บทที่ 4
ส่วนประกอบของโปรแกรมออโตแคดที่เกี่ยวข้อง

4.1 โปรแกรม ออโตแคด (AutoCAD)

โปรแกรมออโตแคด เป็นโปรแกรมสำเร็จรูป ช่วยในการเขียนแบบที่นิยมใช้กันแพร่หลายในปัจจุบัน พัฒนาขึ้นโดย บริษัท ออโตเดสค์ และได้มีการปรับปรุงโปรแกรมให้ดีขึ้นตลอดมา โปรแกรมออโตแคด รุ่นที่ 12 เมื่อติดตั้งแล้ว จะประกอบไปด้วยส่วนต่าง ๆ เช่น

ADS (AutoCAD Development System) เป็นคลังโปรแกรม (Library) พิเศษของฟังก์ชันภาษา ซี แมโคร และค่าคงที่ (predefined constants) เพื่อให้ผู้พัฒนาและผู้ใช้โปรแกรมออโตแคด สามารถที่จะสร้างส่วนขยาย (Extension) ของโปรแกรมออโตแคด ช่วยให้โปรแกรมออโตแคด ทำงานได้อย่างมีประสิทธิภาพโดยใช้โปรแกรมภาษา ซี

API (Application Programming Interface) เป็นกลุ่มกฎเกณฑ์มาตราฐานที่ออโตแคดกำหนดขึ้นสำหรับฟังก์ชันต่างๆ ภายนอก ที่ใช้เชื่อมต่อ เรียกใช้คำสั่ง และฟังก์ชันภายในโปรแกรมออโตแคด มาทำงาน

ASE (AutoCAD SQL Extension) เป็นส่วนขยายของโปรแกรมออโตแคด ที่ใช้เชื่อมต่อทำงานกับระบบฐานข้อมูลภายนอกได้ โดยใช้ ภาษาสอบถามเชิงโครงสร้าง (Structured Query Language : SQL)

DRV เป็นที่เก็บโปรแกรมขับ (Driver) ต่างๆ ที่ใช้ในโปรแกรมออโตแคด

FONTS เป็นที่เก็บแฟ้มข้อมูลรูปแบบตัวอักษร (font files) ที่ใช้ในโปรแกรม ออโตแคด

SAMPLE เป็นที่เก็บแฟ้มข้อมูลที่ใช้เป็นตัวอย่าง มีทั้ง แฟ้มงานเขียนแบบ (drawing files) แฟ้มภาษา ดีซีแอล แฟ้มภาษาออโตลิสป์ ฯลฯ

SOURCE เป็นที่เก็บโปรแกรมต่าง ๆ ที่เป็นรหัสต้นฉบับ (source codes) เช่น แฟ้มรหัสต้นฉบับรูปแบบตัวอักษร (shp files)

SUPPORT เป็นส่วนที่โปรแกรมออโตแคด ใช้เก็บโปรแกรมเสริมต่าง ๆ ที่ใช้ในระหว่างการทำงาน เช่น แฟ้มควบคุมกรอบสนทนา แฟ้มภาษาออโตลิสป์ แฟ้มรายการเลือก (menu files)

TUTORIAL เป็นที่เก็บของกลุ่มโปรแกรมที่ช่วยในการฝึกการใช้งานโปรแกรมออโตแคด

4.1.1 หลักการเบื้องต้นในการเขียนแบบด้วยคอมพิวเตอร์

ในการเขียนแบบด้วยคอมพิวเตอร์นั้น คอมพิวเตอร์จะอาศัยหน่วยวัดของตัวโปรแกรมเองในการเขียน อ้างอิง หรือคำนวณค่าต่างๆ เรียกว่าหน่วยของการเขียน (Drawing unit) ซึ่งผู้ใช้โปรแกรมจะเป็นผู้สมมติว่าระยะหน่วยของการเขียนนั้นจะมีความหมายเท่ากับระยะเท่าใดในโลกของความเป็นจริง เช่น เส้นตรงยาว 1 หน่วย สำหรับบุคคลหรือวงการต่างกันอาจจะอ่านด้วยความหมายต่างกันได้ เช่น อาจจะหมายถึง 1 มิลลิเมตร หรือ 1 เซนติเมตร หรือ 1 เมตร ฯลฯ ก็ได้ จึงสมควรอย่างยิ่งที่ผู้ใช้จะต้องกำหนดค่ามาตราฐานเหล่านี้ให้ชัดเจนเพื่อความสะดวกในการทำงาน

โดยทั่วไปในการเขียนแบบโครงสร้างอาคาร นิยมกำหนดความยาว 1 หน่วยของการเขียนเท่ากับ 1 เมตร และในการเขียนแบบนั้นเป็นการเขียนแบบเท่าของจริงเสมอ ตัวอย่างเช่นถ้าจะเขียนแบบพื้นคอนกรีตที่มีความยาว 1 เมตร ก็จะเขียนแบบในโปรแกรมให้พื้นคอนกรีตมีความยาวเท่ากับ 1 หน่วยของการเขียน ส่วนการควบคุมมาตราส่วน (Scale) จะไปกำหนดมาตราส่วนทีหลังในขั้นตอนการวาด (Plot) ให้ได้มาตราส่วนตามที่ต้องการ

4.1.2 ระบบการกำหนดตำแหน่ง

การกำหนดตำแหน่งในโปรแกรมออโตแคดเป็นระบบการกำหนดตำแหน่งแบบ 3 มิติ โดยใช้พิกัดในแนวแกน X, แกน Y และแกน Z

World Coordinate System (WCS) คือ แกนหลักที่ใช้ในการสร้างแบบ มีแนวแกนหลักอยู่ 3 แนวแกนคือ แนวแกน X แนวแกน Y และแนวแกน Z ถ้าหากเป็นการเขียนแบบ 2 มิติ จะใช้เพียง 2 แนวแกน คือ แนวแกน X และ แนวแกน Y

User Coordinate System (UCS) คือระนาบแกนที่สร้างขึ้นโดยผู้ใช้ เพื่อให้เกิดความสะดวกในการเขียนแบบ โดยเฉพาะอย่างยิ่งในการเขียนระบบ 3 มิติ จะช่วยให้การสร้างองค์ประกอบ 3 มิติเป็นไปได้อย่างรวดเร็ว

การอ้างอิงถึงมุม โดยปกติออโตแคดจะให้ค่ามุมเป็นศูนย์ ในทิศทาง 3 นาฬิกา (ดูตามหน้าปัทม์นาฬิกา ค่ามุมศูนย์จะอยู่ที่เลข 3) จะให้ค่าที่เป็นบวก (+) เมื่อวัดทวนเข็มนาฬิกา และจะให้ค่าที่เป็นลบ (-) เมื่อวัดตามเข็มนาฬิกา

วิธีการกำหนดตำแหน่ง

1. ใช้เลื่อนตัวชี้ไปยังบริเวณที่ต้องการแล้วกดปุ่มซ้ายของเมาส์
2. **แบบสัมบูรณ์ (Absolute)** คือป้อนค่าในแนวแกน X และ Y ลงไปโดยตรงในรูปแบบ **X,Y** ในกรณีระบบ 3 มิติ บางคำสั่งสามารถป้อนค่าในแกน Z ได้ในรูปแบบ **X,Y,Z**

3. **แบบสัมพัทธ์ (Relative)** คือการป้อนค่าที่เปลี่ยนแปลงไปจากจุดเดิม ในรูปแบบของ **@X,Y** ในกรณีระบบ 3 มิติ บางคำสั่งสามารถป้อนค่าในแกน Z ได้ในรูปแบบ **@X,Y,Z**

4. **แบบเชิงมุม (Polar)** คือการป้อนค่าระยะห่างจากจุดเดิม และมุมที่เปลี่ยนแปลงไปในรูปแบบของ **@d<A** โดยที่ d หมายถึงระยะห่าง และ A หมายถึงค่ามุม

5. **แบบทรงกลม (Spherical)** คือกำหนดระยะห่างมุมในระนาบ XY และมุมที่ทำกับระนาบ XY ในรูปแบบของ **d<ang1<ang2** โดยที่ d หมายถึง ระยะทาง ang1 หมายถึงมุมในระนาบ XY และ ang2 หมายถึงมุมที่ทำกับระนาบ XY หากใส่เครื่องหมาย @ ไว้ข้างหน้า จะเป็นการกำหนดตำแหน่งแบบสัมพันธ์กับจุดสุดท้าย ในรูปแบบของ **@d<ang1<ang2**

6. **แบบทรงกระบอก (Cylindrical)** คือกำหนดระยะห่างจากจุดกำเนิด (0,0) มุมที่ทำในระนาบ XY และค่าในแกน Z ในรูปแบบของ **d<ang1<d2** โดยที่ d หมายถึง ระยะในระนาบ XY ang1 หมายถึงมุมที่ทำในระนาบ XY และ d2 หมายถึงระยะในแกน Z หากใส่เครื่องหมาย @ ไว้ข้างหน้า จะเป็นการกำหนดตำแหน่ง แบบสัมพันธ์กับจุดสุดท้าย ในรูปแบบของ **@d<ang1<d2**

4.1.3 ชุดคำสั่งในโปรแกรมออโตแคด รุ่นที่ 12

คำสั่งในโปรแกรมออโตแคด รุ่นที่ 12 สามารถแบ่งออกเป็นหมวดหมู่ตามลักษณะการใช้งานได้ดังนี้

1. **คำสั่งอรรถประโยชน์ (Utilities command)** ใช้สำหรับควบคุมระดับพื้นฐาน และจัดการกับแฟ้มข้อมูล เช่น คำสั่งในการปิด-เปิดแฟ้มข้อมูล คำสั่งบรรจุ-จัดเก็บแฟ้มข้อมูล เป็นต้น

2. **คำสั่งในการวาด (Drawing command)** ใช้สำหรับการวาดชิ้นส่วนต่างๆลงไปในแบบ เช่น คำสั่งในการลากเส้นตรง คำสั่งในการวาดวงกลม เป็นต้น

3. **คำสั่งแก้ไข ดัดแปลง (Editing command)** ใช้สำหรับการแก้ไข และเปลี่ยนแปลงข้อมูลที่เขียนไปแล้ว

4. **คำสั่งช่วยในการเขียน (Drawing aid)** ใช้สำหรับช่วยเหลือการเขียน และให้ข้อมูลเบื้องต้นกับผู้ใช้ เช่น คำสั่งในการวัดระยะ คำสั่งในการหาจุดพิกัด เป็นต้น

5. **คำสั่งเกี่ยวกับข้อมูลจำเพาะของชิ้นส่วน (Entity property)** ใช้สำหรับการควบคุมจัดระเบียบ แยกแยะข้อมูลจำเพาะของชิ้นส่วนที่เขียน

6. **คำสั่งควบคุมการแสดงผล (Display control)** เช่น คำสั่งในการขยายภาพ (zoom) คำสั่งในการเลื่อนภาพ (pan) เป็นต้น

7. **คำสั่งเกี่ยวกับบล็อก (Block) และลักษณะประจำ (Attribute)**

8. **คำสั่งในการให้เส้นบอกระยะ (Dimensioning command)**

9. คำสั่งในการจัดการกับภาพประเภทแผนที่บิต (Bit map)
10. คำสั่งในการสร้างภาพในระบบ 3 มิติ
11. คำสั่งดำเนินการในระบบรูปทรง 3 มิติแบบ Solid Modeling
12. คำสั่งในการส่งข้อมูลออกไปยังเครื่องวาด (Plotting)

4.1.4 ตัวแปรควบคุมระบบในโปรแกรมออโตแคด รุ่นที่ 12

เป็นตัวแปรระบบของโปรแกรมออโตแคด ที่ใช้ควบคุมสภาวะแวดล้อมต่างๆ (ดูใน ภาคผนวก ข.)

4.2 โปรแกรมภาษาออโตลิสป์ (AutoLISP Programming Language)

ออโตลิสป์ จัดเป็นโปรแกรมของตัวแปลภาษา (Interpreter) ชนิดหนึ่ง ที่สามารถใช้เขียนคำสั่งควบคุมการทำงานของโปรแกรมออโตแคดได้

การทำงานของออโตลิสป์ เมื่อป้อนข้อมูลเข้าที่บรรทัดคำสั่ง (command line) ออโตลิสป์จะทำการอ่านข้อมูลเหล่านั้น ถ้าข้อมูลที่ป้อนเข้าไปถูกต้องตามกฎเกณฑ์ ออโตลิสป์ก็จะทำการประเมินค่าข้อมูลตามฟังก์ชัน และแสดงผลลัพธ์ที่ได้ออกมาทางจอภาพ ชุดคำสั่งในภาษาออโตลิสป์ เรียกว่า ลิสป์ รูทีน (LISP Routines) และสามารถจัดเก็บ ลิสป์ รูทีน ได้ในรูปของแฟ้มรหัสแอสกี (ASCII Files) เรียกว่า แฟ้มข้อมูลลิสป์ (LISP Files) แฟ้มข้อมูลลิสป์จะมีชนิดเป็น " . lsp"

4.2.1 ขั้นตอนการสร้าง ลิสป์ รูทีน

1. เขียนรหัสเทียม (Pseudo codes) ขั้นตอนการทำงานของโปรแกรม
2. เขียนคำสั่งภาษาออโตลิสป์ตามรหัสเทียมที่ได้ร่างไว้
3. ทดสอบการทำงานของคำสั่งออโตลิสป์แต่ละคำสั่ง ว่าสามารถทำงานได้ถูกต้องหรือไม่ โดยการพิมพ์คำสั่งเหล่านั้นทางแป้นพิมพ์ที่บรรทัดคำสั่ง ทีละคำสั่ง ออโตลิสป์จะแปลคำสั่งนั้นและจะให้ผลลัพธ์แสดงออกมา ถ้าคำสั่งไม่ถูกต้อง ออโตลิสป์จะแสดงข้อความผิดพลาดให้เห็น
4. ถ้าโปรแกรมถูกต้องสามารถทำงานได้ ให้สร้างเป็นแฟ้มข้อมูลลิสป์เก็บไว้ เพื่อสามารถเรียกใช้งานได้ในภายหลัง

4.2.2 โครงสร้างวากยสัมพันธ์ของออโตลิสป์ (AutoLISP syntax structure)

4.2.2.1 **นิพจน์ (Expression)**

รูปแบบคำสั่งพื้นฐานในออโตลิสป์เรียกว่านิพจน์ นิพจน์จะดำเนินการประเมินค่าของข้อมูลในนิพจน์ และให้ผลลัพธ์กลับออกมา

ข้อกำหนดในการเขียนนิพจน์ของออโตลิสป์

1. นิพจน์ของออโตลิสป์จะต้องเขียนอยู่ในระหว่างเครื่องหมายวงเล็บเปิด (และวงเล็บปิด) เสมอ และจำนวนของเครื่องหมายวงเล็บเปิด และวงเล็บปิดจะต้องเท่ากัน

2. นิพจน์จะถูกอ่านและทำงานจากซ้ายไปขวา

3. ภายในนิพจน์แต่ละนิพจน์ประกอบด้วย ตัวดำเนินการ (operator) หรือเรียกอีกอย่างได้ว่า ฟังก์ชัน (function) และอาร์กิวเมนต์ (arguments) โดยที่ตัวดำเนินการจะอยู่ก่อนอาร์กิวเมนต์ของมัน จำนวนอาร์กิวเมนต์จะมีกี่ตัวจะขึ้นอยู่กับตัวดำเนินการในนิพจน์นั้น

ตัวดำเนินการ หรือ ฟังก์ชัน เป็นคำสั่งที่สั่งการให้ออโตลิสป์ทำงานกับอาร์กิวเมนต์ในนิพจน์นั้นเพื่อให้ได้ผลลัพธ์ตามที่ต้องการ

อาร์กิวเมนต์ เป็นข้อมูลที่ต้องการใช้ในการทำงานของฟังก์ชันนั้น ๆ

4. ตัวดำเนินการ และอาร์กิวเมนต์ จะแยกกันด้วยช่องว่าง (space) อย่างน้อย 1 ช่องว่างเพื่อที่ตัวแปรภาษาออโตลิสป์จะสามารถเข้าใจได้

5. ตัวแปรภาษาออโตลิสป์ไม่สนใจช่องว่างที่ต่อกันยาวๆ หรือ รหัสขึ้นบรรทัดใหม่ (carriage return) ดังนั้นแต่ละนิพจน์จะสามารถเขียนยาวเกินกว่า 1 บรรทัดได้

6. นิพจน์ของออโตลิสป์ ใช้ตัวอักษรมาตราฐานแอสกี และไม่สนใจว่าจะเป็น อักษรตัวเล็ก หรือ อักษรตัวใหญ่ (ไม่เป็น Case sensitive) สามารถใช้ปนกันได้

ตัวอย่างของนิพจน์ เช่น

(distance pt1 pt2)

distance เป็นตัวดำเนินการ หรือฟังก์ชันของนิพจน์

pt1 และ **pt2** เป็นอาร์กิวเมนต์ของนิพจน์ (เป็นค่าพิกัดของจุดในโปรแกรมออโตแคด)

ความหมายของนิพจน์นี้คือ ให้หาระยะทางระหว่างจุด 2 จุด คือ pt1 และ pt2

4.2.2.2 **ตัวแปรในหน่วยความจำ (Memory variables)**

การใช้ตัวแปรเป็นวิธีการพื้นฐานทางคอมพิวเตอร์ที่สามารถจัดการเก็บข้อมูลไว้ในหน่วยความจำ และสามารถเรียกออกมาใช้ได้ในภายหลัง

ตัวแปรในภาษาออโตลิสป์จะมีการทำงานดังนี้

1. พื้นที่หน่วยความจำ (RAM) จะถูกจัดสรรใช้เป็นที่เก็บค่าตัวแปรต่าง ๆ

2. เมื่อสร้างตัวแปร จะต้องตั้งชื่อให้กับตัวแปรนั้น

3. ตัวแปรที่ถูกสร้าง และตั้งชื่อแล้ว จะสามารถรับค่า และเก็บค่าข้อมูลต่าง ๆ ได้โดยการอ้างอิงกับชื่อของตัวแปรนั้น

4. เมื่อตัวแปรมีค่าข้อมูลเก็บอยู่ ฟังก์ชันสามารถเรียกใช้ค่านั้นได้โดยการใช้ชื่อตัวแปร ในขณะที่นิพจน์ถูกประเมินค่า และตัวแปรมีค่าข้อมูลเก็บอยู่ ตัวแปลภาษาออโตลิสป์ จะแทนชื่อของตัวแปรด้วยค่าที่เก็บไว้ในตัวแปรนั้น

ออโตลิสป์ใช้ฟังก์ชันพิเศษ setq ในการสร้างและกำหนดค่าให้กับตัวแปร ฟังก์ชันนี้ต้องการอย่างน้อย 2 อาร์กิวเมนต์ อาร์กิวเมนต์ตัวแรกเป็นชื่อของตัวแปร ส่วนอาร์กิวเมนต์ตัวที่สองจะเป็นค่าที่กำหนดให้กับตัวแปรนั้น ๆ

เช่น (setq x 2)

ตัวแปรที่ไม่มีการเปลี่ยนแปลงค่าเรียกว่า ค่าคงที่ (Constants) เช่นค่าของ pi มีค่าประมาณ 3.1415926 (pi มีค่าเป็นทศนิยมไม่รู้จบ) การใช้ค่า pi ที่เป็นค่าคงที่ จะให้ผลลัพธ์ที่ถูกต้องกว่าการใช้ค่า 3.1415926 เพราะการใช้ค่าตัวเลข 3.1415926 แทนค่า pi จะถูกต้องเพียงทศนิยม 7 ตำแหน่งเท่านั้น ถ้าใช้ในการคำนวณที่ต้องการความละเอียด ผลลัพท์อาจจะคลาดเคลื่อนได้

4.2.2.3 **การซ้อนใน (Nesting)**

ผลลัพธ์ที่ได้จากการประเมินค่าของนิพจน์หนึ่ง สามารถใช้เป็นอาร์กิวเมนต์ของอีกนิพจน์หนึ่งได้ เรียกว่า การซ้อนใน (nesting)

เช่น (* 2.36 (+ 1 1))

จะเห็นว่าผลลัพธ์ของนิพจน์ (+ 1 1) เป็นอาร์กิวเมนต์ของนิพจน์ (* 2.36 (+ 1 1)) ออโตลิสป์จะประเมินค่านิพจน์ในสุดก่อน ตัวแปลภาษาออโตลิสป์ยอมให้ใช้วงเล็บซ้อนกันได้ไม่เกิน 100 ระดับ ลักษณะของการซ้อนในนี้ มักจะพบกันบ่อยในโปรแกรมออโตลิสป์ที่มีความซับซ้อนมาก

4.2.2.4 **ตัวแปรระบบ**

ตัวแปรระบบเป็นตัวแปรที่โปรแกรมออโตแคด ใช้สำหรับเก็บค่าสภาพแวดล้อม (environment) และสภาวะ (state) ต่าง ๆ ของระบบ ออโตลิสป์สามารถจัดการกับตัวแปรระบบเหล่านี้ได้โดยใช้ฟังก์ชันพิเศษ คือฟังก์ชัน setvar และ ฟังก์ชัน getvar

setvar เป็นฟังก์ชันที่ใช้ในการกำหนดค่าให้กับตัวแปรระบบ

getvar เป็นฟังก์ชันที่ใช้ในการอ่านค่าปัจจุบัน (current value) ของตัวแปรระบบ

4.2.2.5 **เรเดียน (Radian)**

การวัดมุมของออโตลิสป์ไม่ใช้ระบบองศา(degree) แต่จะใช้ระบบเรเดียนแทน การแปลงค่าจากองศาเป็นเรเดียนใช้ ฟังก์ชัน (* pi (/ angle 180))

4.2.3 ชนิดของข้อมูลพื้นฐานของออโตลิสป์

ข้อมูลพื้นฐานของออโตลิสป์แบ่งได้ 6 ประเภท ดังนี้

4.2.3.1 ชื่อชิ้นส่วนวัตถุ (Entity names)

4.2.3.2 ตัวบอกชี้แฟ้ม (File descriptors)

4.2.3.3 จำนวนเต็ม และจำนวนจริง (Integer & Real number)

4.2.3.4 รายการ (List)

4.2.3.5 สายอักขระ (Strings)

4.2.3.6 กลุ่มชิ้นส่วนที่เลือก (Selection set)

(รายละเอียดดูในภาคผนวก ค)

4.2.4 การใช้งานโปรแกรมออโตลิสป์ ในโปรแกรมออโตแคด

โปรแกรมออโตลิสป์เป็นแฟ้มที่แยกออกมาจากโปรแกรมออโตแคด ก่อนการใช้งานต้องบรรจุ (load) แฟ้มออโตลิสป์เข้ามาในหน่วยความจำ และจะต้องมีหน่วยความจำเหลืออยู่มากพอที่จะใช้งานด้วย การบรรจุแฟ้มออโตลิสป์ในหน่วยความจำ ใช้ฟังก์ชัน load เช่น

(load "filename")

ฟังก์ชันนี้มีผลทำให้โปรแกรมออโตแคดค้นหาแฟ้มชื่อ "filename.lsp" ในสารบบย่อย (sub directory) ที่กำหนดไว้ ถ้าค้นพบก็จะบรรจุข้อมูลทั้งหมดในแฟ้มนั้นเข้าไปเก็บในหน่วยความจำ และจะทำการแปล และประมวลผลต่อไป

4.2.5 การจัดการหน่วยความจำ

ฟังก์ชัน และรูทีนของโปรแกรมออโตลิสป์เมื่อถูกบรรจุ จะเก็บไว้ในโครงสร้างหน่วยความจำขนาด 12 ไบต์ เรียกว่า "โหนด (node)" โหนดจะอยู่รวมกันเป็นกลุ่มเรียกว่า "เซกเมนต์ (segment)" โดยแต่ละเซกเมนต์จะประกอบด้วย 514 โหนด หรือประมาณ 6,168 ไบต์ ถ้าบรรจุโปรแกรมออโตลิสป์เข้ามาใช้งานมาก ก็จะเหลือพื้นที่หน่วยความจำที่จะใช้งานอื่นน้อยลง ทั้งนี้ขึ้นอยู่กับขนาดหน่วยความจำ(RAM)ที่มีในระบบนั้น ขนาดของแฟ้มงานเขียนแบบ และจำนวนรูทีนภายนอกที่บรรจุเข้ามาใช้งาน พื้นที่หน่วยความจำสำหรับโปรแกรมออโตลิสป์ จะใช้สำหรับเก็บชื่อตัวแปร ฟังก์ชัน สายอักขระ และข้อมูลที่ถูกสร้างขึ้นในระหว่างการประมวลผล

4.3 กรอบสนทนา (Dialog box)

กรอบสนทนา เป็นตัวประสานผู้ใช้แบบกราฟิก (Graphic User Intreface) ที่มีประสิทธิภาพของโปรแกรมออโตแคด ช่วยทำให้ผู้ใช้สามารถทำงานโต้ตอบกับโปรแกรมออโตแคดได้ง่ายขึ้น และนอกจากนี้ผู้ใช้สามารถสร้างออกแบบ และแก้ไขปรับปรุงกรอบสนทนาให้เป็นอย่างไรก็ได้ตามความต้องการ

ในการสร้างกรอบสนทนา อย่างน้อยผู้ใช้ต้องเขียนโปรแกรม 2 โปรแกรมในภาษาที่ต่างกัน โดยผู้ใช้จะสร้างตัวกรอบสนทนา ด้วยโปรแกรมภาษาดีซีแอล (Dialog Control Language : DCL) และสร้างโปรแกรมขับกรอบสนทนา (Dialog box driver) ด้วยภาษาออโตลิสป์ หรือด้วย ADS (AutoCAD Development System) โดยที่โปรแกรมออโตแคด จะควบคุมดูแลการแสดงผลจัดวาง กำหนดขนาดส่วนต่าง ๆ ของ กรอบสนทนาให้เองโดยอัตโนมัติ

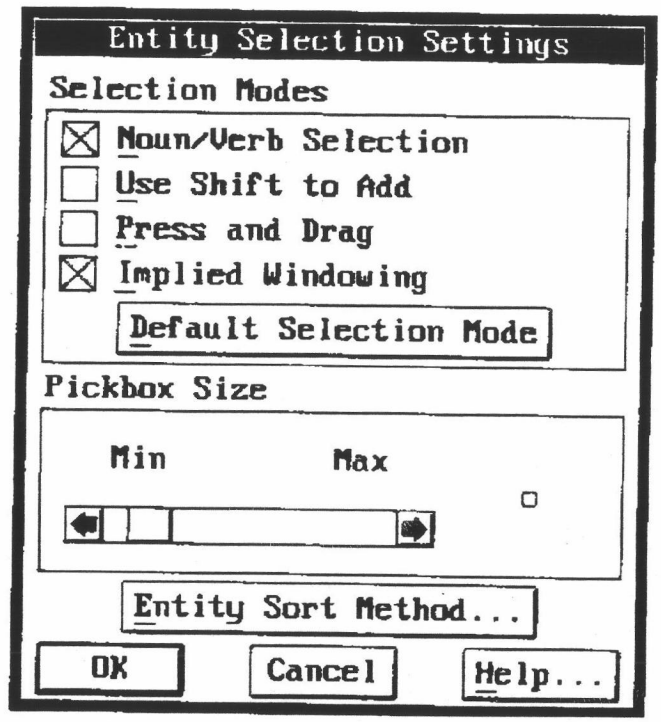
4.3.1 เหตุผลของการใช้กรอบสนทนา

1. เพิ่มประสิทธิภาพการใช้งานโปรแกรม ออโตแคด
2. ผู้ใช้ทั่วไปสามารถเข้าใจและทำงานกับกรอบสนทนา ได้ง่ายเพียงแต่ผู้ใช้เลือกปุ่มทางเลือกที่ต้องการในกรอบสนทนาเท่านั้น การทำงานก็จะดำเนินการโดยโปรแกรมเองโดยอัตโนมัติ
3. การใช้กรอบสนทนาในโปรแกรมทำให้โปรแกรมมีลักษณะเป็นโปรแกรมมืออาชีพ
4. กรอบสนทนาที่พัฒนาขึ้นสามารถใช้กับโปรแกรมออโตแคดที่ทำงานในระบบปฏิบัติการต่างๆ กันได้ ทำให้ลดเวลาการทำงานของผู้พัฒนาลงได้มาก และทำให้ผู้ใช้คุ้นเคยกับการทำงาน ถึงแม้ว่าจะต้องทำงานในระบบปฏิบัติการอื่น ๆ
5. การใช้ กรอบสนทนา จะเพิ่มความเร็วในการทำงาน เพิ่มความยืดหยุ่น (Flexibility) ในการใช้งาน และทำให้ผู้ใช้ ใช้งานโปรแกรมได้อย่างเข้าใจง่ายและสะดวก

4.3.2 แผนภูมิโครงสร้างของกรอบสนทนา

โครงสร้างของกรอบสนทนา จะเกี่ยวโยงกันในลักษณะของต้นไม้ (Tree) ในแต่ละส่วนจะประกอบขึ้นจากชิ้นส่วนย่อยๆ หลายส่วนประกอบกัน ขึ้นอยู่กับการออกแบบกรอบสนทนา และการทำงานของแต่ละส่วนนั้น ส่วนประกอบอิสระของกรอบสนทนา เรียกว่า Predefined active tiles เป็นไทล์ที่โปรแกรมออโตแคด ได้กำหนดรูปแบบไว้ก่อนแล้ว สามารถนำมาใช้ได้ทันที และเมื่อประกอบกันจะเกิดเป็นกรอบสนทนา หรือ กลุ่มของไทล์ (Tile cluster)

ตัวอย่าง ให้พิจารณา กรอบสนทนา ของ Entity selection settings (ดังรูปที่ 4.1) จะพบว่าประกอบไปด้วยไทล์ และกลุ่มของไทล์ต่าง ๆ เรียงลำดับต่อเนื่องกัน ประกอบกันเป็นกรอบสนทนา



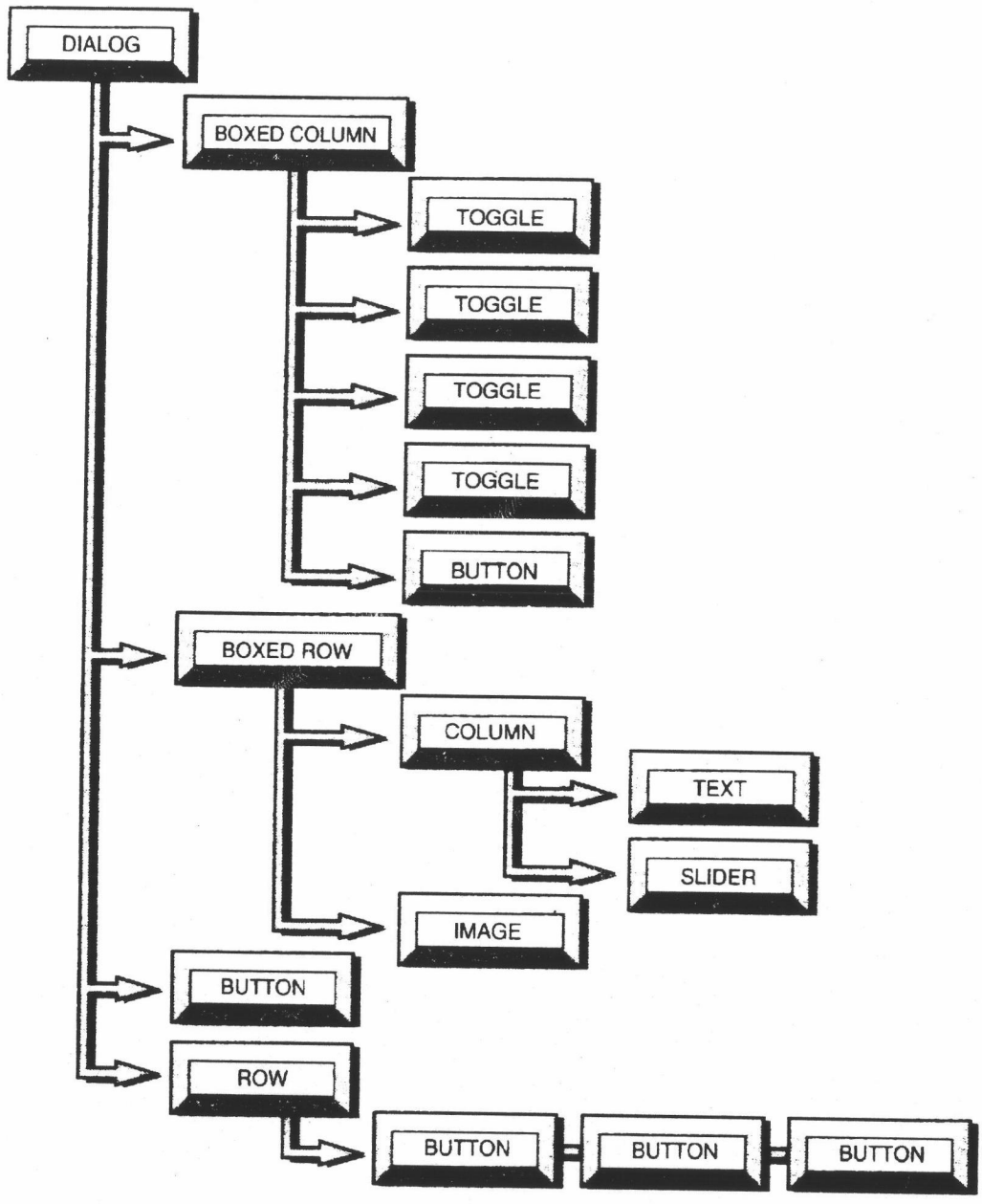
รูปที่ 4.1 Entity selection settings dialog box

โดยที่กลุ่มแรกจะเป็นกลุ่มของไทล์ประเภท Boxed column ของ Selection Modes ซึ่งประกอบด้วย Toggle 4 ไทล์ คือ Noun/Verb Selection ,Use Shift to Add ,Press and Drag และ Implied Windowing และมี Button อีก 1 ไทล์คือ Default Selection Mode เรียงกันลงมาในแนวดิ่งโดยจะอยู่ในเส้นกรอบของ Boxed column

ต่อมาจะเป็นกลุ่มของไทล์ประเภท Boxed row ของ Pickbox Size ที่เรียงไทล์ในแนวนอน โดยแบ่งเป็น 2 ส่วนย่อย ส่วนแรกจะจัดเรียงไทล์ในแนวดิ่ง (Column) บรรทัดแรกได้แก่ Text คือ Min Max และบรรทัดต่อมาเป็น Slider ที่ใช้เลื่อนปรับค่าขนาดของ Pickbox Size ส่วนที่สองจะอยู่ต่อไปทางขวาของส่วนแรก เป็นไทล์ประเภท Image ซึ่งจะแสดงภาพของ Pickbox ที่เปลี่ยนแปลงไปตามการเลื่อนปรับขนาดของ Slider

ต่อมาเป็น Button ของ Entity Sort Method... และในบรรทัดสุดท้ายจะเป็นกลุ่มของ Dialog box exit button ที่เรียงกันอยู่ในแนวนอน ได้แก่ OK Cancel และ Help... ซึ่งใช้ในการออกจากกรอบสนทนา

การจัดเรียงไทล์ของ Entity Selection Settings Dialog Box จะมีโครงสร้างในลักษณะรูปต้นไม้ ดังรูปที่ 4.2



รูปที่ 4.2 แผนภูมิโครงสร้างของ Entity selection settings dialog box

4.3.3 ส่วนประกอบของกรอบสนทนา (COMPONENTS)

ไทล์ต่าง ๆ ที่ใช้กันในกรอบสนทนาแยกได้ 6 ประเภท ดังนี้ (ดูรายละเอียดในภาคผนวก ง.)

ประเภท	ไทล์
Predefined active tiles	Button, dialog, edit box, image button, List box, pop - up list
Tile clusters	Column, boxed column, row, boxed row, radio column, boxed radio column, radio row, boxed radio row
Decorative and informative columns	Image, text, spacer
Text Cluster	Text Part, Concatenation, Paragraph
Dialog box exit button	Single OK button, OK and Cancel, OK Cancel Help, OK Cancel Help Information
Error tiles	Error tile, OK Cancel Help Error Tile

ตารางที่ 4.1 ตารางส่วนประกอบของกรอบสนทนา

4.3.4 ลักษณะประจำ (ATTRIBUTE)

รูปแบบและการทำงานของไทล์ในกรอบสนทนาจะถูกควบคุมโดยลักษณะประจำของไทล์นั้น ค่าข้อมูลในลักษณะประจำของไทล์ มีได้ดังนี้

จำนวนเต็ม (Integer) ใช้กำหนดระยะทางในลักษณะประจำเช่น ความสูง, ความกว้าง

จำนวนจริง (Real) เช่นเดียวกับจำนวนเต็ม ใช้กำหนดระยะทางในลักษณะประจำ ค่าจำนวนจริงจะต้องมีจุดทศนิยม และต้องมีตัวเลขนำหน้าจุดทศนิยม

สายอักขระ (String) ประกอบด้วยตัวอักษร หรือลำดับหลีก (Escape sequence) ที่อยู่ในเครื่องหมายคำพูด “ ” ลำดับหลีกใช้สำหรับใส่ตัวอักษรพิเศษต่าง ๆ ในสายอักขระ เช่น

\" เป็นการใส่เครื่องหมายคำพูด
\\ เป็นการใส่ backslash
\n เป็นการขึ้นบรรทัดใหม่
\t เป็นการเลื่อน tab

คำสงวน (Reserved word) เป็นสายอักขระและตัวเลข ที่ถูกกำหนดไว้ก่อนแล้ว จะต้องเริ่มด้วยตัวอักษร และคำสงวนจะต้องเขียนด้วยตัวอักษรตัวเล็ก

4.3.4.1 **ลักษณะประจำที่ถูกกำหนดไว้ก่อนแล้ว (PREDEFINED ATTRIBUTES)**

เป็นลักษณะประจำที่ถูกกำหนดไว้แล้วและสามารถนำไปใช้ในกรอบสนทนาได้ทันที แบ่งออกได้ตามประเภท ดังนี้ (ดูรายละเอียดในภาคผนวก จ.)

ประเภท	ตัวอย่างลักษณะประจำ
Key and Value attributes	Key, value, list, big-increment, small incerment max_valve, min_valve
Layout sizing and appearance attributes	Alignment, children_alignment, aspect_ratio, color, edit_limit, edit_width, hight, width, fixed_width, fixed_height, children_fixed-height, children_fixed_width, is_bold, label, tabs
Functional Attributes	Action, allow_accept, initial_focus, is_cancel, is_default, is_enabled, is_tab_stop, mnemonic multiple_select

ตารางที่ 4.2 ตารางประเภทของลักษณะประจำที่ถูกกำหนดไว้ก่อนแล้ว

4.3.4.2 **ลักษณะประจำที่กำหนดโดยผู้ใช้ (USER - DEFINED ATTRIBUTES)**

ใช้ในกรณีที่ต้องการกำหนดลักษณะประจำพิเศษให้กับไทล์ โดยจะต้องเป็นไปตามกฎ ดังนี้

1. ชื่อของลักษณะประจำจะต้องเป็นประเภท ตัวอักษร, ตัวเลข และเครื่องหมายขีดเส้นใต้ (Under-score) เท่านั้น
2. ชื่อของลักษณะประจำจะต้องขึ้นต้นด้วยตัวอักษร

4.4 ภาษาดีซีแอล (Dialog Control Language : DCL)

แฟ้มโปรแกรมภาษาดีซีแอล เป็นแฟ้มตัวอักษรรหัสแอสกีที่มีชนิดเป็น .dcl แฟ้มดีซีแอลจะบรรจุคำอธิบายลักษณะของกรอบสนทนา ไทล์ต้นแบบ (prototype tiles) ซึ่งจะมีการกำหนดการทำงาน การแสดงผลของไทล์นั้น และ ส่วนประกอบย่อย (subassemblies) เพื่อใช้อ้างอิงไปยังกรอบสนทนาอื่นได้ แฟ้มดีซีแอลแต่ละแฟ้มจะมีแฟ้มขับ (driver file) ที่ใช้ทำงานคู่กันโดยใช้ชื่อเหมือนกัน และแฟ้มขับ จะถูกเขียนขึ้นด้วยภาษาออโตลิสป์ หรือ ADS

ไทล์ (Tiles) เป็นส่วนประกอบเล็ก ๆ ของ กรอบสนทนา เช่น Buttons Toggles Edit boxes ฯลฯ ซึ่งจะรวมกันประกอบขึ้นเป็นกรอบสนทนา

ลักษณะประจำ(Attributes) เป็นการตั้งค่าควบคุมต่าง ๆ ให้กับไทล์ รวมถึงการแสดงผลทางกายภาพของไทล์ และการทำงานของไทล์นั้น เช่น การตั้งค่าความสูง ความกว้างของไทล์ เป็นต้น

โปรแกรมขับ (Drivers) เป็นโปรแกรมที่ใช้บรรจุแฟ้มดีซีแอล ควบคุม กรอบสนทนา และควบคุมการทำงานของโปรแกรมออโตแคด ให้ขึ้นอยู่กับการกดเลือกทางเลือกของผู้ใช้ในกรอบสนทนา

4.4.1 ประเภทของแฟ้มดีซีแอล

แฟ้มดีซีแอลสามารถประกอบด้วยประเภทของการประกาศค่า (declarations) ที่แตกต่างกันได้ 3 ประเภท คือ การจำกัดความ (definitions), การอ้างอิง (references) และ กรอบสนทนา (dialog boxes)

4.4.1.1 **การจำกัดความ (definitions)** เป็นการจำกัดความของไทล์ต้นแบบ ส่วนประกอบย่อยของบรรทัด (row subassembly) และส่วนประกอบย่อยของแถว (column subassembly)

4.4.1.2 **การอ้างอิง (references)** เป็นการเรียกอ้างอิงไปยังแฟ้มดีซีแอลอื่นๆ เพื่อการเข้าถึงไทล์ต้นฉบับ และส่วนประกอบย่อยที่ประกาศในแฟ้มดีซีแอลนั้น

4.4.1.3 **กรอบสนทนา (dialog boxes)** เป็นผลรวมที่สมบูรณ์ของการประกาศค่าต่างๆ ทั้งไทล์ต้นแบบ, ส่วนประกอบย่อยต่างๆ ฯลฯ

4.4.2 แฟ้มดีซีแอลของออโตแคด (AutoCAD's DCL Files)

ในโปรแกรมออโตแคดจะมีแฟ้มดีซีแอลที่รวมมากับโปรแกรมออโตแคดอยู่หลายแฟ้ม แต่ละแฟ้มจะสร้างกรอบสนทนาของตัวเอง แฟ้มเหล่านี้ถูกเก็บในสารบบย่อยที่ชื่อว่า "support" แฟ้มเหล่านี้เป็นแฟ้มที่มีความสำคัญมาก ตัวอย่างเช่น

4.4.2.1 **ACAD.DCL** เป็นแฟ้มที่เก็บการประกาศค่าต่างๆ ของกรอบสนทนาส่วนใหญ่ของออโตแคด แฟ้มนี้เป็นตัวอย่างที่ดีในการศึกษาการเขียนคำสั่งต่างๆ ในภาษาดีซีแอล

4.4.2.2 **BASE.DCL** เป็นเสมือนคลังเก็บไทล์ต้นฉบับต่างๆ ที่กรอบสนทนาสามารถใช้อ้างอิงได้โดยอัตโนมัติ

4.4.3 วากยสัมพันธ์ของดีซีแอล (DCL syntax)

4.4.3.1 คำจำกัดความไทล์และลักษณะประจำเขียนด้วยอักษรตัวเล็ก ตัวอักษรตัวใหญ่ใช้สำหรับค่าที่กำหนดให้กับลักษณะประจำที่ต้องการ เช่น ค่าที่เป็นตัวอักษร หรือค่าที่เป็นตัวเลข

4.4.3.2 ลักษณะประจำของไทล์ จะถูกจัดกลุ่มรวมกันอยู่ในเครื่องหมายวงเล็บปีกกา

4.4.3.3 การเขียนแฟ้มดีซีแอลจะต้องจัดระดับการย่อหน้า เพื่อแสดงให้เห็นว่าส่วนใดเป็นรายละเอียดของส่วนใด

4.4.3.4 ชื่อของไทล์จะเป็นตัวอักษร ตัวเลข หรือเครื่องหมายขีดเส้นใต้ (underscores) แต่จะต้องขึ้นต้นด้วยตัวอักษรเท่านั้น

4.4.3.5 การกำหนดค่าให้กับลักษณะประจำใช้เครื่องหมายเท่ากับ และจบด้วยเครื่องหมาย ;

4.4.3.6 ภาษาดีซีแอลจะไม่สนใจบรรทัดที่เว้นว่าง

4.4.4 การเขียนอธิบายในแฟ้มดีซีแอล (Documentation)

สามารถเขียนคำอธิบายในแฟ้มดีซีแอลได้ 2 วิธี คือ

4.4.4.1 เขียนคำอธิบายในแต่ละบรรทัด จะใช้เครื่องหมาย // (two forward slashes) นำหน้าคำอธิบายนั้น โปรแกรมจะไม่สนใจข้อความที่อยู่ตั้งแต่หลังใช้เครื่องหมาย // ไปจนจบบรรทัด

4.4.4.2 เขียนคำอธิบายในวิธีการเช่นเดียวกับที่ใช้ในภาษา ซี คำอธิบายจะเขียนอยู่ระหว่างเครื่องหมาย /* และ */ เครื่องหมายทั้งสองนี้จะอยู่คนละบรรทัดก็ได้ ทำให้สามารถเขียนคำอธิบายที่ยาวหลายบรรทัดได้โดยไม่ต้องใช้เครื่องหมาย // นำหน้าทุกบรรทัด

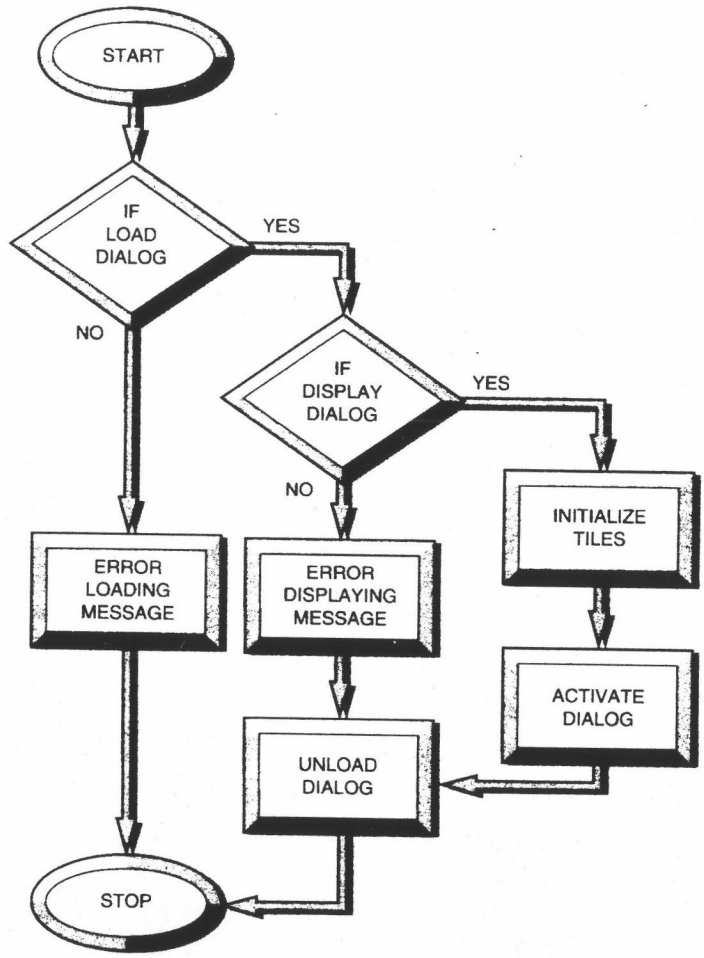
4.4.5 การบรรจุในหน่วยความจำและการตรวจสอบข้อผิดพลาด (Loading and Error Checking)

การบรรจุในหน่วยความจำ และการสั่งการให้กรอบสนทนาทำงานโดยปกติจะเป็นหน้าที่ของโปรแกรมขับกรอบสนทนาที่เขียนในภาษาออโตลิสป์ หรือภาษา ซี อย่างไรก็ตามเมื่อเริ่มสร้างกรอบสนทนา เราสามารถสั่งบรรจุแฟ้มดีซีแอลได้ด้วยตัวเอง เพื่อที่จะตรวจสอบการแสดงผลให้แน่ใจว่าโปรแกรมถูกต้อง ถ้าเกิดข้อผิดพลาดในแฟ้มดีซีแอล โปรแกรมออโตแคดจะสร้างแฟ้ม acad.dce ในสารบบที่กำลังทำงานนั้นเพื่อใช้เก็บข้อความแสดงข้อผิดพลาดที่เกิดขึ้น

4.5 โปรแกรมขับกรอบสนทนาในภาษาออโตลิสป์ (AutoLISP dialog box drivers)

4.5.1 แผนภาพสายงานของโปรแกรมขับกรอบสนทนา

แผนภาพสายงานของโปรแกรมขับกรอบสนทนา จะมีการทำงานเป็นลำดับขั้น (ดังรูปที่ 4.3)



รูปที่ 4.3 แผนภาพสายงานของโปรแกรมขับกรอบสนทนา

4.5.2 ฟังก์ชัน LOAD_DIALOG

เมื่อจะเริ่มทำงานกับกรอบสนทนาต้องบรรจุแฟ้มดีซีแอลเข้ามาก่อน อาจจะเป็นการพิมพ์คำสั่งที่บรรทัดคำสั่งของโปรแกรมออโตแคด หรือจะบรรจุโดยใช้แฟ้มออโตลิสป์ก็ได้ เมื่อแฟ้มดีซีแอลถูกบรรจุเข้ามาในหน่ายความจำ โปรแกรมออโตแคดจะตรวจสอบความถูกต้องของวากยสัมพันธ์ของแฟ้มดีซีแอล

ถ้าการบรรจุไม่สำเร็จจะเกิด Alert box แสดงชื่อของกรอบสนทนา และสาเหตุที่บรรจุไม่สำเร็จ ฟังก์ชัน LOAD_DIALOG จะให้ค่าเป็น NIL และโปรแกรมออโตแคดจะสร้างแฟ้มแสดงความผิดพลาดชื่อว่า acad.dce ในสารบบที่ทำงานอยู่นั้น ซึ่งจะแสดงข้อความผิดพลาดที่ไม่สามารถบรรจุแฟ้มนั้นได้

ถ้าการบรรจุสำเร็จ ฟังก์ชัน LOAD_DIALOG จะให้ค่าเลขจำนวนเต็มตัวหนึ่ง ซึ่งเป็นตัวบ่งชี้ถึงการบรรจุแฟ้มดีซีแอลนั้นๆ เรียกว่า LOAD INDEX ซึ่ง LOAD INDEX จะถูกใช้ในฟังก์ชัน NEW_DIALOG เพื่อแสดงผลกรอบสนทนานั้น และใช้ในฟังก์ชัน UNLOAD_DIALOG เพื่อถอดถอนกรอบสนทนา นั้นออกจากหน่วยความจำ

4.5.3 ฟังก์ชัน NEW_DIALOG

หลังจากบรรจุแฟ้มดีซีแอลแล้วโปรแกรมออโตแคดจะแสดงผลกรอบสนทนานั้นได้ โดยการใช้ฟังก์ชัน NEW_DIALOG รูปแบบของการใช้ฟังก์ชัน NEW_DIALOG

(NEW_DIALOG "DIALOG_BOX_NAME" LOAD_INDEX_NUMBER)

โดยที่ DIALOG_BOX_NAME เป็นชื่อของกรอบสนทนาที่กำหนดไว้ในแฟ้มดีซีแอลนั้น (ชื่อของกรอบสนทนา และชื่อของแฟ้มดีซีแอล ไม่จำเป็นต้องเหมือนกัน) และ LOAD_INDEX_NUMBER เป็นตัวเลขจำนวนเต็ม ที่ได้จากฟังก์ชัน LOAD_DIALOG ฟังก์ชัน NEW_DIALOG จะให้ค่า T ถ้าการแสดงผลกรอบสนทนาสำเร็จ และจะให้ค่า NIL ถ้าการแสดงผลกรอบสนทนาไม่สำเร็จ ควรจะตรวจสอบผลที่ได้จากฟังก์ชัน NEW_DIALOG ว่าแสดงผลได้สำเร็จหรือไม่ ก่อนที่จะดำเนินการขั้นต่อไป เพราะถ้าเรียกใช้ฟังก์ชัน START_DIALOG โดยที่กรอบสนทนานั้นไม่สามารถแสดงผลได้จะทำให้ ระบบติดขัด หรือ เสียหายได้

4.5.4 ขั้นตอนการกำหนดค่าเริ่มต้นให้กับไทล์ (INITIALIZING TILES)

เมื่อกรอบสนทนาปรากฏขึ้น ค่าในไทล์ต่างๆจะเป็นค่าเริ่มต้นที่กำหนดไว้ในแฟ้มดีซีแอลนั้น ซึ่งในบางครั้งจะต้องมีการกำหนดค่าเริ่มต้นให้กับไทล์ต่างๆ ใหม่ จะต้องมีการสร้างรายการสำหรับ List box และ Pop up lists มีการตั้งค่าให้กับ Radio button Toggle และสร้างภาพสำหรับ Image และ Image button ซึ่งทั้งหมดนี้จะอยู่ในขั้นตอนการกำหนดค่าเริ่มต้น ฟังก์ชันที่ใช้กันในขั้นตอนการกำหนดค่าเริ่มต้น แบ่งได้ 2 ประเภท คือ

4.5.4.1 **ฟังก์ชันทั่วไป (General function)** ได้แก่

SET_TILE , MODE_TILE , ACTION_TILE และ DONE_DIALOG

เป็นฟังก์ชันที่สามารถใช้กับไทล์ได้ทุกประเภท

ฟังก์ชัน (SET_TILE KEY VALUE)

เป็นการกำหนดค่าใหม่ให้กับไทล์ที่ต้องการ โดยที่ใช้ KEY เป็นตัวกำหนด

ฟังก์ชัน (ACTION_TILE KEY "AUTOLISP EXPRESSION")

เป็นการกำหนดการทำงานตามนิพจน์ของออโตลิสป์ ให้กับไทล์ที่ต้องการโดยใช้ KEY เป็นตัวอ้างอิง เมื่อไทล์นั้นถูกเลือก จะเกิดการทำงานตาม นิพจน์ของออโตลิสป์ ที่กำหนดไว้

ฟังก์ชัน (DONE_DIALOG STATUS)

เป็นฟังก์ชันที่ใช้ใน ACTION_TILE ใช้สำหรับถอดถอนกรอบสนทนา ออกจากจอภาพคืนการทำงานให้กับโปรแกรมออโตแคดและออโตลิสป์ เพื่อที่สามารถใช้ฟังก์ชันและตัวประสาน (Interface) อื่น ๆ ได้ แต่ ฟังก์ชัน DONE_DIALOG จะไม่ลบแฟ้มดีซีแอลออกจากหน่วยความจำ STATUS จะเป็นค่าที่ส่งไปยังฟังก์ชัน START_DIALOG ซึ่งจะบอกว่ากรอบสนทนา ปิดไปด้วยเหตุใด STATUS จะมีค่าเป็น 1 เมื่อมีการกดปุ่ม OK และจะมีค่าเป็น 0 เมื่อมีการกดปุ่ม CANCEL

ฟังก์ชัน (GET_TILE KEY)

ให้ค่าปัจจุบันของไทล์ที่อ้างอิงโดย KEY ในระหว่างการตั้งค่าเริ่มต้น ค่าปัจจุบันของไทล์จะเท่ากับค่าที่กำหนดไว้ในแฟ้มดีซีแอล

ฟังก์ชัน (GET_ATTR KEY ATTRIBUTE)

ให้ค่าปัจจุบันของลักษณะประจำ (Attribute) ที่ต้องการจากไทล์ที่อ้างอิงโดย KEY

4.5.4.2 **ฟังก์ชันเฉพาะสำหรับไทล์บางไทล์ (Tile specific function)** ได้แก่

START_LIST, ADD_LIST, END_LIST, START_IMAGE, DIMX_TILE, DIMY_TILE, FILL_IMAGE,SLIDE_IMAGE,VECTOR_IMAGE,END_IMAGE,CLIENT_DATA_TILE

เป็นฟังก์ชันที่ใช้เฉพาะสำหรับบางไทล์ เพื่อตั้งค่าเริ่มต้นหรือปรับปรุงค่าในไทล์นั้น ๆ

ฟังก์ชัน (START_LIST KEY OPERATION_INDEX)

ใช้สร้าง, แก้ไข, เพิ่มเติมข้อมูลใน List box หรือ Pop up list ที่กำหนดโดย KEY

ฟังก์ชัน (ADD_LIST ITEM)

ใช้เพิ่มข้อมูลในรายการที่เปิดใช้โดย START_LIST

ฟังก์ชัน (END_LIST)

ใช้ปิดรายการที่เปิดไว้ด้วยฟังก์ชัน START_LIST เพื่อป้องกันการเปลี่ยนแปลงภายหลัง และทำให้รายการนั้นสามารถนำไปใช้งานได้

ฟังก์ชัน (DIMX_TILE KEY)

ให้ค่าตัวเลข (ความกว้างที่สูงสุดของไทล์ ที่กำหนดโดย KEY) -1 โดยวัดตามแกน Xเป็นการวัดขนาดของไทล์ ตามแนวนอน

ฟังก์ชัน (DIMY_TILE KEY)

ให้ค่าตัวเลข (ความสูงที่สูงที่สุดของไทล์ ที่กำหนดโดย KEY) -1 โดยวัดตามแกน Y เป็นการวัดขนาดของไทล์ ตามแนวแกนตั้ง

ฟังก์ชัน (START_IMAGE KEY)

ใช้เปิดการใช้งานของ Image หรือ Image button สำหรับสร้างภาพในไทล์นั้น ที่กำหนดโดย KEY

ฟังก์ชัน (FILL_IMAGE X1 Y1 X2 Y2 COLOR)

แสดงผลเป็นแถบสีรูปสี่เหลี่ยมผืนผ้าใน Image ที่เปิดใช้โดยฟังก์ชัน START_IMAGE ในตำแหน่งที่กำหนดโดยพิกัด X1 Y1 และ X2 Y2

ฟังก์ชัน (SLIDE_IMAGE X1 Y1 X2 Y2 SLIDE_NAME)

แสดงภาพ Slide ลงในไทล์ในตำแน่งที่กำหนดโดยพิกัด X1 Y1 และ X2 Y2

ฟังก์ชัน (VECTOR_IMAGE X1 Y1 X2 Y2 COLOR)

เขียน Vector ลงในไทล์ในตำแน่งที่กำหนด (โดยใช้พิกัด X1 Y1 X2 Y2 ที่กำหนด) ที่เปิดใช้ด้วยฟังก์ชัน START_IMAGE

ฟังก์ชัน (END_IMAGE)

ปิด Image ที่เปิดด้วยฟังก์ชัน START_IMAGE เป็นการหยุดการแก้ไข Image และทำให้ไทล์พร้อมใช้งานได้

ฟังก์ชัน (CLIENT_DATA_TILE KEY CLIENT_DATA)

เชื่อมข้อความที่กำหนดโดยฟังก์ชัน CLIENT_DATA เข้ากับไทล์ที่กำหนดโดย KEY ซึ่งทำได้เฉพาะในช่วงการตั้งค่าเริ่มต้น ข้อความนี้มีลักษณะคล้ายกับ Value attribute สามารถเรียกคืนได้ด้วย $DATA

4.5.5 ฟังก์ชัน START_DIALOG

หลังจากที่กรอบสนทนาปรากฏและไทล์ต่างๆได้ถูกตั้งค่าเริ่มต้นแล้ว จะต้องกระตุ้นให้กรอบสนทนาทำงาน เมื่อกรอบสนทนาทำงานแล้ว เส้นตัด(Cross hair) จะเปลี่ยนไปเป็นลูกศร (Pointer) และจะยังคงทำงานอยู่จนกว่าจะมีการเรียกใช้ฟังก์ชัน DONE_DIALOG ไทล์ต่างๆ สามารถถูกเลือกใช้งานได้ และเมื่อเลือกใช้งานไทล์ใด ฟังก์ชัน(ที่กำหนดโดย ACTION_TILE ให้กับไทล์นั้น) จะทำงานโดยอัตโนมัติ ดังนั้นฟังก์ชันทำงานต่างๆ จะต้องถูกกำหนดก่อนที่จะมีการเรียกใช้ฟังก์ชัน START_DIALOG

4.5.6 ฟังก์ชัน UNLOAD_DIALOG

การทำงานสุดท้ายของโปรแกรมควบคุมกรอบสนทนา คือ การถอดถอนกรอบสนทนา ออกจากหน่วยความจำ ฟังก์ชัน UNLOAD_DIALOG จะคืนหน่วยความจำที่ใช้โดยกรอบสนทนาแก่ระบบ ขั้นตอนนี้จะเกิดขึ้นเมื่อผู้ใช้เลือกไทล์ ที่เรียกใช้ฟังก์ชัน DONE_DIALOG

4.5.7 การเรียกกลับ (CALL BACK)

โดยปกติกรอบสนทนาจะทำงานกับตัวแปร 2 กลุ่ม กลุ่มแรกเป็นตัวแปรเฉพาะที่ (Local variable)ที่ใช้ในกรอบสนทนาและจะถูกปรับค่าเมื่อมีการเรียกกลับเกิดขึ้น ส่วนอีกกลุ่มเป็นตัวแปรส่วนกลาง (Global variable) ที่ผู้ใช้ต้องการกำหนดค่าโดยผ่านกรอบสนทนา ตัวแปรกลุ่มที่สองเป็นตัวแปรสำหรับภายนอกกรอบสนทนา และควรจะถูกปรับแก้ค่าภายหลังเมื่อมีการเลือกปุ่ม OK

ผู้ใช้ไม่ควรจะปรับค่าตัวแปรส่วนกลางในขณะที่ไทล์ต่าง ๆ ถูกเลือกใช้งาน จะต้องรอจนกว่าผู้ใช้กดปุ่ม OK ก่อน ถึงจะปรับค่าตัวแปรส่วนกลางได้ ซึ่งทำให้ผู้ใช้สามารถยกเลิก (Cancel) กรอบสนทนาได้โดยไม่มีการเปลี่ยนแปลงค่าใด ๆ

ขั้นตอนแรกของเรียกกลับเกิดขึ้นเมื่อผู้ใช้กดเลือกไทล์ การเลือกไทล์ จะทำให้การทำงานบางอย่างเกิดขึ้น การทำงานที่เกิดขึ้นแบ่งได้ 3 ประเภท ตามความสำคัญมากไปน้อยดังนี้

- **การทำงานที่กำหนดไว้ใน ACTION_TILE** ในโปรแกรมขับกรอบสนทนาในแต่ละแฟ้ม
- **การทำงานที่กำหนดไว้ใน ACTION ATTRIBUTE** ในแฟ้มดีซีแอล ซึ่งการทำงานนี้จะเกิดขึ้นได้ก็ต่อเมื่อไม่ได้มีการกำหนด ACTION_TILE ในโปรแกรมขับกรอบสนทนา
- **การทำงานโดยปริยาย** สามารถกำหนดได้โดยฟังก์ชัน NEW_DIALOG ซึ่งจะใช้ได้กับปุ่ม ทุกปุ่มที่ไม่ได้มีการกำหนดการทำงานไว้ใน 2 ประเภทแรก

4.5.7.1 ฟังก์ชัน ACTION_TILE

จะทำงานเมื่อไทล์นั้นถูกกดเลือก โดยฟังก์ชัน ACTION_TILE จะไปเรียกใช้ฟังก์ชันที่ผู้ใช้กำหนดไว้มาทำงาน โดยใช้ตัวแปรเหล่านี้ในการผ่านค่าเข้าไปในฟังก์ชันของการเรียกกลับ

$KEY ตัวแปรนี้จะเก็บค่า KEY ATTRIBUTE ของ ไทล์ ที่ถูกเลือก ตัวแปรนี้จะมีประโยชน์ในกรณีที่ใช้ CALL BACK FUNCTION กับหลายๆ ไทล์

$VALUE ตัวแปรนี้จะเก็บค่าปัจจุบันของไทล์ที่ถูกเลือก ตัวอักษรที่อยู่ใน SVALUE จะแปรเปลี่ยนไปตามประเภทของไทล์นั้นๆ ดังนี้

- สำหรับ Radio button และ Toggle tile จะมีค่าเป็น 1 และ 0
- ใน Edit box และ Text tile จะมีค่าเป็น ข้อความ
- ใน Pop up list และ List box จะมีค่าเป็นดัชนีบอกบรรทัด
- ใน List ที่กดเลือก และใน Slider จะมีค่าเป็น ตัวเลข

$DATA จะเก็บข้อมูลเฉพาะของ Application ที่ถูกกำหนดให้กับไทล์ในขั้นตอน Initialization

$REASON ตัวแปรนี้จะเก็บเหตุผลที่การทำงานเกิดขึ้น เป็นสิ่งสำคัญในการทำงานกับ Edit box List box Image button และ Slider ใน $REASON อาจจะมีค่าต่างได้ ดังนี้

- มีค่าเป็น 1 ในกรณีที่เมื่อผู้ใช้เลือกไทล์ทำให้ทราบว่าไทล์นั้นอยู่ในระหว่างการใช้งาน
- มีค่าเป็น 2 ในกรณีที่เมื่อผู้ใช้ออกจาก Edit box โดยไม่จบการเลือก เช่น ผู้ใช้กด TAB Mnemonic หรือกดเลือกที่ไทล์อื่นๆ อาจจะมีค่าค้างอยู่ใน Edit box จังหวะนี้เป็นช่วงเวลาที่ดีที่จะตรวจสอบค่าใน Edit box ว่า เหมาะสมตรงกับประเภทที่ต้องการหรือไม่ อย่าเพิ่งกำหนดค่าใดๆ ให้กับตัวแปรส่วนกลาง ค่าที่กำหนดอาจเปลี่ยนแปลงได้อีก เนื่องจากกรอบสนทนา ยังคงทำงานอยู่
- มีค่าเป็น 3 ในกรณีที่ผู้ใช้เปลี่ยนค่าของ Slider โดยไม่ได้จบการเลือก ในจุดนี้ผู้ใช้ควรจะปรับปรุงไทล์ต่างๆ ที่แสดงผลค่าของ Slider อย่าเพิ่งกำหนดค่าให้กับตัวแปรส่วนกลาง

- มีค่าเป็น 4 ในกรณีที่ผู้ใช้ Double click ที่ไทล์ เช่น List box และ Image button ถ้าเป็น List box การ Double click อาจจะหมายความว่า การเลือกจบแล้ว และกรอบสนทนาสามารถปิดได้ หรืออาจจะหมายความว่าได้ทำการเลือกแล้ว และให้ไปปรับปรุงค่าต่างๆในไทล์อื่นที่เกี่ยวข้องก็ได้ และถ้าเป็น Image button โดยปกติการ Single click จะเป็นการเน้นส่วนที่ต้องการ และการ Double click จะใช้สำหรับเก็บค่าการเลือก และออกจากกรอบสนทนา

$X โดยให้ค่าเป็นพิกัด X ที่เลือกใน Image tile เพราะว่า Image tile แต่ละอันจะมีค่าพิกัดภายในเฉพาะตัว ค่า X จะอยู่ในช่วงระหว่าง 0 ถึง DIMX_TILE

$Y โดยให้ค่าเป็นพิกัด Y ที่เลือกใน Image tile ค่า Y จะอยู่ในช่วงระหว่าง 0 ถึง DIMY_TILE

4.5.7.2 ACCEPT KEY

หนึ่งในการเรียกกลับที่สำคัญมากที่ต้องกำหนดไว้ก็คือ Accept key โปรแกรมขับกรอบสนทนาจะกำหนดค่าตัวแปรเฉพาะที่และปรับปรุงค่านั้นเมื่อมีการเปลี่ยนแปลงในกรอบสนทนา เมื่อ Accept key (ปกติจะเป็นปุ่ม OK) ถูกกดเลือก จะต้องมีการปรับปรุงค่าของตัวแปรส่วนกลางด้วยค่าของตัวแปรเฉพาะที่ กฎ 3 ข้อที่สำคัญในการจัดการการใช้งานของ Accept key มีดังนี้

1. ถ้ามีการสร้างการเรียกกลับสำหรับ Accept key จะต้องเรียกใช้ฟังก์ชัน DONE_DIALOG แต่ถ้าไม่เรียกใช้ฟังก์ชัน DONE_DIALOG จะไม่สามารถออกจากกรอบสนทนาได้เลย
2. เมื่อเรียกใช้ฟังก์ชัน DONE_DIALOG จาก Accept key จะต้องกำหนดสถานะให้กับฟังก์ชัน DONE_DIALOG ให้มีค่าเป็น 1 ไม่เช่นนั้น กรอบสนทนาจะถูกยกเลิก
3. ถ้าไม่ต้องมีการปรับปรุงค่าตัวแปรส่วนกลางในกรอบสนทนา ก็ไม่ต้องกำหนดฟังก์ชัน ACTION_TILE ให้กับ Accept key โปรแกรมออโตแคดจะสร้าง (DONE_DIALOG 1) ให้เองโดยอัตโนมัติเมื่อไทล์ที่เป็น Accept key ถูกกดเลือก

4.5.8 กรอบสนทนาในการเลือกสี (SET COLOR DIALOG BOX)

กรอบสนทนาที่ใช้ในการเลือกสีจะแสดงผลได้ และให้ค่ารหัสสีได้ ก็ต่อเมื่อเรียกใช้ฟังก์ชัน ACAD_COLORDLG โดยที่ฟังก์ชัน ACAD_COLORDLG เป็นส่วนหนึ่งของแฟ้ม ACADAPP.EXP การจะแสดงกรอบสนทนาที่ใช้การเลือกสีใช้คำสั่ง

(ACAD_COLORDLG COLOR_NUMBER)

โดยที่ COLOR_NUMBER จะเป็นค่าโดยปริยายของสีของกรอบสนทนา ค่าสีที่เป็น 0 หมายถึง "BY BLOCK" และค่าสีที่เป็น 256 หมายถึง "BYLAYER " หรือใช้คำสั่ง

(ACAD_COLORDLG COLOR FLAG)

โดยที่ถ้า FLAG มีค่าเป็น NIL ปุ่ม BYBLOCK และ BYLAYER จะไม่สามารถใช้การได้ ถ้า FLAG ไม่ได้เป็น NIL หรือไม่ได้มีการกำหนดไว้ ปุ่ม BYBLOCK และ BYLAYER จะทำงานได้ ค่าที่ส่งออกมาจากฟังก์ชันนี้จะเป็นค่ารหัสของสีที่ได้เลือกไว้ในกรอบสนทนา ภายหลังจากการกดปุ่ม OK

4.5.9 การสร้างและดำเนินการกับรายการ (BUILDING AND DRIVING LIST)

การสร้างและดำเนินการ List box และ Pop up list มี 2 ขั้นตอนคือ การสร้างรายการ และการดำเนินการกับรายการ

4.5.9.1 การสร้างรายการ (BUILDING LIST)

ฟังก์ชัน START_LIST, ADD_LIST และ END_LIST เป็นฟังก์ชันที่ใช้สำหรับทำงานกับรายการ ซึ่งประกอบไปด้วย งาน 4 ประเภท คือ การสร้างรายการ การแก้ไขรายการ การเพิ่มข้อมูลในรายการ และการลบข้อมูลออกจากรายการ งานทั้ง 4 ประเภท ถูกควบคุมโดยอาร์กิวเมนต์ (Operation argument)

4.5.9.1.1 **การสร้างรายการใหม่**

ไม่จำเป็นต้องกำหนดอาร์กิวเมนต์ หรือจะกำหนดอาร์กิวเมนต์ ให้เป็น 3 ก็ได้ โดย

- เรียกใช้ฟังก์ชัน **(START_LIST KEY)** หรือฟังก์ชัน **(START_LIST KEY 3)** โดยที่ KEY เป็น KEY ของ List box หรือ Pop up list ที่จะสร้างนั้น
- เรียกใช้ฟังก์ชัน **(END_LIST)** เพื่อปิดรายการที่เปิดอยู่

4.5.9.1.2 **การเพิ่มเติมรายการ**

เพิ่มบรรทัดใหม่ที่ท้ายรายการ อาร์กิวเมนต์จะมีค่าเป็น 2

- เรียกใช้ฟังก์ชัน **(START_LIST KEY 2)** หรือฟังก์ชัน **(START_LIST KEY 3)** โดย KEY เป็น KEY ของ List box หรือ Pop up list ที่จะเพิ่มเติม
- เรียกใช้ฟังก์ชัน **(ADD_LIST ITEM)** โดยที่ ITEM เป็นตัวอักษรที่เพิ่มเข้าไปในรายการ เรียกใช้ ฟังก์ชัน ADD_LIST ในแต่ละบรรทัดของรายการที่ต้องการเพิ่มเติม
- เรียกใช้ฟังก์ชัน **(END_LIST)** เพื่อปิดรายการที่เปิดอยู่

4.5.9.1.3 **การแก้ไขข้อมูลในรายการ**

เป็นการแก้ไขข้อความในบรรทัดไม่ใช่การแก้ไขตัวเลขลำดับประจำบรรทัดของรายการ อาร์กิวเมนต์จะเป็น 1 โดย

- เรียกใช้ฟังก์ชัน **(START_LIST KEY 1 INDEX)** โดยที่ KEY เป็น KEY ของ List box หรือ Pop up list และ INDEX เป็นตัวเลขประจำของหัวข้อในรายการที่จะแก้ไข โดยตัวเลขจะเริ่มตั้งแต่ 0 ขึ้นไป

- เรียกใช้ฟังก์ชัน **(ADD_LIST ITEM)** โดย ITEM จะเป็นข้อความที่จะนำเข้าไปแทนที่ข้อความเดิมในรายการนั้น

- เรียกใช้ฟังก์ชัน **(END_LIST)** เพื่อปิดรายการที่เปิดอยู่

4.5.9.2 การดำเนินการกับรายการ (Driving a list)

ข้อควรพิจารณาในการเขียนโปรแกรมขับรายการ (List Driver) มีดังนี้

- ในList box สามารถเลือกได้หลายตัวเลือก

- ค่าของ List box หรือ Pop up list จะเป็นตัวเลขดัชนีของบรรทัดที่ได้เลือกไว้ ไม่ใช่ค่าข้อความที่อยู่ในบรรทัดเหล่านั้น

- ค่าดัชนีของ List box มีค่าเป็นตัวอักษร และสามารถมีค่าเป็น NIL (Blank space) ได้

- ฟังก์ชัน ATOI จะให้ค่าเป็น 0 ถ้าตัวอักษรที่จะแปลงนั้นว่างหรือไม่ใช่ตัวเลข

4.5.10 การสร้างรูป (CREATING IMAGES)

เป็นการกำหนดรูป ให้กับ Image tile หรือ Image button ใช้ฟังก์ชันดังนี้

ฟังก์ชัน (START_IMAGE KEY)

ใช้เปิด Image tile ที่กำหนดโดย KEY เพื่อแก้ไข จะต้องเรียกฟังก์ชัน START_IMAGE ก่อนการเรียกใช้ฟังก์ชันของ Image อื่นๆ

ฟังก์ชัน (END_IMAGE)

ปิดไทล์ที่เปิดไว้ด้วยฟังก์ชัน START_IMAGE

ฟังก์ชัน **(FILL_IMAGE X1 Y1 X2 Y2 COLOR)**

เป็นการระบายสีรูปสี่เหลี่ยมผืนผ้าในไทล์ที่ถูกเปิดด้วยฟังก์ชัน START_TILE ค่าพิกัด XY ที่กำหนดจะบอกถึงตำแหน่งจุดมุมสองมุมที่ตรงกันข้ามกันของรูปสี่เหลี่ยมที่จะระบายสี COLOR จะเป็นค่ารหัสสีของโปรแกรมออโตแคดที่ได้กำหนดไว้แล้ว หรืออาจจะมีค่าดังนี้

- มีค่าเป็น -2 คือเป็นสีพื้นฉากหลังของ AutoCAD graphic screen ปัจจุบัน
- มีค่าเป็น -15 คือเป็นสีพื้นฉากหลังของกรอบสนทนาปัจจุบัน
- มีค่าเป็น -16 คือเป็นสีของ Dialog box foreground
- มีค่าเป็น -18 คือเป็นสีของเส้นในกรอบสนทนา

ฟังก์ชัน **(SLIDE_IMAGE X1 Y1 X2 Y2 SLIDE_NAME)**

เป็นการสร้างรูปสไลด์ (Slide) ลงใน Image tile ค่าพิกัด XY ที่กำหนดจะบอกถึงตำแหน่งจุดมุมตรงข้าม 2 จุด ของกรอบสี่เหลี่ยมผืนผ้าในไทล์ SLIDE_NAME เป็นจะชื่อของสไลด์ที่จะบรรจุลงในไทล์ ซึ่งสามารถเป็นแฟ้มสไลด์เอกเทศ หรือจะอยู่ภายในแฟ้มที่เป็นคลังสไลด์ (Slide library file) ก็ได้

ฟังก์ชัน **(VECTOR_IMAGE X1 Y1 X2 Y2 COLOR)**

เป็นการสร้าง Vector ในไทล์ ถูกเปิดด้วยฟังก์ชัน START_TILE ค่าพิกัด XY ที่กำหนดจะบอกถึงตำแหน่งของจุดเริ่มต้นและจุดสิ้นสุดของ Vector COLOR จะมีค่าเป็นรหัสสีที่โปรแกรมออโตแคดกำหนดไว้แล้วหรืออาจจะมีค่าดังนี้

- มีค่า -2 คือเป็นสีพื้นของฉากหลังของ AutoCAD graphic screen ปัจจุบัน
- มีค่า -15 คือเป็นสีพื้นของฉากหลังของกรอบสนทนาปัจจุบัน
- มีค่า -16 คือเป็นสีของ Dialog foreground
- มีค่า -18 คือเป็นสีของเส้นในกรอบสนทนา

4.5.10 IMAGE BUTTON

รูปใน Image buton จะถูกวาดด้วยวิธีเดียวกันกับการสร้าง Image เมื่อ Image button ถูกเลือก โปรแกรมจะสามารถตรวจจับตำแหน่งพิกัด XY ที่เลือกได้โดยใช้ SX และ SY ทำให้สามารถสร้าง Image ได้หลายๆ รูปใน Image button เพียงอันเดียวได้ และสามารถจัดแบ่งส่วนต่างๆของปุ่ม Image button ให้มีการทำงานต่างๆ กันไป

4.5.11 RADIO CLUSTERS

Radio cluster เป็นส่วนของกรอบสนทนา ที่สามารถจัดการควบคุมได้ง่ายถึงแม้ว่า Radio cluster จะแบ่งออกเป็นหลายส่วนก็ตาม ทั้งนี้เพราะโปรแกรมออโตแคดจะดูแลและจัดการควบคุมให้เกือบทั้งหมด เพียงแต่ผู้ใช้จะต้องกำหนดว่าปุ่มใดกดเพื่อจะไปทำงานใด เท่านั้น

4.5.12 แถบเลื่อน (SLIDERS)

การเคลื่อนที่ของแถบเลื่อนขึ้นอยู่กับระบบของเครื่องคอมพิวเตอร์ที่ทำงานนั้นบางระบบสามารถปรับให้มีการเปลี่ยนแปลงในช่วงค่าน้อยหรือมากได้ แต่บางระบบอาจจะปรับค่าของช่วงการเปลี่ยนแปลงของแถบเลื่อนไม่ได้ ค่าของ SREASON จะขึ้นอยู่กับวิธีการที่ผู้ใช้เลือก ถ้า SREASON มีค่าเป็น 1 นั่นคือ ผู้ใช้กดเลือกบนปุ่มเพิ่มค่าหรือลดค่า แต่ถ้า SREASON มีค่าเป็น 3 หมายถึง ผู้ใช้ปรับตำแหน่งที่แถบเลื่อนโดยตรง

แถบเลื่อนมักนิยมใช้คู่กับ Text tile หรือ Edit box เพื่อใช้แสดงค่าปัจจุบันของแถบเลื่อนนั้น เมื่อผู้ใช้ปรับแถบเลื่อนก็ควรจะต้องปรับปรุงค่าในไทล์อื่นๆ ที่เกี่ยวข้องกับแถบเลื่อนนั้นด้วย