บทที่ 3

ทฤษฎีและหลักการของระบบผู้ชำนาญการ

3.1 ระบบผู้ชำนาญการคืออะไร

ระบบผู้ชำนาญการหรือที่เรียกกันว่า Expert System เป็นศาสตร์แขนงหนึ่งในเรื่องของปัญญาประดิษฐ์ (Artificial Intelligence: AI) ซึ่งมีความหมายถึงสิ่งชาญฉลาดที่ถูกประดิษฐ์ขึ้นมา สำหรับระบบผู้เชี่ยวชาญเองจะหมายถึงโปรแกรมคอมพิวเตอร์ที่นำมาใช้แก้ปัญหาโดยอาศัยความรู้ในสาขาใดสาขาหนึ่งที่รวบรวมจากผู้เชี่ยวชาญในสาขานั้นและเก็บไว้เป็นฐานความรู้ในคอมพิวเตอร์ โปรแกรมดังกล่าวต้องเป็นโปรแกรมที่สามารถแก้ปัญหาได้หลายลักษณะโดยไม่จำเป็นต้องมีการเขียนโปรแกรมใหม่เพื่อแก้ปัญหาแต่ละปัญหาโดยเฉพาะ ความสามารถในการแก้ปัญหาของโปรแกรมนี้จะต้องเทียบเท่าความสามารถของผู้มีความชำนาญระดับผู้เชี่ยวชาญในสาขานั้นๆ

ในการพัฒนาระบบผู้ชำนาญการที่ใช้แก้ปัญหาที่มีความซับซ้อน สิ่งแรกที่ต้องพิจารณาคือ สร้างความเข้าใจในกรรมวิธีการทำงานของผู้เชี่ยวชาญอย่างลึกซึ้งเพื่อจะได้ทราบถึงขั้นตอนในการทำงานจริงของผู้เชี่ยวชาญ ความรู้ที่เก็บรวบรวมจากผู้เชี่ยวชาญจะนำมาใช้กับกระบวนการอนุมานของระบบผู้ชำนาญการ ตัวอย่างลักษณะงานที่เหมาะที่จะสร้างเป็นระบบผู้ชำนาญการมีดังนี้

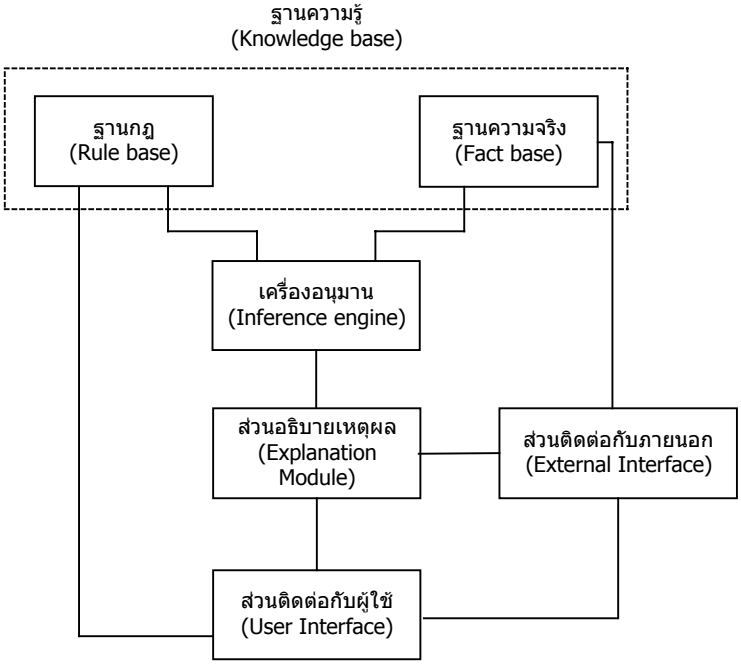
- เป็นงานที่ใช้เหตุผลและความรู้ในการแก้ปัญหา
- เป็นงานที่ใช้เวลาในการแก้ปัญหาพอสมควร
- เป็นงานที่มีผู้ชำนาญในเรื่องนั้นๆ สามารถให้ความรู้ได้
- ความชำนาญที่ใช้ในงานนั้นสามารถถ่ายทอดให้ผู้เริ่มต้นใหม่ได้
- กฎและวิธีการหาเหตุผลในระบบนั้นมีอยู่และสามารถให้คำตอบที่เข้าใจได้
- เงื่อนไขต่างๆ ของกฎสามารถกำหนดได้มีเป้าหมายย่อยที่ไม่ขัดแย้งกัน เป็นต้น

3.2 สถาปัตยกรรมของระบบผู้ชำนาญการ

ระบบผู้ชำนาญการมีส่วนประกอบ ดังแสดงในภาพประกอบ 3-1 มีส่วนที่สำคัญ 5 ส่วน คือ

- ฐานความรู้ (Knowledge Base)
- เครื่องอนุมาน (Inference Engine)
- ส่วนอธิบายเหตุผล (Explanation Module)

- ส่วนติดต่อกับผู้ใช้ (User Interface) และ ส่วนติดต่อกับภายนอก (External Interface)



ภาพประกอบ 3-1 สถาปัตยกรรมของระบบผู้ชำนาญการ

3.3 ฐานความรู้

ฐานความรู้เป็นส่วนที่เก็บความรู้ทุกประเภทที่เกี่ยวข้องกับปัญหา ไม่ว่าจะเป็นความรู้จากตำราหรือความรู้จากผู้เชี่ยวชาญ ความรู้ต่างๆ ในสาขาที่ต้องการแก้ปัญหาจะถูกเก็บไว้ในรูปที่คอมพิวเตอร์สามารถประมวลผลได้ และมีการบำรุงรักษาฐานความรู้นี้ให้ทันสมัยเสมอ โดยปรกติแล้วความรู้ที่รวบรวมมาจะมีจำนวนมาก ปัญหาที่สำคัญประการหนึ่งในการสร้างฐานความรู้ขึ้นมาคือ ทำอย่างไรจึงจะเก็บรวบรวมความรู้ที่มากมายเหล่านั้นให้เก็บไว้ในคอมพิวเตอร์ได้โดยไม่เปลืองเนื้อที่มากนัก ในขณะเดียวกันก็อยู่ในรูปที่สามารถวิเคราะห์หาเหตุผลจากความรู้นั้นออกมาได้

ลักษณะการแทนความรู้ มีรูปแบบ 2 ลักษณะ คือ

- การแทนความรู้ในรูปของข้อเท็จจริงเกี่ยวกับเรื่องที่สนใจศึกษา พร้อมทั้งกฎ (Rules) ในการอนุมานความรู้จากข้อเท็จจริงนั้นๆ เรียกการแทนความรู้แบบนี้ว่า Declaration Method

- การแทนความรู้ในรูปของกระบวนการ กล่าวคือ เมื่อต้องการทราบความรู้ในเรื่องอะไร ก็จะมีกระบวนการสำหรับหาความรู้ในเรื่องนั้นให้เรียกใช้ การแทนในลักษณะนี้เรียกว่า Procedural Method

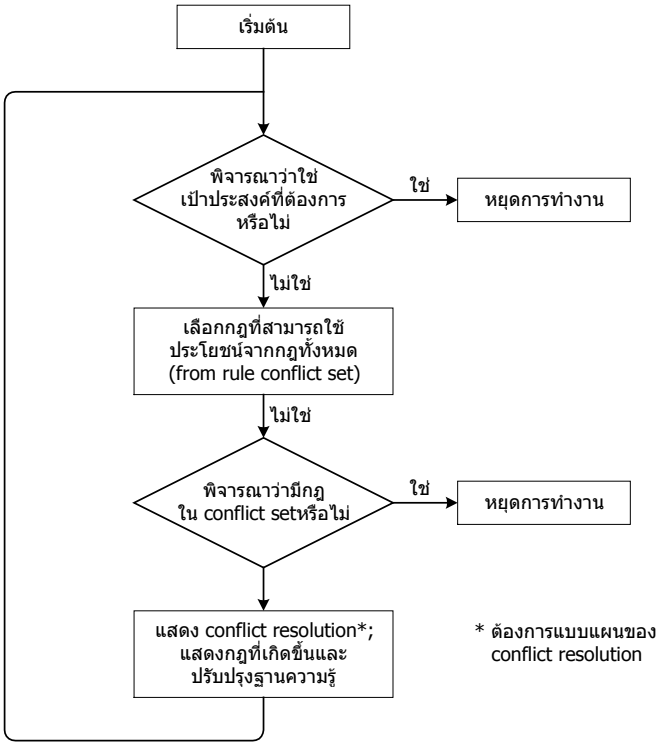
3.4 เครื่องอนุมาน

เครื่องอนุมานเป็นส่วนของโปรแกรมที่ใช้การคิดหาเหตุผลโดยอาศัยความรู้ในฐานความรู้ เพื่อให้ได้วิธีการแก้ปัญหาออกมา วิธีการหาเหตุผลจะสัมพันธ์กับลักษณะการแทนความรู้ กลไกการอนุมานที่เป็นพื้นฐานอยู่ 3 วิธี คือ

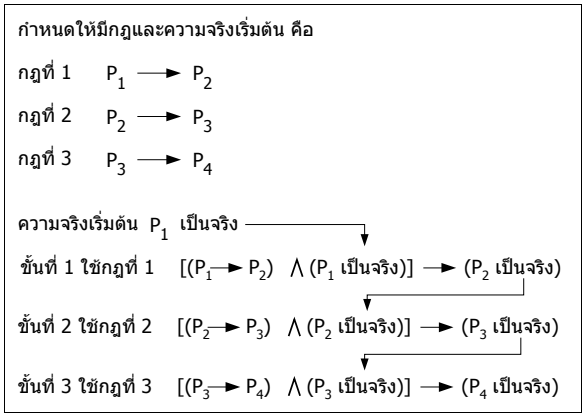
3.4.1 การอนุมานแบบเดินหน้า (Forward-Chaining Method)

การอนุมานแบบเดินหน้าเป็นการแก้ปัญหาโดยที่เริ่มต้นจากสถานะเริ่มต้นไปสู่สถานะเป้าหมายอย่างมีประสิทธิภาพ หรือเป็นกระบวนการในการสร้างฐานความรู้ขึ้นมาใหม่ ซึ่งเราสามารถแสดงการทำงานได้ดังภาพประกอบ 3-2

ข้อดีของการอนุมานแบบเดินหน้า คือ จะค้นหาทุกเป้าหมายที่เป็นไปได้ ส่วนข้อเสีย คือ ในบางครั้งเป้าหมายที่ได้ไม่ใช่สิ่งที่ต้องการ จึงทำให้เสียเวลามาก และเราสามารถแสดงตัวอย่างของการอนุมานแบบเดินหน้าได้ดังภาพประกอบ 3-3 โดยกฎและความจริงที่กำหนดขึ้นจะเป็นส่วนประกอบของฐานความรู้เริ่มต้น



ภาพประกอบ 3-2 การทำงานของการอนุมานแบบเดินหน้า



ภาพประกอบ 3-3 ตัวอย่างของการอนุมานแบบเดินหน้า

เราสามารถอธิบายกระบวนการการอนุมานแบบเดินหน้าได้ดังนี้ 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 ระบบจะตรวจสอบความจริงเริ่มต้น และพบว่าตรงกับกฎที่ 1 ในกฎที่ 1 มี P_1 เป็นเงื่อนไข เมื่อตรวจสอบพบว่า P_1 เป็นจริง ดังนั้น ระบบจะทำตามข้อ

สรุปภายในกฎที่ 1 นั่นคือ ทำตามส่วนข้อสรุปในกฎที่ 1 นั่นคือ P_2 จากนั้น ก็จะเรียกให้งานกฎข้อถัดไป คือ กฎที่ 2 มี P_2 เป็นเงื่อนไข เมื่อตรวจสอบพบว่า P_2 เป็นจริง ดังนั้น ระบบจะทำตามข้อสรุปภายในกฎที่ 2 นั่นคือ ทำตามส่วนข้อสรุปในกฎที่ 2 นั่นคือ P_3 จากนั้น ก็จะเรียกให้งานกฎข้อถัดไป คือ กฎที่ 3 มี P_3 เป็นเงื่อนไข เมื่อตรวจสอบพบว่า P_3 เป็นจริง ดังนั้น ระบบจะทำตามข้อสรุปภายในกฎที่ 3 นั่นคือ ทำตามส่วนข้อสรุปในกฎที่ 3 นั่นคือ P_4 การอนุมานจะเป็นไปในลักษณะเช่นนี้ต่อกันไปเรื่อยๆ จนบรรลุเป้าประสงค์ที่ต้องการ

การอนุมานแบบเดินหน้านี้ ถ้าเกิดกรณีที่มีกฎจำนวนหลายกฎ มีเงื่อนไขตรงกันกับความจริงในฐานความรู้ ระบบจะมีการขจัดข้อขัดแย้ง (Conflict Resolution) ซึ่งมีเกณฑ์การขจัดข้อขัดแย้งหลายลักษณะที่สามารถเลือกใช้ได้ ดังนี้

(1) เลือกใช้อย่างสุ่ม

(2) นำกฎมาเรียงไว้ด้วยกัน กฎใดก็ตามที่ตรงกับความจริงเป็นกฎแรกจะใช้กฎนั้น ส่วนกฎอื่นๆ จะไม่นำมาใช้

(3) กฎใดที่มีเงื่อนไขตรงกับข้อมูลมากที่สุดจะใช้กฎนั้น

(4) กฎใดที่ใช้หลังสุดจะใช้กฎนั้น

(5) กฎใดที่มีตัวแปรที่ใช้ไปหลังสุดจะใช้กฎนั้น

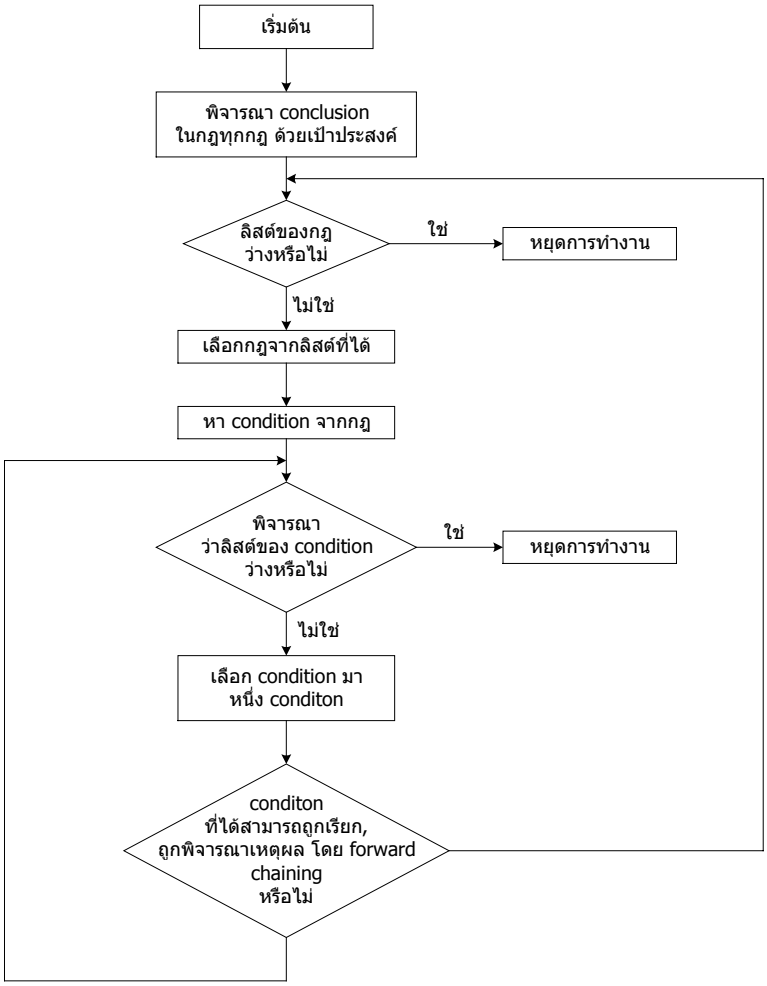
(6) ใช้กฎที่เพิ่มเข้าไปในฐานความรู้หลังสุด

(7) ให้น้ำหนักของกฎและความจริง ใช้กฎที่มีน้ำหนักมากที่สุด

3.4.2 การอนุมานแบบย้อนหลัง (Backward-Chaining Method)

การอนุมานแบบย้อนหลัง เป็นการแก้ปัญหาโดยเริ่มต้นจากสถานะเป้าหมายไปสู่สถานะเริ่มต้นอย่างมีประสิทธิภาพ หรือเป็นกระบวนการพิสูจน์ สมมุติฐานซึ่งเราสามารถแสดงการทำงานของการอนุมานแบบย้อนหลังได้ดังภาพประกอบ 3-4

ข้อดีของการอนุมานแบบนี้ คือ จะค้นหาเฉพาะเป้าหมายที่ต้องการเท่านั้น จึงใช้เวลาน้อย ส่วนข้อเสียของการอนุมานแบบถอยหลัง คือ ในบางครั้งอาจจะไม่ได้เป้าหมายใดออกมาเลย เนื่องจากเงื่อนไขในการที่จะนำไปสู่เป้าหมายนั้นไม่ถูกต้อง และสามารถแสดงตัวอย่างของการอนุมานแบบย้อนหลังได้ ดังภาพประกอบ 3-5



ภาพประกอบ 3-4 การทำงานของการอนุมานแบบย้อนหลัง

กำหนดให้มีกฎและความจริงเริ่มต้น คือ

กฎที่ 1 $P_1 \longrightarrow P_2$

กฎที่ 2 $P_2 \longrightarrow P_3$

กฎที่ 3 $P_3 \longrightarrow P_4$

ความจริงเริ่มต้น P_1 เป็นจริง ถ้าเป้าประสงค์ (goal) ที่ต้องการคือ P_4

ขั้นที่ 1 ตรวจสอบกฎที่ต้องนำมาใช้ในการหาเป้าประสงค์นั้น

เป้าประสงค์ (goal) คือ P_4

กฎที่ต้องนำมาใช้ คือ กฎที่ 3 $P_3 \longrightarrow P_4$

กฎที่ต้องนำมาใช้ คือ กฎที่ 2 $P_2 \longrightarrow P_3$

กฎที่ต้องนำมาใช้ คือ กฎที่ 1 $P_1 \longrightarrow P_2$

สิ้นสุดการตรวจสอบกฎ เนื่องจากส่วนของเงื่อนไขของกฎสุดท้ายที่ได้มาเป็นความจริงเริ่มต้น

ขั้นที่ 2 นำกฎที่ได้ไปอนุมานแบบเดินหน้า โดยเริ่มต้นการอนุมานจากกฎที่ได้มาสุดท้าย
ไปยังกฎที่ได้มาเป็นกฎแรก ซึ่งมีลำดับการอนุมานดังภาพประกอบ 3-2

ภาพประกอบ 3-5 ตัวอย่างของการอนุมานแบบย้อนหลัง

อธิบายกระบวนการการอนุมานแบบย้อนหลังดังนี้ 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 จำเป็นต้องใช้กฎข้อไหนบ้างในในการเดินทางจากเป้าประสงค์กลับไปยังความจริงเริ่มต้น นั่นคือ ระบบจะหาเส้นทางเดินจาก P_4 กลับไปหา P_1 จะพบว่า เส้นทางในการเดินทางนั้นมีดังนี้ เริ่มแรก จากเป้าประสงค์ P_4 ต้องเรียกใช้กฎข้อ3 นั่นคือ ถ้า P_3 แล้ว P_4 ต่อจากนั้น ก็ต้องหาเส้นทางเดินต่อ จึงเรียกใช้กฎข้อ 2 นั่นคือ ถ้า P_2 แล้ว P_3 ระบบก็จะหาเส้นทางเดินต่อไป คือ ใช้กฎข้อ 1 นั่นคือ ถ้า P_1 แล้ว P_2 สุดท้ายก็จากเป้าประสงค์ที่ต้องการ P_4 ก็จะสามารถหาเส้นทางเดินกลับไปยังความจริงเริ่มต้น P_1 ได้นั่นเอง

3.4.3 การอนุมานแบบผสม (Mixed-Method)

การอนุมานแบบผสม เป็นการเริ่มการทำการอนุมานแบบย้อนหลังก่อน เมื่อการอนุมานนั้นสำเร็จเราก็จะได้เส้นทางการอนุมานที่สามารถนำไปสู่เป้าหมายที่มีประสิทธิภาพที่สุดหลังจากนั้น จึงนำเส้นทางการอนุมานนี้มาทำการอนุมานแบบเดินหน้าต่อไป เพื่อให้ได้ความจริงเพิ่มขึ้นตามเส้นทางการอนุมานนั้น ๆ จนกระทั่งได้เป้าหมายออกมา

3.5 ส่วนติดต่อกับผู้ใช้และส่วนติดต่อกับภายนอก

ส่วนติดต่อกับผู้ใช้เป็นส่วนที่ใช้สำหรับการถ่ายทอดข้อมูลการโต้ตอบระหว่างผู้ใช้ระบบและข้อมูลดังกล่าวจะถูกนำไปเก็บไว้ในฐานข้อมูลแบบไดนามิก การที่โปรแกรมจะใช้งานได้ง่ายหรือยากเพียงใดขึ้นกับการออกแบบในส่วนนี้ ส่วนติดต่อกับภายนอกเป็นส่วนที่ระบบต้องการติดต่อกับโปรแกรมภายนอกหรือภายในระบบเอง เช่น การคำนวณ การแสดงภาพ การแสดงข้อความส่วนอธิบาย การสั่งพิมพ์ เป็นต้น ซึ่งในแต่ละกฎสามารถเรียกการแสดงได้ทั้งในส่วนเงื่อนไขและผลสรุป

3.6 ส่วนอธิบายเหตุผล

ส่วนอธิบายเหตุผล คือ ส่วนที่ให้ความมั่นใจกับผู้ใช้ เมื่อผู้ใช้ปรึกษาระบบชำนาญการและได้คำตอบออกมา เพียงคำตอบอย่างเดียว ผู้ใช้อาจไม่มั่นใจว่าคำตอบนั้นจะนำไปแก้ปัญหาได้จริงแค่ไหน การอธิบายเหตุผลให้ผู้ใช้ทราบอาจจะทำให้ผู้ใช้นำคำตอบที่ได้ไปใช้อย่างมั่นใจยิ่งขึ้น

3.7 ภาษาสำหรับระบบผู้ชำนาญการ

ภาษาคอมพิวเตอร์ที่ใช้กันอยู่ในปัจจุบันส่วนใหญ่เป็นภาษาที่จะต้องกำหนดลำดับขั้นตอนในการทำงานที่แน่นอน ซึ่งเรียกว่า Procedural Language ซึ่งผู้เขียนโปรแกรมได้กำหนดลำดับขั้นตอนหรือที่เรียกว่า Alglorithm สำหรับการแก้ปัญหานั้นเรียบร้อยแล้ว สำหรับระบบผู้ชำนาญการต้องอาศัยการตัดสินใจทางตรรกวิทยา และทำการค้นหาคำตอบซึ่งมีรูปแบบหรือขั้นตอนที่ไม่แน่นอน ดังนั้น ภาษาประเภท Procedural Language จึงไม่สามารถที่จะนำมาใช้ได้ หรือในบางกรณีก็อาจจะใช้ได้ในขอบเขตจำกัด แต่ต้องอาศัยการสร้างโปรแกรมที่ซับซ้อนมาก (วริทธิ์ อึ้งภากรณ์, 2531)

3.7.2 ภาษา IPL

ภาษา IPL ย่อมาจากคำว่า Information Processing Language เป็นภาษาที่พัฒนาสำหรับการแก้ปัญหาทางปัญญาประดิษฐ์เป็นภาษาแรก ถูกพัฒนาในปี ค.ศ. 1956 โดย Allen Newell, J.C. Shaw และ H. Simon แห่งมหาวิทยาลัย Carnegie ได้มีการพัฒนาภาษานี้ขึ้นมาหลายรุ่นด้วยกัน โดยมีภาษา IPL-V เป็นรุ่นสุดท้าย โดยจุดประสงค์หลักของภาษา IPL คือ การทำงานกับลิสต์ แต่เนื่องจากภาษานี้คล้ายกับภาษาเครื่อง (Machine Language) จึงใช้งานยากและเลิกใช้กันในเวลาต่อมา

3.7.2 ภาษา LISP

ภาษา LISP ย่อมาจากคำว่า LIST Processing เป็นภาษาที่มีผู้ใช้มากที่สุดในการพัฒนาโปรแกรมทางปัญญาประดิษฐ์ เนื่องจากภาษานี้มีคุณสมบัติที่เหมาะสมหลายประการ ซึ่งถูกพัฒนาโดย J. McCarty แห่งสถาบัน MIT ในปี ค.ศ.1950 เป็นภาษาที่ใช้ได้ดีในการประมวลผลทางสัญลักษณ์ ภาษานี้จึงถูกใช้กันอย่างแพร่หลายในการแก้ปัญหาทางปัญญาประดิษฐ์ ภาษา LISP เป็นภาษาทางคอมพิวเตอร์ชั้นสูงภาษาหนึ่ง เป็นภาษาที่ใช้ได้ดีในการประมวลผลทางสัญลักษณ์ (Symbol) เป็นภาษาที่ใช้กันอย่างแพร่หลายในทางด้านปัญญาประดิษฐ์ (Artificial Intelligence :AI) โดยลักษณะพิเศษของภาษานี้คือ เป็นภาษาที่มีการตอบรับทันทีที่โปรแกรมถูกป้อนเข้าไป และจะมีการตรวจสอบไวยากรณ์ของภาษาด้วยตัวมันเองโดยที่ผู้ใช้ ไม่จำเป็นต้องออกจากสภาพแวดล้อมของภาษา LISP จะส่งผลลัพธ์ของการหาค่า (Evaluation) ทันทีที่มีการพิมพ์โปรแกรมสิ้นสุด ดังนั้นในการโปรแกรมของภาษา เราไม่จำเป็นต้องมีการประกาศชนิดของตัวแปรที่ถูกใช้ในโปรแกรม (นิตยา นินทรกิจ, 2541)

ข้อได้เปรียบของภาษา LISP เมื่อนำมาใช้พัฒนาระบบผู้ชำนาญการ

(1) โครงสร้างทางข้อมูลของภาษา LISP คือ List ซึ่งเป็นโครงสร้างที่โปรแกรม AI ส่วนมากใช้ในการแทนความรู้

(2) สามารถรวบรวมข้อเท็จจริงเกี่ยวกับสิ่งต่างๆ ได้ง่าย โดยการแทนข้อมูลต่างๆ อยู่ในรูป Property list

(3) โครงสร้างในการควบคุมภาษา LISP ที่สำคัญคือ Recursion ซึ่งเหมาะกับการแก้ปัญหาหลายแบบ

(4) การกำหนดค่าให้กับตัวแปรจะกำหนดช่วงไหนก็ได้

โดยในงานวิจัยนี้เลือกใช้ภาษา LISP เป็นภาษาหลักในการเขียนโปรแกรมควบคุมส่วนต่างๆ เช่น กลไกการอนุมาน ฐานความรู้ ส่วนติดต่อกับผู้ใช้ และส่วนติดต่อกับภายนอก โดยภาษา LISP นี้ทำงานบนระบบปฏิบัติการลีนุกซ์

7.2.3 ภาษา PLANNER

ภาษา PLANNER เป็นภาษาซึ่งพัฒนาขึ้นเพื่อใช้ในการแก้ปัญหา (Problem-Solving) หรือในการพิสูจน์ทฤษฎี (Theorem-Proving) ภาษานี้พัฒนาโดย Carl Hewitt แห่งสถาบัน MIT ในปี ค.ศ. 1967 ภาษาโปรแกรมโดยมากเมื่อต้องการจะแก้ปัญหาใด จะต้องบอกให้แน่ชัดว่าจะใช้ลำดับการทำงานไหนในการแก้ปัญหา สำหรับภาษา PLANNER สามารถใส่ลำดับการทำงานที่ใช้แก้ปัญหาหลายๆ ปัญหาไว้ก่อน เมื่อป้อนข้อมูลเข้าไปในภายหลัง PLANNER จะค้นหาลำดับการทำงานที่จะใช้แก้ปัญหานั้นเองและให้คำตอบออกมา

7.2.4 ภาษา CONNIVER

ภาษา CONNIVER เป็นภาษาที่ถูกพัฒนาขึ้นที่สถาบัน MIT ในปี ค.ศ. 1972 โดย G.J. Sussman ภาษา CONNIVER มีลักษณะคล้ายภาษา PLANNER หลายประการ แต่แทนที่จะให้ภาษาทำการ Backtrack แบบอัตโนมัติ กลับให้คนเป็นผู้ควบคุมแทน ทั้งนี้เพราะปรกติภาษา PLANNER ใช้วิธีการค้นหาแบบ Depth-first search ซึ่งในบางกรณีจะพบวิธีการค้นหาแบบนี้ไม่มีประสิทธิภาพ นอกจากนั้นก็ได้มีการเพิ่มฟังก์ชันต่างๆ ขึ้นอีกมาก

7.4.5 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 เป็นผู้นำ ภาษา PROLOG เป็นภาษาที่มีพื้นฐานมาจาก Predicate Calculus โปรแกรมในภาษานี้จะประกอบด้วยกฎสำหรับพิสูจน์ความสัมพันธ์ระหว่างสิ่งของ เป็นภาษาที่มีผู้นิยมใช้ในการพัฒนาการแก้ปัญหาด้านปัญญาประดิษฐ์มากอีกภาษาหนึ่ง

3.8 การพัฒนาระบบผู้ชำนาญการ

ดังที่ได้กล่าวมาแล้วว่าระบบผู้ชำนาญการเป็นระบบที่ใช้แก้ปัญหาที่มีความซับซ้อนและมีความไม่แน่นอน จึงจำเป็นต้องใช้ความรู้ความชำนาญระดับสูง งานที่ต้องใช้ความรู้ความชำนาญระดับสูงพวกนี้มีหลายลักษณะด้วยกันเช่น เป็น

3.8.1 การตีความ (Interpretation)

การตีความ คือ การวิเคราะห์ข้อมูลวิเคราะห์ข้อมูลเพื่อว่าความหมายที่ได้จากข้อมูลคืออะไร เช่น ดูข้อมูลจาก mass spectrometer และพยายามตีความจากข้อมูลว่าโครงสร้างทางเคมีคืออะไร งานในลักษณะนี้จะต้องตีความให้ถูกต้อง โดยปกติมักพิจารณาตีความที่เป็นไปได้ทั้งหมดก่อน แล้วจึงตัดข้อใดข้อหนึ่งทิ้ง เมื่อมีเหตุผลเพียงพอว่าการตีความแบบนั้นผิด ปัญหาของงานในลักษณะนี้คือ ข้อมูลมักมีความคลาดเคลื่อนรวมอยู่ด้วย อาจกล่าวได้ว่าผู้ตีความมีข้อเท็จจริงอยู่เพียงบางส่วน ถ้าข้อมูลที่ได้มาเชื่อถือไม่ได้ การตีความก็จะเชื่อถือไม่ได้ไปด้วย จึงมีความจำเป็นที่ผู้ตีความจะต้องจำแนกให้ได้ว่าข้อมูลส่วนไหนเชื่อถือไม่ได้หรือไม่สมบูรณ์

3.8.2 การวินิจฉัย (Diagnosis)

การวินิจฉัย คือ การตรวจสอบว่ามีอะไรผิดพลาดในระบบ เช่น การวินิจฉัยโรคติดเชื้อ งานในลักษณะนี้ผู้วินิจฉัยจำเป็นต้องเข้าใจว่าระบบประกอบกันขึ้นมาได้อย่างไรและแต่ละส่วนย่อยๆ ในระบบสัมพันธ์กันอย่างไร งานในลักษณะนี้มีความยากตรงที่ความผิดพลาดอย่างหนึ่งอาจมีอาการซึ่งเป็นสาเหตุของความผิดพลาดอีกอย่างหนึ่งปิดบังอยู่

3.8.3 การควบคุมการทำงาน (Monitoring)

การควบคุมการทำงาน คือ การตีความข้อมูลที่ส่งออกมาจากระบบว่ามีอะไรผิดพลาดหรือไม่และต้องแจ้งให้ผู้ทำงานทราบทันทีที่เมื่อเกิดความผิดพลาดขึ้น เช่น การควบคุมการทำงานของเครื่องช่วยหายใจของคนไข้ผ่าตัด งานในลักษณะนี้จะต้องเห็นว่ารวมเอางานวินิจฉัยเอาไว้ด้วย

เพราะจำเป็นต้องวินิจฉัยว่าความผิดพลาดในระบบคืออะไร และต้องแจ้งสาเหตุแห่งความผิดพลาดให้ผู้ใช้เครื่องมือทราบทันที

3.8.4 การทำนาย (Prediction)

การทำนาย คือ การคาดเดาพฤติกรรมในอนาคตจากตัวแบบซึ่งได้จากอดีตและปัจจุบัน เช่น การทำนายผลกระทบเนื่องจากการเปลี่ยนแปลงนโยบายทางเศรษฐกิจ การทำนายเป็นการหาเหตุผลที่เกี่ยวเนื่องกับเวลา ดังนั้นผู้ทำนายจะต้องสามารถอ้างอิงสิ่งต่างๆ ที่เปลี่ยนแปลงไปตามเวลาได้ การทำนายจำเป็นต้องอาศัยความรู้ที่มีอยู่ซึ่งไม่สมบูรณ์มาทำนาย ในการทำนายจริงๆ จึงควรบอกทางที่จะเป็นไปได้ไว้หลายๆ ทางพร้อมทั้งบอกด้วยว่าถ้าข้อมูลมีการเปลี่ยนแปลงจะทำอะไรเปลี่ยนแปลงได้บ้าง

3.8.5 การวางแผน (Planning)

การวางแผน คือ การกำหนดลำดับของการกระทำที่สามารถนำไปสู่เป้าหมายได้ คนที่วางแผนจะต้องสร้างแผนขึ้นมาในลักษณะที่สามารถนำไปสู่เป้าหมายได้โดยไม่ใช้ทรัพยากรมากเกินไป และไม่ขัดแย้งกับข้อจำกัดต่างๆ ที่มี ถ้าเป้าหมายขัดแย้งกัน ผู้วางแผนต้องสามารถจัดลำดับความสำคัญได้ ถ้าข้อมูลหรือข้อจำกัดเกี่ยวกับแผนมีการเปลี่ยนแปลงตามเวลาหรือไม่สมบูรณ์ ผู้วางแผนก็ต้องสามารถปรับเปลี่ยนได้ ปกติแล้วในการวางแผนจะมีบางส่วนที่ต้องใช้การทำนายเข้ามาช่วย ปัญหาในการวางแผนคือ ระบบที่ต้องวางแผนมักใหญ่และซับซ้อน ผู้วางแผนอาจเข้าใจปัญหาในทันทีไม่ได้ จำเป็นต้องใช้เวลาในการศึกษาอย่างละเอียดรอบคอบ ถ้ามีรายละเอียดมาก อาจต้องแบ่งเป็นปัญหาย่อยๆ ก่อน ถ้าปัญหาย่อยๆ นี้มีความสัมพันธ์กัน ผู้วางแผนต้องสามารถดำเนินการกับความสัมพันธ์เหล่านี้ได้ ข้อมูลที่ใช้ในการวางแผนมักมีความไม่แน่นอนรวมอยู่ด้วย ดังนั้นผู้วางแผนควรกำหนดได้ว่าแผนที่วางไว้นั้นโอกาสที่จะต้องใช้เป็นเท่าไร

3.8.6 การออกแบบ (Designing)

การออกแบบ คือ การกำหนดรายละเอียด (Specification) ของสิ่งใดสิ่งหนึ่งให้มีคุณสมบัติตามที่ต้องการ เช่น การออกแบบวงจรดิจิตอล ลักษณะในรายละเอียดของการออแบบจะคล้ายกับการวางแผน การพัฒนาระบบผู้ชำนาญการจึงขึ้นอยู่กับลักษณะของปัญหาในลักษณะใดที่กล่าวมาแล้ว ปัจจัยสำคัญที่จะทำให้การพัฒนาระบบชำนาญการประสบความสำเร็จหรือไม่ คือ การเลือกงานที่จะนำมาสร้างเป็นระบบผู้ชำนาญการ