บรรณานุกรม

1. Hayes-Roth, F., Building Expert Systems (Hayes-Roth, F., D. A. Waterman, and D. B. Lenat, eds.), Addison-Wesley Publishing Company Inc., New York, 1983.
2. Feigenbaum, E. A., and P. McCorduck, Fifth Generation, Addison-Wesley Publishing Company Inc., New York, 1984.
3. Harmon, P., and D. King, Expert Systems : Artificial Intelligence in Business, John Wiley & Sons Inc., New York, 1983.
4. Rich, E., Artificial Intelligence, McGraw-Hill Inc., Singapore, 1983.
5. Buchanan, B. G., and R. O. Duda, "Principles of Rule-based Expert Systems," Advances in Computers (Yovits, M. C., eds.), Vol. 12, pp. 164-216, Academic Press, New York, 1983.
6. Barr, A., P. R. Cohen, and E. A. Feigenbaum, The Handbook of Artificial Intelligence (Barr, A. and E. A. Feigenbaum, eds.), Vol. 1, William-Kaufman Inc., Los Altos, 1981.
7. Stanley, J. R., "The Production System : Architecture and abstraction," Pattern-directed Inference Systems, (D. A. Waterman and F. Hayes-Roth eds.), pp. 525-538, Academic Press, London, 1978.
8. Shortliffe, E. H., "Details of the Consultation System," Rule-based Expert systems : The MYCIN Experiments of the Stanford Heuristic Programming Project, (Buchanan, B. G., and E. H. Shortliffe, eds.), Addison-Wesley, pp. 73-132, Massachusetts, 1984.
9. นิพนธ์ เทพะประสิทธิ์, "ผลของอาหารซึ่งมีโปรตีนระดับต่างๆ ที่มีต่อการเจริญเติบโตของกุ้งกุลาดำ," รายงานผลการทดลองเลี้ยงกุ้งกุลาดำด้วยอาหารชนิดต่างๆ ณ แปลงทดลอง จังหวัดสมุทรสาคร, เอกสารวิชาการฉบับที่ 1/2522, หน่วยงานอาหารสัตว์น้ำ งานทดลองและวิจัยเพื่อการเพาะเลี้ยง กองประมงน้ำกร่อย กรมประมง, 2522.

10. Motoh, H., "Biology and Ecology of Penaeus monodon," Proceeding of the First International Conference on the culture of Penaeid Prawns/Shrimps (Yasuhiko, T., J. H. Primavera, and J. A. Llobera, eds.), pp. 27-64, The Aquaculture Department Southeast Asian Fisheries Development Center, Iloilo, 1985.

11. Primavera, J. H., "Broodstock of Sugpo, Penaeus monodon," SEC/SM/21 SAFIS Manual, No. 21, 2528, (สุปราณี ชินบุตร, การเลี้ยงพ่อแม่พันธุ์กุ้งกุลาดำ, 2528).

12. Apud, F., J. H. Primavera, and P. L. Torres, Jr., Farming of Prawns and Shrimps, The Aquaculture Department Southeast Asian Fisheries Development Center, Iloilo, 3rd., 1985.

13. นิเวศน์ เรืองพานิช, "การเลี้ยงกุ้งกุลาดำให้มีไข่แก่ในบ่อและการเพาะพันธุ์ลูกกุ้ง," สัตว์เศรษฐกิจ, ฉบับที่ 78, หน้า 62 - 67, 2530.

14. สมบูรณ์ หลาวประเสริฐ, "การเร่งกุ้งกุลาดำให้มีไข่แก่โดยการบีบตา," รายงานวิชาการ ฉบับที่ 8/2528, สถานีประมงน้ำกร่อย จังหวัดระยอง, กองประมงน้ำกร่อย กรมประมง, 2528.

15. ศักดิ์ชัย โชติคุณ, "เปรียบเทียบผลการตัดตากุ้งกุลาดำจากนากุ้งและจากธรรมชาติ," เอกสารวิชาการฉบับที่ 25/2530, สถานีพัฒนาการเลี้ยงกุ้ง จังหวัดระยอง, กองประมงน้ำกร่อย กรมประมง, 2530.

16. บุญถม เชื้อชัด, "เปิดตัวปรมาจารย์เพาะกุ้งสำเร็จรายแรก : คุยกับศิษย์เอกเพาะกุ้งจำหน่ายทั่วประเทศ," สัตว์เศรษฐกิจ, ฉบับที่ 46, 2529.

17. นิเวศน์ เรืองพานิช และ คณะ, "ทดลองเร่งกุ้งกุลาดำให้มีไข่แก่ในบ่อซีเมนต์โดยใช้กุ้งจากแหล่งน้ำบริเวณชายฝั่งและจากทะเลสาบสงขลา," เอกสารวิชาการฉบับที่ 8/2528, ฝ่ายผลิตและขยายพันธุ์สัตว์น้ำ สถาบันเพาะเลี้ยงสัตว์น้ำชายฝั่ง จังหวัดสงขลา, กรมประมง, 2528.

18. บังอร ศรีมุกดา และ เตรียม ชิสาเวทย์, "เทคนิคการอนุบาลลูกกุ้งกุลาดำ," รายงานวิชาการฉบับที่ 3/2527, สถานีประมงน้ำกร่อยจังหวัดระยอง กองประมงน้ำกร่อย กรมประมง, 2529.

19. จารุรัตน์ นาถะภัฏ และ สมนึก กบิลรัมย์. "การบริโภคออกซิเจนของลูกกุ้งกุลาดำในระยะการเจริญเติบโตต่างๆ." รายงานวิชาการฉบับที่ 1/2529. สถานีประมงน้ำกร่อย จังหวัดระยอง. กองประมงน้ำกร่อย กรมประมง. 2529.

20. ไพรัตน์ กอสุทธารักษ์. "ผลของแหล่งโปรตีนต่างชนิดในอาหารสำเร็จรูปที่มีต่อลูกกุ้งกุลาดำ." เอกสารวิชาการฉบับที่ 19/2528. สถาบันเพาะเลี้ยงสัตว์น้ำชายฝั่ง จังหวัดสงขลา กรมประมง. 2528.

21. สิริ ทุกข์วินาศ. "ผลของ Nitrite - Nitrogen และ Ammonia - Nitrogen ต่ออัตราการตายของลูกกุ้งกุลาดำวัยอ่อนและลูกปลากระพงขาววัยอ่อน." เอกสารวิชาการ ฉบับที่ 6/2527. สถาบันเพาะเลี้ยงสัตว์น้ำชายฝั่ง จังหวัดสงขลา กรมประมง. 2527.

22. ยนต์ มุสิก และ วรรณา รัตน์โกสีย์กิจ. "ความทนทานต่อการเปลี่ยนแปลงความเค็มของลูกกุ้งกุลาดำ." เอกสารวิชาการฉบับที่ 3/2525. งานทดลองและวิจัยเพื่อการเพาะเลี้ยง. กองประมงน้ำกร่อย กรมประมง. 2525.

23. นิพนธ์ เหมะประสิทธิ์ และ คณะ. "การทดลองการเลี้ยงกุ้งกุลาดำโดยใช้เนื้อปลาสดและอาหารผสมที่จังหวัดสมุทรสาคร." เอกสารวิชาการฉบับที่ 3/2527. ฝ่ายทดลองและวิจัยเพื่อการเพาะเลี้ยง. กองประมงน้ำกร่อย กรมประมง. 2527.

24. นิพนธ์ เหมะประสิทธิ์ และ คณะ. "การทดลองเลี้ยงกุ้งกุลาดำด้วยอาหารเม็ด." เอกสารวิชาการฉบับที่ 4/2527. ฝ่ายทดลองและวิจัยเพื่อการเพาะเลี้ยง. กองประมงน้ำกร่อย กรมประมง. 2527.

25. มะลิ บุญยรัตนผลิน. "อาหารและการให้อาหารกุ้งกุลาดำ." เอกสารประกอบการอบรม. งานโภชนาการอาหาร. สถาบันประมงน้ำจืดแห่งชาติ กรมประมง. 2530.

26. _____. "การควบคุมคุณภาพน้ำ." เกษตรอุตสาหกรรม. ฉบับที่ 29 - 30. หน้า 83 - 84. 2530.

27. บุญส่ง ศิริกุล. "การศึกษาผลผลิตกุ้งกุลาดำในบ่อโดยการปล่อยลงเลี้ยงในอัตราความหนาแน่นต่างๆ." รายงานวิชาการ. ศูนย์การศึกษาการพัฒนาอ่าวคุ้งกระเบนอันเนื่องมาจากพระราชดำริ. สถานีประมงน้ำกร่อย จังหวัดจันทบุรี. 2523.

28. ศักดิ์ชัย โชติคุณ และ คณะ. "การทดลองเลี้ยงกุ้งกุลาดำในน้ำที่มีความเค็มระดับต่างๆ กัน." เอกสารวิชาการฉบับที่ 24/2530. สถานีพัฒนาการเลี้ยงกุ้ง จังหวัดระยอง. 2530.

29. บังอร ศรีมุกดา, "การอนุบาลลูกกุ้งกุลาดำโดยใช้เทคนิคที่ปรับปรุงแล้วให้ได้ผลผลิตสูง," รายงานวิชาการฉบับที่ 10/2528, สถานีประมงน้ำกร่อย จังหวัดระยอง, กองประมงน้ำกร่อย กรมประมง, 2528.

30. ลลิตา เรืองแป้น, "คู่มือการควบคุมและป้องกันโรคกุ้งทะเลสำหรับเกษตรกร," เอกสารเผยแพร่, คลินิคสัตว์น้ำกร่อย กองประมงน้ำกร่อย กรมประมง, 2529.

31. Wilensky, R., LISPcraft, W.W. Norton & Company, Inc., NY, 1984.

32. Winston, P.H., LISP, 2nd Edition, Addison-Wesley Publishing Company, Massachusette, 1984.

33. Anderson, J.R., A. T. Corbett, and B.J. Reiser, Essential LISP, Addison-Wesley Publishing Company, Massachusette, 1987.

34. Charniak, E., C.K. Riesbeck, and D.V. McDermott, Artificial Intelligence Programming, Lawrence Erlbaum Associates, New Jersey, 1980.

35. Hasemer, T., Looking at Lisp, Addison-Wesley Publishing Company, Singapore, 1984.

ภาคผนวก

ภาคผนวก ก.

ความรู้ทั่วไปเกี่ยวกับภาษา Franz Lisp

ภาษา Franz Lisp [31] เป็นภาษาที่ใช้ในงานด้านปัญญาประดิษฐ์เช่นเดียวกับภาษา Prolog และเป็นตระกูลหนึ่งของภาษาลิสป์ (Lisp) ซึ่งย่อมาจาก List Processing นับเป็นภาษาที่มีการพัฒนาขึ้นมาในรุ่นแรกเช่นเดียวกับภาษาฟอร์แทรน มีการพัฒนาอย่างต่อเนื่องและเพิ่มคุณสมบัติต่างๆ ทำให้สามารถใช้กับระบบคอมพิวเตอร์ได้หลายระบบ เช่น MacLisp พัฒนาขึ้นที่สถาบันเทคโนโลยีแห่งแมสสาชูเสท (MIT) InterLisp พัฒนาขึ้นที่ศูนย์วิจัยของซีรอก (Xerox Palo Alto Research Center) Franz Lisp ซึ่งพัฒนาขึ้นที่มหาวิทยาลัยแห่งแคลิฟอร์เนียที่เบอร์คเลย์ (University of California at Berkeley) ทำงานภายใต้ระบบปฏิบัติการยูนิกซ์ (Unix operating system) เป็นต้น

ในที่นี้จะขอกล่าวถึงฟังก์ชัน และหลักการทำงานพื้นฐานทั่วไปของภาษา Franz Lisp ซึ่งเป็นภาษาคอมพิวเตอร์ที่ใช้ในการวิจัยบนระบบคอมพิวเตอร์ VAX 11/750 ภายใต้ระบบปฏิบัติการ ULTRIX-32 การทำงานเป็นลักษณะอินเตอร์พรีตเตอร์ (interpreter) การเรียกใช้เมื่ออยู่ในเครื่องหมายเตรียมพร้อมของระบบ ULTRIX-32 ใช้ฟังก์ชัน lisp ระบบจะเรียกโมดูลต่างๆ ของภาษาและแสดงความพร้อมดังนี้

```
Franz Lisp, Opus 38.41   บอกหมายเลขเวอร์ชันของภาษา Franz Lisp
->                       แสดงเครื่องหมายเตรียมพร้อม
```

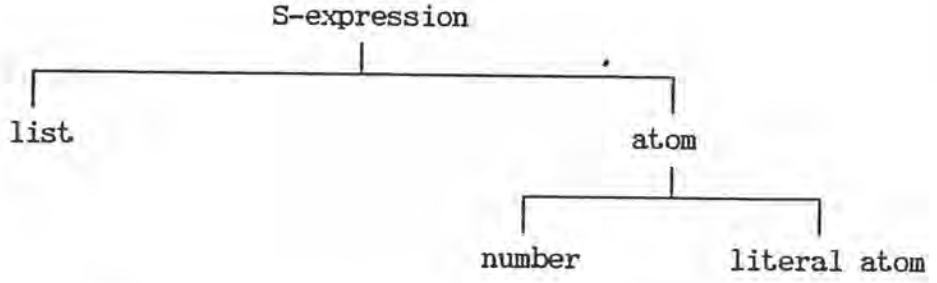
เมื่อต้องการเลิกการทำงานและกลับสู่ระบบปฏิบัติการให้พิมพ์ฟังก์ชัน (exit)

1. โครงสร้างข้อมูลของภาษา Franz Lisp [31,32,33,34,35]

ฟังก์ชันทุกฟังก์ชันจะอยู่ในเครื่องหมายวงเล็บเปิด ปิด ประกอบด้วย ตัวกระทำและอากิวเมนต์ของการทำงานเช่น 8 + 3 ใช้กับฟังก์ชัน plus ดังนี้

```
-> (plus 8 3)
11
```

ตัวกระทำ (operator) เป็นตัวกำหนดการทำงาน โดยมี 8 กับ 3 เป็นอากิวเมนต์ ฟังก์ชันทั่วๆ ไปในภาษา Franz Lisp จะเรียกว่า S-expression ซึ่งย่อมาจาก Symbolic expression หรือนิพจน์สัญญลักษณ์ แผนภาพแสดงโครงสร้างข้อมูลในภาษา Franz lisp แสดงในรูปที่ ก.1



รูปที่ ก.1 แสดงโครงสร้างข้อมูลในภาษา Franz lisp

การทำงานของอินเตอร์พริตเตอร์ภาษา Franz Lisp จะเป็นการประเมินผล (evaluate) นิพจน์สัญญลักษณ์แต่ละนิพจน์ โดยการทำงานจะมองจากวงเล็บนอกสุดว่ามี อากิวเมนต์กี่ตัวแล้วจึงทำการประเมินผลอากิวเมนต์ตัวในสุดก่อน ผลลัพธ์ที่ได้จะใช้เป็นอากิวเมนต์ของตัวปฏิบัติการที่อยู่ภายนอกต่อไป เช่น

```
-> (times 8 (plus 3 7))
80
->
```

จากตัวอย่างการทำงานจะเริ่มจากนำ 3 มาบวกกับ 7 ได้ผลลัพธ์เท่ากับ 10 จากนั้นใช้ 10 เป็นอากิวเมนต์ของการคูณร่วมกับค่า 8 ได้ผลลัพธ์เท่ากับ 80 เป็นต้น

1.1 อะตอม (Atom)

เป็นโครงสร้างข้อมูลพื้นฐานประเภทหนึ่งของภาษา Lisp มี 2 ประเภท คือ ตัวเลข และชื่อตัวแปร หรือสัญญลักษณ์ ค่าตัวเลขอาจเป็นค่าจำนวนเต็ม หรือ floating-point ก็ได้ สัญญลักษณ์ในภาษา Lisp มีทั้งที่เป็นชื่อฟังก์ชัน เช่น times, plus, หรือ add1 เป็นต้น เมื่อผู้ใช้กำหนดฟังก์ชันเหล่านี้ภาษา Lisp ก็จะมีการทำงานบางอย่างตามชนิดของฟังก์ชันนั้นๆ หรือ เป็นค่าคงที่ที่ใช้เป็นชื่อตัวแปร เช่น x, y, หรือ tree เป็นต้น เรา

สามารถทำการกำหนดค่าให้แก่อะตอมได้โดยใช้ฟังก์ชัน setq ไม่ว่าอะตอมนั้นจะเป็นชื่อตัวแปร หรือชื่อฟังก์ชัน แต่เมื่อมีการเรียกใช้ในตำแหน่งของฟังก์ชันก็จะทำงานตามชนิดของฟังก์ชันนั้นๆ เช่นเดิม

```
-> (setq plus 17)        กำหนดค่าให้ตัวแปรชื่อ plus มีค่าเท่ากับ 17
17
-> plus                  แสดงค่าของตัวแปร plus เท่ากับ 17
17
-> (plus 3 6)            เรียกใช้ฟังก์ชัน plus เพื่อทำการบวกค่า
9                        3 และ 6 ตามปกติ
```

ดังนั้นจะเห็นได้ว่าอะตอมสามารถเป็นได้ทั้ง function identifier และตัวแปรในภาษา Lisp

1.2 รายการ (List)

เป็นโครงสร้างข้อมูลที่ประกอบด้วยสมาชิกอยู่ภายในเครื่องหมายวงเล็บโดยมีการทำงานในลักษณะการปฏิบัติการตัวสัญญลักษณ์ (Symbolic operation) สมาชิกที่อยู่ในรายการอาจเป็นอะตอม หรือตัวแปร หรือเป็นรายการซ้อนอยู่ก็ได้ เช่น (a b c) เป็นรายการที่มีสมาชิก 3 ตัว (a (b c) d) เป็นรายการที่มีสมาชิก 3 ตัว โดยสมาชิกตัวที่ 2 เป็นรายการที่มีสมาชิก 2 ตัวคือ a และ b เป็นต้น ฟังก์ชันพื้นฐานที่ใช้ในการประมวลผลโครงสร้างข้อมูลแบบรายการได้แก่ ฟังก์ชัน car และ cdr ซึ่งเป็นฟังก์ชันที่ใช้ในการแยกสมาชิกของรายการออกมา และฟังก์ชัน append ฟังก์ชัน list และ ฟังก์ชัน cons ซึ่งใช้รวมค่าอะตอม หรือรายการอื่นๆ ขึ้นเป็นรายการใหม่ ปกติภาษา Lisp จะมีการทำงานในลักษณะ READ-EVAL-PRINT loop เริ่มจากทำการอ่านค่านิพจน์เข้ามา แล้วทำการประเมินผลนิพจน์นั้นตามฟังก์ชันที่กำหนดและพิมพ์ผลการประเมินผลออกมา เช่น

```
-> (+ 1 2)               จะทำการประเมินผลเมื่อพบฟังก์ชัน +
3                        ทำการบวกค่าอากิวเมนต์เข้าด้วยกัน
-> (a b c)               เมื่อทำการประเมินผลพบว่า a ไม่ใช่ค่า
Error : eval : undefined function a ฟังก์ชันจะแสดงข้อความ
                         ผิดพลาดให้ผู้ใช้ทราบ
<1> : (reset)            ผู้ใช้พิมพ์คำว่า reset เพื่อกลับเข้าสู่สภาวะ
```

[Return to top level] การทำงานปกติของภาษา Lisp

จะเห็นได้ว่าเมื่อทำการประเมินผลและพบค่าที่ไม่ใช่ชื่อฟังก์ชันในภาษา Lisp เป็นอากิวเมนต์ตัวแรกของนิพจน์จะถือว่าเป็นนิพจน์ที่ผิด ดังนั้นถ้าเราต้องการพิมพ์ค่าของรายการ (a b c) เข้าสู่ระบบต้องใช้ฟังก์ชัน quote หรือ " ' " ไว้หน้ารายการ เพื่อยกเลิกการประเมินผลอากิวเมนต์ตัวแรกของนิพจน์ และถือว่านิพจน์นั้นเป็นค่าคงที่ทั้งหมด

```
-> (quote (a b c))
(a b c)
-> (setq x '(a b c))   เป็นการกำหนดค่าให้ตัวแปร x โดยใช้ฟังก์ชัน
(a b c)                setq ซึ่งมีความหมายเท่ากับ set quote
```

1.2.1 ฟังก์ชัน car และ cdr

เป็นฟังก์ชันที่ใช้ในการแยกสมาชิกออกจากรายการ มี อากิวเมนต์ที่กำหนด 1 ตัว และต้องเป็นอากิวเมนต์ชนิดรายการ ฟังก์ชัน car จะแยกสมาชิกตัวแรกของรายการออกมา ในขณะที่ฟังก์ชัน cdr จะแสดงรายการที่เหลือหลังจากแยกสมาชิกตัวแรกออกจากรายการแล้ว ผลการใช้งานฟังก์ชันทั้งสองจะไม่เกิดการเปลี่ยนแปลงค่าของรายการที่เป็น อากิวเมนต์ ซึ่งถือว่าฟังก์ชันทั้งสองมีคุณสมบัติ non-destructive เช่น

```
-> (setq x '(a b c))      กำหนดให้ตัวแปร x มีค่า
(a b c)                   รายการ (a b c)
-> x                      กำหนดให้มีการแสดงค่า x
(a b c)
-> (car x)                เรียกใช้ฟังก์ชัน car กับตัวแปร x
a                         เพื่อแยกสมาชิกตัวแรกของรายการ
-> (cdr x)                เรียกใช้ฟังก์ชัน cdr กับตัวแปร x
(b c)                     เพื่อแสดงรายการที่เหลือหลังจากแยก
                          สมาชิกตัวแรกออกแล้ว
-> x                      พิมพ์ค่าของตัวแปร x
(a b c)                   จะได้รายการเดิมที่กำหนดให้ตัวแปร x
```

ผู้ใช้สามารถกำหนดฟังก์ชัน car และฟังก์ชัน cdr เข้าด้วยกัน เพื่อใช้แยกสมาชิกตัวที่ต้องการออกจากรายการที่กำหนด เช่น ฟังก์ชัน cadr หมายถึงเรียกใช้ฟังก์ชัน cdr กับรายการก่อนจากนั้นจึงใช้ฟังก์ชัน car กับรายการที่ได้นั้นต่อไป

```
-> (cadr '(a b c))
b                    ผลลัพธ์ที่ได้คือสมาชิกตัวที่ 2 ของรายการ
```

1.2.2 รายการที่ไม่มีสมาชิก (Empty list)

หมายถึงรายการที่มีความยาวเท่ากับ 0 หรือเป็น () เกิดขึ้นเมื่อใช้ฟังก์ชัน cdr กับรายการที่มีสมาชิกเพียง 1 ตัว หรือ ผู้ใช้กำหนดค่ารายการว่างนี้ให้ตัวแปรภาษา Lisp จะแสดงค่า nil ซึ่งหมายถึง รายการว่าง เช่น

```
-> (cdr '(c))
nil
```

1.3 ฟังก์ชันที่ใช้สร้างรายการ

มีฟังก์ชันที่สำคัญดังนี้คือ cons list และ append

- ฟังก์ชัน cons เป็นฟังก์ชันที่ใช้อากิวเมนต์2 ตัวและอากิวเมนต์ตัวที่ 2 ต้องเป็นรายการ จะให้ผลลัพธ์เป็นรายการใหม่ที่เกิดจากการนำเอาอากิวเมนต์ตัวแรกใส่เป็นสมาชิกตัวแรกของอากิวเมนต์ตัวที่ 2

- ฟังก์ชัน list เป็นฟังก์ชันที่ต้องใช้อากิวเมนต์กี่ค่าก็ได้ และเป็นอะตอมหรือรายการก็ได้ ผลลัพธ์ที่ได้จะเป็นการสร้างรายการใหม่ซึ่งประกอบด้วยอากิวเมนต์แต่ละตัวเป็นสมาชิก

- ฟังก์ชัน append เป็นฟังก์ชันที่ต้องใช้อากิวเมนต์กี่ตัวก็ได้ แต่อากิวเมนต์ ทุกตัวต้องเป็นรายการ ผลลัพธ์ที่ได้จะเป็นการสร้างรายการขึ้นใหม่ โดยใส่อากิวเมนต์ แต่ละตัวเป็นสมาชิกของรายการที่สร้างขึ้นใหม่

ตัวอย่างการใช้งานฟังก์ชันที่สร้างรายการ เช่น

```
-> (cons 'a '(b c))   อากิวเมนต์ตัวที่ 2 ของฟังก์ชัน cons
(a b c)               เป็นรายการได้รายการใหม่ที่มี อากิวเมนต์
```

แรกเป็นสมาชิกตัวแรกของรายการ

-> (list 'a '(b c) 'd) ฟังก์ชัน list ใช้อากิวเมนต์กี่ตัวก็ได้
(a (b c) d) และอากิวเมนต์แต่ละตัวเป็นอะตอม หรือ
รายการก็ได้

-> (append '(a b) '(c d) '(e f)) ฟังก์ชัน append ต้องการ
(a b c d e f) อากิวเมนต์ทุกตัวเป็นรายการ

2. การกำหนดฟังก์ชันเอง

ในภาษา Franz Lisp การเขียนโปรแกรมจะทำได้โดยกำหนดฟังก์ชันที่มีการทำงานตามโปรแกรมที่ต้องการ โดยใช้ฟังก์ชันพื้นฐานต่างๆ ของภาษามาประกอบกัน และเรียกใช้งานฟังก์ชันนั้นเช่นเดียวกับฟังก์ชันอื่นๆ การกำหนดฟังก์ชันทำโดยใช้ฟังก์ชัน defun ซึ่งหมายถึง "define function"

2.1 ฟังก์ชัน defun

การเรียกใช้ประกอบด้วยอากิวเมนต์ซึ่งเป็นชื่อของฟังก์ชันที่กำหนดขึ้น รายการของพารามิเตอร์ที่ใช้กับฟังก์ชัน และขั้นตอนการทำงานของฟังก์ชันซึ่งมีการเรียกใช้ฟังก์ชันอื่นๆ หรือ ฟังก์ชันที่ใช้ในการเปรียบเทียบ หรือ ฟังก์ชันที่ใช้ทดสอบเงื่อนไขต่างๆ เป็นต้น

รูปแบบการกำหนดฟังก์ชันขึ้นใช้งานประกอบด้วย

```
(defun fname (v1 v2 .. vn)
  (... body of code 1...)
  (... body of code 2...)
  (... body of code n...)
)
```

หลังจากกำหนดฟังก์ชันแล้วเราสามารถเรียกใช้งานได้โดยฟังก์ชัน

```
(fname arg1 arg2 arg3 ... argn)
```

ตัวอย่างเช่นการกำหนดฟังก์ชัน addthree เป็นฟังก์ชันที่ทำการบวกค่า 3 เข้ากับพารามิเตอร์ที่กำหนด สามารถสร้างฟังก์ชันได้โดย

-> (defun addthree (x)	เป็นการเรียกใช้ฟังก์ชัน defun เพื่อ
(plus x 3))	ทำการบวกค่า 3 กับพารามิเตอร์ที่กำหนด
addthree	แสดงชื่อฟังก์ชันที่กำหนดใหม่
-> (addthree 15)	การเรียกใช้ฟังก์ชัน addthree ที่กำหนด
18	กับพารามิเตอร์ค่า 15

2.2 ฟังก์ชันเงื่อนไข และฟังก์ชันตรรกศาสตร์

ภาษา Lisp จะมีฟังก์ชันกลุ่มหนึ่งทำหน้าที่เป็น predicate ใช้ตรวจสอบคุณสมบัติต่างๆ ของข้อมูลซึ่งเมื่อใช้ร่วมกับฟังก์ชันทางตรรกศาสตร์ not and และ or ทำให้มีการตรวจสอบเงื่อนไขต่างๆ ที่ซับซ้อนได้ และเป็นวิธีที่ใช้ควบคุมทิศทางการทำงานภายในฟังก์ชัน

2.2.1 ฟังก์ชันที่ใช้ตรวจสอบคุณสมบัติต่างๆ ของข้อมูล

ในการควบคุมการทำงานใดๆ ก็ตามจำเป็นต้องมีการตรวจสอบค่าของข้อมูล กลุ่มของฟังก์ชันที่ทำหน้าที่นี้เรียกว่า predicate โดยผลการตรวจสอบจะให้ค่าจริงหรือเท็จ ในภาษา Lisp จะแสดงค่าเท็จด้วยค่า nil หรือ รายการว่าง และแสดงค่าเป็นจริงด้วยค่าอื่นใดที่ไม่ใช่ nil หรือ อะตอม t predicate ที่ใช้งานบ่อย เช่น

- atom ใช้ตรวจสอบว่านิพจน์สัญญลักษณ์นั้นเป็นอะตอมหรือไม่ ถ้าเป็นอะตอมจะให้ค่า t ถ้าไม่ใช่จะให้ค่า nil

- listp ใช้ตรวจสอบว่าอากิวเมนต์ของฟังก์ชันเป็นรายการหรือไม่ ถ้าเป็นจริงจะให้ค่า t ถ้าเป็นเท็จจะให้ค่า nil

- null ใช้ตรวจสอบว่า ข้อมูลเป็น nil หรือไม่ ถ้าเป็นจริงจะให้ค่า t ถ้าเป็นเท็จจะให้ค่า nil

- equal ใช้ตรวจสอบว่านิพจน์สัญญลักษณ์เท่ากันหรือไม่ ถ้าเป็นจริงจะให้ค่า t ถ้าเป็นเท็จจะให้ค่า nil

- numbp ใช้ตรวจสอบว่าอากิวเมนต์ของฟังก์ชัน เป็นตัวเลขหรือไม่ ถ้าเป็นจริงจะให้ค่า t ถ้าเป็นเท็จจะให้ค่า nil

- member เป็นฟังก์ชันตรวจสอบ มีอากิวเมนต์2 ตัว และอากิวเมนต์ ตัวที่ 2 ต้องเป็นรายการ ใช้ในการตรวจสอบว่าอากิวเมนต์ตัวแรกเป็นสมาชิกของอากิวเมนต์ รายการตัวที่ 2 หรือไม่ ถ้าเป็นจริงจะแสดงรายการของอากิวเมนต์ตัวที่ 2 โดยเริ่มจากสมาชิกตัวแรกที่เป็นอากิวเมนต์ตัวแรก ถ้าเป็นเท็จจะให้ค่า nil

ตัวอย่างการใช้ฟังก์ชันเหล่านี้เช่น

```
-> (atom a)
t
-> (atom '(a b c))
nil
-> (listp 'a)
nil
-> (listp (cdr '(a b c)))
t
-> (equal '(a b c) '(a b c))
t
-> (equal 'a 'b)
nil
-> (numbp 6)
t
-> (member 'b '(a b c))
(b c)
-> (member '(a b) (a b c))
nil
```

นอกจากนี้ยังมี predicate ที่ใช้ตรวจสอบค่าตัวเลขอีกเช่น zerop, oddp, evenp, lessp และ greaterp เป็นต้น

2.2.2 ฟังก์ชันเงื่อนไข

ฟังก์ชันที่ช่วยเลือกขั้นตอนการทำงานหลังจากทำการตรวจสอบคุณสมบัติของข้อมูลด้วยฟังก์ชัน cond ในลักษณะคล้ายกับฟังก์ชัน "If..then..else" และมีอากิวเมนต์กี่ตัวก็ได้ อากิวเมนต์แต่ละตัวเรียกว่าประโยคเงื่อนไข (cond clause) ประกอบด้วยชุดของนิพจน์สัญญลักษณ์ สมาชิกตัวแรกของประโยคเงื่อนไขแต่ละประโยคคือ เงื่อนไขที่ต้องทำการตรวจสอบ ส่วนที่เหลือคือ สิ่งที่ต้องการทำหลังจากพบว่าเงื่อนไขนั้นเป็นจริง เช่นตัวอย่างการกำหนดฟังก์ชัน cond-example

```
-> (defun cond-example (x)  กำหนดฟังก์ชันชื่อ
     (cond          cond-example มีพารามิเตอร์ 1 ค่า
       ((listp x) (cons 'a x))  ทำการตรวจสอบเงื่อนไข
       ((numbp x) (plus 7 x)) ว่าถ้าพารามิเตอร์เป็นรายการ
       ))          ทำการใส่ a เป็นสมาชิกตัวแรกของรายการ
                   ถ้าพารามิเตอร์เป็นตัวเลขให้เพิ่มค่าโดยบวก 7
                   นอกนั้นให้ค่า 'nil
cond-example
-> (cond-example '(b c))
(a b c)
-> (cond-example 9)
16
-> (cond-example 'z)
nil
->
```

รูปแบบทั่วไปของฟังก์ชัน cond สามารถแสดงได้ดังนี้

```
(cond (exp11 exp12 exp13 ...)
      (exp21 exp22 exp23 ...)
      (exp31 exp32 exp33 ...)
      .
      (expn1 expn2 expn3 ...))
```

โดยทั่วๆ ไปการประเมินผลฟังก์ชัน cond จะเริ่มตรวจสอบประโยคเงื่อนไขแรก โดยตรวจสอบสมาชิกตัวแรกถ้าเป็นจริงคือ ไม่ใช่ nil ภาษา Lisp จะเลือกทำงานในส่วนที่เหลือของประโยคเงื่อนไขจนจบ แล้วส่งค่าที่ได้จากการประเมินผลนิพจน์สุดท้ายกลับมา ส่วนที่เหลือของฟังก์ชัน cond จะไม่ถูกเรียกใช้งาน แต่ถ้าสมาชิกตัวแรกของประโยคเงื่อนไขเป็นเท็จ ภาษา Lisp จะไม่เลือกทำงานสมาชิกที่เหลือของประโยคเงื่อนไขนั้น แต่จะหาประโยคเงื่อนไขต่อไปและทำการประเมินผลซ้ำ ถ้าไม่มีประโยคเงื่อนไขใดในฟังก์ชัน cond เป็นจริงเลยก็จะส่งค่า nil กลับมา

2.2.3 ฟังก์ชันตรรกศาสตร์

ใช้ร่วมกับ predicate เพื่อทำการตรวจสอบคุณสมบัติที่ซับซ้อน ฟังก์ชันทางตรรกศาสตร์คือ not, and และ or มีอากิวเมนต์คือ ค่าจริง-เท็จ และประเมินผลให้ค่าจริง-เท็จ เช่นกัน

- not จะส่งค่า t ถ้าอากิวเมนต์เป็นเท็จ และส่งค่า nil ถ้าอากิวเมนต์เป็นจริง

- and เป็นฟังก์ชันที่มีอากิวเมนต์กี่ตัวก็ได้ แต่ละตัวจะถูกประเมินผลเรียงตามลำดับจนกว่าจะพบอากิวเมนต์ที่ให้ค่า nil ซึ่งฟังก์ชันจะส่งค่า nil กลับ แต่ถ้าทำการประเมินผลจนหมด แต่ไม่มีอากิวเมนต์ใดให้ค่า nil ก็จะส่งค่าอากิวเมนต์สุดท้ายกลับมา

- or จะมีอากิวเมนต์กี่ตัวก็ได้เช่นกัน แต่จะหยุดการทำงานเมื่อพบว่าอากิวเมนต์ตัวแรกที่เป็นจริงและจะส่งค่าอากิวเมนต์นั้นกลับมา นอกนั้นจะส่งค่า nil กลับมา

ตัวอย่างการใช้ฟังก์ชันตรรกศาสตร์ เช่น

```
-> (not (atom x))  ถ้า x เป็นตัวแปรชนิดรายการ
t
-> (or (null x) (numbp x))  ถ้าต้องการทราบค่า x
nil                เป็นรายการว่าง หรือตัวเลขหรือไม่ เมื่อ x
t                  เป็นรายการจะส่งค่า nil ถ้าเป็นตัวเลข
                   หรือ รายการว่างจะให้ค่า t
-> (and (evenp x) (lessp x 100))
ใช้ตรวจสอบว่า x เป็นเลขคี่ที่น้อยกว่า 100 หรือไม่
```

3. การเวียนเกิด (Recursion)

ฟังก์ชันที่ทำงานในลักษณะการเวียนเกิดคือ ฟังก์ชันที่เรียกใช้งานตัวเองในกรณีที่ต้องการประมวลผลสมาชิกในรายการที่มีความยาวไม่แน่นอน ต่างจากการทำซ้ำ (iteration) ซึ่งมีจำนวนสมาชิกแน่นอน

3.1 หลักการพื้นฐานของการเวียนเกิด

ตัวอย่างการกำหนดฟังก์ชันที่มีลักษณะการเวียนเกิดในฟังก์ชัน เช่น

```
-> (defun our-member (e l)          กำหนดฟังก์ชัน our-member
     (cond [(null l) ()]            ตรวจสอบจุดสิ้นสุดการเวียนเกิด
           [(equal e (car l)) l]    ตรวจสอบหาสมาชิกที่เท่ากัน
           [t (our-member e (cdr l))])) เรียกใช้ฟังก์ชัน
                                    our-member แบบเวียนเกิด
```

หลักการกำหนดฟังก์ชันที่มีการเวียนเกิดคือ ขั้นแรกต้องตรวจสอบจุดสิ้นสุดของการเวียนเกิด ถ้ายังไม่สิ้นสุดให้มีการทำงานในส่วนต่อไปของฟังก์ชันนั้น และสุดท้ายจะเรียกใช้ฟังก์ชันนั้นแบบเวียนเกิด กับสมาชิกตัวต่อไปที่กำหนดในรายการพารามิเตอร์ ข้อผิดพลาดที่พบบ่อยคือ การตรวจสอบจุดสิ้นสุดการเวียนเกิดทำให้การเรียกใช้ฟังก์ชันเกิดขึ้นอย่างไม่สิ้นสุด

ตัวอย่างการใช้ฟังก์ชันแบบการเวียนเกิดที่เห็นได้ชัดคือ การกำหนดฟังก์ชันแฟคตอเรียล (factorial) ซึ่งกำหนดได้ดังนี้

```
-> (defun factorial (n)
     (cond [(zerop n) 1]
           [t (times n (factorial (sub1 n)))]))
factorial
-> (factorial 3)
6
```

ฟังก์ชัน factorial จะตรวจสอบขั้นแรกว่าค่าของพารามิเตอร์ n เป็น 0 หรือไม่ ถ้าใช่จะส่งค่า 1 ซึ่งถือเป็นจุดสิ้นสุดของการเวียนเกิด แต่ถ้าไม่ใช่ก็จะมีการเรียกใช้ฟังก์ชัน factorial อีกครั้ง แต่ใช้พารามิเตอร์เท่ากับ n-1 หลังจากนั้นฟังก์ชัน factorial จะทำการคูณผลลัพธ์ด้วย n ได้เป็นค่า factorial ของ n

4. รายการคุณสมบัติ (property list)

โครงสร้างข้อมูลพื้นฐานของภาษา Lisp คือ อะตอม ซึ่งสามารถทำการกำหนดค่าบางอย่างได้เรียกว่า คุณสมบัติ (property) เช่นเดียวกับวัตถุทั่วไปตัวอย่างเช่น เก้าอี้จะมีลักษณะต่างๆ คือ สี น้ำหนัก ความสูง สำหรับเก้าอี้ตัวที่เฉพาะเจาะจงอาจเป็นเก้าอี้ที่มีสีน้ำเงิน มีน้ำหนัก 40 ปอนด์ และสูง 3 ฟุต เป็นต้น

4.1 การกำหนดคุณสมบัติ และการเรียกใช้คุณสมบัติที่กำหนดให้อะตอม

อะตอมสามารถที่จะกำหนดคุณสมบัติกี่อันก็ได้ แต่ละคุณสมบัติจะมีค่าของมันเอง การกำหนดคุณสมบัติให้อะตอมทำได้โดยใช้ฟังก์ชัน putprop ซึ่งมีอากิวเมนต์ 3 ค่าคือ ชื่ออะตอม ค่าของคุณสมบัติ และชื่อคุณสมบัติที่กำหนดให้อะตอมนั้น ตัวอย่างเช่น ถ้าต้องการกำหนดให้อะตอมชื่อ chair3 มีคุณสมบัติสีน้ำเงินโดย

```
-> (putprop 'chair3 'blue 'color)
blue
->
```

การเรียกใช้คุณสมบัติต่างๆ ที่กำหนดให้แก่อะตอมทำได้โดยเรียกใช้ฟังก์ชัน get ซึ่งมีอากิวเมนต์ 2 ค่าคือ ชื่ออะตอม และชื่อคุณสมบัติที่ต้องการ เช่นถ้าต้องการเรียกดูว่าอะตอมชื่อ chair3 มีสีอะไร จะทำได้โดย

```
-> (get 'chair3 'color)
blue
->
```

ภาษา Lisp จะรวบรวมคุณสมบัติต่างๆ ที่กำหนด และค่าที่เกี่ยวข้องไว้เป็นรายการคุณสมบัติ (property list) ของแต่ละอะตอม ตัวอย่างการเรียกใช้รายการคุณสมบัติในฟังก์ชันเช่น ฟังก์ชัน add-book ใช้เก็บข้อมูลหนังสือแต่ละเล่มประกอบด้วย ชื่อหนังสือ ชื่อผู้แต่ง สำนักพิมพ์ เป็นต้น

```
-> (defun add-book (bookref title author publisher)
     (putprop bookref title 'title)
     (putprop bookref author 'author)
     (putprop bookref publisher 'publisher)
     (setq Library (cons bookref Library))
   bookref)
add-book
```

อะตอม bookref จะเป็นตัวแปรที่แทนหนังสือแต่ละเล่ม เช่น book1, book2 เป็นต้น ฟังก์ชัน add-book จะเก็บค่าอากิวเมนต์ต่างๆ ในรายการคุณสมบัติของอะตอมนี้ และใส่ชื่ออะตอมนี้ลงในตัวแปรรายการชื่อ Library

```
-> (setq Library ())
nil
-> (add-book 'book1
             '(War and peace)
             '(Leo Tolstoy)
             '(Frumdeedump Press))
book1
-> (add-book 'book2
             '(Artificial Intelligence)
             '(Patrick Winston)
             '(Addison-Wesley))
book2
-> Library
(book1 book2)
```

5. ฟังก์ชันที่ใช้รับค่า และแสดงค่า

ตามปกติภาษา Lisp จะรับค่าข้อมูลเข้าสู่ระบบทางรายการพารามิเตอร์ที่กำหนดในฟังก์ชัน และแสดงค่าของตัวแปรให้ผู้ใช้ระบบทราบเฉพาะค่าที่ได้จากการประเมินผลนิพจน์สัญญลักษณ์สุดท้ายของแต่ละฟังก์ชัน ในกรณีที่ผู้ใช้ต้องการใส่ค่าเข้าสู่ฟังก์ชันระหว่างการทำงาน หรือต้องการแสดงค่าของพารามิเตอร์ที่สนใจระหว่างการทำงานของระบบ ภาษา Lisp มีฟังก์ชันที่ใช้ทำหน้าที่นี้คือ ฟังก์ชัน read และ ฟังก์ชัน print

- ฟังก์ชัน read เป็นฟังก์ชันที่ไม่มีอากิวเมนต์ใช้ในฟังก์ชันเพื่อให้ระบบหยุดรอรับค่าจากผู้ใช้ แต่จะไม่แสดงข้อความใดๆ บอก มักใช้ร่วมกับฟังก์ชัน setq เพื่อกำหนดค่าที่รับมาให้แก่ตัวแปรสำหรับการทำงานอื่นต่อไปเช่น

```
-> (setq read-in (read)) ทำการรับค่าตัวแปร read-in จากผู้ใช้
```

```
5                              ระบบจะรอให้ผู้ใช้ใส่ค่า และกด enter
5                              ระบบแสดงค่าที่รับจากผู้ใช้
-> read-in                     แสดงค่าตัวแปร read-in ที่รับเข้ามา
5
```

ฟังก์ชันที่ใช้แสดงค่าเป็นฟังก์ชันที่ใช้แสดงข้อความที่เป็น string หรือค่าของตัวแปร มีฟังก์ชัน print และ terpri

- ฟังก์ชัน print จะเป็นการพิมพ์นิพจน์สัญญลักษณ์ที่กำหนดโดยไม่มี new line character
- ฟังก์ชัน terpri จะทำการส่ง newline character เพื่อทำการขึ้นบรรทัดใหม่

ตัวอย่างการใช้ฟังก์ชันรับค่าและแสดงค่าร่วมกัน เช่น

```
-> (defun enter-number ()      กำหนดฟังก์ชัน enter-number
     (print '(please enter number :))  แสดงข้อความบอกให้
     (print (sqrt (read)))     ใส่ค่าตัวเลขและรับค่าจากผู้ใช้ พิมพ์
     (terpri)                  ค่าที่ได้จากการทำงานและขึ้นบรรทัดใหม่
     (enter-number))           เรียกใช้ฟังก์ชัน enter-number อีกครั้ง
enter-number
-> (enter-number)              เริ่มเรียกใช้งานฟังก์ชัน enter-number
(Please enter number :) 25     ผู้ใช้ใส่ค่า 25
5.0                            พิมพ์ค่ารากที่สองของ 25
(Please enter number :) 38     ผู้ใช้ใส่ค่า 38
6.164414002968976              พิมพ์ค่ารากที่สองของ 38
(Please enter number :) Interrupt : ^C  ผู้ใช้ยุติการทำงานเวียนเกิด
Break nil                              โดยกด Ctrl-C เพื่อออกจาก
<1> :                                  การทำงาน
```

ฟังก์ชันที่รับค่า และแสดงค่าจะมีอีกหลายฟังก์ชันแล้วแต่ลักษณะของข้อมูลที่ต้องการ การใช้งานฟังก์ชันต่างๆ นั้นสามารถดูได้จากเอกสารอ้างอิงในวิทยานิพนธ์ฉบับนี้

ภาคผนวก ข

รายชื่อตัวแปรและฟังก์ชันที่ใช้ในระบบเชี่ยวชาญ SFES

จากการออกแบบและการสร้างระบบ SFES ในบทที่ 3 และ บทที่ 4 จะเห็นได้ว่ามีการกำหนดฟังก์ชันต่างๆ จำนวนมากในระบบ SFES ตามลักษณะการพัฒนาโปรแกรมในภาษา Lisp ดังนั้นเพื่อช่วยในการศึกษาการทำงานจะได้สรุปรายละเอียดของแต่ละโปรแกรมย่อยพร้อมทั้งหน้าที่ของฟังก์ชันเหล่านั้นในตารางที่ ข.1 ถึง ข.6 จากนั้นจะเป็นรายชื่อฟังก์ชันต่างๆ และหน้าที่ของฟังก์ชันเหล่านั้นอย่างย่อๆ ลงในตารางที่ ข.7 โดยเรียงตามลำดับอักษร ส่วนรายชื่อตัวแปรที่ใช้ในระบบ SFES ได้ทำการสรุปไว้ในตารางที่ ข.8 รายการตารางต่างๆ ในภาคผนวก ข มีดังนี้

1. ตารางที่ ข.1 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย START
2. ตารางที่ ข.2 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย INFER
3. ตารางที่ ข.3 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย MONITOR
4. ตารางที่ ข.4 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย FINDOUT
5. ตารางที่ ข.5 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย EXPLANATION
6. ตารางที่ ข.6 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย PRT-EXPN-RULE
7. ตารางที่ ข.7 ดัชนีรายชื่อฟังก์ชันทั้งหมดในระบบ SFES
8. ตารางที่ ข.8 ดัชนีรายชื่อตัวแปรในระบบ SFES

ชื่อฟังก์ชัน	การทำงาน
โปรแกรมย่อย START	เริ่มต้นการทำงานของระบบ SFES
ฟังก์ชัน INSTRUCTION-N	เว้นการแนะนำไปเลือกการทำงานต่อไป
ฟังก์ชัน INSTRUCTION-Y	แนะนำ keywords ต่างๆ และความหมาย
ฟังก์ชัน PROBLEM	ถามรายละเอียดเบื้องต้นและระยะของปัญหาที่เกิดขึ้น

ตารางที่ ข.1 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย START

ชื่อฟังก์ชัน	การทำงาน
โปรแกรมย่อย INFER	กำหนดตัวแปรต่างๆ และเรียกประเมินผลกฎเป้าหมายเพื่อเริ่มการทำงาน
ฟังก์ชัน CNTXT-BINDING	กำหนดชื่อเนื้อความใหม่ให้เนื้อความที่เกี่ยวข้อง
ฟังก์ชัน EXEC-GOAL-RULE	เรียกประเมินผลกฎเป้าหมายของระบบ
ฟังก์ชัน EXPLANATION	อธิบายการทำงานของระบบ
ฟังก์ชัน MAKE-CNTXT-TOP	สร้างจุดเนื้อความจุดแรกในต้นไม้เนื้อความ
ฟังก์ชัน PROMPT-END-MESSAGE	แสดงสถานะการทำงานหลังจากทำงานเสร็จ
ฟังก์ชัน REM-SYM	ลบชื่อเนื้อความที่เคยสร้างขึ้นในตารางสัญญลักษณ์ออก
ฟังก์ชัน RULE-CHECK	ตรวจสอบกฎความรู้อย่างละเอียด
ฟังก์ชัน RULE-INTERPRETER	เรียกประเมินผลกฎความรู้ต่างๆ ที่เกี่ยวข้องกับปัญหา

ตารางที่ ข.2 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย INFER

ชื่อฟังก์ชัน	การทำงาน
โปรแกรมย่อย MONITOR	ประเมินผลส่วนหลักฐานและส่วนกระทำของกฎ
ฟังก์ชัน ADD-TREATMENT	ใส่ประโยควิธีการแก้ไขปัญหาในรายการคำแนะนำ
ฟังก์ชัน CHECK-REQ-PARM	ตรวจสอบชุดพารามิเตอร์ที่ต้องหาค่า
ฟังก์ชัน CNTXT-REPLACE	แทนค่าชนิดเนื้อความด้วยชื่อเนื้อความที่กำหนดไว้
ฟังก์ชัน CONCLUDE-WORK	สรุปค่าความจริงที่เกี่ยวข้องสาเหตุของปัญหาใส่ในรายการสรุปสาเหตุของปัญหา
ฟังก์ชัน CREATE-CURR-CNTXT	กำหนดเนื้อความที่เกี่ยวข้องกับการทำงาน
ฟังก์ชัน DEL-REQ-RULE	ลบหมายเลขกฎที่ตรวจสอบแล้วออกจากรายการกฎที่สนใจ
ฟังก์ชัน EVALUATE	ประเมินผลค่าคำตอบที่ได้จากผู้ใช้ระบบ
ฟังก์ชัน EVAL-NO	กำหนดค่าตัวแปรต่างๆ ที่เกี่ยวข้องเมื่อประโยคเงื่อนไขในส่วนหลักฐานเป็นเท็จ
ฟังก์ชัน EVAL-YES	กำหนดค่าตัวแปรต่างๆ ที่เกี่ยวข้องเมื่อประโยคเงื่อนไขในส่วนหลักฐานเป็นจริง
ฟังก์ชัน GET-HYPOTHESIS	เรียกใช้ประโยคสมมติฐานที่เกี่ยวข้องกับประโยคเงื่อนไขมาทำการตรวจสอบ
ฟังก์ชัน MONITOR-ACTION	ตรวจสอบและใช้งานส่วนกระทำของกฎ
ฟังก์ชัน MONITOR-PREMISE	ประเมินผลส่วนหลักฐานของกฎ
ฟังก์ชัน PROMPT-DIAG	เตรียมแสดงข้อความสาเหตุของปัญหา
ฟังก์ชัน RECOMMEND-WORK	เรียกทำการแสดงสาเหตุของปัญหาที่สรุปได้
ฟังก์ชัน TREATMENT-WORK	ใส่ประโยควิธีการแก้ไขปัญหาลงในรายการคำแนะนำ

ตารางที่ ข.3 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย MONITOR

ชื่อฟังก์ชัน	การทำงาน
โปรแกรมย่อย FINDOUT	หาค่าของพารามิเตอร์และกำหนดชื่อเนื้อความ
ฟังก์ชัน ADD-HYPOTHESIS	ใส่ประโยคสมมติฐานลงในหน่วยความจำที่เก็บเหตุการณ์
ฟังก์ชัน ANSWER-CHECK	ตรวจสอบความถูกต้องของคำตอบ
ฟังก์ชัน ASK	ถามคำถามเกี่ยวกับพารามิเตอร์
ฟังก์ชัน CHECK-HAVE-TREE	ตรวจสอบว่ามีจุดเนื้อความในต้นไม้เนื้อความหรือไม่
ฟังก์ชันCHECK-TREE-PROPERTY	ตรวจสอบว่าชื่อเนื้อความนั้นเคยสร้างถูกสร้างหรือไม่
ฟังก์ชัน CNTXT-BINDING	เพิ่มชื่อเนื้อความที่สร้างใหม่ลงในรายการชื่อที่กำหนดให้เนื้อความชนิดต่างๆ
ฟังก์ชัน DISPLAY-WHY	อธิบายขั้นตอนการทำงานของระบบ
ฟังก์ชัน GET-HYPOTHESIS	เรียกประโยคสมมติฐานมาตรวจสอบกับประโยคเงื่อนไข
ฟังก์ชัน MAKE-CNTXT-NODE	สร้างชื่อจุดเนื้อความใหม่
ฟังก์ชัน MAKE-CNTXT-TREE	ใส่จุดเนื้อความที่สร้างใหม่ลงในตำแหน่งที่เหมาะสมในต้นไม้เนื้อความ
ฟังก์ชัน PRINT-TEXT	พิมพ์ข้อความต่างๆ โดยตรวจสอบความยาวไม่เกิน 70
ฟังก์ชัน WRONG-ANSWER	แสดงข้อความว่าคำตอบของผู้ใช้ไม่ถูกต้อง

ตารางที่ ข.4 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย FINDOUT

ชื่อฟังก์ชัน	การทำงาน
โปรแกรมย่อย EXPLANATION	อธิบายขั้นตอนการทำงานของระบบ
ฟังก์ชัน CNCLD-EXPN	อธิบายสาเหตุของปัญหาที่สรุปได้
ฟังก์ชัน EXPN-DISP	แสดงข้อความอธิบายการทำงาน
ฟังก์ชัน LAST-EXPN	คำอธิบายการทำงานขั้นสุดท้าย
ฟังก์ชัน PROMPT-WHY	ถามผู้ใช้ว่าต้องการคำอธิบายอีกหรือไม่
ฟังก์ชัน PRT-CNCLD-CLAUSE	พิมพ์ประโยคข้อสรุปของปัญหา
ฟังก์ชัน WHY-DISP	แปลงข้อความในกฎเป็นภาษาธรรมชาติและแสดงเพื่ออธิบายขั้นตอนการทำงานของระบบ

ตารางที่ ข.5 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย EXPLANATION

ชื่อฟังก์ชัน	การทำงาน
โปรแกรมย่อย PRT-EXPN-RULE	แปลงกฎเป็นภาษาธรรมชาติและแสดงทางจอภาพ
ฟังก์ชัน PRT-CNTXT	พิมพ์ชื่อเนื้อความที่แทนเป็นภาษาธรรมชาติแล้ว
ฟังก์ชัน PRT-EXPN-CLAUSE	พิมพ์คำอธิบายที่แปลงเป็นภาษาธรรมชาติแล้ว
ฟังก์ชัน PRT-PMT	พิมพ์ชื่อพารามิเตอร์ที่แทนเป็นภาษาธรรมชาติแล้ว
ฟังก์ชัน PRT-RULE-ACTION	พิมพ์ส่วนกระทำของกฎ
ฟังก์ชัน PRT-RULE-PREMISE	พิมพ์ส่วนหลักฐานของกฎ

ตารางที่ ข.6 ดัชนีรายชื่อฟังก์ชันในโปรแกรมย่อย PRT-EXPN-RULE

ชื่อฟังก์ชัน	ฟังก์ชันที่เรียกใช้	การทำงาน
ADD-HYPOTHESIS	FINDOUT	ใส่ประโยคสมมติฐานลงในหน่วยความจำที่เก็บเหตุการณ์
ADD-TREATMENT	MONITOR	ใส่ประโยควิธีการแก้ไขปัญหาในรายการคำแนะนำ
ANSWER-CHECK	FINDOUT	ตรวจสอบความถูกต้องของคำตอบ
ASK	FINDOUT	ถามคำถามเกี่ยวกับพารามิเตอร์
CHECK-HAVE-TREE	FINDOUT	ตรวจสอบว่ามีจุดเนื้อความในต้นไม้เนื้อความหรือไม่
CHECK-REQ-RULE	MONITOR	ตรวจสอบชุดพารามิเตอร์ที่ต้องหาค่า
CHECK-TREE-PROPERTY	FINDOUT	ตรวจสอบว่าชื่อเนื้อความนั้นเคยถูกสร้างหรือไม่
CNCLD-EXPN	EXPLANATION	อธิบายสาเหตุของปัญหาที่สรุปได้
CNTXT-BINDING	INFER	กำหนดชื่อเนื้อความใหม่ให้เนื้อความที่เกี่ยวข้อง
CNTXT-REPLACE	MONITOR	แทนค่าชนิดเนื้อความด้วยชื่อเนื้อความที่กำหนด
CONCLUDE-WORK	MONITOR	สรุปค่าความจริงที่เกี่ยวข้องสาเหตุของปัญหาใส่ในรายการสรุปสาเหตุของปัญหา
CREATE-CURR-CNTXT	MONITOR	กำหนดเนื้อความที่เกี่ยวข้องกับการทำงาน
DEL-REQ-RULE	MONITOR	ลบหมายเลขกฎที่ตรวจสอบแล้วออกจากรายการกฎที่สนใจ
DISPLAY-WHY	FINDOUT	อธิบายขั้นตอนการทำงานของระบบ
EVALUATE	MONITOR	ประเมินผลค่าคำตอบที่ได้จากผู้ใช้ระบบ
EVAL-NO	MONITOR	กำหนดค่าตัวแปรต่างๆ ที่เกี่ยวข้องเมื่อประโยคเงื่อนไขในส่วนหลักฐานเป็นเท็จ
EVAL-YES	MONITOR	กำหนดค่าตัวแปรต่างๆ ที่เกี่ยวข้องเมื่อประโยคเงื่อนไขในส่วนหลักฐานเป็นจริง

ตารางที่ ข.7 ดัชนีรายชื่อฟังก์ชันทั้งหมดในระบบ SEFS

ชื่อฟังก์ชัน	ฟังก์ชันที่เรียกใช้	การทำงาน
EXEC-GOAL-RULE	INFER	เรียกประเมินผลกฎเป้าหมายของระบบ
EXPLANATION	INFER	อธิบายการทำงานของระบบ
EXPN-DISP	EXPLANATION	แสดงข้อความอธิบายการทำงาน
GET-HYPOTHESIS	MONITOR	เรียกใช้ประโยคสมมติฐานที่เกี่ยวข้องกับประโยคเงื่อนไขมาทำการตรวจสอบ
INSTRUCTION-N	START	เว้นการแนะนำไปเลือกการทำงานต่อไป
INSTRUCTION-Y	START	แนะนำ keywords ต่างๆ และความหมาย
LAST-EXPN	EXPLANATION	คำอธิบายการทำงานขั้นสุดท้าย
MAKE-CNTXT-NODE	FINDOUT	สร้างชื่อจุดเนื้อความใหม่
MAKE-CNTXT-TOP	INFER	สร้างจุดเนื้อความจุดแรกในต้นไม้เนื้อความ
MAKE-CNTXT-TREE	FINDOUT	ใส่จุดเนื้อความที่สร้างใหม่ลงในตำแหน่งที่เหมาะสมในต้นไม้เนื้อความ
MONITOR-ACTION	MONITOR	ตรวจสอบและใช้งานส่วนกระทำของกฎ
MONITOR-PREMISE	MONITOR	ประเมินผลส่วนหลักฐานของกฎ
PRINT-TEXT	FINDOUT	พิมพ์ข้อความต่างๆ โดยตรวจสอบความยาวไม่เกิน 70
PROBLEM	START	ถามรายละเอียดเบื้องต้นและระยะของปัญหาที่เกิดขึ้น
PROMPT-DIAG	MONITOR	เตรียมแสดงข้อความสาเหตุของปัญหา
PROMPT-END-MESSAGE	INFER	แสดงสถานะการทำงานหลังจากทำงานเสร็จ
PROMPT-WHY	EXPLANATION	ถามผู้ใช้ว่าต้องการคำอธิบายอีกหรือไม่
PRT-CNCLD-CLAUSE	EXPLANATION	พิมพ์ประโยคข้อสรุปของปัญหา
PRT-CNTXT	PRT-EXPN-RULE	พิมพ์ชื่อเนื้อความที่แทนเป็นภาษาธรรมชาติแล้ว
PRT-EXPN-CLAUSE	PRT-EXPN-RULE	พิมพ์คำอธิบายที่แปลงเป็นภาษาธรรมชาติแล้ว

ตารางที่ ข.7 ดัชนีรายชื่อฟังก์ชันทั้งหมดในระบบ SEFS (ต่อ)

ชื่อฟังก์ชัน	ฟังก์ชันที่เรียกใช้	การทำงาน
PRT-PMT	PRT-EXPN-RULE	พิมพ์ชื่อพารามิเตอร์ที่แทนเป็นภาษาธรรมชาติแล้ว
PRT-RULE-ACTION	PRT-EXPN-RULE	พิมพ์ส่วนกระทำของกฎ
PRT-RULE-PREMISE	PRT-EXPN-RULE	พิมพ์ส่วนหลักฐานของกฎ
RECOMMEND-WORK	MONITOR	แสดงสาเหตุของปัญหาที่สรุปได้
REM-SYM	INFER	ลบชื่อเนื้อความที่เคยสร้างขึ้นในตารางสัญญลักษณ์
RULE-CHECK	INFER	ตรวจสอบกฎความรู้อย่างละเอียด
RULE-INTERPRETER	INFER	เรียกประเมินผลกฎความรู้ต่างๆ ที่เกี่ยวข้องกับปัญหา
TREATMENT-WORK	MONITOR	ใส่ประโยควิธีการแก้ไขปัญหาลงในรายการคำแนะนำ
WHY-DISP	EXPLANATION	แปลงข้อความในกฎเป็นภาษาธรรมชาติและแสดงเพื่ออธิบายขั้นตอนการทำงานของระบบ
WRONG-ANSWER	FINDOUT	แสดงคำตอบของผู้ใช้ไม่ถูกต้อง

ตารางที่ ข.7 ดัชนีรายชื่อฟังก์ชันทั้งหมดในระบบ SEFS (ต่อ)

ชื่อตัวแปร	ความหมาย
action_clause	ประโยคกระทำที่จะเรียกทำงาน
action_cntxt	เนื้อความในประโยคกระทำที่เรียกใช้งาน
answer	คำตอบที่ผู้ใช้ตอบเป็นข้อความ
ans_flag	ตัวบอกสถานะคำตอบของผู้ใช้ระบบ
ans_type	ชนิดของคำตอบที่เป็นไปได้
ans_val	ค่าคำตอบที่ผู้ใช้กำหนดเป็น y หรือ n
children	รายชื่อเนื้อความที่เป็น descendent ของเนื้อความนั้น
clause	ประโยคค่าความจริงที่แทนชื่อเนื้อความลงไปแล้ว
cncld_list	รายการค่าความจริงที่สรุปเป็นสาเหตุของปัญหา
cntxt	ชื่อเนื้อความ
cntxt_base	กลุ่มของเนื้อความชนิดต่างๆ ในระบบ
cntxt_bind_list	รายการชนิดเนื้อความพร้อมด้วยชื่อที่กำหนดแล้ว
cntxt_new	ตัวบอกสถานะว่าเนื้อความนั้นสร้างครั้งแรกหรือไม่
cntxt_stat	ตัวบอกสถานะว่าสรุปสาเหตุของเนื้อความนั้นหรือยัง
cntxt_tree	รายการชื่อเนื้อความที่สร้างขึ้นระหว่างการทำงาน
cntxt_type	ชนิดเนื้อความ
cntxt_used	ตัวบอกสถานะว่าเนื้อความนั้นเกี่ยวของกับปัญหาหรือไม่
condition_stat	ตัวบอกสถานะว่าประโยคเงื่อนไขเป็นจริงหรือเท็จ
condition_val	ค่าของพารามิเตอร์ในประโยคเงื่อนไข
continue	ตัวบอกสถานะการทำงานประโยคเงื่อนไขต่อไปในส่วนหลักฐาน
ctx	ชื่อเนื้อความชั่วคราวที่ใช้ใน cntxt_bind_list
curr_action	ประโยคกระทำที่กำลังทำการประเมินผล
curr_clause	ประโยคค่าความจริงที่แทนค่าด้วยชื่อเนื้อความแล้ว

ตารางที่ ข.8 ดัชนีรายชื่อตัวแปรในระบบ SFES

ชื่อตัวแปร	ความหมาย
curr_cntxt	ชื่อเนื้อความที่กำลังทำการหาผลสรุป
curr_condition	ประโยคเงื่อนไขที่กำลังทำการประเมินผล
curr_parameter	พารามิเตอร์ที่กำลังทำการประเมินผล
curr_predicate	predicate ที่กำลังทำการประเมินผล
curr_rule_num	หมายเลขของกฎที่กำลังทำการประเมินผล
curr_value	ค่าของพารามิเตอร์ที่กำลังทำการประเมินผล
diag	ข้อความแสดงข้อผิดพลาดของคำตอบ
diag_clause	ประโยคค่าความจริงที่ใช้แสดงสาเหตุของปัญหา
end_fg	ตัวบอกสถานะว่าสิ้นสุดการทำงานหรือยัง
eval_value	ตัวบอกสถานะผลการประเมินผลประโยคเงื่อนไข
explant_need	ตัวบอกสถานะว่าต้องการคำอธิบายการทำงานหรือไม่
expn_action	ประโยคกระทำที่ใช้ขณะทำการอธิบายการทำงานของระบบ
expn_clause	ประโยคค่าความจริงใดๆ ที่จะใช้พิมพ์เพื่ออธิบายการทำงาน
expn_cntxt	ชื่อเนื้อความที่ใช้อธิบายการทำงาน
expn_message	ประโยคอธิบายการทำงานที่แปลงเป็นภาษาธรรมชาติแล้ว
expn_num	หมายเลขกฎที่กำลังใช้อธิบายการทำงาน
expn_pmt	พารามิเตอร์ที่อยู่ในประโยคซึ่งใช้อธิบายการทำงาน
expn_premise	ส่วนหลักฐานของกฎที่ใช้อธิบายการทำงาน
father	คุณสมบัติของเนื้อความที่เก็บ ancestor ของเนื้อความนั้น
have	ตัวบอกสถานะว่ามีชื่อเนื้อความนั้นในต้นไม้เนื้อความหรือยัง
hist_list	รายการที่เก็บหมายเลขกฎความรู้ต่างๆ ที่เป็นเส้นทางวินิจฉัย
hypothesis_clause	ประโยคสมมติฐานที่ได้จากผู้ใช้และสรุปจากกฎความรู้อื่น
hypothesis_list	รายการประโยคสมมติฐานทั้งหมด
hypothesis_val	ค่าของพารามิเตอร์ที่อยู่ในประโยคสมมติฐาน

ตารางที่ ข.8 ดัชนีรายชื่อตัวแปรในระบบ SFES (ต่อ)

ชื่อตัวแปร	ความหมาย
insert_clause	ประโยคค่าความจริงที่จะใส่ลงในหน่วยความจำที่เก็บเหตุการ์และรายการสาเหตุของปัญหา
knowledge_base	ฐานความรู้ของระบบที่บรรจุเข้าใช้งานในหน่วยความจำ
known	ตัวบอกสถานะว่าเนื้อความนั้นรู้ค่าหรือยัง
last	ตำแหน่งสุดท้ายของประโยคใดๆ ที่จะพิมพ์
message	ประโยคที่จะพิมพ์แสดงทางจอภาพ
new_node	ชื่อเนื้อความที่กำหนดขึ้นใหม่เมื่อจะสร้างในต้นไม้เนื้อความ
node	ชื่อเนื้อความที่เลือกจากรายการชนิดเนื้อความ
notek_fg	ตัวบอกสถานะสภาพการสิ้นสุดการทำงานของระบบ
nxt_con	ตัวบอกสถานะว่าต้องการคำปรึกษาอีกหรือไม่
nxt_d	ตัวบอกสถานะว่าให้รอเพื่อที่จะแสดงจอภาพต่อไป
nxt_expn	ตัวบอกสถานะว่าผู้ใช้ต้องการคำอธิบายหรือไม่
object_status	ตัวบอกสถานะของตัวแปรที่หาค่าว่าเป็นพารามิเตอร์หรือเนื้อความ
old_nm	คุณสมบัติของชื่อเนื้อความแสดงชนิดเนื้อความของชื่อนั้นๆ
pmt	พารามิเตอร์ที่ระบบกำลังหาค่า
pmt_list	รายการของพารามิเตอร์ที่เกี่ยวข้องกับการทำงาน
prop_list	รายการคุณสมบัติทั้งหมดของเนื้อความ
prt_cntxt	ความหมายของเนื้อความที่จะแสดงทางจอภาพ
prt_extent	ส่วนขยายของ predicate ที่ใช้แสดงตอนอธิบาย และสรุป
prt_parm	ความหมายของพารามิเตอร์ที่จะแสดงทางจอภาพ
prt_predicate	ความหมายของ predicate ที่จะแสดงทางจอภาพ
prt_unit	ความหมายของหน่วยของพารามิเตอร์
prt_val	ค่าของพารามิเตอร์ที่จะพิมพ์
require_parm	กลุ่มของพารามิเตอร์ที่ต้องรู้ค่า

ตารางที่ ข.8 ดัชนีรายชื่อตัวแปรในระบบ SFES (ต่อ)

ชื่อตัวแปร	ความหมาย
req_rule	รายการกฎที่ระบบต้องทำการตรวจสอบ
rpmt_list	รายการพารามิเตอร์ที่ระบบยังไม่ได้ทำการตรวจสอบ
rule_action	ส่วนกระทำของกฎที่ทำการตรวจสอบ
rule_cntnt	ส่วนหลักฐานและส่วนกระทำของกฎความรู้
rule_cont	ตัวบอกสถานะว่าจะทำการตรวจสอบประโยคเงื่อนไขต่อไปในส่วนหลักฐานหรือไม่
rule_list	รายการของกฎที่เกี่ยวข้องกับปัญหานั้นๆ
rule_premise	ส่วนหลักฐานของกฎที่กำลังทำการตรวจสอบ
scr_len	จำนวนบรรทัดที่แสดงบนจอภาพ
text_length	ความยาวของข้อความที่แสดงบนจอภาพ
top	ตัวบอกสถานะว่าเป็นเนื้อความจุดแรก
treat_clause	ประโยควิธีแก้ไขที่จะใช้เป็นข้อแนะนำ
treat_cntxt	เนื้อความที่อยู่ในประโยควิธีแก้ไข
treat_line	ประโยควิธีแก้ไขที่จะใส่ลงในรายการข้อแนะนำวิธีแก้ปัญหา
treat_list	รายการข้อแนะนำวิธีการแก้ปัญหา
treat_object	ชื่อเนื้อความที่อยู่ในประโยควิธีการแก้ไข
treat_pmt	พารามิเตอร์ที่อยู่ในประโยควิธีการแก้ไข
treat_val	ค่าของพารามิเตอร์ในประโยควิธีการแก้ไข
val	ค่าของพารามิเตอร์ในประโยคเงื่อนไขที่ระบบต้องหาค่า
work_cntxt	ชนิดเนื้อความที่ระบบต้องทำการสร้างชื่อเฉพาะมาแทน
work_memory_list	รายการค่าความจริงที่ระบบสรุปได้

ตารางที่ ข.8 ดัชนีรายชื่อตัวแปรในระบบ SFES (ต่อ)

ภาคผนวก ค

คำศัพท์ที่สำคัญ

ในการทำวิทยานิพนธ์ฉบับนี้ส่วนหนึ่งที่สำคัญคือ การค้นคว้าเอกสาร และรวบรวมทฤษฎีต่างๆ ที่เกี่ยวข้องทั้งทางปัญญาประดิษฐ์โดยเฉพาะระบบเชี่ยวชาญ และการเพาะเลี้ยงกุ้ง จึงมีคำศัพท์ทางวิชาการ หรือคำที่ใช้เฉพาะภายในวงการบางคำที่ไม่เป็นที่แพร่หลายโดยทั่วไป ดังนั้นผู้วิจัยได้ทำการรวบรวมคำเหล่านี้ที่มีการอ้างถึงในวิทยานิพนธ์ เพื่อช่วยให้เกิดความเข้าใจ และสามารถใช้ประโยชน์จากวิทยานิพนธ์ฉบับนี้ได้เต็มที่ คำศัพท์ทั้งหมดแบ่งออกเป็น 2 ส่วนคือ

คำศัพท์ที่เกี่ยวข้องกับระบบเชี่ยวชาญ

คำศัพท์ที่เกี่ยวข้องกับการเพาะเลี้ยงกุ้ง

คำศัพท์ที่เกี่ยวข้องกับระบบเชี่ยวชาญ

action หรือ ส่วนกระทำของกฎ เป็นส่วนที่มีฟังก์ชันเกี่ยวกับการทำงานของกฎจะถูกใช้งานก็ต่อเมื่อส่วนวินิจฉัยตรวจสอบแล้วส่วนหลักฐานของกฎถูกต้อง ฟังก์ชันจะมีหน้าที่ต่างๆ เช่น ทำการเพิ่มค่าความจริงเข้าสู่หน่วยความจำที่เก็บเหตุการณ์ หรือลบค่าความจริงออกจากหน่วยความจำที่เก็บเหตุการณ์ หรือ แสดงผลการทำงานของระบบ เป็นต้น

action clause หรือ ประโยคกระทำ คือประโยคแต่ละประโยคที่อยู่ในส่วนกระทำประกอบด้วย ฟังก์ชัน ชนิดเนื้อความ ชื่อพารามิเตอร์ และค่าของพารามิเตอร์

AI หรือ Artificial Intelligence หรือ ปัญญาประดิษฐ์ เป็นศาสตร์สาขาหนึ่งทางด้านคอมพิวเตอร์ โดยพยายามศึกษาและจำลองการทำงานของมนุษย์ลงบนเครื่องคอมพิวเตอร์เพื่อให้สามารถคิด และตัดสินใจบางอย่างได้

backtracking หรือ การย้อนหาเหตุผล เป็นวิธีการย้อนเส้นทางการหาเหตุผลที่พบว่าไม่ถูกต้อง โดยมีการเก็บสถานะต่างๆ ของการหาเหตุผลไว้เพื่อตรวจสอบและย้อนเส้นทางพร้อมทั้งเลือกเส้นทางใหม่ได้อย่างถูกต้อง

backward chaining หรือ การหาเหตุผลแบบย้อนหลัง เป็นกลไกการวินิจฉัยวิธีหนึ่งใช้กับระบบที่มีการแทนค่าความรู้แบบกฎความรู้ การทำงานจะเริ่มจากกฎเป้าหมาย และย้อนหาเหตุผลจากส่วนกระทำของกฎความรู้อื่นที่เกี่ยวข้องกับกฎเป้าหมายของระบบ จนกว่าจะได้ข้อสรุปผลการ

ทำงานออกมา

certainty factor หรือ ปัจจัยความเชื่อมั่น เป็นค่าตัวเลขที่ใช้กำหนดน้ำหนักของค่าความจริง เพื่อแสดงถึงความสัมพันธ์ของค่าความจริงนั้นๆ ว่ามีความน่าเชื่อถือมากน้อยแค่ไหน ใช้กับการแทนค่าความรู้แบบ OBJECT-ATTRIBUTE-VALUE TRIPLETS ซึ่งรวมกันอยู่ในกฎความรู้อีกชั้นหนึ่ง

compiled knowledge หรือ ความรู้ที่ถูกจัดรูปแบบแล้ว คือความรู้ที่ถูกจัดให้อยู่ในรูปแบบที่สะดวกในการประมวลผลอาจอยู่ในรูปของย่อยงาน หรือ แยกเป็นกลุ่มๆ ก็ได้ แบ่งออกเป็น deep knowledge และ surface knowledge พบว่าความเชี่ยวชาญเรื่องใดเรื่องหนึ่งจะประกอบด้วยความรู้ที่จัดรูปแบบนี้เป็นจำนวนมาก

condition clause หรือ ประโยคเงื่อนไข ประกอบด้วย พารามิเตอร์ ชนิดเนื้อความ ค่าของพารามิเตอร์ และค่าปัจจัยความเชื่อมั่น ส่วนวินิจฉัยจะทำการตรวจสอบรายการที่มีค่าเหล่านี้ประกอบอยู่ กับค่าความจริงที่ผู้ใช้ตอบเข้าสู่ระบบโดยใช้ฟังก์ชันที่กำหนดเพื่อดูว่าจะเรียกใช้งานกฎความรู้นั้นๆ หรือไม่

conflict resolution หรือ กลวิธีไกล่เกลี่ยความขัดแย้ง เป็นวิธีการเลือกกฎที่เหมาะสมที่สุดจากกลุ่มของกฎที่เกี่ยวข้องกับสภาพปัญหาขณะนั้น โดยมีวิธีการต่างๆ กันเช่น เลือกตามลำดับของกฎ หรือเลือกตามความสำคัญของกฎ เป็นต้น

context tree หรือ ต้นไม้เนื้อความ เป็นโครงสร้างข้อมูลแบบต้นไม้ที่แทนค่าจุดแต่ละจุดในโครงสร้างด้วยชื่อเนื้อความที่เกี่ยวข้องในเหตุการณ์จริงโดยจุดต่างๆ จะมีความสัมพันธ์แบบตามลำดับชั้น

deep knowledge หรือ ความรู้ในแนวลึก หมายถึงความรู้ที่มนุษย์เข้าใจแบบลึกซึ้ง เช่น ทฤษฎีพื้นฐานต่างๆ กฎความจริงต่างๆ ที่ได้จากการเรียน เป็นต้น

domain หรือ ขอบเขตของปัญหา หมายถึงขอบเขตความสนใจในสาขาวิชาใดวิชาหนึ่งโดยเฉพาะ เช่น การเพาะเลี้ยง การแพทย์ การลงทุน หรือ ตลาดหุ้น เป็นต้น ระบบเชี่ยวชาญที่พัฒนาขึ้นนั้นจะเน้นการแก้ปัญหาในขอบเขตใดขอบเขตหนึ่งที่กำหนดเท่านั้น

dynamic database หรือ working memory หรือ ฐานข้อมูลพลวัต เป็นโครงสร้างข้อมูลที่เก็บค่าความจริงต่างๆ ที่ได้ระหว่างการทำงานของระบบ อีกคำที่ใช้คือ หน่วยความจำที่เก็บเหตุการณ์

Expert system หรือ Knowledge based system หรือ ระบบเชี่ยวชาญ คือ โปรแกรมคอมพิวเตอร์ที่ประกอบด้วยส่วนสำคัญ 2 ส่วนคือ ส่วนฐานความรู้ และส่วนวินิจฉัย ทำหน้าที่ในลักษณะการแก้ปัญหาระดับที่ต้องการความรู้จากผู้เชี่ยวชาญโดยใช้เทคนิคทางด้านปัญญาประดิษฐ์

explanation subsystem หรือ ระบบช่วยอธิบายการทำงาน เป็นระบบย่อยที่ทำหน้าที่ช่วย

อธิบายการทำงานของระบบถึงสาเหตุที่ต้องถามคำถามนั้นๆ เมื่อผู้ใช้ระบบถามด้วยคำถาม WHY และอธิบายสาเหตุที่ระบบสรุปค่าความจริงบางอย่างระหว่างการทำงานของระบบ เมื่อผู้ใช้ระบบถามด้วย HOW การทำงานจะแสดงเส้นทางการวินิจฉัยให้ผู้ใช้เข้าใจถึงขั้นตอนการทำงานของระบบ

fact หรือ ค่าความจริง เป็นประโยคค่าความจริงที่เกิดขึ้นได้จากผู้ใช้ระบบตอบคำถาม หรือระบบสรุปได้เองจากกฎความรู้อื่นๆ ในฐานความรู้ ประกอบด้วย พารามิเตอร์ ชื่อเนื้อความ ค่าของพารามิเตอร์ และปัจจัยความเชื่อมั่น

forward chaining หรือ การหาเหตุผลแบบไปข้างหน้า เป็นกลไกการวินิจฉัยอีกวิธีหนึ่งใช้กับระบบเชี่ยวชาญที่มีการแทนค่าความรู้แบบกฎความรู้ การทำงานจะเริ่มจากสภาวะเริ่มต้นที่กำหนด แล้วพยายามหากฎความรู้ที่มีส่วนหลักฐานเป็นจริงสอดคล้องกับสภาวะเริ่มต้นนั้น แล้วทำการคัดเลือกกฎที่เหมาะสมที่สุดเพื่อใช้งานส่วนการกระทำ จากนั้นจะเกิดการทำงานต่อเนื่องกันต่อไปจนกระทั่งได้สภาวะที่เป็นสภาวะเป้าหมาย

frame หรือ กรอบความรู้ เป็นการแทนค่าความรู้ประเภทหนึ่งประกอบด้วยโครงสร้างข้อมูลพื้นฐาน คือ {กรอบความรู้} ::= {ชื่อกรอบความรู้}
{ช่องความรู้}
โดยช่องความรู้ (slot) จะมีมากกว่า 1 ช่อง หลักการทำงานจะเป็นการเติมค่าต่างๆ ของสภาพจริงลงในช่องความรู้ต่างๆ ของกรอบความรู้ที่กำลังทำการพิจารณา

goal rule หรือ กฎเป้าหมาย คือกฎความรู้แรกสุดที่ระบบเชี่ยวชาญเรียกใช้งาน โดยส่วนหลักฐานจะประกอบด้วยประโยคเงื่อนไขต่างๆ เกี่ยวกับปัญหาย่อยที่สำคัญ และส่วนการกระทำจะเป็นการสรุปการทำงานทั้งหมดของระบบ ถือว่าเป็นจุดเริ่มต้นการทำงานของระบบ

heuristic หรือ rule of thumb หรือ กฎตายตัว เป็นเงื่อนไขตายตัวบางอย่างที่สรุปได้จากการสอบถามผู้เชี่ยวชาญที่เคยใช้วิธีการนั้นๆ แก้ไขปัญหาโดยไม่จำเป็นต้องมีเหตุผล หรือทฤษฎีสนับสนุนมีประโยชน์ช่วยลดปริภูมิการค้นหาลงระหว่างการทำงานของระบบ

inference mechanism หรือ กลไกการวินิจฉัย คือวิธีการสรุปหาเหตุผลหรือตัดสินใจของระบบเชี่ยวชาญจากข้อมูลความรู้ที่มีอยู่ในฐานความรู้ มีกลไกการวินิจฉัยแบบต่างๆ เช่น การหาเหตุผลแบบไปข้างหน้า การหาเหตุผลแบบย้อนหลัง หรือ การค้นหาแบบฮิวรีติก เป็นต้น

inference path หรือ วิถีทางการวินิจฉัย เป็นหมายเลขกฎต่างๆ ที่ส่วนวินิจฉัยเลือกใช้งานระหว่างการทำงานตั้งแต่ต้นจนถึงสรุปสาเหตุของปัญหาออกมาจะใช้ช่วยในการอธิบายการทำงานของระบบ

instantiation หรือ การกำหนดค่าเฉพาะให้แก่เนื้อความชนิดใดๆ ในระหว่างการทำงานของระบบซึ่งจะไม่ซ้ำกับชื่อที่มีอยู่แล้ว

knowledge acquisition subsystem หรือ ระบบรับเพิ่มเติมความรู้ เป็นระบบย่อยของระบบเชี่ยวชาญที่เพิ่มเติมขึ้นมาเพื่อช่วยเพิ่มความสามารถ และขอบเขตของความรู้ให้มากขึ้น โดยมีความสัมพันธ์กับส่วนติดต่อกับผู้ใช้ การรับเพิ่มเติมความรู้ต้องมีการตรวจสอบและเปลี่ยนรูปแบบความรู้ที่ได้จากผู้ใช้ใหม่เพื่อเหมาะสมกับกลไกการวินิจฉัย

knowledge base หรือ ฐานความรู้ เป็นส่วนประกอบที่สำคัญของระบบเชี่ยวชาญ ประกอบด้วยกฎความรู้ที่เกี่ยวข้องกับขอบเขตปัญหานั้นๆ กฎความรู้ที่กำหนดวิธีการแก้ปัญหา หรือกฎฮิวริสติกต่างๆ เป็นต้น

knowledge engineer หรือ วิศวกรความรู้ เป็นผู้ที่ทำการพัฒนาระบบเชี่ยวชาญขึ้น โดยเป็นผู้ทำหน้าที่ถ่ายทอดความรู้ และประสบการณ์จากผู้เชี่ยวชาญในสาขาวิชานั้นมาเป็นโปรแกรมคอมพิวเตอร์ที่ช่วยทำการแก้ปัญหา

knowledge representation หรือ การแทนค่าความรู้ คือการแทนค่าความจริงต่างๆ ให้อยู่ในรูปที่เหมาะสมกับการประมวลผลด้วยวิธีต่างๆ การแทนค่าความรู้มีวิธีต่างๆ เช่น วิธีทางตรรกศาสตร์ แบบข่ายซีแมนติก หรือ กฎความรู้เป็นต้น

lisp หรือ List Processing เป็นภาษาคอมพิวเตอร์ที่มีพื้นฐานการทำงานเป็นการประมวลผลโครงสร้างข้อมูลที่เป็นรายการ (list) เป็นภาษาคอมพิวเตอร์ที่นิยมใช้ในการพัฒนาโปรแกรมด้านปัญญาประดิษฐ์

MYCIN หรือ ระบบเชี่ยวชาญ MYCIN เป็นระบบเชี่ยวชาญด้านการแพทย์ช่วยในการวิเคราะห์สาเหตุของการติดเชื้อแบคทีเรียในระบบทางเดินหายใจ โดยระบุชนิดเชื้อแบคทีเรียที่เกิดกับคนไข้ และแนะนำวิธีการรักษาจากค่าความจริงที่ผู้ใช้ระบบหรือแพทย์กำหนดให้ กลไกการวินิจฉัยเป็นการหาเหตุผลแบบย้อนหลัง และมีการแทนค่าความรู้อยู่ในรูปของกฎความรู้ สามารถให้เหตุผลได้ในสภาวะที่มีความไม่แน่นอนของเหตุการณ์เกิดขึ้น

natural language processing หรือ การประมวลผลภาษาธรรมชาติ เป็นสาขาความรู้หนึ่งของปัญญาประดิษฐ์โดยพยายามจำลองให้คอมพิวเตอร์สามารถเข้าใจ และตีความหมายภาษาพูดหรือภาษาเขียนตามธรรมชาติทั่วๆ ไป เมื่อนำมาประกอบกับส่วนติดต่อกับผู้ใช้ของระบบเชี่ยวชาญจะช่วยให้ผู้ใช้สามารถใช้งานระบบสะดวกขึ้น

object-attribute-value triplets เป็นการแทนค่าความรู้ของแต่ละประโยคค่าความจริงโดยแยกส่วนวัตถุ (object) ส่วนบอกลักษณะ (attribute) และค่าของส่วนบอกลักษณะนั้น (value) ใช้เป็นองค์ประกอบพื้นฐานของการแทนค่าในรูปของกฎความรู้ทั้งประโยคเงื่อนไขและประโยคกระทำ

premise หรือ ส่วนหลักฐานของกฎ เป็นที่รวบรวมประโยคเงื่อนไขต่างๆ ในกฎความรู้แต่ละกฎ

ซึ่งส่วนวินิจฉัยจะทำการตรวจสอบความถูกต้องของส่วนหลักฐานสำหรับกฎที่เกี่ยวข้องว่าเป็นจริง หรือ เป็นเท็จเพื่อดูว่าจะเรียกใช้กฎนั้นๆ หรือไม่

Prolog หรือ Programming in Logic เป็นภาษาคอมพิวเตอร์ที่มีพื้นฐานการทำงานของ predicate calculus มีการประมวลผลโดยใช้หลักการตรรกศาสตร์ และโครงสร้างข้อมูลจะเป็นในลักษณะสัญญลักษณ์ (symbol)

prototype system หรือ ระบบต้นแบบ เป็นระบบแรกที่วิศวกรความรู้ทำการพัฒนาขึ้นมาเพื่อให้ผู้เชี่ยวชาญที่ร่วมทำการพัฒนาระบบตรวจสอบผลการทำงาน และประเมินผลฐานความรู้ วิธีการวินิจฉัยช่วยให้สามารถพัฒนาระบบได้ถูกต้อง และเป็นไปตามสภาพการทำงานจริงมากที่สุด

reasoning network หรือ ข่ายงานการหาเหตุผล เป็นโครงสร้างข้อมูลที่ใช้แสดงความสัมพันธ์ระหว่างกฎต่างๆ ในระหว่างการทำงานโดยแสดงพารามิเตอร์แต่ละตัวในส่วนหลักฐานของกฎความรู้ และความสัมพันธ์ระหว่างพารามิเตอร์นั้นกับกฎความรู้อื่น

robotic หรือ หุ่นยนต์ เป็นการประยุกต์ใช้งานความรู้ด้านปัญญาประดิษฐ์ในลักษณะของกลไกทางพลศาสตร์โดยพยายามจำลองการทำงานของเครื่องจักรกลที่สามารถรับรู้ และทำงานบางอย่างได้ เช่น เชื่อมโลหะ ประกอบชิ้นส่วนอุตสาหกรรม หรือ การขนส่ง เป็นต้น ใช้งานมากในด้านอุตสาหกรรมที่เป็นงานค่อนข้างอันตราย และต้องการความแม่นยำสูง

select-execute cycle หรือ recognize-act cycle เป็นการวนรอบการทำงานของระบบเชี่ยวชาญ โดยตรวจสอบกฎความรู้ในฐานความรู้ที่เกี่ยวกับเหตุการณ์จริง และทำการเลือกกฎที่มีส่วนหลักฐานถูกต้องมาใช้งานส่วนภาระทำ เมื่อครบรอบการทำงานก็จะวนรอบอีกครั้งเพื่อเลือกกฎจนสิ้นสุดการทำงานหรือสรุปสาเหตุของปัญหาได้

slot หรือ ช่องความรู้ เป็นส่วนประกอบของวัตถุในระบบกรอบความรู้ ใช้เป็นที่บรรจุคุณสมบัติแท้จริงของความรู้ เช่นชื่อของวัตถุ ชื่อส่วนบอกลักษณะ ค่าของส่วนบอกลักษณะ หรือ ตัวชี้ (pointer) ที่แสดงความสัมพันธ์ระหว่างกฎความรู้ต่างๆ

surface knowledge หรือ ความรู้ในแนวกว้าง เป็นความรู้ที่มนุษย์สะสมมาจากประสบการณ์และนำมาใช้ในการแก้ปัญหาทางปฏิบัติ ประกอบด้วยทฤษฎีเฉพาะสาขา กฎตายตัวจำนวนมาก

value หรือ ค่าของส่วนบอกลักษณะ เป็นจำนวนหรือคุณภาพที่ใช้อธิบายส่วนบอกลักษณะ (attribute) เช่น ถ้าส่วนบอกลักษณะเป็นสี ค่าของส่วนบอกลักษณะที่เป็นไปได้คือ สีแดง สีฟ้า เป็นต้น

voice and pattern recognition หรือ การรับรู้รูปแบบและเสียง เป็นการประยุกต์ใช้งานในด้านรับคลื่นเสียง หรือรูปแบบบางอย่าง เช่น เสียงพูด รูปภาพ หรือ รูปตัวอักษร เป็นต้น มาทำการเปลี่ยนเป็นสัญญลักษณ์ทางคณิตศาสตร์ เมื่อผ่านการประมวลผลตามขั้นตอนบางอย่างจะทำให้คอมพิวเตอร์สามารถรู้ความหมายของรูปแบบนั้นๆ และตอบสนองได้โดยไม่ผิดพลาด

working memory หรือ หน่วยความจำที่เก็บเหตุการณ์ ประกอบด้วยค่าความจริงที่เกิดขึ้นในระหว่างการทำงานของระบบเชี่ยวชาญ ซึ่งอาจได้มาจากผู้ใช้ระบบ หรือระบบสรุปเองจากส่วนกระทำของกฎความรู้อื่น

<u>คำศัพท์ที่เกี่ยวข้องกับการเพาะเลี้ยงกุ้ง</u>

acclimation หรือ การปรับสภาพ เป็นขั้นตอนที่ทำการลดความเค้นของกุ้งที่อาจจะเกิดจากการขนส่งโดยใช้บ่อปรับสภาพที่มีปัจจัยทางกายภาพ และทางเคมีใกล้เคียงกับสภาพเดิมมากที่สุดในช่วงเวลาที่เหมาะสมเพียงพอที่กุ้งจะปรับตัวได้ มีความจำเป็นทั้งสำหรับลูกกุ้ง และพ่อพันธุ์ แม่พันธุ์กุ้ง

biological factor หรือ ปัจจัยทางชีวภาพ เป็นผลกระทบทางนิเวศน์วิทยาที่เกิดจากสิ่งมีชีวิตประเภทอื่นๆ ในระบบนิเวศน์เดียวกัน เช่น การแย่งอาหาร หรือการแย่งที่อยู่อาศัย เป็นต้น

carapace length หรือ ความยาวเปลือกหัว เป็นการวัดขนาดกุ้งแบบหนึ่งโดยวัดความยาวตั้งแต่ปลายกรี (rostrum) จนถึงขอบหัวของกุ้ง

Extensive shrimp farming หรือ การเพาะเลี้ยงแบบดั้งเดิม เป็นการเพาะเลี้ยงที่อาศัยอาหาร และระบบการเปลี่ยนน้ำโดยอาศัยธรรมชาติ ความหนาแน่นของกุ้งที่ปล่อยลงบ่อเลี้ยงไม่สูงนัก ใช้พื้นที่การเพาะเลี้ยงมาก และผลผลิตต่อไร่ต่ำ

eyestalk ablation หรือ การบีบตาแม่กุ้งเพื่อเร่งให้ไข่แก่ เป็นวิธีการเร่งแม่กุ้งให้มีไข่แก่ในบ่อวิธีหนึ่ง เนื่องจากที่บริเวณโคนก้านตากุ้งมีฮอร์โมนยับยั้งการเจริญของรังไข่อยู่ซึ่งในธรรมชาติฮอร์โมนชนิดนี้จะลดลงเมื่อสภาพแวดล้อมมีการเปลี่ยนแปลง เช่น ความเค็มลดลงเมื่อแม่กุ้งย้ายเข้ามาวางไข่ในเขตน้ำกร่อย

feeding หรือ การให้อาหารกุ้ง เป็นขั้นตอนที่สำคัญในการเพาะเลี้ยงกุ้งแบบพัฒนาทุกระยะ เนื่องจากมีความหนาแน่นของกุ้งสูง ตั้งแต่การเพาะฟักลูกกุ้ง จนถึงกุ้งโตเต็มวัยสิ่งที่ต้องคำนึงคือคุณค่าทางอาหารที่สำคัญ และคุณลักษณะของอาหารที่เหมาะสมต่อนิสัยการกินอาหาร และสภาพแวดล้อมในบ่อเลี้ยง

intensive shrimp farming หรือ การเพาะเลี้ยงกุ้งแบบพัฒนา เป็นการเพาะเลี้ยงกุ้งที่พัฒนาเป็นขั้นตอนสุดท้าย โดยทำการเพาะเลี้ยงจากลูกพันธุ์กุ้งที่ได้จากโรงเพาะฟัก มาปล่อยลงเลี้ยงด้วยความหนาแน่นสูง การควบคุมน้ำจะใช้เครื่องสูบน้ำเพื่อเปลี่ยนถ่ายน้ำได้ทันเวลา มีการให้ออกซิเจนแก่น้ำในบ่อ อาหารที่ให้เสริมมีทั้งอาหารเม็ด และอาหารสดที่มีสารโปรตีนสูง ผลผลิตต่อไร่สูง

larval rearing หรือ การหาพันธุ์ลูกกุ้ง เป็นการหาพันธุ์กุ้งที่จะนำมาทำการเพาะเลี้ยงมีหลาย

แหล่งที่มา เช่น อาจทำการเพาะพันธุ์ลูกกุ้งเองในโรงเพาะฟัก ซื้อลูกพันธุ์กุ้งจากโรงเพาะฟักอื่นๆ หรือ ลูกกุ้งที่ติดมากับน้ำที่เปิดเข้าสู่บ่อ เป็นต้น การหาลูกพันธุ์ต้องคำนึงถึงขนาด ความแข็งแรงสมบูรณ์ของลูกกุ้ง และโรคกุ้งด้วย

life cycle หรือ วงจรชีวิต เป็นขั้นตอนการเจริญเติบโตของกุ้งตั้งแต่เริ่มต้นจนถึงโตเต็มวัยและออกไข่อีกครั้งหนึ่งถือว่าครบวงจร มีขั้นตอนทั้งหมด 6 ขั้นตอนคือ ระยะเอมบริโอ ระยะลาวาร์ ระยะจูเวนไนล์ ระยะอะโดเรสเซนต์ ระยะสับอะดัลท์ และระยะอะดัลท์

part per thousand หรือ หนึ่งส่วนในพันส่วน เป็นหน่วยวัดความเข้มข้น เช่น ความเค็มโดยวัดความเข้มข้นของสารบางอย่างที่ละลายในน้ำทะเล

physical and chemincal factor หรือ ปัจจัยทางกายภาพ และทางเคมี เป็นผลกระทบทางนิเวศน์ต่อสิ่งมีชีวิตที่เกิดจากสิ่งไม่มีชีวิต เช่น สารเคมีบางอย่าง หรือ ความเป็นกรด-ด่าง หรือ อุณหภูมิแสงแดด เป็นต้น

Pond management หรือ การจัดการบ่อ เป็นวิธีการดูแลรักษาสภาพแวดล้อมทั้งทางกายภาพ เคมี และชีวภาพของบ่อเลี้ยงกุ้ง เพื่อให้เกิดสภาพแวดล้อมที่เหมาะสมกับการเจริญเติบโตและการอยู่รอดของกุ้งมากที่สุด ถ้าพบว่าไม่เหมาะสมจะต้องมีการทำการแก้ไขในทันทีแล้วแต่สาเหตุของปัญหานั้นๆ

semi-intensive shrimp farming หรือ การเพาะเลี้ยงกุ้งแบบกึ่งพัฒนา เป็นการเพาะเลี้ยงที่ดัดแปลงมาจากการเพาะเลี้ยงแบบดั้งเดิม มีการให้อาหารเสริมร่วมกับอาหารธรรมชาติ การควบคุมน้ำจะใช้เครื่องสูบน้ำช่วยร่วมกับการเปิดน้ำตามธรรมชาติ ผลผลิตที่ได้จะสูงกว่าแบบดั้งเดิม

shrimp maturation หรือ การเจริญพันธุ์ของกุ้ง คือ การที่ไข่ในรังไข่ของแม่พันธุ์กุ้งมีการเจริญเติบโตเต็มที่พร้อมจะรับการผสมกับสเปอร์มของตัวผู้

shrimp rematuration หรือ การเจริญพันธุ์ของกุ้งอีกครั้ง เป็นการที่แม่พันธุ์กุ้งที่เคยทำการเร่งให้ไข่แก่แล้วออกไข่ไม่สมบูรณ์ จะสามารถมีไข่แก่ได้อีกครั้งหลังการออกไข่

shrimp nursing หรือ การอนุบาลลูกกุ้ง เป็นการดูแลลูกกุ้งในช่วงแรกหลังจากที่ได้พันธุ์กุ้งมาแล้ว เพื่อให้ลูกกุ้งที่ได้มามีการเจริญเติบโต และปรับตัวให้แข็งแรงก่อนจะปล่อยลงเลี้ยง

ภาคผนวก ง

ตารางเปรียบเทียบส่วนประกอบระหว่างระบบเชี่ยวชาญ MYCIN กับระบบเชี่ยวชาญ SFES

ภาคผนวก ง.นี้เป็นการเปรียบเทียบส่วนประกอบต่างๆ ของระบบเชี่ยวชาญ MYCIN กับระบบเชี่ยวชาญ SFES ที่ทำการวิจัยขึ้น โดยแยกเป็นหัวข้อต่างๆ ดังนี้

1. ความรู้ของระบบ
2. การใช้กฎให้คำปรึกษา
3. การกระจายต้นไม้เนื้อความ
4. การเลือกวิธีการแก้ไขปัญหา

ในแต่ละหัวข้อจะมีหัวข้อย่อยอีกดังตารางต่อไปนี้

ส่วนประกอบของ ระบบ MYCIN	ส่วนประกอบของระบบ SFES	
	มี	ไม่มี
1. ความรู้ของระบบ	✓	
1.1 กฎที่ช่วยตัดสินใจ	✓	
การแทนค่าในรูปกฎความรู้	✓	
ความหมายแบบแบคคัส-นอร์	✓	
1.2 การแยกประเภทกฎตาม ชนิดเนื้อความ	✓	
ต้นไม้เนื้อความ	✓	
เนื้อความ	✓	
ชนิดของเนื้อความ	✓	
ความสัมพันธ์ระหว่างกฎกับ ต้นไม้เนื้อความ	✓	
การแบ่งกลุ่มกฎ	✓	
1.3 พารามิเตอร์	✓	
ATTRIBUTE-OBJECT-VALUE	✓	
พารามิเตอร์ 3 ชนิด		
พารามิเตอร์ที่มีค่าเดียว	✓	
พารามิเตอร์ที่มีหลายค่า	✓	
พารามิเตอร์ที่มีค่าใช่-ไม่ใช่	✓	
การแบ่งกลุ่มพารามิเตอร์		✓
คุณสมบัติของพารามิเตอร์	✓	
1.4 ปัจจัยความเชื่อมั่น		✓

ตารางที่ ง.1 ตารางเปรียบเทียบส่วนประกอบระหว่างระบบเชี่ยวชาญ MYCIN กับระบบเชี่ยวชาญ SFES

ส่วนประกอบของ ระบบ MYCIN	ส่วนประกอบของระบบ SFES	
	มี	ไม่มี
ประโยคสมมติฐาน	✓	
ปัจจัยความเชื่อมั่นสำหรับ ประโยคสมมติฐาน		✓
1.5 ฟังก์ชันที่ใช้ประเมินผล ประโยคเงื่อนไข	✓	
1.6 โครงสร้างความรู้สถิตย์		✓
1.7 การแปลความหมายกฎ เป็นภาษาอังกฤษ	✓	
2. การใช้กฎให้คำปรึกษา	✓	
2.1 โครงสร้างควบคุม	✓	
กฎที่แสดงผล และการเรียก ใช้ตัวเอง	✓	
ขอบเขตของงานและกฎ เป้าหมาย	✓	
โปรแกรมย่อย MONITOR	✓	
โปรแกรมย่อย FINDOUT	✓	
โปรแกรมย่อย ASK	✓	
การกำหนดพารามิเตอร์ที่ ผ่านการสอบถามแล้ว	✓	
ข่ายการหาเหตุผล	✓	
การถามคำถามกับผู้ใช้ระบบ	✓	

ตารางที่ ง.1 ตารางเปรียบเทียบส่วนประกอบระหว่างระบบเชี่ยวชาญ MYCIN กับระบบเชี่ยวชาญ SFES (ต่อ)

ส่วนประกอบของ ระบบ MYCIN	ส่วนประกอบของระบบ SFES	
	มี	ไม่มี
คำถามสำหรับพารามิเตอร์ที่มีค่าเดียวหรือใช่-ไม่ใช่	✓	
คำถามสำหรับพารามิเตอร์หลายค่า		✓
ทางเลือกที่ใช้ตอบคำถาม	✓	
2.2 การสร้างฐานข้อมูลพลวัต	✓	
ข้อมูลที่ได้จากผู้ใช้ระบบ	✓	
การตอบไม่รู้		✓
ข้อมูลที่สรุปได้จากระบบ	✓	
CONCLIST	✓	
TRANSLIST		✓
CONCLUDE	✓	
อากิวเมนต์ CONCLUDE	✓	
CNTXT	✓	
PARAM	✓	
VALUE	✓	
TALLY		✓
การแปลความหมายข้อสรุป	✓	
คุณสมบัติของเนื้อความ	✓	
2.3 กฎที่อ้างถึงตัวเอง		✓
2.4 การป้องกันการวนรอบ	✓	

ตารางที่ ง.1 ตารางเปรียบเทียบส่วนประกอบระหว่างระบบเชี่ยวชาญ MYCIN กับระบบเชี่ยวชาญ SFES (ต่อ)

ส่วนประกอบของ ระบบ MYCIN	ส่วนประกอบของระบบ SFES	
	มี	ไม่มี
ระหว่างการหาเหตุผล		
3. การขยายต้นไม้เนื้อความ	✓	
3.1 โครงสร้างข้อมูลที่ใช้กระจายต้นไม้เนื้อความ	✓	
3.2 กระบวนการกระจายต้นไม้เนื้อความอย่างชัดเจน	✓	
3.3 กระบวนการกระจายต้นไม้เนื้อความแบบเบ็นนัย		✓
4. การเลือกวิธีการแก้ไขปัญหา	✓	
4.1 การสร้างรายการวิธีแก้ไขที่เป็นไปได้	✓	
4.2 การเลือกวิธีการแก้ไขปัญหาที่ดีที่สุด		✓

ตารางที่ ง.1 ตารางเปรียบเทียบส่วนประกอบระหว่างระบบเชี่ยวชาญ MYCIN กับระบบเชี่ยวชาญ SFES (ต่อ)

ประวัติผู้เขียน

นาย สมศักดิ์ ยศสมบัติ เกิดปี พ.ศ. 2504 สำเร็จการศึกษาวิทยาศาสตร์บัณฑิต ภาควิชาวิทยาศาสตร์ทางทะเล คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปี พ.ศ. 2526