# รายการอ้างอิง


ภาษาไทย

ถาวร ลิขนะไพบูลย์. การออกแบบและพัฒนาระบบจินตทัศน์อัลกอริทึมสำหรับปัญหาทางทฤษฎีกราฟ. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย, 2538.

ภาษาอังกฤษ

Brown, March H. Algorithm Animation. Cambridge, MA: MIT Press, 1988.

Graphics, Visualization and Usibility Center. XTANGO Algorithm Animation System. Atlanta, GA: Collefe Of Computing,Georgia Institute of Technology. Available from http://www.cc.gatech.edu/gvu/softviz/algoanim/xtango.html; INTERNET, 1992.

Graphics, Visualization and Usibility Center. SAMBA Animation Program. Atlanta, GA: College Of Computing,Georgia Institute of Technology. Available from http://www.cc.gatech.edu/gvu/softviz/algoanim/samba.html; INTERNET, 1992.

Graphics, Visualization and Usibility Center. Visualization for Program Understanding and Debugging. Atlanta, GA: College Of Computing,Georgia Institute of Technology, Available from http://www.cc.gatech.edu/gvu/softviz/algoanim/visdebug.html; INTERNET, 1992.

Laffra, Carla. Dijkstra's Shortest Path Algorithm Animation in Java. White Plains, NY: PACE University. Available from http://cs.pace.edu/~www/javademos/DijkstraText.html; INTERNET, 1996.

Microsoft. Microsoft Visual Basic Programming System for Windows Programmer's Guide version 3.0. Redmond,WA. : Microsoft Inc., 1993.

Microsoft. Microsoft Visual Basic Programming System for Windows Professional Feature Book One version 3.0. Redmond,WA : Microsoft Inc., 1993.

Microsoft, Causes of General Protection Faults, *Microsoft Developers' Network : Window 3.x Knowledge Base, version 3.0*, Microsoft Inc., April, 1995.

Petzold, Charles. Programming Windows 3.1. 3rd ed. Redmond,WA : Microsoft Press, 1992.

Prasitjutrakul, S. and Watcharawittayakul, W. Algorithm Visualization System : An Overview of Internal Structure. Proceedings of Third ASEAN Regional Seminar on Microelectronics and Information Technology. Bangkok, 1994.

ภาคผนวก

## ภาคผนวก ก.

## แฟ้มที่มีอยู่ในระบบ AVis

เมื่อทำการติดตั้งระบบ AVis ลงในระบบคอมพิวเตอร์เรียบร้อยแล้ว จะปรากฏแฟ้มและไดเร็คทอรี่ที่ปรากฏขึ้นมีดังนี้

### *1 ไดเร็คทอรี่ AVIS*

| ชื่อ | รายละเอียด |
|---|---|
| AVISDSGN.EXE | โปรแกรมช่วยออกแบบบทการจินตทัศน์ |
| AVISDSGN.MDB | แฟ้มข้อมูลของ AVISDSGN.EXE |
| AVISRUN.EXE | โปรแกรมควบคุมการจินตทัศน์ |
| AVISRUN.MDB | แฟ้มข้อมูลของ AVISDSGN.EXE |
| MAKEAVI.EXE | โปรแกรมช่วยสร้างแฟ้มรูปแบบ AVI |
| ACDK (directory) | ชุดพัฒนาองค์ปรแกรมการจินตทัศน์ |

### *2 ไดเร็คทอรี่ SYSTEM ของวินโดว์*

| ชื่อ | รายละเอียด |
|---|---|
| ARROW.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อสร้างรูปลูกศร |
| AVIREC.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อสร้างแฟ้มรูปแบบ AVI |
| AVISCOMP.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อช่วยพัฒนาองค์ประกอบ |
| AVISCTRL.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อช่วยพัฒนาโปรแกรมควบคุมการจินตทัศน์ |
| AVISDLL.DLL | คลังคำสั่งเชื่อมโยงแบบพลวัตของระบบ AVis |
| CMDIALOG.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิก |
| ICONBTN.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อแสดงปุ่มรูปภาพจาก icon |
| INIMGR.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่ออ่านและเขียนแฟ้มรูปแบบ INI |
| MCLIST.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อสร้าง list box แบบ หลาย column |
| MSGBLAST.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อรับข้อความคำสั่งของวินโดว์ |
| PICBTN.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อแสดงปุ่มรูปภาพ |
| PICCLK.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อแสดงกรอบรอบรูป |
| TAB.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิกเพื่อแสดง tab bar |
| THREED.VBX | ตัวควบคุมเฉพาะของวิชวลเบสิก |

### 3 ไดเร็คทอรี่ AVIS\ACDK

| ชื่อ | รายละเอียด |
|---|---|
| AVIS.BAS | แฟ้มรายละเอีย สำหรับการพัฒนาองค์ประกอบการจินตทัศน์ด้วยภาษาวิชวลเบสิก |
| AVIS.H | แฟ้มรายละเอีย สำหรับการพัฒนาองค์ประกอบการจินตทัศน์ด้วยภาษาซี |
| AVIS.HLP | แฟ้มรายละเอียดของระบบ AVis |
| AVISCOMP.HLP | แฟ้มรายละเอียดของ AVISCOMP.VBX |
| AVISDLL.LIB | IMPORT library ของ AVISDLL.DLL |
| MAKEINI.EXE | โปรแกรมช่วยสร้างแฟ้มข้อมูลขององค์ประกอบ |
| SAMPLES (directory) | ตัวอย่างโปรแกรมองค์ประกอบการจินตทัศน์ |
| SRC (source) | แฟ้มต้นฉบับของ AVISDLL และ โปรแกมช่วยงานต่างๆ |

### 4 ไดเร็คทอรี่ AVIS\SAMPLES

| ชื่อ | รายละเอียด |
|---|---|
| MST (directory) | จัดเก็บตัวอย่างองค์ประกอบสำหรับการจินตทัศนีของอัลกอริทึมค้นหาต้นไม้ทอดข้าม |
| SORT (directory) | จัดเก็บตัวอย่างองค์ประกอบสำหรับการจินตทัศนีของอัลกอริทึมจัดเรียงข้อมูลในหน่วยความจำ |

### 5 ไดเร็คทอรี่ AVIS\SAMPLES\SORT

| ชื่อ | รายละเอียด |
|---|---|
| COLOR (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบแสดงผลแบบแภบสี |
| DOTCNV (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบแปลงคำสั่ง |
| HEAP (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบอัลกอริทึมจัดเรียงข้อมูลแบบฮีป |
| QSORT (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบอัลกอริทึมจัดเรียงข้อมูลแบบเร็ว |
| RANDOM (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบสร้างข้อมูลแบบเลขสุ่ม |
| SELECTC (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบอัลกอริทึมจัดเรียงข้อมูลแบบเลือก(selection sort) ซึ่งพัฒนาด้วยภาษา C |
| STICK (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบแสดงผลแบบแท่งและจุด |

### 6 ไดเร็คทอรี่ AVIS\SAMPLES\MST

| ชื่อ | รายละเอียด |
|---|---|
| GEDIT (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบสร้างข้อมูลแบบกราฟ |
| GPHCNV (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบแปลงคำสั่ง |
| GVIEW (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบแสดงผลของโครงสร้างข้อมูลแบบกราฟ |
| KRUSKAL (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบอัลกอริทึมค้นหาต้นไม้ทอดข้ามที่สั้นที่สุดของครูสกัล |
| PRIM (directory) | จัดเก็บโปรแกรมต้นฉบับขององค์ประกอบอัลกอริทึมค้นหาต้นไม้ทอดข้ามที่สั้นที่สุดของพริม |

### 7 ไดเร็คทอรี่ AVIS\SRC

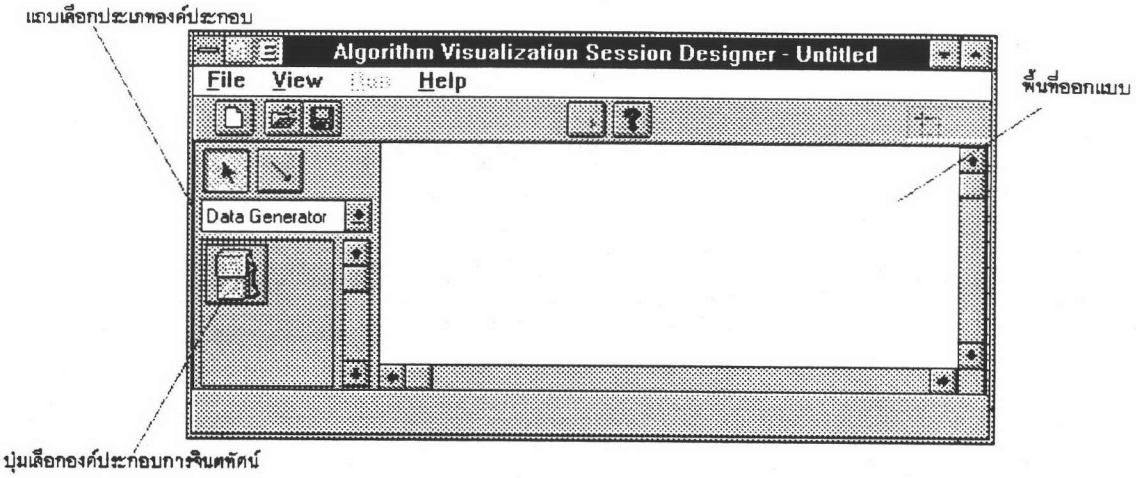
| ชื่อ | รายละเอียด |
|---|---|
| AVISRUN (directory) | จัดเก็บโปรแกรมต้นฉบับของโปรแกรมควบคุมการจินตทัศน์ |
| AVISDSGN (directory) | จัดเก็บโปรแกรมต้นฉบับของโปรแกรมช่วยออกแบบบทการจินตทัศน์ |
| MAKEAVI (directory) | จัดเก็บโปรแกรมต้นฉบับของโปรแกรมช่วยสร้างแฟ้มรูปแบบ AVI |
| MAKEINI (directory) | จัดเก็บโปรแกรมต้นฉบับของโปรแกรมช่วยสร้างแฟ้มข้อมูลขององค์ประกอบ |
| AVISDLL (directory) | จัดเก็บโปรแกรมต้นฉบับของหน่วยบริหารการจินตทัศน์ (AVISDLL .DLL) |
| AVISCOMP (directory) | จัดเก็บโปรแกรมต้นฉบับของตัวควบคุมเฉพาะ AVISCOMP.VBX |
| AVISCTRL (directory) | จัดเก็บโปรแกรมต้นฉบับของตัวควบคุมเฉพาะ AVISCTRL.VBX |

## ภาคผนวก ข.

## คู่มือการใช้งานโปรแกรม AVisDesigner

AVisDesigner (AVISDSGN.EXE) เป็นโปรแกรมบนระบบไมโครซอฟต์วินโดวส์ที่ถูกพัฒนาขึ้นเพื่อช่วยในการออกแบบการจินตทัศน์อัลกอริทึม โดยใช้ขีดความสามารถด้านการติดต่อประสานงานกับผู้ใช้แบบกราฟฟิกของวินโดว์ให้เกิดประโยชน์เพื่อช่วยให้การสร้างบทการจินตทัศน์อัลกอริทึมทำได้สะดวกรวดเร็วยิ่งขึ้น นอกจากนี้หลังจากออกแบบบทการจินตทัศน์อัลกอริทึมเสร็จเรียบร้อยแล้ว AVisDesigner ยังสามารถจะบันทึกบทการจินตทัศน์อัลกอริทึมที่สร้างไว้แล้วเก็บไว้ เพื่อให้ผู้ใช้การจินตทัศน์นำการจินตทัศน์อัลกอริทึมที่สร้างขึ้นมาใช้งานในภายหลังได้อีกด้วย

### *1 เริ่มทำงาน*




รูปที่ ข.1 หน้าจอของโปรแกรม AVisDesigner

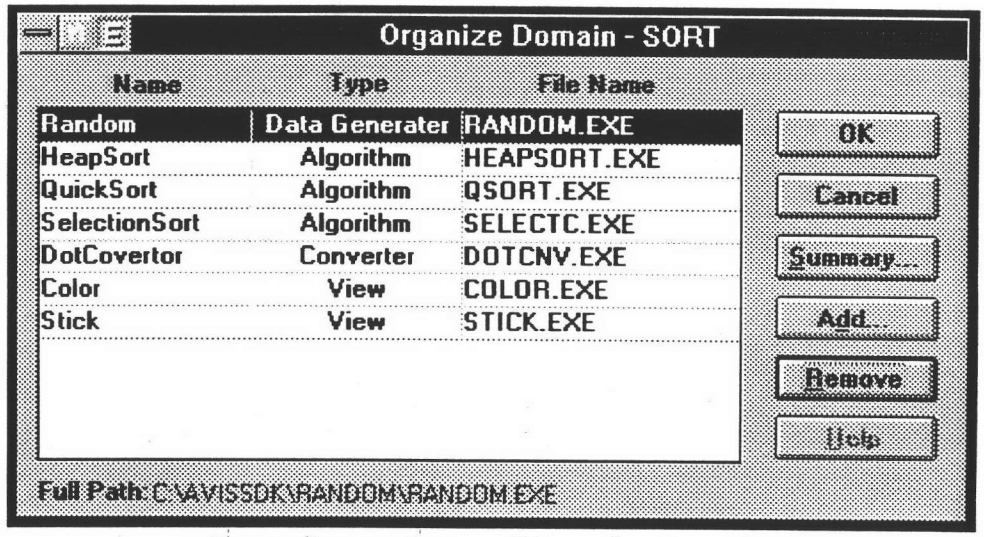
รูปที่ ข.1 แสดงหน้าจอของ AVisDesigner ในขณะทำการออกแบบบทการจินตทัศน์ ในหน้าจอนี้นอกจากส่วนประกอบต่างๆที่เป็นมาตรฐานโดยทั่วไปของโปรแกรมบนวินโดว์เช่น เมนู แถบเลื่อน แถบชื่อโปรแกรม ปุ่มเครื่องมือทางด้านล่างของเมนู และแถบแสดงสถานะโปรแกรมทางด้านล่างของโปรแกรมแล้ว หน้าจอของ AVisDesigner ยังประกอบไปด้วย

1. ปุ่มเครื่องมือซึ่งใช้ในการออกแบบการจินตทัศน์ซึ่งจะมีอยู่สองปุ่มคือ ปุ่มเลือกเครื่องมือ (  ) และปุ่มกำหนดความสัมพันธ์ระหว่างองค์ประกอบ(  )
2. แถบแสดงปุ่มเลือกองค์ประกอบการซึ่งจะแสดงองค์ประกอบการจินตทัศน์ต่างๆ ที่สามารถนำมาใช้ในการออกแบบได้ โดยองค์ประกอบที่ปรากฏขึ้นจะถูกแบ่งเป็นกลุ่มตามประเภทขององค์ประกอบโดยผู้สร้างบทการจินตทัศน์สามารถเลือกประเภทขององค์ประกอบจากแถบเลือกประเภทองค์ประกอบ เพื่อจะลดจำนวนองค์ประกอบที่จะแสดงขึ้นพร้อมๆกัน เพราะหากมีเป็นจำนวนมากผู้สร้างบทการจินตทัศน์อาจสับสนและเลือกใช้ไม่ถูก

3. พื้นที่ออกแบบ ซึ่งจะเป็นพื้นที่สีขาวในรูปที่ ข.1 โดยพื้นที่ออกแบบนี้จะมีขนาดใหญ่กว่าหน้าจอคอมพิวเตอร์จริงจะต้องมีแถบเลื่อนเพื่อใช้เลื่อนเพื่อเลือกแสดงส่วนของพื้นที่ออกแบบที่ต้องการใช้งาน

*2 กลุ่มของปัญหา*

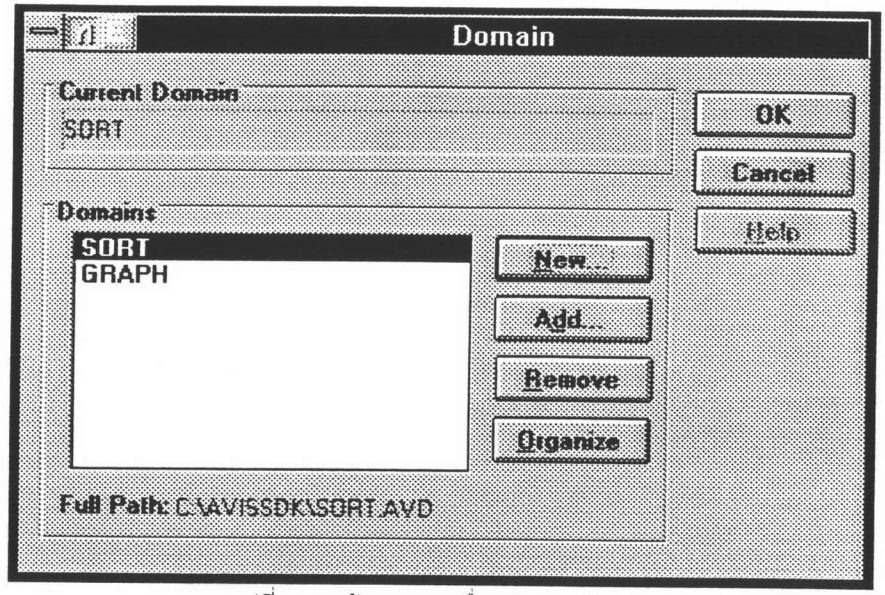
เนื่องจากการที่จะนำองค์ประกอบการจินตทัศน์มาใช้งานร่วมกันได้ องค์ประกอบเหล่านั้นจะต้องรับรู้และเข้าใจข้อความคำสั่งที่ใช้ในการติดต่อกันเป็นอย่างดี AVis เป็นระบบที่ได้รับการออกแบบให้ประยุกต์ใช้งานกับการจินตทัศน์ของอัลกอริทึมได้มากมาย ไม่ขึ้นกับอัลกอริทึมแบบใดแบบหนึ่ง หากในระบบคอมพิวเตอร์หนึ่งๆ มีองค์ประกอบการจินตทัศน์เก็บอยู่มากมายและไม่มีการแบ่งองค์ประกอบเหล่านี้ออกเป็นกลุ่มย่อยๆแล้ว ผู้สร้างบทการจินตทัศน์ซึ่งจะนำองค์ประกอบเหล่านี้ไปใช้งานอาจเกิดความสับสนในการเลือกองค์ประกอบไปใช้งานได้ ทั้งนี้เพราะในการจินตทัศน์ของอัลกอริทึมแบบหนึ่งก็จะมีเพียงองค์ประกอบซึ่งถูกออกแบบให้ใช้กับปัญหาแบบนั้นๆที่สามารถนำมาใช้งานได้ ตัวอย่างเช่น เราไม่สามารถนำส่วนแสดงผลของการจินตทัศน์อัลกอริทึมที่เกี่ยวกับทฤษฎีกราฟไปใช้งานกับการจินตทัศน์ของอัลกอริทึมจัดเรียงข้อมูลได้ ดังนั้นเพื่อลดความสับสนและจำนวนองค์ประกอบที่จะปรากฏขึ้นเพื่อให้ผู้สร้างบทการจินตทัศน์จะนำไปใช้ในการจินตทัศน์หนึ่งๆ AVisDesigner จึงทำการแบ่งองค์ประกอบการจินตทัศน์ออกเป็นกลุ่มย่อยๆตามกลุ่มของปัญหา (problem domain) ต่างๆที่มี โดยผู้พัฒนาองค์ประกอบการจินตทัศน์จะต้องกำหนดว่าองค์ประกอบของตนควรจะอยู่ในกลุ่มปัญหาใด นอกจากนี้ในบางครั้งหากผู้สร้างบทการจินตทัศน์คนใดพบว่าองค์ประกอบการจินตทัศน์บางองค์ประกอบสามารถนำมาใช้งานกับปัญหาที่ตนสนใจได้ ก็อาจเพิ่มองค์ประกอบนั้นเข้ามาในกลุ่มของปัญหาเองก็ได้ ในรูปที่ ข.2 จะแสดงหน้าจอของ AVisDesigner ซึ่งใช้ในการเพิ่มและลดองค์ประกอบจากกลุ่มของปัญหา




รูปที่ ข.2 หน้าจอการเพิ่ม/ลบองค์ประกอบในกลุ่มของปัญหา

หากผู้สร้างบทการจินตทัศน์ต้องการเพิ่มหรือลบองค์ประกอบ จะต้องเลือกคำสั่ง "Domain" ในเมนู "File" จากนั้นจะได้หน้าจอดังในรูปที่ ข.3 ซึ่งจะแสดงชื่อของกลุ่มของปัญหาทั้งหมดที่ AVisDesigner รู้จัก ในหน้าจอนี้หากผู้สร้างบทการจินตทัศน์ไม่พบกลุ่มของปัญหาที่ตนเองต้องการ ก็อาจสร้างกลุ่มของปัญหาขึ้นมาใหม่

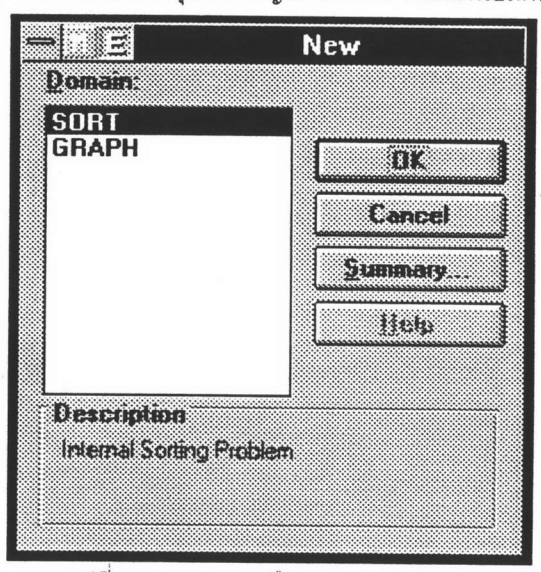
เองก็ได้ เมื่อผู้สร้างบทการจินตทัศน์ทำการเลือกกลุ่มของปัญหาที่ต้องการได้แล้วให้กดปุ่ม Organize เพื่อเข้าสู่หน้าจอในรูปที่ ข.2 ต่อไป




รูปที่ ข.3 หน้าจอแสดงชื่อกลุ่มของปัญหา

### *3 การเริ่มออกแบบบทการจินตทัศน์อัลกอริทึม*

เมื่อผู้สร้างบทการจินตทัศน์เรียกโปรแกรม AVisDesigner มาทำงานและต้องการเริ่มการออกแบบการจินตทัศน์อัลกอริทึมใหม่สามารถทำได้โดยเลือกที่คำสั่ง New ในเมนู File จากนั้นจะปรากฏกรอบโต้ตอบดังในรูปที่ ข.4 เพื่อให้ผู้สร้างบทการจินตทัศน์เลือกกลุ่มของปัญหาของการจินตทัศน์อัลกอริทึมที่จะสร้างขึ้น

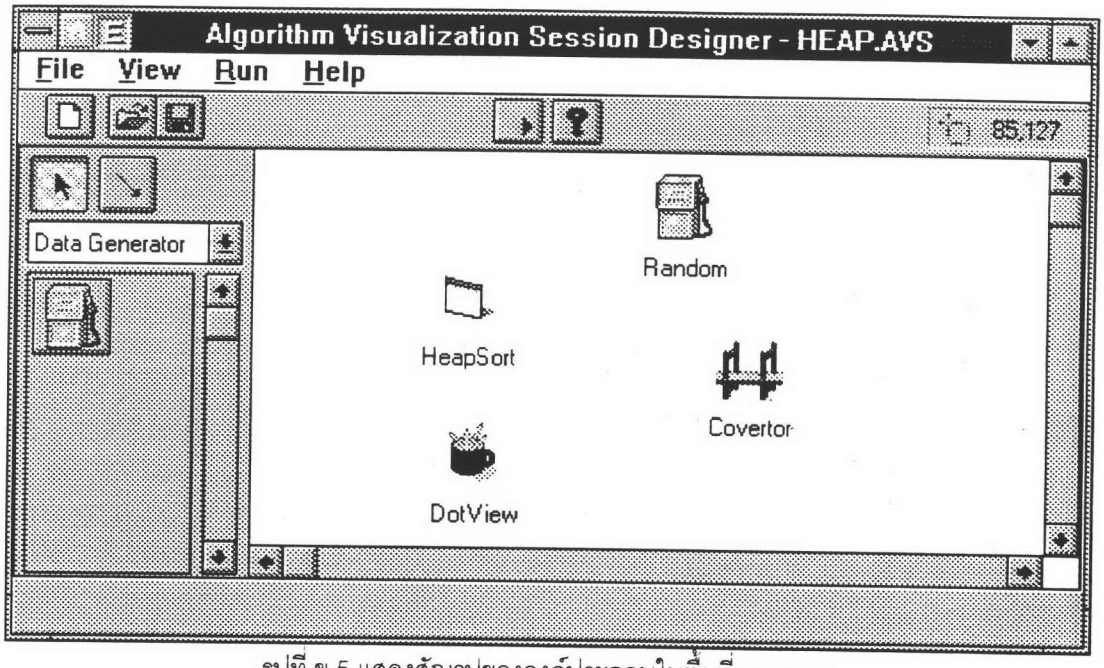



รูปที่ ข.4 แสดงการเลือกกลุ่มของปัญหา

เมื่อเลือกกลุ่มของปัญหาที่ต้องการได้แล้ว ผู้สร้างบทการจินตทัศน์จึงจะเข้าสู่ขั้นตอนของการออกแบบบทการจินตทัศน์ ซึ่งหน้าจอที่ใช้ในการออกแแบบบทการจินตทัศน์ของ AVisDesigner จะมีลักษณะดังในรูปที่ ข.1

*4 การเพิ่มองค์ประกอบเข้าบทสู่การจินตทัศน์*

หลังจากที่ผู้สร้างบทการจินตทัศน์ทำการสร้างหรือเปิดแฟ้มการจินตทัศน์แล้ว ผู้สร้างบทการจินตทัศน์จะต้องกำหนดว่าจะให้มีองค์ประกอบการจินตทัศน์ใดบ้างอยู่ภายในบทการจินตทัศน์ซึ่งกำลังจะสร้างขึ้น การเลือกองค์ประกอบที่ต้องการจะเริ่มจากการเลือกประเภทขององค์ประกอบจากแถบเลือกประเภทองค์ประกอบ จากนั้นจะเห็นสัญรูป (icon) ซึ่งแทนองค์ประกอบต่างๆตามประเภทที่เลือกไว้ปรากฏขึ้น ผู้สร้างบทการจินตทัศน์สามารถเลือกองค์ประกอบที่ตนเองต้องการได้โดยนำเมาส์ไปคลิกที่สัญรูปขององค์ประกอบที่ต้องการ จากนั้นให้เลื่อนเมาส์ไปยังตำแหน่งที่ต้องการบนพื้นที่ออกแบบมาแล้วกดเมาส์ปุ่มซ้ายคางไว้แล้วเลื่อนเมาส์ จะพบว่ามีรูปสี่เหลี่ยมปรากฏขึ้นเมื่อปล่อยปุ่มซ้ายของเมาส์จะปรากฏรูปของสัญรูปซึ่งแทนองค์ประกอบซึ่งต้องการปรากฏขึ้นในพื้นที่ออกแบบดังในรูปที่ ข.5 แสดงว่าผู้สร้างบทการจินตทัศน์ได้เพิ่มองค์ประกอบนั้นเข้ามาในบทการจินตทัศน์แล้ว

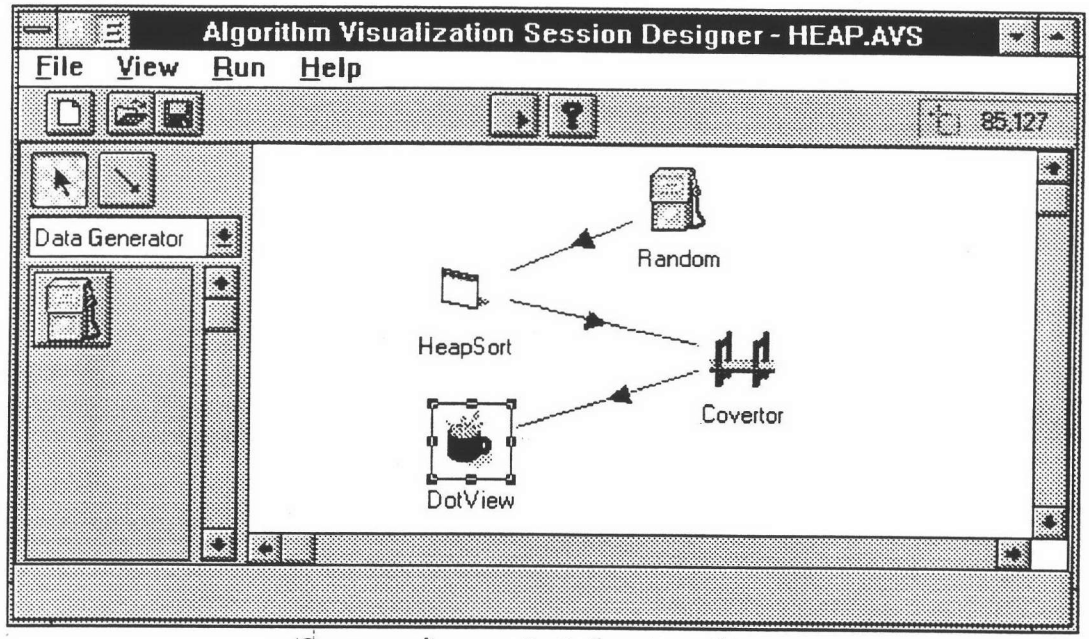



รูปที่ ข.5 แสดงสัญรูปขององค์ประกอบในพื้นที่ออกแบบ

*5 การกำหนดความสัมพันธ์ระหว่างองค์ประกอบการจินตทัศน์*

เมื่อผู้สร้างบทการจินตทัศน์นำองค์ประกอบการจินตทัศน์ที่ต้องการมาวางลงในพื้นที่ออกแบบเรียบร้อยแล้ว ผู้สร้างบทการจินตทัศน์จะต้องกำหนดความสัมพันธ์ระหว่างองค์ประกอบเหล่านี้ เพื่อกำหนดว่าในระหว่างการทำงานองค์ประกอบต่างๆจะส่งข้อความคำสั่งไปให้องค์ประกอบใดบ้าง การกำหนดความสัมพันธ์ระหว่างองค์ประกอบการจินตทัศน์ของโปรแกรม AVisDesigner ทำได้โดยการใช้เมาส์ไปคลิกที่ปุ่มสร้างความสัมพันธ์ จากนั้นให้เลื่อนเมาส์จนตัวชี้ของเมาส์ไปวางทับองค์ประกอบซึ่งเป็นผู้ที่จะส่งข้อมูล ซึ่งก็คือองค์ประกอบที่เป็นผู้ได้รับคำสั่งขอข้อมูล หรือองค์ประกอบซึ่งเป็นผู้ส่งข้อความคำสั่งแสดงผลเช่น ตัวอัลกอริทึม แล้วให้ทำการกดเมาส์ปุ่มซ้ายค้างไว้แล้วเลื่อนเมาส์ไปยังองค์ประกอบซึ่งจะเป็นผู้รับข้อมูล ซึ่งได้แก่ องค์ประกอบซึ่งเป็นผู้ส่งคำสั่งของข้อมูลหรือองค์ประกอบซึ่งเป็นผู้ได้รับคำสั่งข้อมูล เมื่อปล่อยปุ่มซ้ายของเมาส์จะปรากฏลูกศรสีน้ำเงินขึ้นโดยหัวของลูกศรจะแสดงทิศการไหลของข้อมูลดังในรูปที่ ข.6 จะเห็นว่าองค์ประกอบองค์ประกอบหนึ่งเช่น ตัวอัล

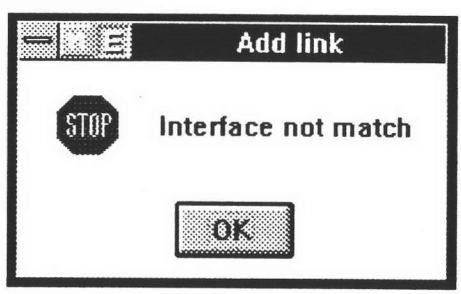
กอริทึม HeapSort สามารถเป็นได้ทั้งผู้ให้และผู้รับข้อมูลได้ในเวลาเดียวกัน โดยองค์ประกอบนี้จะรับข้อมูลจากตัวสร้างข้อมูล Random และส่งข้อมูลและคำสั่งไปให้ตัวแปลงคำสั่ง Convertor




รูปที่ ข.6 การสร้างความสัมพันธ์ระหว่างองค์ประกอบ

ในบางครั้งองค์ประกอบที่ผู้สร้างบทการจินตทัศน์ต้องการสร้างความสัมพันธ์ระหว่างองค์ประกอบที่ไม่สามารถทำงานร่วมกันได้ เนื่องจากมีองค์ประกอบใดองค์ประกอบหนึ่งไม่รู้จักหรือไม่อาจตอบสนองข้อความคำสั่งของอีกองค์ประกอบหนึ่งได้ AVisDesigner จะไม่ยอมให้ผู้สร้างบทการจินตทัศน์ทำการสร้างความสัมพันธ์ระหว่างองค์ประกอบทั้งสอง โดยจะแสดงกรอบข้อความดังในรูปที่ ข.7

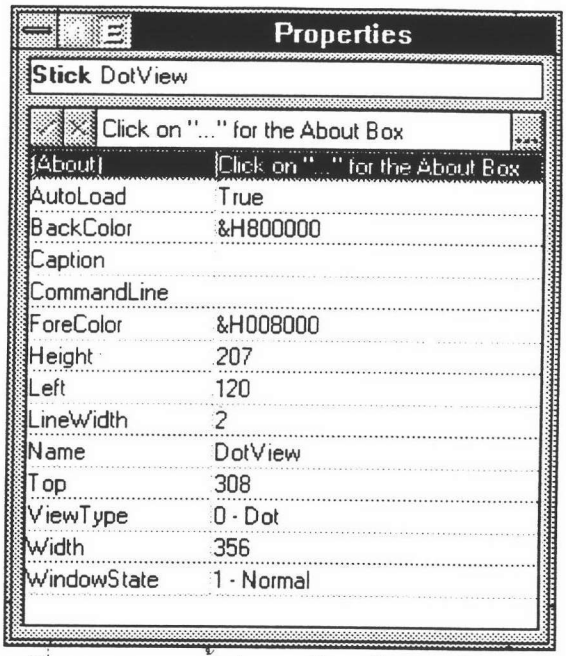
การที่ AVisDesigner จะทราบว่าองค์ประกอบคู่ใดที่สามารถทำงานร่วมกันได้หรือไม่นั้น AVisDesigner จะตรวจสอบจากข้อมูลในแฟ้มรายละเอียดขององค์ประกอบตามที่กล่าวมาแล้ว เพื่อตรวจสอบว่าองค์ประกอบคู่หนึ่งๆจะทำงานร่วมกันได้หรือไม่




รูปที่ ข.7 ข้อความแสดงการปฏิเสธการสร้างความสัมพันธ์ระหว่างองค์ประกอบ

### *6 การกำหนดค่าพารามิเตอร์ขององค์ประกอบการจินตทัศน์*

ขั้นตอนสุดท้ายของการสร้างการจินตทัศน์อัลกอริทึมด้วย AVisDesigner ก็คือ การทำค่าพารามิเตอร์ต่างๆขององค์ประกอบ ซึ่งการตั้งค่าด้วยพารามิเตอร์ AVisDesigner สามารถทำได้โดยการเลื่อนเมาส์ไปคลิกที่องค์ประกอบซึ่งต้องการตั้งค่าแล้วกดเมาส์ปุ่มขวาหรือกดปุ่ม F4 จะปรากฏหน้าจอการตั้งค่าพารามิเตอร์ขององค์ประกอบ ตัวอย่างเช่นในรูปที่ ข.8 ซึ่งแสดงการตั้งค่าพารามิเตอร์ขององค์ประกอบ DotView




รูปที่ ข.8 แสดงการตั้งค่าพารามิเตอร์ขององค์ประกอบ

หน้าจอสำหรับตั้งค่าพารามิเตอร์ขององค์ประกอบการจินตทัศน์จะถูกแบ่งออกเป็นสามส่วนคือ

1. ส่วนบนสุดจะแสดงชื่อขององค์ประกอบการจินตทัศน์(Stick)ที่ผู้พัฒนาองค์ประกอบเป็นผู้ตั้งและชื่อเรียกขององค์ประกอบในบทการจินตทัศน์ซึ่งกำลังออกแบบ(DotView)ที่ผู้สร้างบทการจินตทัศน์เป็นผู้กำหนด
2. ส่วนที่สองซึ่งอยู่ใต้ส่วนแรก เป็นพื้นที่สำหรับแก้ไขค่าพารามิเตอร์ประกอบไปด้วยปุ่มแก้ไขค่า (✓) ปุ่มยกเลิกการแก้ไข (✗) พื้นที่แสดง/แก้ไขค่า และปุ่มเลือกค่า
3. ส่วนประกอบสุดท้ายของหน้าจอแก้ไขค่าพารามิเตอร์ก็คือ ส่วนแสดงค่าพารามิเตอร์ ซึ่งจะแสดงค่าพารามิเตอร์ทั้งหมดขององค์ประกอบ

ผู้สร้างบทการจินตทัศน์สามารถปรับเปลี่ยนค่าพารามิเตอร์โดยใช้เมาส์เลื่อนไปคลิกที่ชื่อของพารามิเตอร์ซึ่งต้องการแก้ไข ซึ่งจะสังเกตเห็นได้ว่าชื่อและค่าของพารามิเตอร์จะถูกแสดงด้วยแถบสีน้ำเงิน และในพื้นที่แก้ไขค่าก็จะปรากฏค่าพารามิเตอร์ที่ต้องการแก้ไขขึ้น ผู้สร้างบทการจินตทัศน์สามารถแก้ไขค่าพารามิเตอร์ได้โดยนำเมาส์ไปคลิกที่ปุ่มเลือกค่า หรือหากไม่มีปุ่มเลือกค่าปรากฏขึ้นผู้สร้างบทการจินตทัศน์สามารถแก้ไขค่าพารามิเตอร์ได้โดยตรงโดยการเลื่อนเมาส์ไปคลิกที่พื้นที่แก้ไขค่าแล้วทำการแก้ไขค่าผ่านทางแป้นพิมพ์

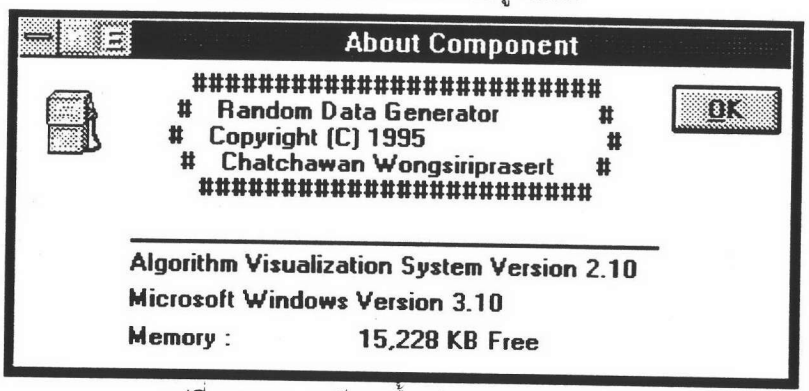
ค่าพารามิเตอร์ขององค์ประกอบจะแบ่งได้เป็นสองกลุ่มได้แก่ ค่าพารามิเตอร์ซึ่ง AVisDesigner เป็นผู้กำหนดขึ้น และค่าพารามิเตอร์ซึ่งผู้พัฒนาองค์ประกอบการจินตทัศน์เป็นผู้กำหนดเอง

ค่าพารามิเตอร์ซึ่ง AVisDesigner กำหนดขึ้นส่วนใหญ่จะเป็นค่าพารามิเตอร์ที่โปรแกรมควบคุมการจินตทัศน์ที่จะกล่าวถึงในภายหลังจะต้องใช้ในการเรียกบทการจินตทัศน์มาทำงาน ค่าพารามิเตอร์เหล่านี้ได้แก่

1. *(About)* เป็นค่าพารามิเตอร์ซึ่งอ่านค่าได้เพียงอย่างเดียว ซึ่งหากผู้สร้างบทการจินตทัศน์เลือกพารามิเตอร์ตัวนี้และคลิกที่ปุ่มเลือกค่า AVisDesigner จะแสดงกรอบข้อความรายละเอียดขององค์ประกอบดังในรูปที่ ข.9

2. *AutoLoad* เป็นค่าซึ่งใช้ในการตรวจแก้ (debug) องค์ประกอบการจินตทัศน์ ผู้สร้างบทการจินตทัศน์ทั่วไปไม่จำเป็นต้องแก้ไขค่าพารามิเตอร์ตัวนี้ โดยทั่วไปค่านี้จะถูกตั้งให้เป็น True รายละเอียดของความหมายของค่าพารามิเตอร์ตัวนี้จะกล่าวถึงอีกครั้งในบทที่สี่
3. *CommandLine* เป็นค่า Command Line ซึ่งจะใช้ในการเรียกองค์ประกอบการจินตทัศน์มาใช้ในการทำงาน ค่านี้มักใช้ในขั้นตอนของการพัฒนาองค์ประกอบการจินตทัศน์โดยผู้พัฒนาองค์ประกอบการจินตทัศน์มากกว่าจะถูกใช้โดยผู้สร้างบทการจินตทัศน์ทั่วไป
4. *Height, Left, Top* และ *Width* จะเก็บขนาดและตำแหน่งของหน้าต่าง (Window) ขององค์ประกอบการจินตทัศน์เมื่อเริ่มทำการจินตทัศน์
5. *WindowState* ใช้ในการกำหนดค่าลักษณะของหน้าต่างขององค์ประกอบเมื่อเริ่มทำการจินตทัศน์

ส่วนรายละเอียดและความหมายของค่าพารามิเตอร์อีกกลุ่มหนึ่งซึ่งผู้พัฒนาองค์ประกอบจินตทัศน์จะเป็นผู้กำหนดนั้น ผู้สร้างบทการจินตทัศน์จะต้องค้นหาจากแฟ้มคู่มือการใช้ของตัวองค์ประกอบเอง ซึ่งหากผู้พัฒนาองค์ประกอบแจกจ่ายแฟ้มคู่มือนี้มาพร้อมกับตัวองค์ประกอบ และแจ้งชื่อแฟ้มคู่มือนี้ไว้ในแฟ้มรายละเอียดขององค์ประกอบ ผู้สร้างบทการจินตทัศน์โปรแกรมของ AVisDesigner สามารถขอดูแฟ้มคู่มือได้โดยการกดปุ่ม F1 ในขณะแก้ไขค่าพารามิเตอร์หรือในขณะที่องค์ประกอบถูกเลือก




รูปที่ ข.9 รายละเอียดเบื้องต้นขององค์ประกอบ

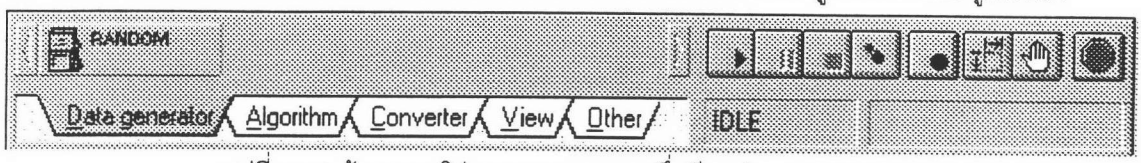
## ภาคผนวก ค.

## คู่มือการใช้งานโปรแกรม AVisController

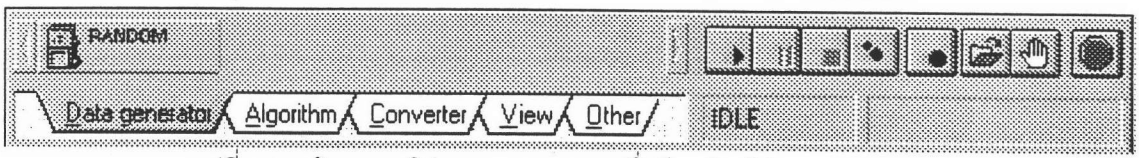
AVisController (AVISRUN.EXE) เป็นโปรแกรมควบคุมการจินตทัศน์ของระบบ AVis ที่ทำงานบนระบบไมโครซอฟต์วินโดวส์เช่นเดียวกับ AVisDesigner

### *1 เริ่มต้นทำงาน*

ผู้ใช้การจินตทัศน์สามารถเรียกโปรแกรม AVisController มาทำงานได้สองวิธีคือ การเรียกโดยตรงผ่านโปรแกรมเมเนเจอร์ของไมโครซอฟต์วินโดวส์ หรือเรียกผ่านเมนู Run ของโปรแกรม AVisDesigner ก็ได้ ซึ่งหน้าจอของ AVisController ซึ่งเรียกจากโปรแกรมทั้งสองจะต่างกันเล็กน้อยดังในรูปที่ ค.1 และ รูปที่ ค.2




รูปที่ ค.1 หน้าจอของโปรแกรม AVISRUN ซึ่งเรียกผ่าน AVISDSGN




รูปที่ ค.2 หน้าจอของโปรแกรม AVISRUN ซึ่งเรียกผ่านโปรแกรมเมเนเจอร์

ข้อแตกต่างระหว่างหน้าจอทั้งสองคือโปรแกรมซึ่งเรียกจากโปรแกรมเมเนเจอร์จะมีปุ่มเปิดแฟ้มการจินตทัศน์ (  ) ในขณะที่โปรแกรมซึ่งเรียกจาก AVisDesigner จะไม่มี แต่จะมีปุ่มจัดเก็บตำแหน่งองค์ประกอบ (  ) แทน ทั้งนี้เพราะหาก AVisController ถูกเรียกจากโปรแกรมเมเนเจอร์แสดงว่าผู้ที่เรียกใช้น่าจะเป็นผู้ใช้การจินตทัศน์ทั่วไปซึ่งไม่ควรที่จะมีสิทธ์แก้ไขบทการจินตทัศน์ที่ถูกสร้างไว้แล้ว

หน้าจอของ AVisController จะถูกแบ่งออกเป็นสี่ส่วน โดยส่วนแรกซึ่งจะอยู่ทางครึ่งซ้ายมือจะแสดงปุ่มซึ่งใช้ในการตั้งค่าพารามิเตอร์ขององค์ประกอบขณะทำงาน โดยปุ่มเหล่านี้จะถูกจัดเป็นกลุ่มตามประเภทขององค์ประกอบ ส่วนที่สองจะเป็นปุ่มคำสั่งต่างๆที่ใช้ในการควบคุมการจินตทัศน์และการทำงานของ AVisController ส่วนประกอบส่วนที่สามของ AVisController ก็คือ แถบแสดงสถานะของการจินตทัศน์ (ในรูปจะปรากฏคำว่า IDLE ขึ้นอยู่แสดงว่ายังไม่ได้เริ่มการจินตทัศน์) และส่วนประกอบสุดท้ายของ AVisController ก็คือแถบแสดงสถานะของโปรแกรมซึ่งอยู่ถัดมาทางด้านขวาของแถบแสดงสถานะของการจินตทัศน์

### *2 ปุ่มควบคุมการทำงาน*

ผู้ใช้การจินตทัศน์สามารถควบคุมการทำงานของ AVisController ได้โดยการใช้ปุ่มคำสั่งต่างๆของ AVisController ที่ความหมายแตกต่างกันออกไปดังนี้

1. *ปุ่มควบคุมการจินตทัศน์* ปุ่มที่ใช้ในการควบคุมการจินตทัศน์จะมีอยู่ทั้งหมดสี่ปุ่มคือ ปุ่มเริ่มทำงาน(-play) ปุ่มหยุดทำงานชั่วคราว(  -pause) ปุ่มยกเลิกการทำงาน(  -reset) และปุ่มทำงานทีละขั้น(  -step) ซึ่งความหมายและวิธีใช้งานปุ่มเหล่านี้จะกล่าวถึงในหัวข้อต่อไป
2. *ปุ่มบันทึกการจินตทัศน์*(  ) ปุ่มนี้จะใช้ในการเก็บบันทึกการทำงานของส่วนแสดงผลลงในแฟ้มรูปแบบ AVI ซึ่งผู้ใช้การจินตทัศน์สามารถนำแฟ้มนี้ไปใช้แสดงการทำงานของอัลกอริทึมโดยไม่ต้องใช้ระบบ AVis ในภายหลังด้วยโปรแกรมแสดงภาพเคลื่อนไหว อย่างไรก็ตามการบันทึกการทำงานนี้สามารถใช้ได้เฉพาะบนเครื่องคอมพิวเตอร์ที่มีระบบ Microsoft Video for Windows (VFW) ติดตั้งอยู่แล้วเท่านั้น หากระบบคอมพิวเตอร์ใดไม่มีระบบ VFW ติดตั้งอยู่ AVisController จะทำการซ่อนปุ่มนี้ไว้ไม่แสดงให้เห็นบนหน้าจอ รายละเอียดของปุ่มนี้จะกล่าวถึงอีกครั้งในภายหลัง
3. *ปุ่มเปิดแฟ้มการจินตทัศน์* ปุ่มนี้จะใช้ในการเปิดแฟ้มการจินตทัศน์อัลกอริทึมที่ต้องการขึ้นมาทำงาน ปุ่มนี้จะปรากฏเฉพาะเมื่อมีการเรียก AVisController มาทำงานจากโปรแกรมเมเนเจอร์เท่านั้น
4. *ปุ่มจัดเก็บตำแหน่งองค์ประกอบ* ใช้ในการเก็บตำแหน่งบนหน้าจอขององค์ประกอบการจินตทัศน์ลงไปในแฟ้มการจินตทัศน์ ทั้งนี้เพื่อที่ผู้ใช้การจินตทัศน์จะได้ไม่ต้องทำการตั้งค่าตำแหน่งหน้าต่างขององค์ประกอบใหม่ทุกครั้งที่มีการเรียกการจินตทัศน์มาทำงาน ปุ่มนี้จะปรากฏขึ้นเฉพาะเมื่อมีการเรียก AVisController มาทำงานจาก AVisDesigner เท่านั้น เพื่อให้ผู้สร้างบทการจินตทัศน์สามารถกำหนดตำแหน่งขององค์ประกอบได้จากการทดลองจัดบนหน้าจอจริง
5. *ปุ่มตั้งค่าตัวเลือก*(  ) ใช้ในการตั้งค่าตัวเลือกต่างๆของ AVisController
6. *ปุ่มจบการทำงาน*(  ) ใช้เพื่อจบการทำงานของ AVisController

### *3 การควบคุมการทำงานของการจินตทัศน์*

ผู้ใช้การจินตทัศน์ระบบ AVis สามารถควบคุมลักษณะการทำงานของการจินตทัศน์ หรืออีกนัยหนึ่งก็คือ ลักษณะการทำงานของตัวอัลกอริทึม ทั้งนี้เพราะการทำงานของการจินตทัศน์อัลกอริทึมจะมีศูนย์กลางหรือจุดเริ่มอยู่ที่อัลกอริทึม ผู้ใช้การจินตทัศน์ AVisController สามารถควบคุมการทำงานของ AVis ให้ทำงานได้ในห้าลักษณะ ผ่านปุ่มควบคุมการจินตทัศน์ที่กล่าวมาแล้วข้างต้นคือ

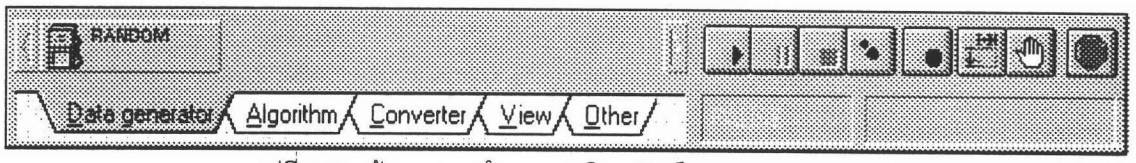
1. *ทำงานปรกติ*(run) อัลกอริทึมจะทำงานไปจนเสร็จเรียบร้อย มักใช้เพื่อศึกษาผลการทำงานอย่างต่อเนื่อง
2. *หยุดทำงานชั่วคราว*(pause) อัลกอริทึมจะหยุดทำงานชั่วคราว ดังนั้นผู้ใช้การจินตทัศน์จึงสามารถศึกษาผลการทำงานในขณะใดขณะหนึ่งอย่างละเอียด
3. *ทำงานทีละขั้น*(step) อัลกอริทึมจะทำงานทีละขั้นแล้วหยุดรอให้ผู้ใช้การจินตทัศน์สั่งให้ทำงานต่อ ซึ่งปริมาณของงานซึ่งจะทำในหนึ่งขั้นนี้จะขึ้นอยู่กับการกำหนดของผู้พัฒนาตัวองค์ประกอบอัลกอริทึม

4. *ยกเลิกการทำงาน*(reset) อัลกอริทึมจะยุติการทำงานลงทันทีแล้วกลับไปรอเตรียมพร้อมที่จะทำงานใหม่เมื่อผู้ใช้การจินตทัศน์สั่ง
5. *ทำงานแบบมีการหน่วงเวลา*(Animation) เป็นการทำงานผสมระหว่างแบบแรกและแบบที่สาม นั่นคืออัลกอริทึมจะทำงานไปจนเสร็จเรียบร้อยคล้ายกับแบบแรก แต่ในระหว่างการทำงานหลังจากที่ทำงานเสร็จหนึ่งขั้นก็จะถูกหน่วงเวลาตามที่ผู้ใช้การจินตทัศน์กำหนดไว้ก่อนที่จะทำงานต่อ ผู้ใช้การจินตทัศน์สามารถสั่งให้เกิดการทำงานในลักษณะนี้ได้โดยการใช้ปุ่มเริ่มทำงานและตั้งค่าหน่วงเวลา(delay time)ให้มากกว่าศูนย์ ซึ่งการตั้งค่าหน่วงเวลาจะกล่าวถึงในหัวข้อการตั้งค่าตัวเลือกของ AVisController

ในการจินตทัศน์อัลกอริทึมหนึ่งๆผู้ใช้การจินตทัศน์สามารถปรับเปลี่ยนลักษณะการทำงานของการจินตทัศน์ได้ตลอดเวลาในขณะทำการจินตทัศน์ โดยไม่จำเป็นต้องใช้การทำงานแบบเดียวกันตลอดการจินตทัศน์ ตัวอย่างเช่น ในตอนแรกผู้ใช้การจินตทัศน์อาจให้อัลกอริทึมทำงานแบบปกติไปก่อน แต่เมื่อเกิดเหตุการณ์บางอย่างที่น่าสนใจขึ้น ผู้ใช้อาจสั่งให้อัลกอริทึมหยุดทำงานชั่วคราว เพื่อสังเกตถึงการทำงานที่น่าสนใจในจุดนั้น จากนั้นจึงค่อยให้อัลกอริทึมค่อยๆทำงานไปทีละขั้น เพื่อดูการทำงานอย่างละเอียด และเมื่อการทำงานของอัลกอริทึมผ่านจุดที่เราสนใจไปแล้วก็อาจให้อัลกอริทึมทำงานแบบปกติต่อไป ผู้ใช้การจินตทัศน์สามารถตรวจดูสถานะการทำงานของการจินตทัศน์ได้จากแถบแสดงสถานะการทำงาน ซึ่งสถานะการทำงานของการจินตทัศน์จะมีอยู่ห้าแบบคือ

1. RUN เป็นการทำงานตามปกติของการจินตทัศน์ การจินตทัศน์จะทำงานอยู่ในสถานะนี้เมื่อผู้ใช้การจินตทัศน์กดปุ่มเริ่มทำงานและตัวเลือกค่าหน่วงเวลาเป็นศูนย์
2. STEP แสดงว่าการจินตทัศน์กำลังทำงานแบบทำทีละขั้น ซึ่งจะเกิดขึ้นเมื่อผู้ใช้การจินตทัศน์สั่งให้การจินตทัศน์ทำงานด้วยปุ่มทำงานทีละขั้น
3. PAUSE แสดงว่าผู้ใช้การจินตทัศน์หยุดการจินตทัศน์ชั่วคราวด้วยการกดปุ่มหยุดทำงานชั่วคราว
4. IDLE แสดงว่าระบบยังไม่ได้เริ่มการจินตทัศน์
5. ANIM แสดงว่าการจินตทัศน์กำลังทำงานแบบหน่วงเวลา การจินตทัศน์จะทำงานอยู่ในสถานะนี้เมื่อผู้ใช้การจินตทัศน์กดปุ่มเริ่มทำงานและตัวเลือกค่าหน่วงเวลามีค่ามากกว่าศูนย์

ในรูปที่ ค.3 แสดงหน้าจอขณะที่กำลังทำการจินตทัศน์การจัดเรียงข้อมูลแบบฮีป และผู้ใช้กำลังหยุดการทำงานของการจินตทัศน์ชั่วคราว

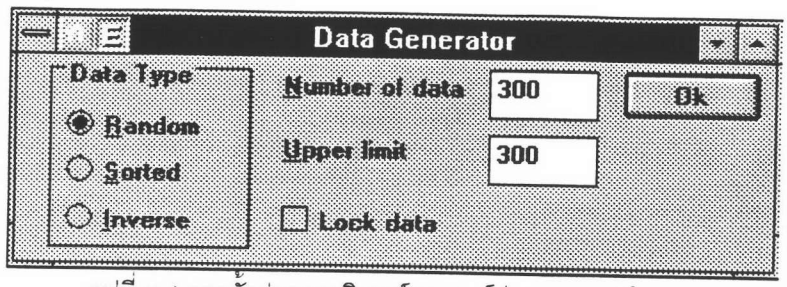



รูปที่ ค.3 หน้าจอขณะทำการการจินตทัศน์ของ AVisController

### *4 การกำหนดค่าพารามิเตอร์ขององค์ประกอบการจินตทัศน์*

นอกจากค่าพารามิเตอร์ซึ่งสามารถตั้งค่าได้ในขณะสร้างบทการจินตทัศน์แล้ว องค์ประกอบบางตัวอาจมีค่าพารามิเตอร์ซึ่งผู้ใช้การจินตทัศน์สามารถกำหนดได้ในขณะทำการจินตทัศน์อีกด้วย ซึ่งในกรณีที่องค์ประกอบมีพารามิเตอร์เช่นนี้ ผู้ใช้สามารถตั้งค่าพารามิเตอร์เหล่านี้ได้โดยการนำเมาส์ไปกดที่ปุ่มตั้งค่าองค์

ประกอบ ในรูปที่ ค.4 แสดงหน้าจอการตั้งค่าพารามิเตอร์ของตัวสร้างข้อมูลแบบสุ่มซึ่งใช้สำหรับการจินตทัศน์ของอัลกอริทึมจัดเรียงข้อมูล

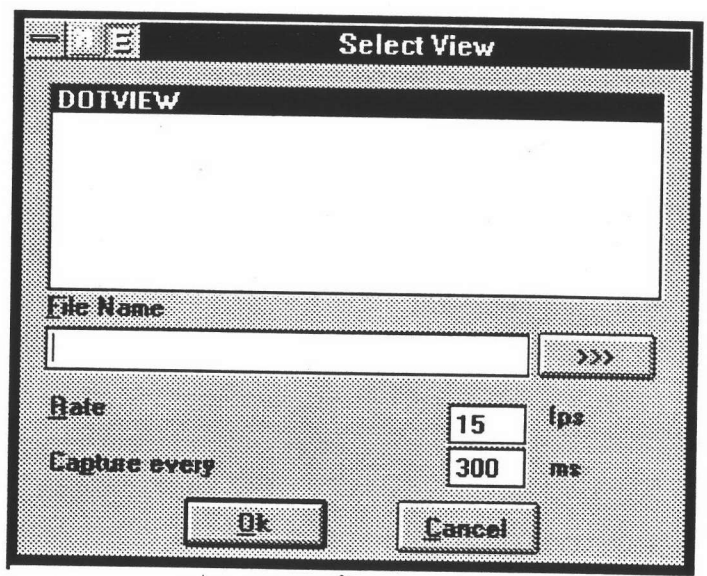



รูปที่ ค.4 การตั้งค่าพารามิเตอร์ขององค์ประกอบการทำงาน

หน้าจอของการตั้งค่าพารามิเตอร์ขององค์ประกอบการจินตทัศน์แต่ละตัวจะแตกต่างกันออกไปขึ้นอยู่กับการออกแบบของผู้พัฒนาองค์ประกอบว่าจะออกแบบให้หน้าจอเป็นเช่นใด และมีค่าพารามิเตอร์ใดที่สามารถกำหนดได้บ้าง

### *5 การเก็บบันทึกภาพการทำงานของการจินตทัศน์*

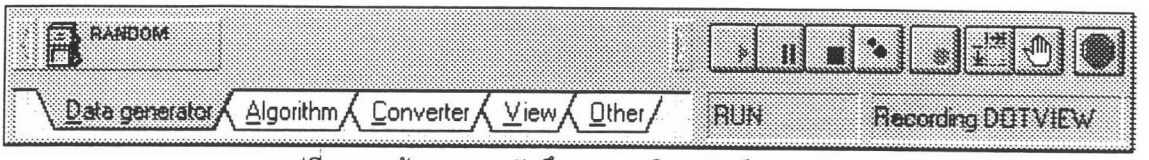
ในระบบคอมพิวเตอร์ซึ่งมีการติดตั้ง Microsoft Video for Windows รุ่น 1.1 ขึ้นไป ผู้ใช้สามารถบันทึกภาพการทำงานของส่วนแสดงผลเก็บไว้ในแฟ้มภาพเคลื่อนไหวรูปแบบ AVI โดยผู้ใช้สามารถนำแฟ้มภาพเคลื่อนไหวนี้ไปใช้เพื่อสังเกตและศึกษาการทำงานของอัลกอริทึมได้ในภายหลังโดยไม่ต้องใช้ระบบ AVis ซึ่งจะมีประโยชน์มากหากผู้ใช้ต้องการนำผลการทำงานที่ได้ไปนำเสนอในรูปแบบอื่นๆ เช่น การนำเสนอในรูปเอกสารอิเล็กโทรนิค




รูปที่ ค.5 หน้าจอตั้งค่าการบันทึกภาพ

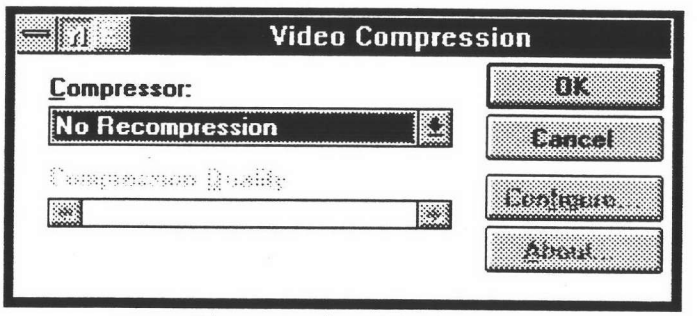
ขั้นตอนในการบันทึกการทำงานนั้นผู้ใช้จะต้องใช้เมาส์เลื่อนไปคลิกที่ปุ่มบันทึกการทำงานก่อนที่จะเริ่มทำการจินตทัศน์ จากนั้นจะปรากฏหน้าจอขึ้นเพื่อให้ผู้ใช้ผลิตส่วนแสดงผล (View) ที่ต้องการเก็บบันทึกภาพ จำนวนภาพ (frame) ต่อหนึ่งวินาที ช่วงเวลาห่างของภาพแต่ละภาพ และชื่อแฟ้มภาพเคลื่อนไหว ดังในรูปที่ ค.5

จากนั้นเมื่อผู้ใช้เริ่มการจินตทัศน์ AVisController จะเริ่มบันทึกภาพซึ่งจะสังเกตได้จากแถบสถานะของ AVisController จะปรากฏคำว่า "Recording : [Viewname]" โดย [Viewname] จะเป็นชื่อส่วนแสดงผลที่ผู้ใช้เลือกไว้แล้ว ดังในรูปที่ ค.6



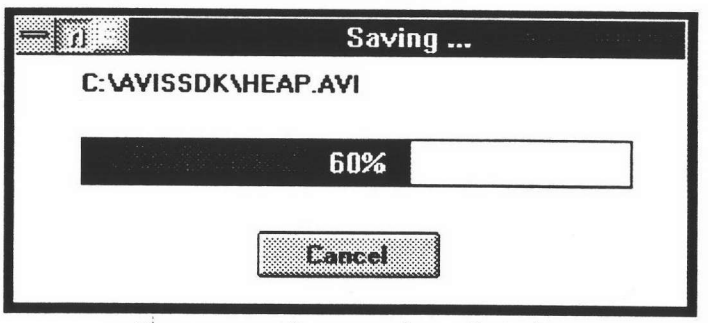

รูปที่ ค.6 หน้าจอขณะบันทึกการการจินตทัศน์ของ AVisController

เมื่อการจินตทัศน์สิ้นสุดลงจะปรากฏหน้าจอเลือกวิธีบีบอัดข้อมูลขึ้นดังในรูปที่ ค.7 ทั้งนี้เพื่อทำการลดขนาดของแฟ้มภาพเคลื่อนไหว วิธีการบีบอัดต่างๆที่ปรากฏขึ้นนั้นจะแตกต่างกันออกไปขึ้นอยู่กับระบบเครื่องคอมพิวเตอร์แต่ละระบบ โดยแต่ละวิธีจะมีข้อดีและข้อเสียแตกต่างกันออกไป ซึ่งรายละเอียดของวิธีบีบอัดข้อมูลแต่ละวิธีผู้ใช้จะต้องศึกษาจากคู่มือของระบบ Microsoft Video for Window เอง




รูปที่ ค.7 การเลือกวิธีบีบอัดข้อมูล

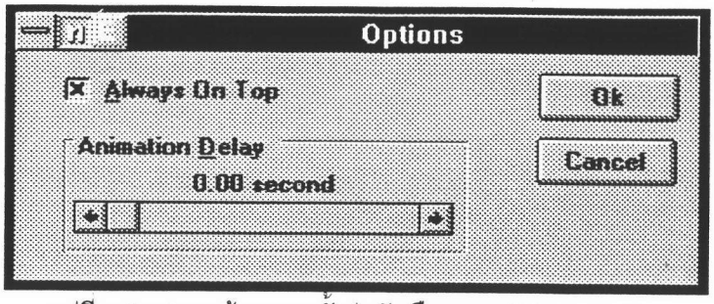
หลังจากผู้ใช้เลือกวิธีบีบอัดข้อมูลแล้ว AVisController จะทำการบีบอัดข้อมูล ซึ่งจะปรากฏหน้าจอเพื่อแสดงการทำงานดังในรูปที่ ค.8 เพื่อแสดงปริมาณงานซึ่งกระทำไปแล้ว และเมื่อเสร็จสิ้นขั้นตอนนี้ก็จะเกิดแฟ้มภาพเคลื่อนไหวขึ้นตามชื่อที่ผู้ใช้กำหนดไว้แล้วในตอนต้น




รูปที่ ค.8 แสดงหน้าจอขณะทำการบีบอัดข้อมูล

### *6 การตั้งค่าตัวเลือกของ AVisController*

เมื่อผู้ใช้กดปุ่มตั้งค่าตัวเลือกของ AVisController จะปรากฏหน้าจอดังในรูปที่ ค.9 โปรแกรม AVisController จะมีตัวเลือกให้ผู้ใช้ตั้งค่าได้สองตัวคือ เวลาหน่วงที่ใช้ในขณะทำงานแบบอนิเมชัน ซึ่งผู้ใช้สามารถแก้ไขได้โดยการเลื่อนแถบเลื่อนซึ่งเวลาหน่วงจะปรับได้ตั้งแต่ 0 ถึง 4 วินาที นอกจากค่าหน่วงเวลาแล้วค่าอีกตัวที่ผู้ใช้สามารถปรับได้คือ ค่าซึ่งกำหนดว่าจะให้ตัว AVisController ปรากฏบนหน้าจอเสมอโดยไม่ถูกโปรแกรมอื่นบัง ซึ่งจะทำให้ใช้งาน AVisController ได้สะดวกขึ้น

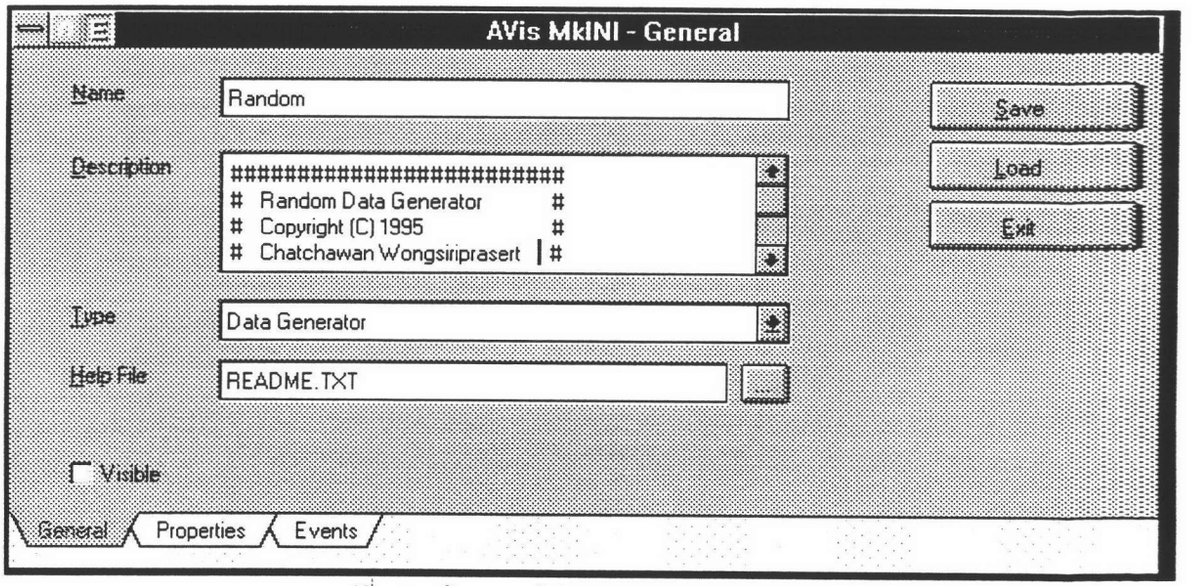



รูปที่ ค.9 แสดงหน้าจอการตั้งค่าตัวเลือกของ AVisController

## ภาคผนวก ง.

## คู่มือการใช้งานโปรแกแกรม AVisIniMaker

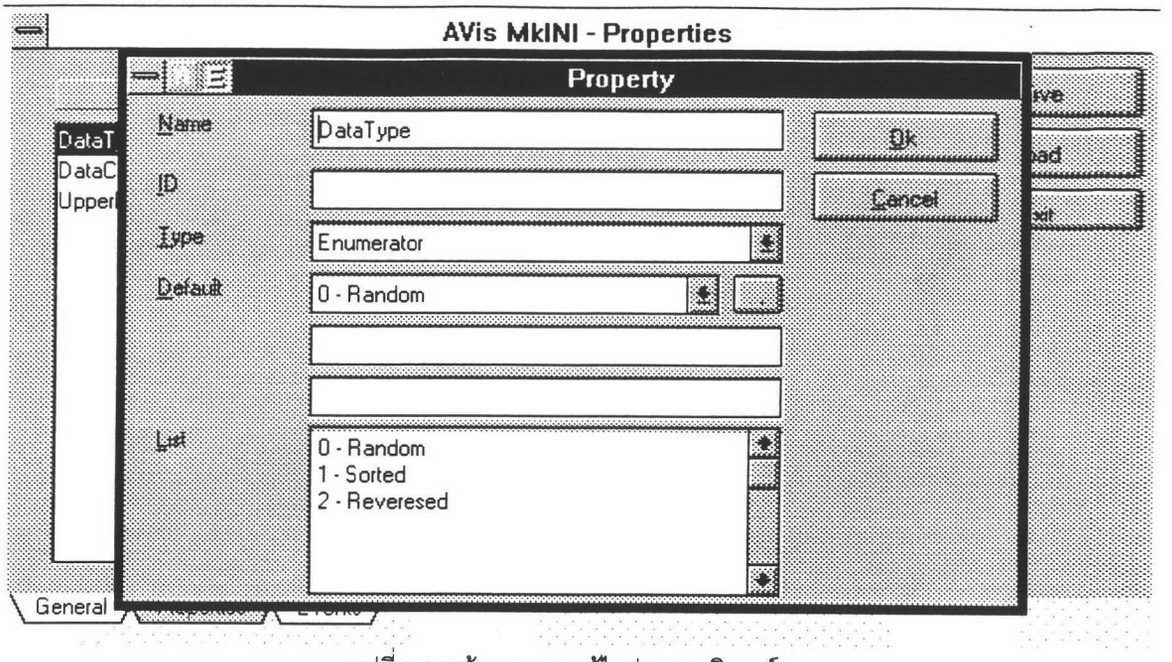
AVisIniMaker (MAKEINI.EXE) เป็นโปรแกรมช่วยงานอีกตัวหนึ่งของ AVis โปรแกรมนี้จะช่วยให้ผู้พัฒนาองค์ประกอบการจินตทัศน์สร้างแฟ้มรายละเอียดขององค์ประกอบได้ง่ายขึ้น โดยโปรแกรมนี้จะแปลงข้อมูลต่างๆในแฟ้มรายละเอียดให้อยู่ในรูปที่เข้าใจและแก้ไขได้ง่ายขึ้น ในรูปที่ ง.1 จะแสดงหน้าจอของโปรแกรมนี้




รูปที่ ง.1 หน้าจอของโปรแกรม MAKEINI.EXE

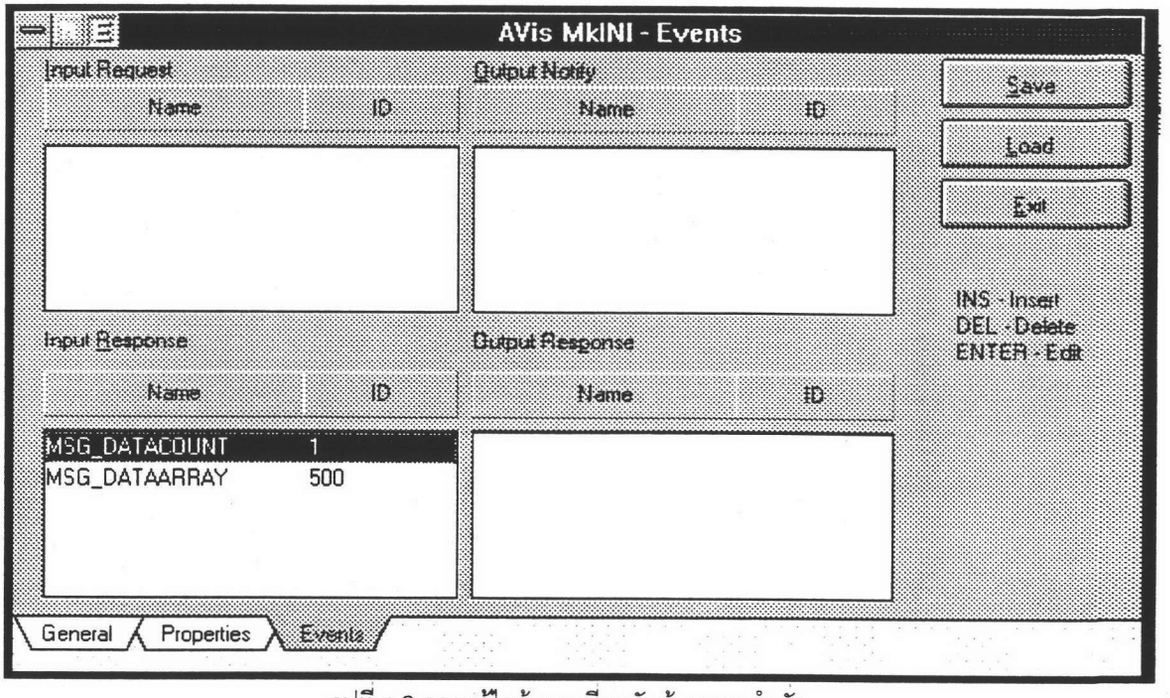
โปรแกรม AVisIniMaker สามารถใช้แก้ไขข้อมูลในแฟ้มองค์ประกอบได้ทุกหัวข้อ โดยผู้พัฒนาองค์ประกอบสามารถเลือกหัวข้อที่ต้องการแก้ไขได้จากแถบเลือกทางด้านล่าง โดยหัวข้อที่เกี่ยวกับข้อความคำสั่งทั้งหมดจะถูกรวมไว้ในหน้าจอ Events

หน้าจอในรูปที่ ง.1 จะเป็นหน้าจอที่ใช้ตั้งค่าหัวข้อ [Info] ส่วนหน้าจอในรูปที่ ง.2 จะเป็นหน้าจอที่ใช้ในการเพิ่มและแก้ไขพารามิเตอร์ต่างๆขององค์ประกอบ ซึ่งจะเห็นว่าจะทำให้การสร้างแฟ้มรายละเอียดด้วย AVisIniMaker จะทำได้ง่ายกว่าการสร้างและแก้ไขด้วยตัวเองมาก




รูปที่ ง.2 หน้าจอขณะแก้ไขค่าพารามิเตอร์

หากผู้พัฒนาองค์ประกอบต้องการแก้ไขเพิ่มเติมหรือลบข้อมูลเกี่ยวกับข้อความคำสั่งต่างๆสามารถทำได้โดยการใช้เมาส์กดที่แถบ "Events" ซึ่งจะทำให้เข้าสู่หน้าจอที่จะแสดงข้อมูลเกี่ยวกับข้อความคำสั่ง โดยผู้ใช้จะสามารถแก้ไขหรือเพิ่มเติมข้อความคำสั่งต่างๆได้จากหน้าจอในรูปที่ ง.3




รูปที่ ง.3 การแก้ไขข้อมูลเกี่ยวกับข้อความคำสั่ง

## ภาคผนวก จ.

## การพัฒนาองค์ประกอบการจินตทัศน์ด้วยวิชวลเบสิก

ในภาคผนวกนี้จะนำเสนอตัวอย่างการพัฒนาองค์ประกอบการจินตทัศน์ด้วยภาษาวิชวลเบสิก โดยแสดงตัวอย่างการพัฒนาองค์ประกอบการจินตทัศน์ของการจินตทัศน์อัลกอริทึมจัดเรียงข้อมูล แต่ในการอธิบายจะนำเสนอโปรแกรมต้นฉบับเพียงบางส่วนเท่านั้น ผู้ที่สนใจสามารถศึกษาโปรแกรมต้นฉบับอย่างละเอียดได้จากแฟ้มต่างๆในไดเร็คทอรี่ SORT ของ ACDK เนื่องจากเนื้อหาในภาคผนวกนี้จะกล่าวถึงเฉพาะการพัฒนาองค์ประกอบการจินตทัศน์ โดยจะข้ามการอธิบายเนื้อหาบางส่วนที่ไม่เกี่ยวข้องกับระบบ AVis ไป ดังนั้นผู้ที่จะศึกษาวิธีการพัฒนาองค์ประกอบการจินตทัศน์ที่กล่าวไว้ในภาคผนวกนี้ควรจะมีทำการศึกษาและทำความเข้าใจกับการพัฒนาโปรแกรมด้วยวิชวลเบสิกเป็นอย่างดีโดยสามารถศึกษาเพิ่มเติมได้จากคู่มือของวิชวลเบสิก[1]

### *1 สิ่งที่ต้องใช้ในการพัฒนาองค์ประกอบการจินตทัศน์*

เครื่องมือและโปรแกรมที่ผู้พัฒนาองค์ประกอบการจินตทัศน์ด้วยวิชวลเบสิกต้องใช้ในการพัฒนาได้แก่

1. แฟ้ม AVIS.BAS ซึ่งจะจัดเก็บอยู่ในไดเร็คทอรี่ AVIS\ACDK
2. แฟ้ม AVISCOMP.VBX ซึ่งจะจัดเก็บอยู่ในไดเร็คทอรี่ SYSTEM ของวินโดว์

### *2 AVISCOMP.VBX*

ในการพัฒนาองค์ประกอบการจินตทัศน์ด้วยวิชวลเบสิก นอกจากแฟ้ม AVIS.BAS ซึ่งจำเป็นต่อการพัฒนาองค์ประกอบเนื่องจากจะเก็บค่าคงที่และการประกาศ (Declare) ชื่อฟังก์ชันต่างๆซึ่งจัดเก็บไว้ใน AVISDLL.DLL และ AVISCOMP.VBX แล้ว ยังมีแฟ้มอีกหนึ่งแฟ้มที่มีความสำคัญมากเนื่องจากเป็นแฟ้มที่ทำให้องค์ประกอบการจินตทัศน์ที่พัฒนาด้วยวิชวลเบสิก สามารถรับข้อความคำสั่งต่างๆของ AVis ได้ แฟ้มนี้ก็คือแฟ้มที่มีชื่อว่า AVISCOMP.VBX ซึ่งเป็นตัวควบคุมเฉพาะของวิชวลเบสิก

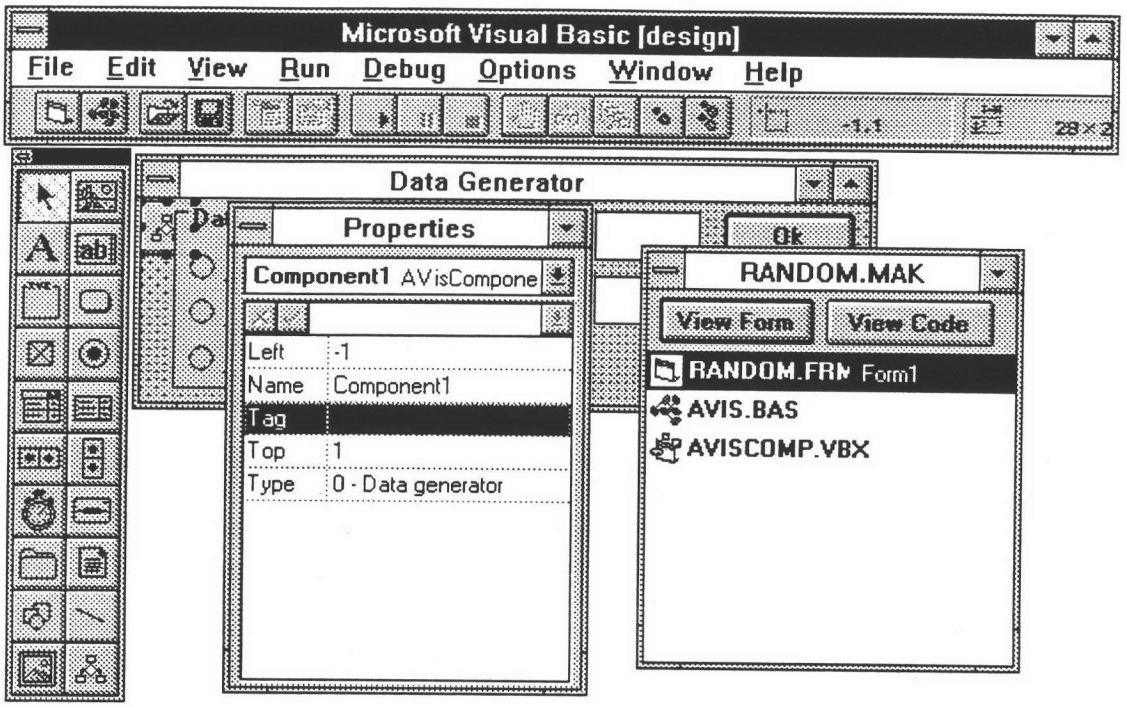
แฟ้มนี้นอกจากจะทำให้องค์ประกอบการจินตทัศน์ที่พัฒนาด้วยวิชวลเบสิก สามารถรับข้อความคำสั่งจาก AVis ได้แล้ว ยังทำให้ผู้พัฒนาองค์ประกอบพัฒนาองค์ประกอบ ได้ง่ายขึ้นเพราะ AVISCOMP.VBX จะช่วยซ่อนความซับซ้อนของกลไกการรับข้อความคำสั่งของ AVis ไว้ภายใน นอกจากนี้ AVISCOMP.VBX ยังมีบริการต่างๆเพิ่มขึ้นจากบริการที่อยู่ในหน่วยบริหารการจินตทัศน์เพื่อช่วยในการพัฒนาองค์ประกอบการจินตทัศน์ด้วย วิชวลเบสิก ทำได้ง่ายขึ้นโดยเฉพาะ รายละเอียดการทำงานของ AVISCOMP.VBX สามารถศึกษาได้จากภาคผนวก ช

การใช้งาน AVISCOMP.VBX คล้ายกับตัวควบคุมทั่วไปของวิชวลเบสิก นั่นคือหลังจากเพิ่มแฟ้มนี้เข้าไปในโปรเจ็คของวิชวลเบสิก และ นำตัวควบคุมไปวางบนฟอร์มที่ต้องการแล้ว ผู้พัฒนาองค์ประกอบการจินตทัศน์จะต้องตั้งค่าคุณสมบัติของตัวควบคุม และเขียนโปรแกรมเพื่อตอบสนองต่อเหตุการณ์ต่างๆที่ตัวควบคุมจะได้รับ

---

[1] Microsoft. Microsoft Visual Basic Programming System for Windows Programmer's Guide version 3.0. Redmond.WA. : Microsoft Inc. 1993.

ในรูปที่ จ.1 จะแสดงหน้าจอของวิชวลเบสิก ขณะพัฒนาองค์ประกอบตัวสร้างเลขสุ่มที่ใช้สำหรับการจินตทัศน์ของอัลกอริทึมจัดเรียงข้อมูล (ปุ่ม  คือตัวควบคุม AVISCOMP.VBX)




รูปที่ จ.1 หน้าจอของ วิชวลเบสิก ขณะพัฒนาองค์ประกอบการจินตทัศน์

### *3 ค่าคุณสมบัติของ AVISCOMP.VBX*

จากรูปที่ จ.1 ซึ่งแสดงหน้าจอตั้งค่าคุณสมบัติของตัวควบคุมด้วย จะพบว่าค่าคุณสมบัติของ AVISCOMP.VBX ซึ่งกำหนดได้ในขณะออกแบบที่แตกต่างออกไปจากตัวควบคุมแบบอื่นของวิชวลเบสิกมีเพียงตัวเดียว ได้แก่ค่าของคุณสมบัติที่ชื่อ "Type" ซึ่งจะเป็นตัวกำหนดประเภทขององค์ประกอบการจินตทัศน์มีให้เลือกได้ 5 ค่า ได้แก่ Data Generator, Algorithm, Converter , View และ Other ค่าสี่ค่าแรกก็คือประเภทขององค์ประกอบต่างๆตามที่เคยกล่าวถึงมาแล้ว ส่วนค่าสุดท้ายจะใช้ในกรณีที่ไม่สามารถกำหนดประเภทขององค์ประกอบการจินตทัศน์ตามสี่ประเภทแรกได้

นอกจากค่าคุณสมบัติที่กำหนดได้ในขณะออกแบบองค์ประกอบการจินตทัศน์แล้ว ยังมีค่าคุณสมบัติอีกกลุ่มหนึ่ง ซึ่งจะใช้งานได้เฉพาะขณะที่องค์ประกอบกำลังทำงาน ค่าเหล่านี้ได้แก่

| ชื่อ | รายละเอียด |
|---|---|
| EnumLink | ใช้ตรวจสอบว่ามีองค์ประกอบการจินตทัศน์ใดเชื่อมต่ออยู่กับองค์ประกอบการจินตทัศน์นี้บ้าง รายละเอียดของค่าคุณสมบัตินี้ศึกษาเพิ่มเติมได้จากแฟ้ม AVISCOMP.HLP ใน ไดเร็คทอรี่ ACDK |
| Error | รหัสค่าข้อผิดพลาดที่เกิดจากการทำงานของ AVISCOMP.VBX ซึ่งค่าคงที่ที่ใช้แทนข้อผิดพลาดเหล่านี้ จะเก็บอยู่ในแฟ้ม AVIS.BAS |
| ID | หมายเลขประจำองค์ประกอบ องค์ประกอบการจินตทัศน์จะต้องใช้ค่านี้อ้างอิง |

| | |
|---|---|
| | ถึงตัวเองเสมอ เมื่อเรียกใช้ฟังก์ชันของ AVis |
| lParam | ค่าพารามิเตอร์ที่ใช้ร่วมกับค่า EnumLink |
| SyncCount | จำนวนครั้งของการทำงานพื้นฐานขององค์ประกอบอัลกอริทึมในแต่ละรอบของการจินตทัศน์ โดยค่านี้จะมีค่าเป็นศูนย์หากองค์ประกอบการจินตทัศน์ไม่ใช่องค์ประกอบอัลกอริทึม |

ค่าทั้งห้านี้เป็นค่าที่อ่านได้อย่างเดียวซึ่งตัวอย่างในการนำค่าเหล่านี้ไปใช้งานจะแสดงให้เห็นในภายหลัง หากผู้ใดสนใจรายละเอียดของ AVISCOMP.VBX อย่างละเอียดสามารถศึกษาได้จากแฟ้ม AVISCOMP.HLP ซึ่งอยู่ในไดเร็คทอรี่ ACDK

### *4 เหตุการณ์ของ AVISCOMP.VBX*

ในการใช้งานตัวควบคุมของ วิชวลเบสิก นั้นนอกจากค่าคุณสมบัติแล้ว อีกสิ่งหนึ่งที่มีความสำคัญอย่างยิ่งในการนำตัวควบคุมไปใช้งานก็คือเหตุการณ์(Event)ของตัวควบคุม ซึ่งเหตุการณ์ของ AVISCOMP.VBX มีดังต่อไปนี้

| ชื่อ | เกิดขึ้นเมื่อ |
|---|---|
| BeginAlgorithm | องค์ประกอบการจินตทัศน์ได้รับข้อความคำสั่งควบคุมการทำงาน "เริ่มการทำงานของอัลกอริทึม" |
| BeginSession | องค์ประกอบการจินตทัศน์ได้รับข้อความคำสั่งควบคุมการทำงาน "เริ่มการจินตทัศน์" |
| ConfigComponent | องค์ประกอบการจินตทัศน์ได้รับข้อความคำสั่งตั้งค่าจากโปรแกรมควบคุมการจินตทัศน์ |
| EndSession | องค์ประกอบการจินตทัศน์ได้รับข้อความคำสั่งควบคุมการทำงาน "จบการจินตทัศน์" |
| EnumLink | มีการตั้งค่า EnumLink ศึกษาเพิ่มเติมได้จากแฟ้ม AVISCOMP.HLP |
| InputRequest | องค์ประกอบการจินตทัศน์ได้รับข้อความคำสั่งขอข้อมูลจากองค์ประกอบการจินตทัศน์อื่น |
| OutputNotify | องค์ประกอบการจินตทัศน์ได้รับข้อความคำสั่งแสดงผลจากองค์ประกอบการจินตทัศน์อื่น |

### *5 เหตุการณ์ Form_Load ของฟอร์ม*

เมื่อเกิดเหตุการณ์ Form_Load ซึ่งเป็นเหตุการแรกที่ฟอร์มจะได้รับ ผู้พัฒนาองค์ประกอบการจินตทัศน์ควรจะทำการตรวจสอบว่าองค์ประกอบสามารถลงทะเบียนเพื่อเข้าไปทำงานในระบบ AVis ได้หรือไม่ หากลงทะเบียนไม่ได้องค์ประกอบก็ควรจะแจ้งผู้ใช้ปลายทางถึงเหตุการณ์ที่เกิดขึ้น โดยการตรวจสอบสามารถทำได้โดยการตรวจดูว่าค่า ID ของตัวควบคุมมีค่าเท่ากับค่าคงที่ AVIS_ID_NOCOMPONENT หรือไม่ ทั้งนี้เพราะหากลงทะเบียนองค์ประกอบไม่ได้ AVISCOMP.VBX จะตั้งค่า ID ให้มีค่าเป็นค่านี้

```
Sub Form_Load ()

    If DGen.ID = AVIS_ID_NOCOMPONENT Then
        MsgBox "Can not register DGen because " & AVisVBGetComponentErrorStr(DGen),16
        End
    End If

    txtDataCount = 300
    txtUpperLimit = 300
    Randomize
    optDataType(0) = True
End Sub
```

รูปที่ จ.2 ตัวอย่างโปรแกรมของเหตุการณ์ Form_Load

ในรูปที่ จ.2 จะแสดงการตอบสนองต่อเหตุการณ์ Form_Load ของ ตัวสร้างข้อมูลแบบเลขสุ่ม ซึ่งนอกจากจะมีส่วนของโปรแกรมที่ทำหน้าที่ตั้งค่าคล้ายกับโปรแกรมของวิชวลเบสิกโดยทั่วไปแล้ว ก็จะมีส่วนตรวจสอบการลงทะเบียนขององค์ประกอบหากพบว่าลงทะเบียนไม่ได้ ก็จะมีการเรียกใช้ฟังก์ชัน AVisVBGetComponentErrorStr ในแฟ้ม AVISCOMP.VBX ประกอบกับคำสั่ง MsgBox เพื่อแสดงข้อผิดพลาดที่เกิดขึ้นให้กับผู้ใช้ทราบ

### *6 การส่งข้อความคำสั่งขององค์ประกอบ*

เมื่อองค์ประกอบการจินตทัศน์ต้องการส่งข้อความแสดงผล สามารถทำได้โดยการเรียกใช้ฟังก์ชันของ AVISCOMP.VBX ที่ชื่อ AVisVBInputRequest และ AVisVBOutputNotify การทำงานของฟังก์ชันทั้งสองจะเหมือนกันในแง่ที่ว่าฟังก์ชันทั้งคู่มีไว้เพื่อส่งข้อความคำสั่งไปยังองค์ประกอบอื่นๆ จะต่างกันก็ตรงที่ฟังก์ชัน AVisVBInputRequest นั้นจะส่งข้อความคำสั่งขอข้อมูลไปยังองค์ประกอบที่เป็นผู้สร้างหรือเก็บข้อมูล ในขณะที่ฟังก์ชัน AVisVBOutputNotify จะส่งข้อความคำสั่งแสดงผลไปยังองค์ประกอบที่คอยรับคำสั่งหรือข้อมูล ฟังก์ชันทั้งสองเมื่อเรียกใช้จะมีพารามิเตอร์อยู่ห้าตัวคือ ตัวควบคุมที่ใช้ติดต่อกับระบบ AVis หมายเลขของข้อความคำสั่งและค่าพารามิเตอร์ของข้อความคำสั่งอีกสามค่า

ในรูปที่ จ.3 แสดงถึงต้นแบบของฟังก์ชันทั้งสองในแฟ้ม AVIS.BAS ซึ่งนอกจากจะแสดงค่าพารามิเตอร์ต่างๆแล้ว ยังแสดงค่าผลการทำงานของฟังก์ชันทั้งสองโดย AVisVBOutputNotify จะไม่มีค่าผลการทำงาน ส่วน AVisVBInputRequest จะให้ค่าผลการทำงานหนึ่งค่าเป็นตัวเลขแบบ Long ซึ่งความหมายของค่านี้จะขึ้นอยู่กับตัวข้อความคำสั่งที่ส่งไป

```
Declare Function AVisVBInputRequest Lib "AVISCOMP.VBX"  (Ctrl As Control,
        ByVal wMsg%, ByVal lParam1&, ByVal lParam2&,ByVal LPSTR$) As Long

Declare Sub AVisVBOutputNotify Lib "AVISCOMP.VBX"  (Ctrl As Control,
        ByVal wMsg%, ByVal lParam1&,  ByVal lParam2&,ByVal LPSTR$)
```

รูปที่ จ.3 ต้นแบบของฟังก์ชันส่งข้อความใน AVIS.BAS

การส่งข้อมูลด้วยฟังก์ชันทั้งสองนี้หากเกิดข้อผิดพลาดใดๆขึ้นจะทำให้เกิด Runtime Error ของวิชวลเบสิกขึ้น ซึ่งผู้พัฒนาองค์ประกอบสามารถตรวจสอบและแก้ไขได้ด้วยคำสั่ง On Error Goto เช่นเดียวกับกับ Runtime Error ทั่วไปของวิชวลเบสิก จากรูปจะเห็นว่าในการเรียกใช้ฟังก์ชันทั้งสองผู้เรียกใช้ไม่ต้องระบุว่าจะให้องค์ประกอบการจินตทัศน์ใดเป็นผู้รับข้อมูลซึ่งทำให้ผู้พัฒนาองค์ประกอบไม่ต้องกังวลในเรื่องนี้เพราะ AVis จะ

จัดการเรื่องนี้ให้เอง แต่หากผู้พัฒนาองค์ประกอบต้องการส่งข้อความคำสั่งโดยระบุองค์ประกอบที่จะเป็นผู้รับข้อมูลก็สามารถทำได้โดยการเรียกใช้ฟังก์ชัน AVisVBInputRequestEx และ AVisVBOutputNotifyEx แทน ซึ่งรายละเอียดของฟังก์ชันทั้งสองสามารถศึกษาได้จากแฟ้ม AVISCOMP.HLP

ในรูปที่ จ.4 แสดงตัวอย่างของส่วนอ่านข้อมูลของอัลกอริทึมจัดเรียงข้อมูลแบบเร็ว ซึ่งจะอ่านข้อมูลจากตัวสร้างข้อมูลในรูปแบบแถวลำดับ (array) ของตัวแปรแบบ integer เมื่อได้รับข้อมูลแล้วก็จะส่งข้อมูลที่ได้ออกไปให้ตัวแปลงค่าที่ต่ออยู่กับอัลกอริทึมด้วย

```
Sub GetDgenData ()
Dim i%, l&
On Error Goto Err_GetDGenData

    DataCount = AVisVBInputRequest(QuickSort,MSG_DATACOUNT,0,0,"")
    If DataCount <= 0 Then
        MsgBox "Error in number of data " & DataCount
        Exit Sub
    End If

    Call AVisVBOutputNotify(QuickSort,MSG_DATACOUNT,ataCount,0,"")

    ReDim DgenData(DataCount)
    l = AVisVBInputRequest(QuickSort,MSG_DATAARRAY,0,0,"")
    i = AVisVBGetArrayParam(DgenData(1),l,AVIS_SIZE_INTEGER,DataCount)

    l = AVisVBCreateArrayParam(DgenData(1))
    Call AVisVBOutputNotify(QuickSort,MSG_DATAARRAY,l,
                            DataCount,Trim$(Str$(AVIS_SIZE_INTEGER)))
    Exit Sub
Err_GetDGenData:
    MsgBox Error$, 16, "Get Data Error"
    Exit Sub
End Sub
```

รูปที่ จ.4 แสดงตัวอย่างโปรแกรมของส่วนรับข้อมูลขององค์ประกอบการจินตทัศน์

การขอข้อมูลของอัลกอริทึมจะเริ่มจากการส่งข้อความคำสั่งขอข้อมูลไปยังตัวสร้างข้อมูลเพื่อขอทราบจำนวนข้อมูล เมื่อตรวจสอบว่าจำนวนข้อมูลอยู่ในช่วงที่ถูกต้องแล้ว ก็จะส่งค่าจำนวนข้อมูลออกไปให้ตัวแปลงค่า จากนั้นจะทำการจองแถวลำดับเพื่อจัดเก็บค่าข้อมูลแล้วจึงร้องขอค่าข้อมูลทั้งหมด เมื่อได้รับค่าในรูปของเลขแทน (handle) ของข้อมูลทั้งหมดแล้วก็จะใช้เลขแทนนี้กับคำสั่ง AVisVBGetArrayParam เพื่อคัดลอกข้อมูลนี้ลงในแถวลำดับที่จองขึ้นใหม่ (DGenData) จากนั้นก็จะสร้างเลขแทนจากแถวลำดับนี้ด้วยคำสั่ง AVisVBCreateArrayParam เพื่อส่งต่อไปให้ตัวแปลงค่าต่อไป

### *7 การรับข้อความคำสั่งขององค์ประกอบ*

การส่งข้อความคำสั่งระหว่างองค์ประกอบการจินตทัศน์ต่างๆจะทำให้เกิดเหตุการณ์ขึ้นที่ฝั่งผู้รับ โดยเหตุการณ์ที่เกิดขึ้นจะมีอยู่สองเหตุการณ์ขึ้นกับชนิดของข้อความคำสั่งที่ได้รับได้แก่เหตุการณ์ InputRequest และ OutputNotify สำหรับข้อความคำสั่งขอข้อมูลและข้อความคำสั่งแสดงผลตามลำดับ

ส่วนของส่วนโปรแกรมที่ตอบสนองเหตุการณ์ต่อข้อความคำสั่งทั้งสองประเภทนั้นเกือบจะเหมือนกัน ต่างกันเพียงในเหตุการณ์ InputRequest จะมีพารามิเตอร์เพิ่มอีกหนึ่งตัวคือ RetVal ซึ่งจะเป็นค่าที่ผู้ส่งข้อความคำสั่งจะได้รับจากคำสั่ง InputRequest ดังนั้นในโปรแกรมที่ตอบสนองต่อเหตุการณ์ InputRequest จะต้องทำ

การกำหนดค่าของพารามิเตอร์ตัวนี้ก่อนจบการทำงานเสมอ ส่วนค่าพารามิเตอร์ที่เหลือก็จะเป็นค่าที่เหมือนกับค่าที่องค์ประกอบที่เป็นผู้ส่งข้อความคำสั่งมากำหนดไว้

ในส่วนของโปรแกรมที่ผู้พัฒนาองค์ประกอบจะต้องเพิ่มเข้าไปนั้นจะเป็นคำสั่งใดของวิชวลเบสิกก็ได้ตามต้องการเพื่อทำงานตามคำสั่งที่ได้รับ ดังตัวอย่างในรูปที่ จ.5

จะเห็นว่าตัวโปรแกรมขององค์ประกอบแสดงผลแบบจุดจะประกอบไปด้วยคำสั่ง Select Case เพื่อใช้แยกหมายเลขของข้อความคำสั่งที่ได้รับ และคำสั่งของวิชวลเบสิก และ ของ AVis ต่างๆ ที่ใช้ในการจัดการกับข้อความคำสั่งแสดงผลแต่ละข้อความ

```
Sub DotView _InputRequest(SenderID As Integer, message As Integer,
        LParam1 As Long, LParam2 As Long,   StrParam As String,RetVal As Long)

    'Place code here
    RetVal = ...
End Sub

Sub DotView_OutputNotify(SenderID As Integer, message  As  Integer,
        LParam1 As Long, LParam2 As Long, StrParam As String)

    Select Case message
        Case MSG_DATAARRAY
            If Val(StrParam) <> AVIS_SIZE_INTEGER Then
                MsgBox "Invalid data type":Exit sub
            End If
            'Code for read data.
            'Code for draw all dot in an initial state
        Case MSG_SETVAL
            Call MoveDot(LParam1,LParam2,   pictView.ForeColor)
        Case MSG_FINISH
            'Code for update screen to show finished status
    End Select

End Sub
```

รูปที่ จ.5 ตัวอย่างโปรแกรมตอบสนองต่อข้อความคำสั่งแสดงผล

### *8 ข้อความคำสั่งควบคุมการทำงาน*

ในส่วนของโปรแกรมที่ใช้ตอบสนองต่อข้อความคำสั่งควบคุมการทำงานขององค์ประกอบการจินตทัศน์นั้น จะมีลักษณะดังในรูปที่ จ.6 และรูปที่ จ.7 โดยในรูปที่ จ.6 จะแสดงตัวอย่างของส่วนของโปรแกรมตอบสนองเหตุการณ์ BeginSession และเหตุการณ์ EndSession ขององค์ประกอบส่วนสร้างข้อมูลแบบเลขแบบสุ่ม โดยเหตุการณ์ทั้งสองนี้เป็นเหตุการณ์ที่องค์ประกอบการจินตทัศน์ทุกตัวในระบบควรจะใช้เป็นจุดตั้งค่าเริ่มต้นและจุดลบค่าของตัวแปรและทรัพยากรต่างๆที่จะใช้ในการจินตทัศน์ เพื่อที่องค์ประกอบจะได้สามารถทำงานได้อย่างถูกต้องหากผู้ใช้ทำการจินตทัศน์มากกว่าหนึ่งรอบ

องค์ประกอบตัวสร้างข้อมูลแบบสุ่มจะใช้โปรแกรมตอบสนองต่อเหตุการณ์ทั้งสองเป็นจุดสร้างข้อมูลและลบข้อมูลของตัวเลขสุ่มที่จะใช้ในการจินตทัศน์ตามลำดับ เลขสุ่มจะถูกสร้างขึ้นเมื่อมีการเริ่มการจินตทัศน์แต่ละรอบและจะถูกลบทิ้งหลังจากการจินตทัศน์รอบนั้นๆ ทำให้ทุกครั้งที่ผู้ใช้ปลายทางสั่งให้เริ่มการจินตทัศน์องค์ประกอบอัลกอริทึมจะได้ข้อมูลชุดใหม่ไปทำงานเสมอ และเมื่อการจินตทัศน์สิ้นสุดลงข้อมูลก็จะถูกลบทิ้งเพื่อจะได้ไม่สิ้นเปลืองหน่วยความจำที่จะใช้เก็บข้อมูล

```
Sub DGen_BeginSession()
Dim i%, j%, k%
Dim UpperLimit%

    'If user does not specified number of data
    'set number of data to 100.
    If Val(txtDataCount) = 0 Then txtDataCount = 100

    DataCount = Val(txtDataCount)
    UpperLimit = Val(txtUpperLimit)
    'No upper limit , set upper limit to number of data
    If UpperLimit = 0 Then UpperLimit = DataCount
     'Allocate space for new data
    ReDim GenData(DataCount)

    Randomize
    S = UpperLimit / DataCount
    'Create an uniform data first
    'Just an integer form 1 to UpperLimit
    For j = 1 To DataCount
        GenData(j) = Int(j * S) 'Data, no random here
    Next j
    'Now this is the time to random
    For j = 1 To 2 * DataCount
        k = Int(Rnd * DataCount + 1)
        i = Int(Rnd * DataCount + 1)
        Swap(GenData(i),GenData(j)) ' Shuffle data
    Next j
End Sub

Sub DGen_EndSession()
    'A time to free the memory
     Erase GenData
End Sub
```

รูปที่ จ.6 แสดงโปรแกรมตอบสนองเหตุการณ์ของข้อความคำสั่งควบคุมการทำงาน

### 9 *การทำงานขององค์ประกอบอัลกอริทึม*

รูปที่ จ.7 แสดงตัวอย่างของส่วนของโปรแกรมที่ตอบสนองต่อเหตุการณ์ BeginAlgorithm ในองค์ประกอบอัลกอริทึมของการจัดเรียงข้อมูลแบบฟองของเลขจำนวนเต็ม โดยถือว่าการเปรียบเทียบค่าของคำสั่ง For ก็เป็นสิ่งที่มีผลต่อประสิทธิภาพของอัลกอริทึมด้วย เมื่อได้รับเหตุการณ์ให้เริ่มการจินตทัศน์ ส่วนอัลกอริทึมจะทำการขอข้อมูลจากส่วนสร้างข้อมูล เมื่อตรวจสอบว่าข้อมูลที่ได้รับถูกต้องก็จะแจ้งหน่วยบริหารการจินตทัศน์ว่าจะเริ่มทำงานด้วยฟังก์ชัน AVisVBOpenSync แล้วจัดเก็บค่าเวลาเริ่มต้นของการทำงานเพื่อจะเปรียบเทียบกับค่าเมื่อทำงานเสร็จเรียบร้อยแล้ว จากนั้นจึงเริ่มการจัดเรียงข้อมูลซึ่งในอัลกอริทึมจัดเรียงข้อมูลการทำงานพื้นฐานที่กำหนดไว้ก็คือการเปรียบเทียบค่า ดังนั้นเมื่อใดที่มีการเปรียบเทียบค่าไม่ว่าจะเป็นการเปรียบเทียบค่าโดยตรงหรือการเปรียบเทียบค่าโดยอ้อมเช่นการตรวจสอบจำนวนรอบทำงานของ For องค์ประกอบอัลกอริทึมควรจะต้องแจ้งหน่วยบริหารการจินตทัศน์ด้วยฟังก์ชัน AVisVBSync เสมอ แต่ในการพัฒนาองค์ประกอบอัลกอริทึมจะพบว่ามักจะมีการทำงานพื้นฐานอยู่ติดๆกัน หากจะต้องทำการเรียกใช้ฟังก์ชัน AVisVBSync ทุกครั้ง ก็จะเป็นการไม่สะดวก

และทำให้โปรแกรมที่ได้ยาวขึ้นโดยไม่จำเป็น ดังนั้น AVis จึงได้เพิ่มพารามิเตอร์ของฟังก์ชัน AVisVBSync ขึ้นอีกหนึ่งตัวได้แก่จำนวนการทำงานพื้นฐานที่ทำงานผ่านไปแล้ว ซึ่งในกรณีที่มีการทำงานพื้นฐานอยู่ติดกันหลายอันผู้พัฒนาองค์ประกอบก็อาจเรียกใช้ฟังก์ชัน AVisVBSync เพียงครั้งเดียวโดยระบุจำนวนการทำงานพื้นฐานที่ทำไปแล้ว จากนั้นฟังก์ชันสุดท้ายที่ผู้พัฒนาองค์ประกอบอัลกอริทึมจะต้องเรียกใช้ก็คือฟังก์ชัน AVisVBCloseSync เพื่อแจ้งให้หน่วยบริหารการจินตทัศน์ทราบว่าอัลกอริทึมทำงานเสร็จสิ้นลงแล้ว

```
Sub BubbleSort_BeginAlgorithm ()

Dim i%, J%,IsSwap%, ST!
On Error GoTo Err_BeginAlg

 Call GetDgenData          'Get data from data generator
 If DataCount <= 0 Then MsgBox "Error in data generator":Exit Sub
 Call AVisVBOpenSync(BubbleSort)

 ST = Timer
 For i = 1 To DataCount - 1
    Call AVisVBSync(BubbleSort, 1)
    IsSwap = False
    For J = 1 To DataCount - i
        If DgenData(J) > DgenData(J + 1) Then
            IsSwap = True
            Swap(DGenData(J),DGenData(J+1))
            Call AVisVBOutputNotify(BubbleSort,MSG_SWAP,J,J + 1,"")
        End If
        DoEvents
            Call AVisVBSync(BubbleSort, 2)
    Next J
    If Not IsSwap Then Exit For
    Call AVisVBSync(BubbleSort, 1)
 Next I

 Call AVisVBOutputNotify(BubbleSort,MSG_FINISH, BubbleSort.SyncCount,0, Timer - ST)
 Call AVisVBCloseSync(BubbleSort)
 Exit Sub

Err_BeginAlg:
    MsgBox Error$, 16, "Bubble Sort-Error"
    Exit Sub
End Sub
```

รูปที่ จ.7 แสดงโปรแกรมตอบสนองเหตุการณ์ BeginAlgorithm

หากสังเกตโปรแกรมในรูปที่ จ.7 จะพบว่านอกจากจะมีฟังก์ชันของ AVis ที่ใช้ในการทำงานต่างๆ แล้วยังมีคำสั่งของ วิชวลเบสิก คำสั่งหนึ่งซึ่งแทรกอยู่ตรงกลางของตัวอัลกอริทึม คำสั่งนี้ก็คือคำสั่ง DoEvents ซึ่งจะแทรกเข้ามาเพื่อให้การสลับการทำงานขององค์ประกอบอัลกอริทึมและโปรแกรมอื่นๆในระบบวินโดว์กระทำได้อย่างไม่ติดขัด ทั้งนี้เพราะการสลับการทำงานของอัลกอริทึมที่กระทำโดยหน่วยบริหารการจินตทัศน์เมื่อได้รับการร้องขอผ่านฟังก์ชัน AVisVBSync ได้รับการออกแบบให้ทำงานโดยไม่คำนึงถึงภาษาที่ใช้ในการพัฒนาองค์ประกอบ ซึ่งเมื่อนำกลไกการสลับการทำงานนี้มาใช้กับองค์ประกอบที่พัฒนาด้วยวิชวลเบสิกพบว่ากลไกนี้ไม่สามารถทำงานได้ผลเต็มที่ ดังนั้นผู้พัฒนาองค์ประกอบอัลกอริทึมด้วยวิชวลเบสิกจึงควรแทรกคำสั่ง DoEvents ลงในส่วนของโปรแกรมของอัลกอริทึมเพื่อช่วยให้การสลับการทำงานทำได้ผลอย่างเต็มที่

### *10 ข้อความคำสั่งตั้งค่า*

นอกจากข้อความคำสั่งควบคุมและข้อความคำสั่งขององค์ประกอบที่ได้กล่าวถึงไปแล้วยังมีข้อความคำสั่งอีกชนิดหนึ่งซึ่งองค์ประกอบการจินตทัศน์จะได้รับในขณะทำงานก็คือ ข้อความคำสั่งตั้งค่าที่จะถูกส่งมาจากโปรแกรมควบคุมการจินตทัศน์เพื่อตั้งค่าพารามิเตอร์ต่างๆขององค์ประกอบ โดยค่าพารามิเตอร์เหล่านี้ผู้พัฒนาองค์ประกอบการจินตทัศน์จะเป็นผู้กำหนดขึ้นและแจ้งไว้ในแฟ้มรายละเอียดขององค์ประกอบ เมื่อใดก็ตามที่มีการส่งข้อความคำสั่งตั้งค่ามาให้องค์ประกอบหนึ่ง AVISCOMP.VBX จะแจ้งองค์ประกอบนั้นด้วยเหตุการณ์ ConfigComponent โดยเหตุการณ์นี้จะมีข้อมูลที่ถูกส่งมาด้วยสองค่า(ดูรูปที่ จ.8 ) ข้อมูลตัวแรกจะเป็นหมายเลขของพารามิเตอร์ที่โปรแกรมควบคุมการจินตทัศน์ต้องการตั้งค่า ซึ่งจะตรงกับหมายเลขที่กำหนดไว้ในแฟ้มรายละเอียดขององค์ประกอบ ส่วนข้อมูลอีกค่าจะเป็นค่าของพารามิเตอร์ซึ่งจะเก็บอยู่ในลักษณะของสายอักขระ

```
Sub DGen_ConfigComponent(ParamID As Integer,ParamValue As String)
Dim i%
Dim SS

   Select Case ParamID
      Case 0
         Me.Visible = True
      Case 1        'Type
         i = Val(ParamValue)
         If (i < 0) Or (i > 2) Then Exit Sub
         optDataType(i) = True
      Case 2        'Number of data
         i = Val(ParamValue)
         If i <= 0 Then Exit Sub
         txtDataCount = i
      Case 3        'Upper bound
         i = Val(ParamValue)
         txtUpperLimit = i
   End Select

End Sub
```

รูปที่ จ.8 ตัวอย่างการตอบสนองเหตุการณ์ ConfigComponent

ในส่วนของหมายเลขพารามิเตอร์นั้น AVis ได้กำหนดให้มีค่าพิเศษไว้หนึ่งค่าก็คือ พารามิเตอร์หมายเลข 0 ซึ่งเป็นค่าสงวนสำหรับโปรแกรมควบคุมการจินตทัศน์ที่จะส่งให้กับองค์ประกอบเมื่อต้องการให้องค์ประกอบแสดงหน้าจอตั้งค่าพารามิเตอร์ขององค์ประกอบ ตัวอย่างเช่น AVisController จะส่งข้อความคำสั่งตั้งค่าพารามิเตอร์หมายเลข 0 ไปให้องค์ประกอบเมื่อผู้ใช้กดปุ่มตั้งค่าพารามิเตอร์ขององค์ประกอบ ในรูปที่ จ.8 จะแสดงโปรแกรมที่ใช้ตอบสนองต่อเหตุการณ์ ConfigComponent ของตัวสร้างข้อมูล จะเห็นว่าเมื่อได้รับข้อความคำสั่งตั้งค่าของพารามิเตอร์หมายเลขศูนย์ก็จะแสดงหน้าจอตั้งค่าซึ่งเป็นหน้าจอหลักของโปรแกรมให้ปรากฏขึ้นเพื่อให้ผู้ใช้ตั้งค่า โดยหน้าจอนี้ตามปรกติจะถูกซ่อนไว้ไม่ให้ผู้ใช้เห็น

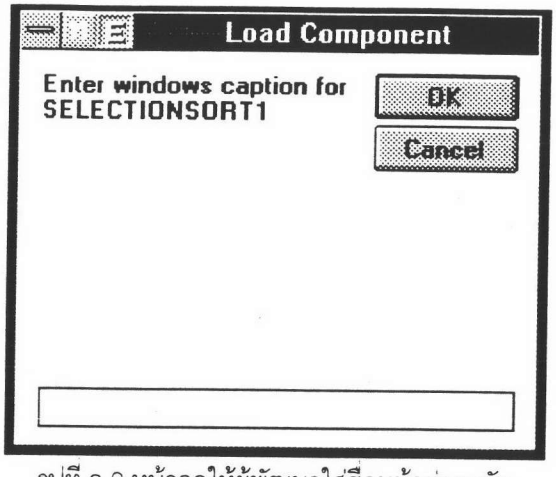
### *11 การตรวจแก้องค์ประกอบการจินตทัศน์*

ปัญหาอย่างหนึ่งซึ่งจะพบได้บ่อยในขณะทำการพัฒนาองค์ประกอบการจินตทัศน์ก็คือ องค์ประกอบที่สร้างขึ้นทำงานผิดพลาด ซึ่งผู้พัฒนาจำเป็นต้องตรวจแก้(debug)โปรแกรม แต่ขั้นตอนการตรวจแก้โปรแกรมทั่วไปจะไม่สามารถใช้กับองค์ประกอบการจินตทัศน์ได้ในทันที ทั้งนี้เพราะองค์ประกอบการจินตทัศน์เป็นโปรแกรมซึ่งจะต้องทำงานร่วมกับส่วนประกอบอื่นๆของ AVis เสมอ ดังนั้นในการตรวจแก้องค์ประกอบจะต้องได้รับความร่วม

มือจากระบบ AVis เพิ่มเติมจากขั้นตอนการตรวจแก้โปรแกรมบนวินโดว์ทั่วไปด้วย ขั้นตอนต่างๆที่ผู้พัฒนาองค์ประกอบต้องกระทำเมื่อต้องการตรวจแก้องค์ประกอบการจินตทัศน์ก็คือ

1.กำหนดค่าพารามิเตอร์ AutoLoad ขององค์ประกอบที่ต้องการตรวจแก้ให้มีค่าเป็น False ซึ่งจะทำได้โดยการใช้โปรแกรม AVisDesigner แก้ไขบทการจินตทัศน์

2.เริ่มการจินตทัศน์ด้วยวิธีการปกติ แต่เมื่อโปรแกรมควบคุมการจินตทัศน์ต้องการเรียกองค์ประกอบที่ต้องการตรวจแก้มาทำงานจะเกิดหน้าจอดังรูปที่ จ.9 เพื่อให้ผู้พัฒนาองค์ประกอบใส่ชื่อของหน้าต่างหลักของโปรแกรมองค์ประกอบ




รูปที่ จ.9 หน้าจอให้ผู้พัฒนาใส่ชื่อหน้าต่างหลัก

3.เมื่อเกิดหน้าจอนี้ขึ้นให้ผู้พัฒนาสั่งให้โปรแกรมช่วยตรวจแก้(debugger)เรียกการองค์ประกอบที่ต้องการตรวจแก้มาทำงาน ซึ่งก่อนหน้าที่จะเรียกองค์ประกอบมาทำงาน ผู้พัฒนาองค์ประกอบควรกำหนดจุดหยุด(break point)ลงไปในตำแหน่งที่ต้องการตรวจแก้

4.หลังจากที่สั่งให้องค์ประกอบเริ่มทำงานแล้ว ให้ผู้พัฒนากลับไปที่หน้าต่างในรูปที่ จ.9 แล้วใส่ชื่อหน้าต่างหลักของโปรแกรมองค์ประกอบ ซึ่งหากใส่ชื่อที่ถูกต้องระบบก็จะเข้าสู่ขั้นตอนการทำงานและการตรวจแก้ตามปกติต่อไป

## ภาคผนวก ฉ.

## การพัฒนาองค์ประกอบการจินตทัศน์ด้วยภาษา C

ในภาคผนวกนี้จะนำเสนอตัวอย่างการพัฒนาองค์ประกอบการจินตทัศน์ด้วยภาษาซี โดยแสดงตัวอย่างการพัฒนาองค์ประกอบการจินตทัศน์ของการจินตทัศน์อัลกอริทึมจัดเรียงข้อมูล แต่ในการอธิบายจะนำเสนอโปรแกรมต้นฉบับเพียงบางส่วนเท่านั้น ผู้ที่สนใจสามารถศึกษาโปรแกรมต้นฉบับอย่างละเอียดได้จากแฟ้มต่างๆในไดเร็คทอรี่ SORT ของ ACDK เนื่องจากเนื้อหาในภาคผนวกนี้จะกล่าวถึงเฉพาะการพัฒนาองค์ประกอบการจินตทัศน์ โดยจะข้ามการอธิบายเนื้อหาบางส่วนที่ไม่เกี่ยวข้องกับระบบ AVis ไป ดังนั้นผู้ที่จะศึกษาวิธีการพัฒนาองค์ประกอบการจินตทัศน์ที่กล่าวไว้ในภาคผนวกนี้ควรจะมีทำการศึกษาและทำความเข้าใจกับการพัฒนาโปรแกรมบนระบบวินโดว์ด้วยภาษาซีเป็นอย่างดีโดยสามารถศึกษาเพิ่มเติมได้จากหนังสือของ Charles Pezold[1]

### *1 สิ่งที่ต้องใช้ในการพัฒนาองค์ประกอบการจินตทัศน์*

เครื่องมือและโปรแกรมที่ผู้พัฒนาองค์ประกอบการจินตทัศน์ด้วยภาษาซีต้องใช้ในการพัฒนาได้แก่

1. แฟ้ม AVIS.H ซึ่งจะจัดเก็บอยู่ในไดเร็คทอรี่ AVIS\ACDK
2. แฟ้ม AVIS.LIB ซึ่งจะจัดเก็บอยู่ในไดเร็คทอรี่ AVIS\ACDKเช่นกัน

### *2 ข้อความคำสั่งที่องค์ประกอบการจินตทัศน์จะได้รับ*

จากที่ได้กล่าวมาแล้วในบทที่ 4 ว่าการติดต่อระหว่างส่วนประกอบต่างๆใน AVis จะติดต่อกันโดยใช้กลไกการรับส่งข้อความคำสั่งของระบบไมโครซอฟต์วินโดวส์ ดังนั้นองค์ประกอบการจินตทัศน์จะได้รับข้อความคำสั่งเฉพาะ(user defined message)ชุดหนึ่งซึ่งแตกต่างจากโปรแกรมบนวินโดว์อื่นๆ เพราะเป็นข้อความคำสั่งที่ AVis จะส่งให้องค์ประกอบเพื่อแทนข้อความคำสั่งต่างๆของ AVis ข้อความคำสั่งเฉพาะเหล่านี้ได้แก่

| ชื่อ | คำอธิบาย |
|---|---|
| AVIS_WM_BEGINALG | ข้อความคำสั่งควบคุมเริ่มการทำงานของอัลกอริทึม |
| AVIS_WM_BEGINSESSION | ข้อความคำสั่งควบคุมเริ่มการจินตทัศน์ |
| AVIS_WM_CONFIGCOMPONENT | ข้อความคำสั่งตั้งค่า |
| AVIS_WM_ENDSESSION | ข้อความคำสั่งควบคุมการจบการจินตทัศน์ |
| AVIS_WM_INPUTREQUEST | ข้อความคำสั่งขอข้อมูล |
| AVIS_WM_OUTPUTNOTIFY | ข้อความคำสั่งแสดงผล |

[1] Charles Petzold. Programming Windows 3.1. 3rd ed. Redmond.WA : Microsoft Press. 1992.

### *3 การลงทะเบียนองค์ประกอบ*

สิ่งแรกที่องค์ประกอบการจินตทัศน์จะต้องทำหลังจากถูกเรียกมาทำงานแล้วก็คือ ทำการลงทะเบียนองค์ประกอบด้วยฟังก์ชัน AVisRegisterComponent จุดที่เหมาะสมที่สุดในการลงทะเบียนองค์ประกอบก็คือ ทำการลงทะเบียนเมื่อองค์ประกอบได้รับข้อความคำสั่ง WM_CREATE จากระบบวินโดว์ดังในรูปที่ ฉ.1

```
long FAR PASCAL __export MainWndProc(HWND hWnd, UINT message,
                                      WPARAM wParam,LPARAM lParam)
{
    ... /* more code here */
    switch (message)
    {
        case WM_CREATE:
            /* register component to the system */
            if((g_ID=AVisRegisterComponent(hWnd,AVIS_CT_ALG,NULL)) <=
                AVIS_ID_NOCOMPONENT ) return -1L;
            break;
        case WM_DESTROY:
            if ( g_ComponentID != AVIS_ID_NOCOMPONENT )
                AVisUnregisterComponent(g_ID);
            PostQuitMessage(0);
            break;
        default:
            return DefWindowProc(hWnd,message,wParam,lParam);
    }
    return 0L;
}
```

รูปที่ ฉ.1 ตัวอย่างการลงและถอนทะเบียนองค์ประกอบ

ในการเรียกใช้ฟังก์ชัน AVisRegisterComponent ผู้ใช้ต้องส่งค่าพารามิเตอร์ไปให้ระบบสามค่า ค่าแรกจะเป็นหมายเลขประจำหน้าต่าง (Window Handle - HWND) ที่จะรับข้อความคำสั่งของ AVis ค่าที่สองจะเป็นค่าคงที่ซึ่งแทนชนิดขององค์ประกอบ และค่าสุดท้ายจะเป็นฟังก์ชันที่ใช้ส่งต่อข้อความคำสั่งของวินโดว์ที่องค์ประกอบได้รับในขณะที่องค์ประกอบหยุดรอการทำงาน ซึ่งหากไม่มีการกำหนดค่าพารามิเตอร์ค่านี้ ระบบจะใช้คำสั่งในรูปที่ ฉ.2 แทน และเมื่อเรียกใช้ฟังก์ชัน AVisRegisterComponent แล้ว AVis จะให้ค่าหมายเลขประจำองค์ประกอบกลับมา โดยหมายเลขนี้จะต้องใช้เมื่อต้องการติดต่อกับ AVis แต่หากการลงทะเบียนทำไม่สำเร็จ ฟังก์ชัน AVisRegisterComponent จะให้ค่า AVIS_ID_NOCOMPONENT กลับมา

```
BOOL far PASCAL _export CallWndProc(MSG FAR * msg)
{
    /* Default message processing */
    TranslateMessage(msg);
    DispatchMessage(msg);
    return TRUE;
}
```

รูปที่ ฉ.2 คำสั่งส่งต่อข้อความคำสั่งของวินโดว์แบบปกติขององค์ประกอบการจินตทัศน์

### *4 การถอนทะเบียน*

ก่อนที่องค์ประกอบจะยุติการทำงาน องค์ประกอบควรจะถอนทะเบียนออกจากระบบ เพื่อที่ AVis จะได้ทำการปลดปล่อยทรัพยากรต่างๆที่ใช้ในการเก็บข้อมูลต่างๆขององค์ประกอบ จุดที่องค์ประกอบควรใช้ในการถอนทะเบียนองค์ประกอบก็คือ เมื่อได้รับข้อความคำสั่ง WM_DESTROY จากระบบวินโดว์ ซึ่งการถอนทะเบียนจะทำได้โดยการเรียกใช้ฟังก์ชัน AVisUnregisterComponent ดังในรูปที่ ฉ.3

### 5 *Window Procedure ขององค์ประกอบการจินตทัศน์*

Window Procedure ขององค์ประกอบการจินตทัศน์จะแตกต่างจาก Window Procedure ของโปรแกรมที่พัฒนาด้วยภาษาซีทั่วไป เนื่องจาก AVis จะทำการส่งข้อความคำสั่งของวินโดว์ด้วยวิธีการที่แตกต่างจากการส่งข้อความคำสั่งของวินโดว์โดยทั่วไป ทั้งนี้เพื่อป้องกันปัญหาเรื่องการติดตาย(dead lock) ของระบบวินโดว์ โดย Window Procedure ขององค์ประกอบการจินตทัศน์จะมีลักษณะดังในรูป

```
long FAR PASCAL __export MainWndProc(HWND hWnd, UINT message,
                                     WPARAM wParam,LPARAM lParam)
{
    long ret;
    // Check for AVis Message
    if ( AVisDispatchEvent(hWnd,message,wParam,lParam, AVisEventProc,&ret) )
        return ret;
    switch  (message) {
        case WM_CREATE:
            . . .
            break;
        case WM_CLOSE:
        . . .
    }
}
```

รูปที่ ฉ.3 Windows Procedure ขององค์ประกอบการจินตทัศน์

ฟังก์ชัน AVisDispatchEvent เป็นฟังก์ชันของ AVis ซึ่งจะทำการตรวจสอบว่าข้อความคำสั่งที่ได้รับเป็นข้อความคำสั่งต่างๆของ AVis หรือไม่ หากใช่ก็จะทำการเรียกใช้ฟังก์ชันของ AVis เพื่อป้องกันการเกิดติดตายแล้วจึงทำการเรียกส่วนของโปรแกรมขององค์ประกอบที่แทนด้วยตัวชี้ AVisEventProc มาทำงานเพื่อจัดการกับข้อความคำสั่งของ AVis ที่องค์ประกอบได้รับต่อไป ในการทำงานของ AVisDispatchEvent จะให้ค่าผลการทำงานเป็น 1 หากข้อความคำสั่งที่ได้รับเป็นข้อความคำสั่งของ AVis และ ค่าผลการทำงานของข้อความคำสั่งนั้นจะถูกเก็บไว้ในพารามิเตอร์ตัวสุดท้ายของ AVisDispatchEvent แต่หากข้อความคำสั่งที่ได้รับเป็นข้อความคำสั่งของวินโดว์ AVisDispatchEvent จะให้ค่าผลการทำงานเป็น 0 องค์ประกอบการจินตทัศน์จะต้องจัดการกับข้อความคำสั่งนี้ในลักษณะเดียวกับโปรแกรมภาษาซีทั่วไป

ดังนั้นในการพัฒนาองค์ประกอบการจินตทัศน์นอกจากจะต้องแก้ไข Window Procedure ขององค์ประกอบแล้ว ผู้พัฒนาองค์ประกอบจะต้องทำการสร้าง AVis Event Procedure ที่จะใช้จัดการกับข้อความคำสั่งของ AVis ซึ่งจะมีลักษณะคล้ายกับ Window Procedure ของโปรแกรมบนวินโดว์ทั่วไป เพียงแต่จะได้รับเฉพาะข้อความคำสั่งของ AVis เท่านั้น

```
long FAR PASCAL __export AVisEventProc(HWND hWnd,UINT message,
                                       WPARAM wParam,LPARAM lParam)
{
    switch (message) {
        case AVIS_WM_BEGINALG:
            ...
            break;
        case AVIS_WM_INPUTREQUEST:
            ...
            break;
    }
    return 0;
}
```

รูปที่ ฉ.4 ตัวอย่าง AVis Event Procedure

ในรูปที่ ฉ.4 แสดงตัวอย่างของ AVis Event Procedure ขององค์ประกอบการจินตทัศน์ซึ่งสนใจข้อความคำสั่งของ AVis สองข้อความคำสั่งคือ AVIS_WM_BEGINALG และ AVIS_WM_INPUTREQUEST

### *6 การส่งข้อความคำสั่งขององค์ประกอบการจินตทัศน์*

เมื่อองค์ประกอบการจินตทัศน์ต้องการส่งข้อความแสดงผล สามารถกระทำได้โดยการเรียกใช้บริการของหน่วยบริหารการจินตทัศน์ผ่านฟังก์ชันที่มีชื่อว่า AVisInputRequest และ AVisOutputNotify การทำงานของฟังก์ชันทั้งสองจะเหมือนกันในแง่ที่ฟังก์ชันทั้งคู่มีไว้เพื่อส่งข้อความคำสั่งไปยังองค์ประกอบอื่นๆ จะต่างกันก็ตรงที่ฟังก์ชัน AVisInputRequest นั้นจะส่งข้อความคำสั่งขอข้อมูลไปยังองค์ประกอบที่เป็นผู้สร้างหรือเก็บข้อมูล ในขณะที่ฟังก์ชัน AVisOutputNotify จะส่งข้อความคำสั่งแสดงผลไปยังองค์ประกอบที่คอยรับคำสั่ง คำสั่งทั้งสองเมื่อเรียกใช้จะมีพารามิเตอร์อยู่ห้าตัวคือ หมายเลขประจำองค์ประกอบ หมายเลขของข้อความคำสั่ง และ ค่าพารามิเตอร์ของข้อความคำสั่งอีกสามค่า

ในรูปที่ ฉ.5 แสดงถึงต้นแบบของคำสั่งทั้งสองในแฟ้ม AVIS.H จะเห็นว่า ฟังก์ชัน AVisInputRequest จะมีค่าพารามิเตอร์มากกว่าหนึ่งค่าเป็นตัวชี้ซึ่งจะชี้ไปยังตัวเลขแบบ Long ที่ใช้เก็บค่าข้อมูลที่เป็นผลการส่งข้อความคำสั่งขอข้อมูล โดยความหมายของค่านี้จะขึ้นอยู่กับตัวข้อความคำสั่งที่ส่งไป

```
int AVisInputRequest(int nID,UINT wMsg,LPARAM lParam1,
                     LPARAM Param2,LPSTR lpStr,long far *lpRetVal);

int AVisOutputNotify(int nID,UINT wMsg,LPARAM lParam1,
                       LPARAM lParam2,LPSTR lpStr);
```

รูปที่ ฉ.5 ต้นแบบของคำสั่งส่งข้อความใน AVIS.H

ในการส่งข้อความคำสั่งด้วยคำสั่งทั้งสอง หากเกิดข้อผิดพลาดใดๆขึ้นคำสั่งทั้งสองจะให้ค่าผลการทำงานเป็นค่าอื่นที่ไม่ใช่ค่า AVIS_ERR_NOERROR ซึ่งผู้พัฒนาองค์ประกอบการจินตทัศน์จะต้องตรวจสอบค่าและแก้ไขข้อผิดพลาดเอง ซึ่งรายละเอียดและความหมายของค่าผลการทำงานต่างๆที่อาจเกิดขึ้นได้นี้สามารถศึกษาเพิ่มเติมได้จากแฟ้ม AVISCOMP.HLP

```
void GetDgenData ()
{
    long l;
    g_DataCount = 0;
    if (AVisInputRequest(g_ComponentID, MSG_DATACOUNT,0,0,NULL,&l) !=
            AVIS_ERR_OK ) return;
    if ((g_DataCount = (int)l) <= 0 ) return;

    if ((g_Data=(int *)malloc(sizeof(int)*(g_DataCount))) == NULL)
    { g_DataCount = 0;  return; }

    if ( AVisInputRequest(g_ComponentID, MSG_DATAARRAY,0,0, NULL,&l) !=
            AVIS_ERR_OK ) return;
    memcpy(g_Data,(void *)l,sizeof(int)*(g_DataCount));

    AVisOutputNotify(g_ComponentID, MSG_DATACOUNT,g_DataCount,0,"");
    AVisOutputNotify(g_ComponentID,MSG_DATAARRAY,l,g_DataCount, "2");
}
```

รูปที่ ฉ.6 ตัวอย่างโปรแกรมของส่วนอ่านข้อมูลของอัลกอริทึมจัดเรียงข้อมูล

จากรูปจะเห็นว่าในการเรียกใช้ฟังก์ชันทั้งสองผู้เรียกใช้ไม่ต้องระบุว่าจะให้องค์ประกอบการจินตทัศน์ใดเป็นผู้รับข้อมูลซึ่งทำให้ผู้พัฒนาองค์ประกอบไม่ต้องกังวลในเรื่องนี้เพราะ AVis จะจัดการเรื่องนี้ให้เอง แต่หากผู้

พัฒนาองค์ประกอบต้องการส่งข้อความคำสั่งโดยระบุองค์ประกอบการจินตทัศน์ที่จะเป็นผู้รับข้อมูลก็สามารถทำได้โดยการเรียกใช้ฟังก์ชัน AVisInputRequestEx และ ฟังก์ชัน AVisOutputNotifyEx แทน ซึ่งรายละเอียดของฟังก์ชันทั้งสองสามารถศึกษาได้จากแฟ้ม AVISCOMP.HLP ในรูปที่ ฉ.6 แสดงตัวอย่างของส่วนอ่านข้อมูลของอัลกอริทึมจัดเรียงข้อมูล ซึ่งจะข้อมูลจากตัวสร้างข้อมูลในรูปแบบของแถวลำดับของตัวแปรแบบ integer เมื่อได้รับข้อมูลแล้วก็จะส่งข้อมูลที่ได้ออกไปให้ตัวแปลงค่าที่ต่ออยู่กับอัลกอริทึมด้วย

การขอข้อมูลของอัลกอริทึมจะเริ่มจากการส่งข้อความคำสั่งไปยังตัวสร้างข้อมูลเพื่อขอทราบจำนวนข้อมูล เมื่อตรวจสอบว่าจำนวนข้อมูลอยู่ในช่วงที่ถูกต้องแล้ว ก็จะทำการจองข้อมูลแบบแถวลำดับ (array) เพื่อจัดเก็บค่าข้อมูล แล้วจึงร้องขอค่าข้อมูลทั้งหมด เมื่อได้รับค่าในรูปของ ตัวชี้ที่ชี้ไปยังของข้อมูลทั้งหมดแล้ว ก็จะทำการใช้คัดลอกข้อมูลเหล่านี้ใส่ลงในข้อมูลบบแถวลำดับที่จองขึ้นใหม่(g_Data) จากนั้นก็จะส่งข้อมูลที่ได้ไปให้ตัวแปลงค่าต่อไป

### *7 การรับข้อความคำสั่งขององค์ประกอบ*

เมื่อมีการส่งข้อความคำสั่งขององค์ประกอบไปยังองค์ประกอบการจินตทัศน์ AVis จะทำงานร่วมกับระบบวินโดว์เพื่อแจ้งให้ผู้รับทราบด้วยข้อความคำสั่งของวินโดว์ ข้อความคำสั่งของวินโดว์ที่เกิดขึ้นจะมีสองข้อความคำสั่งขึ้นกับชนิดของข้อความคำสั่งขององค์ประกอบที่องค์ประกอบได้รับ ข้อความคำสั่งของวินโดว์ทั้งสองข้อความคำสั่งได้แก่ AVIS_WM_INPUTREQUEST และ AVIS_WM_OUTPUTNOTIFY สำหรับข้อความคำสั่งขอข้อมูลและข้อความคำสั่งแสดงผลของ AVis ตามลำดับ

```
/* Algorithm visualization message structure */
typedef struct {
        int nSenderID;     /* Sender */
        UINT wMsg;         /* message */
        LPARAM lParam1;      /* message's parameters */
        LPATAM lParam2;
        LPSTR  lpstrParam;
} AVIS_MSG;
```

รูปที่ ฉ.7 โครงสร้างของ AVIS_MSG

ในการรับข้อความคำสั่งของวินโดว์ทั้งสองค่าพารามิเตอร์ lParam ของข้อความคำสั่งทั้งสอง จะเป็นตัวชี้ที่ชี้ไปยังข้อมูลแบบ AVIS_MSG ที่จะใช้เก็บข้อความคำสั่งขององค์ประกอบที่ถูกส่งมา โครงสร้างของ AVIS_MSG จะมีลักษณะดังในรูปที่ ฉ.7

ส่วนของโปรแกรมตอบสนองต่อข้อความคำสั่งทั้งสองประเภทเกือบจะเหมือนกัน จะต่างกันเพียงที่ในข้อความคำสั่ง AVIS_WM_INPUTREQUEST นั้น ผลการทำงานของ AVis Event Procedure จะเป็นค่าที่ผู้ส่งข้อความคำสั่งจะได้รับจากฟังก์ชัน AVisInputRequest ในรูปที่ ฉ.8 ซึ่งแสดงตัวอย่างโปรแกรมที่จัดการกับข้อความคำสั่งทั้งสอง

```
long FAR PASCAL export AVisEventProc(HWND hWnd,UINT message,
                                     WPARAM wParam,LPARAM lParam)
{
    AVIS_MSG far * lpMsg = (AVIS_MSG far *)lParam;
    switch (message) {
        case AVIS_WM_INPUTREQUEST:
            switch ( lpMsg->wMsg ) {
                case MSG_DATAARRAY:
                    /* data created in AVIS_WM_BEGINSESSION */
                    return (long)lpData;
                case MSG_DATACOUNT;
                    return nDataCount;
            }
            break;
        case AVIS_WM_OUTPUTNOTIFY:
            if ((lpMsg=(AVIS_MSG far *)lParam) == NULL) break;
            switch ( lpMsg->wMsg ) {
    }}
    return 0;
}
```

รูปที่ ฉ.8 ตัวอย่างโปรแกรมตอบสนองต่อข้อความคำสั่งขององค์ประกอบ

### *8 ข้อความคำสั่งควบคุมการทำงาน*

ในส่วนของข้อความคำสั่งควบคุมการทำงานของ AVis นั้นจะถูกแทนด้วยข้อความคำสั่งของวินโดว์ สามข้อความคือ AVIS_WM_BEGINSESSION AVIS_WM_BEGINALG และ AVIS_WM_ENDSESSION ข้อความคำสั่งทั้งสามเป็นข้อความคำสั่งที่ไม่มีข้อมูลเพิ่มเติมใดๆ ดังนั้นค่า lParam และ wParam ของข้อความคำสั่งทั้งสามจึงจะเป็นศูนย์เสมอ

ผู้พัฒนาองค์ประกอบการจินตทัศน์ควรใช้ข้อความคำสั่ง AVIS_WM_BEGINSESSION และ AVIS_WM_ENDSESSION เป็นจุดตั้งค่าเริ่มต้นและจุดลบค่าของตัวแปรและทรัพยากรต่างๆที่จะใช้ในการจินตทัศน์ตามลำดับ เพื่อที่องค์ประกอบจะได้สามารถทำงานได้อย่างถูกต้อง หากผู้ใช้ทำการจินตทัศน์มากกว่าหนึ่งรอบ ทั้งนี้หากนำการตั้งค่าและลบค่าตัวแปรและทรัพยากรต่างๆ ที่ใช้ในการจินตทัศน์ไปใส่ไว้เฉพาะที่จุดเริ่มทำงานและสิ้นสุดการทำงานของโปรแกรม อาจทำให้การจินตทัศน์อัลกอริทึมในรอบที่ไม่ใช่การทำการจินตทัศน์รอบแรกทำงานผิดพลาดได้

### *9 การทำงานขององค์ประกอบอัลกอริทึม*

รูปที่ ฉ.9 จะแสดงตัวอย่างของส่วนของโปรแกรมที่ตอบสนองต่อข้อความคำสั่ง AVIS_WM_BEGINALG ในองค์ประกอบอัลกอริทึมของการจัดเรียงข้อมูลแบบเลือกสำหรับข้อมูลจำนวนเต็มแบบ integer โดยถือว่าการเปรียบเทียบค่าใน loop ก็มีผลต่อการทำงานของอัลกอริทึมด้วย

จากรูปเมื่อได้รับเหตุการณ์ให้เริ่มการจินตทัศน์ ส่วนอัลกอริทึมจะทำการขอข้อมูลจากส่วนสร้างข้อมูล เมื่อตรวจสอบว่าข้อมูลที่ได้รับถูกต้องก็จะแจ้งหน่วยบริหารการจินตทัศน์ว่าจะเริ่มทำงานด้วยฟังก์ชัน AVisOpenSync แล้วจัดเก็บค่าเวลาเริ่มต้นของการทำงานเพื่อจะเปรียบเทียบกับค่าเมื่อทำงานเสร็จเรียบร้อยแล้ว จากนั้นจึงเริ่มการจัดเรียงข้อมูลซึ่งในการอัลกอริทึมจัดเรียงข้อมูลการทำงานพื้นฐานที่กำหนดไว้ก็คือการเปรียบเทียบค่า ดังนั้นเมื่อใดที่มีการเปรียบเทียบค่า ไม่ว่าจะเป็นการเปรียบเทียบค่าโดยตรง หรือการเปรียบเทียบค่าโดยอ้อมเช่นการตรวจสอบจำนวนรอบทำงานของ For องค์ประกอบอัลกอริทึมควรจะต้องแจ้งหน่วยบริหารการจินต

ทัศน์ด้วยฟังก์ชัน AVisSync เสมอ แต่ในการพัฒนาองค์ประกอบอัลกอริทึมจะพบว่ามักจะมีการทำงานพื้นฐานอยู่ติดๆกันหากจะต้องทำการเรียกใช้ฟังก์ชัน AVisSync ทุกครั้ง ก็จะเป็นการไม่สะดวกและทำให้โปรแกรมที่ได้ยาวขึ้นโดยไม่จำเป็น ดังนั้น AVis จึงได้เพิ่มพารามิเตอร์ของฟังก์ชัน AVisSync ขึ้นอีกหนึ่งตัวได้แก่จำนวนการทำงานพื้นฐานที่ทำงานผ่านไปแล้ว ซึ่งในกรณีที่มีการทำงานพื้นฐานอยู่ติดกันหลายอันผู้พัฒนาองค์ประกอบก็อาจเรียกใช้ฟังก์ชัน AVisSync เพียงครั้งเดียวโดยระบุจำนวนการทำงานพื้นฐานที่ทำไปแล้ว จากนั้นคำสั่งสุดท้ายที่ผู้พัฒนาองค์ประกอบอัลกอริทึมจะต้องเรียกใช้ก็คือฟังก์ชัน AVisCloseSync เพื่อแจ้งให้หน่วยบริหารการจินตทัศน์ทราบว่าอัลกอริทึมทำงานเสร็จสิ้นลงแล้ว

จากโปรแกรมจะสังเกตว่าทุกครั้งที่มีการเรียกใช้ฟังก์ชันของ AVis ผู้เรียกจะต้องคอยตรวจสอบผลการทำงานเสมอเพราะเมื่อใดก็ตามที่ฟังก์ชันของ AVis ให้ค่าผลการทำงานว่าทำงานผิดพลาดอัลกอริทึมก็จะหยุดทำงานทันที แล้วกลับไปสู่สถานะรอให้มีการเริ่มการจินตทัศน์ขึ้นใหม่

```
long FAR PASCAL export AVisEventProc(HWND hWnd, UINT message,
                                      WPARAM wParam,LPARAM lParam)
{
 switch (message) {
     case AVIS_WM_BEGINALG:
         if ( !GetDgenData() ) return;
         if ( AVisOpenSync(g_ComponentID) != AVIS_ERR_OK ) {
             MessageBox(hWnd,"Can not start alg","SelectC",MB_ICONSTOP);
             return 0;
         }
         if ( SelectionSort() )   {
             AVisOutputNotify(g_ComponentID,MSG_FINISH,
                              AVisGetSyncCount(g_ComponentID), 0, NULL);
         }
         AVisCloseSync(g_ComponentID);
     }
}

int SelectionSort()
{
 int i, j, maxP;
 for(i = g_DataCount-1;i>=1;--i) {
   if ( AVisSync(g_ComponentID, 2) != AVIS_TRUE ) return FALSE;
   maxP = 0;
   for(j = 0;j<=i;j++) {
         if ( g_Data[j] > g_Data[maxP] ) maxP = j;
         if ( AVisSync(g_ComponentID, 2)!=AVIS_TRUE) return FALSE;
 }
   Swap(&g_Data[i],&g_Data[maxP]);
   if (AVisOutputNotify(g_ComponentID,MSG_SWAP,i+1,maxP+1,NULL) != AVIS_ERR_OK)
         break;
 }
 return i == 0;
}
```

รูปที่ ฉ.9 แสดงโปรแกรมตอบสนองข้อความคำสั่ง AVIS_WM_BEGINALG

### *10 ข้อความคำสั่งตั้งค่า*

นอกจากข้อความคำสั่งควบคุมและข้อความคำสั่งขององค์ประกอบที่ได้กล่าวถึงไปแล้วยังมีข้อความคำสั่งอีกชนิดหนึ่งซึ่งองค์ประกอบการจินตทัศน์จะได้รับในขณะทำงานก็คือ ข้อความคำสั่งตั้งค่าที่จะถูกส่งมาจาก

โปรแกรมควบคุมการจินตทัศน์เพื่อทำการตั้งค่าพารามิเตอร์ต่างๆขององค์ประกอบ โดยค่าพารามิเตอร์เหล่านี้ผู้พัฒนาองค์ประกอบการจินตทัศน์จะเป็นผู้กำหนดขึ้นเองและแจ้งไว้ในแฟ้มรายละเอียดขององค์ประกอบ

ข้อความคำสั่งตั้งค่าจะถูกส่งให้องค์ประกอบในรูปของข้อความคำสั่งของวินโดว์ที่ชื่อ AVIS_WM_CONFIGCOMPONENT ซึ่งถูกกำหนดไว้เป็นค่าคงที่ในแฟ้ม AVIS.H พร้อมกับข้อมูลเพิ่มเติมสองค่า ค่าข้อมูลตัวแรกในwParamจะเป็นหมายเลขของพารามิเตอร์ที่โปรแกรมควบคุมการจินตทัศน์ต้องการตั้งค่า ซึ่งจะตรงกับหมายเลขที่กำหนดไว้ในแฟ้มรายละเอียดขององค์ประกอบ ส่วนค่าข้อมูลที่ถูกส่งมีอีกค่าซึ่งอยู่ในพารามิเตอร์ lParam จะเป็นค่าของพารามิเตอร์ซึ่งจะเก็บอยู่ในลักษณะของสายอักขระ

ในส่วนของหมายเลขพารามิเตอร์ AVis ได้กำหนดให้มีค่าพิเศษไว้หนึ่งค่าก็คือ พารามิเตอร์หมายเลข 0 ซึ่งเป็นค่าสงวนสำหรับโปรแกรมควบคุมการจินตทัศน์ที่จะส่งให้กับองค์ประกอบเมื่อต้องการให้องค์ประกอบแสดงหน้าจอตั้งค่าพารามิเตอร์ขององค์ประกอบ ตัวอย่างเช่น AVisController ข้อความคำสั่งตั้งค่าพารามิเตอร์หมายเลข 0 ไปให้องค์ประกอบเมื่อผู้ใช้กดปุ่มตั้งค่าพารามิเตอร์ขององค์ประกอบ ในรูปที่ ฉ.10 จะแสดงโปรแกรมที่ใช้ตอบสนองต่อข้อความคำสั่ง AVIS_WM_CONFIGCOMPONENT ของตัวสร้างข้อมูล จะเห็นว่าเมื่อได้รับข้อความคำสั่งตั้งค่าของพารามิเตอร์หมายเลขศูนย์ก็จะแสดงหน้าจอตั้งค่าซึ่งเป็นหน้าจอหลักของโปรแกรมให้ราบฏขึ้นเพื่อให้ผู้ใช้ตั้งค่า

```
long FAR PASCAL __export AVisEventProc(HWND hWnd,UINT msg,
                                        WPARAM wParam,LPARAM lParam)
{
    AVIS_MSG far * lpMsg;
    switch (msg) {
        case AVIS_WM_CONFIGCOMPONENT:
            swicth ( wParam ) {
                case 0:
                    ShowWindow(hWnd,SW_SHOW);
                    break;
                case 1:
                    g_DataCount = atoi((LPSTR)lParam);
                    break;
            }
            break;
        case AVIS_WM_BEGINSESSION:
            break;
        . . .
    }
}
```

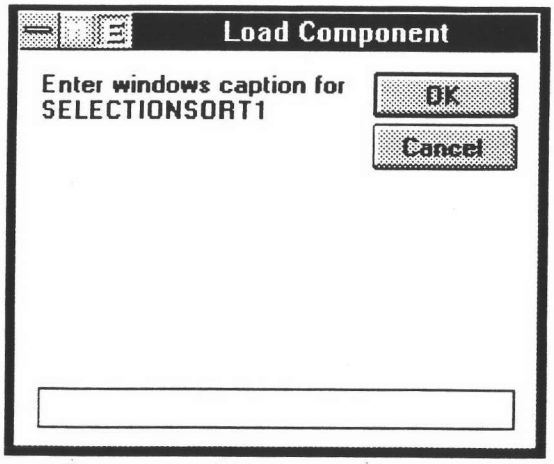
รูปที่ ฉ.10 ตัวอย่างการตอบสนองข้อความคำสั่ง AVIS_WM_CONFIGCOMPONENT

### *11 การตรวจแก้องค์ประกอบการจินตทัศน์*

ปัญหาอย่างหนึ่งซึ่งจะพบได้บ่อยในขณะทำการพัฒนาองค์ประกอบการจินตทัศน์ก็คือ องค์ประกอบที่สร้างขึ้นทำงานผิดพลาด ซึ่งผู้พัฒนาจำเป็นต้องตรวจแก้(debug)โปรแกรม แต่ขั้นตอนการตรวจแก้โปรแกรมทั่วไปจะไม่สามารถใช้กับองค์ประกอบการจินตทัศน์ได้ในทันที ทั้งนี้เพราะองค์ประกอบการจินตทัศน์เป็นโปรแกรมซึ่งจะต้องทำงานร่วมกับส่วนประกอบอื่นๆของ AVis เสมอ ดังนั้นในการตรวจแก้องค์ประกอบจะต้องได้รับความร่วมมือจากระบบ AVis เพิ่มเติมจากขั้นตอนการตรวจแก้โปรแกรมบนวินโดว์ทั่วไปด้วย ขั้นตอนต่างๆที่ผู้พัฒนาองค์ประกอบต้องกระทำเมื่อต้องการตรวจแก้องค์ประกอบการจินตทัศน์ก็คือ

1.กำหนดค่าพารามิเตอร์ AutoLoad ขององค์ประกอบที่ต้องการตรวจแก้ให้มีค่าเป็น False ซึ่งจะทำได้โดยการใช้โปรแกรม AVisDesigner แก้ไขบทการจินตทัศน์

2.เริ่มการจินตทัศน์ด้วยวิธีการปกติ แต่เมื่อโปรแกรมควบคุมการจินตทัศน์ต้องการเรียกองค์ประกอบที่ต้องการตรวจแก้มาทำงานจะเกิดหน้าจอดังรูปที่ ฉ.11 เพื่อให้ผู้พัฒนาองค์ประกอบใส่ชื่อของหน้าต่างหลักของโปรแกรมองค์ประกอบ




รูปที่ ฉ.11 หน้าจอให้ผู้พัฒนาใส่ชื่อหน้าต่างหลัก

3.เมื่อเกิดหน้าจอนี้ขึ้นให้ผู้พัฒนาสั่งให้โปรแกรมช่วยตรวจแก้(debugger)เรียกการองค์ประกอบที่ต้องการตรวจแก้มาทำงาน ซึ่งก่อนหน้าที่จะเรียกองค์ประกอบมาทำงาน ผู้พัฒนาองค์ประกอบควรกำหนดจุดหยุด(break point)ลงไปในตำแหน่งที่ต้องการตรวจแก้

4.หลังจากที่สั่งให้องค์ประกอบเริ่มทำงานแล้ว ให้ผู้พัฒนากลับไปที่หน้าต่างในรูปที่ ฉ.11 แล้วใส่ชื่อหน้าต่างหลักของโปรแกรมองค์ประกอบ ซึ่งหากใส่ชื่อที่ถูกต้องระบบก็จะเข้าสู่ขั้นตอนการทำงานและการตรวจแก้ตามปกติต่อไป

## ภาคผนวก ช.

## AVISCOMP.VBX

AVISCOMP.VBX เป็นส่วนประกอบย่อยของระบบ AVis ที่อยู่ภายนอกหน่วยบริหารการจินตทัศน์ โดยทำงานอยู่ในบริบท (context) ขององค์ประกอบการจินตทัศน์ เป็นส่วนเพิ่มขยายของหน่วยบริหารการจินตทัศน์ที่ทำให้ผู้พัฒนาองค์ประกอบการจินตทัศน์ด้วยวิชวลเบสิกสามารถพัฒนาองค์ประกอบได้สะดวกและรวดเร็วขึ้น AVISCOMP.VBX พัฒนาขึ้นในรูปแบบของตัวควบคุมเฉพาะของวิชวลเบสิก (Visual Basic Custom Control) โดยจะมีหน้าที่หลักอยู่สองประการคือ (1) การรับข้อความคำสั่งของ AVis และ (2) ทำให้การขอใช้บริการของ AVis จากองค์ประกอบที่พัฒนาด้วยวิชวลเบสิกทำได้ง่ายขึ้น

อย่างไรก็ตามเนื้อหาที่กล่าวถึงในบทนี้จะไม่กล่าวถึงขั้นตอนการพัฒนาตัวควบคุมเฉพาะของวิชวลเบสิกทั้งหมด แต่จะกล่าวถึงเฉพาะขั้นตอนที่พัฒนาขึ้นเพื่อใช้กับ AVISCOMP.VBX เท่านั้น สำหรับผู้ที่สนใจศึกษารายละเอียดเกี่ยวกับการพัฒนาตัวควบคุมเฉพาะของวิชวลเบสิกสามารถศึกษาเพิ่มเติมได้จากคู่มือของวิชวลเบสิก[1]

### *1 การรับข้อความคำสั่ง*

AVISCOMP.VBX จะแปลงข้อความคำสั่งต่างๆของ AVis ให้อยู่ในรูปของเหตุการณ์ (event) ของวิชวลเบสิก เพื่อให้ผู้พัฒนาองค์ประกอบสร้างโปรแกรมวิชวลเบสิกตอบสนองต่อเหตุการณ์เหล่านั้น การที่ต้องมีตัวควบคุมเฉพาะเพิ่มเข้ามาในโปรแกรมก็เนื่องจากข้อจำกัดของโปรแกมที่พัฒนาด้วยวิชวลเบสิกรุ่น 3.0 ไม่สามารถสร้างเหตุการณ์ใหม่ได้ด้วยตัวโปรแกรมเอง

```
LONG DoOutputNotify(HCTL hctl,AVIS_LPMSG lpMsg)
{
    OUTPUTNOTIFY_PARAMS params;
    if ( VBGetMode() == MODE_BREAK ) {
        AVisSuspendMessage(nID,SM_BREAK);
        return AVIS_MSGRESULT_SUSPENDED;
    }
    AVisSuspendMessage(MyID,SM_OUTPUT);
    . . .
    /* ReplyMessage ,prevent dead lock when user debug VB program */
    ReplyMessage(AVIS_MSGRESULT_OK);
    /* Fire OutputNotify Event !! */
    VBFireEvent(hctl,IEVENT_COMPONENT_OUTPUTNOTIFY,&params);
    . . .
    AVisResumeMessage(MyID,SM_OUTPUT);

    /* Allow  Sender to continue */
    AVisReplyMessage(lpMsg->nSenderID,0);
    return 0;  // return value is not  used
}
```

รูปที่ ช.1 ตัวอย่างการแปลงข้อความคำสั่งเป็นเหตุการณ์ของวิชวลเบสิก

---

[1] Microsoft. Microsoft Visual Basic Programming System for Windows Professional Feature Book One version 3.0. Redmond.WA : Microsoft Inc. 1993.

ในรูปที่ ช.1 จะแสดงส่วนของโปรแกรมของ AVISCOMP.VBX ที่ใช้แปลงข้อความคำสั่ง AVIS_WM_OUTPUTNOTIFY ให้เป็นเหตุการณ์ OutputNotify ของวิชวลเบสิก การทำงานจะเริ่มจากการตรวจสอบว่าในขณะที่ได้รับข้อความคำสั่งตัวองค์ประกอบกำลังอยู่ระหว่างการหยุดรอคำสั่งจากผู้พัฒนาเพื่อการตรวจสอบการทำงาน (break mode) อยู่หรือไม่ ถ้าใช่แสดงว่าในขณะนี้องค์ประกอบยังไม่พร้อมที่จะรับข้อความคำสั่งใดๆ หากตรวจสอบแล้วพบว่าองค์ประกอบพร้อมที่จะทำงานก็จะทำการเรียกใช้ฟังก์ชันเพื่อป้องกันการซ้อนทับของข้อความคำสั่งและการติดตายตามลำดับ แล้วจึงสร้างเหตุการณ์ OutputNotify ของตัวควบคุมขึ้น ด้วยคำสั่ง VBFireEvent หลังจากที่วิชวลเบสิกทำงานในส่วนของโปรแกรมตอบสนองเหตุการณ์เรียบร้อยแล้วก็จะกลับมาทำงานยังโปรแกรมในรูปที่ ช.1 ต่อโดยจะยกเลิกการป้องกันการทับซ้อนของข้อความคำสั่ง แล้วแจ้งเตือนหน่วยบริหารการจินตทัศน์ว่าทำงานเสร็จเรียบร้อยแล้วด้วยฟังก์ชัน AVisReplyMessage

### *2 การเรียกใช้บริการของ AVis*

เนื่องจากระบบ AVis ถูกออกแบบให้ใช้งานร่วมกับภาษาการพัฒนาโปรแกรมหลายภาษา ดังนั้นจึงไม่สามารถออกแบบให้นำความสามารถทั้งหมดของวิชวลเบสิกมาใช้ได้เนื่องจากความสามารถเหล่านี้ไม่สามารถใช้ได้ในภาษาการพัฒนาโปรแกรมภาษาอื่น ตัวอย่างของความสามารถเหล่านี้ได้แก่การจัดการข้อผิดผลาด On Error Goto ของวิชวลเบสิก ซึ่งเป็นระบบที่ทำให้การพัฒนาโปรแกรมบนวิชวลเบสิกทำได้ง่ายขึ้น ด้วยเหตุนี้จึงได้เพิ่มฟังก์ชันเข้าไปใน AVISCOMP.VBX เพื่อให้ผู้พัฒนาองค์ประกอบเรียกใช้แทนฟังก์ชันของ AVis ฟังก์ชันเหล่านี้จะทำให้สามารถใช้ความสามารถของคำสั่ง On Error Goto ของวิชวลเบสิกกับฟังก์ชันของ AVis ได้ ชื่อของฟังก์ชันที่ถูกเก็บอยู่ใน AVISCOMP.VBX จะแสดงไว้ในตารางที่ ช.1

| ชื่อ | หน้าที่ |
|---|---|
| AVisVBOpenSync | เหมือนกับ AVisOpenSync |
| AVisVBCloseSync | เหมือนกับ AVisCloseSync |
| AVisVBSync | เหมือนกับ AVisSync |
| AVisVBInputRequest | เหมือนกับ AVisInputRequest |
| AVisVBInputRequestEx | เหมือนกับ AVisInputRequestEx |
| AVisVBOutputNotify | เหมือนกับ AVisOutputNotify |
| AVisVBOutputNotifyEx | เหมือนกับ AVisOutputNotifyEx |
| AVisVBGetComponentErrorStr | เหมือนกับ AVisGetErrorMsg |

ตารางที่ ช.1 ฟังก์ชันของ AVISCOMP.VBX

การทำงานของฟังก์ชันเหล่านี้ทั้งหมดจะคล้ายกันก็คือจะฟังก์ชันเหล่านี้จะทำหน้าที่เป็นเพียงจุดเรียกใช้และรับผลการทำงานของฟังก์ชันที่เกี่ยข้อง โดยจะทำหน้าที่แปลงค่าพารามิเตอร์ที่รับจากโปรแกรมที่พัฒนาด้วยวิชวลเบสิกให้อยู่ในรูปแบบที่ฟังก์ชันของ AVis ต้องการ และคอยแปลงค่าผลการทำงานของ AVis ให้อยู่ในรูปที่โปรแกรมภาษาวิชวลเบสิกนำไปใช้งานได้ง่ายขึ้น

รูปที่ ช.2 แสดงการทำงานของฟังก์ชัน AVisVBOutputNotify ที่ทำการแปลงค่าพารามิเตอร์ของฟังก์ชันให้อยู่ในรูปที่ AVisOutputNotify ต้องการ แล้วทำการเรียกฟังก์ชัน AVisOutputNotify มาทำงาน และเมื่อทำงานเสร็จเรียบร้อยแล้วแทนที่จะให้ค่าผลการทำงานเป็นตัวเลขซึ่งผู้เรียกใช้ฟังก์ชันต้องตรวจสอบค่าอีกครั้ง AVisVBOutputNotify จะตรวจสอบว่าค่าผลการทำงานที่ได้ว่าเป็นค่าแสดงความผิดผลาดหรือไม่ ถ้าใช่ก็จะทำการกำหนดข้อความแสดงความผิดผลาดที่จะปรากฏขึ้น แล้วเรียกใช้ฟังก์ชัน VBRuntimeError เพื่อทำให้เกิด runtime error ในวิชวลเบสิก ซึ่ง runtime error ที่เกิดขึ้นนี้จะตรวจจับได้จากการใช้คำสั่ง On Error Goto ของวิชวลเบสิก

```
void FAR PASCAL _export AVisVBOutputNotify(HCTL hctl,UINT wMsg,
                                LPARAM lP1,LPARAM lP2,LPSTR str)
{
    int id;
    LPCOMPONENT pComponent = DEREFCOMPONENT(hctl);

    /* data validation */
    if ( pComponent == NULL || pComponent->lSignature != SIGN_COMPONENT )
        ErrorReturn(AVIS_ERR_INVALIDID);

    /* Call AVis Function */
    id = pComponent->nID;
    pComponent->nError = AVisOutputNotify(id,wMsg,lP1,lP2,str);

    if(pComponent->nError != AVIS_ERR_OK ) /* Error !!! */
      ErrorReturn(pComponent->nError);
}

void ErrorReturn(int errcode)
{
  if ( errcode != AVIS_ERR_OK ) {
    VBSetErrorMessage(errcode.AVisGetErrorMsg(errcode))
    VBRuntimeError(errcode);
  }
}
```

รูปที่ ช.2 การทำงานของฟังก์ชัน AVisVBOutputNotify

## ภาคผนวก ซ.

## โปรแกรมควบคุมการจินตทัศน์

โปรแกรมควบคุมการจินตทัศน์เป็นโปรแกรมบนระบบไมโครซอฟต์วินโดว์รุ่น 3.1 ซึ่งจะทำหน้าที่ควบคุมการจินตทัศน์ให้เป็นไปตามที่ผู้ใช้การจินตทัศน์ต้องการ โดยโปรแกรมควบคุมการจินตทัศน์สามารถควบคุมการจินตทัศน์ได้โดยการเรียกใช้ฟังก์ชันต่างๆของหน่วยบริหารการจินตทัศน์ นอกจากนี้โปรแกรมควบคุมการจินตทัศน์จะต้องคอยตอบสนองต่อข้อความคำสั่งต่างๆ ที่หน่วยบริหารการจินตทัศน์จะส่งให้เพื่อแจ้งสถานะของระบบ

### *1 การและถอนทะเบียน*

สิ่งที่โปแกรมควบคุมการจินตทัศน์จะต้องกระทำเป็นสิ่งแรกก่อนที่จะเริ่มติดต่อกับหน่วยบริหารการจินตทัศน์ก็คือการลงทะเบียน ซึ่งจะทำได้โดยการเรียกใช้ฟังก์ชัน AVisRegisterController ระบบ AVis จะยินยอมให้มีโปรแกรมควบคุมการการจินตทัศน์เพียงโปรแกมเดียวทำงานอยู่ ซึ่งหากในขณะนั้นมีโปรแกรมควบคุมการการจินตทัศน์ทำงานอยู่แล้วฟังก์ชันนี้จะให้ค่าผลการทำงานผิดผลาดกลับมา ซึ่งหลังจากลงทะเบียนแล้วโปรแกรมที่ทำการลงทะเบียนก็จะมีสิทธิ์เรียกใช้ฟังก์ชันที่สงวนไว้สำหรับโปรแกรมควบคุมการการจินตทัศน์

เมื่อโปรแกรมควบคุมการการจินตทัศน์ต้องการยุติการทำงานของตัวเองลงจะต้องทำการเรียกใช้ฟังก์ชัน AVisUnregisterController ก่อนเพื่อแจ้งให้ระบบทราบเพื่อที่ระบบจะได้ยินยอมให้โปรแกรมอื่นทำหน้าที่เป็นโปรแกรมควบคุมการจินตทัศน์แทน

### *2 บทการจินตทัศน์*

หลังจากที่โปรแกรมควบคุมการการจินตทัศน์ทำการลงทะเบียนเรียบร้อยแล้วและต้องการเรียกบทการจินตทัศน์มาทำงาน โปรแกรมควบคุมการการจินตทัศน์จะต้องแจ้งให้ระบบทราบก่อนด้วยฟังก์ชัน AVisOpenSession เพื่อที่ระบบจะได้จัดเตรียมทรัพยากรต่างๆให้พร้อมที่จะทำการจินตทัศน์ต่อไป และเมื่อต้องการยุติการทำงานของบทการจินตทัศน์เสร็จสิ้นแล้วโปรแกรมควบคุมการการจินตทัศน์จะต้องเรียกฟังก์ชัน AVisCloseSession มาทำงานเพื่อให้ระบบยุติการทำงาน (terminate program)ขององค์ประกอบการจินตทัศน์ทั้งหมดภายในบทการจินตทัศน์ และยกเลิกการใช้ทรัพยากรต่างๆที่เตรียมไว้เมื่อมีการเรียกฟังก์ชัน AVisOpenSession

### *3 การเรียกองค์ประกอบการจินตทัศน์มาทำงาน*

การเรียกองค์ประกอบการจินตทัศน์มาทำงานตามบทการจินตทัศน์มีขั้นตอนอยู่สองขั้นตอนคือ

1. เรียกองค์ประกอบการจินตทัศน์มาทำงานด้วยฟังก์ชัน AVisLoadComponent
2. เชื่อมองค์ประกอบต่างๆเข้าด้วยกัน เพื่อสร้าง routing graph สำหรับการทำงานของ AVisSync ซึ่งจะทำได้โดยการเรียกใช้ฟังก์ชัน AVisBind สำหรับองค์ประกอบแต่ละคู่ที่เกี่ยวข้องกัน

เมื่อจบการการทำงานของบทการจินตทัศน์แล้ว โปรแกรมควบคุมการจินตทัศน์ไม่จำเป็นต้องยุติการทำงานขององค์ประกอบเพราะระบบจะยุติการทำงานขององค์ประกอบเหล่านี้ รวมทั้งลบข้อมูลที่เกียวข้องทั้งหมดใน routing graph ให้โดยอัตโนมัติเมื่อมีการเรียกใช้ฟังก์ชัน AVisCloseSession อย่างไรก็ตามหากโปรแกรมควบ

คุมการจินตทัศน์ต้องการยุติการทำงานหรือลบข้อมูลใน routing graph เองก็สามารถทำได้โดยการเรียกใช้ฟังก์ชัน AVisUnloadComponent และ AVisUnbind ตามลำดับ

### *4 การควบคุมการจินตทัศน์*

เมื่อทำการเรียกและเชื่อมต่อองคีประกอบการจินตทัศน์ต่างๆ เข้าด้วยกันแล้ว โปรแกรมควบคุมการจินตทัศน์สามารถสั่งให้เริ่มการจินตทัศน์ได้ด้วยการสั่งให้ AVisSync ทำงานในภาวะ การทำงานตามปรกติ (AVIS_SM_RUN) และทำงานทีละจังหวะ (AVIS_SM_STEP) ได้ด้วยการใช้ฟังก์ชัน AVisSetSyncMode ตั้งค่าภาวะของ AVisSync ตามต้องการ นอกจากนี้ฟังก์ชันนี้ยังใช้ในหยุดการทำงานชั่วคราว (Pause) และการยุติการทำงาน (Stop) ของการจินตทัศน์ได้อีกด้วยด้วยการตั้งค่าภาวะให้มีค่าเป็น AVIS_SM_PAUSE และ AVIS_SM_STOP ตามลำดับ

ฟังก์ชันอื่นๆของ AVis ที่ใช้ในการควบคุมการจินตทัศน์ของ AVis ได้แก่ฟังก์ชัน AVisStepNextSync ซึ่งจะใช้เมื่อ AVisSync ทำงานอยู่ในภาวะทำงานทีละจังหวะ โดยเมื่ออัลกอริทึมทุกอัลกอริทึมทำงานได้หนึ่งจังหวะแล้ว AVisSync จะแจ้งให้โปรแกรมควบคุมการจินตทัศน์ทราบด้วยข้อความ AVIS_NM_FINISHSYNCSTEP แล้วจะหยุดรอจนกว่าโปรแกรมควบคุมการจินตทัศน์เรียกใช้ฟังก์ชัน AVisStepNextSync AVisSync จึงจะยอมให้อัลกอริทึมเริ่มทำงานใหม่อีกหนึ่งจังหวะ

นอกจากฟังก์ชันที่ใช้ควบคุมการทำงานของ AVisSync แล้ว ยังมีฟังก์ชันอีกสองฟังก์ชันที่โปรแกรมควบคุมการจินตทัศน์อาจเรียกใช้ได้เพื่อตรวจสอบค่าสถานะของ AVisSync ฟังก์ชันทั้งสองได้แก่ AVisGetSyncMode ซึ่งจะให้ค่าภาวะการทำงานของ AVisSync ในขณะนั้น และ AVisGetSyncStatus ที่จะให้ค่าสถานะการทำงานของ AVisSync

### *5 การตรวจสอบข้อผิดผลาด*

หน้าที่อีกประการหนึ่งของโปรแกรมควบคุมการจินตทัศน์ก็คือจะต้องร่วมมือกับ AVisErr เพื่อตรวจหาองค์ประกอบการจินตทัศน์ที่หยุดทำงานโดยไม่ถอนทะเบียนออกจากระบบก่อน การตรวจสอบนี้จะทำได้โดยการเรียกใช้ฟังก์ชัน AVisCheckComponent เป็นระยะๆ เพื่อให้ AVisErr ทำการตรวจสอบและแก้ไขข้อผิดพลาดต่างๆที่เกิดขึ้น ซึ่งหากตรวจพบ AVisErr จะแจ้งเตือนให้โปรแกรมควบคุมการจินตทัศน์ทราบด้วยข้อความ AVIS_NM_TERMINATED เพื่อให้โปรแกรมควบคุมการจินตทัศน์แก้ไขข้อผิดผลาดที่เกิดขึ้นต่อไป

### *6 การตั้งค่าพารามิเตอร์ขององค์ประกอบ*

โปรแกมควบคุมการจินตทัศน์สามารถตั้งค่าพารามิเตอร์ขององค์ประกอบการจินตทัศน์ได้ด้วยการใช้ฟังก์ชัน AVisConfigComponent ซึ่งในการรียกใช้ฟังก์ชันนี้โปรแกมควบคุมการจินตทัศน์จะต้องระบุหมายเลขขององค์ประกอบที่ต้องการตั้งค่า หมายเลขของพารามิเตอร์ และค่าของพารามิเตอร์

### *7 ข้อความแจ้งเตือนของระบบ AVis*

ระบบ AVis จะส่งข้อความแจ้งเตือน (Notify Message) ในรูปของข้อความคำสั่งของข้อความคำสั่งของวินโดว์ (Windows Message) มาให้โปรแกรมควบคุมการจินตทัศน์เพื่อแจ้งเตือนถึงสภาพการทำงานของระบบ ข้อความคำสั่งเหล่านี้ได้แก่

| ข้อความแจ้งเตือน | แจ้งเตือนเมื่อ |
|---|---|
| AVIS_NM_REGISTER | มีองค์ประกอบลงทะเบียนเข้าสู่ระบบใหม่ |
| AVIS_NM_UNREGISTER | มีองค์ประกอบกำลังถอนทะเบียนออกจากระบบ |
| AVIS_NM_TERMINATED | องค์ประกอบยุติการทำงานโดยไม่ได้ถอนทะเบียนออกจากระบบ |
| AVIS_NM_FINISHSYNCSTEP | องค์ประกอบอัลกอริทึมทุกองค์ประกอบทำงานเสร็จหนึ่งจังหวะ จะเกิดขึ้นเฉพาะเมื่อ AVisSync ทำงานในภาวะทำงานทีละจังหวะ |
| AVIS_NM_FINISHALLSYNC | องค์ประกอบอัลกอริทึมทุกองค์ประกอบทำงานเสร็จเรียบร้อยแล้ว |
| AVIS_NM_OPENSYNC | มีองค์ประกอบอัลกอริทึมกำลังขอเริ่มการทำงานของอัลกอริทึมด้วยการเรียกใช้ฟังก์ชัน AVisOpenSync |
| AVIS_NM_CLOSESYNC | มีองค์ประกอบอัลกอริทึมกำลังขอจบการทำงานของอัลกอริทึมด้วยการเรียกใช้ฟังก์ชัน AVisCloseSync |

โปรแกรมควบคุมการจินตทัศน์ไม่จำเป็นต้องตอบสนองต่อข้อความแจ้งเตือนเหล่านี้ รายละเอียดของข้อความแจ้งเตือนและพารามิเตอร์สามารถศึกษาเพิ่มเติมได้จากแฟ้ม AVIS.HLP

### *8 AVISCTRL.VBX*

AVISCTRL.VBX เป็นตัวควบคุมเฉพาะของวิชวลเบสิก ซึ่งพัฒนาขึ้นเพื่อช่วยให้การพัฒนาโปรแกรมควบคุมการจินตทัศน์ด้วยวิชวลเบสิกทำได้ง่ายขึ้น AVISCTRL.VBX เป็นตัวควบคุมที่จะทำหน้าที่สองประการคือ (1) ลงทะเบียนและถอนทะเบียนโปรแกรมควบคุมการจินตทัศน์ และ (2) สร้างเหตุการณ์ของวิชวลเบสิกเพื่อแจ้งให้โปรแกรมควบคุมการจินตทัศน์ทราบเมื่อมีข้อความแจ้งเตือนต่างๆ ถูกส่งมาจากหน่วยบริหารการจินตทัศน์

### *9. สรุป*

ในภาคผนวกนี้ได้กล่าวถึงหน้าที่และการทำงานต่างๆ ของโปรแกรมควบคุมการจินตทัศน์อย่างคร่าวๆ รวมถึงการเรียกใช้ฟังก์ชันต่างๆของหน่วยบริหารการจินตทัศน์เพื่อควบคุมและตรวจสอบให้การจินตทัศน์เป็นไปอย่างเรียบร้อย นอกจากนี้ยังได้แสดงถึงข้อความแจ้งเตือนที่หน่วยบริหารการจินตทัศน์ส่งให้โปรแกรมควบคุมการจินตทัศน์เพื่อแจ้งสถานะของระบบให้ทราบ

อย่างไรก็ตามข้อมูลที่กล่าวไว้ในบทนี้เพียงอย่างเดียวอาจจะไม่เพียงพอในการสร้างโปรแกรมควบคุมการจินตทัศน์ขึ้นมาใหม่ หากผู้ใดสนใจการพัฒนาโปรแกรมควบคุมการจินตทัศน์สามารถศึกษาได้จากโปรแกรมต้นฉบับของ AVISRUN ซึ่งอยู่ในชุดโปรแกรมของ AVis ก็จะทำให้เข้าใจการทำงานของโปรแกรมควบคุมการจินตทัศน์ได้ดียิ่งขึ้น

## ภาคผนวก ฌ.

## รูปแบบแฟ้มข้อมูล

### *1 แฟ้มรายละเอียดขององค์ประกอบ*

แฟ้มข้อมูลองค์ประกอบของระบบ AVis เป็นแฟ้มตัวอักษรที่จัดเก็บอยู่ในรูปแบบเดียวกับแฟ้มค่าเริ่มต้นการทำงานของโปรแกรม (initialization file - INI) ของระบบวินโดว์ โดยข้อมูลที่เก็บอยู่ในแฟ้มจะถูกแบ่งออกเป็นหัวข้อ (topic) และข้อมูลจะถูกเก็บอยู่รูปแบบชื่อข้อมูล=ค่า (entry-value) เป็นคู่ๆไป ในการจัดเก็บชื่อของหัวข้อจะต้องอยู่ในเครื่องหมาย [ ] เสมอ เพื่อทำให้สามารถแยกชื่อหัวข้อออกจากคู่ (pair) ของชื่อข้อมูลและค่าข้อมูลได้

แฟ้มรายละเอียดขององค์ประกอบจะมีหัวข้อของข้อมูลอยู่ทั้งหมด 6 หัวข้อได้แก่

1. [Info] เก็บรายละเอียดต่างๆของตัวองค์ประกอบได้แก่

| | |
|---|---|
| Name | ชื่อขององค์ประกอบซึ่งอยู่ในเครื่องหมาย " " |
| Description | รายละเอียดอย่างย่อขององค์ประกอบการจินตทัศน์ซึ่งอยู่ในเครื่องหมาย " " |
| Type | ชนิดขององค์ประกอบ โดยที่<br>0 แทนตัวสร้างข้อมูล<br>1 แทนอัลกอริทึม<br>2 แทนตัวแปรคำสั่ง<br>3 แทนส่วนแสดงผล และ<br>4 แทนองค์ประกอบที่ไม่สามารถจัดเข้าใน 4 ชนิดแรกได้ |
| HelpFile | แสดงชื่อแฟ้มคำอธิบายรายละเอียดขององค์ประกอบ |
| Visible | หากกำหนดให้มีค่าเป็น 0 โปรแกรมควบคุมการจินตทัศน์จะซ่อนตัวโปรแกรมไว้ไม่ให้ผู้ใช้มองเห็นในขณะทำการจินตทัศน์ โดยทั่วไปองค์ประกอบทุกชนิดยกเว้นส่วนแสดงผลจะกำหนดให้ค่านี้เป็น 0 |

2.[Properties] เก็บข้อมูลของค่าพารามิเตอร์ต่างๆขององค์ประกอบที่ผู้ใช้สามารถกำหนดได้ ซึ่งเก็บอยู่ในรูปแบบที่กำหนดด้านล่าง เครื่องหมาย [] จะแสดงถึงเขตข้อมูล (field) ที่สามารถละทิ้งได้

name = id,type[,[default][,[min][,[max][,[misc]]]]]

| | |
|---|---|
| name | แทนชื่อของพารามิเตอร์ |
| id | แทนหมายเลขประจำคุณสมบัติ มีค่าได้ตั้งแต่ 1 ถึง 32767 และจะต้องไม่มีค่าซ้ำกัน |
| type | ชนิดของพารามิเตอร์ซึ่งจะมีอยู่ 8 ชนิดได้แก่ DT_INTEGER, DT_BOOLEAN, DT_COLOR, DT_ENUM, DT_FLOAT, DT_FILENAME, DT_STRING และ DT_FONT |
| default | ค่าปริยายของพารามิเตอร์ |
| min | ค่าน้อยที่สุดที่จะยอมให้ผู้ใช้กำหนด ในกรณีที่พารามิเตอร์เป็นค่าตัวเลข |

max ค่ามากที่สุดที่จะยอมให้ผู้ใช้กำหนด ในกรณีที่พารามิเตอร์เป็นค่าตัวเลข

misc รายละเอียดเพิ่มเติมของพารามิเตอร์

รายละอียดของการกำหนดค่าพารามิเตอร์ชนิดต่างๆ มีดังต่อไปนี้

2.1. พารามิเตอร์แบบ DT_INTEGER และ DT_FLOAT

name = id,{ DT_INTEGER | DT_FLOAT }[,[default][,[min][,[max]]]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตข้อมูลต่างๆเกือบทุกค่ายกเว้นค่า misc โดย ค่าเขตข้อมูลทั้งหมดที่ใช้จะเป็นตัวเลขที่เหมาะสมกับชนิดของข้อมูล เพื่อกำหนดค่าปรกติและค่าสูงสุดต่ำสุดของพารามิเตอร์

2.2. พารามิเตอร์แบบ DT_BOOLEAN

name = id,DT_BOOLEAN[,[-1|0]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตข้อมูลเพียงสามค่าคือ name,type และ default ค่าของเขตข้อมูล default จะมีค่าได้สองค่าคือ -1 แทน True และ 0 แทน False

2.3. พารามิเตอร์แบบ DT_COLOR

name = id,DT_COLOR[,[default]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตข้อมูลเพียงสามค่าคือ name,type และ default ค่าของเขตข้อมูล default จะเป็นตัวเลขแบบ Long ของวิชวลเบสิก ซึ่งแทนค่าสีในระบบ RGB โดยแต่ละไบต์นับจากไบต์ที่มีค่าประจำหลักต่ำสุด(less significant byte) แทนค่า R , G และ B ตามลำดับ ส่วนไบต์สุดท้ายจะต้องเป็นค่าศูนย์เสมอ

2.4. พารามิเตอร์แบบ DT_ENUM

name = id,DT_ENUM[,[default][,,,[enum-list]]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตข้อมูลสี่ค่าคือ name,type,default และ Misc ค่าของเขตข้อมูล default จะเป็นตัวเลขจำนวนเต็มซึ่งแทนค่าของตำแหน่งข้อมูลใน enum-list ซึ่งจะเก็บอยู่ในเขตข้อมูล Misc ซึ่งค่าของ enum-list จะอยู่ในรูปแบบค่าข้อมูลคั่นด้วยเครื่องหมายคอมมา (*entry,entry, ...* )

2.5. พารามิเตอร์แบบ DT_FILENAME

name = id,DT_FILENAME[,[default][,,,[file-filter]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตข้อมูลสี่ค่าคือ name,type,default และ Misc ค่าของเขตข้อมูล default จะเป็นชื่อของแฟ้มข้อมูล file-filter ซึ่งจะเก็บอยู่ในเขตข้อมูล Misc ใช้กำหนดนามสกุล (extention) ของแฟ้มข้อมูลที่จะใช้กับค่าพารามิเตอร์ โดยค่า file-filter จะอยู่ในรูปแบบ description| wild-card ดังเช่น Text file | *.TXT | Help File | *.HLP เป็นต้น

2.6. พารามิเตอร์แบบ DT_STRING

name = id,DT_STRING[,[default][,,,max-length]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตข้อมูลสี่ค่าคือ name,type,default และ Misc ค่าของเขตข้อมูล Misc จะเก็บค่าความยาวสูงสุดของสายอักษรที่พารามิเตอร์จะรับได้

2.7. พารามิเตอร์แบบ DT_FONT

name = id,DT_FONT[,[default]]

ค่าพารามิเตอร์นี้จะใช้ค่าเขตเพียงสามค่าคือname,type และ default ค่าของเขตข้อมูล default จะถูกจัดเก็บอยู่ในรูป font-name,size,bold,italic ดังเช่นค่า AngsanaUPC,16,1,0 จะแทนรูปแบบตัวอักษรแบบ AngsanaUPC ขนาด 16 point และตัวอักษรเป็นแบบตัวหนาและไม่เอน

3.[Input Response] เก็บข้อมูลเกี่ยวกับข้อความคำสั่งขอข้อมูลที่องค์ประกอบรู้จัก ซึ่งจะจัดเก็บอยู่ในรูปแบบ

msg_name = value

msg_name จะแทนชื่อของข้อความคำสั่ง

value จะแทนหมายเลขประจำข้อความคำสั่ง

4.[Input Request] เก็บข้อมูลที่เกี่ยวกับข้อความคำสั่งขอข้อมูลที่องค์ประกอบจะสร้างขึ้นเพื่อขอข้อมูลจากองค์ประกอบอื่น ข้อมูลจะถูกจัดเก็บอยู่ในลักษณะเดียวกับข้อมูลในหัวข้อ [Input Response]

5.[Output Response] เก็บข้อมูลเกี่ยวกับข้อความคำสั่งแสดงผลที่องค์ประกอบรู้จัก จัดเก็บอยู่ในรูปแบบเดียวกับข้อมูลในหัวข้อ [Input Response]

6.[Output Request] เก็บข้อมูลเกี่ยวกับข้อความคำสั่งแสดงผลที่องค์ประกอบอาจสร้างขึ้น จัดเก็บอยู่ในรูปแบบเดียวกับข้อมูลในหัวข้อ [Input Response]

### *2 แฟ้มบทการจินตทัศน์*

แฟ้มบทการจินตทัศน์จะมีลักษณะดังในรูปที่ 1.1 ซึ่งจะเห็นว่าข้อมูลในแฟ้มการจินตทัศน์จะถูกแบ่งออกเป็นสามส่วนตามลักษณะของข้อมูลได้แก่

1. รายละเอียดของการจินตทัศน์ (บรรทัดที่ 1- 6)
2. องค์ประกอบการจินตทัศน์ทั้งหมดที่ใช้ในการจินตทัศน์ และค่าพารามิเตอร์ขององค์ประกอบเหล่านี้ (บรรทัดที่ 7- 35)
3. ความสัมพันธ์ระหว่างองค์ประกอบการจินตทัศน์ (บรรทัดที่ 36- 40)

### 2.1 รายละเอียดของการจินตทัศน์

รายละเอียดส่วนนี้จะประกอบไปด้วยข้อมูลย่อย 6 ค่าได้แก่

| ชื่อ | ความหมาย |
|---|---|
| VERSION | รุ่นของแฟ้มบทการจินตทัศน์ |
| DOMAIN | ชื่อกลุ่มของปัญหาที่ใช้ |
| NAME | ชื่อบทการจินตทัศน์ |
| DESCRIPTION | รายละเอียดอย่างย่อของบทการจินตทัศน์ |
| AUTHOR | ชื่อผู้สร้างบทการจินตทัศน์ |
| COMMENTS | หมายเหตุต่างๆ |

ค่าข้อมูลเหล่านี้จะต้องอยู่ในบันทัดเดียวกับชื่อข้อมูล โดยใช้ช่องว่าง (blank) คั่นระหว่างชื่อและค่าข้อมูล

## 2.2 รายละเอียดขององค์ประกอบในบทการจินตทัศน์

ข้อมูลในส่วนนี้จะจัดเก็บรายละเอียดต่างๆขององค์ประกอบการจินตทัศน์ทั้งหมดที่ใช้ในบทการจินตทัศน์ในลักษณะ

Begin *Component_Path Component_name*

*PreDefined_Parameter = value*

*UserDefined_Parameter <ID> = value*

End

โดย

| | |
|---|---|
| Component_Path | แทนชื่อเต็มของโปรแกรมและไดเร็คทอรี่ขององค์ประกอบ |
| Component_Name | แทนชื่อขององค์ประกอบที่จะใช้อ้างในบทการจินตทัศน์ |
| Predefined_Parameters | แทนชื่อพารามิเตอร์ซึ่งโปรแกรมออกแบบการจินตทัศน์กำหนดขึ้น ศึกษาเพิ่มเติมได้จากภาคผนวก ข |
| UserDefined_Parameters | แทนชื่อพารามิเตอร์ที่ผู้พัฒนาองค์ประกอบการจินตทัศน์กำหนดขึ้น |
| ID | แทนหมายเลขของพารามิเตอร์ที่ผู้ใช้กำหนดขึ้น ค่านี้จะต้องตรงกับค่าในแฟ้มรายละเอียดการจินตทัศน์ |

## 2.3 ความสัมพันธ์ระหว่างองค์ประกอบการจินตทัศน์

ข้อมูลในส่วนนี้จะจัดเก็บความสัมพันะระหว่างองค์ประกอบต่างๆในบทการจินตทัศน์ในลักษณะ

Binding

*DataProvider_Name,DataReceiver_Name*

...

End

โดย DataProvide_Name และ DataReceiver_Name เป็นชื่อขององค์ประกอบที่ต้องการกำหนดความสัมพันธ์

จากแฟ้มในรูปที่ ฌ.1 จะพบว่าการจินตทัศน์อัลกอริทึมนี้จะประกอบไปด้วยองค์ประกอบการจินตทัศน์ 4 องค์ประกอบ ซึ่งแต่ละองค์ประกอบก็จะมีค่าพารามิเตอร์ต่างกันออกไป โดยองค์ประกอบเหล่านี้จะมีอยู่ 3 คู่ที่มีความสัมพันธ์กันในขณะทำการจินตทัศน์ได้แก่ความสัมพันธ์ระหว่าง HeapSort1-Convertor1,Random1-HeapSort1และConvertor1-DotView1

```
VERSION   1.00
DOMAIN    SORT.AVD
NAME      Untitled
DESCRIPTION
AUTHOR
COMMENTS

Begin DOTCNV\DOTCNV.EXE Covertor1
    DesignPos = (181,166)
    AutoLoad = -1
End
Begin HEAPSORT\HEAPSORT.EXE HeapSort1
    DesignPos = (102,82)
    AutoLoad = -1
End
Begin RANDOM\RANDOM.EXE Random1
    DesignPos = (183,5)
    AutoLoad = -1
    DataType<2> = 0
    DataCount<3> = 300
End
Begin STICK\STICK.EXE DotView1
    DesignPos = (95,240)
    AutoLoad = -1
    WindowState = 1
    Left = 120
    Top = 308
    Width = 356
    Height = 207
    ForeColor<2> = 32768
    BackColor<3> = 8388608
    ViewType<4> = 0
    LineWidth<5> = 2
End

Binding
    HeapSort1.Covertor1
    Random1.HeapSort1
    Covertor1.DotView1
End

End.
```

รูปที่ ฌ.1 แฟ้มการจินตทัศน์อัลกอริทึม

# ประวัติผู้เขียน

นายชัชวาล วงศ์ศิริประเสริฐ เกิดเมื่อวันที่ 6 ตุลาคม พ.ศ. 2515 ที่กรุงเทพมหานคร สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตร์ สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2536 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ ที่จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ. 2538 และเป็นผู้นำเสนอผลงานวิจัยชิ้นนี้ซึ่งได้ตีพิมพ์ผลงานในการประชุมใหญ่วิชาการทางวิศวกรรมประจำปี 2538 ในเดือนสิงหาคม พ.ศ. 2538