f1(buf,CFile::modeCreate | CFile::modeWrite);
                    sprintf(msg,"Advertise ID = %ld\n",ulAdvertiseID);          strcpy(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Advertise Name = %s\n",pszAdvertiseName);      strcpy(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Package ID = %ld\n",ulPkgID);                  strcpy(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Package Name = %s\n",pszPkgName);              strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Computer Name = %s\n",pszComputerName);        strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Host Name = %s\n",pszHostName);                strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"IP Address = %s\n",pszIPAddress);              strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Source Package = %s\n",pszSourcePackage);      strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Command Executive = %s\n",pszCommandExec);             strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Work Directory  = %s\n",pszWorkDirectory);     strcat(str,msg);
                    f1.WriteString(msg);
                    sprintf(msg,"Install Mode = %ld\n",ulInstallMode);          strcpy(str,msg);
```

```
                    f1.WriteString(msg);
                    f1.Close();
  }
  catch(CFileException *e)
  {
            if(e->m_cause == CFileException::accessDenied)
                    MessageBox(NULL,"Insufficial Privacy write File","SWD MAKER",MB_OK);
            else MessageBox(NULL,"Error Write File","SWD MAKER",MB_OK);
  }
USES_CONVERSION;
strcpy(achHostname,pszHostName);
  AnsiToUnicode((PCHAR) achHostname, &psolHostname);
 COSERVERINFO  ServerInfo = { 0, psolHostname, NULL, 0 };
  MULTI_QI     MultiQI   = { &IID_ISWDLocator, NULL, NOERROR };
  hResult = CoCreateInstanceEx(CLSID_SWDLocator1,
                    NULL, CLSCTX_SERVER,
                    &ServerInfo,
                    1, &MultiQI);
  if (FAILED(hResult)) {
    // We'll let the client report the error
    // ReportError("Could not create a new PizzaLocator object.", hResult);
            AfxMessageBox("Can not create SWDLocator object.");
            return hResult;
  }
  // Free up our hostname string
  CoTaskMemFree(psolHostname);
  // Just for convenience
  pISWDLocator = (ISWDLocator*) MultiQI.pItf;
 // Look for any PizzaMaker objects that might be currently running
  // on the same machine as the locator
  hResult = pISWDLocator->Locate(A2OLE("SWDInven"),
                    IID_ISWDInven,
                    (PPVOID) &pISWDInven);
 // Clean up some interface pointers...
  pISWDLocator->Release();
  pISWDLocator = NULL;
  if (FAILED(hResult)) {
    // We'll let the client report the error
            AfxMessageBox("Can not connect SWDInven");
            return hResult;
  }
hResult=pISWDInven->UpdateClient(ulAdvertiseID,ulPkgID,bstrComputerName);
  if (FAILED(hResult))
  { AfxMessageBox("Error Update Client Package");}
//  AfxMessageBox("Update ClientComplete..");
pISWDInven->Release();
```

```
  sprintf(buf,"%s\\cuswd\\bin\\PCM.EXE",dir);
  j = WinExec(buf,SW_SHOWNORMAL);
  if (j < 31)                AfxMessageBox("Error: File PCM.EXE not Found");
  return NOERROR;
}
```

3.2 โปรแกรมติดตั้งโปรแกรมสำเร็จรูป (SWDPCM.exe)

3.2.2 NewPackage.h กำหนดค่าเริ่มต้นของการทำการติดตั้งโปรแกรมสำเร็จรูป

```
#if !defined(AFX_NEWPACKAGE_H__D0301053_95A8_11D3_9BD7_00AA00B85006__INCLUDED_)
#define AFX_NEWPACKAGE_H__D0301053_95A8_11D3_9BD7_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// NewPackage.h : header file
//
/////////////////////////////////////////////////////////////////////////////
// CNewPackage dialog
class CNewPackage : public CDialog
{
// Construction
public:
        CNewPackage(CWnd* pParent = NULL);  // standard constructor
// Dialog Data
        //{{AFX_DATA(CNewPackage)
        enum { IDD = IDD_NEWPACKAGE };
                // NOTE: the ClassWizard will add data members here
        //}}AFX_DATA
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CNewPackage)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);  // DDX/DDV support
        //}}AFX_VIRTUAL
// Implementation
protected:
        // Generated message map functions
        //{{AFX_MSG(CNewPackage)
        virtual void OnOK();
        virtual void OnCancel();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_NEWPACKAGE_H__D0301053_95A8_11D3_9BD7_00AA00B85006__INCLUDED_)
```

3.2.5 Testcomm.h กำหนดค่าเริ่มต้นของการตรวจสอบการติดตั้งโปรแกรมสำเร็จรูป

```
// testcomm.h : main header file for the TESTCOMM application
//
#if !defined(AFX_TESTCOMM_H__A703BB15_8979_11D3_9E78_00AA00B85006__INCLUDED_)
#define AFX_TESTCOMM_H__A703BB15_8979_11D3_9E78_00AA00B85006__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#ifndef __AFXWIN_H__
	#error include 'stdafx.h' before including this file for PCH
#endif
#include "resource.h"		// main symbols
/////////////////////////////////////////////////////////////////////////////
// CTestcommApp:
// See testcomm.cpp for the implementation of this class
class CTestcommApp : public CWinApp
{
public:
	CTestcommApp();
	char	AdvertList[50][20];
// Overrides
	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CTestcommApp)
	public:
	virtual BOOL InitInstance();
	virtual void Install_Admin(int MaxA);
	//}}AFX_VIRTUAL
// Implementation
	//{{AFX_MSG(CTestcommApp)
		// NOTE - the ClassWizard will add and remove member functions here.
		//    DO NOT EDIT what you see in these blocks of generated code !
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_TESTCOMM_H__A703BB15_8979_11D3_9E78_00AA00B85006__INCLUDED_)
```

3.2.10 NewPackage.cpp ฟังก์ชั่นการตรวจสอบการติดตั้งโปรแกรมสำเร็จรูปเมื่อได้รับคำสั่งให้ติดตั้งโปรแกรม

```
// NewPackage.cpp : implementation file
//
```

```
#include "stdafx.h"
#include "testcomm.h"
#include "NewPackage.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CNewPackage dialog
CNewPackage::CNewPackage(CWnd* pParent /*=NULL*/)
	: CDialog(CNewPackage::IDD, pParent)
{
	//{{AFX_DATA_INIT(CNewPackage)
		// NOTE: the ClassWizard will add member initialization here
	//}}AFX_DATA_INIT
}
void CNewPackage::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CNewPackage)
		// NOTE: the ClassWizard will add DDX and DDV calls here
	//}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CNewPackage, CDialog)
	//{{AFX_MSG_MAP(CNewPackage)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CNewPackage message handlers
void CNewPackage::OnOK()
{
	CDialog::OnOK();
}
void CNewPackage::OnCancel()
{
	CDialog::OnCancel();
}
```

3.2.12 Testcomm.cpp ฟังก์ชั่นในการติดตั้งโปรแกรมสำเร็จรูป

```
// testcomm.cpp : Defines the class behaviors for the application.
#include <stdlib.h>
#include "stdafx.h"
#include "testcomm.h"
#include "testcommDlg.h"
```

```
#include "NewPackage.h"
#include "utility.h"
#include "GUIDS.H"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
//////////////////////////////////////////////////////////////
// CTestcommApp
BEGIN_MESSAGE_MAP(CTestcommApp, CWinApp)
        //{{AFX_MSG_MAP(CTestcommApp)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG
        ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()
//////////////////////////////////////////////////////////////
// CTestcommApp construction
CTestcommApp::CTestcommApp()
{
}
//////////////////////////////////////////////////////////////
// The one and only CTestcommApp object
CTestcommApp theApp;
//////////////////////////////////////////////////////////////
// CTestcommApp initialization
BOOL CTestcommApp::InitInstance()
{
#ifdef _AFXDLL
        Enable3dControls();                     // Call this when using MFC in a shared DLL
#else
        Enable3dControlsStatic();       // Call this when linking to MFC statically
#endif
        CStdioFile f1;
        CString   PathName,buf1,buf2,buf3;
        int     i=0,j=0,k=0,l=0,Max,MaxA=0;
        char    dir[50];
        i = GetWindowsDirectory(dir,50);
        PathName = "dir ";                                      PathName +=  dir;
        PathName += "\\CUSWD\\Request\\*.pka /w> ";      PathName += dir;
        PathName += "\\CUSWD\\tmp\\pcm.log";
        system(PathName);                                       PathName = dir;
        PathName += "\\CUSWD\\tmp\\pcm.log";
        f1.Open(PathName,CFile::modeRead);
```

```
while (!feof(f1.m_pStream))
{
   f1.ReadString(buf1);
            if ((k=buf1.Find("pka")) >0)  // Install Admin mode
            {
                        l = buf1.GetLength();
                        while ((k=buf1.Find("pka")) > 0)
                        {
                                    buf3 = buf1.Left(k+3);
                                    buf3.TrimRight();
                                    strcpy(AdverList[MaxA],buf3);
                                    MaxA++;
                                    buf2 = buf1.Right(l-k-3);
                                    l = buf2.GetLength();
                                    buf1 = buf2;
                        }
            }
            if ((k=buf1.Find("PKA")) >0)   // install Admin mode
            {
                        l = buf1.GetLength();
                        while ((k=buf1.Find("PKA")) > 0)
                        {
                                    buf3 = buf1.Left(k+3);
                                    buf3.TrimRight();
                                    strcpy(AdverList[MaxA],buf3);
                                    MaxA++;
                                    buf2 = buf1.Right(l-k-3);
                                    l = buf2.GetLength();
                                    buf1 = buf2;
                        }
            }
}
f1.Close();
i = GetWindowsDirectory(dir,50);
PathName = "dir ";                                  PathName += dir;
PathName += "\\CUSWD\\Request\\*.pkg /w> ";          PathName += dir;
PathName += "\\CUSWD\\tmp\\pcm.log";
system(PathName);
PathName = dir;
PathName += "\\CUSWD\\tmp\\pcm.log";
f1.Open(PathName,CFile::modeRead);
i=0;
while (!feof(f1.m_pStream))
{
   f1.ReadString(buf1);
            if ((k=buf1.Find("pkg")) >0)
```

```
                {
                        l = buf1.GetLength();
                        while ((k=buf1.Find("pkg")) > 0)
                        {
                                buf3 = buf1.Left(k+3);
                                buf3.TrimRight();
                                i++;
                                buf2 = buf1.Right(l-k-3);
                                l = buf2.GetLength();
                                buf1 = buf2;
                        }
                }
                if ((k=buf1.Find("PKG")) >0)
                {
                        l = buf1.GetLength();
                        while ((k=buf1.Find("PKG")) > 0)
                        {
                                buf3 = buf1.Left(k+3);
                                buf3.TrimRight();
                                i++;
                                buf2 = buf1.Right(l-k-3);
                                l = buf2.GetLength();
                                buf1 = buf2;
                        }
                }

        }
        f1.Close();
        Max=i;
        if ( MaxA > 0 )         Install_Admin(MaxA);
        if ( Max > 0)
        {
                CNewPackage dlg;
                if (dlg.DoModal() != IDCANCEL)
                {
                  CTestcommDlg pcm;
                  pcm.DoModal();
                  // dismissed with Cancel
                }
        }
        // Since the dialog has been closed, return FALSE so that we exit the
        //  application, rather than start the application's message pump.
        return FALSE;
}
void CTestcommApp::Install_Admin(int Maxx)
{
```

```
CStdioFile f1,f2;
CString   PathName,buf1,buf2,buf3;
CString   sAdvertiseName,sPkgName,sComputerName,sHostName,sIPAddress;
CString   sSourcePackage,sCommandExec,sWorkDirectory;
CString   sAdvertiseID,sPkgID;
char      dir[50],cPkgID[10],tmp[100];;
int                    i=0,j=0,k=0,l=0;
CTime     TheTime = CTime::GetCurrentTime();
i = GetWindowsDirectory(dir,50);
for (i=0;i<Maxx;i++)
{
   PathName =  dir;
   PathName += "\\CUSWD\\Request\\";
   PathName += AdvertList[i];
   f1.Open(PathName,CFile::modeRead);
   while (!feof(f1.m_pStream))
            {
               f1.ReadString(buf1);
               if ((k=buf1.Find("tise ID")) >0)
                        {  l = buf1.GetLength();
                           j = buf1.Find("=");
                                    buf2 = buf1.Right(l-j-1);
                                    buf2.TrimLeft();
                                    buf2.TrimRight();
                                    sAdvertiseID=buf2;
                        }
                        if ((k=buf1.Find("tise Name")) >0)
                        {  l = buf1.GetLength();
                           j = buf1.Find("=");
                                    buf2 = buf1.Right(l-j-1);
                                    buf2.TrimLeft();
                                    buf2.TrimRight();
                                    sAdvertiseName=buf2;
                        }
               if ((k=buf1.Find("kage ID")) >0)
                        {  l = buf1.GetLength();
                           j = buf1.Find("=");
                                    buf2 = buf1.Right(l-j-1);
                                    buf2.TrimLeft();
                                    buf2.TrimRight();
                                    sPkgID=buf2;
                        }
               if ((k=buf1.Find("kage Name")) >0)
                        {  l = buf1.GetLength();
                           j = buf1.Find("=");
                                    buf2 = buf1.Right(l-j-1);
```

```
                buf2.TrimLeft();
                buf2.TrimRight();
                sPkgName=buf2;
        }
if ((k=buf1.Find("puter Name")) >0)
        {  l = buf1.GetLength();
           j = buf1.Find("=");
                buf2 = buf1.Right(l-j-1);
                buf2.TrimLeft();
                buf2.TrimRight();
                sComputerName=buf2;
        }
if ((k=buf1.Find("ost Name")) >0)
        {  l = buf1.GetLength();
           j = buf1.Find("=");
                buf2 = buf1.Right(l-j-1);
                buf2.TrimLeft();
                buf2.TrimRight();
                sHostName=buf2;
        }
if ((k=buf1.Find("P Address")) >0)
        {  l = buf1.GetLength();
           j = buf1.Find("=");
                buf2 = buf1.Right(l-j-1);
                buf2.TrimLeft();
                buf2.TrimRight();
                sIPAddress=buf2;
        }
if ((k=buf1.Find("ce Package")) >0)
        {  l = buf1.GetLength();
           j = buf1.Find("=");
                buf2 = buf1.Right(l-j-1);
                buf2.TrimLeft();
                buf2.TrimRight();
                sSourcePackage=buf2;
        }
if ((k=buf1.Find("Execu")) >0)
        {  l = buf1.GetLength();
           j = buf1.Find("=");
                buf2 = buf1.Right(l-j-1);
                buf2.TrimLeft();
                buf2.TrimRight();
                sCommandExec=buf2;
        }
if ((k=buf1.Find("Directory")) >0)
        {  l = buf1.GetLength();
```

```
                j = buf1.Find("=");
                        buf2 = buf1.Right(l-j-1);
                        buf2.TrimLeft();
                        buf2.TrimRight();
                        sWorkDirectory=buf2;
                }
        }
        f1.Close();
        int pid,svid;
        pid = atoi(sAdvertiseID);
        svid = atoi(sPkgID);
        GetWindowsDirectory(dir,50);
        PathName = dir;
        PathName += "\\CUSWD\\Log\\PCM.log";
        if(!f2.Open(PathName,CFile::modeReadWrite))
        {
                // File not Create
                f2.Open(PathName,CFile::modeReadWrite|CFile::modeCreate);
        }
        f2.SeekToEnd();
        CString STime = TheTime.Format("%B %d, %Y  %H:%M:%S");
        buf1 = "PCM Program Start : ";          buf1 += STime;
        buf1 += "\n";
        f2.WriteString(buf1);
        f2.Close();
        if (!AfxOleInit()) {
                AfxMessageBox("Could not initialize OLE subsystem.");
        }
        USES_CONVERSION;
        ISWDLocator*  pISWDLocator = NULL;
        LPOLESTR     pszHostname  = NULL;
        ISWDInven*   pISWDInven = NULL;
        HRESULT hResult;
        char        achHostname[20];
        BSTR        bstrComputerName;
        bstrComputerName = sComputerName.AllocSysString();
        // define Server to connect SWD Maker COM
        strcpy(achHostname,sHostName);
        AnsiToUnicode((PCHAR) achHostname, &pszHostname);
        COSERVERINFO  ServerInfo = { 0, pszHostname, NULL, 0 };
        MULTI_QI     MultiQI    = { &IID_ISWDLocator, NULL, NOERROR };
        hResult = CoCreateInstanceEx(CLSID_SWDLocator1,
                                        NULL, CLSCTX_SERVER,
                                        &ServerInfo,
                                        1, &MultiQI);
        if (FAILED(hResult)) {
```

```
            // We'll let the client report the error
            AfxMessageBox("Can not create SWDLocator object.");
            return;
}
// Free up our hostname string
CoTaskMemFree(pszHostname);
// Just for convenience
pISWDLocator = (ISWDLocator*) MultiQI.pItf;
// Look for any SWDInven objects that might be currently running
// on the same machine as the locator
hResult = pISWDLocator->Locate(A2OLE("SWDInven"),
                        IID_ISWDInven,
                              (PPVOID) &pISWDInven);
// Clean up some interface pointers...
pISWDLocator->Release();
pISWDLocator = NULL;
if (FAILED(hResult)) {
            // We'll let the client report the error
            AfxMessageBox("Can not connect SWDInven");
            return;
}
//  AfxMessageBox("PCM Start Step 4.");
hResult=pISWDInven->UpdatePcmRetry(avid,pid,bstrComputerName);
if (FAILED(hResult))
{
            AfxMessageBox("Error PCM Package");
            //              return hResult;
            return;
}
PathName = sSourcePackage;                  PathName += "\\";
PathName += sCommandExec;
j = WinExec(PathName,SW_SHOWNORMAL);
if (j > 31 )
{
            hResult=pISWDInven->UpdatePcmComplete(avid,pid,bstrComputerName);
            if (FAILED(hResult))
            { AfxMessageBox("Error PCM Package");
              return;
            }
            PathName = dir;
            PathName += "\\CUSWD\\Log\\Pcm.log";
            f2.Open(PathName,CFile::modeReadWrite);
            f2.SeekToEnd();
            sprintf(tmp,"PCM Install Package %ld : %s \n",pid,
                        sPkgName);
            buf1 = tmp;
```

```
                    f2.WriteString(buf1);
                    TheTime = CTime::GetCurrentTime();
                    STime = TheTime.Format("%B %d, %Y  %H:%M:%S");
                    buf1 = "Complete at " + STime;
                    buf1 += "\n";
                    f2.WriteString(buf1);
                    f2.Close();
                    PathName = "move ";                    PathName += dir;
                    PathName += "\\CUSWD\\Request\\";
                    sprintf(cPkgID,"%ld",avid);
                    PathName += cPkgID;                    PathName += ".pka ";
                    PathName += dir;
                    PathName += "\\CUSWD\\Complete";
                    system(PathName);
                }
                else {
                    PathName = dir;
                    PathName += "\\CUSWD\\Log\\Pcm.log";
                    f2.Open(PathName,CFile::modeReadWrite);
                    f2.SeekToEnd();
                    sprintf(tmp,"Can not PCM Install Package %ld : %s \n",pid,sPkgName);
                    buf1 = tmp;
                    f2.WriteString(buf1);
                    TheTime = CTime::GetCurrentTime();
                    STime = TheTime.Format("%B %d, %Y  %H:%M:%S");
                    buf1 = "Not complete at " + STime;                    buf1 += "\n";
                    f2.WriteString(buf1);
                    f2.Close();
                    AfxMessageBox("Error can not executive PCM ..");
                }
                pISWDInven->Release();
            } // END loop for
}
```

4. โปรแกรมติดตั้งและติดตามผล การติดตั้งโปรแกรมสำเร็จรูป (SWDMain.exe)

4.1 Main.h กำหนดค่าเริ่มต้นของโปรแกรมติดตามการกระจายการติดตั้งโปรแกรมสำเร็จรูป

```
// Main.h : main header file for the Main application
#if !defined(AFX_BOY_H__A99701C5_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
#define AFX_BOY_H__A99701C5_9C7C_11D3_A69C_008048DE5B94__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#ifndef __AFXWIN_H__
        #error include 'stdafx.h' before including this file for PCH
#endif
```

```
#include "resource.h"       // main symbols
////////////////////////////////////////////////////////////////
// CBoyApp:
// See boy.cpp for the implementation of this class
class CBoyApp : public CWinApp
{
public:
        CBoyApp();
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CBoyApp)
        public:
        virtual BOOL InitInstance();
        //}}AFX_VIRTUAL
// Implementation
        //{{AFX_MSG(CBoyApp)
        afx_msg void OnAppAbout();
                // NOTE - the ClassWizard will add and remove member functions here.
                //    DO NOT EDIT what you see in these blocks of generated code !
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_BOY_H__A99701C5_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
```

4.2 MainDoc.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล MainDoc.cpp

```
// MainDoc.h : interface of the CBoyDoc class
//
////////////////////////////////////////////////////////////////
#if !defined(AFX_BOYDOC_H__A99701CB_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
#define AFX_BOYDOC_H__A99701CB_9C7C_11D3_A69C_008048DE5B94__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "boySet.h"
/*
class CPackage
{
public:
        long        PkgID;
        char        PkgName[60];
        char        PkgPDFTypeName[50];
        char        PkgVender[50];
```

```
        char    PkgCreatedate[30];
        char    PkgComment[100];
        void    init(long id,LPSTR Name,LPSTR Type,LPSTR Ven,LPSTR date,LPSTR remark);
};
*/
class CBoyDoc : public CDocument
{
protected: // create from serialization only
        CBoyDoc();
        DECLARE_DYNCREATE(CBoyDoc)
// Attributes
public:
        CBoySet m_boySet;
        CListCtrl *m_list;
  BOOL m_IsConnectionOpen;
  _ConnectionPtr m_pConnection;
// Operations
public:
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CBoyDoc)
        public:
        virtual BOOL OnNewDocument();
        virtual void OnCloseDocument();
        //}}AFX_VIRTUAL
// Implementation
public:
        virtual ~CBoyDoc();
#ifdef _DEBUG
        virtual void AssertValid() const;
        virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Generated message map functions
protected:
        //{{AFX_MSG(CBoyDoc)
                // NOTE - the ClassWizard will add and remove member functions here.
                //    DO NOT EDIT what you see in these blocks of generated code !
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_BOYDOC_H__A99701CB_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
```

4.3 MainView.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล MainView.cpp

```
// MainView.h : interface of the CBoyView class
//
/////////////////////////////////////////////////////////////////////////////
#if !defined(AFX_BOYVIEW_H__A99701CD_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
#define AFX_BOYVIEW_H__A99701CD_9C7C_11D3_A69C_008048DE5B94__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CBoySet;
class CBoyView : public CRecordView
{
protected: // create from serialization only
        CBoyView();
        DECLARE_DYNCREATE(CBoyView)
public:
        //{{AFX_DATA(CBoyView)
        enum { IDD = IDD_BOY_FORM };
        CListCtrl    m_list;
        CBoySet* m_pSet;
        //}}AFX_DATA
// Attributes
public:
        CBoyDoc* GetDocument();
// Operations
public:
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CBoyView)
        public:
        virtual CRecordset* OnGetRecordset();
        virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV support
        virtual void OnInitialUpdate(); // called first time after construct
        virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
        virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
        virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
        //}}AFX_VIRTUAL
// Implementation
public:
        virtual ~CBoyView();
#ifdef _DEBUG
        virtual void AssertValid() const;
        virtual void Dump(CDumpContext& dc) const;
#endif
protected:
```

```
// Generated message map functions
protected:
        //{{AFX_MSG(CBoyView)
        afx_msg void OnPaint();
        afx_msg void OnClickList1(NMHDR* pNMHDR, LRESULT* pResult);
        afx_msg void OnReportComplete();
        afx_msg void OnReportFail();
        afx_msg void OnReportCompLowspace();
        afx_msg void OnReportCompNotcomplete();
        afx_msg void OnReportCompComplete();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in boyView.cpp
inline CBoyDoc* CBoyView::GetDocument()
   { return (CBoyDoc*)m_pDocument; }
#endif
/////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_BOYVIEW_H__A99701CD_9C7C_11D3_A69C_008048DE5894__INCLUDED_)
```

4.4 MainSet.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล MainSet.cpp

```
// boySet.h : interface of the CBoySet class
/////////////////////////////////////////////////////////////////////////
#if !defined(AFX_BOYSET_H__A99701D1_9C7C_11D3_A69C_008048DE5894__INCLUDED_)
#define AFX_BOYSET_H__A99701D1_9C7C_11D3_A69C_008048DE5894__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
class CBoySet : public CRecordset
{
public:
        CBoySet(CDatabase* pDatabase = NULL);
        DECLARE_DYNAMIC(CBoySet)
// Field/Param Data
        //{{AFX_FIELD(CBoySet, CRecordset)
        CString     m_TEST1;
        CString     m_TEST2;
        CString     m_TEST3;
        //}}AFX_FIELD
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CBoySet)
        public:
        virtual CString GetDefaultConnect();  // Default connection string
```

```
        virtual CString GetDefaultSQL();    // default SQL for Recordset
        virtual void DoFieldExchange(CFieldExchange* pFX);    // RFX support
        //}}AFX_VIRTUAL
// Implementation
#ifdef _DEBUG
        virtual void AssertValid() const;
        virtual void Dump(CDumpContext& dc) const;
#endif
};
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_BOYSET_H__A99701D1_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
```

4.5 LeftView.h กำหนดค่าเริ่มต้นของแฟ้มข้อมูล LeftView.cpp

```
// LeftView.h : interface of the CLeftView class
/////////////////////////////////////////////////////////////////////////////
#if !defined(AFX_LEFTVIEW_H__A99701CF_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
#define AFX_LEFTVIEW_H__A99701CF_9C7C_11D3_A69C_008048DE5B94__INCLUDED_
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "boySet.h"
class CBoyDoc;
class CPdfHeader
{
public:
        long    cPdfID;
        char    sPkgName[100];
        int     cFlag;  // 0 = PDF , 1 = = Non PDF
        void    init(long id,LPSTR Name,int f);
};
class CPackage
{
public:
        long    cPkgID;
        char    sPkgName[100];
        char    sPkgPDFTypeName[50];
        char    sPkgVender[50];
        char    sPkgCommandExec[60];
        char    sPkgCreateDate[30];
        char    sPkgComment[100];
        void    init(long id,LPSTR Name,LPSTR Type,LPSTR Ven,LPSTR ComExe,LPSTR date,LPSTR remark);
};
```

```
class CAdvertise
{
public:
        long    cAdverID;
        char    sAdverName[100];
        char    sPkgName[100];
        char    sPkgPDFTypeName[50];
        char    sStartDate[30];
        char    sExpireDate[30];
        void    init(long id,LPSTR AName,LPSTR PName,LPSTR Type,LPSTR StrDate,LPSTR ExpDate);
};
class CLeftView : public CTreeView
{
protected: // create from serialization only
        CLeftView();
        DECLARE_DYNCREATE(CLeftView)

// Attributes
public:
        CBoyDoc*  GetDocument();
        CTreeCtrl  *m_tree;
        CListCtrl  *m_list;
        CBoySet                 *m_pSet;
        CPdfHeader  APdfH[200];    // Max Array = 200
        CPackage   APkg[20];      // Max Array = 50
        CAdvertise   AAdver[200];   // Max Array = 200
        int MaxPkg;
        int MaxAdver;
        int MaxSubPkg;
  HTREEITEM  phtreeitem,adhtreeitem,sphtreeitem,sadhtreeitem;
// Operations
public:
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CLeftView)
        public:
        virtual void OnDraw(CDC* pDC);  // overridden to draw this view
        virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
        protected:
        virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
        virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
        virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
        virtual void OnInitialUpdate(); // called first time after construct
        //}}AFX_VIRTUAL
private :
        BOOL Gen_Advertise(HTREEITEM selected);
```

```
        BOOL Show_Node_Advertise(HTREEITEM selected);
        BOOL Gen_Package(HTREEITEM selected);
        BOOL Show_Status_Package(HTREEITEM selected);
        BOOL Show_Status_Advertise(HTREEITEM selected);
        BOOL Show_Node_Computers(HTREEITEM selected);
        BOOL Show_Node_Package(HTREEITEM selected);
        BOOL Show_Node_All_Systems(int index);
        BOOL Show_Detail_Package(HTREEITEM selected);
        BOOL Show_Detail_Advertise(HTREEITEM selected);
        int  SearchPdfID(CString str);
        int  SearchAdverID(CString str);
// Implementation
public:
        virtual ~CLeftView();
#ifdef _DEBUG
        virtual void AssertValid() const;
        virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Generated message map functions
protected:
        //{{AFX_MSG(CLeftView)
        afx_msg void OnSelchanged(NMHDR* pNMHDR, LRESULT* pResult);
        afx_msg void OnRclick(NMHDR* pNMHDR, LRESULT* pResult);
        afx_msg void OnRButtonDown(UINT nFlags, CPoint point);
        afx_msg void OnDistributionNew();
        afx_msg void OnDistributionRefresh();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
#ifndef _DEBUG  // debug version in LeftView.cpp
inline CBoyDoc* CLeftView::GetDocument()
   { return (CBoyDoc*)m_pDocument; }
#endif
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.
#endif // !defined(AFX_LEFTVIEW_H__A99701CF_9C7C_11D3_A69C_008048DE5B94__INCLUDED_)
```

4.9 MainDoc.cpp ฟังก์ชั่นมาตรฐานในการจัดการของโปรแกรม

```
// MainDoc.cpp : implementation of the CBoyDoc class
//
#include "stdafx.h"
#include "boy.h"
#include "boySet.h"
#include "boyDoc.h"
```

```
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CBoyDoc
IMPLEMENT_DYNCREATE(CBoyDoc, CDocument)
BEGIN_MESSAGE_MAP(CBoyDoc, CDocument)
        //{{AFX_MSG_MAP(CBoyDoc)
                // NOTE - the ClassWizard will add and remove mapping macros here.
                //    DO NOT EDIT what you see in these blocks of generated code!
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CBoyDoc construction/destruction
CBoyDoc::CBoyDoc()
{
}
CBoyDoc::~CBoyDoc()
{
}
BOOL CBoyDoc::OnNewDocument()
{
  if (!CDocument::OnNewDocument())
        return FALSE;
  HRESULT hr;
  try
  {
        hr = m_pConnection.CreateInstance( __uuidof( Connection ) );
        if (SUCCEEDED(hr))
        {
                hr = m_pConnection->Open(
                _bstr_t(L"CUSWD"),    // System DSN of SQL Server
                _bstr_t(L"cusa"),     // User name
                _bstr_t(L"cusaa"),    // Password
                adModeUnknown);
                if (SUCCEEDED(hr))    m_IsConnectionOpen = TRUE;
        }
  }
  catch( _com_error &e )
  {
        // Get info from _com_error
        _bstr_t bstrSource(e.Source());
        _bstr_t bstrDescription(e.Description());
        TRACE( "Exception thrown for classes generated by #import" );
```

```
            TRACE( "\tCode = %081lx\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
            return hr;
  }
  catch(...)
  {
            TRACE( " *** Unhandled Exception *** ");
            return hr;
  }
          return TRUE;
}
////////////////////////////////////////////////////////////
// CBoyDoc diagnostics
#ifdef _DEBUG
void CBoyDoc::AssertValid() const
{
          CDocument::AssertValid();
}
void CBoyDoc::Dump(CDumpContext& dc) const
{
          CDocument::Dump(dc);
}
#endif //_DEBUG
////////////////////////////////////////////////////////////
// CBoyDoc commands
void CBoyDoc::OnCloseDocument()
{
  // Close Database Connection
  if (m_IsConnectionOpen)
  {
            m_IsConnectionOpen = FALSE;
            m_pConnection->Close();
  }
  CDocument::OnCloseDocument();
}
```

4.10 MainSet.cpp ฟังก์ชั่นมาตรฐานในการติดต่อระบบฐานข้อมูล

```
// MainSet.cpp : implementation of the CBoySet class
#include "stdafx.h"
#include "boy.h"
#include "boySet.h"
#ifdef _DEBUG
#define new DEBUG_NEW
```

```
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
////////////////////////////////////////////////////////////////////
// CBoySet implementation
IMPLEMENT_DYNAMIC(CBoySet, CRecordset)
CBoySet::CBoySet(CDatabase* pdb)
        : CRecordset(pdb)
{
        //{{AFX_FIELD_INIT(CBoySet)
        m_TEST1 = _T("");
        m_TEST2 = _T("");
        m_TEST3 = _T("");
        m_nFields = 2;
        //}}AFX_FIELD_INIT
        m_nDefaultType = snapshot;
}
CString CBoySet::GetDefaultConnect()
{
        return _T("ODBC;DSN=CUSWD;UID=cusa;PWD=cusaa");
}
CString CBoySet::GetDefaultSQL()
{
        return _T("[dbo].[Advertise]");
}
void CBoySet::DoFieldExchange(CFieldExchange* pFX)
{
        //{{AFX_FIELD_MAP(CBoySet)
        pFX->SetFieldType(CFieldExchange::outputColumn);
        RFX_Text(pFX, _T("[AdvertiseID]"), m_TEST1);
        RFX_Text(pFX, _T("[AdvertiseName]"), m_TEST2);
        // RFX_Text(pFX, _T("[TEST3]"), m_TEST3);
        //}}AFX_FIELD_MAP
}
////////////////////////////////////////////////////////////////////
// CBoySet diagnostics
#ifdef _DEBUG
void CBoySet::AssertValid() const
{
        CRecordset::AssertValid();
}
void CBoySet::Dump(CDumpContext& dc) const
{
        CRecordset::Dump(dc);
}
#endif //_DEBUG
```

4.11 MainView.cpp ฟังก์ชั่นมาตรฐานในการแสดงผลของโปรแกรม

```
// MainView.cpp : implementation of the CBoyView class
#include "stdafx.h"
#include "boy.h"
#include "boySet.h"
#include "boyDoc.h"
#include "boyView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CBoyView
IMPLEMENT_DYNCREATE(CBoyView, CRecordView)
BEGIN_MESSAGE_MAP(CBoyView, CRecordView)
        //{{AFX_MSG_MAP(CBoyView)
        ON_WM_PAINT()
        ON_NOTIFY(NM_CLICK, IDC_LIST1, OnClickList1)
        ON_COMMAND(ID_REPORT_COMPLETE, OnReportComplete)
        ON_COMMAND(ID_REPORT_FAIL, OnReportFail)
        ON_COMMAND(ID_REPORT_COMP_LOWSPACE, OnReportCompLowspace)
        ON_COMMAND(ID_REPORT_COMP_NOTCOMPLETE, OnReportCompNotcomplete)
        ON_COMMAND(ID_REPORT_COMP_COMPLETE, OnReportCompComplete)
        //}}AFX_MSG_MAP
        // Standard printing commands
        ON_COMMAND(ID_FILE_PRINT, CRecordView::OnFilePrint)
        ON_COMMAND(ID_FILE_PRINT_DIRECT, CRecordView::OnFilePrint)
        ON_COMMAND(ID_FILE_PRINT_PREVIEW, CRecordView::OnFilePrintPreview)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CBoyView construction/destruction
enum ICON_IMAGE
{
        MAIN = 0,
        ALL_SYSTEM = 1,
        ONE_SYSTEM = 2,
        PACKAGE = 3,
        ADVERTISE = 4,
        LOG = 5
};
CBoyView::CBoyView()
        : CRecordView(CBoyView::IDD)
{
```

```
        //{{AFX_DATA_INIT(CBoyView)
        // NOTE: the ClassWizard will add member initialization here
        m_pSet = NULL;
        //}}AFX_DATA_INIT
}
CBoyView::~CBoyView()
{
}
void CBoyView::DoDataExchange(CDataExchange* pDX)
{
        CRecordView::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CBoyView)
        DDX_Control(pDX, IDC_LIST1, m_list);
        //}}AFX_DATA_MAP
}
BOOL CBoyView::PreCreateWindow(CREATESTRUCT& cs)
{
        return CRecordView::PreCreateWindow(cs);
}
void CBoyView::OnInitialUpdate()
{
        m_pSet = &GetDocument()->m_boySet;
        GetDocument()->m_list = & m_list;
        CRecordView::OnInitialUpdate();
        GetParentFrame()->RecalcLayout();
        ResizeParentToFit();
        CImageList *img;
        CBitmap bmp;
        img = new CImageList();
        img->Create(16,16,ILC_COLORDDB,10,0);
        //              ALL_SYSTEM = 0
        bmp.LoadBitmap(IDB_MAIN);
        img->Add(&bmp,RGB(0,0,0));
        bmp.DeleteObject();
        //              S1_NO_RECORD = 1
        bmp.LoadBitmap(IDB_ALL_SYSTEM);
        img->Add(&bmp,RGB(0,0,0));
        bmp.DeleteObject();
        //              S1_HAVE_RECORD = 2
        bmp.LoadBitmap(IDB_COM);
        img->Add(&bmp,RGB(0,0,0));
        bmp.DeleteObject();
        //              S2_NO_RECORD = 3
        bmp.LoadBitmap(IDB_PACKAGE);
        img->Add(&bmp,RGB(0,0,0));
        bmp.DeleteObject();
```

```
        //              S2_HAVE_RECORD = 4
        bmp.LoadBitmap(IDB_DISK);
        img->Add(&bmp,RGB(0,0,0));
        bmp.DeleteObject();
        //              ONE_SYSTEM = 5
        bmp.LoadBitmap(IDB_LOG);
        img->Add(&bmp,RGB(0,0,0));
        bmp.DeleteObject();
        m_list.SetImageList(img,LVSIL_SMALL);
}
/////////////////////////////////////////////////////////////////////////////
// CBoyView printing
BOOL CBoyView::OnPreparePrinting(CPrintInfo* pInfo)
{
        // default preparation
        return DoPreparePrinting(pInfo);
}
void CBoyView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
}
void CBoyView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
}
/////////////////////////////////////////////////////////////////////////////
// CBoyView diagnostics
#ifdef _DEBUG
void CBoyView::AssertValid() const
{
        CRecordView::AssertValid();
}
void CBoyView::Dump(CDumpContext& dc) const
{
        CRecordView::Dump(dc);
}
CBoyDoc* CBoyView::GetDocument() // non-debug version is inline
{
        ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CBoyDoc)));
        return (CBoyDoc*)m_pDocument;
}
#endif //_DEBUG
/////////////////////////////////////////////////////////////////////////////
// CBoyView database support
CRecordset* CBoyView::OnGetRecordset()
{
        return m_pSet;
}
```

```
////////////////////////////////////////////////////////////
// CBoyView message handlers
void CBoyView::OnPaint()
{
	CPaintDC dc(this); // device context for painting
	CRect R;
	GetClientRect(&R);
	m_list.MoveWindow(&R);
}
void CBoyView::OnClickList1(NMHDR* pNMHDR, LRESULT* pResult)
{
	*pResult = 0;
}
void CBoyView::OnReportComplete()
{
	LV_COLUMN column;
	LV_ITEM insert;
	m_list.DeleteAllItems();
	while(m_list.DeleteColumn(0));
	// Insert Header
	column.mask = LVCF_FMT | LVCF_SUBITEM | LVCF_TEXT | LVCF_WIDTH;
	column.fmt = LVCFMT_LEFT;
	column.pszText = "Advertise Name ( Complete )";
	column.iSubItem = 0;
	column.cx = 400;
	m_list.InsertColumn(0,&column);
	column.pszText = "Advertise ID";
	column.iSubItem = 1;
	column.cx = 100;
	m_list.InsertColumn(1,&column);
	insert.mask = LVIF_TEXT|LVIF_IMAGE;
	insert.iItem = 0;
	insert.iImage = PACKAGE;
	insert.iSubItem = 0;
	CString ssqlsa = "select a.AdvertiseID,a.AdvertiseName ";
	ssqlsa += "FROM Advertise as a,AdvertisePackage as av ";
	ssqlsa += "WHERE a.AdvertiseID = av.AdvertiseID AND (ClientTranFlag = 1 ";
	ssqlsa += " AND ServerTranFlag = 1 AND PcmFlag = 1) ";
	_variant_t vRecsAffected(0L);
	LONG cAdverID=0;
	_RecordsetPtr pRecordSet;
	CBoyDoc * pDoc;
	pDoc =  GetDocument();
	CString str;
int i=0;
char s[300];
```

```
_bstr_t  bstrAdverName;
LONG                    cPcmRetry=0;
LONG    cClient=0;
LONG    cPcmComplete=0;
LONG    cPcmFailed=0;


try
{
        pRecordSet = pDoc->m_pConnection->Execute(_bstr_t(asqlss),&vRecsAffected,adOptionUnspecified);

        _variant_t vAdverID;
        _variant_t vAdverName;
        _variant_t vPcmRetry;
        _variant_t vClient;
        _variant_t vPcmComplete;
        char svar[50];
        int  j=0;

        while (!pRecordSet->GetadoEOF())
        {
                vAdverID = pRecordSet->GetCollect(L"AdvertiseID");
                cAdverID = vAdverID;

                vAdverName = pRecordSet->GetCollect(L"AdvertiseName");
                bstrAdverName = (_bstr_t) vAdverName;
                // Insert in List Control
                sprintf(svar,"      %d",cAdverID);
                 insert.mask = LVIF_TEXT|LVIF_IMAGE;
                 insert.pszText = (LPSTR) bstrAdverName;
                 insert.iImage = ADVERTISE;
                 insert.iItem = i;
                 insert.iSubItem = 0;
                 j= m_list.InsertItem(&insert);
                 insert.mask = LVIF_TEXT;
                 insert.pszText = svar ;
                 insert.iItem = j;
                 insert.iSubItem = 1;
                 m_list.SetItem(&insert);
                 pRecordSet->MoveNext();
        }
        pRecordSet->Close();
}
catch( _com_error &e )
{       // Get info from _com_error
        _bstr_t bstrSource(e.Source());
```

```
            _bstr_t bstrDescription(e.Description());
            TRACE( "Exception thrown for classes generated by #import" );
            TRACE( "\tCode = %08lx\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
    }
    catch(...)
    {
            TRACE( " *** Unhandled Exception *** ");
    }
}
void CBoyView::OnReportCompComplete()
{
        LV_COLUMN  column;
        LV_ITEM insert;
        m_list.DeleteAllItems();
        while(m_list.DeleteColumn(0));
        // Insert Header
        column.mask = LVCF_FMT | LVCF_SUBITEM | LVCF_TEXT | LVCF_WIDTH;
        column.fmt = LVCFMT_LEFT;
        column.pszText = "Computer Name ";
        column.iSubItem = 0;                    column.cx = 150;
        m_list.InsertColumn(0,&column);
        column.pszText = "Advertise Name";
        column.iSubItem = 1;                    column.cx = 400;
        m_list.InsertColumn(1,&column);
        column.pszText = "Start Schedual";
        column.iSubItem = 2;                    column.cx = 150;
        m_list.InsertColumn(2,&column);
        column.pszText = "End Schedual";
        column.iSubItem = 3;                    column.cx = 150;
        m_list.InsertColumn(3,&column);
        column.pszText = "Last Access";
        column.iSubItem = 4;                    column.cx = 150;
        m_list.InsertColumn(4,&column);
        column.pszText = "Package Retry";
        column.iSubItem = 5;                    column.cx = 100;
        m_list.InsertColumn(5,&column);
        column.pszText = "MachineID";
        column.iSubItem = 6;                    column.cx = 100;
        m_list.InsertColumn(6,&column);
        column.pszText = "AdvertiseID";
        column.iSubItem = 7;                    column.cx = 100;
        m_list.InsertColumn(7,&column);
        insert.mask = LVIF_TEXT|LVIF_IMAGE;
```

```
insert.iItem = 0;
insert.iImage = ONE_SYSTEM;
insert.iSubItem = 0;
CString ssqlsa = "select ComputerName,AdvertiseName,StartSchedual, \
                        EndSchedual,LastAccess,PcmRetry,inv.MachineID,av.AdvertiseID ";
ssqlsa += "FROM Inventory as inv,AdvertisePackage as av,Advertise as ad ";
ssqlsa += "WHERE PcmFlag = 1 AND inv.MachineID = av.MachineID AND av.AdvertiseID = ad.AdvertiseID ";
_variant_t vRecsAffected(0L);
LONG cMachineID=0;
LONG cAdvertiseID=0;
LONG cPcmRetry=0;
_RecordsetPtr pRecordSet;
CBoyDoc * pDoc;
pDoc =  GetDocument();
CString str;
 int  i=0;
char s[300];
_bstr_t  bstrComputerName;
_bstr_t  bstrAdvertiseName;
 _bstr_t  bstrStartSchedual;
_bstr_t  bstrEndSchedual;
_bstr_t  bstrLastAccess;
try
{
  pRecordSet = pDoc->m_pConnection->Execute(_bstr_t(ssqlsa),&vRecsAffected,adOptionUnspecified);
  _variant_t vComputerName;
  _variant_t vMachineID;
  _variant_t vAdvertiseName;
  _variant_t vAdvertiseID;
  _variant_t vStartSchedual;
  _variant_t vEndSchedual;
  _variant_t vLastAccess;
  _variant_t vPcmRetry;
  char svar[50];
  int j=0;
  LPSTR    strdate;
  while (!pRecordSet->GetadoEOF())
  {
            vComputerName = pRecordSet->GetCollect(L"ComputerName");
            bstrComputerName = (_bstr_t) vComputerName;
            vAdvertiseName   = pRecordSet->GetCollect(L"AdvertiseName");
            bstrAdvertiseName = (_bstr_t) vAdvertiseName;

            vStartSchedual   = pRecordSet->GetCollect(L"StartSchedual");
            bstrStartSchedual = (_bstr_t) vStartSchedual;
            vEndSchedual     = pRecordSet->GetCollect(L"EndSchedual");
```

```
bstrEndSchedual  = (_bstr_t) vEndSchedual;
vLastAccess      = pRecordSet->GetCollect(L"LastAccess");
bstrLastAccess   = (_bstr_t) vLastAccess;
vPcmRetry        = pRecordSet->GetCollect(L"PcmRetry");
cPcmRetry        = vPcmRetry;
vMachineID       = pRecordSet->GetCollect(L"MachineID");
cMachineID       = vMachineID;
vAdvertiseID     = pRecordSet->GetCollect(L"AdvertiseID");
cAdvertiseID     = vAdvertiseID;
// Insert in List Control
 insert.mask = LVIF_TEXT|LVIF_IMAGE;
 insert.pszText = (LPSTR) bstrComputerName;
 insert.iImage = ONE_SYSTEM;
 insert.iItem = i;          insert.iSubItem = 0;
 j= m_list.InsertItem(&insert);
 insert.mask = LVIF_TEXT;
 insert.pszText = (LPSTR) bstrAdvertiseName;
 insert.iItem = j;          insert.iSubItem = 1;
 m_list.SetItem(&insert);
 insert.mask = LVIF_TEXT;
 insert.pszText = (LPSTR) bstrStanSchedual;
 insert.iItem = j;          insert.iSubItem = 2;
 m_list.SetItem(&insert);
 strdate=(LPSTR) bstrEndSchedual;
 if ((strcmp(strdate,"1/1/99"))==0)
 {
          CString Msg="Not set Expire Date";
          sprintf(strdate,"        -");
 }
  insert.mask = LVIF_TEXT;
 insert.pszText = strdate;
 insert.iItem = j;          insert.iSubItem = 3;
 m_list.SetItem(&insert);
 strdate = (LPSTR) bstrEndSchedual;
 insert.mask = LVIF_TEXT;
 insert.pszText = (LPSTR) bstrLastAccess;
 insert.iItem = j;          insert.iSubItem = 4;
 m_list.SetItem(&insert);
 sprintf(svar,"      %d",cPcmRetry);
 insert.pszText = svar ;
 insert.iItem = j;          insert.iSubItem = 5;
m_list.SetItem(&insert);
 sprintf(svar,"      %d",cMachineID);
 insert.pszText = svar ;
 insert.iItem = j;          insert.iSubItem = 6;
 m_list.SetItem(&insert);
```

```
                    sprintf(svar,"      %d",cAdvertiseID);
                    insert.pszText = svar ;
                    insert.iItem = j;          insert.iSubItem = 7;
                    m_list.SetItem(&insert);
                    i++;
                    pRecordSet->MoveNext();
            }
            pRecordSet->Close();
    }
    catch( _com_error &e )
    {       // Get info from _com_error
            _bstr_t bstrSource(e.Source());
            _bstr_t bstrDescription(e.Description());
            TRACE( "Exception thrown for classes generated by #import" );
            TRACE( "\tCode = %08lx\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
    catch(...)
    {
            TRACE( " *** Unhandled Exception *** ");
    }
}
```

4.12 LeftView.cpp ฟังก์ชั่นมาตรฐานในการแสดงรายละเอียดของหน้าต่างด้านซ้ายของโปรแกรม

```
// LeftView.cpp : implementation of the CLeftView class
#include "stdafx.h"
#include "boy.h"
#include "boyDoc.h"
#include "boySet.h"
#include "LeftView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
////////////////////////////////////////////////////////////////////////
// CLeftView
IMPLEMENT_DYNCREATE(CLeftView, CTreeView)

BEGIN_MESSAGE_MAP(CLeftView, CTreeView)
        //{{AFX_MSG_MAP(CLeftView)
        ON_NOTIFY_REFLECT(TVN_SELCHANGED, OnSelchanged)
```

```
	ON_NOTIFY_REFLECT(NM_RCLICK, OnRclick)
	ON_WM_RBUTTONDOWN()
	ON_COMMAND(ID_DISTRIBUTION_NEW, OnDistributionNew)
	ON_COMMAND(ID_DISTRIBUTION_REFRESH, OnDistributionRefresh)
	//}AFX_MSG_MAP
	// Standard printing commands
	ON_COMMAND(ID_FILE_PRINT, CTreeView::OnFilePrint)
	ON_COMMAND(ID_FILE_PRINT_DIRECT, CTreeView::OnFilePrint)
	ON_COMMAND(ID_FILE_PRINT_PREVIEW, CTreeView::OnFilePrintPreview)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CLeftView construction/destruction
enum ICON_IMAGE
{
	MAIN = 0 ,
	ALL_SYSTEM = 1,
	ONE_SYSTEM = 2,
	PACKAGE = 3,
	ADVERTISE = 4,
	LOG = 5
};
int x,y;
CLeftView::CLeftView()
{
	m_tree = & GetTreeCtrl();
}
CLeftView::~CLeftView()
{
}
BOOL CLeftView::PreCreateWindow(CREATESTRUCT& cs)
{
	cs.style |= TVS_HASLINES | TVS_HASBUTTONS | TVS_SHOWSELALWAYS;
	return CTreeView::PreCreateWindow(cs);
}
/////////////////////////////////////////////////////////////////////////////
// CLeftView drawing
void CLeftView::OnDraw(CDC* pDC)
{
	CBoyDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
}
/////////////////////////////////////////////////////////////////////////////
// CLeftView printing
BOOL CLeftView::OnPreparePrinting(CPrintInfo* pInfo)
{
	// default preparation
```

```
	return DoPreparePrinting(pInfo);
}
void CLeftView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
}
void CLeftView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
}
void CPackage::init(long id,LPSTR Name,LPSTR Type,LPSTR Ven,LPSTR ComExe,LPSTR date,LPSTR remark)
{
	cPkgID= id;
	sprintf(sPkgName,"%s",Name);
	sprintf(sPkgPDFTypeName,"%s",Type);
	sprintf(sPkgVender,"%s",Ven);
	sprintf(sPkgCommandExec,"%s",ComExe);
	sprintf(sPkgCreateDate,"%s",date);
	sprintf(sPkgComment,"%s",remark);
}
void CPdfHeader::init(long id,LPSTR Name,int f)
{
	cPdfID = id;
	sprintf(sPkgName,"%s",Name);
	cFlag  = f;
}
void CAdvertise::init(long id,LPSTR AName,LPSTR PName,LPSTR Type,LPSTR StrDate,LPSTR ExpDate)
{
	cAdverID= id;
	sprintf(sAdverName,"%s",AName);
	sprintf(sPkgName,"%s",PName);
	sprintf(sPkgPDFTypeName,"%s",Type);
	sprintf(sStartDate,"%s",StrDate);
	if ((strcmp(ExpDate,"1/1/99"))==0)
	{
		CString Msg="Not set Expire Date";
		sprintf(sExpireDate,"  -  ");
	}
	else sprintf(sExpireDate,"%s",ExpDate);
}
////////////////////////////////////////////////////////////////////
// CLeftView diagnostics
#ifdef _DEBUG
void CLeftView::AssertValid() const
{
	CTreeView::AssertValid();
}
void CLeftView::Dump(CDumpContext& dc) const
```

```
{
        CTreeView::Dump(dc);
}
CBoyDoc* CLeftView::GetDocument() // non-debug version is inline
{
        ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CBoyDoc)));
        return (CBoyDoc*)m_pDocument;
}
#endif //_DEBUG
//////////////////////////////////////////////////////////////////
BOOL CLeftView::Show_Detail_Package(HTREEITEM selected)
{
        _RecordsetPtr pRecordSet;
        CBoyDoc * pDoc;
        pDoc =  GetDocument();
        CString  str;
        int     cPdfID;
        str =   m_tree->GetItemText(selected);
        cPdfID = SearchPdfID(str);
        if ( cPdfID < 0)
        {
                MessageBox("Can not Search PdfID in Array Header");
                return FALSE;
        }
        char sPdfID[5];
        sprintf(sPdfID,"%d",cPdfID);
        CString ssqlpdf = "Select PkgID,PkgName,InstallTypeName,p.Vender,CommandExec,p.CreateDate,p.Remark ";
        ssqlpdf += "From Package AS p,PackagePDFType AS t,PackagePDF as d ";
        ssqlpdf += "Where p.InstallType = t.InstallType AND p.InstallType = d.InstallType ";
        ssqlpdf += "AND p.PdfID = ";                    ssqlpdf += sPdfID;
        ssqlpdf += " AND d.PdfID = ";                   ssqlpdf += sPdfID;
        _variant_t vRecsAffected(0L);
        LONG cPkgID=0;
        int  i=0;
        char s[300];
        _bstr_t  bstrPkgName;
        _bstr_t  bstrPkgPDFTypeName;
        _bstr_t  bstrVender;
        _bstr_t  bstrCommendExec;
        _bstr_t  bstrCreateDate;
        _bstr_t  bstrRemark;
        try
        {
          pRecordSet = pDoc->m_pConnection->Execute(_bstr_t(ssqlpdf),&vRecsAffected,adOptionUnspecified);
          _variant_t vPkgID;
          _variant_t vPkgName;
```

```
            _variant_t vPkgPDFTypeName;
            _variant_t vVender;
            _variant_t vCommandExec;
            _variant_t vCreateDate;
            _variant_t vRemark;
            while (!pRecordSet->GetadoEOF())
            {
                        vPkgID = pRecordSet->GetCollect(L"PkgID");
                        cPkgID = vPkgID;
                        vPkgName = pRecordSet->GetCollect(L"PkgName");
                        bstrPkgName = (_bstr_t) vPkgName;
                        vPkgPDFTypeName = pRecordSet->GetCollect(L"InstallTypeName");
                        bstrPkgPDFTypeName = (_bstr_t) vPkgPDFTypeName;
                        vVender = pRecordSet->GetCollect(L"Vender");
                        bstrVender = (_bstr_t) vVender;
                        vCommandExec = pRecordSet->GetCollect(L"CommandExec");
                        bstrCommandExec = (_bstr_t) vCommandExec;
                        vCreateDate = pRecordSet->GetCollect(L"CreateDate");
                        bstrCreateDate = (_bstr_t) vCreateDate;
                        vRemark = pRecordSet->GetCollect(L"Remark");
                        bstrRemark = (_bstr_t) vRemark;
                        APkg[i].init(cPkgID,(LPSTR) bstrPkgName,(LPSTR) bstrPkgPDFTypeName,
                                    (LPSTR) bstrVender,(LPSTR) bstrCommandExec,(LPSTR) bstrCreateDate,(LPSTR) bstrRemark);
                        pRecordSet->MoveNext();
                        i++;
            }
            pRecordSet->Close();
  }
  catch( _com_error &e )
  {         // Get info from _com_error
            _bstr_t bstrSource(e.Source());
            _bstr_t bstrDescription(e.Description());
            TRACE( "Exception thrown for classes generated by #import" );
            TRACE( "\tCode = %08lx\n", e.Error());
            TRACE( "\tCode meaning = %s\n", e.ErrorMessage());
            TRACE( "\tSource = %s\n",  (LPCTSTR) bstrSource);
            TRACE( "\tDescription = %s\n", (LPCTSTR) bstrDescription);
}
  catch(...)
  {
            TRACE( "*** Unhandled Exception ***" );
  }
  MaxSubPkg = i;
  // Insert in to List Control
   LV_COLUMN  column;
   LV_ITEM insert;
```

```
m_list = GetDocument()->m_list;
m_list->DeleteAllItems();
while(m_list->DeleteColumn(0));
// Insert Header
column.mask = LVCF_FMT | LVCF_SUBITEM | LVCF_TEXT | LVCF_WIDTH;
column.fmt = LVCFMT_LEFT;
column.pszText = "Package Type Name";
column.iSubItem = 0;            column.cx = 200;
m_list->InsertColumn(0,&column);
column.pszText = "Vender";
column.iSubItem = 1;            column.cx = 100;
m_list->InsertColumn(1,&column);
column.pszText = "Command Execute";
column.iSubItem = 2;            column.cx = 150;
m_list->InsertColumn(2,&column);
column.pszText = "Available Date";
column.iSubItem = 3;            column.cx = 100;
m_list->InsertColumn(3,&column);
column.pszText = "Comment";
column.iSubItem = 4;            column.cx = 500;
m_list->InsertColumn(4,&column);
column.pszText = "Package ID";
column.iSubItem = 5;            column.cx = 100;
m_list->InsertColumn(5,&column);
insert.mask = LVIF_TEXT|LVIF_IMAGE;
insert.iItem = 0;
insert.iImage = PACKAGE;
insert.iSubItem = 0;
int j=0;
for (i=0;i<MaxSubPkg;i++)
{
        insert.mask = LVIF_TEXT|LVIF_IMAGE;
        insert.pszText = APkg[i].sPkgPDFTypeName;
        insert.iImage = PACKAGE;
        insert.iItem = i;           insert.iSubItem = 0;
        j = m_list->InsertItem(&insert);
        insert.mask = LVIF_TEXT;
        insert.pszText = APkg[i].sPkgVender;
        insert.iItem = j;           insert.iSubItem = 1;
        m_list->SetItem(&insert);
        insert.pszText = APkg[i].sPkgCommandExec;
        insert.iItem = j;           insert.iSubItem = 2;
        m_list->SetItem(&insert);
        insert.pszText = APkg[i].sPkgCreateDate;
        insert.iItem = j;           insert.iSubItem = 3;
        m_list->SetItem(&insert);
```

```
            insert.pszText = APkg[i].sPkgComment;
            insert.iItem = j;           insert.iSubItem = 4;
            m_list->SetItem(&insert);
            char pid[10];
            sprintf(pid,"%d",APkg[i].cPkgID);
            insert.pszText = pid;
            insert.iItem = j;           insert.iSubItem = 5;
            m_list->SetItem(&insert);
    }
return TRUE;
}
BOOL CLeftView::Show_Node_Package(HTREEITEM selected)
{
            LV_COLUMN  column;
            CString str;
            LV_ITEM insert;
            m_list = GetDocument()->m_list;
            m_list->DeleteAllItems();
            while(m_list->DeleteColumn(0));
            column.mask = LVCF_FMT | LVCF_SUBITEM | LVCF_TEXT | LVCF_WIDTH;
            column.fmt = LVCFMT_CENTER;
            column.pszText = "Package Description";
            column.iSubItem = 0;                column.cx = 500;
            m_list->InsertColumn(0,&column);
            selected = m_tree->GetNextItem( selected , TVGN_CHILD );
            insert.mask = LVIF_TEXT|LVIF_IMAGE;
            insert.iItem = 0;
            insert.iImage = PACKAGE;
            insert.iSubItem = 0;
            do
            {
                        str = m_tree->GetItemText(selected);
                        insert.pszText = str.GetBuffer(10);
                        insert.iItem = m_list->InsertItem(&insert);
            }while((selected = m_tree->GetNextItem(selected,TVGN_NEXT)) != NULL);
            return TRUE;
}
void CLeftView::OnRclick(NMHDR* pNMHDR, LRESULT* pResult)
{
            RECT boy;
            CMenu main , *popup;
            main.LoadMenu(IDR_Advertise);
            popup = main.GetSubMenu(0);
            this->GetWindowRect(&boy);
            popup->TrackPopupMenu(TPM_LEFTALIGN | TPM_RIGHTBUTTON,x+boy.left,y+boy.top,m_tree);
            *pResult = 0;
```

```
}
void CLeftView::OnRButtonDown(UINT nFlags, CPoint point)
{
	x = point.x;
	y = point.y;
	SendMessage(NM_RCLICK);
	CTreeView::OnRButtonDown(nFlags, point);
}
void CLeftView::OnDistributionNew()
{
	int i = WinExec("d:\\CU_SWD\\Distribute\\Debug\\Distribute.exe",SW_SHOWNORMAL);
}
```

4.1 MainFrm.cpp ฟังก์ชั่นมาตรฐานในการกำหนดหน้าต่างของโปรแกรม

```
// MainFrm.cpp : implementation of the CMainFrame class
#include "stdafx.h"
#include "boy.h"
#include "MainFrm.h"
#include "LeftView.h"
#include "boyView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CMainFrame
IMPLEMENT_DYNCREATE(CMainFrame, CFrameWnd)
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
	//{{AFX_MSG_MAP(CMainFrame)
	ON_WM_CREATE()
	ON_COMMAND(IDC_VIEW_TABLE, OnViewTable)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()
static UINT indicators[] =
{
	ID_SEPARATOR,	// status line indicator
	ID_INDICATOR_CAPS,
	ID_INDICATOR_NUM,
	ID_INDICATOR_SCRL,
};
/////////////////////////////////////////////////////////////////////////////
// CMainFrame construction/destruction
CMainFrame::CMainFrame()
{
}
```

```
CMainFrame::~CMainFrame()
{
}
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
	if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
		return -1;
	if (!m_wndToolBar.CreateEx(this, TBSTYLE_FLAT, WS_CHILD | WS_VISIBLE | CBRS_TOP
		| CBRS_GRIPPER | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC) ||
		!m_wndToolBar.LoadToolBar(IDR_MAINFRAME))
	{
		TRACE0("Failed to create toolbar\n");
		return -1;	// fail to create
	}
	if (!m_wndStatusBar.Create(this) ||
		!m_wndStatusBar.SetIndicators(indicators,
		  sizeof(indicators)/sizeof(UINT)))
	{
		TRACE0("Failed to create status bar\n");
		return -1;	// fail to create
	}
	// TODO: Delete these three lines if you don't want the toolbar to
	//  be dockable
	m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
	EnableDocking(CBRS_ALIGN_ANY);
	DockControlBar(&m_wndToolBar);
	return 0;
}

BOOL CMainFrame::OnCreateClient(/*LPCREATESTRUCT /*lpcs*/,
	CCreateContext* pContext)
{
	// create splitter window
	if (!m_wndSplitter.CreateStatic(this, 1, 2))
		return FALSE;
	if (!m_wndSplitter.CreateView(0, 0, RUNTIME_CLASS(CLeftView), CSize(180, 100), pContext) ||
		!m_wndSplitter.CreateView(0, 1, RUNTIME_CLASS(CBoyView), CSize(100, 100), pContext))
	{
		m_wndSplitter.DestroyWindow();
		return FALSE;
	}
	return TRUE;
}
BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
	if( !CFrameWnd::PreCreateWindow(cs) )
```

```
                return FALSE;
        return TRUE;
}
////////////////////////////////////////////////////////////
// CMainFrame diagnostics
#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
        CFrameWnd::AssertValid();
}
void CMainFrame::Dump(CDumpContext& dc) const
{
        CFrameWnd::Dump(dc);
}
#endif //_DEBUG
////////////////////////////////////////////////////////////
// CMainFrame message handlers
void CMainFrame::OnViewTable()
{
        int  xxx=0;
        CString   PathNamex;
        char      dirx[50];
        GetWindowsDirectory(dirx,50);
         PathNamex = dirx;
        PathNamex += "\\Cuswd\\bin\\SWD_Cal.exe";
        xxx =  WinExec(PathNamex,SW_SHOWNORMAL);
         if (xxx < 31)
        {
                MessageBox("Error: File SWD_Cal.exe not found..");
        }
}
```

4.15 CUSWDMain.cpp ฟังก์ชั่นหลักในการจัดการและแสดงผลการกระจายการติดตั้งโปรแกรมสำเร็จรูป

```
// CUSWDMain.cpp : Defines the class behaviors for the application.
#include "stdafx.h"
#include "boy.h"
#include "MainFrm.h"
#include "boySet.h"
#include "boyDoc.h"
#include "LeftView.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
```

```
static char THIS_FILE[] = __FILE__;
#endif
/////////////////////////////////////////////////////////////////////////////
// CBoyApp
BEGIN_MESSAGE_MAP(CBoyApp, CWinApp)
	//{{AFX_MSG_MAP(CBoyApp)
	ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
		// NOTE - the ClassWizard will add and remove mapping macros here.
		//    DO NOT EDIT what you see in these blocks of generated code!
	//}}AFX_MSG_MAP
	// Standard print setup command
	ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)
END_MESSAGE_MAP()
/////////////////////////////////////////////////////////////////////////////
// CBoyApp construction
CBoyApp::CBoyApp()
{
}
/////////////////////////////////////////////////////////////////////////////
// The one and only CBoyApp object
CBoyApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CBoyApp initialization
BOOL CBoyApp::InitInstance()
{
	AfxEnableControlContainer();
	AfxOleInit();
	// Standard initialization
	// If you are not using these features and wish to reduce the size
	//  of your final executable, you should remove from the following
	//  the specific initialization routines you do not need.

#ifdef _AFXDLL
	Enable3dControls();			// Call this when using MFC in a shared DLL
#else
	Enable3dControlsStatic();	// Call this when linking to MFC statically
#endif
	// Change the registry key under which our settings are stored.
	// TODO: You should modify this string to be something appropriate
	// such as the name of your company or organization.
	SetRegistryKey(_T("Local AppWizard-Generated Applications"));
	LoadStdProfileSettings();  // Load standard INI file options (including MRU)
	// Register the application's document templates.  Document templates
	//  serve as the connection between documents, frame windows and views.
	CSingleDocTemplate* pDocTemplate;
```

```
        pDocTemplate = new CSingleDocTemplate(
                IDR_MAINFRAME,
                RUNTIME_CLASS(CBoyDoc),
                RUNTIME_CLASS(CMainFrame),      // main SDI frame window
                RUNTIME_CLASS(CLeftView));
        AddDocTemplate(pDocTemplate);
        // Parse command line for standard shell commands, DDE, file open
        CCommandLineInfo cmdInfo;
        ParseCommandLine(cmdInfo);
        // Dispatch commands specified on the command line
        if (!ProcessShellCommand(cmdInfo))
                return FALSE;
        // The one and only window has been initialized, so show and update it.
        m_pMainWnd->ShowWindow(SW_SHOW);
        m_pMainWnd->UpdateWindow();
        return TRUE;
}
////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About
class CAboutDlg : public CDialog
{
public:
        CAboutDlg();
        // Dialog Data
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV support
        //}}AFX_VIRTUAL
// Implementation
protected:
        //{{AFX_MSG(CAboutDlg)
                // No message handlers
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
        //{{AFX_DATA_INIT(CAboutDlg)
        //}}AFX_DATA_INIT
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
```

```
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAboutDlg)
        //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
        //{{AFX_MSG_MAP(CAboutDlg)
                // No message handlers
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()
// App command to run the dialog
void CBoyApp::OnAppAbout()
{
        CAboutDlg aboutDlg;
        aboutDlg.DoModal();
}
/////////////////////////////////////////////////////////////////////////////
// CBoyApp message handlers
```

ประวัติผู้เขียน

นายอนิรุต เอี่ยมสกุล เกิดที่อำเภอเมือง จังหวัดจันทบุรี สำเร็จการศึกษาปริญญาตรี สาขาวิทยาการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้า วิทยาเขตพระนครเหนือ เข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต (นอกเวลาราชการ) สาขาวิทยาศาสตร์คอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยเมื่อ พ.ศ. 2539