บทที่ 4

ทฤษฎีและหลักการของระบบผู้เชี่ยวชาญ

4.1 คำจำกัดความของระบบผู้เชี่ยวชาญ

ระบบผู้เชี่ยวชาญ คือ ระบบของโปรแกรมคอมพิวเตอร์ที่เก็บรวบรวมฐานความรู้ที่ได้จากผู้เชี่ยวชาญมาใส่เป็นฐานความรู้ให้กับระบบและมีกระบวนการอนุมานเพื่อที่จะนำไปสู่เป้าหมายหรือเป้าประสงค์คำตอบของปัญหานั้น ๆ โดยกรรมวิธีในการนำไปสู่คำตอบเปรียบให้เป็นเสมือนกับการทำงานของระบบผู้เชี่ยวชาญ

ดังนั้น ในการที่จะสร้างระบบผู้เชี่ยวชาญขึ้นมาใช้สำหรับการแก้ปัญหาที่สลับซับซ้อนนั้นสิ่งที่ผู้สร้างจะต้องเข้าใจคือกรรมวิธีการทำงานของระบบผู้เชี่ยวชาญเป็นอย่างดี ซึ่งจะเป็นผลดีในการที่จะเก็บรวบรวมเอาความรู้ที่จะนำมาใช้ในกระบวนการอนุมานได้อย่างถูกต้องและสมบูรณ์ที่สุด

จึงพอสรุปได้ว่า ระบบผู้เชี่ยวชาญเป็นโปรแกรมคอมพิวเตอร์ที่ใช้แก้ปัญหาโดยอาศัยความรู้ในสาขาใดสาขาหนึ่งที่รวบรวมความรู้จากผู้เชี่ยวชาญในสาขานั้น ๆ และเก็บไว้เป็นฐานความรู้ในคอมพิวเตอร์ โปรแกรมดังกล่าวต้องเป็นโปรแกรมเพื่อแก้ปัญหาโดยเฉพาะ ความสามารถในการแก้ปัญหาของโปรแกรมจะเทียบเท่าความสามารถของผู้มีความชำนาญระดับผู้เชี่ยวชาญนั้น ๆ

ในการพัฒนาระบบผู้เชี่ยวชาญที่ใช้แก้ปัญหาที่มีความซับซ้อน สิ่งที่ต้องพิจารณา คือ การสร้างความเข้าใจในกรรมวิธีการทำงานของผู้เชี่ยวชาญอย่างลึกซึ้ง เพื่อจะได้ทราบถึงขั้นตอนการทำงานจริงของระบบผู้เชี่ยวชาญ ตัวอย่างลักษณะงานที่เหมาะที่จะสร้างเป็นระบบผู้เชี่ยวชาญมีดังนี้

- เป็นงานที่ใช้เหตุผลและความรู้ในการแก้ปัญหา
- เป็นงานที่มีผู้ชำนาญในเรื่องนั้น ๆ สามารถให้ความรู้ได้
- ความชำนาญที่ใช้ในงานนั้นสามารถถ่ายทอดให้ผู้เริ่มต้นใหม่ได้
- กฎและวิธีการหาเหตุผลในระบบนั้นมีอยู่และสามารถให้คำตอบที่เข้าใจได้
- เงื่อนไขต่าง ๆ ของกฎสามารถกำหนดได้ มีเป้าหมายย่อยที่ไม่ขัดแย้งกัน เป็นต้น

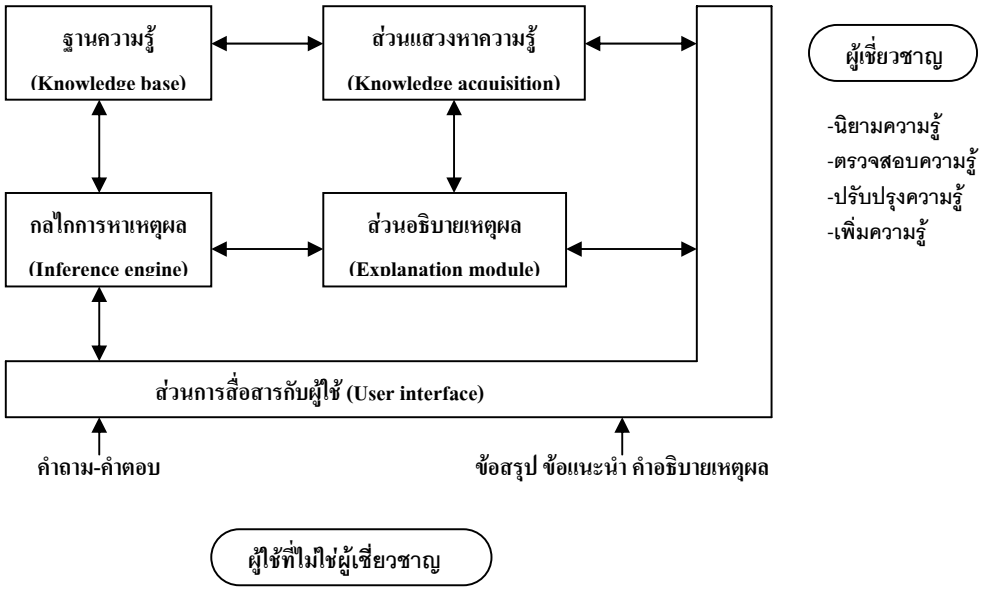
4.2 องค์ประกอบของระบบผู้เชี่ยวชาญ

ระบบผู้เชี่ยวชาญโดยทั่วไปประกอบด้วยส่วนประกอบพื้นฐาน 5 ส่วน รายละเอียดโดยย่อของแต่ละส่วนสามารถอธิบายได้ดังนี้

1. ฐานความรู้ (Knowledge base) เป็นส่วนที่เก็บความรู้ทุกประเภทในสาขาต่าง ๆ ที่ต้องการแก้ปัญหาไว้ในรูปแบบที่คอมพิวเตอร์สามารถเข้าใจและสามารถประมวลผลได้ และจะต้องมีการปรับปรุงความรู้ความรู้นี้ให้มีความทันสมัยอยู่เสมอ โดยปกติแล้วความรู้ที่รวบรวมมาจะมีจำนวนมาก ซึ่งปัญหาที่สำคัญอีกประการหนึ่งในการสร้างฐานความรู้ขึ้นมา คือทำอย่างไรจึงจะแทนความรู้จำนวนมากเหล่านั้นเก็บไว้ในคอมพิวเตอร์โดยไม่สิ้นเปลืองเนื้อที่มากนัก ในขณะเดียวกันก็อยู่ในรูปที่สามารถวิเคราะห์หาเหตุผลจากความรู้นั้นออกมาได้

2. กลไกการหาเหตุผลหรือเครื่องอนุมาน (Inference engine) เป็นส่วนของโปรแกรมที่ใช้ในการคิดหาเหตุผลโดยอาศัยความรู้จากฐานความรู้ข้างต้น เพื่อให้ได้วิธีแก้ปัญหาออกมา วิธีการหาเหตุผลปกติจะสัมพันธ์กับลักษณะการแทนความรู้ที่เลือกใช้ในการสร้างฐานข้อมูล

3. ส่วนการสื่อสารกับผู้ใช้ (User interface module) เป็นส่วนของโปรแกรมที่ใช้สื่อสารระหว่างผู้ใช้กับระบบ เช่น อาจเป็นการตั้งคำถามมายังผู้ใช้เพื่อให้ผู้ใช้ป้อนข้อมูลที่จำเป็นเพื่อนำไปใช้ในการแก้ปัญหาต่อไป หรือผู้ใช้อาจตั้งคำถามให้ระบบอธิบายว่าระบบกำลังทำอะไรอยู่หรือระบบช่วย



ภาพประกอบ 4-1 สถาปัตยกรรมของระบบผู้เชี่ยวชาญ

หาคำตอบให้ ถ้าการสื่อสารระหว่างผู้ใช้กับระบบผู้เชี่ยวชาญทำได้ง่าย ระบบผู้เชี่ยวชาญก็จะมีประโยชน์ต่อผู้ใช้มากขึ้น

4. ส่วนแสวงหาความรู้ (Knowledge acquisition module) การนำความรู้จากผู้เชี่ยวชาญมาสร้างความรู้นั้น จำเป็นต้องมีการรวบรวมความรู้จากผู้เชี่ยวชาญ แล้วแทนความรู้นั้นให้อยู่ในรูปที่คอมพิวเตอร์สามารถประมวลผล คิดค้นหาเหตุผลหรือความรู้จากคำตอบจากความรู้นั้นได้ และรวบรวมให้เป็นฐานความรู้ขึ้นมา กระบวนการเหล่านี้เรียกว่ากระบวนการแสวงหาความรู้ (Knowledge acquisition) ดังนั้นโปรแกรมส่วนนี้คือส่วนที่ช่วยผู้เชี่ยวชาญในการนิยามฐานความรู้ขึ้นมา รวมทั้งช่วยในการปรับปรุงฐานความรู้นั้นให้ทันสมัยอยู่เสมอ

5. ส่วนอธิบายเหตุผล (Explanation module) คือ เมื่อผู้ใช้ปรึกษาระบบผู้ชำนาญการและคำตอบออกมาเพียงคำตอบเดียว ผู้ใช้อาจไม่มั่นใจว่าคำตอบที่ได้นั้นจะนำไปสู่การนำไปใช้ได้จริง การอธิบายเหตุผลให้ผู้ใช้ทราบอาจทำให้ผู้ใช้นำคำตอบที่ได้ไปใช้งานอย่างมั่นใจ โปรแกรมส่วนนี้จึงเป็นส่วนที่จำเป็นที่ใช้อธิบายเหตุผล ให้ผู้ใช้ได้ทราบว่าข้อสรุปของระบบที่เป็นคำตอบออกมาได้อย่างไร

ในระบบผู้เชี่ยวชาญบางระบบอาจมีส่วนประกอบไม่ครบทั้ง 5 ส่วนดังกล่าวข้างต้นก็ได้ แต่ส่วนที่สำคัญและขาดเสียมิได้ คือ ส่วนของฐานความรู้และเครื่องอนุมาน

4.3 กลไกการหาเหตุผล

เครื่องอนุมานเป็นส่วนของโปรแกรมที่ใช้การคิดหาเหตุผล โดยอาศัยความรู้ในฐานความรู้เพื่อให้ได้วิธีการแก้ปัญหาออกมา ดังนั้นเครื่องอนุมานจึงเป็นส่วนที่สำคัญที่สุดในระบบผู้เชี่ยวชาญ ซึ่งจะเป็นตัวกำหนดความสามารถของระบบ กำหนดความเร็ว และความถูกต้องของการแก้ปัญหา

การอนุมานที่ใช้ในระบบผู้เชี่ยวชาญมีอยู่ด้วยกันหลายวิธีขึ้นอยู่ลักษณะของปัญหาและรูปแบบของการแทนความรู้ ระบบผู้เชี่ยวชาญที่มีการทำงานเป็นลำดับขั้นตอนและใช้การแทนความรู้ในรูปของกฎ จะมีวิธีอนุมานอยู่ด้วยกัน 3 วิธี คือ การอนุมานแบบเดินหน้า (forward chaining inference) การอนุมานแบบถอยหลัง (backward chaining inference) และการอนุมานแบบผสม (mix chaining inference) ซึ่งสามารถอธิบายหลักการของการอนุมานแต่ละแบบได้ดังนี้

4.3.1 การอนุมานแบบเดินหน้า

การอนุมานจะเริ่มจากข้อมูลความจริงที่มีอยู่ในฐานข้อมูล โดยจะถูกเปรียบเทียบกับเงื่อนไขของกฎ ซึ่งกฎที่เงื่อนไขตรงกับข้อเท็จจริงที่มีอยู่ก็จะถูกปฏิบัติการตามข้อสรุป กฎต่าง ๆ จะถูกอนุมาน

ในลักษณะนี้ไปเรื่อย ๆ จนกว่าจะได้คำตอบหรือบรรลุเป้าหมาย ซึ่งจะเป็นกระบวนการในการสร้างฐานความรู้ขึ้นมาใหม่

การอนุมานแบบเดินหน้า จะค้นหาทุกเป้าประสงค์ที่เป็นไปได้ แต่ในบางครั้งเป้าประสงค์ที่ได้อาจจะไม่ใช่คำตอบที่ต้องการ ดังนั้นจึงเป็นการเสียเวลามากต่อระบบที่ต้องค้นหาคำตอบของแต่ละปัญหา ภาพประกอบ 4-2 จะแสดงการทำงานของการอนุมานแบบเดินหน้า โดยกฎและฐานความจริงที่กำหนดขึ้นจะเป็นส่วนประกอบของฐานความรู้เริ่มต้น

กำหนดให้มีกฎและความจริงเริ่มต้น
กฎที่ 1 P1 $\rightarrow$ P2
กฎที่ 2 P2 $\rightarrow$ P3
กฎที่ 3 P3 $\rightarrow$ P4
ความจริงเริ่มต้น P1 เป็นจริง
ขั้นที่ 1 ใช้กฎที่ 1 [(P1 $\rightarrow$ P2) $\wedge$(P1 เป็นจริง)] $\rightarrow$ (P2 เป็นจริง)
ขั้นที่ 2 ใช้กฎที่ 2 [(P2 $\rightarrow$ P3) $\wedge$(P2 เป็นจริง)] $\rightarrow$ (P3 เป็นจริง)
ขั้นที่ 3 ใช้กฎที่ 3 [(P3 $\rightarrow$ P4) $\wedge$(P3 เป็นจริง)] $\rightarrow$ (P4 เป็นจริง)

ภาพประกอบ 4-2 ตัวอย่างการอนุมานแบบเดินหน้า

เราสามารถอธิบายกระบวนการอนุมานแบบเดินหน้าได้ดังนี้ ระบบมีฐานความรู้ที่เป็นกฎของระบบอยู่ 3 กฎ คือ กฎที่ 1 : ถ้า P_1 แล้ว P_2 , กฎที่ 2 : ถ้า P_2 แล้ว P_3 , กฎที่ 3 : ถ้า P_3 แล้ว P_4 และมีความจริงเริ่มต้นของระบบ คือ P_1 เริ่มแรก ระบบจะทำการตรวจสอบความจริงเริ่มต้น และพบว่าตรงกับกฎที่ 1 ซึ่งมี P_1 เป็นเงื่อนไข เมื่อตรวจสอบพบว่า P_1 เป็นจริง ดังนั้น ระบบจะทำตามข้อสรุปภายในกฎที่ 1 นั่นคือ P_2 จากนั้น ก็จะใช้งานกฎข้อถัดไป คือ กฎที่ 2 มี P_2 เป็นเงื่อนไข เมื่อตรวจสอบพบว่า P_2 เป็นจริง ดังนั้น ระบบจะทำตามข้อสรุปภายในกฎที่ 2 นั่นคือ P_3 จากนั้น ก็จะใช้งานกฎข้อถัดไป คือ กฎที่ 3 มี P_3 เป็นเงื่อนไข เมื่อตรวจสอบพบว่า P_3 เป็นจริง ดังนั้น ระบบจะทำตามข้อสรุปภายในกฎที่ 3 นั่นคือ P_4 การอนุมานจะเป็นไปในลักษณะเช่นนี้ไปเรื่อย ๆ จนบรรลุเป้าหมายที่ต้องการ

การอนุมานแบบเดินหน้านี้ ถ้าเกิดกรณีที่มีกฎจำนวนหลาย ๆ กฎ มีเงื่อนไขตรงกันกับความจริงในฐานความรู้ ระบบจะมีการขจัดข้อขัดแย้ง (conflict resolution) ซึ่งเกณฑ์การขจัดข้อขัดแย้งมีหลายลักษณะที่สามารถเลือกใช้ได้ ดังนี้

1) เลือกใช้อย่างสุ่ม

2) นำกฎมาเรียงไว้ด้วยกัน กฎใดที่ตรงกับความจริงเป็นกฎแรกจะใช้กฎนั้น

3) กฎใดที่มีเงื่อนไขตรงกับข้อมูลมากที่สุดจะใช้กฎนั้น

4) กฎใดที่ใช้หลังสุดจะใช้กฎนั้น

5) กฎใดที่มีตัวแปรใช้ไปหลังสุดจะใช้กฎนั้น

6) ใช้กฎที่เพิ่มเข้าไปในฐานความรู้หลังสุด

7) ให้น้ำหนักของกฎและความจริง ใช้กฎที่มีน้ำหนักมากที่สุด

4.3.2 การอนุมานแบบย้อนหลัง

การอนุมานแบบย้อนหลัง เป็นการแก้ปัญหาโดยเริ่มต้นจากสถานะเป้าหมายซึ่งเป็นสมมุติฐานที่สนใจไปสู่สถานะเริ่มต้นอย่างมีประสิทธิภาพ หรือเป็นกระบวนการพิสูจน์สมมุติฐาน

การอนุมานแบบย้อนหลัง จะค้นหาเฉพาะเป้าประสงค์ที่ต้องการเท่านั้นจึงใช้เวลาน้อยกว่า แต่ในบางครั้งอาจจะหาเป้าหมายไม่ได้เลย เนื่องจากเงื่อนไขในการนำไปสู่เป้าประสงค์นั้นไม่ถูกต้อง ภาพประกอบ 4-3 จะแสดงการทำงานของการอนุมานแบบย้อนหลัง

กำหนดให้มีกฎและความจริงเริ่มต้น
กฎที่ 1 P1 → P2
กฎที่ 2 P2 → P3
กฎที่ 3 P3 → P4
ความจริงเริ่มต้น P1 เป็นจริง ถ้าเป้าประสงค์ที่ต้องการคือ P4
ขั้นที่ 1 ตรวจสอบกฎที่ต้องนำมาใช้ในการหาเป้าประสงค์นั้น
เป้าประสงค์ คือ P4
กฎที่ต้องนำมาใช้ คือ กฎที่ 3 P3 → P4
กฎที่ต้องนำมาใช้ คือ กฎที่ 2 P2 → P3
กฎที่ต้องนำมาใช้ คือ กฎที่ 1 P1 → P2
สิ้นสุดการตรวจสอบกฎ เนื่องจากส่วนเงื่อนไขของกฎสุดท้ายที่ได้มาเป็นความจริงเริ่มต้น
ขั้นที่ 2 นำกฎที่ได้ไปอนุมานแบบเดินหน้า โดยเริ่มต้นการอนุมานจากกฎที่ได้มาสุดท้ายไปยังกฎที่ได้มาเป็นกฎแรก

ภาพประกอบ 4-2 ตัวอย่างการอนุมานแบบย้อนหลัง

เราสามารถอธิบายกระบวนการอนุมานแบบเดินหลังได้ดังนี้ ระบบมีฐานความรู้ที่เป็นกฎของระบบอยู่ 3 กฎ คือ กฎที่ 1 : ถ้า P_1 แล้ว P_2 , กฎที่ 2 : ถ้า P_2 แล้ว P_3 , กฎที่ 3 : ถ้า P_3 แล้ว P_4 และมีความจริงเริ่มต้นของระบบ คือ P_1 และมีเป้าประสงค์ที่ต้องการคือ P_4 เริ่มแรกระบบจะตรวจสอบว่าเป้าประสงค์ที่เราต้องการนั้น จำเป็นจะต้องใช้กฎข้อไหนบ้างในการเดินทางจากเป้าประสงค์กลับไปยังความจริงเริ่มต้น นั่นคือ ระบบจะหาเส้นทางเดินจาก P_4 กลับไปยัง P_1จะพบว่า เส้นทางในการเดินทางมีดังนี้ เริ่มแรกจากเป้าประสงค์ P_4ต้องเรียกใช้กฎข้อ 3 นั่นคือ ถ้า P_3 แล้ว P_4 ต่อจากนั้นจะเรียกใช้กฎข้อ 2 นั่นคือ ถ้า P_2 แล้ว P_3 ระบบจะหาเส้นทางเดินต่อไป คือ ใช้กฎข้อที่ 1 นั่นคือ ถ้า P_1 แล้ว P_2 สุดท้ายจากเป้าประสงค์ที่ต้องการP_4 ก็จะสามารถหาเส้นทางเดินกลับไปยังความจริงเริ่มต้น P_1 ได้นั่นเอง

4.3.3 การอนุมานแบบผสม

การอนุมานแบบผสมจะเป็นลักษณะการอนุมานที่รวมกันระหว่างการอนุมานแบบเดินหน้าและการอนุมานแบบถอยหลัง โดยจะเริ่มการอนุมานแบบถอยหลังก่อน เมื่อการอนุมานนั้นสำเร็จ มีเป้าประสงค์เป็นจริง ระบบจะนำเส้นทางของการอนุมานนั้นไปทำการอนุมานแบบเดินหน้าต่อไปเพื่อให้ได้ความจริงใหม่ตามเส้นทางของการอนุมานนั้นๆ จนกระทั่งได้เป้าหมายออกมา

4.4 การพัฒนาระบบผู้เชี่ยวชาญ

ดังที่ได้กล่าวมาแล้วว่า ระบบผู้เชี่ยวชาญเป็นระบบที่ใช้แก้ปัญหาที่มีซับซ้อน จึงจำเป็นต้องใช้ความรู้ความชำนาญระดับสูง ดังนั้นการพัฒนาระบบผู้เชี่ยวชาญจึงมีความหลากหลายด้วยกัน ซึ่งได้แก่

การตีความ (interpretation)

การตีความ คือ การวิเคราะห์ข้อมูลเพื่อหาความหมายที่ได้จากข้อมูลนั้น เช่น ข้อมูลจาก mass spectrometer และพยายามตีความจากข้อมูลว่าโครงสร้างทางเคมีคืออะไร งานลักษณะนี้จะต้องตีความให้ถูกต้อง โดยปกติมักจะตีความที่เป็นไปได้ทั้งหมดก่อน แล้วจึงตัดข้อใดข้อหนึ่งทิ้งเมื่อมีเหตุผลเพียงพอว่าการตีความแบบนั้นผิด ปัญหาของงานในลักษณะนี้คือ ข้อมูลมักมีความคาดเคลื่อนปนอยู่ด้วย อาจกล่าวได้ว่าผู้ตีความมีข้อเท็จจริงอยู่เพียงบางส่วน ถ้าข้อมูลที่ได้มาเชื่อถือไม่ได้ การตีความก็จะเชื่อถือไม่ได้ไปด้วยจึงมีความจำเป็นที่ผู้ตีความจะต้องจำแนกให้ได้ว่าข้อมูลส่วนไหนเชื่อถือไม่ได้หรือไม่สมบูรณ์

การวินิจฉัย (diagnosis)

การวินิจฉัย คือ การตรวจสอบว่ามีอะไรผิดพลาดในระบบ เช่น การวินิจฉัยโรคติดเชื้อ งานลักษณะนี้ผู้วินิจฉัยจำเป็นต้องเข้าใจว่าระบบประกอบขึ้นมาได้อย่างไร แต่ละส่วนย่อย ๆ ในระบบสัมพันธ์กันอย่างไร งานลักษณะนี้มีความยากตรงที่ความผิดพลาดอย่างหนึ่งอาจมีอาการซึ่งเป็นสาเหตุของความผิดพลาดอีกอย่างหนึ่งปิดบังอยู่

การควบคุมการทำงาน (monitoring)

การควบคุมการทำงาน คือ การตีความข้อมูลที่ส่งออกมาจากระบบว่ามีอะไรผิดพลาดหรือไม่ และต้องแจ้งให้ผู้ทำงานทราบทันทีที่เกิดความผิดพลาดขึ้น เช่น การควบคุมการทำงานของเครื่องช่วยหายใจของคนไข้ผ่าตัด งานลักษณะนี้จะเห็นว่าจะรวมเอางานวินิจฉัยเอาไว้ด้วย เพราะจำเป็นต้องวินิจฉัยว่าความผิดพลาดในระบบคืออะไร และต้องแจ้งสาเหตุแห่งความผิดพลาดให้ผู้ใช้เครื่องมือทราบทันที

การทำนาย (prediction)

การทำนาย คือ การคาดเดาพฤติกรรมในอนาคตจากตัวแบบซึ่งได้จากอดีตและปัจจุบัน เช่น การทำนายผลกระทบเนื่องจากการเปลี่ยนแปลงนโยบายทางเศรษฐกิจ การทำนายเป็นการหาเหตุผลที่เกี่ยวเนื่องกับเวลา ดังนั้นผู้ทำนายจะต้องสามารถอ้างอิงสิ่งต่าง ๆ ที่เปลี่ยนแปลงไปตามเวลาได้ การทำนายจำเป็นต้องอาศัยความรู้ที่มีอยู่ซึ่งไม่สมบูรณ์ ดังนั้นในการทำนายจึงควรบอกทางที่เป็นไปได้ไว้หลาย ๆ ทาง พร้อมทั้งบอกด้วยว่า ถ้าข้อมูลมีการเปลี่ยนแปลงจะส่งผลให้อะไรเปลี่ยนแปลงได้บ้าง

การวางแผน (planning)

การวางแผน คือ การกำหนดลำดับของการกระทำที่สามารถนำไปสู่เป้าหมายได้ คนที่วางแผนจะต้องสร้างแผนขึ้นมาในลักษณะที่สามารถนำไปสู่เป้าหมายได้โดยไม่ใช้ทรัพยากรมากเกินไป และไม่ขัดแย้งกับข้อจำกัดต่าง ๆ ถ้าเป้าหมายขัดแย้งกัน ผู้วางแผนต้องสามารถจัดลำดับความสำคัญได้ ถ้าข้อมูลหรือข้อจำกัดเกี่ยวกับแผนมีการเปลี่ยนแปลงหรือไม่สมบูรณ์ ผู้วางจะต้องสามารถปรับเปลี่ยนได้ ปกติแล้วการวางแผนจะมีบางส่วนที่จะต้องใช้การทำนายเข้ามาช่วย ปัญหาในการวางแผน คือ ระบบที่ต้องวางแผนมักใหญ่และซับซ้อน ผู้วางแผนอาจเข้าใจปัญหาในทันทีไม่ได้ จำเป็นต้องใช้เวลาใน

การศึกษาอย่างละเอียดรอบคอบ ถ้ามีรายละเอียดมาก อาจต้องแบ่งเป็นปัญหาย่อย ๆ แล้วหาความสัมพันธ์หรือเชื่อมโยงปัญหาเหล่านั้นได้ ข้อมูลที่ใช้ในการวางแผนมักมีความไม่แน่นอนรวมอยู่ด้วย ดังนั้นผู้วางแผนควรกำหนดไว้ว่าแผนที่วางไว้นั้น โอกาสที่ต้องใช้เป็นเท่าไร

การออกแบบ (designing)

การออกแบบ คือการกำหนดรายละเอียด (specification) ของสิ่งใดสิ่งหนึ่งให้มีคุณสมบัติตามที่ต้องการ เช่น การออกแบบวงจรดิจิตอล ลักษณะงานของการออกแบบจะคล้ายกับการวางแผน

การพัฒนาระบบผู้เชี่ยวชาญจะขึ้นอยู่กับลักษณะของปัญหาในลักษณะใดลักษณะหนึ่งที่ได้กล่าวมาแล้ว ปัจจัยสำคัญที่จะทำให้การพัฒนาระบบผู้เชี่ยวชาญประสบความสำเร็จหรือไม่ คือ การเลือกงานที่จะนำมาสร้างเป็นระบบผู้เชี่ยวชาญ

4.5 ภาษาสำหรับระบบผู้เชี่ยวชาญ

ภาษาคอมพิวเตอร์ที่ใช้กันอยู่ในปัจจุบันส่วนใหญ่เป็นภาษาที่จะต้องกำหนดลำดับขั้นตอนในการทำงานที่แน่นอน ซึ่งเรียกว่า procedural language ผู้เขียนโปรแกรมจะกำหนดลำดับขั้นตอน (algorithm) สำหรับการแก้ปัญหานั้นเรียบร้อยแล้ว สำหรับระบบผู้เชี่ยวชาญต้องอาศัยการตัดสินใจทางตรรกวิทยา และทำการค้นหาคำตอบซึ่งมีรูปแบบและขั้นตอนที่ไม่แน่นอน ดังนั้นภาษาประเภท procedural language จึงไม่สามารถที่จะนำมาใช้ได้ หรือในบางกรณีก็อาจจะใช้ได้ในขอบเขตจำกัด แต่ต้องอาศัยการสร้างโปรแกรมที่ซับซ้อนมาก (วริทธิ์ อึ้งภากรณ์ ,2531)

ต่อไปนี้จึงจะขอยกตัวอย่างภาษาคอมพิวเตอร์ที่ใช้สำหรับแก้ปัญหาทางปัญญาประดิษฐ์ซึ่งนำมาใช้พัฒนาระบบผู้เชี่ยวชาญ

ภาษา IPL

ภาษา IPL ย่อมาจากคำว่า International Processing Language เป็นภาษาที่พัฒนาสำหรับการแก้ปัญหาทางปัญญาประดิษฐ์เป็นภาษาแรก ถูกพัฒนาในปี ค.ศ. 1956 โดย Allen Newell, J.C.Shaw และ H.Simon แห่งมหาวิทยาลัย Carnegie ได้พัฒนาภาษานี้ขึ้นมาหลายรุ่นด้วยกัน โดยมีภาษา IPL-V เป็นรุ่นสุดท้าย จุดประสงค์หลักของภาษา IPL คือ การทำงานกับลิสต์ แต่เนื่องจากภาษานี้คล้ายกับภาษาเครื่อง (machine language) จึงใช้งานยากและเลิกใช้งานในเวลาต่อมา

ภาษา LISP

ภาษา LISP ย่อมาจากคำว่า LIST Processing เป็นภาษาทางคอมพิวเตอร์ชั้นสูงภาษาหนึ่งที่ใช้ได้ดีในการประมวลผลทางสัญลักษณ์ (symbol) จึงเป็นภาษาที่ใช้กันอย่างแพร่หลายในการพัฒนาโปรแกรมด้านปัญญาประดิษฐ์ (artificial intelligence:AI) ถูกพัฒนาโดย J.McCarthy แห่งสถาบัน MIT ในปี ค.ศ.1950 โดยลักษณะพิเศษของภาษานี้คือ เป็นภาษาที่มีการตอบรับทันทีที่โปรแกรมถูกป้อนเข้าไป และจะมีการตรวจสอบไวยากรณ์ของภาษาด้วยตัวของมันเอง โดยที่ผู้ใช้ไม่จำเป็นต้องออกจากสภาพแวดล้อมของภาษา LISP จะส่งผลลัพธ์ของการหาค่า (Evaluation) ทันทีที่มีการพิมพ์โปรแกรมสิ้นสุด ดังนั้นในการโปรแกรมของภาษา ไม่จำเป็นต้องมีการประกาศชนิดของตัวแปรที่ถูกใช้ในโปรแกรม (นิตยา นินทรกิจ,2541)

ข้อได้เปรียบของภาษา LISP ในการนำมาใช้พัฒนาระบบผู้เชี่ยวชาญ

(1) โครงสร้างทางข้อมูลของภาษา LISP คือ List ซึ่งเป็นโครงสร้างที่โปรแกรม AI ส่วนมากใช้ในการแทนความรู้
(2) สามารถรวบรวมข้อเท็จจริงเกี่ยวกับสิ่งต่าง ๆ ได้ง่าย โดยการแทนข้อมูลต่าง ๆ ให้อยู่ในรูป Property list
(3) โครงสร้างในการควบคุมภาษา LISP ที่สำคัญคือ Recursion ซึ่งเหมาะกับการแก้ปัญหาหลายแบบ
(4) การกำหนดค่าให้กับตัวแปรจะกำหนดช่วงไหนก็ได้

ภาษา PLANNER

ภาษา PLANNER เป็นภาษาซึ่งพัฒนาขึ้นเพื่อใช้ในการแก้ปัญหา (problem solving) หรือใช้ในการพิสูจน์ทฤษฎี (theorem proving) ภาษานี้พัฒนาโดย Carl Hewitt แห่งสถาบัน MIT ในปี ค.ศ.1967 ภาษาโปรแกรมโดยมากเมื่อต้องการจะแก้ปัญหาใด จะต้องบอกให้แน่ชัดว่าจะใช้ลำดับการทำงานไหนในการแก้ปัญหา สำหรับภาษา PLANNER สามารถใส่ลำดับการทำงานที่ใช้แก้ปัญหาหลาย ๆ ปัญหาไว้ก่อน เมื่อป้อนข้อมูลเข้าไปภายหลัง PLANNER จะค้นหาลำดับการทำงานที่จะใช้แก้ปัญหานั้นเอง และให้คำตอบออกมา

ภาษา CONNIVER

ภาษา CONNIVER เป็นภาษาที่ถูกพัฒนาขึ้นที่สถาบัน MIT ในปี ค.ศ. 1972 โดย G.J.Sussman ภาษา CONNIVER มีลักษณะคล้ายกับภาษา PLANNER หลายประการ แต่แทนที่จะให้ภาษาทำการ

backtrack แบบอัตโนมัติ กลับให้คนเป็นผู้ควบคุมแทน โดยปกติภาษา PLANNER จะใช้วิธีค้นหาแบบ depth-first search ซึ่งบางกรณีจะพบวิธีการค้นหาแบบนี้ไม่มีประสิทธิภาพ นอกจากนั้นก็ได้มีการเพิ่มฟังก์ชั่นต่าง ๆ ขึ้นด้วย

ภาษา PROLOG

ภาษา PROLOG ย่อมาจากคำว่า Programming in Logic ถูกพัฒนาโดยกลุ่มนักวิชาการ Group d' Intelligence Artificielle แห่งมหาวิทยาลัย Universite d' Aix Marseilles ประเทศฝรั่งเศส ในปี ค.ศ. 1972 โดยมี Alain Colmerauer และ P. Roussel เป็นผู้นำภาษา PROLOG เป็นภาษาที่มีพื้นฐานมาจาก Predicate Calculus โปรแกรมในภาษานี้จะประกอบด้วยกฎสำหรับพิสูจน์ความสัมพันธ์ระหว่างสิ่งของ เป็นภาษาที่มีผู้นิยมใช้ในการพัฒนาการแก้ปัญหาด้านปัญญาประดิษฐ์มากอีกภาษาหนึ่ง

ในงานวิจัยนี้จะใช้ภาษา LISP เป็นภาษาหลักในการเขียนโปรแกรมควบคุมส่วนต่าง ๆ เช่น ส่วนฐานความรู้ กลไกการอนุมาน ส่วนติดต่อกับผู้ใช้ และส่วนติดต่อกับภายนอก ซึ่งภาษา LISP ที่ใช้นี้ทำงานบนระบบปฏิบัติการลินุกซ์