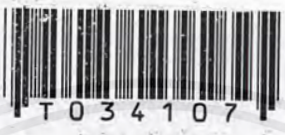
สำนักหอสมุดกลาง_พระจอมเกล้าลาดกระบัง

ระบบผู้เชี่ยวชาญโดยการอ้างเหตุผลด้วยกรณี

EXPERT SYSTEM WITH CASE-BASED REASONING



โดย

นาย วีรชัย เหล่าเรืองวัฒนะ

นาย ศิรส สุภาวิตา

อาจารย์ที่ปรึกษา

ผศ.ดร. เอื้อน ปิ่นเงิน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2541

เลขหมู่.....
เลขทะเบียน 34107
วัน, เดือน, ปี - 5 ต.ค. 2542

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2541

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบผู้เชี่ยวชาญโดยการอ้างเหตุผลด้วยกรณี

EXPERT SYSTEM WITH CASE-BASED REASONING

ผู้จัดทำ

1. นาย วีรชัย เหล่าเรืองวัฒนะ รหัสประจำตัว 38014475
2. นาย ศิรส สุภาวิตา รหัสประจำตัว 38014492

อาจารย์ที่ปรึกษา

(ผศ.ดร. เอื้อน ปิ่นเงิน)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบผู้เชี่ยวชาญโดยการอ้างเหตุผลด้วยกรณี

วีรชัย เหล่าเรืองวัฒนะ 38014475
ศิรส สุภาวิตา 38014492
ผศ.ดร. เอื้อน ปิ่นเงิน อาจารย์ที่ปรึกษา
ปีการศึกษา 2541

บทคัดย่อ

ระบบผู้เชี่ยวชาญโดยการอ้างเหตุผลด้วยกรณีนั้น จัดทำขึ้นเพื่อทำการศึกษาทฤษฎี ซึ่งจะประกอบด้วยระบบผู้เชี่ยวชาญที่เป็นส่วนหนึ่งของวิชาปัญญาประดิษฐ์ ทางด้านการอ้างเหตุผลด้วยกรณีก็จะเป็นการศึกษาวิธีนำกรณีมาใช้ประโยนช์ ซึ่งต้องมีการทำไลบรารีของกรณี ดัชนีของกรณี และต้องมีวิธีการในการดึงกรณีมาใช้ให้เกิดประโยชน์ซึ่งจะมีการปรับปรุงกรณีให้มีความทันสมัยพร้อมสำหรับการแก้ไขปัญหา

นอกจากส่วนของทฤษฎีแล้ว ยังมีส่วนของการประยุกต์ใช้ โดยได้นำความรู้ทางด้านการวินิจฉัยโรค ซึ่งจะแบ่งรายละเอียดของแต่ละโรคออกเป็นกรณี แล้วให้ความสำคัญของกรณีไว้ไม่เท่ากันตามความสำคัญของอาการรักษาโรค และมีวิธีการวินิจฉัยโรค การปรับปรุงให้เข้ากับอาการของโรคใหม่ๆที่ไม่มีในฐานข้อมูล รวมทั้งการเปลี่ยนแปลงแก้ไขให้กรณีมีความทันสมัยโดยอาศัยผลลัพธ์จากการนำไปใช้เพื่อให้เหมาะแก่การวินิจฉัยโรคได้อย่างมีประสิทธิภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Expert System with Case-based Reasoning

Mr.Weerachai Lawroungwattana
Mr.Siros Supavita
Asst.Prof.Dr.Ouen Pinngern Advisor

ABSTRACT

Expert System with Case-Base Reasoning is studied and introduced. In this thesis we will explain an Expert System which is a branch of Artificial Intelligent. Case-Base Reasoning is used for apply in various applications. Case library and case indexing will be created for retrieving and adapting case for the best solution. Updating is the last component for reality case, correctly and capability, which use for new case.

Moreover, a diagnosis system with Case-Base Reasoning is developed. Case library contained real conditions. Each case is weighted by priority for diagnosis. Adaptation use for new case, which is not in case library, for new solution. Updating case use feedback from users to develop case library for future use.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ขอขอบพระคุณบิดามารดาที่ให้พวกเราลืมตาดูโลก ที่น่าอัศจรรย์

อาจารย์ เอื้อน ปิ่นเงิน ที่ให้โอกาสพวกเราได้สร้างสรรค์งาน

คุณหมออัครพันธ์ สุรวิลาศ แห่งโรงพยาบาลกรุงเทพ ที่เล็งเห็นประโยชน์และให้คำปรึกษาเป็นอย่างดี

คณาจารย์ทุกท่านที่ให้ความรู้ต่างๆ ให้เข้าใจถึงโลกที่กว้างใหญ่

ขอบใจเพื่อนๆ Gang4D เยอะจนมิสามารถพิมพ์ลงในนี้ได้หมด ที่ให้ความเป็นกันเอง ความรัก ความเข้าใจ เอื้อเฟื้อเผื่อแผ่ ขนมอร่อยๆ และอื่นๆอีกมากมากมาย

วีรชัย เหล่าเรืองวัฒนะ

ศิรส สุภาวิตา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้าที่
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญภาพ	VI
บทที่ 1 บทนำ	1
1.1 ความเป็นมา	1
1.2 วัตถุประสงค์	2
1.3 ขอบเขตของโครงงาน	2
บทที่ 2 ระบบผู้เชี่ยวชาญ	3
2.1 ระบบผู้เชี่ยวชาญคืออะไร	3
2.2 องค์ประกอบของผู้เชี่ยวชาญ	4
2.3 ข้อจำกัดของระบบผู้เชี่ยวชาญ	5
2.4 ลักษณะงานที่เหมาะสมกับระบบผู้เชี่ยวชาญ	7
2.5 ขั้นตอนการพัฒนาระบบผู้เชี่ยวชาญ	8
2.6 การจัดหาความรู้	9
2.7 การแทนความรู้	12
2.8 วิธีการอนุมานและการให้เหตุผล	15
2.9 ค่าความแน่นอน	16
2.10 ฟัซซีโลจิก	17
2.11 ตัวอย่างการใช้งานระบบผู้เชี่ยวชาญ	18
2.12 บทสรุป	19
บทที่ 3 ทฤษฎีและหลักการ	21
3.1 การอ้างเหตุผลด้วยกรณี	21
3.2 กระบวนการและการเกิดผล	24
บทที่ 4 กรณี	28
4.1 การแทนดัชนี	28
4.2 ส่วนประกอบของกรณี	28
4.3 สารบัญของการแก้ปัญหา	29
4.4 ลำดับการหาเหตุผล	29
4.5 การหาเหตุผลสนับสนุน	29

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

		หน้าที่
	4.6 การทำดัชนีคำศัพท์	30
	4.7 คุณสมบัติที่ดีของดัชนี	31
	4.8 การเลือกคำศัพท์	31
	4.9 วิธีการเลือกดัชนี	32
บทที่ 5	การดึงกรณีมาใช้	34
	5.1 โครงสร้างองค์กรและวิธีการดึงมาใช้	34
	5.2 การจับคู่และการจัดลำดับของกรณี	38
	5.3 การจัดดัชนีและการเรียกคืนกรณี	40
	5.4 การประเมินสถานการณ์ที่เข้ามาใหม่	41
	5.5 การจัดการดัชนี	42
	5.6 ประสิทธิภาพ ความแม่นยำ และความยืดหยุ่น	42
บทที่ 6	การใช้กรณีและการดัดแปลงกรณ์	44
	6.1 วิธีและกลยุทธ์การดัดแปลง	44
	6.2 การควบคุมการดัดแปลง	48
บทที่ 7	การออกแบบไลบรารีของกรณีและโครงสร้างข้อมูล	52
	7.1 หลักการทำดัชนี	52
	7.2 การออกแบบโครงสร้างข้อมูล และอัลกอริธึมสำหรับไลบรารีของกรณี	54
	7.3 การออกแบบไลบรารีของผลลัพธ์ของกรณี	62
บทที่ 8	การออกแบบอัลกอริธึม	64
	8.1 การออกแบบอัลกอริธึมสำหรับเรียกค้นไลบรารีของกรณี	64
	8.2 การออกแบบอัลกอริธึมในการดัดแปลงผลลัพธ์	69
บทที่9	การออกแบบการทำงานของระบบโดยรวม	72
	9.1 การทำงานของระบบโดยรวม	72
บทที่10	ผลการทดสอบ	82
บทที่ 11	บทวิจารณ์และสรุป	84
ภาคผนวก	ตัวอย่างกรณีที่ใช้ในระบบ	86
บรรณานุกรม		92

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ

	หน้าที่
รูปที่ 2-1 ส่วนประกอบของระบบผู้เชี่ยวชาญ	5
รูปที่ 3-1 กระบวนการของการอ้างเหตุผลเชิงกรณี	26
รูปที่ 7-1 โครงสร้างของไลบรารีของกรณี	53
รูปที่ 7-2 โครงสร้างทางข้อมูลโดยรวมของกรณี	56
รูปที่7-3 แสดงโครงสร้างของไลบรารีของกรณี โดยพิจารณาลักษณะของกรณีเป็นอาร์เรย์ซ้อนกัน 2 ชั้น	58
รูปที่ 7-4 แสดงลักษณะของไลบรารีกรณีในบัฟเฟอร์ โดยบล็อกความยาวหนึ่งหน่วยแทนข้อมูล 1 ไบต์	61
รูปที่ 9-1 หน้าจอเลือกไฟล์ที่จะใช้เป็นไลบรารี	73
รูปที่ 9-2 หน้าจอก่อนเข้าสู่การทำงานของหน้าจอหลัก	73
รูปที่ 9-3 หน้าจอหลักของการทำงาน	74
รูปที่ 9-4 หน้าจอรับข้อมูลอาการ	75
รูปที่ 9-5 หน้าจอแสดงผลลัพธ์จากการค้นกรณี	76
รูปที่ 9-6 หน้าจอแสดงรายละเอียดของผลลัพธ์	77
รูปที่ 9-7หน้าจอแสดงข้อมูลยา	78
รูปที่ 9-8 หน้าจอแสดงกรณีอื่นๆ ที่อาจเป็นได้	78
รูปที่ 9-9 หน้าจอแสดงรายละเอียดของกรณี	79
รูปที่ 9-10 หน้าจอรับกรณีใหม่ ในกรณีที่ผลลัพธ์ไม่ถูกต้อง	79
รูปที่ 9-11 หน้าจอรับข้อมูลโรค	80
รูปที่ 9-12 หน้าจอรับข้อมูลยา	81

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่1
บทนำ

1.1 ความเป็นมาของโครงงาน

ความรู้ทางด้านปัญญาประดิษฐ์ได้มีการพัฒนาอย่างแพร่หลายต่อเนื่องตั้งแต่ช่วงปลายศตวรษที่ 20 โดยมีการนำไปประยุกต์ใช้ทั้งทางด้านธุรกิจ อุตสาหกรรม การธนาคาร การเงิน รวมไปถึงการแพทย์ ระบบผู้เชี่ยวชาญก็เป็นอีกสาขาหนึ่งในศาสตร์ของปัญญาประดิษฐ์ ที่สามารถแบ่งออกเป็นอีกหลายสาขา ทั้งนิวรอนเน็ตเวิร์ค(Neuron network) ฟัซซีโลจิก(Fuzzy logic) เจเนติกอัลกอลิทึม(Genetic Algorithm) การอ้างเหตุผลด้วยกรณี(Case-base reasoning) และอื่นๆ

สมัยก่อนประวัติศาสตร์การยังมีการดำรงชีวิตโดยการลองผิดลองถูกเป็นวิธีการที่ต้องใช้ ทั้งอาหารการกิน การเพาะปลูกกสิกรรม และในส่วนต่างๆมากมาย สืบกันมาเรื่อย เมื่อค้นพบวิธีที่ดีหรือสิ่งที่ดีกว่าก็มีการประยุกต์ใช้ และในปัจจุบันยังให้ความสำคัญกับวิชาประวัติศาสตร์ เพราะประวัติศาสตร์มักจะซ้ำรอย ซึ่งระบบอ้างเหตุผลด้วยกรณี เป็นการอ้างถึงสิ่งต่างๆที่เคยเกิดขึ้นมาแล้วมาประยุกต์ใช้เพื่อให้เกิดประโยชน์ ซึ่งวงการแพทย์ก็มีปรากฏการณ์นี้เป็นประจำ ผู้ป่วยมาพบแพทย์ด้วยอาการคล้ายเดิม เช่น อาจปวดหัวตัวร้อน ปวดท้อง ซึ่งจะใช้วิธีเดิมๆในการรักษาจึงเป็นต้นเหตุของโครงงานนี้

เนื้อหาของโครงงานนี้จะกล่าวถึงระบบผู้เชี่ยวชาญโดยเป็นรายละเอียดทั่วๆไป เช่น การสร้างระบบผู้เชี่ยวชาญ ส่วนประกอบของผู้เชี่ยวชาญ กลไกการอนุมานโดยจะกล่าวถึงวิธีการอนุมาน การแทนความรู้ การรับความรู้ และวิธีประกอบรวมกันให้สมบูรณ์

ทฤษฎีของการอ้างเหตุผลด้วยกรณีที่ต้องศึกษาเพื่อสามารถนำมาประยุกต์ใช้ ซึ่งจะมีส่วนของการอ้างเหตุผลของกรณี , ไลบรารีของกรณีที่ทำหน้าที่เก็บกรณีที่เกิดขึ้นและน่าสนใจเหมาะกับการประยุกต์ใช้ , การทำดัชนีของกรณีเพื่อช่วยในกระบวนการคิดและการนำไปใช้ , การดึงกรณีมาใช้ให้เหมาะสมกับกรณี วิธีการคิดของระบบ การปรับปรุงให้เข้ากับสถานะการณ์จริงที่ยังไม่มีในกรณี การเปลี่ยนแปลงแก้ไขให้เหมาะสมและทันสมัยกับสถานะการณ์ใหม่

และส่วนสุดท้ายจะเป็นส่วนของชิ้นงานจริง ลักษณะการออกแบบไลบรารีของกรณี โดยอาศัยอาการของแต่ละโรคที่เกิดขึ้นจริงและน่าเชื่อถือ โครงสร้างข้อมูลของระบบซึ่งจะเก็บเป็นรหัสเพื่อให้ง่ายต่อการใช้ และเหตุผลในการออกแบบ , ส่วนที่สองจะเป็นลักษณะวิธีการดึงกรณีขึ้นมาใช้ ว่าลักษณะอาการที่เกิดขึ้นใหม่นั้นตรงกับอาการที่เกิดขึ้นมาแล้วมากน้อยเพียงใด และทำการปรับปรุงให้เข้ากับกรณีที่เคยเกิดขึ้นให้มีค่าที่ดีที่สุด ตลอดจนถึงการนำข้อมูลใหม่มาเปลี่ยนแปลงแก้ไขให้ระบบมีความน่าเชื่อถือ คือเมื่อมีการเปลี่ยนแปลงของโรค หรือของอาการ ซึ่งอาจถือได้ว่าเป็นกรณีใหม่ จะต้องมีวิธีการในการจัดเก็บลงไลบรารีของกรณีให้ดีและเหตุผลในการเลือกวิธีที่ใช้ต่างๆ , ส่วนของการติดต่อกับผู้ใช้ เพื่ออำนวยความสะดวกแก่ผู้ใช้ , การทดสอบถึงคุณภาพและประสิทธิภาพในการวินิจฉัยโรคเพื่อความปลอดภัย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อศึกษาการทำงานของ การอ้างเหตุผลด้วยกรณี (Case-based Reasoning) ให้ได้เข้าใจถึงลักษณะการสร้างระบบ
2. เพื่อศึกษาการสร้างระบบผู้เชี่ยวชาญและสร้างระบบผู้เชี่ยวชาญเพื่อประยุกต์ใช้กับงานจริง โดยใช้การอ้างเหตุผลด้วยกรณี
3. เพื่อสร้างระบบให้ใช้งานได้อย่างแพร่หลาย ทั้งนี้เป็นการช่วยลดปัญหาการขาดบุคลากรด้านการแพทย์ และยกระดับคุณภาพชีวิตให้สูงขึ้น

1.3 ขอบเขตของโครงงาน

1. ทำการศึกษาหลักการของการสร้างระบบผู้เชี่ยวชาญ
2. การอ้างเหตุผลด้วยกรณีแล้วทำการสร้างระบบขึ้นมาให้ทำงานตามสาขาการประยุกต์ใช้ที่เลือกขึ้นมา ซึ่งสาขาที่เลือกทำการศึกษา
3. ระบบสามารถวินิจฉัยโรคจากอาการบางชนิด ซึ่งตัวระบบที่จะทำการพัฒนาจะวิเคราะห์อาการ แล้วให้คำวินิจฉัยโรค โดยที่จะจำกัดขอบเขตของโรคด้วยการพิจารณาอาการบางชนิดเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ระบบผู้เชี่ยวชาญ (Expert System)

การศึกษาถึงระบบผู้เชี่ยวชาญนี้ จำเป็นที่จะต้องศึกษาถึงลักษณะเฉพาะของระบบผู้เชี่ยวชาญก่อน เพื่อที่จะได้เข้าใจถึงจุดเด่น และข้อจำกัด ของระบบผู้เชี่ยวชาญ รวมทั้งข้อแตกต่างระหว่างระบบผู้เชี่ยวชาญกับระบบข้อมูลทั่วไป ในส่วนแรกของบทความนี้จะกล่าวถึง ลักษณะเฉพาะของระบบผู้เชี่ยวชาญ และข้อจำกัดต่างๆ

ขั้นตอนการพัฒนาระบบผู้เชี่ยวชาญ สำหรับงานในแต่ละแบบนั้นแตกต่างกันไป ซึ่งรวมทั้งองค์ประกอบของระบบผู้เชี่ยวชาญด้วย แต่โดยทั่วไปก็จะมีส่วนที่เหมือนๆ กันสำหรับทุกระบบ นั่นคือส่วนที่เป็นพื้นฐานของระบบผู้เชี่ยวชาญนั่นเอง

วิศวกรผู้ซึ่งจะเป็นผู้พัฒนานอกจากจะต้องมีความรู้ในสาขาวิศวกรรมคอมพิวเตอร์แล้ว ยังต้องเข้าใจถึงบทบาท และหน้าที่ของตนเอง รวมทั้งมีคุณสมบัติพิเศษอื่นๆ อีก ซึ่งหน้าที่หลักของวิศวกรที่ดูแลระบบผู้เชี่ยวชาญ ก็คือ การรวบรวมความรู้เพื่อจัดสร้างฐานความรู้ ซึ่งเป็นหัวใจของระบบผู้เชี่ยวชาญนั่นเอง

นอกจากทฤษฎีพื้นฐาน แล้วยังมีทฤษฎีอื่นๆ ที่จำเป็นอีก เช่น ค่าความไม่แน่นอน ฟัซซีโลจิก (Fuzzy Logic) ซึ่งเป็นส่วนที่สำคัญในการประยุกต์ใช้บางประเภท ที่ใช้ในโลกความจริง

2.1 ระบบผู้เชี่ยวชาญ คืออะไร

ระบบผู้เชี่ยวชาญ คือ ระบบคอมพิวเตอร์ ที่ใช้แก้ปัญหาต่างๆ ที่ต้องใช้มนุษย์ โดยใช้เทคนิคของการใช้ความรู้ (Knowledge) ข้อเท็จจริง (Fact) และการอ้างเหตุผล (Reasoning) นั่นคือใช้คอมพิวเตอร์มาแก้ปัญหาแทนมนุษย์นั่นเอง

ความรู้ที่ ระบบผู้เชี่ยวชาญ ใช้จะถูกสร้างมาจาก กฎ (Rule) หรือประสบการณ์ (Experience) ที่อยู่ใน ขอบเขตของสาขานั้น โดยกฎ จะเป็นการอธิบายถึงสภาวะ ประสบการณ์จะเป็นการรวบรวมประสบการณ์ในสาขาเฉพาะนั้นๆ

จุดมุ่งหมายของระบบผู้เชี่ยวชาญ คือ

1. ใช้แทนมนุษย์ ในกรณีที่ต้องการใช้ผู้เชี่ยวชาญ ที่มีจำนวนไม่เพียงพอ เพื่อเป็นการลดการสูญเปล่าในการจ้างผู้เชี่ยวชาญในองค์กร

2. ในการที่จะนำความรู้และประสบการณ์ของผู้เชี่ยวชาญหลายๆคนมาใช้ เป็นเรื่องที่ยาก ดังนั้นจึงมีการนำ ระบบผู้เชี่ยวชาญ มาใช้โดยการนำความรู้และประสบการณ์จากผู้เชี่ยวชาญหลายๆ คนมาสร้าง

3. ใช้ในการฝึกหัดผู้เชี่ยวชาญ โดยจะนำ ระบบผู้เชี่ยวชาญ มาใช้ในการสร้างบุคลากรที่เป็นผู้เชี่ยวชาญในสาขานั้นๆ ขึ้นมา

4. ใช้ในงานที่ผู้เชี่ยวชาญไม่สามารถรับผิดชอบได้ เช่น ในการพัฒนาระบบ ผู้เชี่ยวชาญ หลังจากที่ได้สร้างระบบจนสามารถใช้งานได้แล้ว ไม่สามารถรับผิดชอบในส่วนที่ดูแลบำรุงรักษา (Maintenance) ระบบได้ตลอดไป จึงมีการสร้าง ระบบผู้เชี่ยวชาญ ที่คอยให้คำแนะนำในการปรับปรุงระบบ ซึ่งใน ระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผู้เชี่ยวชาญ จะรวบรวมข้อมูลของการพัฒนาระบบไว้เก็บความรู้(Knowledge) เช่น ข้อมูลเกี่ยวกับการออกแบบ , ดาตาโฟล(Data flow) ซึ่งโดยส่วนใหญ่แล้วข้อมูลเหล่านี้ไม่สามารถสร้างเป็นเอกสารได้

5. ในการผลิตผลิตภัณฑ์ใหม่ๆ การออกแบบ และการผลิตจะดำเนินการคนละสถานที่กัน ดังนั้นในขั้นตอนการผลิต ถ้าเกิดปัญหาขึ้นมา จะต้องมีการติดต่อผู้เชี่ยวชาญเพื่อแก้ปัญหา ทำให้ ต้องเสียเวลาในการติดต่อ จึงมีการนำ ระบบผู้เชี่ยวชาญ เข้ามาแก้ปัญหาเหล่านี้

6. ในการตัดสินใจในทางธุรกิจ บางครั้งก็ได้มีการนำ ระบบผู้เชี่ยวชาญ เข้ามาใช้ เพื่อเป็นการหาแนวโน้มของการตัดสินใจ

2.2 องค์ประกอบของ ระบบผู้เชี่ยวชาญ

ใน ระบบผู้เชี่ยวชาญ ต่างๆ กัน จะมีองค์ประกอบที่ต่างกัน ตำราหลายๆ เล่มได้ระบุองค์ประกอบของ ระบบผู้เชี่ยวชาญ ไว้ต่างๆ กัน แต่ส่วนประกอบหลักๆ ของ ระบบผู้เชี่ยวชาญ ที่ใช้กันทั่วไป ได้แก่

1. ส่วนฐานความรู้(Knowledge base) จะเป็นฐานข้อมูลความรู้ ที่รวบรวมกฎ ข้อเท็จจริง และความรู้ที่ได้จากผู้เชี่ยวชาญ ซึ่งโดยส่วนใหญ่จะอยู่ในรูปของกฎ ซึ่งแสดงในรูป ถ้า...แล้ว(IF...THEN) ซึ่งในขั้นตอนของการวางแผนการออกแบบกฎ จะต้องมีการสร้างทรีตัดสินใจ(Decision tree) ที่ใช้อธิบายการตัดสินใจในแต่ละขั้นตอน หลังจากตรวจสอบ ทรีตัดสินใจ อย่างละเอียดแล้ว จึงจะมีการจัดการเป็นกระบวนการของกฎ เพื่อให้แน่ใจว่าจัดลำดับของกฎได้อย่างถูกต้อง

หัวใจของฐานความรู้ คือ กฎ กฎที่แสดงถึงข้อสรุปด้านเดียวที่เป็นไปได้สามารถเขียนได้อย่างไม่ยาก แต่กฎที่แสดงถึงผลสรุปที่เป็นไปได้หลายๆ อย่างจะต้องมีปัจจัยของความเป็นไปได้เข้ามาเกี่ยวข้อง นั่นคือ ค่าความแน่นอน(Confidence factor) ซึ่งมีค่าตั้งแต่ -1 ถึง 1 ซึ่งค่านี้จะบอกถึงความน่าจะเป็นที่จะเกิดผลสรุปนั้นๆ

กฎ โดยทั่วไปจะมี 2 แบบ คือ กฎที่กำหนดขึ้น(Definition rule) และ กฎฮิวริสติก(Heuristic rule) โดยกฎที่กำหนดขึ้นจะเป็นกฎที่แสดงสิ่งที่เป็นจริงแน่นอน แต่กฎฮิวริสติกจะเป็นกฎที่อนุมานได้เป็นระดับของความแน่นนอน นั่นคือมีค่าความแน่นอน เข้ามาเกี่ยวข้องด้วย

2. กลไกอนุมาน(Inference Engine) เป็นส่วนหนึ่งของโปรแกรมคอมพิวเตอร์ซึ่งทำหน้าที่เชื่อมโยงเหตุผลและอนุมานเพื่อหาผลสรุป โดยใช้กฎจากฐานความรู้เป็นหลัก โดยจะใช้หลักการพื้นฐานของการอนุมานคือ "ถ้า A เป็นจริง และถ้า A ชี้นำ (Imply) ให้ B เป็นจริง แล้ว B จะเป็นจริง" ส่วนอนุมานจะตรวจสอบกฎ และรวบรวมกฎ เพื่อหาข้อสรุป ในการอนุมานจะต้องมีส่วนเกี่ยวข้องกับระดับของความแน่นอนเสมอ เพราะกฎส่วนใหญ่จะอยู่ในรูปกฎฮิวริสติก

เรามักจะพบกฎที่มีหลายข้อสรุปเสมอเมื่อระบบค้นหากฎที่เหมาะสมจะใช้ แต่ว่าแต่ละกฎ จะมีระดับของความแน่นอนต่างกัน ซึ่งกลไกอนุมาน จะมีวิธีในการค้นหากฎ 2 แบบ คือ แบควาร์ดเชนนิง (Backward chaining) และ ฟอร์วาร์ดเชนนิง(Forward chaining) แบบแบควาร์ดจะเป็นการเริ่มต้นด้วย ผลลัพธ์ที่ต้องการ แล้วค้นหาไปยังกฎที่สอดคล้องกับผลลัพธ์นั้น นั่นคือ เริ่มต้นด้วยผลที่เกิด(ส่วนตามหลัง THEN) แล้วย้อนกลับไปหาเหตุ(ส่วนตามหลัง IF) ส่วนฟอร์วาร์ดเชนนิงจะเป็นการเริ่มจากข้อมูล ซึ่งจะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

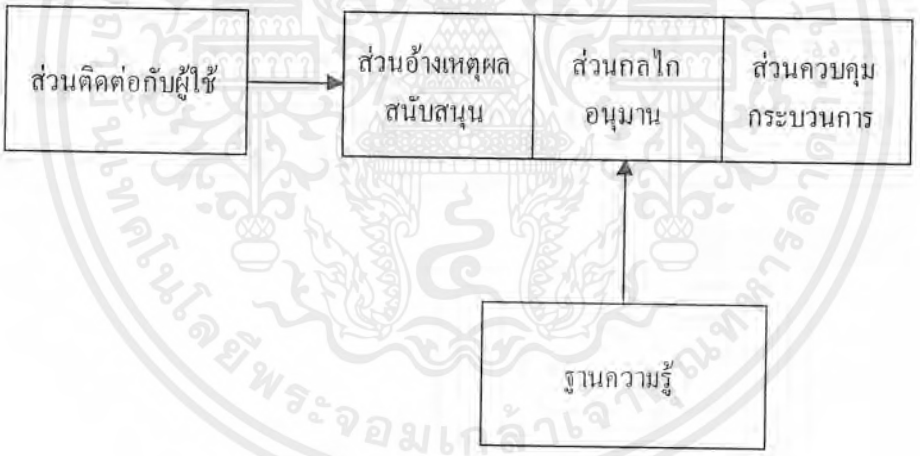
เชื่อมโยงไปสู่ผลลัพธ์สุดท้าย ซึ่งจะเริ่มต้นด้วยเหตุ แล้ว เชื่อมโยงไปสู่ผลลัพธ์ ทั้ง 2 วิธีนี้ จะเป็นการทำให้ได้ผลลัพธ์ที่ "พอใจ" เท่านั้น ไม่ใช่ผลลัพธ์ที่ดีที่สุด

3. ส่วนอ้างเหตุผลสนับสนุนและส่วนควบคุมกระบวนการ(Justifier/Scheduler) มนุษย์ทั่วไป ไม่สามารถอธิบายขั้นตอนของการตัดสินใจได้ แต่ ระบบผู้เชี่ยวชาญ สามารถทำได้ โดยใช้ส่วนอ้างเหตุผลซึ่งเป็นส่วนที่ใช้แสดงเหตุผล เพื่อที่จะให้ผู้ใช้เข้าใจการตัดสินใจ และมีความเชื่อถือในระบบ แต่ก็มีข้อยกเว้นในบางระบบ เช่น ระบบในทางการทหารที่ต้องการการตัดสินใจที่รวดเร็ว ไม่มีเวลาที่จะพิจารณาเหตุผล หรือในระบบที่มีความน่าเชื่อถือสูงมากๆ

อีกส่วนหนึ่งของกลไกอนุมาน คือ ส่วนควบคุมกระบวนการซึ่งทำหน้าที่ควบคุมลำดับของกฎ ซึ่งรายการนี้จะเป็นส่วนที่สำคัญ เพราะ จะเป็นส่วนที่ทำให้การใช้ความรู้ เป็นไปอย่างมีประสิทธิภาพ

4. ส่วนติดต่อกับผู้ใช้ (User Interface) เป็นส่วนที่สื่อสารระหว่างผู้ใช้และระบบ โดยที่ระบบจะถามข้อมูลจากผู้ใช้ โดยที่ผู้ใช้สามารถขอคำอธิบายคำถามจากระบบได้ ส่วนติดต่อกับผู้ใช้(User Interface) เป็นส่วนที่สำคัญ เพราะเป็นส่วนที่จะหาข้อมูลเพิ่มเติม เพื่อที่จะให้ระบบสามารถทำงานได้

ความสัมพันธ์ของส่วนต่างๆ ของส่วนประกอบของระบบผู้เชี่ยวชาญ สามารถแสดงได้ดังรูปที่ 2.1



รูปที่ 2.1 ส่วนประกอบของระบบผู้เชี่ยวชาญ

2.3 ข้อจำกัดของ ระบบผู้เชี่ยวชาญ

คนส่วนใหญ่จะมีความเชื่อและความเข้าใจที่ผิดๆ เกี่ยวกับระบบผู้เชี่ยวชาญซึ่งทำให้เข้าใจข้อจำกัดของ ระบบผู้เชี่ยวชาญ ผิดพลาด ซึ่งความเชื่อเหล่านั้น ได้แก่

1. ระบบผู้เชี่ยวชาญ สามารถเรียนรู้ได้จากประสบการณ์(Experience) ไม่เป็นความจริงเพราะระบบผู้เชี่ยวชาญ จะเรียนรู้ โดยการเพิ่มกฎ และปรับปรุง(update) ฐานความรู้ ถึงแม้ว่าในบางระบบจะมีความสามารถในการเรียนรู้ได้ แต่ก็ไม่เหมือนมนุษย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ระบบผู้เชี่ยวชาญ จะมาแทนมนุษย์ ในกรณีที่ขาดแคลนผู้เชี่ยวชาญจะเป็นความจริง ซึ่งในปัจจุบันนี้ จุดประสงค์หลักของ ระบบผู้เชี่ยวชาญ คือเพื่อเพิ่มประสิทธิภาพให้กับการทำงาน

3. ระบบผู้เชี่ยวชาญ สามารถทำงานได้ดีกว่ามนุษย์ ไม่เป็นความจริง เพราะ ระบบผู้เชี่ยวชาญ ไม่สามารถทำงานแบบความคิดสร้างสรรค์ของมนุษย์ หรือทำงานกับสิ่งที่ไม่พบมาก่อนได้

4. ระบบผู้เชี่ยวชาญ จะไม่ทำให้เกิดข้อผิดพลาด ไม่จริง เพราะ อาจเกิดความผิดพลาด ขึ้นจากกฎได้ โดยเฉพาะกรณีพิเศษต่างๆ

5. เฉพาะวิศวกรความรู้ (Knowledge Engineer) ที่มีความชำนาญเท่านั้น จึงจะสามารถสร้างระบบผู้เชี่ยวชาญ ได้ จะเป็นจริงในกรณีที่เป็นระบบใหญ่ๆ ที่มีความซับซ้อน เพราะปัจจุบันมีเชลล์(Shell) จำนวนมากที่ใช้กัน ทำให้ผู้ที่ไม่มีประสบการณ์ สามารถเรียนรู้ที่จะพัฒนา ระบบผู้เชี่ยวชาญ ได้

6. จะต้องใช้คอมพิวเตอร์เฉพาะในการใช้งาน ระบบผู้เชี่ยวชาญ ไม่เป็นจริง เพราะมีเชลล์ที่หลากหลาย สามารถทำงานได้ทั้งบนพีซี(PC) หรือเมนเฟรม(Mainframe) หรือแม้กระทั่ง ทั้งบนพีซีและเมนเฟรม แต่โดยแนวโน้มแล้ว จะนิยมใช้กันบนพีซี มากกว่า

7. ผู้เชี่ยวชาญไม่สามารถรวมความคิดกันได้ โดยส่วนใหญ่แล้วความรู้ของผู้เชี่ยวชาญจะสามารถรวมกันได้ ทั้งนี้ขึ้นอยู่กับทูล(Tool) ที่วิศวกรนำมาใช้ในการรวบรวมข้อมูลจากผู้เชี่ยวชาญ เช่น การสัมภาษณ์ การอภิปราย

8. ระบบผู้เชี่ยวชาญ ใช้ค่าใช้จ่ายที่สูงมาก จะเป็นจริง ในกรณีที่การวางแผนขาดประสิทธิภาพ ขาดการสนับสนุนจากองค์กร ไม่มีทีมพัฒนาที่มีความสามารถ

9. ระบบผู้เชี่ยวชาญ สามารถอ้างเหตุผลระดับลึกได้ ระบบผู้เชี่ยวชาญ เป็นระบบที่ใช้เหตุผลแค่ระดับตื้นๆ เท่านั้น ดังนั้นจึงไม่เป็นจริง เทคโนโลยีใหม่ๆ ที่สามารถสร้าง ระบบผู้เชี่ยวชาญ ที่อ้างเหตุผลระดับลึกๆ ได้ ต้องอาศัยความร่วมมือจากผู้เชี่ยวชาญและวิศวกรหลายๆ คน ในการพัฒนาระบบ

10 . ระบบผู้เชี่ยวชาญ สามารถแก้ปัญหาได้ทุกชนิด ไม่เป็นจริง เพราะ ระบบผู้เชี่ยวชาญ ใช้ได้กับปัญหาในแนวทางแคบๆ แนวทางหนึ่งเท่านั้น ไม่สามารถใช้แก้ปัญหาที่มีขอบเขตกว้างๆ ได้ ถ้าจะมีการใช้ระบบผู้เชี่ยวชาญ แก้ปัญหาในแนวทางว้างๆ จะต้องมีการศึกษาความเป็นไปได้ และมีการวางแผนเป็นอย่างดี

ซึ่งจากข้อเท็จจริงข้างต้นสามารถสรุปเป็นข้อจำกัดของ ระบบผู้เชี่ยวชาญ ได้ดังนี้

1. ความรู้ที่ต้องการอาจจะหาไม่ได้จริง หมายความว่า ความรู้ที่จำเป็นจะต้องนำมาใช้ในการสร้างระบบ อาจไม่มีวิธีที่จะนำความรู้นั้นมาได้ ทำให้ไม่สามารถสร้างฐานความรู้ได้

2. ขาดความเหมาะสมทางงบประมาณ คือ ผู้บริหารอาจไม่พิจารณาที่จะสร้างระบบขึ้นมา เนื่องจากจะต้องลงทุนสูง

3. ขาดผู้เชี่ยวชาญ ถ้าไม่มีผู้เชี่ยวชาญที่มีความสามารถแท้จริงมาให้ข้อมูล เพื่อที่จะสร้างความรู้ ก็ไม่สามารถสร้าง ระบบผู้เชี่ยวชาญ ได้

4. ขาดข้อสรุปที่ดีจากผู้เชี่ยวชาญ นั่นคือ ถ้ามีประเด็นใดที่ ผู้เชี่ยวชาญหลายๆ คน ลงความเห็นไม่ตรงกัน ก็จะทำให้วิศวกรไม่สามารถตัดสินใจได้ว่า ควรจะเลือกนำความเห็นของใครมาสร้างเป็นความรู้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. ผู้เชี่ยวชาญไม่สามารถอธิบายขั้นตอนการตัดสินใจได้ ทำให้วิศวกรไม่สามารถนำความรู้และประสบการณ์ของ ผู้เชี่ยวชาญ มาสร้างเป็นกฎได้

6. ผู้บริหารระดับสูงไม่สนับสนุน นั่นคือ ผู้บริหารอาจมองไม่เห็นความจำเป็นของระบบ และคิดว่าไม่จำเป็นต้องสร้างระบบ

7. หาวิศวกรความรู้ (Knowledge Engineer) ที่มีความสามารถสูงได้ยาก และต้องลงทุนจ้างสูง เมื่อไม่มีผู้ที่จะวางแผน ออกแบบ และพัฒนาระบบ ก็ไม่สามารถสร้างระบบขึ้นมาได้

2.4 ลักษณะงานที่เหมาะสมกับ ระบบผู้เชี่ยวชาญ

1. การควบคุม(Control) คืองานที่เกี่ยวกับการควบคุมกระบวนการ ซึ่งระบบนั้นจะสังเกตการทำงานของกระบวนการ ถ้ากระบวนการทำงานผิดไปจากปรกติ ระบบผู้เชี่ยวชาญ จะเข้าไปควบคุมการทำงาน เช่น ระบบที่อยู่ในห้องผู้ป่วยอันตราย ที่ต้องคอยสังเกตอาการตลอดเวลา

2. การดีบัก(Debugging) คืองานที่เกี่ยวกับการตรวจสอบหาข้อผิดพลาด แล้วเสนอแนะวิธีแก้ไข ตัวอย่างเช่น ระบบผู้เชี่ยวชาญ ที่ใช้ตรวจสอบการทำงานของเครื่องจักร เมื่อเครื่องจักรทำงานผิดพลาด ระบบผู้เชี่ยวชาญ จะตรวจสอบหาที่ผิดพลาด แล้วเสนอแนวทางแก้ไขแก่ผู้ใช้

3. การวินิจฉัย(Diagnosis) คืองานที่เกี่ยวกับการวินิจฉัย ตรวจสอบ หาสมมติฐาน เช่น ระบบผู้เชี่ยวชาญ ที่ใช้ในการวินิจฉัยผู้ป่วย ซึ่งเมื่อผู้ป่วยบอกอาการ ระบบผู้เชี่ยวชาญ จะสามารถวินิจฉัยได้ว่าป่วยเป็นโรคอะไร

4. คำสั่ง/การฝึกสอน(Instruction/Training) คืองานที่เกี่ยวกับการให้คำปรึกษา และฝึกหัด โดยจะรวบรวมข้อมูลจากผู้เชี่ยวชาญเอาไว้ แล้วเมื่อมีการป้อนคำถามเข้าไป แล้ว ระบบผู้เชี่ยวชาญ จะสามารถตอบคำถามนั้นได้ รวมทั้งบอกเหตุผลได้ด้วย ทำให้สามารถเรียนรู้การตัดสินใจ จาก ระบบผู้เชี่ยวชาญ ได้

5. การตีความ(Interpretation) คือ งานที่เกี่ยวกับการตีความ โดยจะเปรียบเทียบกระบวนการ กับการทำงานปกติ โดยจะทำงานกับข้อมูลที่ยังไม่สมบูรณ์ ซึ่งจะได้สภาวะที่มีค่าความแน่นอนเข้ามาเกี่ยวข้อง

6. การดูแลควบคุม(Monitoring) คือ งานที่เกี่ยวกับการสังเกต ซึ่งจะคอยสังเกตสิ่งผิดปกติ ที่เกิดขึ้นกับสิ่งที่มันสังเกต แล้วจะแจ้งข้อผิดพลาดให้กับผู้ใช้ โดยที่ ท้ายที่สุดแล้ว การตัดสินใจในการแก้ปัญหาก็ยังคงเป็นของผู้ใช้อยู่

7. การวางแผน/กำหนดรายการ(Planning/Schedule) คือ งานที่เกี่ยวกับการวางแผน และ การจัดลำดับในการทำงาน ก่อนที่จะมีการลงมือทำงานจริงๆ นั่นคือใช้เป็นเหมือนที่ปรึกษาในการวางแผนงาน

8. การพยากรณ์(Predication) คือ งานที่เกี่ยวกับการคาดหมายพยากรณ์ โดยจะมีการรวบรวมข้อมูลเป็นสถิติ ประกอบกับข้อมูลอื่นๆ เพื่อหาความเป็นไปได้ในการเกิดเหตุการณ์ต่างๆ ตัวอย่างเช่น ระบบผู้เชี่ยวชาญ ที่ใช้พยากรณ์การเกิดพายุ หรือ แผ่นดินไหว

9. การแก้ไข(Repair) คือ งานที่เกี่ยวกับการซ่อมบำรุงดูแล โดยจะคอยสังเกต ความผิดปกติของการทำงนของระบบ เมื่อพบข้อผิดพลาดแล้วจะทำการซ่อมแซมแก้ไข

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 ขั้นตอนการพัฒนา ระบบผู้เชี่ยวชาญ

ในการพัฒนา ระบบผู้เชี่ยวชาญ มีความแตกต่างกับการพัฒนาระบบทั่วๆไปอยู่หลายอย่าง แต่ส่วนที่แตกต่างกันอย่างชัดเจนคือ ในระบบข้อมูลทั่วไป จะพิจารณาข้อมูลข่าวสารจากผู้ใช้ โดยผู้ใช้จะมีบทบาทอย่างมากในการพัฒนาระบบ ส่วนระบบผู้เชี่ยวชาญนั้น จะสนใจพิจารณาความรู้ ที่ได้รับมาจากผู้เชี่ยวชาญเท่านั้น ข้อแตกต่างอีกด้านก็คือ ระบบทั่วไปไม่นิยมการทำต้นแบบ(Prototype) หลายๆ ครั้ง ซึ่งต่างกับ ระบบผู้เชี่ยวชาญ ที่จะต้องทำต้นแบบขึ้นมาแล้วดัดแปลงแก้ไข จนกว่าระบบจะทำงานได้ถูกต้องเป็นที่พอใจ

จุดเหมือนของขั้นตอนการพัฒนาระบบ ของทั้ง 2 ระบบ คือ ทั้ง 2 ระบบ จะต้องเริ่มต้นด้วยการเก็บข้อมูลเหมือนกัน (ในระบบทั่วไปจะต้องรวบรวมข้อมูลเพื่อกำหนดความต้องการ(requirement) ส่วนใน ระบบผู้เชี่ยวชาญ จะเป็นการรวมความรู้เพื่อสร้างเป็นฐานความรู้)

การพัฒนา ระบบผู้เชี่ยวชาญ แบ่งออกเป็นขั้นตอนได้ดังนี้

1. กำหนดขอบเขตของปัญหา (Identify the Problem Domain) ซึ่งจะเริ่มต้นโดยการหาสิ่งจูงใจหรือเหตุผลในการสร้างระบบ โดยตั้งปัญหาต่างๆ ขึ้นมา เช่น มีผู้เชี่ยวชาญพร้อมที่จะให้ความช่วยเหลือในการสร้างระบบหรือไม่ หลังจากนั้น จะเป็นการระบุขอบเขตของปัญหา โดยจะระบุให้แคบพอที่จะสามารถพัฒนาได้โดยไม่กินเวลาที่มากนัก โดยเฉพาะในองค์กรที่พัฒนา ระบบผู้เชี่ยวชาญ เป็นครั้งแรก

ขั้นตอนต่อไป จะเป็นการสำรวจความเป็นไปได้ในการสร้าง ระบบผู้เชี่ยวชาญ โดยการตั้งคำถามขึ้นมา เช่น สามารถสร้างระบบขึ้นมาได้ในเวลาที่เหมาะสมหรือไม่ หรือ ระบบที่สร้างขึ้นมาให้ผลประโยชน์คุ้มค่าหรือไม่ หลังจากผ่านขั้นตอนนี้แล้ว จะเป็นการสำรวจผู้ใช้ ว่าผู้ใช้จะยอมรับระบบได้มากน้อยเพียงใด สามารถฝึกฝนให้ผู้ใช้สามารถใช้ระบบนี้ได้หรือไม่

2. การรับความรู้ (Knowledge Acquisition) จะเป็นการรับความรู้จากผู้เชี่ยวชาญ มาสร้างเป็นฐานความรู้ โดยจะสร้างเป็น ต้นแบบ ให้ผู้เชี่ยวชาญพิจารณา เพิ่มเติมกฎ แล้วแก้ไขจนกว่าผู้เชี่ยวชาญจะเห็นว่าฐานความรู้นั้นครอบคลุมเพียงพอแล้ว ซึ่งเรียกวิธีนี้ว่าต้นแบบรวบรัด(Rapid Prototyping) ซึ่งจะมีข้อดีคือ สามารถแก้ไขข้อผิดพลาดได้เร็ว และทำให้เกิดความเข้าใจใน ความต้องการของผู้ใช้ ควบคู่ไปกับแนวทางของผู้เชี่ยวชาญ

ส่วนสำคัญอีกส่วนของส่วนรับความรู้(Knowledge Acquisition) คือ การเลือกผู้เชี่ยวชาญที่จะมาให้ความรู้ เพื่อนำมาสร้างฐานความรู้ของระบบ ซึ่งในการเลือกผู้เชี่ยวชาญ จะต้องตั้งคำถามดังต่อไปนี้

- จะรู้ได้อย่างไรว่าผู้เชี่ยวชาญ มีความเชี่ยวชาญในสาขานั้นจริง
- ผู้เชี่ยวชาญสามารถอยู่ร่วมพัฒนาระบบได้จนระบบสมบูรณ์หรือไม่
- ถ้าในกรณีที่ผู้เชี่ยวชาญ ไม่อยู่ร่วมพัฒนาระบบต่อจนระบบสมบูรณ์ จะมีวิธีแก้ไขอย่างไร
- วิศวกรความรู้จะรู้ได้อย่างไรว่าอะไร อยู่หรือไม่อยู่ในขอบเขตความรู้ของผู้เชี่ยวชาญ

3. การแทนความรู้(Knowledge Representation) หลังจากได้รวบรวมความรู้จนพอใจ แล้วก็จะเริ่มสร้างฐานความรู้จนพอใจ แล้วก็จะเริ่มสร้างฐานความรู้ โดยใช้เครื่องมือต่างๆ เช่น เครือข่ายซีเมนติก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(Semantic Network) ,สคริปต์(Scripts) , ทรีตัดสินใจ หรือ ตารางตัดสินใจ(Decision Table) เป็นการเริ่มต้นการออกแบบ

ในฐานความรู้จะแสดงความรู้ได้ 2 แบบ คือ ในรูปของ กฎ (Rule) และ เฟรม (Frame) โดยกฎจะแสดงในรูป ถ้า...แล้ว(IF...THEN) หรือ ถ้า...แล้ว...ไม่แล้ว(IF...THEN...ELSE) ส่วนเฟรมจะเป็นการเก็บความรู้ กฎต่างๆ รวมกันโดยแบ่งเป็นส่วนๆ เรียกว่า สล๊อต(Slot) โดยในสล๊อตจะเก็บสิ่งที่เฉพาะเจาะจงเอาไว้

4. การนำไปสร้างใช้งาน (Implementation) หลังจากที่ได้สร้างฐานความรู้แล้ว ขั้นตอนต่อไปจะเป็นการนำระบบไปสร้างให้ใช้งานบนคอมพิวเตอร์ โดยการนำฐานความรู้มารวมเข้ากับ กลไกอนุมาน โดยเลือก Shell และภาษาที่จะใช้ให้เหมาะสม โดยการสร้างนี้จะเป็นการทำ ต้นแบบ ขึ้นมาก่อนแล้วค่อยดัดแปลงจนกว่าจะได้ระบบที่ทำงานได้ดี " เพียงพอ " ต่อความต้องการ

5. พิสูจน์ความถูกต้องตามความต้องการและทดสอบความถูกต้อง(Verification and Validation) ในส่วนพิสูจน์จะเป็นการทดสอบ ระบบผู้เชี่ยวชาญ ว่าทำงานได้ถูกต้องตามที่ได้ออกแบบหรือไม่ ส่วนของการทดสอบ จะเป็นการตรวจสอบว่าระบบตรงตามความคาดหวังของผู้เชี่ยวชาญหรือไม่ โดยตรวจสอบว่าผลลัพธ์ที่ได้จากฐานความรู้นั้น ใกล้เคียงกับความรู้ของผู้เชี่ยวชาญเพียงพอหรือไม่

6. การดูแลบำรุงรักษา(Maintenance) จะเป็นการปรับปรุงฐานความรู้ เพื่อรักษาประสิทธิภาพของระบบ เช่นเดียวกับมนุษย์ที่สามารถเรียนรู้ และสะสมประสบการณ์ได้

2.6 การจัดหาความรู้ (Knowledge Acquisition)

บุคคลที่สำคัญ และมีผลต่อการรับความรู้มากที่สุด คือ วิศวกรความรู้และผู้เชี่ยวชาญ เพราะทั้ง 2 บุคคลจะต้องร่วมกันสร้างฐานความรู้ขึ้นมา

หน้าที่หลักๆ ของวิศวกรความรู้ คือ

1. กำหนดการใช้งานของระบบ โดยกำหนดวัตถุประสงค์ และวางขอบเขตของระบบ
2. พัฒนาโครงร่างของโครงงาน
3. พัฒนาแผนงานของโครงงาน กำหนดงาน และควบคุมการดำเนินงาน
4. แสวงหาความรู้ที่นำมาสร้างฐานความรู้
5. ประเมินการออกแบบ และวิธีการสร้าง
6. ควบคุมการสร้าง และการทดสอบระบบ
7. ฝึกอบรมผู้ใช้ให้สามารถใช้ระบบได้
8. ปรับปรุง และดูแลรักษาระบบ

วิศวกรความรู้จำเป็นที่จะต้องมีความรู้ต่อไปนี้คือ

1. เทคโนโลยีทางคอมพิวเตอร์ ทั้งด้านฮาร์ดแวร์ ซอฟต์แวร์และระบบปฏิบัติการ หลักการของปัญญาประดิษฐ์ และสถาปัตยกรรมคอมพิวเตอร์

2. ความรู้ในสาขาที่จะทำการพัฒนา เพื่อให้เข้าใจถึงขอบเขตของปัญหา และปัจจัยที่มีผลต่อขอบเขตของปัญหา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. การโปรแกรมเชลล์ และภาษาที่เป็นไม่เป็นโครงสร้าง(Nonprocedural)

4. เครื่องมือ และกลวิธีที่ใช้ในการรับ และแสดงความรู้ เช่น การสัมภาษณ์ผู้เชี่ยวชาญ การวิเคราะห์หลักการ การรวมความคิด การโปรแกรมโดยใช้ Shell หรือภาษา อย่างเช่น โปรลอก(PROLOG) หรือ แอลไอเอสพี(LISP)

5. เทคโนโลยีระบบข้อมูล เช่น วิศวกรรมซอฟต์แวร์(Software Engineering) , การออกแบบระบบ (System Design)

6. การออกแบบส่วนติดต่อกับผู้ใช้และการใช้กราฟฟิก

7. จิตวิทยาการอ้างเหตุผลเบื้องต้น

ในอีกมุมหนึ่ง บุคคลที่มีอิทธิพลต่อการรับความรู้ ไม่แพ้ วิศวกรความรู้ คือ ผู้เชี่ยวชาญ ซึ่งทำหน้าที่เป็นผู้ให้ความรู้แก่ วิศวกรความรู้ นอกจากจะต้องมีความเชี่ยวชาญในสาขานั้นๆ แล้วผู้เชี่ยวชาญจะต้องมีคุณสมบัติเพิ่มเติมต่อไปนี้คือ

1. รู้และเข้าใจว่าเมื่อใดควรจะใช้การประเมิน และเมื่อใดจะใช้ข้อยกเว้น
2. มองภาพรวมกว้างๆ ได้
3. มีทักษะการสื่อสารที่ดี
4. มีความอดทน
5. มีความคิดสร้างสรร
6. มีความเปิดเผยจริงใจ
7. รักษาความน่าเชื่อถือของตนเองได้
8. คิดอย่างมีระบบโครงสร้าง
9. ใช้ความรู้แบบแยกส่วนได้
10. มีแรงบันดาลใจ และความกระตือรือร้นในการทำงาน
11. พร้อมให้ความช่วยเหลืออย่างเต็มใจ
12. มีนิสัยของครูที่ดี นั่นคือ สามารถถ่ายทอดความรู้ได้ดี

สิ่งที่มีผลอีกอย่างหนึ่ง ก็คือ การตัดสินใจเลือกที่จะใช้ผู้เชี่ยวชาญเพียงคนเดียว หรือใช้ผู้เชี่ยวชาญหลายคน ซึ่งก็จะมีข้อดีข้อเสียแตกต่างกัน

ข้อดีของการใช้ผู้เชี่ยวชาญคนเดียวคือ

1. สามารถสร้าง ระบบผู้เชี่ยวชาญ อย่างง่ายได้รวดเร็ว
2. ในสาขาเฉพาะอย่าง จำเป็นต้องใช้ผู้เชี่ยวชาญเพียงคนเดียว
3. สามารถตกลงนัดหมายในการทำงานได้ง่ายกว่า
4. การสื่อสารเป็นไปได้อย่างสะดวก
5. ผู้เชี่ยวชาญคนเดียว จะมีความแน่นอนในการให้คำตอบมากกว่า

ข้อเสียของการใช้ผู้เชี่ยวชาญหลายคนคือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. บางครั้งผู้เชี่ยวชาญไม่สามารถอธิบาย หรือสื่อสารปัญหาบางอย่างได้

2. ผู้เชี่ยวชาญคนเดียวไม่สามารถให้เหตุผลที่ลึกได้

3. ผู้เชี่ยวชาญคนเดียว มีแนวโน้มที่จะขอเลื่อนนัดหมายกำหนดการได้ง่ายกว่า

4. ผู้เชี่ยวชาญคนเดียว อาจมองข้ามจุดบอดบางอย่างไป

ข้อดีของการใช้ผู้เชี่ยวชาญหลายคนคือ

1. ไม่เกิดจุดบอด หรือส่วนที่มองข้ามไป ของปัญหา

2. ได้ความรู้ที่มีคุณภาพมากกว่า

3. ได้มุมมองหลายๆ มุม ทำให้ วิศวกรความรู้ ได้มีโอกาสเลือกแนวทางความรู้

4. การประชุมอย่างเป็นทางการ ทำให้เกิดการรวมความคิด

ข้อเสียของการใช้ผู้เชี่ยวชาญหลายคน คือ

1. สามารถนัดหมายกำหนดการได้ลำบาก

2. เกิดการไม่เห็นพ้องต้องกันได้บ่อย

3. สื่อสารกันได้ลำบาก

4. อาจจำเป็นต้องใช้ วิศวกรความรู้ มากกว่า 1 คน ทำให้ต้องเสียค่าใช้จ่ายเพิ่มขึ้น

วิธีในการรับความรู้มีหลายวิธี เช่น การสัมภาษณ์ (Interviewing), การสังเกตจากการทำงานจริง (On-Site Observation), การรวมความคิด(Brainstorming) แต่ในที่นี้จะกล่าวถึงวิธีที่นิยมใช้กันแพร่หลายที่สุดเท่านั้น นั่นคือ การสัมภาษณ์

การสัมภาษณ์เป็นวิธีที่เก่าแก่ และเป็นที่นิยมกันมากที่สุด ซึ่ง วิศวกรความรู้ จะต้องพิจารณาข้อมูลและสังเกตกระบวนการคิดหาคำตอบของผู้เชี่ยวชาญ ซึ่งการสัมภาษณ์ จะมีประโยชน์ 4 ข้อ คือ

1. มีความยืดหยุ่นสูง ทำให้เป็นวิธีที่สามารถตั้งคำถามแบบต่อเนื่องได้

2. ทำให้ประเมินความน่าเชื่อถือของข้อมูลที่ได้รับมาได้

3. เป็นวิธีที่ดีในการหาข้อมูลจากเรื่องที่ซับซ้อน

4. ส่วนใหญ่คนทั่วไปจะชอบการให้สัมภาษณ์มากกว่า

ข้อเสียที่สำคัญของการสัมภาษณ์ คือ ต้องเสียค่าใช้จ่ายสูง เพราะเป็นวิธีที่สิ้นเปลืองเวลามากที่สุด เนื่องจากต้องการเวลาจากผู้เชี่ยวชาญมากที่สุด

การสัมภาษณ์มี 2 แบบคือ

1. แบบไม่มีโครงสร้าง (Unstructured Interview) ใช้ในกรณีที่ วิศวกรความรู้ ต้องการจะขยายประเด็น โดยใช้คำถามที่เป็นการถามแบบให้ตอบอย่างเปิด แต่วิธีนี้สามารถวางแผนและควบคุมได้ยาก

2. แบบมีโครงสร้าง (Structured Interview) จะใช้ในกรณีที่ วิศวกรความรู้ ต้องการข้อมูลเฉพาะเจาะจง ซึ่งถ้าเป็นกรณีที่ใช้ผู้เชี่ยวชาญหลายๆ คนจะต้องใช้คำถามที่เหมือนกันด้วย คำถามแบบมีโครงสร้าง แบ่งได้เป็น 3 แบบ คือ

- คำถามแบบหลายตัวเลือก (Multiple-choice question)
- คำถามแบบ 2 ทางเลือก (Dichotomous question)
- คำถามเรียงลำดับ (Ranking scale question)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แนวทางในการดำเนินการสัมภาษณ์ แบ่งเป็นขั้นตอนดังนี้

1. การสัมภาษณ์เบื้องต้น (Setting the stage and Establish rapport) เป็นขั้นตอนแรก เรียกว่า " Ice Breaking " เป็นการพูดคุยทักทาย แลกเปลี่ยนความรู้กันอย่างไม่เป็นทางการ โดย วิศวกรความรู้ จะต้องอธิบายถึงจุดมุ่งหมายของการสัมภาษณ์ บทบาทของผู้เชี่ยวชาญ และที่สำคัญ จะต้องสังเกตลักษณะท่าทางของผู้เชี่ยวชาญ เพื่อที่จะได้วางแผนการสัมภาษณ์ได้อย่างมีประสิทธิภาพ

2. จัดการสัมภาษณ์ตามแผนการ (Phrasing the question) เป็นการดำเนินคำถามตามที่ได้มีการเตรียมตัวไว้ล่วงหน้าแล้ว โดยในการใช้คำถาม จะต้องพิจารณาถึง หัวข้อที่จะถาม การใช้คำในคำถาม และรูปแบบของคำถาม

3. ตั้งใจฟัง และหลีกเลี่ยงพฤติกรรมต่างๆ (Good listening and avoiding arguments) ต่อไปนี้

- อย่าขัดจังหวะการอธิบาย หรือการแสดงความเห็นของผู้เชี่ยวชาญ
- อย่าถามคำถามที่เป็นการจู่โจมผู้เชี่ยวชาญ
- หลีกเลี่ยงการใช้ศัพท์เฉพาะทาง ปัญญาประดิษฐ์ หรือ ระบบผู้เชี่ยวชาญ
- พยายามพูดและใช้คำถามที่เป็นการเจาะจงมากกว่าที่จะกล่าวลอยๆ
- พยายามควบคุมการสัมภาษณ์ให้อยู่ในแนวทาง
- หลีกเลี่ยงการแสดงอาการขัดข้องใจ ในคำตอบที่คลุมเครือไม่ชัดเจน
- หลีกเลี่ยงการกล่าวถึงรายละเอียดที่อยู่นอกวาระการประชุม
- อย่างเสแสร้งทำเป็นเข้าใจ
- หลีกเลี่ยงการสัญญาในสิ่งที่ไม่อาจให้ความช่วยเหลือได้

2.7 การแทนความรู้ (Knowledge Representation)

การแทนความรู้จะเป็นการเขียนความรู้ในรูปแบบที่เหมาะสม ซึ่งขั้นตอนนี้เป็นขั้นตอนที่สำคัญที่สุด เพราะระบบจะสามารถให้ผลลัพธ์ได้ถูกต้องหรือไม่นั้นขึ้นอยู่กับว่า วิศวกรความรู้ สามารถเลือกใช้เครื่องมือในการแทนความรู้ได้เหมาะสมเพียงใด

วิธีที่จะใช้แสดงความรู้มีหลายวิธี ซึ่งจะเลือกใช้ตามสถานการณ์ โดยอาจแสดงในรูปกราฟิกหรือตาราง ดังนี้

1. เครือข่ายซีเมนติกจะเป็นการแสดงถึงความสัมพันธ์ระหว่างวัตถุ ซึ่งเรียกว่าโหนด(Node) โดยกลุ่มของโหนด จะเชื่อมโยงกันเป็นเครือข่าย โดยวิศวกรความรู้ จะใช้การเชื่อมโยงนี้แสดงค่าอธิบายถึงความรู้ โดยเส้นเชื่อมระหว่างโหนด จะบอกลักษณะของความสัมพันธ์เส้นนี้เราเรียกว่า อาค(arc)

เครือข่ายซีเมนติกสามารถใช้แสดงการถ่ายทอดจากสิ่งหนึ่งในกลุ่มหนึ่งในยังกลุ่มอื่นที่มันเป็นสมาชิกอยู่ได้ โดยจะแสดงความสัมพันธ์แบบกราฟิก ซึ่งสามารถแสดงความสัมพันธ์แบบ ซับซ้อนได้ แต่อย่างไรก็ดีในระบบคอมพิวเตอร์ไม่สามารถทำเช่นนั้นได้ จึงต้องแสดงในรูปของตารางแสดงความสัมพันธ์ระหว่างโหนด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อดีของการใช้เครือข่ายซีเมนติก คือสามารถแสดงความสัมพันธ์ได้เป็นรูปแบบของลำดับชั้นอย่างชัดเจน และมีความสะดวกในการเพิ่มโหนด ใหม่เข้าไปได้โดยสะดวก ข้อเสียที่สำคัญของ Semantic Network คือ เป็นวิธีการแสดงความรู้ที่ไม่สมบูรณ์ เพราะไม่สามารถแสดงความรู้เชิงปฏิบัติการ (Operational Knowledge) ได้ นอกจากนั้นยังไม่มีการกำหนดมาตรฐานของการกำหนดความหมายของ Node ดังนั้นจึงอาจเกิดความสับสนในการใช้

2.เฟรมจะเป็นการแทนความรู้ด้วยโครงสร้างซึ่งแสดงความรู้แบบรูปธรรมแทนที่จะเป็นกระบวนการ โดยใช้หลักการเหมือนตำราอาหารที่มีทั้งส่วนประกอบและขั้นตอน

เฟรมจะประกอบด้วยสล๊อต และ เฟสเซต(Facet) โดยสล๊อตจะเป็นการอธิบายถึงวัตถุ หรือคุณสมบัติของวัตถุ ส่วนเฟสเซตจะเป็นค่าของสล๊อตโดยในสล๊อตหนึ่งอาจประกอบด้วยเฟสเซตหลายๆ ตัวก็ได้ ซึ่งอาจจะเป็น ขอบเขตของค่าที่เป็นไปได้(Range), ค่าที่กำหนดให้เมื่อไม่มีการกำหนดค่าอื่น (Default), วิธีการตัดสินค่าที่ถูกต้อง (If - needed), ทำอย่างไรเมื่อมีการเปลี่ยนแปลงค่าของสล๊อต(If - changed)

เฟรมจะถูกนำมาเชื่อมต่อกันเป็นลำดับชั้น โดยมีการจัดลำดับเป็นเฟรมลูก(Child Frame) และ เฟรมพ่อ(Parent Frame) เทคนิคในการแก้ปัญหาโดยใช้เฟรม เรียกว่า การจับคู่(matching) โดยการจับคู่ค่าของสิ่งที่ต้องการ กับค่าของสล๊อตในเฟรม

ข้อดีของการใช้ เฟรม คือ สามารถแสดงความรู้ได้เป็นรูปแบบซึ่งสามารถใช้ได้กับความรู้เชิงการประกาศ (Declarative Knowledge) และความรู้เชิงการปฏิบัติการ (Operational Knowledge) ข้อเสียของเฟรม คือความซับซ้อนของเฟรมทำให้การทำงานของมันช้ากว่าวิธีอื่น เพราะกว่าจะเจอเฟรมที่มีค่าที่ต้องการ จะต้องมีการค้นหาผ่านลำดับของเฟรมที่เป็นลำดับชั้น นอกจากนั้นวิธีของการอนุมาน (Inference) และการอธิบายความรู้ยังมีความยุ่งยากอีกด้วย

3. ผลจากกฎ(Production Rule) จะแสดงความรู้ในรูปของกฎโดยให้เงื่อนไขและการกระทำ โดยใช้ ถ้า...แล้ว หรือ ถ้า...แล้ว...ไม่แล้ว โดยในเงื่อนไข (Premise) จะต้องเป็นบูลีน(Boolean) โดยก่อนที่จะเกิดการกระทำใด ๆ ขึ้นจะต้องมีการตรวจสอบเงื่อนไขก่อน โดยที่ในกรณีที่เป็นหลาย ๆ เงื่อนไข เราจะพิจารณาถึงความสำคัญของ และ(AND) ก่อน หรือ(OR) นอกจากนั้นก็ยังต้องพิจารณาถึงเครื่องหมายทางคณิตศาสตร์ด้วยคือ เครื่องหมายคูณ หาร จะต้องมีความสำคัญเหนือเครื่องหมายบวก ลบ

ส่วนการกระทำ (Action) จะเกิดขึ้นหลังจากตรวจสอบเงื่อนแล้ว โดยถ้าเป็นในกรณีที่มีหลาย ๆ การกระทำจะต้องคั่นด้วยเครื่องหมายลูกน้ำ(comma) หรือใช้ AND

ระบบที่ใช้ผลจากกฎ จะเรียกว่า ระบบอ้างโดยกฎ(Rule-Based System) ซึ่งส่วนมากจะเป็นระบบการตัดสินใจทางธุรกิจ เพราะมีความเหมาะสม และสามารถทำความเข้าใจ และบันทึกความรู้ได้ง่าย ข้อเสียของการใช้ผลจากกฎ คือ มีข้อจำกัดในการแสดงความรู้ต่อหนึ่งกฎ

4. ฟอร์มอลโลจิก(Formal Logic) เป็นการใช้ตรรกะในรูปทั่วไปแสดงความรู้โดยการใช้ตรรกะ (Logic) นี้ เป็นการให้เหตุผลแบบที่เก่าแก่ที่สุด โดยมีรูปแบบทั่ว ๆ ไปที่ใช้กัน 2 แบบ คือ

■ ข้อความตรรกะ(Propositional Logic) เป็นรูปแบบของตรรกะทั่ว ๆ ไปแบบพื้นฐานในการแสดงความรู้โดยใช้ตัวเชื่อมเป็น AND , OR , NOT , IF....THEN , EQUIVALENT โดย Proposition

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมายถึง ข้อความ นั่นคือจากข้อความที่มีอยู่ จะนำมาวิเคราะห์หาความถูกต้องของข้อความใหม่ที่ได้อาจได้มาจากการนำข้อความที่มีอยู่แล้วมารวมกัน ด้วยปฏิบัติการทางตรรกะก็ได้

การใช้ข้อความตรรกะมีข้อดี คือสามารถนำตรรกะมารวมกันเพื่อสร้างเป็นตรรกะใหม่ๆ ได้ แต่ก็มีข้อจำกัดในการอธิบายความรู้ ซึ่งบางครั้งเราไม่ต้องการอ้างถึงค่าถูกหรือผิด แต่ต้องการแสดงถึงการเจาะจงวัตถุ หรือความสัมพันธ์ซึ่งนำมาสู่การใช้ predicate logic

■ พรีดิเคตโลจิก(Predicate Logic) เป็นข้อความตรรกะที่นำมาขยาย โดยใช้กฎเดียวกับข้อความตรรกะ แต่มีความสามารถในการแสดงรายละเอียดได้ดีกว่า

พริดิเคต(Predicate) คือประโยคที่อธิบายถึงวัตถุโดยอธิบายถึงตัวมันเองและความสัมพันธ์กับวัตถุอื่น ส่วนวัตถุ (object) ในพรีดิเคตโลจิก จะถูกเรียกว่า ออร์กิวเมนท์(argument) ซึ่งจากการใช้ออร์กิวเมนท์นี้ทำให้เราสามารถเชื่อมโยง ความสัมพันธ์ระหว่างวัตถุหลาย ๆ ชิ้นได้พรีดิเคตโลจิกก็มีค่าความถูกต้องเช่นเดียวกับข้อความตรรกะแต่ค่าความถูกต้องนั้นจะขึ้นกับออร์กิวเมนท์ นั่นคือ จะเป็นจริงสำหรับกลุ่มของออร์กิวเมนท์กลุ่มหนึ่งเท่านั้น นอกนั้นจะเป็นเท็จ

ตัวอย่างเช่น Like (Dave , Finance)

จะแสดงว่า Dave like finance ซึ่งจะเป็นจริงเมื่อมี argument เป็น Dave และ finance เท่านั้น

(Like เป็น Predicate ส่วน Dave และ Finance เป็น argument)

ข้อดีของการใช้พรีดิเคตโลจิก คือ เราสามารถสร้างข้อเท็จจริงใหม่ๆ ได้จากข้อเท็จจริงเดิมที่มีอยู่ และนอกจากนี้พรีดิเคตโลจิก ยังมีรูปแบบที่ชัดเจนและเข้าใจง่าย ทำให้การเพิ่มตรรกะเข้าไปในฐานความรู้เป็นไปอย่างไม่สับสน

ข้อเสียประการสำคัญของการใช้พรีดิเคตโลจิก คือไม่สามารถนำมาใช้ในระบบที่ใช้ในการตัดสินใจในทางธุรกิจได้ เนื่องจากรูปแบบในการแทนความรู้ของมันไม่เหมาะสมนั่นเอง เพราะการหาข้อสรุปจากพรีดิเคตโลจิกจะได้ค่าจริงหรือเท็จ ไม่สามารถให้ค่าในส่วนที่ก้ำกึ่งได้ เหมือนในชีวิตจริง

การใช้ Formal Logic เหมาะสมกับปัญหาที่ใช้ความรู้เพียงตื้นๆ เท่านั้น สำหรับปัญหาที่ใช้ความรู้ระดับลึก การใช้ Semantic Network และ เฟรม ดูเหมือนจะใช้ได้ดีกว่า เพราะ Semantic Network มีความยืดหยุ่นกว่า และแสดงความรู้ ได้ลึกกว่า แม้ว่าการสร้างฐานความรู้ชนิดนี้จะค่อนข้างซับซ้อนก็ตาม ส่วนเฟรม สามารถแสดงศักยภาพในการจัดการระบบระเบียบของข้อมูลได้ดี

5. ตารางตัดสินใจ เป็นวิธีหนึ่งในการแทนความรู้ โดยจะรวบรวมสภาวะที่เป็นไปได้ทั้งหมด และข้อสรุปทั้งหมด มาแสดงในรูปตาราง ซึ่งจะแสดงว่าเมื่อสภาวะใดเกิดขึ้น หรือไม่เกิดขึ้น แล้วจะได้ข้อสรุปอย่างไร นั่นคือ แสดงการเกิด/ไม่เกิด สภาวะด้วย Y หรือ N ส่วนการเลือกว่าจะใช้ข้อสรุปใดจะใช้เครื่องหมาย X ซึ่งเป็นวิธีคล้ายกับการใช้กฎนั่นเอง การสร้างตารางตัดสินใจ จะมีความยุ่งยากซับซ้อนมาก ดังนั้นในการสร้างตารางตัดสินใจ จะต้องใช้ทูลช่วยในการจัดการกฎ

6. ทรีตัดสินใจ จะมีลักษณะเป็นลำดับชั้นคล้ายซีแมนติก และมีความใกล้เคียงกับตารางตัดสินใจ โดยจะมีโหนด ที่แสดงถึงการตัดสินใจ หรือผลสรุป โดยเริ่มเปรียบเทียบจากด้านซ้ายไปขวา

ตารางตัดสินใจจะมีประโยชน์มาในการตรวจสอบกฎ ส่วนทรีตัดสินใจจะใช้ในการตรวจสอบตรรกะทางกราฟฟิกในกรณีที่มีความซับซ้อนมาก เพราะรูปทางกราฟิกมีความสะดวกในการตรวจสอบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มากกว่า นอกจากนั้นยังมีจำนวนข้อสรุปที่จำกัดด้วยทำให้สามารถตรวจสอบได้ง่าย ข้อเสียที่สำคัญของตารางตัดสินใจ คือ ไม่สามารถแสดงรายละเอียดในทรีได้

2.8 วิธีการอนุมานและการให้เหตุผล

กลไกอนุมาน จะเป็นส่วนที่นำความรู้จากฐานความรู้มาใช้ โดยวิธีการนำความรู้มาใช้นี้จะเรียกว่าการอนุมาน (Inference) ซึ่งจะเลียนแบบมาจากการคิดและการให้เหตุผลของมนุษย์ ซึ่งได้แก่

1. เปรียบเทียบระหว่าง 2 สิ่ง
2. ใช้วิธีการสรุป (Inductive) และการหักออก (Deductive)
3. ใช้กฎในรูปถ้า...แล้ว
4. แสดงความรู้ในรูปแบบเฉพาะ
5. บังเอิญในการคิดปัญหาหนึ่ง แล้วได้ผลลัพธ์ของอีกปัญหา

การให้เหตุผล และหาข้อสรุปในระบบผู้เชี่ยวชาญมีด้วยกันหลายวิธี ซึ่งแต่ละวิธีก็ดัดแปลงมาจากการคิด และให้เหตุผลของมนุษย์ดังที่กล่าวมาแล้ว ซึ่งส่วนใหญ่จะเป็นวิธีคิดอย่างง่าย เพื่อให้สามารถจำลองการทำงานได้อย่างไม่ยุ่งยาก และซับซ้อนเกินไป การให้เหตุผลโดยทั่วไปจะใช้หลักการต่อไปนี้

1. การให้เหตุผลด้วยตรรกะ มี 2 แบบ คือ

■ โมดัสโปเนนส์(Modus ponens) เป็นการใช้กฎที่มีอยู่สร้างเป็นข้อเท็จจริงขึ้นมาใหม่ โดยใช้หลักการคือ ถ้าประโยค A และ $A \rightarrow B$ เป็นจริงแล้ว B จะเป็นจริงด้วย

■ โมดัสโปเลนส์(Modus tolens) ใช้หลักการคล้ายกับโมดัสโปเนนส์ แต่เป็นการใช้ในทางตรงกันข้าม นั่นคือ ถ้าประโยค $A \rightarrow B$ เป็นจริง และ B เป็นเท็จ แล้ว A จะต้องเป็นเท็จด้วย

2. การให้เหตุผลโดยการลดรูป (Deductive Reasoning) เป็นการหาผลสรุปจากสิ่งที่เรารู้อยู่แล้วโดยการลดรูป เช่น

IF mother are women AND Sarah is mother

THEN Sarah is a woman.

3. การให้เหตุผลโดยการสรุป (Inductive Reasoning) เป็นการสรุป โดยใช้ข้อมูลเดิมที่มีอยู่แล้วเป็นการสรุปโดยรวม เช่น ชายผู้หนึ่งเข้าไปพบแท็กซี่ในเมืองหนึ่งมีสีฟ้า และสีขาวหลายๆ ครั้ง เขาอาจสรุปได้ว่าในเมืองนี้แท็กซี่จะมีสีฟ้าและสีขาว

การสรุป(Induction) จะมีประโยชน์มากในการค้นหาความรู้ที่ไม่สามารถได้รับจากผู้เชี่ยวชาญได้ ข้อเสียประการสำคัญของการให้เหตุผลโดยการสรุป คือไม่สามารถรับรองได้ว่าผลลัพธ์ที่ได้มานี้มีความถูกต้อง 100 เปอร์เซ็นต์ ดังนั้นจึงต้องมีปัจจัยความไม่แน่นอนเข้ามาเกี่ยวข้องด้วย

การอนุมานจากกฎจะมี 2 วิธี ตามที่ได้กล่าวมาแล้วในตอนต้น คือ

1. แบควาร์ดเชนนิงเป็นการเริ่มต้นจากจุดหมาย คือ ผลสรุป แล้วย้อนกลับไปยังสาเหตุของมัน เรียกว่า goal - driven เปรียบได้กับการทำงานของ เชอร์ล็อก โฮล์ม ซึ่งจะพิจารณาถึงสิ่งที่เกิดขึ้นก่อน แล้วสืบย้อนกลับไปหาสาเหตุ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ฟอร์วาร์ดเชนนิง เป็นการเริ่มต้นจากสาเหตุแล้วเชื่อมไปยังผลลัพธ์ของมัน เรียกว่า data - driven เปรียบเทียบกับการเดินทางของ โคลัมบัส ซึ่งเมื่อเริ่มออกเดินทางได้รับคำสั่งให้เดินทางสำรวจ โดยใช้ความรู้ที่มีอยู่ ทักษะ และข้อมูลเกี่ยวกับการเดินทางเก่าๆ

นอกจากจะใช้ระบบอ้างโดยกฎ แล้วยังมีการใช้ระบบฐานความรู้แบบอื่นๆ อีก ซึ่งการอนุมานจากฐานความรู้แบบอื่นนี้ ก็จะมีวิธีที่แตกต่างกันออกไป

ในระบบที่ทำหน้าที่ตรวจสอบและเฝ้าดูการทำงานของเครื่องจักร การใช้ระบบอ้างโดยกฎอาจเกิดความผิดพลาดได้ เพราะจะใช้เซนเซอร์(Sensor) รับข้อมูลจากเครื่องจักรเข้ามาตรวจสอบในระบบ ซึ่งอาจเกิดข้อผิดพลาดจากการอ่านข้อมูลของเซนเซอร์ได้

ระบบอ้างโดยจำลองรูปแบบ(Model-based system) จึงเกิดขึ้นโดยจะรวมส่วนที่เป็นแบบจำลองการทำงานของเครื่องจักรด้วย วิธีการทำงานของ ระบบอ้างโดยจำลองรูปแบบจะมีการจำลองการทำงานของเครื่องจักรเพื่อหาข้อมูลที่ควรจะเป็น แล้วนำมาเปรียบเทียบกับข้อมูลจากเซนเซอร์

ระบบอ้างด้วยกรณี (Case-based system) ก็เป็นอีกระบบหนึ่งที่มีการพัฒนาขึ้นเพื่อจุดประสงค์เฉพาะ คือ มีการบันทึกกรณีการทำงานเก่าๆ ไว้ เมื่อมีการทำงานกับกรณีที่คล้ายกับ หรือตรงกับกรณีที่เคยพบมาแล้ว จะสามารถดึงข้อมูลเก่านั้นออกมาใช้ได้ทันที วิธีนี้ได้มาจากหลักการเลือกปรึกษาแพทย์ของคนไข้ สำหรับศัลยแพทย์แล้ว คนไข้มักจะเลือกศัลยแพทย์ที่ค่อนข้างมีอายุและมากด้วยประสบการณ์ มากกว่าที่จะเลือก ศัลยแพทย์อายุน้อย เพราะด้วยเหตุผลที่ว่า ศัลยแพทย์ที่มีอายุจะมีประสบการณ์มากกว่า นั่นคือมีกรณีในการรักษามากนั่นเอง

ข้อจำกัดที่สำคัญของระบบอ้างด้วยกรณี คือ ต้องมีการปรับปรุงอย่างสม่ำเสมอเพื่อรักษาความน่าเชื่อถือของระบบ ซึ่งเป็นปัญหาใหญ่ เพราะ ต่างกับมนุษย์ที่สามารถเรียนรู้ได้โดยการทำงาน เราต้องเพิ่มกลไกในการเรียนรู้ให้กับระบบ ซึ่งเป็นเรื่องที่ซับซ้อนมาก

2.9 ค่าความแน่นอน (Certainty Factor,CF)

ในระบบที่มีปัจจัยความไม่แน่นอนเข้ามาเกี่ยวข้องจะต้องมีการอ้างถึง ค่าความแน่นอน ซึ่งจะบอกถึงความน่าเชื่อถือของกฎนั้น ไม่ใช่การแสดงความน่าจะเป็น ซึ่งความไม่แน่นอนนี้เกิดจากสาเหตุต่อไปนี้

1. ความกำกวม (Ambiguity) คือ ประโยคหนึ่งๆ มีหลายความหมาย หรือ ตีความได้หลายความหมาย เช่น ประโยค " Do I turn left here ? " ผู้ตอบคอบว่า " Right " อาจหมายความว่าให้เลี้ยวขวาหรือ ถูกต้อง ให้เลี้ยวซ้ายก็ได้

ความกำกวมนี้ ต่างกับ ความไม่ชัดเจน (Fuzziness) เพราะความไม่ชัดเจน คือ การที่ไม่สามารถระบุความหมาย หรือไม่สามารถตีความระดับได้

2. ความไม่สมบูรณ์ของประโยค

3. การแทนความรู้อย่างไม่ถูกต้อง เกิดข้อผิดพลาด คือ กฎ ถูกต้อง แทนที่จะไม่ถูกต้อง หรือในทางตรงกันข้าม

4. ความผิดพลาดในหน่วยวัด

5. ความผิดพลาดในการอ้างเหตุผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

6. ความผิดพลาดในกฎที่เป็นอิสระ

7. ความผิดพลาดที่เป็นกระทบจากกฎอื่นๆ

ค่าความแน่นอน (CF) จะมีค่าอยู่ระหว่าง +1.0 ถึง -1.0 โดย +1.0 จะหมายถึง เชื่อถือได้ 100 เปอร์เซ็นต์ ส่วน -1.0 ก็จะหมายถึงไม่น่าเชื่อถือ การใช้ค่า CF ใน ระบบผู้เชี่ยวชาญ มีหลายทางดังนี้

1. ใช้โดยรับจากผู้ใช้

2. ผลสรุปจะได้รับการกำหนดค่า CF ในกฎ

3. ผลสรุป จะได้รับ ค่า CF โดยตรงจากสภาวะของมัน

4. CF อาจเกิดจากรวมหลายๆ สภาวะเข้าด้วยกันด้วย AND หรือ OR

การหาค่า CF ของผลสรุปจากสภาวะที่นำมา AND กัน จะได้ค่า CF ไม่เกิน ค่า CF ของสภาวะที่มีค่า CF น้อยที่สุด ในบางระบบจะนำ ค่า CF ของสภาวะที่นำมา AND กัน มาคูณกัน ผลลัพธ์ที่ได้จะเป็นค่า CF รวมของผลสรุป

การหาค่า CF ของผลสรุปจากสภาวะที่นำมา OR กัน จะนำค่า CF ของสภาวะตรงข้ามของสภาวะที่กำหนดให้ มาหาผลคูณ ผลที่ได้จะเป็น ค่า CF ของผลสรุปตรงกันข้ามกับผลสรุปที่ต้องการ

สำหรับ CF ของสภาวะ จะมีการกำหนดว่าต้องมีค่ามากกว่าเท่าไร กฎนั้นจึงจะยอมรับได้ นั่นคือให้ผลสรุปออกมาได้ถูกต้อง ซึ่งโดยทั่วไปถ้าไม่ได้กำหนด จะให้มีค่าเป็น 20 เปอร์เซ็นต์

2.10 ฟัซซีโลจิก(Fuzzy Logic)

ในบางกรณีที่เกิดความไม่ชัดเจนของการใช้กฎจำพวกบรรยายคุณลักษณะ เช่น บอกถึงความสูง ความร้อน บอกจำนวนมาก/น้อย ซึ่งไม่สามารถบรรยายได้ว่า มากเท่าไรจึงจะเรียกว่ามาก ร้อนเท่าไรจึงจะนับว่าร้อน จึงต้องมีการนำฟัซซีโลจิกเข้ามาใช้

ฟัซซีโลจิกเป็นตรรกะชนิดที่สามารถแสดงค่าได้มากกว่าแค่ ถูก หรือ ผิด นั่นคือถ้าเปรียบเทียบถูกผิด เป็นสีขาว สีดำ ฟัซซีโลจิกก็จะเสมือนมีความสามารถในการแสดงสีเทาได้ และสามารถแสดงได้หลายความเข้มด้วย ซึ่งอยู่ในช่วงที่กำหนดให้แสดงได้ ช่วงนี้เรียกว่า กลุ่มสมาชิก(Membership set) ซึ่งใช้แทนที่ค่าเพียง 2 ค่า (ถูก / ผิด) ซึ่งการใช้ฟัซซีโลจิก นี้จะช่วยเพิ่มความคล่องตัวในการสร้างระบบ โดยจะเพิ่มความสามารถในการรองรับระบบที่อาจเกิดความไม่แน่นอนได้

กลุ่มฟัซซี(Fuzzy Set) จะเป็นกลุ่มขององค์ประกอบซึ่งมีความไม่ชัดเจน จะมีค่าที่บ่งชี้ถึงระดับของความเป็นสมาชิกภาพ (Degree of Membership) นั้น โดยจะมีค่าตั้งแต่ 0 ถึง 1 โดยที่ค่า 0 แสดงถึงจะหมายความว่าไม่มีความเกี่ยวข้องกับกลุ่มฟัซซี นั้นเลย ค่า 1 แสดงถึงความเกี่ยวข้องอย่างเต็มที่

การศึกษาการใช้ ฟัซซีโลจิก ควรพิจารณาหลักการที่สำคัญ ซึ่งได้แก่

1. หลักการพิจารณาความสูง ซึ่งไม่อาจตัดสินได้ว่าที่ ความสูงเท่าไรจึง จะจัดว่าสูง ซึ่งนำ ฟัซซีโลจิก มาใช้โดยใช้กราฟเป็นเส้นตรง โดยกำหนดให้ เมื่อความสูงเพิ่มขึ้น ค่าระดับของสมาชิก(Degree of Membership) ก็จะเพิ่มขึ้นด้วย จนถึงที่ความสูงหนึ่งค่านี้ จะมีค่าเป็น 1 และจะไม่เพิ่มขึ้นอีก ส่วนทางด้านซ้ายก็เช่นกัน เมื่อเส้นกราฟลดลงจนระดับของสมาชิก เป็น 0 แล้ว ก็จะไม่ลดลงอีก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

34107

2. หลักการพิจารณาอายุวัยกลางคน จะคล้ายกับข้อแรก เพียงแต่กราฟจะเป็นรูปถ้วยคว่ำ โดยที่อายุค่าหนึ่ง จะมีค่าระดับของสมาชิก เป็น 1 จากนั้นก็ลดลงทั้ง 2 ข้าง

การสร้างระบบที่ใช้ ฟัซซีโลจิก จะต้องเริ่มด้วยการสร้างส่วนที่เป็นกราฟความสัมพันธ์ระหว่างกลุ่มฟัซซี กับระดับของสมาชิกโดยที่ เราจะต้องแบ่งช่วงต่างๆ ในสมาชิก แล้วสร้างกฎขึ้นมาเพื่อใช้กับแต่ละช่วง ด้วยวิธีนี้สามารถทำให้คอมพิวเตอร์สามารถให้เหตุผลได้เช่นเดียวกับมนุษย์

2.11 ตัวอย่างการใช้งานระบบผู้เชี่ยวชาญ

มัยซิน(MYCIN) เป็นระบบผู้เชี่ยวชาญที่พัฒนาขึ้นครั้งแรกในช่วง ค.ศ. 1972 - 1980 ที่มหาวิทยาลัยสแตนฟอร์ด(Stanford) ซึ่งทำหน้าที่ช่วยแพทย์ในการวินิจฉัยสาเหตุของอาการติดเชื้อของคนไข้ และให้คำแนะนำเกี่ยวกับขนาดและชนิดของยาที่ควรให้แก่คนไข้ โดยใช้ข้อมูลจากฐานความรู้ ร่วมกับข้อมูลที่รวบรวมมาจาก ประวัติคนไข้ อาการของโรค และผลการตรวจจากห้องทดลอง

ในการพัฒนามัยซิน ขึ้นครั้งแรกนั้นมีจุดประสงค์ดังนี้

1. เพื่อให้เกิดประสิทธิภาพสูงสุด เพราะจากสถิติการสั่งจ่ายยาของประชากร 1 ใน 4 ของสหรัฐอเมริกาในช่วง ค.ศ. 1970 มีถึง 90 % ที่ได้รับการสั่งยาโดยไม่จำเป็น

2. โปรแกรมที่สร้างขึ้นมาเพื่อเป็นเครื่องมือให้กับแพทย์เท่านั้น มิได้ออกแบบมาเพื่อใช้แทนแพทย์จริงๆ

3. โปรแกรมที่สร้างขึ้นจะต้องมีความสามารถในการเปลี่ยนแปลง และ เพิ่มเติมความรู้ได้

4. ระบบต้องมีความสามารถในการโต้ตอบเป็นการสนทนาได้ รวมทั้งยังต้องมีการอธิบาย ทั้งในความรู้และการแสดงเหตุผล

5. มีความเร็วสูง และสามารถใช้งานได้ง่าย

จากการพัฒนามัยซิน ได้แสดงถึงจุดเด่นของมันหลายประการ ประการแรกคือ ฐานความรู้ถูกสร้างจากกฎ หลายร้อยกฎ ประการที่สอง คือ กฎ เหล่านี้จะเกี่ยวข้องกับความน่าจะเป็น ประการที่สาม คือ มัยซิน สามารถอธิบายกระบวนการให้เหตุผลของมันเองได้

มัยซินสร้างขึ้นโดยการอ้างเหตุผลแบบแบควาร์ดเชนนิง เขียนขึ้นด้วยภาษาแอลไอเอสพี ซึ่งการพัฒนามัยซินมีจุดเริ่มต้นมาจาก ระบบอื่นๆ เช่น เดนดรอล(DENDRAL) หรือ เมดิฟอร์(MEDIPHOR) โดยในมัยซิน นี้ถูกกำหนดให้มีลักษณะเด่น คือ สามารถใช้งานได้เป็นประโยชน์ ใช้ทำการศึกษาได้ สามารถอธิบายและให้คำแนะนำได้ สามารถตอบสนองต่อคำถามทั่วไป มีความสามารถในการเรียนรู้ความรู้ใหม่ๆ และทำการแก้ไขได้ง่าย ซึ่งโดยรวมแล้วมัยซิน จะมีขนาดใหญ่กว่า เดนดรอล และสามารถรองรับการใช้งานที่เกิดความไม่แน่นอนได้

โครงสร้างของ มัยซิน จะมีส่วนสำคัญดังนี้

1. โปรแกรมที่ปรึกษา(Consultation Program) จะรับข้อมูลจากแพทย์ จากอาการของคนไข้ และจากฐานความรู้ที่ใช้ประกอบการตัดสินใจของระบบเข้ามา แล้วประมวลผลส่งไปยังส่วนแสดงการอธิบาย

2. โปรแกรมอธิบาย(Explanation Program) จะรับข้อมูลจากโปรแกรมที่ปรึกษา เข้ามาเพื่อทำการอธิบายเหตุผลของการตัดสินใจของระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. ความจริงสถิตและความรู้ที่ถูกตัดสิน(Static Factual & Judgmental Knowledge) เป็นส่วนที่เก็บความรู้ที่ใช้ประกอบการตัดสินใจของระบบ โดยความรู้นี้จะได้รับมาจาก การรับความรู้โปรแกรม

4. การรับความรู้ โปรแกรมเป็นส่วนที่รับความรู้ใหม่ๆ เข้ามาในระบบ โดยความรู้อาจจะได้รับมาจากผู้เชี่ยวชาญทางสาขานี้โดยตรง หรืออาจจะได้รับจากข้อมูลอาการป่วยของคนไข้ก็ได้

กฎที่ใช้ใน มัยซิน จะอยู่ในรูป ถ้า ... แล้ว ทั่วไป โดยจะแบ่งเป็น 2 ส่วน คือ ส่วนเงื่อนไข (Premise) และส่วนการกระทำ (Action) ตัวอย่างเช่น

PREMISE : ($AND (SAME CNTXT GRAM graaneg)

(SAME CNTXT MORPH rod)

(SAME CNTXT AIR aerobic))

ACTION : (CONCLUDE CNTXT CLASS ENTEROBACTERIACEAE TALLY .8)

การตัดสินใจของระบบ แบ่งออกเป็น 4 ขั้นตอน คือ

1. ค้นหาส่วนที่ผิดปกติ
2. ตรวจสอบว่าอาการผิดปกตินั้น คืออะไร
3. ตัดสินใจว่าสามารถใช้ยาชนิดใดในการรักษาได้บ้าง (โดยใช้กระบวนการพิเศษ)
4. เลือกยาชนิดที่เหมาะสมที่สุด (โดยใช้กระบวนการเฉพาะ)

แนวทางในการพัฒนาเพื่อเพิ่มประสิทธิภาพให้กับ มัยซิน

1. จัดลำดับของกฎแบบไดนามิก(Dynamic)
2. จัดลำดับของสภาวะในกฎแบบไดนามิก
3. คัดเลือกกฎก่อนใช้งานจริง (Prescreening)
4. จัดให้ความรู้ในฐานความรู้ทั้งหมดแสดงอยู่ในรูปของกฎ
5. ใช้ คอนเท็กกราฟ(Context Graph) แทนที่ คอนเท็กทรี(Context Tree) ที่ใช้อยู่ในปัจจุบัน

2.12 บทสรุป

การใช้ระบบผู้เชี่ยวชาญได้นำมาซึ่งความสะดวกสบายในการทำงานในด้านต่างๆ ทั้งลดต้นทุน ประหยัดแรงงาน ลดความยุ่งยากในการบริหาร ซึ่งในอนาคตระบบผู้เชี่ยวชาญจะยิ่งเข้ามามีบทบาทในชีวิตประจำวันของเรามากขึ้นเรื่อยๆ

ระบบผู้เชี่ยวชาญ ในปัจจุบันก็ยังไม่สามารถทำงาน ได้เท่าเทียมกับมนุษย์ ถึงแม้ว่าบางครั้งจะสามารถทำงานได้รวดเร็วกว่า และมีประสิทธิภาพมากกว่า แต่ระบบผู้เชี่ยวชาญก็ยังขาดความสามารถในการเรียนรู้ อีกทั้งยังขาดทักษะ ความคิดสร้างสรรค์ และไม่มีความสามารถแก้ปัญหาที่ไม่เคยพบมาก่อนได้ ซึ่งต่างกับมนุษย์ที่สามารถเรียนรู้และเก็บเกี่ยวประสบการณ์ได้จากการทำงาน นอกจากนั้นความคิดสร้างสรรค์ก็เป็นข้อได้เปรียบประการสำคัญที่มนุษย์มีเหนือระบบผู้เชี่ยวชาญ

ตัวอย่างของระบบผู้เชี่ยวชาญที่มีผลต่อชีวิตประจำวันของเรา ในปัจจุบันนี้ยังคงมีไม่มากนัก แต่ในอนาคต เมื่อมีการใช้ระบบผู้เชี่ยวชาญอย่างแพร่หลายแล้ว ทุกๆ คน ก็คงจะต้องได้รับอิทธิพลจากระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผู้เชี่ยวชาญอย่างแน่นอน เช่น ระบบจัดการการจราจรในท้องถนน อุปกรณ์ที่ใช้ Fuzzy Loic เป็นอุปกรณ์ภายใน ระบบผู้เชี่ยวชาญในทางบริหารธุรกิจ รวมทั้ง ระบบผู้เชี่ยวชาญในทางการแพทย์ด้วย

เทคโนโลยีใหม่ๆ ยังคงเกิดขึ้นทุกๆ วัน ระบบผู้เชี่ยวชาญก็เช่นกัน เพื่อให้เกิดประสิทธิภาพที่สูงขึ้น จึงมีการพัฒนาทฤษฎี และหลักการใหม่ๆ ขึ้นมาเพื่อใช้ในการสร้างระบบผู้เชี่ยวชาญอย่างไม่หยุดยั้ง รวมทั้งการขยายของเขตการใช้งานที่มีอยู่เดิมแล้ว ให้กว้างไกลกว่าเดิมอีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
ทฤษฎีและหลักการ

3.1 การอ้างเหตุผลด้วยกรณี

การอ้างเหตุผลด้วยกรณี เป็นการสะสมประสบการณ์ที่ได้เรียนรู้มา แล้วนำมาประยุกต์ใช้กับสถานการณ์ใหม่ๆ ซึ่งมีความคล้ายคลึงกับเหตุการณ์เดิมที่เคยได้พบมา ซึ่งจะต้องประกอบด้วยหลักการสำคัญดังนี้

- การอ้างอิงถึงกรณี (Case) เก่าๆ เป็นประโยชน์อย่างมากในการจัดการกับเหตุการณ์ที่เกิดขึ้นซ้ำๆ การอ้างอิงถึงกรณีเก่าๆ เหล่านี้จำเป็นที่จะต้องมีการจัดการกับความซับซ้อนของเหตุการณ์ ซึ่งทำให้การจดจำกรณีเพื่อที่จะใช้แก้ปัญหาต่อไปในอนาคต เป็นกระบวนการเรียนรู้ที่สำคัญ
- เนื่องจากคำบรรยายถึงลักษณะของปัญหามักจะไม่สมบูรณ์ จึงจะต้องมีขั้นตอนในการตีความปัญหา ก่อนที่จะมีการค้นหาเหตุผล ซึ่ง การอ้างเหตุผลด้วยกรณี จะไม่สามารถเรียกกรณีซ้ำ (Recall case) ได้ ถ้าไม่เข้าใจปัญหา ซึ่งไม่เฉพาะ การอ้างเหตุผลด้วยกรณี เท่านั้น ไม่ว่าการหาเหตุผลด้วยวิธีไหนก็จะต้องมีการตีความปัญหาเพื่อให้เข้าใจถึงรายละเอียด และเข้าใจอย่างชัดเจนด้วยศัพท์ที่ตัวอ้างเหตุผล (Reasoner) สามารถวิเคราะห์ความรู้เพื่อใช้ในการวิเคราะห์ได้
- เนื่องจากกรณีเก่าๆ ไม่ได้เหมือนกับเหตุการณ์ใหม่ๆ ทั้งหมด จึงจำเป็นที่จะต้องมีการปรับปรุง (Adapt) ผลลัพธ์เก่าๆ เพื่อให้ได้ผลลัพธ์ใหม่
- ถ้ามีการค้นพบกระบวนการใหม่ๆ ซึ่งใช้ได้ดีในการแก้ปัญหาที่ซับซ้อน จะถือว่าเป็นการเรียนรู้กระบวนการใหม่ๆ สำหรับจัดการกับกรณีของสถานการณ์ใหม่ โดยกระบวนการจะถูกเก็บไว้ในไลบรารีของกรณี (Case library) ส่วนตัวระบุลักษณะของกรณีจะเก็บไว้ในดัชนีของหน่วยความจำ(Memory) ซึ่งจะถูกดึงขึ้นมาใช้เมื่อเห็นว่ากระบวนการนั้นใช้ประโยชน์ได้ ซึ่งถ้าเกิดปัญหาในการใช้กระบวนการของกรณี ใหม่นี้ในสถานการณ์ใหม่ ตัวอ้างเหตุผล ก็จะรู้ว่ากระบวนการของกรณี นั้นผิดพลาด หรือ ดัชนี (Index) ของมันไม่แม่นยำ จะต้องมีการวิเคราะห์ผลลัพธ์ของสถานการณ์ใหม่นี้ และแก้ปัญหา แล้วฝังความรู้ที่ขัดเกลาหรือปรับปรุงแล้วลงในไลบรารีของกรณี ซึ่งจะมีดัชนี ระบุว่ากรณี นี้จะเป็นประโยชน์เมื่อไร และ ดัชนีของกรณี เก่าๆ ก็จะถูกปรับปรุงตามจากที่ได้วิเคราะห์ ซึ่งยิ่งมีการเรียนรู้มากขึ้นก็จะยิ่งทำให้การดึงกรณี ขึ้นมาใช้เกิดประสิทธิภาพมากขึ้น
- การติดตามผลจากการใช้กรณี ถือเป็นส่วนสำคัญ เพราะถ้าหากไม่มีการควบย้อนกลับ (Feedback) แล้วการเรียนรู้ก็จะไม่เกิดขึ้น ทำให้การอ้างอิงถึงกรณีเก่าๆ ไม่น่าเชื่อถืออีกต่อไป

คุณภาพของการให้คำตอบของการอ้างเหตุผลด้วยกรณี ขึ้นอยู่กับปัจจัย 5 ข้อ คือ

1. ประสบการณ์ที่ตัวมันมี หรือกรณีนั่นเอง
2. ความสามารถในการเข้าใจสถานการณ์ใหม่ในรูปแบบของประสบการณ์เก่า
3. ความอ่อนตัวในการปรับปรุงกรณี
4. ความอ่อนตัวในการประเมินผล และแก้ไข
5. ความสามารถในการรวมสถานการณ์ใหม่ๆ ในหน่วยความจำอย่างเหมาะสม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอ้างเหตุผลที่มีประสบการณ์น้อย (มีจำนวนกรณีน้อย) จะมีประสบการณ์ในการทำงานน้อยกว่าตัวที่มีประสบการณ์มาก แต่ไม่จำเป็นว่าจะต้องให้คำตอบที่แย่กว่า ถ้า ตัวอ้างเหตุผล มีความสร้างสรรค์ในการเข้าใจ และการปรับปรุงเพียงพอ และมีประสบการณ์ที่จำเป็นอยู่บ้าง โดยการอ้างกรณีของ ตัวอ้างเหตุผล จะต้องเริ่มต้นด้วยการมีกรณีอยู่จำนวนหนึ่ง ซึ่งจะต้องครอบคลุมถึงเป้าหมาย และ เป้าหมายย่อย ทั้งที่สำเร็จและพลาด โดยประสบการณ์ที่สำเร็จจะใช้ในการหาผลสรุปสำหรับปัญหาใหม่ๆ ข้อผิดพลาดจะใช้สำหรับเตือนในกรณีที่อาจมีความผิดพลาดที่อาจเกิดขึ้น

ความสามารถในการเข้าใจปัญหาใหม่ในรูปแบบของปัญหาเก่าๆ จะแบ่งออกเป็น 2 ส่วนคือ การเรียกซ้ำ (recall) ประสบการณ์เก่า และ การตีความสถานการณ์ใหม่ในรูปแบบของปัญหาเก่าที่ดึงขึ้นมา การเรียกซ้ำ จะเรียกว่า ดัชนีของปัญหา (Indexing Problem) มองในมุมกว้างๆ จะหมายถึงการหาประสบการณ์ในหน่วยความจำ ที่มีความใกล้เคียงกับสถานการณ์ใหม่ๆ ถ้าจะกล่าวถึงความหมายเฉพาะเจาะจง จะหมายถึงการกำหนดดัชนีสำหรับประสบการณ์ในหน่วยความจำซึ่งจะถูก เรียกซ้ำมาในสถานการณ์ที่เหมาะสม

การตีความเป็นกระบวนการในการเปรียบเทียบสถานการณ์ใหม่กับประสบการณ์ที่เรียกซ้ำขึ้นมา โดยจะเปรียบเทียบและหาความแตกต่างระหว่างสถานการณ์ของปัญหาเก่ากับปัญหาใหม่ โดยจะได้ผลลัพธ์คือ ผลจากการตีความสถานการณ์ใหม่ รวมกับความรู้ที่อ้างถึงในสถานการณ์ใหม่ หรือประเภทของสถานการณ์ ซึ่งการตีความจะมีความสำคัญมาก สำหรับปัญหาที่ไม่มีความเข้าใจเป็นอย่างดี

การปรับปรุง (Adaptation) เป็นกระบวนการในการแก้ไขผลลัพธ์เก่าๆ เพื่อให้ตรงตามความต้องการปัญหาใหม่ ซึ่งอาจจะใช้วิธีเพิ่มบางสิ่งลงในผลลัพธ์ หรือลบ หรือแทนด้วยตัวอื่น ซึ่งคำตอบที่สร้างสรรค์ จะมาจากการดัดแปลงการปรับปรุงไปในทางใหม่ๆ

จุดเด่นที่สำคัญของการอ้างกรณีของตัวอ้างเหตุผล คือความสามารถในการเรียนรู้ด้วยประสบการณ์ ซึ่งต้องการควบคุมย้อนกลับ เพื่อวิเคราะห์ว่าสิ่งใดทำแล้วถูกหรือผิด สำหรับผลลัพธ์ของมัน ซึ่งถ้าไม่มีการควบคุมย้อนกลบ อาจทำให้ ตัวอ้างเหตุผล ทำงานได้เร็วขึ้น แต่ก็จะผิดพลาดซ้ำที่เดิมทุกครั้ง และไม่สามารถที่จะเพิ่มประสิทธิภาพได้ ดังนั้นการประเมินผลและปรับปรุง จึงเป็นส่วนที่สำคัญของการอ้างกรณีของตัวอ้างเหตุผล

ประสิทธิภาพของการอ้างกรณีของตัวอ้างเหตุผล สามารถเพิ่มได้ใน 2 ทาง ทางแรกคือ มีประสิทธิภาพในการแก้ปัญหามากขึ้น ซึ่งสามารถหาคำตอบของปัญหาได้รวดเร็ว โดยจดจำวิธีเก่าไว้ ไม่ต้องย้อนคิดและปรับปรุงวิธีใหม่ อีกทางหนึ่งคือ เพิ่มความแม่นยำให้แก่คำตอบ โดยมีบทเรียนซึ่งจะช่วยหลีกเลี่ยงข้อผิดพลาดที่เคยเกิดขึ้นแล้ว

กรณีใหม่ๆ จะให้สิ่งแวดล้อม (Context) แก่ ตัวอ้างเหตุผล เพื่อแก้ปัญหา และ ประเมินสถานการณ์ ตัวอ้างเหตุผล ซึ่งกรณีครอบคลุมในขอบเขต (Domain) ที่กว้างกว่าจะดีกว่า ตัวซึ่งครอบคลุม และ ตัวอ้างเหตุผล ซึ่งครอบคลุมตัวอย่างของข้อผิดพลาดมากพอๆ กับผลสำเร็จ จะดีกว่าตัวที่ครอบคลุมแต่ผลสำเร็จ ดัชนีใหม่ๆ จะทำให้ ตัวอ้างเหตุผล ปรับแต่งกลไกการ เรียกซ้ำ ของมัน ดังนั้นมันจะจดจำ กรณี ได้ที่เวลาที่เหมาะสมขึ้น

สัญชาตญาณของ Case-based Reasoning จะรู้ว่าเหตุการณ์สามารถเกิดซ้ำๆ กันได้เป็นเรื่องปกติ สิ่งที่เคยทำมาแล้วในสถานการณ์หนึ่ง สามารถดัดแปลงมาใช้ได้ในสถานการณ์ที่คล้ายๆ กัน ซึ่งถ้าเรารู้ว่าวิธีไหนใช้ได้กับสถานการณ์ที่คล้ายๆ กัน เราก็จะใช้วิธีนี้เป็นจุดเริ่มต้นในการให้เหตุผลในสถานการณ์ใหม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งต่างกันอย่างชัดเจนกับวิธีการทางปัญญาประดิษฐ์ และนักจิตวิทยา ที่ได้รับการถ่ายทอดกันมาให้มองความรู้(Knowledge) ในทางปัญญาประดิษฐ์จะมองว่า ความรู้ จะต้องนำมาแสดงในรูปที่ธรรมดาที่สุด เพื่อให้สามารถประยุกต์ใช้ได้กว้างขวาง รูปแบบ(Model) จะอธิบายการทำงานของชิ้นส่วน อุปกรณ์ เช่น เราอาจจะประยุกต์รูปแบบของหัวใจกับคนไข้หลายๆ ราย จุดประสงค์ของการทำรูปแบบ ก็เพื่อค้นหาว่าสิ่งใดที่เป็นสิ่งปกติสำหรับวัตถุหลายๆ แบบ ถึงแม้ว่ากฎบางอย่างจะเฉพาะเจาะจง แต่วัตถุประสงค์ คือจะต้องสร้างกฎซึ่งสามารถประยุกต์ใช้ได้อย่างเรียบง่าย ดังนั้น จึงมีคำถามว่าทำไมเราจึงยังต้องการที่จะสร้างความรู้ที่เฉพาะเจาะจงกับสถานการณ์

ความรู้ในรูปทั่วไป จะมีประโยชน์ในทางด้านประหยัดเนื้อที่ในการเก็บความรู้ เพราะสามารถแทนการทำงานของสิ่งใดๆ ด้วยรูปแบบอย่างง่ายได้ นอกจากนั้นยังสามารถทำงานกับ ความไม่แน่นอน (Uncertainty) ได้ ทำให้เราสามารถให้เหตุผลได้โดยที่ไม่ต้องรู้รายละเอียดทั้งหมด

แต่ ความรู้โดยทั่วไป(General Knowledge) ก็มีข้อเสีย ข้อที่สำคัญคือ ขาด โอเปอเรชันนอไลเซชัน (Operationalization) นั่นคือ ความรู้ในรูปทั่วไปบางครั้งจะอยู่ในรูปนามธรรมเกินกว่าจะทำให้เข้าใจ และนำมาปฏิบัติได้ ข้อเสียอีกข้อคือ เพราะตัวมันเองอธิบายความรู้ในรูปทั่วไป ดังนั้นจึงไม่สามารถทำงานกับเหตุการณ์ที่ผิดคาด หรือผิดปกติได้

ปัญหาจะยิ่งเลวร้าย ยิ่งความรู้ขาดความสมบูรณ์ ซึ่ง กรณี สามารถครอบคลุมสถานการณ์เหล่านี้ได้ โดยการอ้างเหตุผลของตัวอ้างเหตุผล เหมาะเป็นอย่างยิ่งที่จะใช้ทำงานกับ ความรู้ไม่สมบูรณ์(Incomplete Knowledge)

กรณี จะแทนความรู้ในระดับปฏิบัติการ(Operational Level) โดยจะแยกสัดส่วนของการใช้ความรู้ในการแก้ปัญหาและวิธีการในการแก้ปัญหาออกจากกัน นอกจากนั้นยังช่วยในกรณีที่ไม่สามารถสร้าง รูปแบบทั่วไป(General Model) ได้ ซึ่งสามารถให้เหตุผลได้ โดยใช้ความรู้ที่เฉพาะเจาะจง ข้อดีอีกอย่างคือช่วยเชื่อมความรู้หลายๆ ส่วนเข้าด้วยกัน ซึ่งช่วยป้องกันปัญหาการเข้ากันไม่ได้ของ สิ่งแวดล้อม โดย กรณี จะประกอบความรู้ให้เอง

ในระบบที่ใช้งานจริง จะไม่ใช้แค่เพียงกรณีเท่านั้น แต่จะอ้างถึงความรู้ในรูปทั่วไปด้วย โดยสิ่งที่แตกต่างจากธรรมดาจะถูกเก็บเป็นกรณี ซึ่งความรู้ที่แสดงความธรรมดา อาจเปลี่ยนแปลงได้ตามเวลา ซึ่งจะพบว่าถ้าเราเก็บประสบการณ์ทุกอย่างที่เกิดขึ้น ก็จะทำให้เกิดกรณีมากมาย ดังนั้นทุกครั้งที่เราพบประสบการณ์ใหม่ เราจะพิจารณาและเก็บเฉพาะ กรณีซึ่งมีความแตกต่างกับ กรณี เดิมที่มีอยู่แล้ว และความแตกต่างนั้นสร้างความยุ่งยากในการหาเหตุผล นอกนั้นถึงแม้เราจะเก็บไว้ แต่ก็ไม่ได้เพิ่มประสิทธิภาพในการหาเหตุผล ซึ่งต้องเปลืองความจำในการเก็บข้อมูลเพิ่มขึ้น

เป็นการยากที่จะพิจารณาว่า กรณี ใดควรจะเก็บหรือไม่ หลักที่ควรจะพิจารณาคือ "ถ้ามีความแตกต่างที่สอนบทเรียนที่สามารถใช้ได้ในอนาคต ซึ่งไม่สามารถอนุมานได้โดยง่าย จาก กรณี ที่มีอยู่แล้ว ก็ควรจะเก็บ กรณี เพิ่มเข้าไป" นอกจากนี้ไปความแตกต่างที่เกิดขึ้น จะต่างจากเหตุการณ์ปกติเพียงเล็กน้อยเท่านั้น

เมื่อเวลาผ่านไป ก็ทำให้เกิดความเปลี่ยนแปลง การกระทำบางอย่างในแนวธรรมดาก็ถือเป็นบทเรียน ถ้ามันไม่ถูกเรียนรู้มาก่อน เมื่อมันเกิดขึ้นซ้ำๆ กันโดยไม่มีข้อมูลอื่นเพิ่มเติม ก็ทำให้ไม่มีการเก็บ กรณี ใหม่ กรณี จึงไม่เพียงแสดงถึงข้อแตกต่างเท่านั้น แต่ยังแสดงถึงแนวทางปกติ ซึ่งต่างจากความรู้ที่มีมาก่อน การเก็บ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กรณี จำทำให้กรณีที่ความแตกต่างนั้นทำให้เกิดบทเรียน ซึ่งไม่สามารถอนุมานได้โดยง่าย เป็นวิธีที่แสดงถึงความรู้ทั่งที่เป็นความรู้ทั่วไปและเฉพาะเจาะจง

เราจึงต้องพิจารณาว่า กรณี ประเภทไหนที่ควรจะนำมาสอน แนวทางที่ดีที่สุดคือ เก็บเฉพาะ กรณี ที่สามารถนำไปสู่เป้าหมายได้ สำหรับมนุษย์ซึ่งมีเป้าหมายเป็นจำนวนมาก ความแตกต่างเพียงเล็กน้อย ก็อาจสร้างบทเรียนที่นำไปสู่เป้าหมายหนึ่งได้ ในมุมมองของแบบจำลองทางความรู้(Cognitive Model) หรือแบบจำลองทางความคิด เป็นการยากที่จะคาดเดาอย่างแน่นอนว่า กรณี ใดควรจะเก็บไว้ แต่สำหรับเครื่องจักรจริงจะค่อนข้างง่ายกว่า เพราะมีเป้าหมายจำนวนจำกัด บางส่วนเป็นการกระทำ บางส่วนเป็นการให้เหตุผล

พอจะสรุปได้ว่ากรณี คือ ส่วนของความรู้ที่มีขอบเขตแสดงถึงประสบการณ์ซึ่งสอนหลักการบทเรียนที่สามารถนำไปสู่เป้าหมายของ ตัวอ้างเหตุผล ซึ่ง กรณี สามารถแบ่งได้ 2 ส่วน คือ บทเรียนที่ กรณี สอน และ สิ่งแวดล้อม ที่บทเรียนนั้นใช้ได้ ซึ่ง สิ่งแวดล้อม นี้เรียกว่า ดัชนี ของ กรณี

เปรียบเทียบการอ้างเหตุผลด้วยกรณีกับการอ้างเหตุผลโดยใช้รูปแบบ(Model-based Reasoning) จะเห็นว่า การอ้างโดยใช้รูปแบบจะใช้ได้ดีกับกรณีปกติธรรมดาเท่านั้น และจะต้องใช้กับงานซึ่งเรารู้กระบวนการทำงานดีอยู่แล้ว แต่การอ้างเหตุผลด้วยกรณี สามารถแก้ปัญหาเหล่านี้ได้

3.2 การบวนการและการเกิดผล (Process and Issue)

การทำงานของการอ้างอิงกรณีของตัวอ้างเหตุผล พอจะแบ่งเป็นขั้นตอนได้ ดังนี้

1. การดึงกรณี (Case Retrieval) คือ การดึงกรณี เก่าๆ ขึ้นมาจากไลบรารีของกรณี สามารถแบ่งย่อยได้ 2 ขั้นตอน คือ

   - การเรียกกรณีเก่าซ้ำ(Recall previous case) เป็นการเรียกกรณีซ้ำ ซึ่งมีประโยชน์ในการทำงานในสถานการณ์ใหม่ โดยใช้ลักษณะของสถานการณ์ใหม่เป็นดัชนี ในการค้นหาไลบรารีของกรณี
   - การเลือกสับเซตที่ดีที่สุด(Select the best subset) เป็นการเลือกกรณี ที่มีความเหมาะสมในการใช้งานที่สุดจาก กรณี ที่โดนดึงมาใช้(Retrieve) ขึ้นมา ซึ่งอาจเลือก กรณี ที่ดีที่สุดเพียง กรณี เดียว หรือเลือกเป็นกลุ่มของ กรณี ก็ได้

ปัญหาหลักๆ ของขั้นตอนนี้คือ ปัญหาการจับคู่(Matching problem) นั่นคือ ทำอย่างไรจึงจะทราบว่ากรณี ใดเหมาะที่จะ เรียกซ้ำ ขึ้นมาใช้ ซึ่งบางครั้ง กรณี ที่นำมาเปรียบเทียบกัน ก็ไม่ได้มีความคล้ายคลึงกัน แต่กรณี นั้นมีประโยชน์ที่จะ เรียกซ้ำ ขึ้นมา ซึ่งในปัญหานี้เราจะใช้ดัชนีเป็นตัวแทนของ กรณี ในการเปรียบเทียบ

บางครั้งเรามีข้อมูลไม่พอสำหรับสถานการณ์ใหม่ ดังนั้นเพื่อความรัดกุมของระบบ จะต้องมีการเตรียมการปัญหา (Situation-assessment) ซึ่งในการพิจารณาสถานการณ์ต่างๆ จะต้องพิจารณาแบบอัตราส่วน (Derived feature) มากกว่าพิจารณาแบบค่าแท้จริง (Surface feature)

อีกประเด็นหนึ่งคือ วิธีการดึงมาใช้(Retrieval วิธีการ) ซึ่งก็คือการค้นหา กรณี ในไลบรารีของกรณีนั่นเอง ปัญหาอยู่ที่การจัดระบบการใช้ดัชนีของกรณี ซึ่งเราสามารถพัฒนาขีดความสามารถได้ด้วยการใช้ กลไกคู่ขนาน(Parallel machine) ซึ่งจะทำให้ปัญหาขอขวด (Bottleneck problem) ไปตกอยู่ที่ วิธีการจับคู่ (Matching วิธีการ) แทน

จากปัญหาทั้งหมดจะพบว่าเป็นปัญหาการทำดัชนี(Indexing problem) นั่นคือ จะทำอย่างไรให้ดึงกรณีเก่าๆ มาใช้ประโยชน์ได้ในเวลาที่เหมาะสม โดยการใช้ดัชนีเป็นตัวกำหนดเงื่อนไขที่กรณีเป็นประโยชน์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมก็จะใช้สถานการณ์ใหม่เป็นคีย์(Key) ในการค้นหาดัชนีที่ใกล้เคียงที่สุด ผู้ที่คิดวิธีการ(Algorithm) และ ฮิวริสติก(Heuristic) สำหรับดัชนีแบบอัตโนมัติ (automatic) จัดระบบของกรณีสำหรับดัชนีค้นหาหน่วยความจำโดยใช้ดัชนีเหล่านั้น และเลือก กรณี ที่ดีที่สุดที่เรียกซ้ำขึ้นมา

2. เสนอผลลัพธ์ขั้นแรก (Proposing a Ballpark Solution) เป็นการตัดบางส่วนของการแก้ไข (Solution) หรือ การแปล(Interpretion) ของกรณีที่เรียกซ้ำขึ้นมา

มีประเด็นที่สำคัญหลายประเด็นเกี่ยวข้องกับขั้นตอนนี้ เช่น ขอบเขตในการเลือกกรณีขึ้นมาควรเป็นอย่างไร ซึ่งก็ขึ้นอยู่กับ 2 ปัจจัยคือ เป้าหมายของตัวอ้างเหตุผล ที่ต้องการโฟกัส(Focus) และ โครงสร้างภายในของ กรณี

อีกประเด็นหนึ่งก็คือควรจะมีการปรับปรุง(Adjustment) เล็กน้อยก่อนจะส่งไปยังขั้นต่อไปหรือไม่ หรือที่เรียกว่าสามัญนึก(Commonsense) ซึ่งบางครั้งการปรับเปลี่ยนบางอย่างเล็กน้อยตามอากิวเมนท์ (Argument) ของสถานการณ์จะทำให้ขั้นตอนต่อไปง่ายยิ่งขึ้น

อีกประเด็นที่สำคัญสำหรับการตีความการอ้างเหตุผลด้วยกรณี คือ จุดเริ่มต้นของการใช้เหตุผล ดังนั้นกรณีแรกที่เลือกจึงเป็นสิ่งสำคัญ โดยเฉพาะทางเลือกไม่ได้ไปในทางเดียวกัน ทางเลือกหรือ กรณีที่เราเลือกมาใช้การตีความ เป็นอันดับแรกจะมีผลอย่างมาก

3. การปรับปรุง มี 2 ขั้นตอน คือ กำหนดส่วนที่ต้องมีการดัดแปลง และการดัดแปลง ในการกำหนดส่วนที่ต้องทำการดัดแปลงจะต้องพิจารณาถึงวิธีที่จะทำการดัดแปลง เช่นการลบ ส่วนประกอบ(Component) รองๆ คือลบส่วนที่ทำหน้าที่ที่ไม่จำเป็นทิ้งไป อีกสิ่งที่สำคัญในการพิจารณาส่วนที่ต้องทำการแก้ไข คือการพิจารณาความแตกต่างระหว่างกรณีเก่าๆ และกรณีใหม่

4. การประเมินการหาเหตุผล : การหาเหตุผลสนับสนุนและการวิจารณ์(Evaluative Reasoning : Justification and Criticism) เป็นการพิสูจน์ความถูกต้องของคำตอบหรือคำแปล ที่ได้จาก ตัวอ้างเหตุผลก่อนนำไปใช้งานจริง แต่บางครั้งก็มีบางสิ่งซึ่งจำเป็นสำหรับการตรวจสอบข้อผิดพลาด ซึ่งเป็นสิ่งที่เราไม่ทราบ เราจึงจำเป็นต้องใช้วิธีในการให้เหตุผลโดยการเปรียบเทียบและหาข้อแตกต่างระหว่างคำตอบที่ได้กับคำตอบอื่นๆ ซึ่งต้องใช้รีเคอร์สีบ(Recursive) ในการดึงกรณีมาใช้ ซึ่งมีคำตอบเหมือนกัน ทางเลือกอื่นอาจเป็นการจำลอง(Simulation) เพื่อตรวจสอบคำตอบ ซึ่งหลังจากตรวจสอบอาจจะต้องมีการแก้ไขในคำตอบเรียกว่า ซ่อมแซม(Repair)

ประเด็นหลักอยู่ที่ วิธีการประเมินผลโดยใช้กรณี วิธีการดึงกรณีมาใช้เพื่อตีความ ประเมินผล และพิสูจน์เหตุผล การสร้างสถานการณ์ทดสอบและการใช้

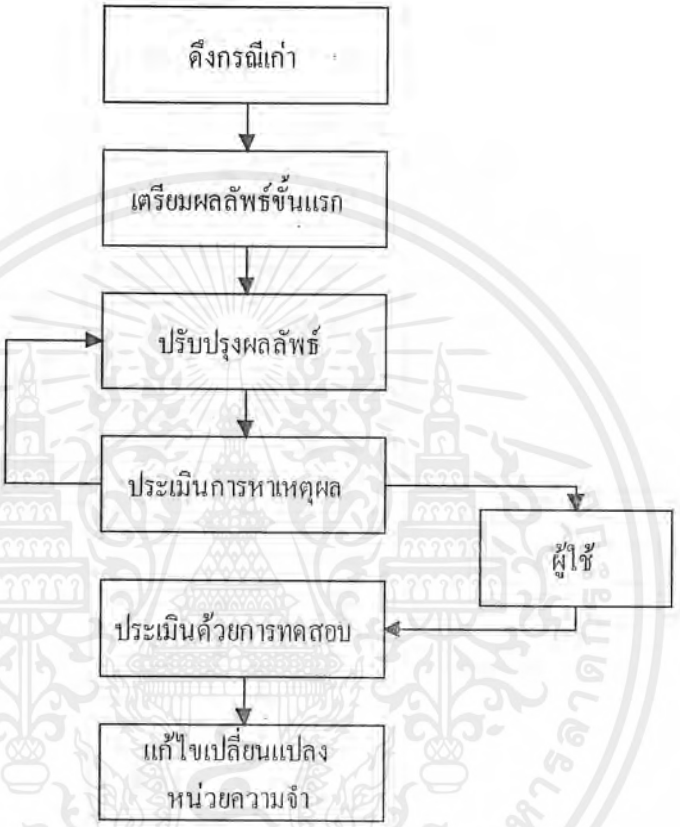
5. การประเมินการทดสอบ(Evaluative Testing) เป็นการนำการควบคุมย้อนกลับจากการใช้งานจริงมาประเมินผลคำตอบ ถ้าคำตอบใช้งานได้ตามต้องการก็ไม่ต้องมีการวิเคราะห์คำตอบ แต่ถ้าเกิดความแตกต่างก็จะต้องมีการวิเคราะห์ว่าอะไรคือสาเหตุของความแตกต่าง

ขั้นตอนนี้เป็นขั้นตอนที่สำคัญที่สุดของการอ้างอิงของตัวอ้างเหตุผลเพราะจะทำให้สามารถหลีกเลี่ยงข้อผิดพลาดที่อาจเกิดขึ้นซ้ำในการทำงานครั้งต่อไปได้ บางครั้งการประเมินผลก็อาจจะทำโดยใช้ สิ่งแวดล้อมของกรณีก่อนหน้าหรือใช้การควบคุมย้อนกลับจากการใช้งานจริง หรือแม้แต่จากการจำลองการประเมินผลจะต้องอธิบายความแตกต่าง หาเหตุผลของความแตกต่าง คาดคะเนผลที่เกิดขึ้น เปรียบเทียบและลำดับการเลือกที่จะเป็นไปได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. การเปลี่ยนแปลงแก้ไขหน่วยความจำ(Memory Update) เป็นการเก็บกรณีใหม่ๆ ในไลบรารีของกรณีซึ่งประกอบด้วยปัญหา คำตอบ ข้อเท็จจริง และเหตุผลที่เกี่ยวข้อง ซึ่งในการเปลี่ยนแปลงแก้ไข จะต้องใช้กระบวนการที่สำคัญคือใช้ดัชนีโดยการกำหนดดัชนีที่เหมาะสมให้กับ กรณี ใหม่และต้องแน่ใจว่า กรณี อื่นๆ ยังคงใช้การได้

กระบวนการของการอ้างเหตุผลเชิงกรณี สรุปได้ดังรูปที่ 3.1



รูปที่ 3.1 กระบวนการของการอ้างเหตุผลเชิงกรณี

ประโยชน์ของการอ้างเหตุผลด้วยกรณี

1. การอ้างเหตุผลด้วยกรณีมี ตัวอ้างเหตุผล ที่สามารถให้คำตอบได้อย่างรวดเร็ว ไม่ต้องเสียเวลาในการค้นหาคำตอบโดยการเริ่มต้นใหม่ทุกครั้ง

2. การอ้างเหตุผลด้วยกรณีมี ตัวอ้างเหตุผล ที่สามารถให้คำตอบในขอบเขตความรู้ที่เราอาจจะไม่เข้าใจโดยสมบูรณ์ได้

3. การอ้างเหตุผลด้วยกรณีมี ตัวอ้างเหตุผล ที่สามารถประเมินคำตอบได้ เมื่อไม่สามารถประเมินคำตอบได้ ก็สามารถใช้ สิ่งแวดล้อม จากสถานการณ์เก่ามาประเมินผลได้

4. กรณี เป็นประโยชน์ในการตีความหลักการ ซึ่งอาจจะไม่เข้าใจอย่างถ่องแท้ได้

5. การจำกรณี ได้จะช่วยในการเตือนการเกิดปัญหา ซึ่งอาจเกิดขึ้นซ้ำได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. กรณี จะช่วยให้ ตัวอ้างเหตุผล เจาะเข้าไปยังส่วนสำคัญของปัญหาได้ โดยชี้ไปยังจุดที่สำคัญของปัญหา

ข้อเสียของการอ้างเหตุผลด้วยกรณี

1. การอ้างกรณีของตัวอ้างเหตุผล อาจจะพยายาม กรณี เก่าๆ มากจนเกินไปจน ละเลยสถานการณ์ปัจจุบัน
2. การอ้างกรณีของตัวอ้างเหตุผล อาจจะพยายามปรับเปลี่ยนตัวแปรบางชนิดมากจนเกินจำเป็นในการแก้ปัญหา (He-She-It)
3. คนทั่วไปมักจะละเลย ไม่ถูกเตือนถึงกลุ่มของ กรณี ที่สำคัญและเหมาะสมขณะทำการหาเหตุผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
กรณี (Case)

4.1 การแทนดัชนี(Representing Case)

กรณี คือ อะไร

- กรณีเป็นการแทนความรู้เฉพาะที่เชื่อมโยงกับสิ่งแวดล้อมซึ่งจะบันทึกความรู้ลงในระดับการปฏิบัติการ
- กรณีสามารถได้มาในรูปร่างและขนาดที่แตกต่างกัน (อาจจะพบบ่อยครั้งหรืออาจพบในบางเหตุการณ์)
- กรณีเป็นการบันทึกประสบการณ์ซึ่งจะแตกต่างจากการคาดการณ์ กรณีที่มีการบันทึกควรเป็นบทเรียนที่มีประโยชน์
- บทเรียนที่มีประโยชน์ มีลักษณะคือ สามารถช่วยให้บรรลุเป้าหมาย หรือ ส่วนต่างๆของเป้าหมายในอนาคต หรือ เป็นการเตือนถึงความเป็นไปได้ถึงความล้มเหลว หรือ มองไปที่ปัญหาที่ยังไม่เกิดขึ้น

อาจกล่าวได้ว่า กรณี คือ ส่วนต่างๆของสิ่งแวดล้อมของประสบการณ์ที่นำมาแทนความรู้ ซึ่งสอนถึงบทเรียนพื้นฐาน เพื่อให้บรรลุเป้าหมายของผู้มีเหตุผล

กรณีสามารถแบ่งเป็นสองส่วนใหญ่ๆ

1. บทเรียน (Lesson) เป็นส่วนที่กระจายความรู้
2. สิ่งแวดล้อม (Context) เป็นส่วนที่สามารถทำให้กระจายความรู้ได้
3. ประเด็นที่สัมพันธ์กับการแทนความรู้ของกรณีที่มีอยู่ทั้งหมด จะเน้นไปที่ประเด็นต่อไปนี้
4. อะไร คือ ส่วนประกอบของกรณี
5. ความรู้ประเภทใดที่กรณีต้องการการเข้าถึง
6. อะไร คือ รูปแบบและวิธีการที่เหมาะสมสำหรับการแทนในแต่ละกรณี
7. ทำอย่างไร เราจึงสามารถรู้ขอบเขตของกรณี และ ทำอย่างไร จึงสามารถแบ่งขอบเขตของกรณีเป็นกลุ่มๆ ที่มีขนาดพอเหมาะได้

เราจะให้ความสำคัญกับสารบัญของกรณี ซึ่งจะต้องรวมความรู้เข้าไปด้วย เพื่อจะได้ใช้ประโยชน์ในการเรียนรู้

4.2 ส่วนประกอบของกรณี

โดยทั่วไปเรามักจะมองเห็นถึงส่วนอธิบายปัญหา และการแก้ปัญหา แต่ในบางขอบเขตของปัญหาซึ่งเราไม่รู้และไม่สามารถคาดการณ์ได้ ดังนั้น ผลลัพธ์จึงมีบทบาทควบคู่ไปกับส่วนอธิบายปัญหาและการแก้ปัญหา เช่น ผลลัพธ์เป็นเช่นไร หรือจะประสบความสำเร็จหรือไม่

ดังนั้นส่วนประกอบหลักของการแทนปัญหา จะมีดังนี้

1. เป้าหมายที่จะได้รับในการแก้ปัญหา จะหมายถึง จุดประสงค์ของผู้กระทำในสถานะการณ์หนึ่งๆ โดยทั่วไปตัวอ้างเหตุผลจะพยายามให้ถึงเป้าหมาย หรือบรรลุจุดประสงค์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ขอบเขตของเป้าหมาย โดยขอบเขตจะเป็นเงื่อนไขเพื่อมุ่งสู่เป้าหมาย
3. ลักษณะสถานะการณ์ของปัญหา และความสัมพันธ์ในแต่ละส่วน โดยลักษณะสถานะการณ์ของปัญหา ซึ่งจะเก็บข้อมูลเกี่ยวกับความสัมพันธ์ และสถานะการณ์ เพื่อให้ได้สถานะการณ์ของเป้าหมาย

โดยทั่วไปจะมีแนวทาง 2 แบบซึ่งควรจะนำมาร่วมในการตัดสินใจ ซึ่งเป็นข้อมูลรายละเอียดของปัญหา

1. รายละเอียดของปัญหา รวมเอาข้อมูลรายละเอียดทั้งหมดที่แน่นอนนำมาใช้ เพื่อให้ได้เป้าหมาย
2. รายละเอียดของปัญหา รวมเอาประเภทของข้อมูลที่ใช้โดยทั่วไป เพื่อให้เห็นประเด็นว่าอยู่ในกรณีไหน

4.3 สารบัญของการแก้ปัญหา

มีการแก้ปัญหาหลายประเภท ซึ่งการแก้ปัญหาก็คือการวางแผนที่จะกระทำ และมนุษย์ก็เป็นผู้ออกแบบ และการแก้ปัญหาแต่ละประเภทก็ขึ้นอยู่กับปัญหานั่นเอง

ในที่นี้ ผู้แก้ปัญหาจะได้รับกรณีมาเพื่อนำไปใช้ แล้วจะได้การแก้ปัญหาใหม่ แต่การแก้ปัญหามีส่วนประกอบหลายส่วน ซึ่งเปลี่ยนแปลงตามจุดมุ่งหมาย โดยอาจกำหนดหัวข้อต่างๆดังนี้

- ตัวปัญหาที่จะแก้
- กลุ่มของลำดับการหาเหตุผลที่เคยแก้ปัญหา
- กลุ่มของเหตุผลสนับสนุนสำหรับการตัดสินใจซึ่งถูกทำขึ้นในการแก้ปัญหา
- การแก้ปัญหาที่ยอมรับได้ ซึ่งยังไม่ได้มีการเลือก
- การแก้ปัญหาที่ไม่สามารถยอมรับ ซึ่งอยู่นอกเหนือบทบาทที่มี
- การคาดเดาผลลัพธ์ที่จะได้ ซึ่งขึ้นอยู่กับการเปลี่ยนแปลงของการแก้ปัญหา

4.4 ลำดับการหาเหตุผล

โดยทั่วไปผู้สร้างระบบจะบันทึกกลุ่มของลำดับการหาเหตุผลที่เคยได้รับจากการแก้ปัญหาในการแทนกรณี ซึ่งมีการยอมรับลำดับการหาเหตุผลนำมาใช้อีกครั้ง เมื่อเกิดการแก้ปัญหา เช่น การเรียนตอนยังเด็ก อาจารย์ได้แก้ปัญหาซึ่งมีลำดับขั้นตอน และได้ให้โจทย์ เราก็ใช้ลำดับขั้นตอนนั้นๆ ในการแก้ปัญหา

4.5 การหาเหตุผลสนับสนุน

มีไว้เพื่อเตรียมทางเลือก ของแนวทางการเปลี่ยนแปลงของวิธีการแก้ปัญหาที่เคยผ่านมา โดยการหาเหตุผลสนับสนุนรองรับจุดประสงค์ 2 อย่างดังนี้

1. เพื่อเตรียมแนวทางระหว่างประเมินค่าของการแก้ปัญหา ถ้ามีส่วนของการแก้ปัญหาก่อนหน้านี้ได้ถูกใช้สนับสนุนทางเหตุผล แล้วก็สามารถประเมินได้ ว่าผลของวิธีการแก้ปัญหาครั้งก่อน จะนำมาใช้ในการแก้ปัญหาครั้งต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การหาเหตุผลสนับสนุน ยังช่วยแนะนำในการทำดัชนี ดัชนีจะเป็นตัวกำหนดว่าส่วนไหนสำคัญต่อกรณี และส่วนไหนสำคัญต่อการแก้ปัญหา

การหาเหตุผลสนับสนุน และลำดับการหาเหตุผลมีความสัมพันธ์กัน เพราะการหาเหตุผลสนับสนุนจะเป็นส่วนที่เตรียมทั้งการแก้ปัญหา และหาเหตุผลในการแก้ปัญหา

ในส่วนของการทำโปรแกรมจะเก็บส่วนนี้ไว้ในการรักษาเหตุผล

- ทางเลือกของการแก้ปัญหา หลังจากได้ข้อมูลที่มีประโยชน์ ก็นำมาใช้เป็นทางเลือก เพื่อสามารถนำไปใช้
- การคาดการณ์ถึงผลลัพธ์ เมื่อได้รับความรู้ที่เป็นประโยชน์ที่สามารถนำมารวมในการแก้ปัญหา เมื่อมีการแทนความรู้นั้นก็จะได้ผลลัพธ์ (โดยการคาดเดา)

สารบัญของผลลัพธ์ของกรณี

ผลลัพธ์ของกรณีจะกำหนดถึงผลที่เกิดขึ้นจากการแก้ปัญหา หรือประสิทธิภาพในการแก้ปัญหา ข้อมูลของผลลัพธ์จะรวมเอาการตอบสนอง จากความเป็นจริงว่า เกิดอะไรขึ้นบ้างเมื่อได้ผลลัพธ์ หรือขยายแนวทางการแก้ปัญหา ซึ่งจะนำไปตรวจสอบ ถึงความสำเร็จจริง หรืออาจเป็นการคาดการณ์ ผลลัพธ์ควรมีส่วนต่างๆดังนี้

- ตัวผลลัพธ์
- ผลลัพธ์นั้นเกิดขึ้นจริง หรือเป็นการคาดการณ์
- ผลลัพธ์นั้นประสบความสำเร็จ หรือล้มเหลว
- มีการวางแผนแก้ไข
- แนวทางการหลีกเลี่ยงปัญหา
- ส่วนชี้ถึงส่วนที่แก้ปัญหา

การพิจารณากรณี

เมื่อเข้าใจถึงประโยชน์ของการแทนกรณีแล้ว ก็จะต้องหาวิธีแทนความรู้ทั้งหมดเข้าไปในกรณี เช่น การใช้วิธีเฟรม(Frame),ซีเมนติก(semantic),เครือข่าย(network),พรีดิกเคต(predicate) เพื่อให้คอมพิวเตอร์เข้าใจ

4.6 การทำดัชนีคำศัพท์

จะเป็นการทำให้คอมพิวเตอร์เข้าใจตรงกับมนุษย์ อาจมีการเปลี่ยนแปลงข้อมูลจากลักษณะของข้อความเป็นรหัส โดยจะมีส่วนที่เกี่ยวข้องหลายส่วน

- ส่วนที่เมื่อเกิดปัญหาแล้วอาจจะมีชื่อเรียกไม่เหมือนกันในบางครั้งดังนั้นต้องมีการทำไลบรารีของกรณีเพื่อให้แน่ใจว่าความหมายตรงกัน
- ส่วนช่วยในการค้นหา เหมือนกับดัชนีท้ายเล่มของหนังสือทั่วไป

ดังนั้นดัชนีควรมีส่วนต่างๆดังนี้

- ดัชนีต้องมีการเดาคำศัพท์ที่ได้รับมา
- ดัชนีต้องใช้อธิบายความหมาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ดัชนีต้องเดาคำศัพท์ที่แวดล้อมอยู่

4.7 คุณสมบัติที่ดีของดัชนี

ดัชนีที่ดีสามารถนำมาช่วยในการทำนายให้เกิดประโยชน์ ดัชนีที่ดีแม้ว่าจะเป็นนามธรรม แต่ก็สามารถทำให้เป็นรูปธรรมพอที่จะจดจำได้ ดังนั้นคำศัพท์ที่ทำเป็นดัชนีจะต้องสื่อความหมายของตังเอง ดังนั้นคุณสมบัติที่ดีของดัชนีจะต้องมี

- จุดเด่นในการทำนาย โดยการทำนายจะหมายถึง พื้นฐานของกรณีที่อะบายปัญหา,การแก้ปัญหา แล้วได้ผลลัพธ์ของการแก้ปัญหา ซึ่งส่วนที่ต้องการ คือผลลัพธ์ ดังนั้นจุดเด่นในการทำนาย คือ การรวมเอารายละเอียดของกรณี ที่ตอบสนองต่อการแก้ปัญหา นำมารวมกันเพื่อแก้ปัญหา ซึ่งมีผลกับผลลัพธ์
- ความเป็นนามธรรมของดัชนี เพราะกรณีถูกกำหนดเฉพาะ ดัชนีที่นำมาใช้จะต้องครอบคลุมเหตุการณ์ได้อย่างเหมาะสม
- ความเป็นรูปธรรมของดัชนี ความเป็นนามธรรมของดัชนี จะทำให้มีขอบเขตที่กว้างเกินไป ทำให้อนุมานยาก ดัชนีจะต้องมีความสามารถในการนำไปใช้ได้
- ดัชนีที่มีประโยชน์ ดัชนีที่ถูกเลือกสามารถนำมาใช้ในการทำนาย และใช้ประโยชน์ในช่วงการหาเหตุผล และใช้เป็นแนวทางช่วยการตัดสินใจ โดยอาจแบ่งได้เป็น

1. การนำไปแก้ปัญหา
2. การคาดการณ์ปัญหาที่เป็นไปได้
3. อธิบายเหตุผลในการเกิดข้อผิดพลาด และล้มเหลว
4. สามารถกู้คืนจากเหตุผลที่ผิดพลาด และล้มเหลว
5. ประเมินข้อเสนอแนะในการแก้ปัญหา

4.8 การเลือกคำศัพท์

คำศัพท์ที่จะนำมาใช้แทนจะมี 2 ส่วน คือ ส่วนที่เป็นกลุ่มของมิติ (ขนาด) ที่คำศัพท์นั้นครอบคลุมอยู่ และกลุ่มที่เป็นสัญลักษณ์ที่แทนค่า ซึ่งครอบคลุมรายละเอียดของมิติ โดยอาจมองสล๊อตและส่วนที่เพิ่มเติมเมื่อใช้วิธีแทนแบบเฟรม หรือทำนายและสร้างข้อโต้แย้งโดยการคำนวณ แต่รายละเอียดก็ไม่ได้กำหนดเฉพาะเจาะจง ว่าขนาดเท่าไรที่จะใช้,ใช้สัญลักษณ์อะไรแทน หรือรายละเอียดเมื่อมีการแทนค่า

มีอยู่ 2 กลุ่มที่จะนำไปใช้ในการตรวจสอบ

1. การเข้าถึงฟังก์ชัน โดยการตรวจสอบถึงส่วนที่ได้มาจากกรณีและส่วนที่ต้องนำไปสนับสนุน
2. การทบทวน โดยอาศัยความชำนาญของผู้เชี่ยวชาญ

ผลที่ได้จากการวิเคราะห์

1. ขอบเขตของหน้าที่ของผู้สร้างที่ต้องตอบสนอง
2. ขอบเขตของกรณีที่มีอยู่ที่สนับสนุนงานนั้นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. อัตราและเส้นทางที่ระบบจะต้องขยายในอนาคต

การตรวจสอบการครอบคลุมของคำศัพท์

1. นำคำศัพท์ที่ได้ไปแทนในกรณี
2. ขอบเขตของหน้าที่ที่กรณีจะอ้างถึงในการปฏิบัติ

กระบวนการเลือกดัชนีคำศัพท์

กระบวนการแบบฟังก์ชัน โดยจะมุ่งเน้นไปที่จุดประสงค์ของดัชนี โดยจะมีกระบวนการดังนี้

1. รวบรวมกลุ่มที่จะนำไปแทนในกรณี นำกรณีทั้งหมดที่จะนำไปแทนปัญหา ซึ่งอยู่ในขอบเขต และตรงตามปัญหา เพื่อให้ได้แนวทางการแก้ปัญหา และผลลัพธ์
2. กำหนดจุดที่สนใจในกรณี ที่สามารถนำมากระทำ,ใช้เป็นบทเรียน หรือสอนได้
3. กำหนดลักษณะเหตุการณ์ ซึ่งแต่ละกรณี ทำได้เหมาะสมในจุดที่สนใจนั้นๆ และทำให้แน่ใจว่า ได้รวมเอาลักษณะต่างๆ นำไปหาเหตุผลเพื่อใช้ถึงเป้าหมาย
4. อธิบายดัชนีถึงความสามารถในการนำไปใช้ในแต่ละเหตุการณ์ และทำให้แน่ใจว่า เป็นนามธรรมพอที่จะครอบคลุมในส่วนที่นำไปใช้ และเป็นรูปธรรมพอที่จะจดจำ และเข้าใจ
5. เลือกกลุ่มของมิติ และสัญลักษณ์ซึ่งบอกถึงสารบัญของดัชนีที่จะถูกแทน

จุดที่สนใจ

โดยการวิเคราะห์ถึงจุดที่สนใจจะมี 4 ขั้นตอน คือ

1. เงื่อนไขในการประสบความสำเร็จ โดยให้ความสำคัญกับเป้าหมาย โดยพิจารณาว่าถ้าเป้าหมายสามารถประสบความสำเร็จในเหตุการณ์หนึ่งด้วยกระบวนการบางอย่าง กระบวนการนั้นน่าจะใช้ได้ผลกับเหตุการณ์ที่คล้ายกัน
2. เงื่อนไขในการประสบความล้มเหลว
3. การทำนายผล สามารถรายงานผลลัพธ์
4. อธิบายการเกิดผลลัพธ์

ขั้นตอนถัดมาจะดูความเหมาะสมของสถานะการณ์ ว่าสถานะการณ์นั้นควรมีลักษณะดังนี้

1. สนับสนุนการเข้าถึงเป้าหมาย
2. ต้องอยู่ในขอบเขตของงานนั้นๆ
3. รายละเอียดของเหตุการณ์จะต้องมีส่วนเชื่อมเป้าหมาย และหน้าที่อยู่

4.9 วิธีการเลือกดัชนี

การเลือกดัชนีนั้นจะนำมาใช้ เพื่อเลือกดัชนีที่อยู่ในรายการการตรวจสอบ เพื่อให้ผู้สร้างระบบรู้ถึงลักษณะต่างๆ ซึ่งจะนำไปถึงการทำนาย การแก้ปัญหา และผลลัพธ์ และมุ่งเน้นไปที่ความกรณีใหม่ และกรณีที่มีอยู่ในไลบรารีของกรณี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.9.1 การเลือกดัชนีโดยมนุษย์

จะเป็นวิธีที่กล่าวไปแล้วข้างต้นโดยการตรวจสอบถึงขอบเขตของดัชนีคำศัพท์ โดยมีลำดับขั้นตอนดังนี้

1. ตรวจสอบว่า กรณีจะนำมาใช้ประโยชน์อะไร โดยมองจุดที่สนใจกับงานที่จะให้ผลลัพธ์
2. ตรวจสอบว่าสภาวะแวดล้อมอะไรบ้าง ที่ชี้ให้เห็นถึงประโยชน์ของงาน
3. การแปลความให้สภาวะแวดล้อมจะเป็นที่เข้าใจและจดจำ และก่อให้เกิดการนำไปใช้ได้

4.9.2 การเลือกดัชนีโดยเครื่องจักร

บางครั้งมนุษย์ก็ไม่สามารถที่จะให้เหตุผล ในการตรวจสอบกรณีที่มีความซับซ้อนสูง โดยจะให้คอมพิวเตอร์ช่วย โดยใส่ความรู้ที่มี ให้แก่คอมพิวเตอร์เพื่อไปประมวลผล โดยจะต้องมีกระบวนการในการเลือกดังนี้

1. กระบวนการอ้างถึงรายการการตรวจสอบ ซึ่งจะขึ้นอยู่กับตารางของผู้สร้างที่ดึงเอาลักษณะเด่นมาใช้ประโยชน์
2. กระบวนการอ้างถึงส่วนอธิบาย จะเป็นส่วนที่ช่วยมนุษย์ในการเลือกดัชนี

กระบวนการอ้างถึงส่วนที่แตกต่าง โดยดัชนีจะถูกเลือกมาจาก 2 กระบวนการที่เฉพาะเจาะจงในไลบรารีของกรณี โดยหาจุดเด่นที่แตกต่างระหว่างกรณีทั่วไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
การดึงกรณีมาใช้ (Retrieving Case)

5.1 โครงสร้างองค์กรและวิธีการดึงมาใช้ (Organizational Structures and Retrieval Algorithms)

วิธีการในการสืบค้นในไลบรารีของกรณีจะขึ้นอยู่กับโครงสร้างองค์กรของกรณีเสมอ ดังนั้น จะไม่สามารถพิจารณาวิธีการดึงมาใช้ได้โดยไม่กล่าวถึงโครงสร้างองค์กรเลย

การดึงกรณีมาใช้จะต้องทำได้ในเวลาที่ไม่มากจนเกินไปจนไม่สามารถยอมรับได้ และการ จับคู่บางครั้งก็ไม่เป็นการจับคู่ ที่เหมือนกันโดยสมบูรณ์ แต่เป็นการหากรณีที่จับคู่เพียงบางส่วนที่ใกล้เคียงกัน (close partial match) ประเด็นสำคัญของการจับคู่ คือดัชนีจากที่ได้กล่าวมาแล้ว เป้าหมายของการดึงกรณีมาใช้ก็เพื่อเลือกกลุ่มกรณีที่มีจำนวนไม่มากที่มีประโยชน์ต่อสถานการณ์ใหม่นี้ และขณะเดียวกัน ก็เพื่อเป็นการรับประกันด้วยว่ามี กรณี อย่างน้อยกลุ่มหนึ่งที่มีประโยชน์ที่สุด จากกลุ่มที่เลือกขึ้นมา

การจับคู่จะต้องพิจารณาถึงหลักการ 2 ข้อที่สำคัญ คือ

1. มิติ (Dimension) หรือ ลักษณะ (Descriptor) ที่ใช้บ่งบอกความเหมือนหรือประโยชน์ของกรณี จะต้องระบุลำดับความสำคัญของแต่ละมิติ
2. ระดับของการเหมือนกันของแต่ละมิติ จะต้องบอกระดับของการเหมือนกัน เพราะการจับคู่มักจะเป็นความเหมือนกันเพียงบางส่วน(Partial match)

การพิจารณาว่า กรณี ใดจับคู่กับสถานการณ์ จะต้องพิจารณาจากทั้ง 2 หลักเกณฑ์ ดังนั้นตอนนี้ขอกลับเข้ามาในเรื่องโครงสร้างองค์กร และ วิธีการดึงมาใช้ต่อ แต่หลักการของการจับคู่ ตอนนี้จะเป็นประโยชน์ในการพิจารณาข้อดีข้อเสียของโครงสร้างองค์กรแต่ละแบบ ซึ่งจะได้กล่าวต่อไป

จากที่ได้กล่าวว่า โครงสร้างองค์กรและ วิธีการดึงมาใช้ไม่สามารถพิจารณาแยกกันได้ ต่อไปนี้จะกล่าวถึงทั้ง 2 หัวข้อรวมกัน โดยเสนอรูปแบบของโครงสร้างองค์กรแต่ละแบบ และ วิธีการดึงข้อมูลที่ใช้

1. แฟลทเมโมรี่(Flat Memory) , การค้นหาแบบอนุกรม(Serial Search) เป็นการเก็บ กรณี ในลักษณะของรายการ(List)ธรรมดา โดยสามารถการดึงมาใช้ โดยใช้ วิธีการจับคู่กับทุกกรณี ทีละกรณีแล้วส่งค่ากรณีที่ใกล้เคียงที่สุด

ข้อดีของวิธีนี้คือ การค้นหาจะเกิดขึ้นกับทั้งไลบรารีของกรณีความแม่นยำจึงขึ้นอยู่กับ วิธีการจับคู่(Matching function) ข้อดีอีกอย่างคือ การจัดองค์กรเป็นรูปแบบที่เรียบง่าย การเพิ่มกรณี ใหม่ๆ จึงไม่ต้องใช้วิธีการที่พิเศษ

ข้อเสียหลักคือ วิธีนี้ค่อนข้างจะสิ้นเปลือง เพราะยิ่ง ไลบรารีของกรณี มีขนาดใหญ่ ก็จะยิ่งต้องใช้เวลาในการดึงมาใช้มาก ซึ่งก็สามารถมีการปรับเปลี่ยนบางอย่างเพื่อแก้ไขข้อเสียข้อนี้ ดังนี้

- ชาโลวอินเด็กซิง(Shallow indexing) เหมือนกับการใช้ อินเวิร์ทเตดอินเด็ก(inverted index) ในระบบฐานข้อมูล เป็นการเลือกส่วนรายละเอียด(descriptor) เพื่อใช้เป็นดัชนี ในการพิจารณาแต่ละกรณี ซึ่งจะพิจารณาแต่ละกรณี เฉพาะ กรณี ที่ถูกชี้หรือดัชนีโดย ส่วนรายละเอียด จากสถานการณ์ใหม่เท่านั้น ดังนั้นแทนที่จะค้นกาทั้งไลบรารีของกรณีเราก็จะพิจารณาเฉพาะ กรณี ที่ควรค่าแก่การนำมาใช้กับวิธีการจับคู่เท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มี 2 สิ่งที่ควรพิจารณาสำหรับวิธีนี้ สิ่งแรกคือ มิติใดบ้างที่จะนำมารวมในกลุ่มของ ดัชนี มิติที่จะนำมารวมจะต้องมีประโยชน์ในการแยกแยะประโยชน์ของกรณี ซึ่งถ้าดัชนี เฉพาะเจาะจงเกินไป ก็อาจจะทำให้เราพลาด กรณี บางกลุ่มที่มีประโยชน์ แต่ถ้าดัชนี คลุมเครือเกินไป ก็อาจจะทำให้ไม่สามารถจำกัดจำนวนกรณี ที่เรียกซ้ำ ขึ้นมาได้ ซึ่งอาจจะเกินขีดจำกัดของระบบ

สิ่งที่สองที่จะต้องพิจารณาคือ ทำอย่างไรจะให้จับคู่บางส่วนเกิดขึ้นในทุกมิติ เพราะ กรณี ที่ดีที่สุดไม่ควรจะเป็น กรณี ที่จับคู่โดยตรงในบางมิติ แต่ควรเป็นจับคู่บางส่วนในทุกๆ มิติ ซึ่งอาจทำโดยดัชนีของกรณีด้วยทั้งส่วนลักษณะ(feature) ที่สังเกตได้ และ ส่วนลักษณะที่ทำให้สามารถตรวจสอบ ความเหมือนในลักษณะเพียงบางส่วนได้ ดังนั้นการเลือกมิติที่ใช้เป็นดัชนีจึงถือว่าสำคัญมาก

- การแบ่งส่วนของไลบรารีของกรณี(Partitioning of the case library) เป็นการแบ่งไลบรารีของกรณี ออกเป็นส่วน (Partition) เมื่อมีการดึงมาใช้ เราก็จะพิจารณาว่าสถานการณ์ใหม่นี้ตรงกับส่วนไหน การจับคู่ของกรณีก็จะทำกับเฉพาะส่วนนั้นเท่านั้น สำหรับวิธีนี้ ส่วนที่สำคัญที่สุดคือ การเลือกวิธีการแบ่งส่วน
- การดึงมาใช้แบบขนาน(Parallel retrieval) เป็นการค้นหาและจับแบบขนาน ซึ่งจำเป็นต้องใช้หน่วยประมวลผลหลายๆ ตัว ซึ่งถ้ายังไม่พอกับขนาดของไลบรารีของกรณี ก็อาจจะต้องใช้ชาโลวอินเด็กซิงหรือแบบแบ่งส่วนเข้ามาช่วยอีก

2. เครือข่ายการที่ใช้ลักษณะร่วมกัน(Shared-feature Network) , การค้นหาโดยใช้เส้นทางการกระจายก่อน(Breadth-first Graph Search) เป็นการจัดรวบรวมกรณี(Organized case) เป็นลำดับ(Hierarchy) โดยหลักการเบื้องต้นของการจัดองค์กรเป็นลำดับขั้นนั้นคือการแบ่ง กรณีออกเป็นส่วน(Cluster) แล้วแบ่งต่อเป็นส่วนย่อยๆ(Subcluster) ต่อไป แสดงได้เป็นลักษณะของลำดับขั้น

สำหรับเป็นการจัดกลุ่มกรณีโดยให้กรณี ที่มีลักษณะหลายๆตัว ร่วมกัน มาอยู่ในส่วนเดียวกัน ในแต่ละอินเทอร์นอลโหนด(Internal node) ในทรี(tree) จะเก็บลักษณะ ที่ถูกใช้ร่วมกันโดย กรณี ที่อยู่ใต้โหนด(node) นั้น ส่วนลีฟโหนด(Leaf node) จะเก็บตัว กรณี

การดึงกรณีมาใช้จากเครือข่ายที่ใช้ลักษณะร่วมกัน จะใช้วิธีการค้นหาโดยใช้เส้นทางการกระจาย ซึ่งในการค้นหา จะใช้วิธีการจับคู่ ในการเปรียบเทียบความใกล้เคียงระหว่างสถานการณ์ใหม่กับลักษณะของแต่ละโหนด ในระดับนั้นเพื่อเลือกโหนด ที่มีความใกล้เคียงกับสถานการณ์นั้นมากที่สุด

การสร้างและการเพิ่ม กรณี ใหม่ๆ ลงไปในเครือข่ายที่ใช้ลักษณะร่วมกัน ค่อนข้างจะง่ายเพราะใช้วิธีการทั่วไปสำหรับทรี สิ่งที่ต้องเพิ่มเติมลงไปคือ ความสามารถในการทำส่วน เพื่อแบ่งจัดกลุ่ม กรณี ที่มีลักษณะเหมือนกัน

ข้อดีหลักๆ ของเครือข่ายที่ใช้ลักษณะร่วมกัน และเป็นข้อดีของการจัดองค์กร แบบลำดับขั้น คือ การดึงกรณีมาใช้ ไม่ต้องทำการจับคู่ทั้งไลบรารีของกรณี ซึ่งทำให้ทำงานได้เร็วกว่าแบบเฟลทเมโมรีแต่ก็มีข้อเสียคือ การเพิ่มกรณีใหม่จำเป็นที่จะต้องมีการทำงานบางส่วนเพิ่ม เพราะต้องใส่ กรณี ให้ถูกตำแหน่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่างกับแฟลทเมโมรีที่สามารถเพิ่มกรณีได้โดยตรง ข้อเสียอีกข้อคือ เปลืองเนื้อที่ในการเก็บเป็นลำดับขั้น เพราะเนื้อที่ทั้งหมดที่ต้องใช้มากกว่าเนื้อที่จริงที่ใช้เก็บ กรณี เพราะต้องใช้ในการเก็บลักษณะ และลิงค์ (link) ระหว่างโหนด แต่บางครั้งก็คุ้มค่าเพราะพื้นที่หน่วยความจำ ปัจจุบันราคาไม่แพงมากนัก เมื่อเทียบกับความเร็วในการดึงมาใช้ที่เพิ่มขึ้น ข้อเสียอีกข้อที่สำคัญคือ บางครั้งเราอาจพลาดกรณีที่ดีที่สุดได้เพราะเราไม่ได้ใช้การจับคู่กับทั้งไลบรารีของกรณี

สิ่งที่ควรพิจารณาเพิ่มเติมคือการทำให้เครือข่ายหรือทรีได้ดี เพราะการทำให้เครือข่ายให้ดี ตลอดเวลาเป็นเรื่องที่สิ้นเปลือง ต้องใช้วิธีการที่ยุ่งยากในการจัดการ ดังนั้นบางครั้งก็ต้องมีการสร้างเครือข่ายใหม่ (Rebuilt) อีกสิ่งที่ควรพิจารณา คือ ลักษณะที่สำคัญของเครือข่าย ควรจะอยู่ด้านบน (Root) ของ เครือข่าย เพื่อให้เราสามารถ จับคู่ สถานการณ์กับ ลักษณะ ที่สำคัญก่อน ลักษณะ ที่สำคัญรองลงมา ดังนั้นจึงมีการสร้าง เครือข่าย โดยให้ ลักษณะ ที่สำคัญอยู่ด้านบนของ เครือข่าย เรียงตามความสำคัญ(Priority) เรียกว่า เครือข่ายที่ใช้ลักษณะสำคัญร่วมกัน(Prioritized Shared-feature Network) ซึ่งในบางระบบ การดึงมาใช้แต่ละครั้ง ตัวอ้างเหตุผล มีจุดประสงค์ต่างกัน ดังนั้นลักษณะหลักที่พิจารณาอาจจะต่างกัน ซึ่งอาจจะมีการสร้างเครือข่ายที่ใช้ลักษณะที่สำคัญร่วมกัน หลายชุด ซึ่งมีการจัดความสำคัญของ ลักษณะ ต่างๆ กัน เพื่อใช้กับการดึงมาใช้ที่ต่างกัน

3. เครือข่ายแบ่งแยกความสำคัญ(Prioritized Discrimination Network) , ใช้การสืบค้นโดยใช้เส้นทางการกระจายก่อนและจะมีความคล้ายคลึงกับ เครือข่ายที่ใช้ลักษณะร่วมกัน ต่างกันที่เครือข่ายที่ใช้ลักษณะร่วมกัน จะเน้นที่การรวมกลุ่มส่วนลักษณะ แต่เครือข่ายแบบนี้จะเน้นที่การแยกแยะกลุ่มของ กรณี

ในเครือข่ายแบบนี้แต่ละอินเทอร์นอลโหนด จะเป็นคำถามที่ใช้แบ่งกลุ่มของกรณี ซึ่งอยู่ใต้โหนดนั้นโหนดลูก(child node) แต่ละตัวจะเป็นคำตอบของคำถาม และที่ลีฟโหนดจะเป็นตัว กรณี

การดึงกรณีมาใช้ จะใช้การค้นหาโดยใช้ความลึก โดยเมื่อถึงแต่ละโหนด เราจะพิจารณาจากคำถามว่าสถานการณ์ใหม่ตรงกับคำตอบข้อใด แล้วลงไปยังระดับที่ต่ำลงมาวิธีการจับคู่ ไม่ต้องทำกับ โหนดลูก ของ Node คำถามเหมือนใน เครือข่ายที่ใช้ลักษณะร่วมกัน

การสร้าง และการเพิ่มกรณีในเครือข่ายแบบแยกนี้ ก็ไม่มีความยุ่งยากมาก ใช้วิธีการ สำหรับทรีแล้วเพิ่มส่วนในการตั้งคำถามเพื่อแยกแยะกลุ่มของกรณีคล้ายๆ กับการสร้างและเพิ่ม กรณีในเครือข่ายที่ใช้ลักษณะร่วมกัน แต่ถ้าเป็นเครือข่ายแบบแยก สิ่งที่จะต้องพิจารณาเพิ่มคือการเลือกคำถามที่มีความสำคัญมากที่สุดมาเป็นคำถามแรกในเครือข่าย ซึ่งก็เป็นหลักการเดียวกับ เครือข่ายที่ใช้ลักษณะที่สำคัญร่วมกัน

ข้อดี ข้อเสียของเครือข่ายแบบแยกก็จะเหมือนกับของ เครือข่ายที่ใช้ลักษณะร่วมกัน มีเพียงบางส่วนเท่านั้นที่ต่างกัน ข้อดีที่เครือข่ายแบบนี้มีเหนือกว่าคือ การดึงมาใช้มีประสิทธิภาพมากกว่า เพราะการตอบคำถามแล้วเลือกเส้นทางในทรีสามารถ อิมพลีเมนท์(implement) ได้อย่างมีประสิทธิภาพกว่า อีกข้อหนึ่งคือความสัมพันธ์ระหว่างดัชนีกับเครือข่ายขององค์กรชัดเจนเข้าง่าย และ ข้อสุดท้ายคือ รายละเอียดของลักษณะ(Attribute) และค่า(Value) แยกจากกัน ทำให้ง่ายต่อการแยกแยะว่ารายละเอียดของลักษณะไหนมีประโยชน์มากกว่าตัวอื่นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อเสียของเครือข่ายแบบแยกทีต่างจาก เครือข่ายที่ใช้ลักษณะร่วมกัน คือ บางครั้งการตัดสินใจผิดพลาดเพียงจุดเดียวจะทำให้เราพลาดกรณีที่ดีที่สุดได้ เพราะเราใช้คำถามเพียงคำถามเดียวในการตัดสินใจแต่ละครั้ง แต่ในเครือข่ายที่ใช้ลักษณะร่วมกัน โอกาสพลาดจะน้อยกว่า เพราะในการตัดสินใจจะพิจารณาลักษณะหลายๆ ตัว

ข้อเสียที่ทั้งสองแบบมีร่วมกัน คือ ถ้าในการตัดสินใจครั้งใดมีข้อมูลที่ไม่สมบูรณ์จะทำให้ไม่สามารถเลือกได้ว่าจะไปตามทรีทางโหนดไหน

4. เครือข่ายแบ่งแยกส่วนที่ซ้ำ(Redundant Discrimination Network) , ใช้การสืบค้นตามเส้นทางกระจายก่อน เป็นวิธีที่ช่วยแก้ปัญหาข้อมูลที่ใช้ในการตัดสินใจไม่สมบูรณ์ได้ โดยการเก็บกรณีในเครือข่ายที่แยกกันหลายๆ ตัวซึ่งจัดกรณีต่างกัน แต่กรณีที่อยู่ภายในเหมือนกัน นั่นคือเก็บซ้ำซ้อนกันนั่นเอง ในการค้นหาจะค้นหาแบบขนาน เมื่อเครือข่ายใดพบคำถามที่มีข้อมูลไม่เพียงพอในการหาตอบจะยกเลิกการหาในเครือข่ายนั้น ดังนั้นการที่มีเครือข่ายหลายๆ ตัวซ้ำซ้อนกันจึงเป็นวิธีที่ทำให้สามารถเข้าถึงข้อมูลได้ในหลายๆทาง ถึงแม้จะมีปัญหาว่าทางใดมีข้อมูลไม่พอในการพิจารณาก็ไม่เกิดปัญหา เพราะยังมีทางอื่นที่เข้าถึงข้อมูลได้อีก

ข้อดีของเครือข่ายประเภทนี้ จะเหมือนกับเครือข่ายที่ให้ความสำคัญ แต่มีข้อดีบางอย่างที่เพิ่มเข้ามา อย่างแรกคือ เนื่องจากเข้าถึงข้อมูลได้หลายทาง ทำให้ปัญหาในการขาดข้อมูลในการัดสินใจหมดไป อีกอย่างหนึ่ง คือการเพิ่มกรณีเข้าไปในเครือข่าย จะใช้ความรู้แบบทั่วไป (Generalization) ถึงแม้ว่าตัวระบบอ้างกรณี จะไม่ใช้ความรู้แบบนี้ แต่การใช้ความรู้ในบางส่วนก็ทำให้ระบบมีประสิทธิภาพมากขึ้น

แต่เครือข่ายแบบนี้ก็มีข้อเสียที่นอกเหนือจากข้อเสียของเครือข่ายแบบแยกตามความสำคัญ (ยกเว้นปัญหาข้อมูลไม่สมบูรณ์) คือ ต้องใช้เนื้อที่จำนวนมากในการเก็บ กรณีและต้องใช้เวลามากในการแก้ไขปรับปรุงเครือข่าย เพราะต้องปรับปรุงในทุกๆ เครือข่าย ที่ซ้ำซ้อนกัน ปัญหาที่สำคัญที่สุดคือวิธีนี้อาจทำให้กรณีที่ดึงมาใช้ ขึ้นมามีจำนวนมากจนเกินไป จนทำให้ประสิทธิภาพโดยรวมของระบบลดลง

5. แฟลทเมโมรี(Flat Memory) , การค้นหาแบบขนาน(Parallel Search) เป็นการใช้องค์กร เป็นแฟลทเมโมรีธรรมดา แต่ใช้หน่วยประมวลผลหลายๆ ตัวในการค้นหาเป็นการค้นหาแบบขนาน ซึ่งวิธีนี้จะช่วยเพิ่มประสิทธิภาพในการค้นหา โดยที่ไม่ต้องใช้วิธีการที่ยุ่งยาก และสามารถ ค้นหา ได้ทั้ง ไลบรารีของกรณี

ข้อดีของวิธีนี้จะเหมือนกับการใช้แฟลทเมโมรีแบบค้นหาแบบอนุกรม แต่จะมีข้อดีอีกด้านที่เพิ่มขึ้นมาคือ ความเร็วในการค้นหา ข้อเสียคือ ต้องใช้ฮาร์ดแวร์ที่มีราคาแพง (Multiprocessor) และการใช้วิธีการจับคู่ต้องใช้ในรูปแบบเรียบง่ายซึ่งไม่ขึ้นกับ สิ่งแวดล้อม ในการจับคู่ เพราะต้อง ค้นหา ทั้งไลบรารีของกรณี จึงไม่สามารถกำหนด สิ่งแวดล้อม ที่เฉพาะเจาะจงได้

6. หน่วยความจำลำดับขั้น(Hierarchical Memory) , โดยการค้นหาแบบขนานเป็นการใช้ องค์กรแบบลำดับขั้น และใช้ Multiprocessor ในการ ค้นหา ซึ่งจะทำให้ได้ข้อดีของทั้งองค์กรแบบลำดับขั้น และข้อดีของการค้นหาแบบขนาน ข้อดีที่เพิ่มขึ้นมาคือ กรณีที่มีความใกล้เคียงกันจะอยู่ใกล้กันในไลบรารีของกรณี ซึ่งทำให้สามารถค้นหากรณีที่ใกล้เคียงกันได้ง่าย ทำให้การดึงมาใช้ มีประสิทธิภาพมากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อเสียของวิธีนี้ คือ ยากที่จะทำให้รู้แน่ชัดถึง กรณีที่ดีที่สุดเพราะการอิมพลีเมนท์ มีหลากหลาย และอีกข้อที่สำคัญคือ กรณี ที่ได้อาจมากจนเกินความจำเป็นทำให้ประสิทธิภาพการดึงมาใช้ลดลง

5.2 การจับคู่และการจัดลำดับของกรณี(Matching and Ranking Cases)

การจับคู่(Matching) คือการเปรียบเทียบว่า กรณีที่เราพิจารณานั้นใกล้เคียงกันมากแค่ไหน ส่วนการจัดลำดับเป็นการประเมินว่าในกลุ่มของกรณีที่เราดึงมาใช้ขึ้นมาทำการจับคู่นั้น กรณี ใดมีประโยชน์ในสถานการณ์นั้นที่สุด

จากที่ได้กล่าวมาแล้วว่าการจับคู่นั้นจะต้องพิจารณาใน 2 ประเด็นหลัก คือ การคำนวณหาความใกล้เคียงของมิติ และ การลำดับความสำคัญของแต่ละมิติ ซึ่งทั้ง 2 อย่างนี้จะต้องกระทำบน สิ่งแวดล้อมของเป้าหมายของ ตัวอ้างเหตุผล หรือสถานการณ์ใหม่นั่นเอง

ก่อนที่จะกล่าวถึงโครงสร้างในการจับคู่ และการจัดลำดับจำเป็นที่จะต้องทำความเข้าใจถึงหลักการเบื้องต้นเช่น ความหมาย หรือ คำจำกัดความขององค์ประกอบสำคัญของวิธีการเหล่านี้ก่อน

ส่วนรายละเอียด(Descriptor) หมายถึง ค่าของลักษณะที่ใช้ในการบรรยาย หรือบอกลักษณะของกรณี

มิติ (Dimension) หมายถึง คู่ของลักษณะ จากส่วนรายละเอียด ที่ใช้ในการเปรียบเทียบระหว่างกรณี

ลักษณะในบางครั้งจะหมายถึง ส่วนรายละเอียดแต่ในบางครั้งจะหมายถึงมิติ แต่ในที่นี้จะหมายถึง ส่วนรายละเอียด

การจับคู่ จะมี 2 แบบ คือ การจับคู่ในมิติ(Dimension match) และ การจับคู่โดยรวม(Aggregate match) ในการค้นหาไลบรารีของกรณี เพื่อทำการดึงมาใช้แต่ละครั้งจะมีการเรียก การจับคู่ในมิติ เพื่อเปรียบเทียบค่าระหว่างกรณี กับสถานการณ์ใหม่ ส่วนการจับคู่โดยรวมจะใช้ในการ การจับคู่ และการจัดลำดับ หลังจากที่มีการดึงมาใช้ขึ้นมาแล้วซึ่งก็จะเป็นการทำการจับคู่ในมิติ ในแต่ละมิติ แล้วนำค่าที่ได้มาประกอบกับลำดับความสำคัญของแต่ละมิติ

เกณฑ์ในการจับคู่(Matching Criteria) เป็นการระบุลำดับความสำคัญของมิติที่ต่างกัน ซึ่งจะแบ่งเป็นโกบอล(Global) และ โลคอล(Local) ซึ่งถ้าเป็นโกบอล จะเป็นการระบุเกณฑ์การจับคู่ เพียงเล็กน้อย นั่นหมายความว่าการจับคู่ จะขึ้นกับสิ่งแวดล้อมน้อย ซึ่งกรณีนี้จะใช้กับกลุ่มของ กรณี ขนาดใหญ่ซึ่งอาจเป็นไลบรารีของกรณี หรือ ส่วนของกรณี ถ้าต้องการจะให้สิ่งแวดล้อม มีผลมากขึ้น ก็จำต้องพิจารณาเป็นโลคอล ซึ่งมีเกณฑ์การจับคู่มากกว่า และใช้กับกลุ่มของกรณีขนาดเล็ก

ผลลัพธ์ที่ได้จากการการจับคู่จะเป็นตัวบ่งชี้ถึงความใกล้เคียงของกรณี นั้นกับสถานการณ์ใหม่ ซึ่งทำได้ 2 แบบ คือ ค่าจับคู่สัมบูรณ์(absolute match score) ซึ่งเป็นการคำนวณโดยพิจารณาแต่ละกรณีเป็นอิสระ และ ค่าความสัมพันธ์ในการจับคู่(relative match score) เป็นการคำนวณโดยการเปรียบเทียบระดับความใกล้เคียงกับ กรณี อื่นๆ ด้วย

วิธีของการจับคู่ และการจัดลำดับ จะต้องมีอินพุทที่จำเป็น 5 ตัว คือ

- สถานการณ์ใหม่ เพื่อใช้เป็นเป้าหมายของการจับคู่ และการจัดลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- จุดประสงค์ของการดึงกรณีมาใช้ หรือจุดประสงค์ของ ตัวอ้างเหตุผล นั่นคือระบุว่า ตัวอ้างเหตุผล จะการดึงกรณีมาใช้ ไปเพื่อทำอะไร
- กลุ่มกรณีที่เรียกซ้ำขึ้นมา
- ดัชนี ใช้ในการเรียกซ้ำ เพื่อใช้ระบุหลักสำคัญที่ใช้ในการดึงมาใช้
- เกณฑ์การจับคู่เพื่อระบุว่าเมื่อใดควรจะหยุดทำการจับคู่ และการจัดลำดับ เพื่อทำการ ส่งค่ากรณีที่เป็นประโยชน์ ซึ่งบางครั้งในการดึงกรณีมาใช้ ก็ไม่จำเป็นต้องค้นหา กรณี ที่ดีที่สุดเสมอไป บางครั้งถ้าหากได้กรณีที่ดีพอก็เพียงพอแล้วสำหรับบางกรณี

จากคำจำกัดความขององค์ประกอบข้างต้นเราสามารถพิจารณาถึงวิธีการของการจับคู่ และการจัดลำดับได้ ซึ่งในการจะให้กรณี ผลลัพธ์จากการจับคู่ และการจัดลำดับ เป็นประโยชน์ที่สุด จะต้องพิจารณาถึงประเด็นหลักๆ 3 ประเด็น คือ

1. พิจารณาความตรงกันของ ลักษณะ (Finding Correspondences) เป็นการประเมินว่า ลักษณะใดเหมาะที่จะนำมาเปรียบเทียบกัน การจะเหมือนกัน นั่นหมายถึง ลักษณะนั้นมีบทบาทหน้าที่ที่เหมือนกัน ซึ่งพิจารณาได้หลายๆ แบบ ดังนี้

- พิจารณาค่า 2 ค่าจาก 2 กรณี ซึ่งอยู่ใน ลักษณะ ชนิดเดียวกัน (ชื่อเดียวกัน) ซึ่งวิธีนี้ค่อนข้างจะง่าย เพราะเป็นการนำค่ามาเทียบกันโดยตรง แต่ก็มีข้อจำกัดคือ ในการเก็บ กรณี บางครั้งใน 1 ลักษณะ อาจจะเก็บค่าที่มากกว่า 1 ค่า หรือ บางลักษณะ ไม่มีค่า ทำให้การเปรียบเทียบไม่สามารถทำได้อย่างชัดเจน
- ใช้สามัญสำนึกในการสร้างฮิวริสติกหรือเป็นการใช้ฮิวริสติกซึ่งอาศัยสามัญสำนึกเข้ามาช่วย โดยใช้หลัก ฮิวริสติก 5 ข้อ ดังนี้
  1. จับคู่ลักษณะที่ชื่อเหมือนกัน (เหมือนวิธีที่แล้ว)
  2. จับคู่ลักษณะที่มีบทบาทหน้าที่เหมือนกัน
  3. จับคู่ลักษณะที่มีข้อบังคับ (constraint) เหมือนกัน
  4. จับคู่ลักษณะที่เหมือนกัน หรือมีความคล้ายคลึงกัน
  5. สามารถจับคู่ลักษณะเดี่ยวๆ กับกลุ่มของลักษณะได้
- ใช้รูปแบบการใช้เหตุผล(Causal Model) ทำให้สามารถระบุได้ว่า ลักษณะใดที่ทำหน้าที่เหมือนกัน ระหว่างกรณี 2 กรณี ซึ่งในการใช้รูปแบบ จะพิจารณาบทบาทของแต่ละ ลักษณะ โดย ลักษณะหนึ่งๆ อาจจะทำหน้าที่ต่างๆ กันในสถานการณ์ที่เปลี่ยนไป
- ใช้ การแมปโครงสร้าง(Structure Mapping) เพื่อให้สามารถทำการเปรียบเทียบ ลักษณะ ซึ่งมีบทบาทเหมือนกัน แต่แสดงอยู่บนโครงสร้างที่ต่างกันได้ ซึ่งวิธีนี้จำเป็นมากสำหรับระบบที่มีการใช้ ลักษณะ ซึ่งใช้โครงสร้างหลายๆ แบบร่วมกัน

2. คำนวณ Degree ของความเข้ากันของ ลักษณะ (Computing Degree of Similarity of Corresponding Features) ซึ่งมีวิธีต่างๆ ดังนี้

- ใช้การสรุปเป็นลำดับขั้น(Abstraction Hierarchy) คำนวณออกมาในรูปของค่า MSCA (most specific common abstraction) โดยให้ MSCA มีค่าอยู่ระหว่าง 0 กับ 1 โดย 0 หมายถึงเหมือน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กันน้อยที่สุด และ 1 หมายถึง เหมือนกันมากที่สุด การพิจารณาความเหมือนจะใช้ การสรุปเป็นลำดับขั้นเป็นหลัก ปัญหาที่เกิดขึ้น คือ ถ้าการสรุปเป็นลำดับขั้นถูกสร้างขึ้นมาโดยวิธีสรุปที่ต่างกัน ซึ่งจะทำให้ค่าที่ได้ต่างจากการพิจารณาค่า MSCA ตามปกติ

- ใช้การวัดระยะทางปริมาณและคุณภาพ โดยการแบ่งค่าของ ลักษณะ ออกเป็นช่วงๆ แล้วทำการเปรียบเทียบระยะระหว่างช่วง วิธีนี้อาจจะเกิดปัญหาเมื่อค่าที่พิจารณาอยู่ที่ขอบของช่วง ซึ่งแก้ได้ 2 วิธีคือ พิจารณาค่าที่อยู่ที่ขอบของช่วงเป็นกรณีพิเศษ เช่นร่นระยะ หรืออีกวิธีหนึ่งคือ ใช้ค่าของช่วงให้ซ้อนเหลื่อมกัน นอกจากนี้ยังมีอีกวิธีคือการนำค่า 2 ค่ามาหาค่าความต่างกันก่อน แล้วดูว่าผลต่างอยู่ในช่วงใด วิธีนี้จะใช้ได้ดีโดยเฉพาะกับการเปรียบเทียบเชิงปริมาณ แต่การเปรียบเทียบเชิงคุณภาพก็ยังสามารถทำได้ ถึงแม้จะต้องทำในขั้นตอนที่ซับซ้อนกว่า

3. ลำดับความสำคัญในแต่ละมิติ (Weighting Dimension of a Representation) ขั้นตอนนี้จะเป็นการนำเอาค่าที่ได้จากการคำนวณในขั้นตอนที่แล้วมาทำการถ่วงน้ำหนักตามความสำคัญของแต่ละมิติ ซึ่งความสำคัญของแต่ละมิติอาจเป็นได้ทั้งแบบ Static (กำหนดแน่นอนตายตัว) และ Dynamic (สามารถเปลี่ยนแปลงตามสถานการณ์) ซึ่งการจะได้มาซึ่งความสำคัญของมิตินั้นมีได้หลายวิธี บางระบบจะใช้ความเชี่ยวชาญของมนุษย์ในการวิเคราะห์หาลำดับความสำคัญ บางระบบใช้สถิติ หรือ บางระบบก็ใช้ลำดับขั้น

.ในการ การจัดลำดับ มีหลายวิธีที่ใช้กัน เช่น ตัวเลข(Numerical) , ฮิวริสติก หรือ แบบผสม ซึ่งแต่ละวิธีก็มีข้อดีแตกต่างกันไป แบบตัวเลขสามารถคำนวณได้ง่ายโดยการนำค่าที่ได้จากการเปรียบเทียบแต่ละมิติมาคูณกับค่าความสำคัญของมิตินั้น แล้วนำมารวมกันจะได้เป็นค่าความเหมือนของกรณีนั้น หลังจากนั้นก็นำมาเปรียบเทียบกับกรณีอื่นๆ แล้วเลือกกรณีที่ให้ค่าที่ดีที่สุด ส่วนแบบ ฮิวริสติก ก็เป็นการเลือกกรณี โดยใช้ ฮิวริสติก เข้าช่วย ซึ่ง กรณี ที่ ดึงมาใช้ อาจจะไม่ใช่ กรณี ที่ดีที่สุดก็ได้

5.3 การจัดดัชนี และการเรียกคืนกรณี

จากที่ได้กล่าวมาแล้วถึงส่วนต่างๆ ของระบบในการอ้างเหตุผลด้วยกรณี จะเห็นได้ว่าส่วนต่างนั้นมีความซับซ้อนอย่างยิ่ง สิ่งหนึ่งที่ต้องให้ความสำคัญเป็นอย่างมากคือ ความสัมพันธ์ในระหว่างกระบวนการ ซึ่งซับซ้อนมาก สิ่งที่จะต้องพิจารณาในขณะนี้คือ การประเมินสถานการณ์ที่เข้ามาใหม่ทั้ง ก่อน และระหว่างการทำการเรียกกรณี เพราะเราไม่สามารถรู้ได้ว่ากรณีใดคือกรณีที่สามารถนำมาใช้ได้ในสถานการณ์ใหม่ที่เกิดขึ้น และไม่สามารถรู้ได้เลยว่ากรณีที่จะมีประโยชน์นั้น ถูกชี้อยู่ด้วยดัชนีตัวใด

สิ่งที่สองที่จะต้องพิจารณาคือ การสร้างดัชนี จากที่ได้กล่าวมาแล้ว ดัชนีอาจจะสร้างเป็นส่วนแขนของ เครือข่ายที่แยกกัน หรืออาจสร้างเป็นน้ำหนัก (Weight) ของมิติ ซึ่งใช้ในการประเมินค่าของฟังก์ชั่นทางตัวเลข แต่ดัชนีก็อาจจะสร้างโดยใช้เป็นตัวอ้างถึงลักษณะที่มีผลต่อผลลัพธ์ของกรณีก็ได้ ซึ่งจะกล่าวต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.4 การประเมินสถานการณ์ที่เข้ามาใหม่

ในการเพิ่มกรณีใหม่ๆ เข้าไปในไลบรารีของกรณี จะต้องเริ่มต้นจากการประเมินว่า ณ สถานการณ์ใดที่กรณีนี้จะเป็นประโยชน์ แล้วสร้างคำอธิบายลักษณะของสถานการณ์ทั้งหมด แล้วเลือกส่วนหนึ่งเพื่อนำมาสร้างเป็นดัชนี

แต่ก็มีปัญหาในกรณีที่เราจะดึงกรณีจากไลบรารีของกรณี ซึ่งต่างจากการเพิ่มกรณีลงไปในไลบรารีของกรณี เพราะการดึงกรณีเพื่อนำมาใช้ในสถานการณ์ใหม่ เราไม่เข้าใจถึงสถานการณ์นั้นดีพอที่จะทำการดึงกรณีที่มีประโยชน์ที่สุดขึ้นมาได้ จึงต้องมีการประเมินสถานการณ์ของปัญหา เพื่อที่จะระบุว่าดัชนีที่จะใช้ในการค้นหากรณีคืออะไร

การประเมินสถานการณ์ปัญหา หรือการวิเคราะห์สถานการณ์ คือการประเมินถึงสถานการณ์ที่เกิดขึ้น ว่าเป็นประเภทไหน ส่วนสำคัญคืออะไร และมีอะไรอีกบ้างที่เกี่ยวข้องกับสถานการณ์นี้ ซึ่งในการประเมินจะมีผลต่อการดึงกรณีขึ้นมาใช้งาน

ก่อนทำการค้นหาในไลบรารีของกรณี จะต้องมีการประเมินว่าในสถานการณ์ใหม่นี้จะใช้ มิติใดเป็นดัชนีในการค้นหากรณี ซึ่งในกระบวนการนี้เป็นงานหนักของวิศวกรความรู้ ที่จะต้องทำให้ระบบสามารถวิเคราะห์ได้ว่าในสถานการณ์ใด เหมาะจะใช้วิธีการในการค้นหาดัชนีอย่างไร ซึ่งก็คือการเลือกมิติที่จะนำมาใช้เป็นดัชนีนั่นเอง ในวิธีการนี้จะต้องระบุรายการของมิติ หรือรายการของวิธีการในการค้นหากรณี เพื่อที่จะให้ระบบใช้ในการค้นหากรณี ซึ่งจะตัดสินใจเลือกตามสถานการณ์ที่เกิดขึ้น

ถึงแม้ว่าการทำรายการควบคุมการค้นหากรณี จะเป็นแบบรวมทั้งหมด แต่ก็อาจจะขึ้นกับลักษณะการนำกรณีไปใช้ หรือบริบทนั่นเอง หรืออาจจะแตกต่างกันไปตามชนิดของกรณีก็ได้

ระหว่างการทำการค้นหาไลบรารีของกรณี อาจจะต้องมีการปรับแต่ง หรือเพิ่มเติมบริบทเพิ่มเติม ซึ่งขึ้นอยู่กับลักษณะของปัญหาที่ระบบจะต้องรองรับ ซึ่งมีทั้งแบบถาวร (Static) และแบบที่ปรับเปลี่ยนตามสถานการณ์ (Dynamic) ในแบบถาวรจะเป็นการใช้หน่วยความจำแบบแยกเครือข่าย โดยตั้งคำถามตามเส้นทางที่จะต้องเดินทางผ่าน ซึ่งคำตอบนั้นจะได้มาจากสถานการณ์ใหม่ที่เข้ามา ซึ่งถ้าคำถามใดยังไม่สามารถระบุคำตอบได้ ก็จะต้องทำการค้นหาคำตอบ คำถามเหล่านี้จะเป็นตัวกำหนดรายการแผนที่จะปฏิบัติ

ส่วนแบบที่เปลี่ยนแปลงตามสถานการณ์ จะเรียกกรณีทั้งหมดที่เข้ากันได้บางส่วนกับสถานการณ์ใหม่ขึ้นมา แล้วเสนอแผนการซึ่งอยู่บนพื้นฐานของความเหมือนและความต่างของกรณีที่เรียกขึ้นมา ซึ่งหลังจากนั้นก็จะทำการเรียกกรณีขึ้นมาอีกครั้งจากแผนที่ได้มา ถ้ายังจำเป็นต้องให้ได้กลุ่มของกรณีที่เล็กลงมาอีก ก็จะทำกระบวนการเหล่านี้ซ้ำอีก

หลังจากทำการเรียกกรณีขึ้นมาแล้ว จะต้องระบุถึงบริบทของปัญหาอีกครั้งหนึ่ง เพราะหลังจากที่ได้กลุ่มของกรณีขึ้นมาแล้ว ก็ควรจะประเมินอีกครั้งว่ากรณีที่เรียกขึ้นมานั้น ตรงตามความต้องการของปัญหาหรือไม่ หากไม่ก็แสดงว่า การประเมินสถานการณ์ไม่ตรงตามความต้องการของปัญหา จะต้องมีการประเมินสถานการณ์ใหม่อีกครั้งหนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.5 การจัดการดัชนี

ถ้าพิจารณาถึง การค้นหา (Search) การจับคู่ (Matching) และการประเมินสถานการณ์ (Situation Assessment) จะเห็นว่า ดัชนี (Index) แสดงบทบาทสำคัญๆ ต่างกัน ในระบบการอ้างเหตุผลด้วยกรณี บทบาทแรกคือการระบุ หรือเจาะจงตัวกรณี นั่นคือ ดัชนี อธิบายถึงสถานการณ์ซึ่งกรณีนั้นสามารถเป็นประโยชน์ หรือนำไปเป็นแนวทางได้ในอนาคต รวมทั้งใช้จำแนกกรณีที่มีความคล้ายคลึงกันออกจากกัน

บทบาทที่สองคือ เป็นเครื่องมือในการค้นหากรณีในไลบรารีของกรณี นั่นคือเป็นตัวชี้ในโครงสร้างของฐานข้อมูลกรณี หรือในทางกายภาพก็เป็นตัวแยกกลุ่มกรณีออกจากกันนั่นเอง เสมือนเป็นตัวที่ชี้แนวทางในการค้นหากรณีว่า ควรจะพิจารณากรณีในกลุ่มใดบ้าง

บทบาทหน้าที่ที่สามคือ เป็นตัวระบุความสำคัญของแต่ละมิติในกรณี ในการจัดลำดับด้วยวิธีเชิงตัวเลข จากที่ได้พิจารณามาแล้วว่าในการจับคู่ และการจัดลำดับ ในแต่ละมิติของกรณีจะต้องมีการกำหนดระดับความสำคัญ ซึ่งจะเป็นประโยชน์มากในการประเมินว่ากรณีนั้นเหมาะกับสถานการร์ใหม่เพียงใด ซึ่งระดับความสำคัญของมิติของกรณี ก็คือ ดัชนีนั่นเอง

จากที่ได้พิจารณาบทบาททั้ง 3 ของกรณี ไม่จำเป็นที่ดัชนีจะต้องทำหน้าที่ทั้ง 3 อย่าง ในระบบการอ้างเหตุผลเชิงกรณี ขึ้นอยู่กับอัลกอริธึ่มที่เลือกใช้ เช่น การใช้วิธีแยกเครือข่าย และ รีดันแดนท์ดิสคริมิเนชันเน็ทเวิร์ค จะใช้ดัชนี เป็นคำถามและคำตอบในการค้นหาเน็ทเวิร์ค ดังนั้นในกรณีนี้ ดัชนีจึงทำหน้าที่ทั้ง 2 อย่าง คือเป็นทั้ง ตัวระบุกรณี และเครื่องมือในการค้นหากรณี ในการค้นหากรณีแบบขนาน บนแฟลท เมโมรี่ ดัชนีจะมีบทบาทในกระบวนการประเมินการจับคู่ โดยแสดงถึงลักษณะเด่นที่สำคัญของกรณี หรือในการลำดับความสำคัญของกรณีด้วยวิธีเชิงตัวเลข ดัชนีจะเป็นตัวระบุถึงความสำคัญของแต่ละมิติของกรณี เพื่อให้กระบวนการประเมินการจับคู่นำไปประเมินได้ จะเห็นได้ว่า บทบาทของดัชนีนั้นแตกต่างกันไปขึ้นอยู่กับวิธีและอัลกอริธึ่มที่เลือกใช้

5.6 ประสิทธิภาพ ความแม่นยำ และความยืดหยุ่น

ประเด็นที่สำคัญอีกอย่างหนึ่งในการพิจารณาเลือกวิธีการจัดการดัชนี และกระบวนการดึงกรณี คือ ผลต่างของวิธีต่างๆ ทั้งทางด้าน ประสิทธิภาพ ความแม่นยำ และความยืดหยุ่น ในการค้นหากรณีแบบซีเรียลที่มีประสิทธิภาพ ต้องการดัชนีที่สามารถเจาะจงไปยังกลุ่มกรณีที่เป็นประโยชน์ ซึ่งดัชนีจะทำหน้าที่ในการจำแนกกรณีออกจากกรณีอื่นๆ และจัดส่วนแบ่งไลบรารีของกรณีให้เป็นส่วนอย่างเหมาะสม ความต้องการในประสิทธิภาพ หมายความถึง ต้องการดัชนี ซึ่งสามารถเจาะจงไปยังกลุ่มของกรณีขนาดเล็กที่ต้องทำการค้นหาได้ ความต้องการในความแม่นยำ หมายถึง ต้องการให้ดัชนีที่ใช้ในการจำแนก หรือแบ่งกลุ่มของกรณี สามารถระบุไปถึงกลุ่มของกรณีที่มีประโยชน์ ได้อย่างเพียงพอ ซึ่งถ้าต้องการความแม่นยำจะต้องพิจารณาดัชนี โดยนึกถึงประโยชน์เป็นสำคัญ โดยดัชนีอาจเป็นลักษณะ หรือกลุ่มลักษณะซึ่งสามารถคาดการณ์ถึง การใช้ประโยชน์ของกรณีนั้นในอนาคต

ความต้องการด้านความยืดหยุ่นก็เป็นอีกข้อจำกัดหนึ่ง ถ้าดัชนีมีความยืดหยุ่นดี ก็ย่อมจะมีประโยชน์ต่อการใช้งานในอนาคต ยกตัวอย่างเช่น ถ้าเราอ่านหนังสือสักเล่ม แล้วเราพบสถานการณ์ซึ่งเราไม่เข้าใจ และไม่มีประสบการณ์มาก่อน เราอาจคาดเดาได้จากตัวบริบท แต่ก็ไม่เสมอไป ซึ่งก็เปรียบเทียบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ได้กับระบบการอ้างข้อมูลเชิงกรณีเช่นเดียวกัน ถ้าเรากำหนดดัชนีเผื่อไว้เฉพาะที่คาดการณ์เอาไว้ ก็จะไม่สามารถดึงกรณีสำหรับสถานการณ์ที่อยู่นอกเหนือการคาดการณ์ได้ เพื่อป้องกันเหตุการณ์นี้ เราอาจให้วิธีกำหนดดัชนี บนพื้นฐานที่สอดคล้องกับลักษณะที่สามารถเข้าใจได้เป็นอย่างดี ซึ่งอาจจะไม่ได้จำแนกกรณีออกจากกรณีอื่นเป็นอย่างดี แต่ก็สามารถอธิบายได้อย่างเพียงพอ ซึ่งจะทำให้มีโอกาสในการจับคู่แบบบางส่วนในสถานการณ์ในอนาคต ความต้องการในความยืดหยุ่น จึงหมายถึง ดัชนีที่จัดให้สอดคล้องกับลักษณะที่สามารถอธิบายได้เป็นอย่างดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
การใช้กรณี และการดัดแปลงกรณี

6.1 วิธีและกลยุทธ์การดัดแปลง (Adaption Method and Strategies)

เพราะกรณีที่เราค้นขึ้นมานั้น ไม่สามารถตรงกับสถานการณ์ใหม่ได้ทั้งหมด จึงต้องมีการดัดแปลงผลลัพธ์ของกรณีนั้น มิฉะนั้นกรณีที่ค้นขึ้นมาก็ไม่สามารถนำมาใช้ประโยชน์ในสถานการณ์ใหม่ได้อย่างมีประสิทธิภาพ ซึ่งวิธีในการดัดแปลงกรณีก็มีหลายๆ วิธีต่างกันไป

6.1.1 วิธีแทนค่า (Substitution)

วิธีแทนค่าเป็นกระบวนการของการเลือก และทำการแทนค่าบางส่วนของส่วนประกอบของผลลัพธ์ของกรณีที่เราค้นขึ้นมา ซึ่งก็มีหลายวิธีในการแทนค่า อาจแทนค่าส่วนประกอบส่วนใดส่วนหนึ่ง กลุ่มของส่วนประกอบ หรือส่วนประกอบทั้งหมดของผลลัพธ์ก็ได้ ขึ้นอยู่กับสถานการณ์

6.1.1.1 รีอินสแตนทิเอชั่น(Reinstantiation)

วิธีนี้ใช้สำหรับกรณีที่ ผลลัพธ์เก่า กับผลลัพธ์ที่ต้องการมีโครงสร้างที่เหมือนกัน แต่แตกต่างกันในด้านของบทบาทหน้าที่ของผลลัพธ์นั้น แต่หน้าที่ของผลลัพธ์มีความสัมพันธ์กัน โดยจะมีการแทนค่าลงในส่วนประกอบที่เป็นส่วนประกอบแบบเดียวกันในโครงสร้างของผลลัพธ์ สิ่งที่ควรพิจารณาในการแทนค่าด้วยวิธีนี้คือ

- ความเกี่ยวข้องระหว่างบทบาทหน้าที่ของกรณีเก่า และกรณีใหม่
- โครงสร้างของผลลัพธ์ตัวเก่า

จะเห็นได้ว่าวิธี Reinstantiation เป็นวิธีที่ง่ายมาก เมื่อสามารถระบุโครงสร้างของผลลัพธ์ของทั้งผลลัพธ์ตัวเก่าและผลลัพธ์ใหม่ที่เราต้องการ และสามารถระบุบทบาทของผลลัพธ์ที่เราต้องการในปัญหาได้ จะเห็นได้ว่าในความหมายโดยตรงตัวก็ตรงกับวิธีในความเป็นจริงของ Reinstantiation กล่าวอย่างง่ายคือ เสมือนเป็นการรีอินสแตนทิเอชั่น ผลลัพธ์ขึ้นมาใหม่สำหรับสถานการณ์ใหม่ โดยใช้โครงสร้างของผลลัพธ์เก่า

อย่างไรก็ตาม ถึงแม้วิธี รีอินสแตนทิเอชั่น จะสามารถเข้าใจการทำงาน และสร้างขึ้นใช้ได้ง่าย แต่ก็มีสถานการณ์ที่วิธีนี้ไม่สามารถทำงานได้ดี คือ

- เมื่อโครงสร้างของผลลัพธ์เก่าไม่สามารถระบุได้ชัดเจน
- เมื่อบทบาทหน้าที่ของคำบรรยายในผลลัพธ์ไม่สามารถระบุได้ชัดเจน
- เมื่อบทบาทหลายๆตัว ทำหน้าที่ร่วมกัน

นอกจากนั้นยังมีสิ่งที่รีอินสแตนทิเอชั่น ทำไม่ได้ เช่น การปรับค่าของส่วนประกอบบางส่วน หรือการปรับเปลี่ยนโครงสร้างของผลลัพธ์ ซึ่งทำให้หลังจากการใช้วิธี Reinstantiation แล้ว จะต้องมีวิธีอื่นๆ ที่นำมาใช้เพิ่มเติมอีก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.1.1.2 วิธีปรับพารามิเตอร์ (Parameter Adjustment)

วิธีปรับพารามิเตอร์ เป็นการปรับเปลี่ยนค่าในส่วนประกอบของผลลัพธ์เก่า โดยพิจารณาจากระดับความแตกต่างของผลลัพธ์เก่า และผลลัพธ์ที่เราต้องการ ซึ่งนั่นก็คือ การเปลี่ยนค่าในส่วนประกอบเพื่อให้สอดคล้องกับความต่างของรายละเอียดของปัญหาของกรณีเก่าและสถานการณ์ใหม่ โดยมีขั้นตอนหลัก 2 ขั้นตอนคือ

- ทำการเปรียบเทียบความแตกต่างระหว่างคำอธิบายปัญหาของปัญหาเก่า และปัญหาใหม่
- ทำการดัดแปลงค่าที่จำเป็นโดยใช้วิธีการดัดแปลงด้วยวิธีทาง ฮิวริสติก ซึ่งจะเห็นได้ว่าตัววิธีเฉพาะนี้มักจะอยู่ในรูปของกฎ (Rule)

สิ่งที่สำคัญที่ควรระลึกในการใช้วิธีปรับพารามิเตอร์ คือ วิธีนี้เป็นเพียงการปรับเปลี่ยนดัดแปลงค่าเท่านั้น นั่นคือต้องมีผลลัพธ์เก่าแล้วนำมาดัดแปลงค่า โดยอยู่บนพื้นฐานของผลลัพธ์เก่า ไม่สามารถนำมาสร้างผลลัพธ์ใหม่ได้

6.1.1.3 โลคอลเซิร์ส(Local Search)

โลคอลเซิร์สเป็นกระบวนในการค้นหาในบางส่วนแอบสแตกชันไฮราคี(Abstraction Hierarchy) ของส่วนที่จะทำการเปลี่ยนแปลง เพื่อนำมาแทนในผลลัพธ์ คล้ายวิธีรีอินสแตนทิเอชันแต่วิธีนี้ต่างกับ รีอินสแตนทิเอชัน คือ รีอินสแตนทิเอชัน เป็นการแทนหน้าที่ของผลลัพธ์ใหม่ โดยแทนทั้งหมดของค่า หรือเป็นการเปลี่ยนบทบาทหน้าที่ของผลลัพธ์นั่นเอง แต่วิธีโลคอลเซิร์ส จะทำการแทนค่าแค่เพียงบางส่วนของบทบาทหน้าที่เท่านั้น โดยผลลัพธ์เก่าที่จะนำมาใช้นั้นจะต้องเป็นผลลัพธ์ที่ค่อยข้างจะใช้ได้อยู่แล้ว ต้องการเพียงการแทนค่าบางส่วน เพื่อปรับให้เข้ากับสถานการณ์ย่อยส่วนน้อยเท่านั้น

ขั้นตอนในการทำ โลคอลเซิร์ส ก็จะเริ่มจากโหนด ที่ผลลัพธ์เก่าที่เราจะทำการแทนค่า แล้วทำการค้นหาจากจุดนั้น ซึ่งอาจจะค้นหาขึ้นหรือลงตามโครงสร้างก็ได้ บางระบบอาจทำให้การค้นหาในส่วนนี้เป็นไปอย่างมีประสิทธิภาพมากขึ้น โดยกำหนดลิงค์ที่เชื่อมโยงกรณีที่มีความเกี่ยวข้องไว้ในไลบรารีของกรณี

6.1.1.4 คิวรีเมโมรี(Query Memory)

พิจารณา โลคอลเซิร์ส อีกครั้งหนึ่ง จะเห็นว่า โลคอลเซิร์ส จะทำการค้นหาค่าที่จะทำการแทนในส่วนแอบสแตกชันไฮราคี ของส่วนที่จะถูกแทนค่าเท่านั้น ซึ่งสามารถค้นหาได้โดยง่าย ถึงแม้จะสามารถสร้าง Network ให้สามารถเชื่อมโยงความสัมพันธ์ของค่าต่างๆ ได้ แต่ถ้าแอบสแตกชันไฮราคีต้องทำหน้าที่หลายๆ อย่าง เป็นไปไม่ได้เลยที่จะกำหนดความสัมพันธ์ระหว่างเครือข่ายนั้นๆ ได้หมด

วิธีคิวรีเมโมรี สามารถแก้ปัญหาในการไม่สามารถหาค่าที่อยู่ในแอบสแตกชันไฮราคี เดียวกันมาทำการแทนค่าได้ของโลคอลเซิร์สได้ โดยคิวรีเมโมรีจะทำการสร้างคำอธิบายบางส่วน เพื่อระบุถึงรายการที่จะมีประโยชน์ในการนำมาใช้แทนค่า เพื่อใช้เป็นหลักในการค้นหา โดยแทนที่จะทำการค้นหาเฉพาะส่วนแบบ โลคอลเซิร์ส ก็จะทำการค้นหาตั้งแต่ด้านบนของเครือข่าย โดยใช้คำอธิบายที่สร้างขึ้นมาเป็นข้อมูลในการค้นหา ซึ่งวิธีในการค้นหาในเครือข่าย ก็ขึ้นอยู่กับโครงสร้างของเครือข่าย ตัวนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.1.1.5 การค้นหาเฉพาะ(Specialized Search)

การค้นหาเฉพาะ อาจมองได้คล้ายกับ คิวรีเมโมรี ซึ่งกำหนดรายละเอียดบางส่วนในการค้นหาต่างกันตรงที่ การค้นหาเฉพาะ จะระบุรายละเอียดของการค้นหามากขึ้น คือกำหนดวิธีในการค้นหาผลลัพธ์ที่ต้องการ เปรียบเทียบกับ โลคอลเซิร์ส ซึ่งไม่ได้ระบุรายละเอียดในการค้นหา สามารถพิจารณาประสิทธิภาพของทั้ง 2 วิธีได้ดังนี้

- ความสิ้นเปลืองในการค้นหา ขึ้นอยู่กับลักษณะเฉพาะตัวของแต่ละเครือข่าย แต่โดยทั่วไป โลคอลเซิร์ส มักจะสิ้นเปลืองมากกว่า
- ความสิ้นเปลืองในการหาคำตอบของคำถามที่ใช้หลังการค้นหา นั่นคือต้องประมวลผลค่าที่ค้นขึ้นมามากน้อยแค่ไหน โดยทั่วไปวิธี การค้นหาเฉพาะ จะใช้จำนวนคำถามน้อยกว่า เพราะมีการคัดเลือกค่าที่ค่อนข้างเฉพาะเจาะจงว่าต้องใกล้เคียงกับผลลัพธ์ที่ต้องการ มากกว่าวิธี โลคอลเซิร์ส
- โอกาสในการค้นพบค่าที่สามารถนำมาใช้งานได้ ถ้าเครือข่ายถูกจัดขึ้นมาอย่างถูกต้อง สามารถแบ่งกลุ่มของค่าที่เกี่ยวข้องกันได้อย่างถูกต้องวิธีโลคอลเซิร์ส จะสามารถค้นค่าที่เกี่ยวข้องขึ้นมาได้อย่างรวดเร็ว แต่ถ้าหากเครือข่ายไม่ได้ถูกแบ่งกลุ่มอย่างถูกต้อง การค้นหาก็จะนานจนเกินที่จะยอมรับได้ แต่ถ้าเป็น การค้นหาเฉพาะ แล้ว สามารถค้นหาค่าที่ต้องการขึ้นมาได้อย่างรวดเร็ว เพราะมันระบุตำแหน่งเฉพาะที่จะทำการค้นหาได้

ในการแทนค่าผลลัพธ์แบบ การค้นหาเฉพาะ จะต้องมี 2 ส่วนที่ต้องให้ความสำคัญ คือ วิธีการค้นหาเฉพาะ ซึ่งต้องเป็น ฮิวริสติก และ แผนการเลือกใช้แต่ละ ฮิวริสติก ในกรณีต่างๆ ซึ่ง วิธีค้นหาแบบ ฮิวริสติก จะต้องสามารถค้นหาค่าทุกแบบ ซึ่งจะเป็นที่ต้องการและต้องมีวิธีทั่วไปในการเข้าถึงค่านั้น นอกจากนั้นตัวประมวลเหตุผลจะต้องมีแนวทางในการเลือก ฮิวริสติก ในแต่ละแบบ ซึ่งจะต้องมีตัวแผนการใช้ ฮิวริสติก เป็นตัวกำหนดการเลือกใช้ ฮิวริสติก ในกรณีต่างๆ

6.1.1.6 ตัวแทนการอ้างกรณี(Case-Based Substitution)

พิจารณา โลคอลเซิร์ส และ การค้นหาเฉพาะ อีกครั้งหนึ่ง โลคอลเซิร์ส จะทำการค้นหาค่าที่จะนำมาแทนจากบริเวณที่ใกล้เคียงใน แอบสแตกชันไฮราคี ส่วน การค้นหาเฉพาะ จะทำการกำหนดตำแหน่งที่เหมาะจะค้นหาค่าจาก แอบสแตกชันไฮราคี แล้วทำการค้นหาค่าที่จะนำไปแทน แต่ในบางครั้ง ค่าที่เหมาะสมที่จะนำไปแทนอาจอยู่ที่กรณีอื่นๆ ก็ได้ โดยวิธี ตัวแทนการอ้างกรณี จะทำการค้นหาค่าที่จะนำไปแทนจากกรณีอื่นๆ ที่เป็นทางเลือกในสถานการณ์นั้นๆ

อาจเกิดคำถามขึ้นว่า เหตุใดกรณีที่ให้คำแนะนำที่เป็นทางเลือกนั้น ไม่ถูกเรียกขึ้นมาเมื่อเราทำการค้นกรณี คำตอบคือ กรณีเหล่านี้ไม่สามารถนำมาใช้ในสถานการณ์นั้นได้ ถึงแม้จะมีคำแนะนำที่เป็นทางเลือกที่มีประโยชน์ เพราะกรณีเหล่านี้ไม่ได้เข้ากันอย่างดีกับสถานการณ์ใหม่ที่สุด

ขั้นตอนที่สำคัญของ ตัวแทนการอ้างกรณี คือ

- ค้นหากรณีที่มีส่วนที่เหมือนกับส่วนที่ต้องการจะทำการแก้ไข โดยค้นหาจากไลบรารีของกรณี ซึ่งในการที่จะค้นหากรณีที่เหมาะสมจะต้อง มี 2 สิ่งที่สำคัญคือ ระบบจะต้องสามารถ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สร้างคำอธิบายถึงสิ่งที่ต้องการจะค้นหาได้ และ กรณีจะต้องถูกจัดให้สามารถค้นหาได้โดยใช้บค่าที่บรรจุอยู่ในส่วนภายในกรณี

- เลือกค่าที่จะนำมาแทนจากตัวเลือกคำแนะนำทั้งหมด

ในการสร้างคำอธิบายถึงสิ่งที่ต้องการค้นหา เรียกว่า การตรวจสอบคอนเท็กซ์ (Context Determination) ซึ่งก็คือ การกำหนดว่ากรณีแบบใดที่ เหมาะจะนำค่ามาแทน

6.1.2 การเปลี่ยนแปลงรูปแบบ (Transformation)

พิจารณาการแทนค่า จะเห็นว่าวิธีแทนค่าเหมาะสำหรับกรณีที่มีค่า หรือหลักการวิธีการที่สามารถถูกแทนค่าได้ แต่ไม่สามารถใช้ได้ในกรณีที่ค่าที่ต้องการ ยังไม่มี หรือกรณีที่ต้องมีการเพิ่มหรือลบบางส่วนของผลลัพธ์

การเปลี่ยนแปลงรูปแบบ เป็นกระบวนการในการแปลงผลลัพธ์ตัวเก่าให้สามารถนำมาใช้ในสถานการณ์ใหม่ได้ โดยการเพิ่ม การลบ หรือการแปลงค่าในส่วนของผลลัพธ์เก่า โดยสามารถแบ่งย่อยได้เป็น 2 แบบ คือ

6.1.2.1 การแปลงค่าโดยใช้สามัญสำนึก (Commonsense Transformation)

การแปลงค่าโดยใช้สามัญสำนึก ถึงแม้จะใช้กลุ่มของ ฮิวริสติก จำนวนไม่มากนัก แต่จำเป็นจะต้องประกอบด้วยส่วนที่สำคัญต่อไปนี้

- ระบบต้องสามารถระบุถึงส่วนประกอบของผลลัพธ์ที่ทำการแปลง ที่ต้องการการแก้ไข
- ระบบจะต้องแสดงถึงความต่างระหว่างส่วนประกอบหลัก และส่วนประกอบรองของผลลัพธ์
- ระบบจะต้องดูแลให้ความสัมพันธ์ภายในระหว่างส่วนประกอบของปัญหา เป็นไปอย่างถูกต้อง

6.1.2.2 การแปลงค่าโดยใช้แนวทางจากแบบจำลอง (Model-Guided Repair)

ในวิธีการแปลงค่าโดยใช้สามัญสำนึก จะใช้หลักการในโลกความจริงทั่วๆ ไป แต่การแปลงโดยใช้แนวทางจากแบบจำลอง จะเป็นการแปลงที่ขึ้นอยู่กับความรู้จากหลักการเหตุผลของระบบ หรือเหตุการณ์บางชนิด

เช่นเดียวกับวิธีปรับพารามิเตอร์ การแปลงโดยใช้แนวทางจากแบบจำลองจะต้องมีการประเมินความต่างของคำอธิบายปัญหาเก่า กับปัญหาใหม่ ขั้นแรกจะต้องประเมินความต่างโดยใช้แบบจำลองเป็นเครื่องมือ หลังจากนั้นจึงทำการแปลงค่าที่เกี่ยวเนื่องในแต่ละความต่างที่ประเมินได้ ซึ่งอาจจะคล้ายกันกับวิธีปรับพารามิเตอร์ แต่ก็มีข้อแตกต่างกัน คือ

- วิธีปรับพารามิเตอร์สามารถทำได้แค่ปรับค่า หรือปริมาณของค่า แต่การแปลงค่าโดยใช้แนวทางจากแบบจำลอง สามารถแก้ไขโครงสร้าง หรือแก้ไขส่วนประกอบบางตัวได้
- ฮิวริสติก ในการปรับพารามิเตอร์เป็น ฮิวริสติก เฉพาะ ใช้สำหรับปรับค่าพารามิเตอร์แบบใดแบบหนึ่งโดยเฉพาะ แต่สำหรับ การแปลงค่าโดยใช้แนวทางจากแบบจำลอง จะใช้ ฮิวริสติก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่สามารถทำงานโดยใช้ หลักหน้าที่ทั่วไป โดยไม่ขึ้นกับชนิดค่าที่ทำการแปลง แต่จะขึ้นอยู่กับหลักการความรู้ที่อ้างเหตุผล

6.1.3 การปรับเปลี่ยนโดยวิธีเฉพาะ (Special-Purpose Adaptation and Repair ฮิวริสติก)

จากที่ได้กล่าวถึงวิธีการดัดแปลงทั้งหมด เป็นแบบหน้าที่ทั่วไป คือไม่เจาะจงขอบเขตการทำงาน นั่นคือเป็นอิสระต่อขอบเขต ถึงแม้จะขึ้นอยู่กับความรู้ที่จำกัดขอบเขตก็ตาม แต่ประสบการณ์ในด้าน AI ได้แสดงให้เห็นว่า การใช้ความรู้ในรูปทั่วไปเป็นวิธีที่ไม่ดีเท่าไรนัก ถึงแม้จะเป็นที่ยอมรับกันอย่างกว้างขวาง ความรู้และหลักการเฉพาะ ย่อมจะให้ผลลัพธ์ที่ดีกว่า การดัดแปลงก็เช่นเดียวกัน ถ้าเป็นไปได้ การใช้ความรู้เฉพาะทาง หรือ ฮิวริสติก ในการดัดแปลงย่อมจะให้ผลลัพธ์ที่ดีกว่า ซึ่งจากหลักการนี้ทำให้เราสามารถแบ่งประเภทของการดัดแปลงได้เป็น 3 ประเภทคือ

- การดัดแปลงโดยใช้ขอบเขตความรู้เฉพาะ หมายถึง การใช้กฎของการแทนค่า หรือการปรับเปลี่ยนที่เฉพาะกับขอบเขตความรู้ หรือขอบเขตของปัญหา
- การปรับเปลี่ยนโครงสร้าง ที่กล่าวมาแล้วคือ การแปลงค่าโดยใช้แนวทางจากแบบจำลอง ซึ่งใช้กฎทั่วไปในการปรับโครงสร้าง แต่ส่วนใหญ่การปรับโครงสร้างมักจะใช้ ความรู้เฉพาะทาง
- การดัดแปลงโดยใช้หลักการทั่วไป

ในการใช้งานการดัดแปลงด้วยวิธีการเฉพาะ บางครั้งมีความซับซ้อนมาก ในการจัดการเลือกใช้ฮิวริสติก ในสถานการณ์ที่เหมาะสม ซึ่งจะต้องมีการจัดการโดยใช้ลักษณะที่ ฮิวริสติก ที่เป็นประโยชน์นั้นเป็นตัวเชื่อมโยงสถานการณ์กับตัว ฮิวริสติก

6.1.4 ดีริเวชันรีเพลย์(Derivation Replay)

ดีริเวชันรีเพลย์ เป็นการดัดแปลงผลลัพธ์ โดยทำการสร้างผลลัพธ์ขึ้นมาใหม่ โดยไม่ได้ทำการแทนค่า หรือเปลี่ยนแปลงรูปแบบของตัวผลลัพธ์ แต่จะทำการสร้างผลลัพธ์ขึ้นมาใหม่ ด้วยวิธีที่ใช้ในการสร้างผลลัพธ์ตัวเก่า โดยไม่ได้ใช้ผลลัพธ์ตัวเก่าโดยตรง

ในการจะใช้วิธีนี้ จำเป็นที่จะต้องมีการบันทึกข้อมูลเกี่ยวกับตัวผลลัพธ์เพิ่มขึ้น คือ วิธีการอ้างเหตุผล หรือวิธีการสร้างผลลัพธ์ รวมไปถึงเหตุผลว่า ทำไมวิธีนี้จึงเหมาะสมในการหาผลลัพธ์ในสถานการณ์นี้

6.2 การควบคุมการดัดแปลง (Controlling Adaptation)

ในการจัดการควบคุมกระบวนการดัดแปลงผลลัพธ์ จะต้องพิจารณาสิ่งที่สำคัญ 3 สิ่งคือ

- ตัวอ้างเหตุผลจะรู้ได้อย่างไรว่าจะดัดแปลงอะไร
- ตัวอ้างเหตุผลจะเลือกใช้วิธีในการดัดแปลงได้อย่างไร
- มีอะไรบ้างที่จะเป็นแนวทาง ที่นำไปสู่การดัดแปลงที่ถูกวิธี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2.1 การระบุสิ่งที่จะต้องทำการดัดแปลง

เราสามารถระบุถึงส่วนที่จำเป็นต้องทำการดัดแปลง ได้ด้วยวิธีต่างๆ 6 วิธี คือ

6.2.1.1 ใช้ความต่างระหว่างรายละเอียดของปัญหาเก่า กับปัญหาใหม่ โดยแบ่งเป็นขั้นตอนย่อย 2 ขั้นตอน คือ

- หาความแตกต่างระหว่างกรณีเก่า กับสถานการณ์ใหม่
- ระบุความเกี่ยวข้องระหว่างคำอธิบายปัญหา และผลลัพธ์ ซึ่งจะต้องพิจารณา 2 ส่วนที่สำคัญ คือ วิธีการอธิบายสถานการณ์ กับวิธีการอ้างถึงของผลลัพธ์ และความสัมพันธ์ระหว่างสถานการณ์เก่ากับผลลัพธ์ที่ได้มา

6.2.1.2 ใช้รายการที่ระบุปัญหามาตรฐาน (Standard Problem Checklist) ซึ่งถ้าเกิดปัญหาตามรายการแล้ว จะต้องทำการดัดแปลง ใช้ในการแก้ปัญหาที่อาจเกิดขึ้นจากวิธีแรก 2 ประการ ประการแรก ถ้าคำอธิบายของกรณีมีขนาดใหญ่ จะทำให้สิ้นเปลืองเป็นอย่างมากในการประมวลผลความแตกต่างของลักษณะทั้งหมดของกรณี กับสถานการณ์ใหม่ ประการที่สองคือ เมื่อลักษณะสำคัญที่นำไปสู่การดัดแปลงต้องใช้การอ้างเหตุผลที่ซับซ้อน ซึ่งความแตกต่างของลักษณะภายนอก ไม่สามารถนำไปสู่การดัดแปลงที่ถูกต้องเหมาะสมได้

โดยในรายการปัญหามาตรฐาน จะรวบรวมวิธีการตรวจสอบปัญหาที่จะนำไปสู่การดัดแปลง รวมทั้งวิธีการดัดแปลงที่เหมาะสมกับปัญหาที่เกิดขึ้นในแต่ละแบบด้วย

6.2.1.3 ใช้ความขัดแย้งกันระหว่างผลลัพธ์ที่ได้ขณะนั้น กับเป้าหมาย และข้อบังคับของปัญหาใหม่ แล้วทำการดัดแปลงเพื่อลบข้อขัดแย้งนั้น มีความซับซ้อนที่ควรจะทำการพิจารณาคือ เมื่อเกิดกรณีที่ผลลัพธ์ขัดแย้งกับข้อห้าม จะต้องทำการดัดแปลงผลลัพธ์ ซึ่งมีทางเลือกหลายทาง โดยทั่วไปแล้วจะเลือกพิจารณาการแทนค่าก่อนการเปลี่ยนแปลงรูปแบบ

6.2.1.4 ใช้การโปรเจ็กชั่น (Projection) ผลลัพธ์เพื่อระบุถึงปัญหาที่ต้องได้รับการแก้ไข เสมือนเป็นการนำผลลัพธ์นั้นไปลองใช้ แล้วพิจารณาผลที่ได้จากผลลัพธ์นั้น โดยจะต้องมีระบบที่สามารถตรวจสอบผลลัพธ์ได้ มารองรับการทำงานนี้ ซึ่งสามารถใช้วิธีในการทำโปรเจ็กชั่นได้ต่างๆ กัน ดังนี้

- โมเดล-เบสโปรเจคชัน(Model-Based Projection) เป็นการใช้โมเดล-เบส มาตรวจสอบผลลัพธ์ โดยเป็นการนำผลลัพธ์มาทำการจำลอง ในโมเดลนี้ เพื่อตรวจดูผลที่ได้ว่าตรงตามที่ต้องการหรือไม่
- เคสเบสโปรเจ็คชัน(Case-Based Projection) เป็นการใช้เคส-เบสในการทำโปรเจ็คชัน มักใช้ในขอบเขตของปัญหาที่ไม่สามารถสร้างโมเดลขึ้นมาทดสอบผลลัพธ์ได้ โดยนำผลลัพธ์ไปเปรียบเทียบกับกรณีที่ใกล้เคียงกับผลลัพธ์ เพื่อทดสอบว่าผลลัพธ์นั้นทำงานได้ตามต้องการหรือไม่
- ซิมูเลทเบสโปรเจ็คชัน(Simulation-Based Projection) เป็นการนำผลลัพธ์มาทำการจำลอง เพื่อทดสอบความถูกต้องของผลลัพธ์ โดยจะต่างกับ 2 วิธีที่กล่าวมา โมเดล และเคสจะสามารถแสดงให้เห็นถึงผลของการใช้ผลลัพธ์ได้ แต่ซิมูเลชั่น จะแสดงถึงข้อผิดพลาดของผลลัพธ์โดยเฉพาะ

6.2.1.5 ใช้ผลจากการนำผลลัพธ์ไปใช้งานจริง แล้วจึงนำค่าป้อนกลับ (Feedback) มาวิเคราะห์หาส่วนที่จะต้องทำการดัดแปลง วิธีนี้จะคล้ายกันกับการทำโปรเจ็คชัน แต่จุดต่างที่สำคัญคือ วิธีนี้จะส่งผล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลัพธ์ออกไปยังผู้ใช้ เพื่อให้ผู้ใช้ป้อนกลับผลจากการใช้ผลลัพธ์ ดังนั้นหากเกิดข้อผิดพลาดในผลลัพธ์นั้น จะไม่อาจย้อนกลับมาแก้ไขได้

6.2.1.6 ใช้ผลจากการดัดแปลงครั้งก่อน เพื่อระบุส่วนที่ต้องทำการแก้ไข ในการดัดแปลงแต่ละครั้งย่อมจะมีความแตกต่างกัน ดังนั้นย่อมจะเกิดข้อแก้ไขในการดัดแปลงแต่ละครั้ง เช่น ความไม่พอดีของการดัดแปลง วิธีนี้จะเป็นการนำความผิดพลาดเนื่องจากความไม่พอดีในอดีต มาเป็นแนวทางในการเลือกสิ่งที่จะต้องทำการดัดแปลงที่จะเกิดขึ้นในสถานการณ์ใหม่

6.2.2 การเลือกวิธีการดัดแปลง

เมื่อมีการกำหนดว่าจะต้องมีการทำการดัดแปลงแล้ว ขั้นตอนต่อมาก็คือการเลือกวิธีที่จะนำมาใช้ในการดัดแปลง ซึ่งจะต้องพิจารณาปัจจัยต่างๆ ต่อไปนี้

6.2.2.1 ระบุให้แน่นอนว่าสิ่งใดที่จะต้องทำการดัดแปลง เนื่องจากในการประเมินสิ่งที่ต้องการการดัดแปลง อาจมีหลายสิ่งที่ต้องการการดัดแปลง ซึ่งจะต้องเลือกว่าสิ่งที่เราต้องการจะดัดแปลงจริงๆ คืออะไรก่อนที่จะทำการเลือกวิธีดัดแปลง โดยทั่วไปแล้วจะเลือกทำการดัดแปลงด้วยวิธีที่ง่ายที่สุด นั่นคือเลือกการแทนค่า แทนที่จะเลือกการปรับเปลี่ยนรูปแบบ วิ่งก็เหมาะเฉพาะกับกรณีที่ผลลัพธ์เก่ามีโครงสร้างแบบเดียวกับผลลัพธ์ที่ต้องการ มิฉะนั้นจะต้องใช้การปรับเปลี่ยนโครงสร้าง

จะเห็นได้ว่าเราต้องการที่ทำการดัดแปลงด้วยวิธีที่ง่ายที่สุดก่อน ดังนั้นในการเลือกว่าจะดัดแปลงส่วนประกอบตัวไหนนั้น เราต้องคำนึงถึงวิธีที่เราจะนำมาใช้ในการดัดแปลงด้วย จึงต้องพิจารณาคุณสมบัติของส่วนที่เราจะทำการดัดแปลง ดังนี้

- การติดต่อกันกับส่วนอื่นๆ ที่อยู่ในผลลัพธ์ ยิ่งมีความสัมพันธ์กับส่วนประกอบอื่นมาก ก็ยิ่งจะทำให้วิธีการในการดัดแปลงยิ่งยากขึ้น
- ความเป็นศูนย์กลางในผลลัพธ์ ถ้าส่วนที่เราจะดัดแปลงมีความเป็นศูนย์กลางมาก ก็จะยิ่งทำให้การดัดแปลงทำได้ยาก และหาวิธีที่เหมาะสมได้ยากขึ้น เพราะส่วนที่เป็นศูนย์กลางของผลลัพธ์นั้น จะหาค่ามาแทนได้ยาก ซึ่งทำให้ต้องใช้วิธีการปรับเปลี่ยนรูปร่าง ถ้าการปรับเปลี่ยนไม่สามารถทำให้ผลลัพธ์นั้นตรงกับความต้องการของปัญหา ก็จะต้องทำการแทนค่าทั้งหมดของผลลัพธ์นั้น เพราะง่ายและเหมาะสมกว่าการหาค่ามาแทนในส่วนศูนย์กลาง
- ความใกล้เคียงของค่าที่สามารถนำมาแทน สามารถช่วยให้ความยากในการดัดแปลงเนื่องจากปัจจัยทั้งสองข้อแรกที่ได้กล่าวมา บรรเทาลงได้ เพราะเราสามารถหาค่าที่ใกล้เคียงสามารถนำมาใช้ในวิธีการแทนค่าได้

6.2.2.2 หาวิธีการดัดแปลงหรือแก้ไขที่เหมาะสม ซึ่งมี 3 แนวทางดังนี้

- บางวิธีที่ใช้ในการเลือกสิ่งที่จะทำการดัดแปลง จะเกี่ยวข้องอยู่กับการเลือกวิธีการดัดแปลงอยู่แล้ว เช่น การใช้รายการในการเลือกส่วนประกอบที่จะทำการดัดแปลง
- บางค่าที่จะทำการดัดแปลง กำหนดวิธีที่จะเหมาะสมที่จะใช้ในการดัดแปลงตัวมันอยู่แล้ว เช่น บางค่ามีค่าใกล้เคียงที่สามารถนำมาใช้แทนค่าได้ จึงเหมาะจะใช้กับวิธีการแทนค่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การดัดแปลงด้วยวิธีทาง ฮิวริสติก บางวิธี จะมีดัชนีที่จะใช้ในการเลือกใช้วิธีเหล่านี้ให้ตรงตามสถานการณ์อยู่แล้ว

6.2.2.3 เลือกวิธีการจากกลุ่มของวิธีการดัดแปลงที่เหมาะสม พิจารณาโดยใช้หลักการต่อไปนี้

- พิจารณาวิธีที่ทำหน้าที่เฉพาะเจาะจงก่อน วิธีที่ทำหน้าที่ทั่วไป
- พิจารณาวิธีที่ง่ายก่อน วิธีที่ยาก
- ถ้าปัจจัยอื่นๆ ไม่ต่างกัน ให้พิจารณาวิธีที่เคยประสบความสำเร็จมาก่อน
- ให้เปลี่ยนไปใช้วิธีที่ทำหน้าที่ทั่วไป และวิธีที่ซับซ้อน หลังจากล้มเหลวในวิธีที่ทำหน้าที่เฉพาะเจาะจง และง่าย
- ให้ใช้วิธีที่มีแนวโน้มจะล้มเหลวได้ ในกรณีที่ผลลัพธ์ที่ได้จากวิธีที่มีประสิทธิภาพ ได้มาด้วยวิธีที่ยาวนานกว่าวิธีอื่นมากๆ โดยจะต้องมีโอกาสล้มเหลวต่ำ และมีโอกาสที่จะตรวจค้นความผิดพลาดได้โดยไม่ยาก หรือก่อนที่จะส่งผลลัพธ์ให้แก่ผู้ใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 7
การออกแบบไลบรารีของกรณีและโครงสร้างข้อมูล

7.1 หลักการทำดัชนี

การทำดัชนีนั้นทำได้โดยกำหนดทั้งชื่อของโรค, อาการของโรค, ลักษณะของอาการ และวิธีการรักษารวมถึงคำแนะนำต่างๆไว้เป็นสัญลักษณ์ทางรหัสตัวเลขเพื่อใช้ในการจับคู่ และเป็นข้อความในส่วนของคำแนะนำ โดยจะแยกเก็บเป็นไฟล์โดยแบ่งออกเป็น 3 ไฟล์คือ

1. ดัชนีของกรณีทั้งหมดที่เกิดขึ้น โดยจะเป็นส่วนที่จะนำมาเปรียบเทียบกับกรณีใหม่ที่เกิดขึ้นเพื่อทำการวินิจฉัยโดยจะมีรายละเอียดดังนี้

รหัสของโรคที่ได้วินิจฉัย...
กลุ่มอาการของโรคกลุ่มที่1
อาการที่1...
ลักษณะของอาการที่1...
อาการที่2...
ลักษณะของอาการที่2...
...
อาการที่8...
ลักษณะของอาการที่8...

กลุ่มอาการของโรคกลุ่มที่2
...
กลุ่มอาการของโรคกลุ่มที่...

โดยรายละเอียดอธิบายได้ดังนี้

- รหัสโรคจะเป็นตัวเลขที่เชื่อมกับดัชนีของชื่อโรคและคำแนะนำ
- กลุ่มอาการของโรคโดยจะเก็บอาการเดียวกันไว้ด้วยกัน
- อาการที่อยู่ในกลุ่มจะเก็บไว้เป็นรหัสเฉพาะที่แยกจากส่วนอื่น
- ลักษณะของอาการจะเก็บค่าเพื่อบ่งบอกถึงขนาดความหนัก-เบาหรือความรุนแรงของอาการ

2. ดัชนีของชื่อโรคและคำแนะนำในการรักษา ซึ่งจะทำการเก็บรหัสของชื่อโรคไว้เมื่อเปรียบเทียบกับรหัสที่ได้จากการเปรียบเทียบจากกรณีโดยจะแบ่งรายละเอียดได้ดังนี้

รหัสของชื่อโรค...

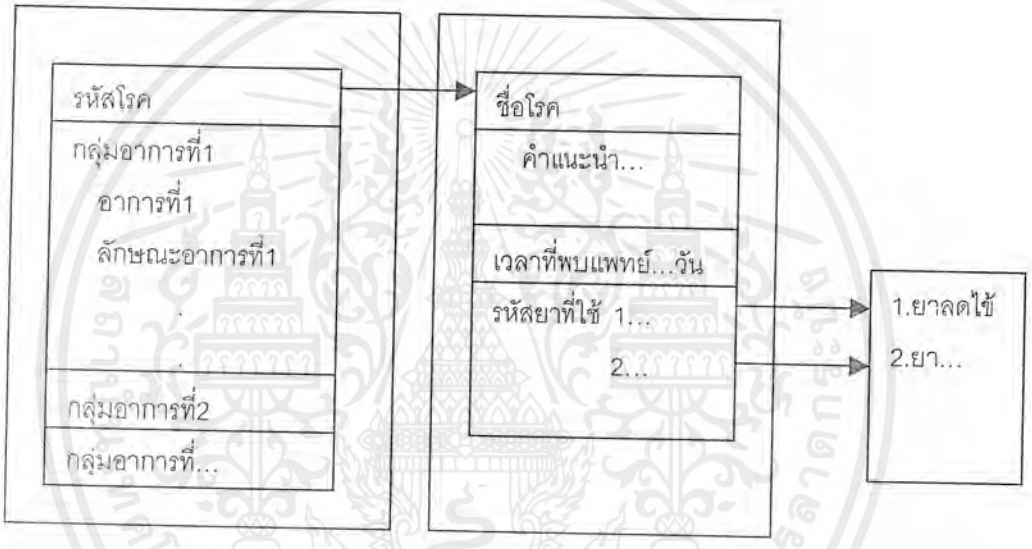
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำแนะนำ...
เวลาที่สมควรพบแพทย์...
รหัสยาที่ใช้รักษา...

ซึ่งแต่ละส่วนสามารถอธิบายได้ดังนี้

- รหัสของชื่อโรคจะเป็นรหัสตัวเดียวกับดัชนีของกรณีทั้งหมด
- คำแนะนำจะเก็บรายละเอียดวิธีการรักษาโรคนั้นๆ
- รหัสยาที่ใช้รักษาโรคก็จะเก็บรหัสยาที่ใช้รักษาในแต่ละโรค

3. ดัชนีของชื่อยาโดยเก็บชื่อยาเอาไว้



รูปที่ 7.1 โครงสร้างของไลบรารีของกรณี

โดยเขียนเป็นแผนภาพได้ดังรูปที่ 7.1

การที่ออกแบบโครงสร้างแบบนี้เพราะ

- อาจจะมีโอกาสที่หลายกรณีสามารถที่เป็นโรคเดียวกันได้ เช่น ไทฟอยด์อาจจะเกิดขึ้นได้จากการมีอาการเพียงมีไข้นานกว่า 7 วัน หรืออาจเกิดจากมีอาการตับหรือม้ามโตหรือมีการระบาดของไทฟอยด์ก็ได้จึงได้มีการแยกกรณีกับส่วนของชื่อโรคและคำแนะนำเพื่อลดความซ้ำซ้อน
- เพื่อใช้ในการปรับปรุงเมื่อการจับคู่ไม่แม่นตรง(ค่าความแน่นอนไม่เป็น 1) จึงต้องใช้คำแนะนำของทั้ง 2 โรคขึ้นไปในการแนะนำ เช่น มีอาการเป็นโรคปอดอักเสบและ เป็นไข้ 2 อาการร่วมกัน แล้วจะทำให้มีค่าในการจับคู่สูงขึ้น (ค่าความแน่นอนเข้าใกล้ 1) ก็ใช้คำแนะนำของทั้งสองโรคในการแนะนำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- บางครั้งอาการของโรคบางชนิดอาจมีหลายอาการร่วมกันโดยสามารถแบ่งเป็นกลุ่มของอาการได้ จึงได้แบ่งเป็นกลุ่มอาการ โดยจะสามารถวินิจฉัยโดยถ้าได้พบอาการใดอาการหนึ่งในกลุ่มอาการเพียงอาการเดียวหรือมากกว่านั้น ก็สรุปได้ว่าเกิดกลุ่มอาการนั้น

7.2 การออกแบบโครงสร้างข้อมูล และอัลกอริธึมสำหรับไลบรารีของกรณี

จากที่ได้กล่าวถึงการออกแบบโครงสร้างของไลบรารีของกรณีในแบบนามธรรม ต่อไปจะกล่าวถึงการนำโครงสร้างที่ได้ไปออกแบบเป็นโครงสร้างข้อมูลที่ใช้จริงในระบบ และพิจารณาอัลกอริธึมที่ใช้สำหรับไลบรารีของกรณี

เนื่องจากในระบบนี้ทำการพัฒนาด้วย บอร์แลนด์ เดลไฟ (Borland Delphi) ซึ่งเป็นมีพื้นฐานอยู่บนภาษาพาสคัล จึงขออธิบายในส่วนโครงสร้างข้อมูล และอัลกอริธึมด้วยโค้ดของภาษาพาสคัลเป็นหลัก

พิจารณาถึงประเด็นแรกของการออกแบบไลบรารีของกรณี จะเห็นได้ว่า ถึงแม้จะสามารถออกแบบให้ทุกกรณีมีโครงสร้างที่เหมือนกันได้ แต่สิ่งหนึ่งที่ทำให้ไลบรารีของกรณีต่างไปจากฐานข้อมูลทั่วไป คือ ขนาดที่ไม่เท่ากันในแต่ละกรณี เพราะลักษณะหรือฟีเจอร์ของกรณีมีขนาดไม่แน่นอน บางกรณีมาก บางกรณีน้อย แตกต่างกันไป รวมทั้งโครงสร้างภายในของลักษณะของกรณีด้วย ทำให้เราต้องพิจารณาเฉพาะส่วนของลักษณะของกรณีเป็นแบบไดนามิก (Dynamic)

ลักษณะของกรณีเกิดจากการกลุ่มของลักษณะย่อยมากระการทางลอจิกแอนด์กัน ส่วนภายในกลุ่มของลักษณะย่อยก็เกิดจากการกระทำการทางลอจิกออร์ของลักษณะย่อย ซึ่งตามหลักแล้ว ไม่สามารถจำกัดได้ว่าในแต่ละกลุ่มควรจะมีลักษณะกี่ตัว และเช่นกัน ไม่สามารถจำกัดได้ว่าในลักษณะของกรณีควรจะมีกลุ่มของลักษณะย่อยกี่ตัว ดังนั้นทางเลือกที่เหมาะสมในการเลือกใช้ชนิดของข้อมูล คือ การเลือกใช้ข้อมูลที่เป็นไดนามิก นั่นคือการใช้พอยท์เตอร์นั่นเอง

พิจารณาการออกแบบเริ่มต้นที่กรณีหนึ่งประกอบด้วย 2 ส่วนสำคัญ คือ ผลลัพธ์ นั่นก็คือรหัสของโรค และลักษณะของกรณี ก็คือกลุ่มอาการ สามารถเขียนเป็นโครงสร้างข้อมูลได้ดังนี้

```
TYPE
    CaseRecord = record
                   SolutionNo : Word;
                   FeatureGroup : AndPtr;
                   end;
```

รหัสของโรคเลือกใช้ชนิดข้อมูลเป็นเวิร์ด (2 ไบต์) ส่วน ลักษณะอาการจะต้องเป็นกลุ่มของอาการ ในที่นี้จะใช้เป็นพอยท์เตอร์ชี้ไปยังกลุ่มของอาการ

พิจารณาย่อยลงไปในลักษณะของกรณี จะต้องประกอบด้วยกลุ่มของลักษณะย่อยมากระทำการลอจิกแอนด์กัน ออกแบบโครงสร้างข้อมูลได้ดังนี้

```
TYPE
    AndPtr = ^SubGroup;
    SubGroup = record
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
OrGroup: OrPtr;
NextAnd : AndPtr;
end;
```

จะเห็นว่าจากตัวกรณี ฟีเจอร์กรุ๊ป คือพอยท์เตอร์ที่ชี้ไปยัง เรคอร์ดของกลุ่มของลักษณะย่อย ภายในกลุ่มลักษณะย่อยก็ประกอบด้วยข้อมูล 2 ตัว ตัวแรกคือ ออร์กรุ๊ป ซึ่งกำหนดให้เป็นกลุ่มของลักษณะย่อย ซึ่งภายในเป็นกลุ่มของลักษณะที่กระทำการลอจิกออร์กัน ข้อมูลตัวที่สองคือ เนกซ์แอนด์ ซึ่งก็คือ กลุ่มลักษณะอาการย่อยอื่นที่จะมากระทำลอจิกแอนด์ด้วย ซึ่งชนิดข้อมูลเป็น แอนด์พอยท์เตอร์ ซึ่งเป็นพอยท์เตอร์ที่ชี้มายังข้อมูลประเภท ซับเรคอร์ด ซึ่งเป็นกลุ่มของลักษณะย่อย

จะเห็นได้ว่าการสร้างลักษณะของกรณี ด้วยการทำลอจิกแอนด์ของกลุ่มลักษณะย่อย โดยการใช้พอยท์เตอร์เชื่อมระหว่างกลุ่มอาการย่อยนั้น เหมือนกับการสร้างลิงค์ลิสท์ของกลุ่มอาการย่อยนั่นเอง โดยมีเฮดเดอร์อยู่ที่เรคอรด์ของกรณี (เคสเรคอร์ด)

พิจารณาต่อไปลึกลงไปในระดับของลักษณะย่อยที่กระทำลอจิกออร์กันอยู่ภายในกลุ่มลักษณะย่อย ออกแบบโครงสร้างข้อมูลดังนี้

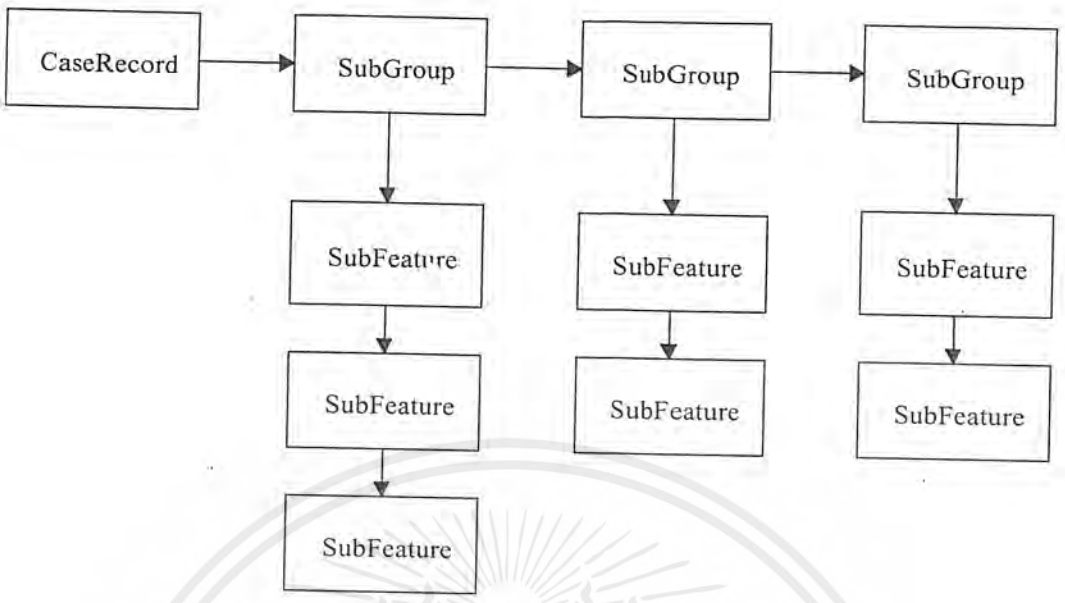
```
TYPE
    OrPtr = ^SubFeature;
    SubFeature = record
        Feat : Word;
        Attr : byte;
        NextOr : OrPtr;
        end;
```

ลักษณะย่อยแทนด้วยเรคอร์ด ซับฟีเจอร์ โดยภายในจะต้องเก็บค่าที่สำคัญที่เกี่ยวกับลักษณะนั้น นั่นก็คือ รหัสของลักษณะ (ฟีเจอร์) และค่าของลักษณะนั้น (แอททริบิวต์) นอกจากนั้นจะต้องมีส่วนที่เชื่อมโยงไปยังลักษณย่อยอื่นที่กระทำการลอจิกออร์ด้วย จึงแทนด้วยพอยท์เตอร์ที่ชี้ไปยังเรคอร์ดที่เป็นลักษณะย่อยเหมือนกัน (เนกซ์ออร์)

พิจารณาที่กลุ่มลักษณะย่อยอีกครั้งหนึ่ง จะเห็นว่าข้อมูลภายในที่เป็น กลุ่มของลักษณะย่อย (ออร์กรุ๊ป) จะใช้พอย์เตอร์ชี้มายัง เรคอร์ดซับฟีเจอร์ ซึ่งเป็นลักษณะย่อย จะเห็นได้ว่าการสร้างกลุ่มลักษณะย่อย ด้วยการออร์กันของลักษณะย่อย โดยใช้พอยท์เตอร์เป็นตัวเชื่อมต่อลักษณะย่อย เสมือนเป็นการสร้างลิงค์ลิสท์ของลักษณะย่อย โดยมีเฮดเดอร์ อยู่ที่กลุ่มของลักษณะย่อย (ซับกรุ๊ป)

ย้อนกลับไปพิจารณาที่ระดับของตัวกรณีอีกครั้ง จะเห็นได้ว่ามีการใช้งานพอยท์เตอร์ 2 ชนิด ชนิดแรกเป็นการสร้างลิสท์ของกลุ่มลักษณะย่อย หรือกระทำลอจิกแอนด์กัน ชนิดที่สองเป็นการสร้างลิสท์ของลักษณะย่อย เพื่อสร้างเป็นกลุ่มชองลักษณะย่อย หรือกระทำลอจิกออร์กัน ซึ่งถ้ามองในระดับของกรณีจะเห็นว่ามีลักษณะคล้ายการสร้างทรีด้วยพอยท์เตอร์ ดังรูปที่ 7.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7.2 โครงสร้างทางข้อมูลโดยรวมของกรณี

สรุป โครงสร้างข้อมูลของกรณีได้ดังนี้

```
TYPE
    OrPtr = ^SubFeature;
    SubFeature = record
            Feat : Word;
            Attr : byte;
            NextOr : OrPtr;
            end;

    AndPtr = ^SubGroup;
    SubGroup = record
            OrGroup: OrPtr;
            NextAnd : AndPtr;
            end;

    CaseRecord = record
            SolutionNo : Word;
            FeatureGroup : AndPtr;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    end;
CaseLibrary = Array[1..200] of CaseRecord;
```

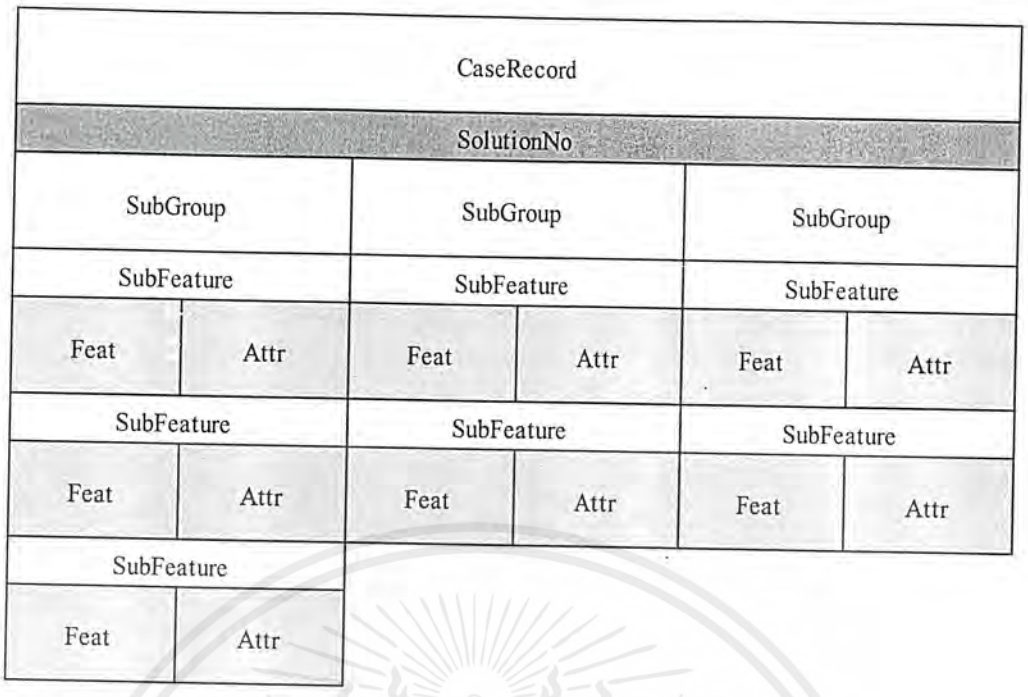
เคสไลบรารี คือชนิดข้อมูลที่ใช้แทนไลบรารีของกรณีโดยจะใช้อาร์เรย์ของเคสเรคอร์ดแทนชนิดข้อมูล การนำไปใช้ก็เพียงแต่กำหนดตัวแปรที่ต้องการ ให้เป็นข้อมูลชนิดเคสไลบรารีเท่านั้น

นอกจากจะต้องพิจารณาไลบรารีของกรณีซึ่งอยู่ในหน่วยความจำขณะทำงานแล้ว ยังต้องออกแบบไลบรารีของกรณีในรูปของไฟล์ด้วย เพราะไลบรารีของกรณีจะต้องอ่านขึ้นมาจากไฟล์เพื่อนำมาใช้ในระบบ หลังจากที่มีการเปลี่ยนแปลงไลบรารีของกรณี ก็จะต้องเขียนไลบรารีของกรณีที่แก้ไขใหม่นั้นกลับลงไปในไฟล์ด้วย

จากที่ได้กล่าวมาแล้วว่าโครงสร้างของกรณีที่ออกแบบนั้นแน่นอน แต่มีขนาดไม่แน่นอน ดังนั้นการจะเขียนลงในไฟล์ ไม่สามารถใช้ข้อมูลประเภทไทพ์ไฟล์ของพาสคัลได้ จำเป็นที่จะต้องใช้ไฟล์แบบอันไทพ์ไฟล์ ซึ่งเป็นการเขียนข้อมูลลงในไฟล์แบบเป็นบล็อก บล็อกละ 128 ไบต์ แต่ข้อมูลในไลบรารีของกรณีที่อยู่ในหน่วยความจำขณะทำงานของระบบ ยังไม่ได้เป็นข้อมูลประเภทไบท์ ดังนั้นก่อนที่เขียนไลบรารีของกรณีลงในไฟล์ จะต้องทำการเปลี่ยนข้อมูลเหล่านี้ให้อยู่ในรูปไบท์ และเก็บอยู่ในบัฟเฟอร์ ซึ่งเป็นอาร์เรย์ของไบท์ก่อน

ในการแปลงข้อมูลจากโครงสร้างเดิมซึ่งมีลักษณะคล้ายทรี ซึ่งใช้พอยท์เตอร์ในการสร้าง จะต้องเปลี่ยนโครงสร้างให้เป็นแบบไม่ใช่พอยท์เตอร์ แต่ให้มีลักษณะคล้ายลิสท์ที่สร้างด้วยอาร์เรย์ ซึ่งซ้อนกัน 2 ชั้นแทน ดังรูปที่ 7.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่7.3 แสดงโครงสร้างของไลบรารีของกรณี โดยพิจารณาลักษณะของกรณีเป็นอาร์เรย์ซ้อนกัน 2 ชั้น

สิ่งที่สำคัญที่จะต้องเพิ่มขึ้นมาคือการออกแบบโครงสร้างของไฟล์ ให้สามารถดำรงไว้ซึ่งลักษณะของไลบรารีของกรณีแบบเดียวกับที่ไลบรารีกรณีในหน่วยความจำขณะทำงานนั้นทำได้ นั่นก็คือ ความยืดหยุ่นของขนาดของแต่ละกรณี หรือจะกล่าวให้ชัดเจนก็คือ จำนวนกลุ่มของลักษณะย่อย และจำนวนลักษณะย่อยภายในกลุ่มลักษณะย่อยที่สามารถเปลี่ยนแปลงได้ ดังนั้นเราจึงต้องแปลงรูปแบบข้อมูลจากแบบเดิมซึ่งทำงานบนหน่วยความจำขณะทำงาน ซึ่งมีชนิดข้อมูลหลายแบบ (เวิร์ด,ไบต์,พอยท์เตอร์) มาเป็นข้อมูลในลักษณะไบต์เพียงชนิดเดียว พิจารณาเปรียบเทียบจากส่วนลักษณะย่อย ขึ้นไปหาตัวกรณี และไลบรารีของกรณี

ส่วนลักษณะย่อยจะประกอบด้วย 2 ส่วน คือ ฟีเจอร์ และแอททริบิวต์ ส่วนฟีเจอร์เป็นข้อมูลชนิดเวิร์ด มีขนาด 2 ไบต์ ต้องแปลงเป็นข้อมูลชนิดไบต์ 2 ตัว โดยใช้การดิฟ และม็อดเข้าช่วย ส่วนตัวแอททริบิวท์เป็นข้อมูลชนิดไบต์อยู่แล้วจึงไม่ต้องทำอะไร (จากที่ได้กล่าวมาแล้ว ข้อมูลพอยท์เตอร์ทั้งหมดจะต้องถูกเปลี่ยนรูปให้เป็นอินเด็กซ์ของอาร์เรย์ ดังนั้นจะไม่ขอกล่าวถึงพอยท์เตอร์อีก) รวมทั้งหมดในส่วนลักษณะย่อยต้องใช้ข้อมูลขนาด 3 ไบต์

```
VAR
        A : SubFeature;
        Buffer : Array[1..3] of byte;
        ...
Begin
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
...
Buffer[1] := A.Feat div 255;
Buffer[2] := A.Feat mod 255;
Buffer[3] := A.Attr;
...
End;
```

ต่อไปพิจารณาในระดับกลุ่มของลักษณะย่อย ซึ่งก็คืออาร์เรย์ของลักษณะย่อยนั่นเอง ข้อมูลภายในส่วนนี้ก็จะเป็นลักษณะย่อยทั้งหมด และขนาดของกลุ่มลักษณะย่อยนี้ หรือจำนวนลักษณะย่อยภายในกลุ่มนั่นเอง ซึ่งค่าที่ได้นำมาจากการวนลูปเพื่อนับจำนวนลักษณะภายในกลุ่ม

```
VAR
        B : SubGroup;
        Buffer : Array[1..max] of byte;
        BuffCounter : byte;
        OrCountPtr : Orptr;
        OrCounter : byte;
        ...
Begin
        ...
        OrCountPtr := B.OrGroup;
        OrCounter := 0;
        While OrCountPtr <> Nil do
        Begin
                Inc(OrCounter);
                OrCountPtr := OrCountPtr^.NextOr;
        End;
        BuffCounter := 1;
        Buffer[BufferCounter] := OrCounter
        Inc(BuffCounter);
        ...
        วนลูปแปลงค่าจากลักษณะย่อยแต่ละตัว ตามวิธีที่ได้อธิบายไว้แล้ว
        ...
End;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อไปพิจารณาในระดับของตัวกรณี ซึ่งประกอบด้วยรหัสของผลลัพธ์ (โซลูชันนัมเบอร์) และลักษณะของกรณี ซึ่งเป็นลิสท์ซึ่งเกิดจากการเชื่อมต่อกันของกลุ่มลักษณะย่อย การแปลงในระดับนี้ก็จะคล้ายกับการแปลงในระดับของกลุ่มลักษณะย่อย เพราะมีส่วนประกอบเป็นข้อมูลในลักษณะคล้ายอาร์เรย์เหมือนกัน

```
VAR
        C : CaseRecord;
        Buffer : Array[1..max] of byte;
        BuffCounter : byte;
        AndCountPtr : AndPtr;
        AndCounter : byte;
        ...
Begin
        ...
        BuffCounter := 1;
        Buffer[BuffCounter] := C.SolutionNo div 255;
        Buffer[BuffCounter+1] := C.Solution mod 255;
        BuffCounter := BuffCounter + 2;

        AndCountPtr := C.FeatureGroup;
        AndCounter := 0;
        While AndCountPtr <> Nil do
        Begin
                Inc(AndCounter);
                AndCountPtr := AndCountPtr^.NextAnd;
        End;
        Buffer[BuffCounter] := AndCounter;
        Inc(BuffCounter);
        ...
        ทำการวนลูปแปลงค่ากลุ่มลักษณะย่อยแต่ละตัว ตามวิธีที่ได้กล่าวมาแล้ว
        ...
End;
```

หลังสามารถแปลงข้อมูลของแต่ละกรณีเป็นข้อมูลแบบไบต์ลงในบัฟเฟอร์ได้แล้ว ก็จะเป็นการแปลงข้อมูลในระดับไลบรารีของกรณี ซึ่งสิ่งที่ต้องทำก็มีเพียงระบุจำนวนกรณีที่มีอยู่ทั้งหมดในไลบรารี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ของกรณี ซึ่งเราใช้ชนิดข้อมูลของไลบรารีกรณีเป็นอาร์เรย์ของกรณี (เคสเรคอร์ด) หมายความว่าในการใช้งานไลบรารีของกรณีนี้ เราย่อมรู้ขนาดของไลบรารีนี้อยู่แล้ว เพราะเราต้องมีดัชนีซึ่งอ้างถึงอาร์เรย์นี้ (อาจเป็นจำนวนเต็มชนิดใดชนิดหนึ่ง ซึ่งระบุจำนวนของกรณีที่มีอยู่ในไลบรารีขณะนั้น ในกรณีนี้ใช้ข้อมูลชนิดไบต์) ดังนั้นเราไม่จำเป็นต้องมีการแปลงข้อมูลเพิ่มเติม เพียงแต่ใส่ค่าลงไปในบัฟเฟอร์ได้เลย ดังนั้นเราจะได้บัฟเฟอร์ที่พร้อมจะ

<table>
<tr><td colspan="4">จำนวนกรณีทั้งหมด</td></tr>
<tr><td colspan="2">จำนวนกลุ่มลักษณะย่อย1</td><td colspan="2">จำนวนลักษณะย่อยในกลุ่ม1.1</td></tr>
<tr><td colspan="3">Feat1.1.1</td><td>Attr1.1.1</td></tr>
<tr><td colspan="3">Feat1.1.2</td><td>Attr1.1.2</td></tr>
<tr><td colspan="3">Feat1.1.3</td><td>Attr1.1.3</td><td rowspan="2">จำนวนลักษณะย่อยในกลุ่ม1.2</td></tr>
<tr><td colspan="3">Feat1.2.1</td><td>Attr1.2.1</td></tr>
<tr><td colspan="3">Feat1.2.2</td><td>Attr1.2.2</td><td rowspan="3">จำนวนลักษณะย่อยในกลุ่ม1.3</td></tr>
<tr><td colspan="3">Feat1.3.1</td><td>Attr1.3.1</td></tr>
<tr><td colspan="3">Feat1.3.2</td><td>Attr1.3.2</td></tr>
<tr><td colspan="2">จำนวนกลุ่มลักษณะย่อย2</td><td>จำนวนลักษณะย่อยในกลุ่ม2.1</td><td>...</td><td>...</td></tr>
</table>

รูปที่ 7.4 แสดงลักษณะของไลบรารีกรณีในบัฟเฟอร์ โดยบล็อกความยาวหนึ่งหน่วยแทนข้อมูล 1 ไบต์

เขียนลงในไฟล์ พร้อมขนาดของบัฟเฟอร์ ซึ่งได้มาจากการนับในขณะแปลงข้อมูลและเขียนข้อมูลลงในบัฟเฟอร์ ดังในรูปที่ 7.4

เมื่อได้ข้อมูลในบัฟเฟอร์แล้ว ก็ทำการเปิดไฟล์แบบอันไทพ์ไฟล์แล้วทำการเขียนข้อมูลลงในไฟล์ ขั้นตอนที่ต้องจัดการคือ การนับจำนวนบล็อกที่จะต้องเขียนลงในไฟล์ ซึ่งเราไม่จำเป็นต้องเขียนบัฟเฟอร์ทั้งหมดลงในไฟล์ จำนวนบล็อกที่จะเขียนลงในไฟล์ได้มาจาก การนำจำนวนไบต์มาดิฟด้วย 128 (จำนวนไบต์ต่อบล็อก) แล้วบวกด้วย จำนวนไบต์มอดด้วย 128

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับการอ่านจากไฟล์ขึ้นมาในหน่วยความจำไลบรารีของกรณี ก็ทำเช่นเดียวกับวิธีการเขียนลงในไฟล์ แต่กระทำย้อนกลับ เริ่มต้นด้วยการอ่านไฟล์มาเก็บลงในบัฟเฟอร์ (โดยไม่ต้องระบุจำนวนบล็อกที่จะอ่าน กำหนดจำนวนบล็อกที่จะอ่านให้มากที่สุดเท่าที่คิดว่าไฟล์นั้นจะมีได้ แล้วใช้โพรซีเดอร์ในการอ่านที่ส่งค่ากลับจำนวนบล็อกที่อ่านได้ โพรซีเดอร์นี้จะทำการอ่านไฟล์จนพบ เอนด์ออฟไฟล์ แล้วจะหยุดการทำงานเอง โดยไม่เกิดความผิดพลาด) แล้วทำการเขียนข้อมูลลงในไลบรารีของกรณีในหน่วยความจำ โดยการแปลงข้อมูลย้อนกลับ ในขั้นตอนเหล่านี้จะต้องวนลูปโดยใช้ค่าจำนวนกลุ่มของลักษณะย่อย และจำนวนลักษณะย่อยภายในกลุ่มเป็นตัวนับในการวนลูป นอกจากนั้นยังต้องมีการจองพื้นที่แบบไดนามิกของพอยท์เตอร์ เพื่อสร้างลิงค์ลิสท์ด้วย

ข้อสำคัญในการตรวจสอบการทำงานของอัลกอริธึมเหล่านี้คือ การใช้งานพอยท์เตอร์ ซึ่งต้องทำการจองพื้นที่ทุกครั้งก่อนที่จะใช้งาน และหลังจากใช้งานแล้วจะต้องทำการคืนพื้นที่นั้นให้แก่หน่วยความจำด้วย

ในส่วนของอัลกอริธึมของการสร้างและดัดแปลงไลบรารีของกรณีขอละไว้ เพราะรูปแบบโครงสร้างข้อมูล และขั้นตอนการทำงาน ไม่ต่างไปจากวิธีการอ่านและเขียนไฟล์ไลบรารีของกรณีเท่าไรนัก

7.3 การออกแบบไลบรารีของผลลัพธ์ของกรณี

จากที่ได้กล่าวถึงการออกแบบไลบรารีของกรณี จะแยกส่วนของลักษณะของกรณีออกจากส่วนของผลลัพธ์ โดยการจะเชื่อมต่อระหว่างกรณีไปยังผลลัพธ์จะใช้รหัสของผลลัพธ์เป็นตัวเชื่อมต่อ ทบทวนอีกครั้งหนึ่งถึงโครงสร้างทางนามธรรมของผลลัพธ์

- รหัสของผลลัพธ์
- ชื่อผลลัพธ์ หรือชื่อโรค
- แนวทางในการรักษา
- ระยะเวลาที่รอดูอาการก่อนไปพบแพทย์
- รหัสยาที่ควรใช้ในการรักษา (ไม่เกิน 5 ชนิด)

จากข้อมูลข้างต้นนำมาสร้างเป็นโครงสร้างทางข้อมูลได้ดังนี้

```
TYPE
    SolutionRecord = record
        SolutionNo : Word;
        Name : String[30];
        Suggestion : String[255];
        Day : byte;
        Med : Array[1..5] of byte;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
End;
```

โดยข้อมูลนี้จะถูกนำไปเก็บในไทพ์ไฟล์ ที่มีชนิดของไฟล์เป็น ไฟล์ของเรคอร์ดนี้

```
TYPE
    SolutionFile = File of SolutionRecord;
```

ในการกำหนดค่าให้กับแต่ละผลลัพธ์ จะมี 2 ค่าที่จะต้องใส่คือ โซลูชันนัมบอร์ กับชื่อของโซลูชัน ส่วนค่าอื่นๆ ถ้าไม่ใส่ จะถูกกำหนดให้เป็นค่าปกติของมัน ดังนี้

- คำแนะนำ เป็นสตริงว่าง
- ระยะเวลาดูอาการ มีค่า 0 (ค่า 255 หมายถึงพบแพทย์ด่วน)
- รหัสยา มี่ค่าเป็น 0

ในการใช้งาน เมื่อเริ่มการทำงานของระบบ ก็ทำการอ่านไฟล์ของผลลัพธ์นี้มาเก็บลงในหน่วยความจำ เมื่อมีการอ้างถึงผลลัพธ์จากกรณี ก็ทำการอ่านค่าจากผลลัพธ์ที่ต้องการขึ้นมา โดยไลบรารีของผลลัพธ์ในหน่วยความจำขณะทำงานจะแทนด้วยโครงสร้างข้อมูลแบบอาร์เรย์เช่นเดียวกับไลบรารีของกรณี

```
TYPE
    SolutionLibrary = Array[1..200] of SolutionRecord;
```

ส่วนรหัสยา ก็จะเป็นการเชื่อมต่อระหว่างรหัสยาจากผลลัพธ์ไปยังไลบรารีที่เก็บข้อมูลเกี่ยวกับยานั้นอีกทีหนึ่ง เหตุผลที่ต้องแยกข้อมูลของกรณีและผลลัพธ์ออกจากกัน และแยกเป็นหลายๆ ชั้นก็เพราะว่าในกรณีต่างๆ อาจมีผลลัพธ์ที่เหมือนกัน รวมทั้งในผลลัพธ์ต่างๆ อาจอ้างถึงการใช้ยาตัวเดียวกัน นั่นหมายความว่า ถ้าทำการเก็บข้อมูลทุกอย่างรวมกันในกรณีทั้งหมด ทั้งลักษณะและผลลัพธ์ จะเป็นการเก็บข้อมูลที่ซ้ำซ้อนกัน ซึ่งมีผลเสียนอกจากจะเปลืองเนื้อที่ในการเก็บข้อมูลมากขึ้นโดยเปล่าประโยชน์แล้ว ยังทำให้การปรับปรุงเปลี่ยนแปลงข้อมูลเกี่ยวกับผลลัพธ์ ต้องทำซ้ำซ้อนกันหลายครั้งด้วย

ข้อมูลยา แสดงเป็นโครงสร้างทางข้อมูลได้ดังนี้

```
TYPE
    MedicineRecord = record
                        MedNo : byte;
                        MedName : String[30];
                        End;
    MedicineLibrary = Array[1..100] of MedicineRecord;
    MedicineFile = File of MedicineRecord;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 8
การออกแบบอัลกอริธึม

8.1 การออกแบบอัลกอริธึมสำหรับเรียกค้นไลบรารีของกรณี

จากโครงสร้างข้อมูลของไลบรารีของกรณีเราสามารถวิเคราะห์ และออกแบบอัลกอริธึมสำหรับการค้นหาและจับคู่กรณีที่เหมาะจะนำมาประยุกต์ในสถานการณ์ใหม่ได้ จากหลักการทฤษฎีที่ได้ทำการศึกษามาแล้ว โดยการทำงานจะเป็นแบบการจับคู่และประเมินความเหมาะสมด้วยวิธีเชิงจำนวน ทำให้หลังจากทำการค้นหาจากทั้งไลบรารีของกรณีแล้ว สามารถระบุกรณีซึ่งมีความสอดคล้องกับสถานการณ์ใหม่มากที่สุดได้

ก่อนอื่นเราจะต้องพิจารณาโครงสร้างข้อมูลของสถานการณ์ใหม่ที่จะนำเข้าเป็นตัวเปรียบเทียบในการค้นหาไลบรารีของกรณีก่อน สถานการณ์ใหม่ควรมีลักษณะข้อมูลที่คล้ายคลึงกับลักษณะของกรณีให้มากที่สุด เพื่อที่จะให้การทำงานในการตับคู่เป็นไปอย่างสะดวก และถูกต้อง

```
VAR
    SituationPtr : ^NewSituation;
    NewSituation = record
        Feat : word;
        Attr : byte;
        Next : SituationPtr;
        End;
```

จากโครงสร้างข้อมูลจะเห็นว่า ใช้ลักษณะข้อมูลฟีเจอร์ และแอททริบิวท์เหมือนกับลักษณะในกรณี โครงสร้างของสถานการณ์ใหม่จะต้องมีลักษณะเป็นลิสท์ ในกรณีนี้เลือกวิธีการสร้างเป็นลิงค์ลิสท์ (อาจใช้เป็นอาร์เรย์ได้ แต่โดยทั่วไป ไม่สามารถระบุได้ในขณะทำการพัฒนาระบบว่าสถานการณ์ใหม่ที่เข้ามานั้น จะมีจำนวนเท่าไร ดังเช่นในกรณีนี้ ดังนั้นการใช้ลิงค์ลิสท์จึงเป็นทางออกในการแก้ปัญหาการกำหนดขนาดของอาร์เรย์ได้) ซึ่งเมื่อมีการรับสถานการณ์ใหม่ หรือปัญหาใหม่เข้ามา จะทำการสร้างลิงค์ลิสท์ที่รวบรวมคำอธิบายสถานการณ์นั้น ในที่นี้ก็คือรวบรวมลักษณะหรือฟีเจอร์ของสถานการณ์ เพื่อนำมาใช้ในการค้นหาไลบรารีของกรณี

อ้างอิงถึงโครงสร้างข้อมูลของไลบรารีของกรณีที่กล่าวมาแล้ว การทำงานในการค้นหาและจับคู่ลักษณะหรือฟีเจอร์ทั้งหมด ล้วนใช้พอยท์เตอร์ทั้งนั้น การค้นหาจะเริ่มต้นที่กรณีแรกในไลบรารีของกรณี กำหนดพอยท์เตอร์ที่ใช้ในการค้นหาให้ชี้ไปที่กลุ่มลักษณะของไลบรารีของกรณี (ฟีเจอร์กรุ๊ป ในเคสเรคอร์ด) แล้วกำหนดพอย์เตอร์อีกตัวหนึ่งให้ชี้ไปที่ส่วนหัวของลิสท์ของลักษณะสถานการณ์ใหม่ ในการค้นหาและจับคู่ จะสนใจเฉพาะลักษณะหรือฟีเจอร์ที่เป็นตัวเดียวกัน (ค่า ฟีเจอร์เดียวกัน) เท่านั้น

พิจารณาการค้นหาจากส่วนย่อย ขึ้นไปหาการค้นหาในทั้งไลบรารีของกรณี การค้นหาและจับคู่ในหน่วยที่เล็กที่สุด คือการจับคู่ฟีเจอร์ต่อฟีเจอร์ ในหน่วยของลักษณะย่อย โดยจะต้องเปรียบเทียบฟีเจอร์ในสถานการณ์ กับฟีเจอร์ในลักษณะย่อย หากมีค่าเท่ากัน ก็จะนำค่าในแอททริบิวต์ของทั้ง 2 ตัวมาเปรียบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เทียบกัน การเปรียบเทียบเป็นลักษณะจับคู่ตรงตัว ถ้าค่าแอททริบิวต์เท่ากันระดับความเข้ากันได้ก็จะมีค่าเป็น 1

สรุปเป็นโค้ดได้ ดังนี้

```
VAR
        SitPtr : SituationPtr;
        CaseFeatPtr : OrPtr;
        MatchValue : Real;
        ...
Begin
        ...
        If (SitPtr^.Feat = CaseFeatPtr^.Feat) then
        Begin
                If (SitPtr^.Attr = CaseFeatPtr^.Attr) then
                        MatchValue := 1
                Else
                        MatchValue := 0;
        End;
        ...
End;
```

ต่อมาพิจารณาในระดับการจับคู่ สถานการณ์ทั้งหมดต่อหนึ่งลักษณะย่อยในกรณี จะต้องทำการวนลูปเพื่อตรวจสอบว่าในลักษณะของสถานการณ์ทั้งหมด มีตัวใดที่สามารถจับคู่ได้กับลักษณะย่อยบ้าง (ซึ่งจะมีเพียงตัวเดียว เพราะลักษณะของสถานการณ์ที่รับเข้ามา จะซ้ำซ้อนกันไม่ได้ ตัวอย่างเช่น อาการอาเจียน จะระบุทั้งมีและไม่มีพร้อมกันไม่ได้)

```
VAR
        SitPtr : SituationPtr;
        CaseFeatPtr : OrPtr;
        ...
Begin
        ...
        {วนลูปจับคู่ลักษณะของสถานการณ์ทุกตัวกับ ลักษณะย่อย}
        While SitPtr <> Nil do
        Begin
                จับคู่ฟีเจอร์ที่ SitPtr ชี้อยู่กับ ฟีเจอร์ที่ CaseFeat ชี้อยู่;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                SitPtr := SitPtr^.Next;
            End;
            ...
        End;
```

ลำดับต่อมาพิจารณาการจับคู่ระหว่างลักษณะสถานการณ์ทั้งหมด กับกลุ่มลักษณะย่อยของกรณี ซึ่งโครงสร้างทางตรรกะของกลุ่มลักษณะย่อยฯองกรณีเป็นแบบออร์ นั่นหมายความว่าการจับคู่ในระดับนี้หมายถึง หากมีลักษณะในสถานการณ์ตัวใด สามารถจับคู่ได้กับลักษณะย่อยตัวใดตัวหนึ่งในกลุ่มของลักษณะย่อยนี้ จะถือว่าเกิดการจับคู่ระหว่างกลุ่มลักษณะย่อยกับลักษณะของสถานการณ์ทันที โดยในความเป็นจริง ค่าความเข้ากันได้นี้จะต้องได้มาจาก การจับคู่ระหว่างลักษณะของสถานการณ์กับลักษณะย่อยภายในกลุ่มนี้ ที่ให้ค่าความเข้ากันได้สูงที่สุด นั่นคือพิจารณาการจับคู่ที่ดีที่สุดนั่นเอง

```
VAR
        SitPtr : SituationPtr;
        CaseFeatPtr : OrPtr;
        MatchValue : Real;
        BestMatch : Real;
        ...
Begin
        ...
        {วนลูปจับคู่ทุกลักษณะย่อยในกลุ่มลักษณะย่อย กับลักษณะในสถานการณ์ใหม่}
        BestMatch := 0;
        While (CaseFeatPtr <> Nil) do
        Begin
                วนลูปจับคู่ลักษณะย่อยที่ CaseFeatPtr ชี้อยู่กับลักษณะทุกตัวในสถานการณ์
                ใหม่ตามวิธีที่ได้กล่าวมาแล้ว;
                CaseFeatPtr := CaseFeatPtr^.NextOr;
                If (MatchValue > BestMatch ) then
                        BestMatch := Matchvalue;
        End;
        ...
End;
```

หลังจากพิจารณาในระดับกลุ่มลักษณะย่อยแล้ว ต่อไปจะพิจารณาในระดับกลุ่มลักษณะทั้งหมด โดยโครงสร้างทางตรรกะในระดับนี้เป็นการแอนด์กันของกลุ่มลักษณะย่อย ดังนั้นการจับคู่ในลักษณะนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะอยู่ในลักษณะต้องเฉลี่ยค่าความเข้ากันได้ โดยให้ความสำคัญกับทุกๆ กลุ่มของลักษณะย่อย ค่าความเข้ากันได้จากการจับคู่ในระดับนี้ ถือเป็นค่าความเข้ากันได้ทั้งหมดที่ได้จากการจับคู่กรณีนั้น กับสถานการณ์ใหม่ โดยได้มาจากการคำนวณจากสมการ

ค่าความเข้ากันได้ทั้งหมด = (ค่าความเข้ากันได้ของกลุ่มลักษณะย่อย1 +
ค่าความเข้ากันได้ของกลุ่มลักษณะย่อย2 +
ค่าความเข้ากันได้ของกลุ่มลักษณะย่อย3 +
...
ค่าความเข้ากันได้ของกลุ่มลักษณะย่อย) / ค่าถ่วงน้ำหนักในแต่ละมิติ

n คือจำนวนกลุ่มลักษณะย่อยทั้งหมดของกรณี

ซึ่งค่าถ่วงน้ำหนักในแต่ละมิติ ได้มาจากสมการ

ค่าถ่วงน้ำหนัก = จำนวนมิติทั้งหมดที่จับคู่กันได้ + จำนวนฟีเจอร์ที่ป้อนเข้ามาแล้วไม่สามารถจับคู่ได้

ค่าความเข้ากันได้ทั้งหมดจะมีค่าอยู่ระหว่าง 0 ถึง 1 และค่าความเข้ากันได้ของกลุ่มลักษณะย่อยก็มีค่าอยู่ระหว่าง 0 ถึง 1 เช่นกัน ดังนั้นค่าถ่วงน้ำหนักของแต่ละกลุ่มลักษณะย่อยทั้งหมดต้องรวมกันแล้วได้ 1 พอดี ดังนั้นค่าถ่วงน้ำหนักของกลุ่มลักษณะย่อยแต่ละตัว จะได้มาจากการเฉลี่ยค่าจากจำนวนกลุ่มลักษณะย่อยทั้งหมดของกรณี

```
VAR
    SitPtr : SituationPtr;
    TempSit : SituationPtr;
    FeatGroupPtr : AndPtr;
    CaseRec : CaseRecord;
    NumFeat : byte;
    NumFeatMatch : byte;
    DimensionMatch : byte;
    MatchValue : Real;
    BestMatch : Real;
    AllMatch : Real;
    ...
Begin
    ...
    AllMatch := 0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
TempSit := SitPtr;
NumFeat := 0;
While (TempSit <> Nil) do
Begin
        TempSit := TempSit^.Next;
        Inc(NumFeat);
End;
FeatGroupPtr := CaseRec.Feature;
While (FeatGroupPtr <> Nil) do
Begin
        วนลูปจับคู่กลุ่มลักษณะย่อยที่ FeatGroupPtr ชี้อยู่กับลักษณะของสถานการณ์;
        ถ้าสามารถจับคู่ฟีเจอร์ที่รับเข้ามากับฟีเจอร์หนึ่งๆในกรณีได้;
        ให้เพิ่มค่า NumFeat ขึ้น 1;
        If BestMatch > 0 then
        Begin
            Inc(DimensionMatch);
            AllMatch := AllMatch + BestMatch;
        End;
        FeatGroupPtr := FeatGroupPtr^.NextAnd;
End;
AllMatch := AllMatch / ((NumFeat – NumMatchFeat) + DimensionMatch);
...
End;
```

ขั้นตอนสุดท้าย คือการนำอัลกอริธึมที่กล่าวมา มาทำซ้ำกับกรณีอื่นๆ ที่อยู่ในไลบรารีของกรณี โดยทุกครั้งที่มีการจับคู่ระหว่างสถานการณ์กับกรณี หากค่าความเข้ากันได้มีค่ามากกว่า 0 นั่นคือมีความเข้ากันได้อยู่บ้าง จะต้องมีการจดจำว่ากรณีนี้เกิดการจับคู่ขึ้น โดยเก็บค่าเลขที่ของกรณี (คือ ดัชนีที่ระบุถึงกรณีนี้ บนอาร์เรย์ของกรณี หรือไลบรารีของกรณีนั่นเอง) และค่าความเข้ากันได้กับสถานการณ์ใหม่ (เป็นข้อมูลชนิดเรียล) เพื่อจะได้นำไปเปรียบเทียบกับค่าความเข้ากันได้ ระหว่างสถานการณ์นี้กับกรณีอื่นๆ ในไลบรารีของกรณี

จากที่ได้กล่าวมาใน 2 บทที่ผ่านมา จะพบว่าขณะนี้โครงสร้างข้อมูลของไลบรารีของกรณี และอัลกอริธึมในการค้นหาและจับคู่ระหว่างกรณีในไลบรารีของกรณี กับสถานการณ์ใหม่ ถูกกำหนดไว้อย่างละเอียดแล้ว เมื่อมาถึงจุดนี้แล้วเราสามารถสร้างระบบอ้างเหตุผลเชิงกรณีชนิดที่ 1 ซึ่งทำหน้าที่ในการเรียกค้นกรณีที่เข้ากันได้กับสถานการณ์ใหม่ออกมาแสดงผลตามลำดับความเข้ากันได้ ได้แล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8.2 การออกแบบอัลกอริธึมในการดัดแปลงผลลัพธ์

หลังจากที่เราสามารถเรียกกลุ่มกรณีซึ่งมีความเกี่ยวข้องกับสถานการณ์ใหม่ขึ้นมาได้แล้ว ก็จะเข้าสู่กระบวนการในการหาคำตอบที่แท้จริง โดยคำคอบนี้ย่อมได้มาจากกลุ่มกรณีที่เรียกค้นขึ้นมา

ในการเลือกกรณีที่จะใช้เป็นหลักในการสร้างผลลัพธ์ จะเลือกกรณีที่มีความเกี่ยวเนื่องกับสถานการณ์ใหม่มากที่สุด โดยดูได้จากค่าที่ได้จากการรับคู่กรณีนั้น กับสถานการณ์ใหม่ หรือค่าความเข้ากันได้ ซึ่งได้มาจากการคำนวนในอัลกอริธึมของการเรียกค้น และการจับคู่ดังที่ได้กล่าวมาแล้ว

หลังจากนั้นจะทำการเลือกสิ่งที่จะทำการดัดแปลง จากผลลัพธ์ของกรณีที่เราเลือกขึ้นมาเป็นจุดเริ่มต้นในการดัดแปลง ก่อนที่จะกล่าวถึงในส่วนนี้ จะต้องทำการเลือก และออกแบบวิธีที่จะใช้ในการดัดแปลงเสียก่อน เพื่อที่จะออกแบบวิธีในการเลือกส่วนที่จะดัดแปลง ให้สอดคล้องกับวิธีการในการดัดแปลง

วิธีการในการดัดแปลงที่เลือกใช้มี 2 วิธีหลักคือ

1. วิธีการแทนค่า โดยวิธีที่เลือกใช้ในการแทนค่า จากทฤษฎีของวิธีนี้ วิธีที่เหมาะสมจะใช้คือวิธีปรับพารามิเตอร์ และวิธีการแทนค่าโดยใช้หลักการของกรณี ซึ่งรายละเอียดและหลักการของทั้งสองวิธีนี้ได้กล่าวไว้แล้วอย่างละเอียด ในทฤษฎีของการดัดแปลงกรณี

วิธีปรับพารามิเตอร์จะใช้ได้กับเฉพาะส่วนของจำนวนวันในการสังเกตอาการเท่านั้น เพราะเป็นเพียงส่วนเดียวที่เป็นข้อมูลเชิงจำนวน (รหัสยาถึงแม้จะเป็นตัวเลข แต่ก็ไม่ได้มีความเกี่ยวข้องกับจำนวนและระดับของค่า) ซึ่งตรงตามหลักการของวิธีปรับพารามิเตอร์ โดยการปรับค่าจะขึ้นอยู่กับการประเมินความต่างระหว่างสถานการณ์ใหม่ กับกรณีที่ใช้เป็นต้นแบบของผลลัพธ์ แล้วนำความต่างนี้มาเป็นปัจจัยในการปรับความค่า ซึ่งความต่างที่จะมีผลต่อการปรับค่าจำนวนวันในการสังเกตอาการ จะต้องเป็นความต่างที่เกี่ยวข้องกับความรุนแรงของอาการเท่านั้น

วิธีการแทนค่าโดยใช้หลักการของกรณี ซึ่งจะเป็นการเรียกกรณีอื่นๆ ที่เกี่ยวข้องขึ้นมาช่วยในการแทนค่า โดยรายละเอียดที่ใช้ในการเรียกค้นได้มาจาก การประเมินความต่างของลักษณะของสถานการณ์ใหม่ กับปัญหา หรือกรณีเก่า โดยหลังจากเรียกค้นกรณีขึ้นมาแล้วก็จะนำกรณีนั้นมาพิจารณาความต่างของค่า แล้วเลือกกรณีที่จะนำค่ามาแทนในผลลัพธ์ต้นแบบ

2. วิธีการปรับเปลี่ยนรูปร่าง เป็นวิธีที่เลือกใช้ถ้าไม่สามารถใช้วิธีในการแทนค่าได้ โดยในระบบนี้จะใช้ ฮิวริสติกในการปรับเปลี่ยนโครงสร้าง โดยแท้ที่จริงแล้ว เสมือนเป็นการแทนค่า แต่เป็นการแทนค่าลงในส่วนของข้อเสนอแนะ มีโครงสร้างเป็นสตริง ดังนั้นจะใส่ข้อมูลอย่างไรก็ได้ โดยไม่กำหนดลักษณะของข้อมูล ซึ่งข้อมูลที่ว่านี้อาจจะเกี่ยวกับ การดูแลรักษาเบื้องต้น คำแนะนำในการรักษาตัว หรือการตรวจร่างกายที่ต้องทำเพิ่มเติม ซึ่งในส่วนนี้อิสระต่อโครงสร้างข้อมูล แต่จะต้องใช้ฮิวริสติกในส่วนนี้ ซึ่งการปรับเปลี่ยนบางอย่างอาจใช้หลักการของกรณีเข้ามาเกี่ยวข้องด้วยก็ได้ ซึ่งตามหลักการแล้วจะต้องมีการประเมินความต่างของสถานการณ์ใหม่กับกรณีเก่า การแบ่งแยกส่วนประกอบหลัก และส่วนประกอบรอง (ในที่นี้ ส่วนประกอบหลัก หมายถึง ชื่อโรค , เวลาในการดูอาการ และยาที่ใช้ ส่วนประกอบรองคือคำอธิบายเพิ่มเติม) ซึ่งส่วนประกอบรอง ก็คือส่วนที่นำมาทำการเปลี่ยนแปลงได้ หรือเปลี่ยนโครงสร้างโดยไม่กระทบถึงความสัมพันธ์ภายในของผลลัพธ์นั่นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากที่ได้กล่าวถึงวิธีการในการดัดแปลงผลลัพธ์ของกรณีที่เลือกใช้ทั้งสองแบบ จะเห็นได้ว่าในการกำหนด สิ่งที่จะทำการดัดแปลงนั้น จะได้มาจากการประเมินความต่างของสถานการณ์ใหม่ กับกรณีที่เข้ากับสถานการณ์นั้นที่สุด แล้วนำความต่างนี้มาเป็นปัจจัยในการเลือกดัดแปลงส่วนประกอบของผลลัพธ์ เช่น ถ้าความต่างนั้นเป็นความต่างในด้านระดับความรุนแรงของอาการ ส่วนที่ต้องทำการดัดแปลงก็คือเวลาที่ใช้ในการสังเกตอาการ

แต่ในความเป็นจริง ความแตกต่างระหว่างสถานการณ์ใหม่ กับผลลัพธ์ของกรณีเก่า จะมีหลายๆ ด้าน ดังนั้นจากการประเมิน เราจะได้กลุ่มของส่วนประกอบที่ควรทำการดัดแปลง ซึ่งในการดัดแปลงจะยึดถือส่วนประกอบที่ขึ้นอยู่กับ ความแตกต่างที่มากที่สุดเป็นสำคัญ นอกจากนี้ปัจจัยที่ใช้ในการเลือกส่วนที่จะทำการดัดแปลง (ตามทฤษฎี) ยังขึ้นอยู่กับความยากง่ายของวิธีที่สามารถใช้ในการดัดแปลงอีกด้วย (แน่นอนว่า วิธีที่สะดวก ไม่ซับซ้อนย่อมเป็นที่ต้องการมากกว่า)

หลังจากกำหนดส่วนที่จะทำการดัดแปลงแล้ว ก็จะต้องทำการเลือกวิธีที่จะใช้ในการดัดแปลง ซึ่งเป็นกลไกภายในของตัวอ้างเหตุผลที่จะเลือกวิธีที่มีอยู่ให้เหมาะสม ดังนี้

- ถ้าส่วนที่จะทำการแก้ไขสามารถใช้วิธีปรับพารามิเตอร์ได้ นั่นคือเป็น ส่วนเวลาที่ใช้ในการสังเกตอาการ ให้เลือกใช้วิธีปรับพารามิเตอร์ โดยพิจารณาระดับที่จะปรับ จากระดับความต่างของความรุนแรงของอาการของสถานการณ์ใหม่ กับกรณีเก่า
- ถ้าส่วนที่ทำการแก้ไขสามารถใช้วิธีการแทนค่าได้ นั่นคือสามารถหาค่ามาแทนในส่วนที่ต้องการได้ โดยการเรียกค้นกรณีเพิ่ม หรือจากกรณีที่เกี่ยวข้องเป็นอันดับรองลงมาจากการเรียกค้นในขั้นต้น โดยกรณีที่จะนำมาใช้ จะต้องมีความสัมพันธ์กับสถานการณ์ใหม่ และมีความใกล้ชิดกับกรณีที่ใช้ผลลัพธ์เป็นต้นแบบ
- ถ้าไม่สามารถใช้ 2 วิธีข้างต้นได้ จะต้องทำการปรับเปลี่ยนโครงสร้าง โดยการปรับเปลี่ยนจะกระทำที่ส่วนคำแนะนำ ด้วยวิธีทางฮิวริสติก หรือการอ้างอิงกรณีอื่นๆ ถ้าเป็นไปได้ให้เลือกใช้วิธีที่เป็นฮิวริสติกก่อน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่9
การออกแบบการทำงานของระบบโดยรวม

9.1 การทำงานของระบบโดยรวม

การทำงานของระบบจะต้องประกอบด้วยส่วนที่ทำหน้าที่หลักๆ ต่อไปนี้ คือ

- การสร้างคำอธิบายของปัญหาหรือสถานการณ์ใหม่
- การเรียกค้นไลบรารีของกรณีโดยใช้คำอธิบายที่สร้างขึ้น
- การเลือกผลลัพธ์ที่เกี่ยวเนื่องที่สุด และทำการปรับเปลี่ยนผลลัพธ์นั้น ให้เหมาะสมกับสถานการณ์ใหม่ยิ่งขึ้น
- การรับค่าป้อนกลับผลจากการใช้งานผลลัพธ์ เพื่อนำไปปรับปรุงประสิทธิภาพการทำงานของไลบรารีของกรณี
- การแก้ไข หรือเพิ่มเติมไลบรารีของกรณี ในกรณีที่มีการเรียนรู้สถานการณ์ หรือความรู้ใหม่ๆ ที่จะสามารถนำไปใช้ประโยชน์ได้ในอนาคต

ซึ่งเมื่อทำการพัฒนาระบบขึ้นมาจะต้องออกแบบให้หน้าที่ต่างๆ เหล่านี้ ถูกฝังซ่อนอยู่ภายใต้ส่วนติดต่อกับผู้ใช้ ให้สามารถทำงานได้อย่างกลมกลืน และไม่เป็นที่สับสนต่อการใช้งานของผู้ใช้ ซึ่งในที่นี้ระบบที่พัฒนาขึ้นตั้งใจจะใช้เป็น เครื่องมือช่วยแพทย์ในการวินิจฉัยโรค โดยสามารถให้คำตอบเกี่ยวกับอาการของโรคซึ่งผู้ใช้ป้อนเข้ามา และสามารถเสนอแนะแนวทางในการวินิจฉัยอื่นๆ รวมทั้งผลลัพธ์ที่อาจเกี่ยวข้อง หรือมีความเป็นไปได้ ต่อผู้ใช้ด้วย

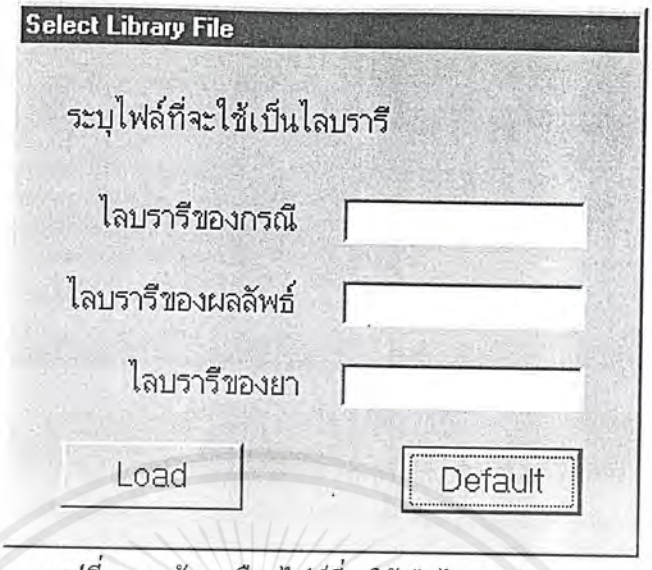
ส่วนติดต่อกับผู้ใช้ของระบบ ออกแบบให้แยกเป็นส่วนต่างๆ ดังนี้

9.1.1 หน้าจอต้อนรับเข้าสู่ระบบ

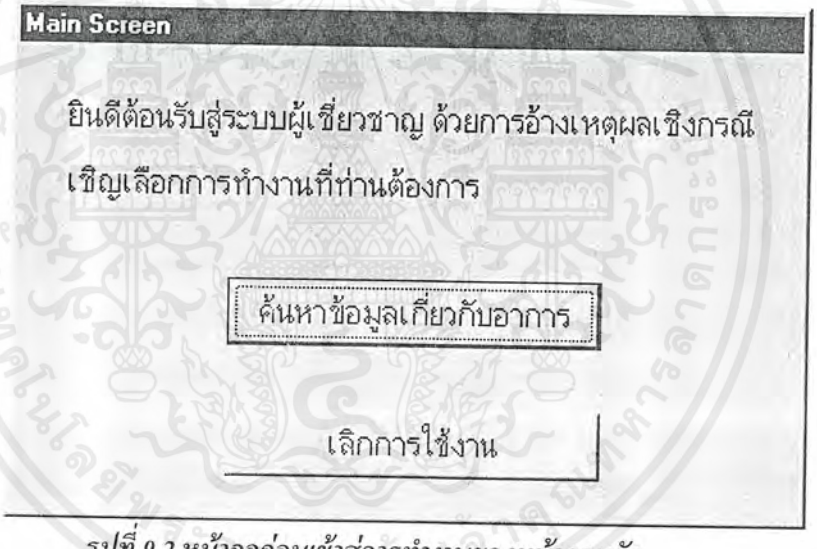
หน้าจอต้อนรับเข้าสู่ระบบ (รูปที่ 9.1) เป็นหน้าจอแรกที่ผู้ใช้จะพบ ที่หน้าจอนี้จะรับค่าชื่อไลบรารีที่จะใช้งานจากผู้ใช้ (สามารถเลือกค่าปกติของระบบได้ แล้วระบบจะทำการเลือกไลบรารีปกติของระบบขึ้นมาทันที) ในขั้นตอนนี้จะทำให้ผู้ใช้สามารถเลือกไฟล์ที่จะใช้เป็นไลบรารีที่จะใช้ทำงานในระบบได้

หลังจากผู้ใช้ระบุไลบรารีและตอบรับการเข้าใช้งานระบบแล้ว ก็จะปิดหน้าจอนี้แล้วนำผู้ใช้ไปสู่หน้าจอถัดไป (รูปที่ 9.2) โดยก่อนที่จะแสดงหน้าจอถัดไป จะต้องทำการอ่านไฟล์ไลบรารีจากดิสก์ เข้ามาสู่หน่วยความจำของระบบก่อน โดยขั้นตอนในการอ่านไฟล์ และแปลงรูปแบบข้อมูลจากไฟล์ ไปสู่รูปแบบข้อมูลของไลบรารีในหน่วยความจำได้กล่าวมาแล้วในบทที่ผ่านมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9.1 หน้าจอเลือกไฟล์ที่จะใช้เป็นไลบรารี



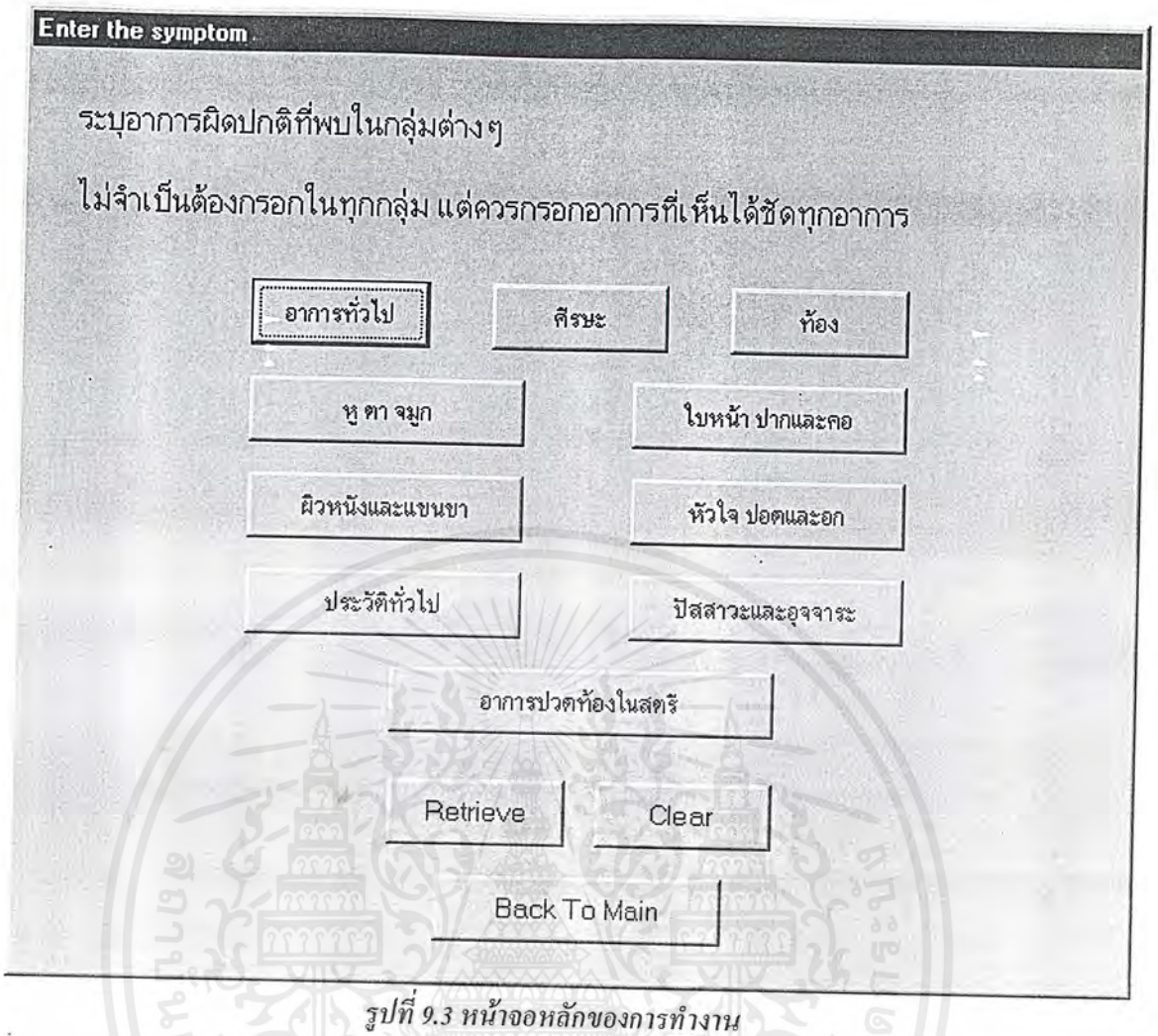
รูปที่ 9.2 หน้าจอก่อนเข้าสู่การทำงานของหน้าจอหลัก

9.1.2 หน้าจอหลัก

หน้าจอหลัก (รูปที่ 9.3) เป็นหน้าจอที่แสดงผลโดยยูนิตหลักของระบบ ซึ่งในยูนิตนี้จะเป็นที่เก็บรวบรวมข้อมูลที่สำคัญ ที่ต้องใช้ร่วมกันของระบบ และถือเป็นส่วนศูนย์กลางของระบบด้วย ในส่วนจะเชื่อมโยงไปสู่หน้าจอที่ข้อมูลจากผู้ใช้จะถูกป้อนเข้ามา (รูปที่ 9.4) โดยการใช้ฟอร์ม ซึ่งแบ่งข้อมูลที่รับจากผู้ใช้ตามลักษณะอาการ โดยข้อมูลที่รับก็คืออาการผิดปรกติที่เกิดขึ้นกับร่างกายนั่นเอง ซึ่งการรับข้อมูลก็มีทั้งแบบใช้เช็คบ๊อกซ์ และเรดิโอบัททอน หรือซีเล็กชันบ็อกซ์

ในจากรับข้อมูลอาการจากผู้ใช้ ระบบจะทำการสร้างลิงค์ลิสท์เพื่อรับอาการแต่ละอย่าง เพื่อที่จะนำไปเป็นข้อมูลในการเรียกค้นไลบรารีของกรณี โดยโครงสร้างข้อมูลและขั้นตอนในการสร้างลิสท์ของอาการได้กล่าวมาแล้วในเรื่องของอัลกอริธึมในการเรียกค้นข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9.3 หน้าจอหลักของการทำงาน

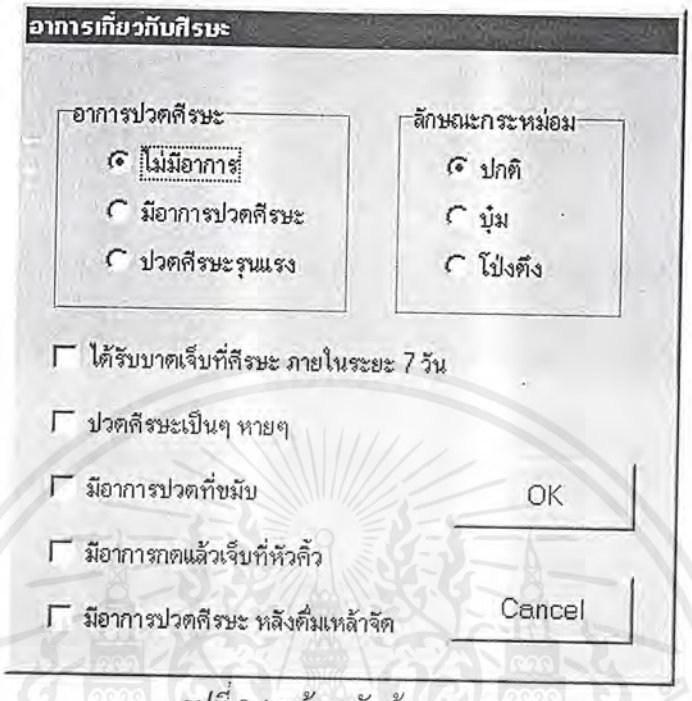
เมื่อระบบได้รับคำสั่งให้ประมวลผล หลังจากผู้ใช้ระบุอาการจนเป็นที่พอใจแล้ว ระบบจะนำรายการของอาการที่ผู้ใช้กรอกเข้ามาไปทำการเรียกค้น และทำการจับคู่ในไลบรารีของกรณี โดยใช้อัลกอริธึมที่กล่าวมาแล้วในเรื่องของอัลกอริธึมในการเรียกค้นข้อมูล จากกระบวนการนี้ผลลัพธ์ที่ได้คือ รายการของกรณีซึ่งมีความเกี่ยวข้องกับสถานการณ์ใหม่ และค่าความเกี่ยวข้อง ซึ่งระบุระดับความเป็นประโยชน์ของกรณีต่อสถานการณ์ใหม่ ถึงแม้ส่วนนี้จะยังไม่ได้นำออกแสดงผลต่อผู้ใช้ แต่ก็เป็นประโยชน์ต่อการทำงานในขั้นตอนต่อไป

หลังจากได้กลุ่มของกรณีที่มีความเกี่ยวข้องกับสถานการณ์ใหม่แล้ว ก็จะต้องทำการเลือกผลลัพธ์ของกรณีที่ดีที่สุดมาทำการดัดแปลง หรือแก้ไขเพื่อให้ผลลัพธ์นั้นเหมาะสมกับสถานการณ์ใหม่ให้มากยิ่งขึ้น ในขั้นตอนนี้อาจมีการนำผลลัพธ์ของกรณีอื่นๆ ที่ถูกเรียกค้นขึ้นมาเป็นอันดับรองๆ หรือมีการเรียกค้นกรณีด้วยรายละเอียดใหม่ๆ ขึ้นมา เพื่อนำผลลัพธ์มาช่วยในการดัดแปลงผลลัพธ์ต้นแบบ ซึ่งในขั้นตอนการดัดแปลงนี้ได้กล่าวมาแล้วในเรื่องของการดัดแปลงผลลัพธ์ของกรณี

เมื่อสิ้นสุดการดัดแปลงผลลัพธ์แล้ว ก็เป็นการจบการทำงานในส่วนหน้าจอหลัก ผลลัพธ์ที่ถูกทำการดัดแปลงจะถูกส่งไปแสดงผลยังหน้าจอถัดไป แต่ยังคงเก็บข้อมูลอื่นๆ ที่จำเป็นไว้ในส่วนนี้ เช่น ข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อาการที่ผู้ใช้ป้อนเข้ามา หรือรายการของกรณีที่ถูกเรียกขึ้นมาใช้ เพื่อเป็นข้อมูลสนับสนุนในการทำงานที่อาจต้องอ้างถึงขั้นตอนในการประมวลผลคำตอบ (เช่น การประมวลผลค่าป้อนกลับจากผู้ใช้)



รูปที่ 9.4 หน้าจอรับข้อมูลอาการ

9.1.3 หน้าจอแสดงผลลัพธ์

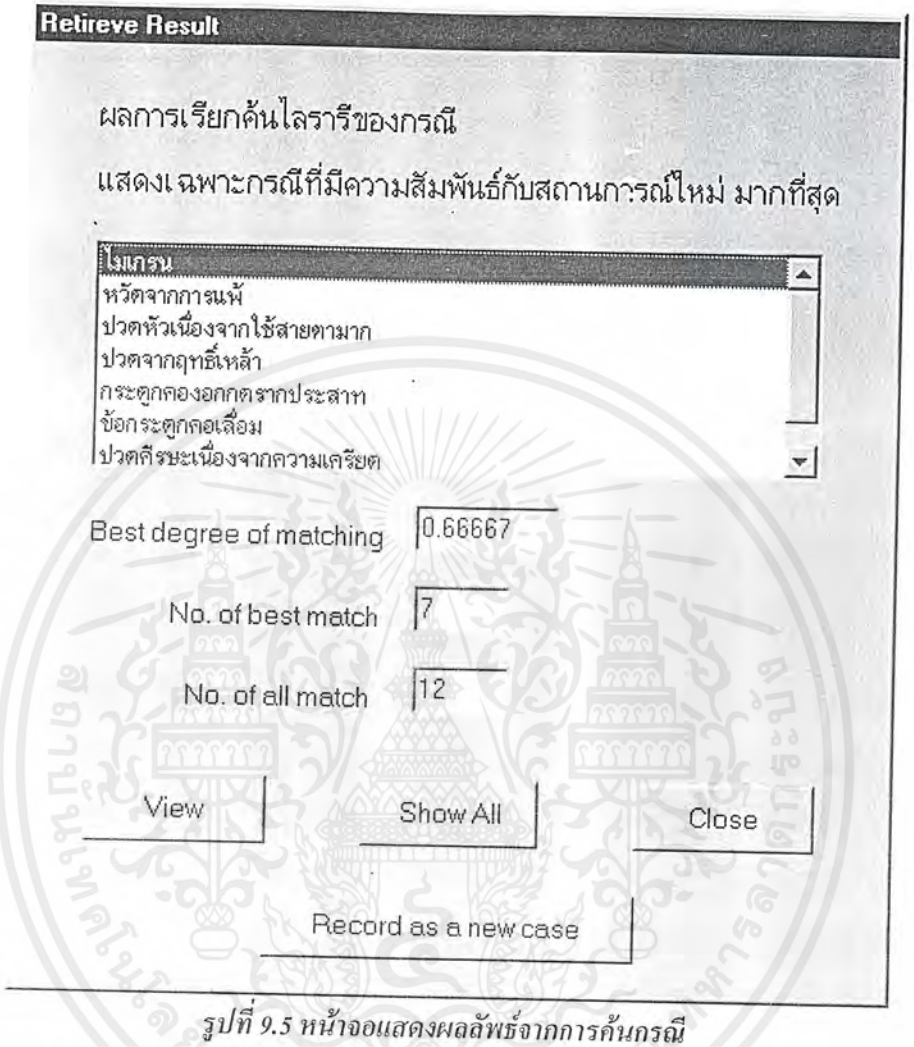
หลังจากหน้าจอหลักทำการประมวลผล หาผลลัพธ์และดัดแปลงผลลัพธ์จนเสร็จเรียบร้อยแล้ว ก็จะส่งผลลัพธ์มาแสดงผลยังหน้าจอแสดงผลลัพธ์ (รูปที่ 9.5) ซึ่งในขั้นตอนนี้จะแสดงผลถึงรายละเอียดที่เกี่ยวข้องกับอาการที่ผู้ใช้ป้อนเข้ามา (รูปที่ 9.6 และ 9.7) ซึ่งรวมไปถึงผลลัพธ์อื่นที่เกี่ยวข้องด้วย เสมือนเป็นการเสนอแนะแนวทางทางเลือกอื่นที่อาจเป็นไปได้ในสถานการณ์นั้น โดยทางเลือกจะได้มาจากผลลัพธ์ของกรณีที่ใกล้เคียงกับสถานการณ์นั้นเป็นลำดับรองๆ ลงมา (รูปที่ 9.8 และ 9.9)

เมื่อได้แสดงผลให้กับผู้ใช้ตรวจสอบผลลัพธ์แล้ว หน้าจอนี้ยังถามผลของการนำ ผลลัพธ์นี้ไปใช้ในสถานการณ์จริง เพื่อนำผลนี้มาป้อนกลับเพื่อยืนยันความถูกต้องในการทำงานของระบบ ซึ่งถ้าขาดส่วนนี้ไป ประสิทธิภาพที่เป็นแบบความจำแบบไดนามิกของระบบจะไม่สามารถเกิดขึ้นได้เลย เพราะความผิดพลาดที่เคยเกิดขึ้นแล้วก็จะเกิดขึ้นซ้ำอีก

ถ้าผู้ใช้ยืนยันความถูกต้องของผลแล้ว ระบบก็จะทำการตรวจสอบผลลัพธ์นั้นอีกครั้งหนึ่ง ถึงกระบวนการในการได้มาซึ่งผลลัพธ์นั้น ถ้าผลลัพธ์นั้นใช้กระบวนการในการอ้างเหตุผลและค้นหาคำตอบที่ยุ่งยากและสิ้นเปลือง ก็จะต้องมีการจดจำกรณีนั้น หรือเขียนกรณีใหม่ลงในไลบรารีนั่นเอง ซึ่งถือเป็นการเรียนรู้ความรู้ภายใต้สถานการณ์ใหม่ๆ ซึ่งในการเพิ่มกรณีลงในไลบรารีจะต้องทำการเพิ่มทั้งในไลบรารีของกรณี และไลบรารีของผลลัพธ์ (ถ้าเกิดผลลัพธ์ใหม่) และจะต้องบันทึกลงในไฟล์ของไลบรารี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ด้วย หลังจากนั้นก็กลับไปยังหน้าจอหลัก เพื่อกลับเข้าสู่การทำงานหลัก ซึ่งก็คือการรับรายการอาการจากผู้ใช้



รูปที่ 9.5 หน้าจอแสดงผลลัพธ์จากการค้นกรณี

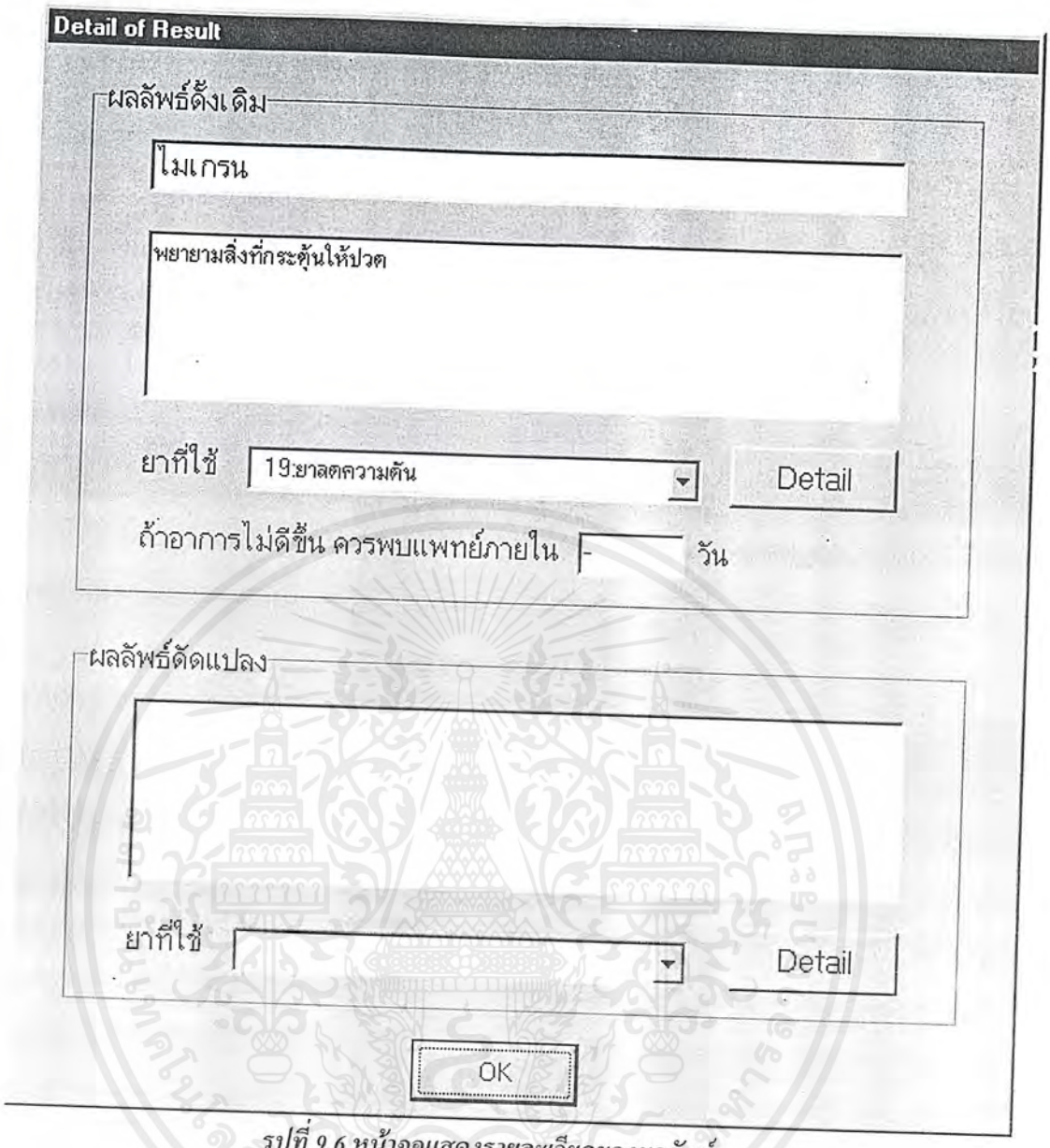
ถ้าผู้ใช้ไม่ยืนยันว่าผลลัพธ์นั้นถูกต้อง ก็จะต้องไปทำงานยังหน้าจอถัดไปซึ่งจะเป็นส่วนที่จะวิเคราะห์ความผิดพลาดที่เกิดขึ้น และทำการแก้ไขไลบรารีของกรณีให้ถูกต้อง ในกรณีที่เกิดความผิดพลาดจากไลบรารีของกรณี

9.1.4 หน้าจอแก้ไขและตรวจสอบความผิดพลาด

ที่หน้าจอนี้ (รูปที่ 9.10) ผู้ใช้จะต้องระบุความผิดพลาดที่เกิดขึ้นจากการนำไปใช้ รวมทั้งระบุคำอธิบายของสถานการณ์ (อาการผิดปรกติ) เพิ่มเติม (ถ้ามี) (รูปที่ 9.11 และ 9.12) โดยผลลัพธ์ที่ได้จากหน้าจอนจะนาไปใชในการแกไข หรอเพมเตมไลบราร ซงในขนตอนนตองการผลลพธจากการนาไปใชงานจรงมาเปรยบเทยบกบผลลพธ เพอคนหาสวนทผดพลาดของผลลพธเดม ซงถาผใชสามารถระบถงผลลพธทถกตองได จะทาใหการคนหาขอผดพลาด และการเรยนรเปนไปไดงายยงขน

ถพฒฝฉ9.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

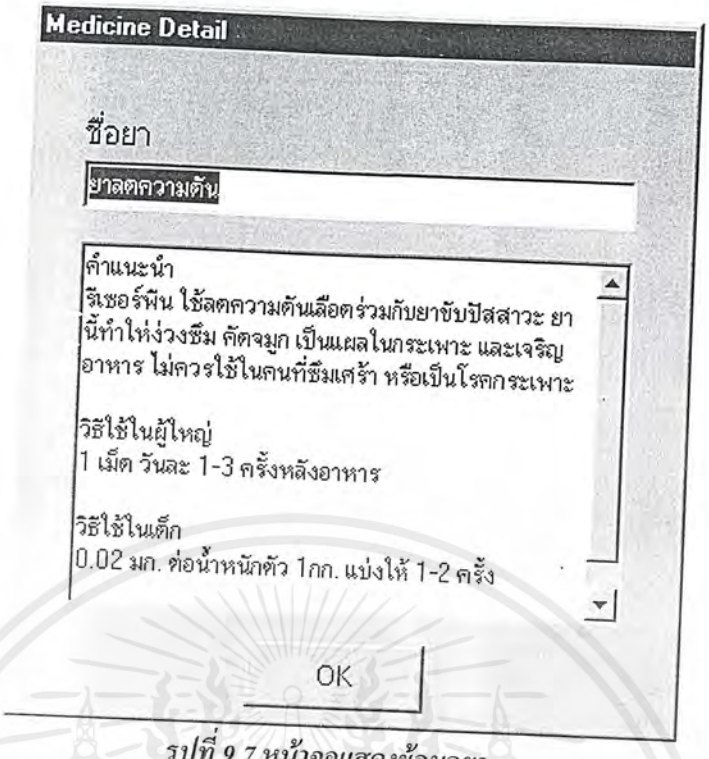


รูปที่ 9.6 หน้าจอแสดงรายละเอียดของผลลัพธ์

หลังจากแก้ไขผลลัพธ์ให้ตรงกับการใช้งานจริงแล้ว ก็จะกลับสู่หน้าจอหลัก โดยก่อนจะกลับสู่หน้าจอหลัก เช่นเดียวกับการทำงานของหน้าจอแสดงผลลัพธ์ จะต้องทำการตรวจสอบว่า ต้องมีการดัดแปลงแก้ไขไลบรารีหรือไม่ ถ้ามีจะต้องจัดการให้เรียบร้อยก่อนที่จะกลับเข้าสู่การทำงานในหน้าจอหลัก รวมทั้งเขียนไลบรารีที่แก้ไขแล้วลงในไฟล์ด้วย

จะเห็นว่าทุกครั้งที่มีการแก้ไขที่มีการแก้ไข หรือเพิ่มเติมไลบรารีของกรณี จะทำการเขียนลงในไฟล์ของไลบรารีทันที เนื่องจากการดัดแปลงแก้ไขไลบรารี ไม่ใช่สิ่งที่จะพบเจอบ่อยๆ ในการทำงานของระบบ เฉพาะสถานการณ์ที่เป็นประโยชน์คุ้มค่าต่อการเก็บเป็นกรณี และความผิดพลาดที่ชัดเจนเท่านั้นที่จะถูกนำไปเพิ่ม หรือแก้ไขในไลบรารี ดังนั้นการเขียนไลบรารีลงในไฟล์ทันที จึงไม่ใช่เรื่องที่สิ้นเปลืองหรือเสียหายอะไร ทั้งยังช่วยป้องกันการสูญหายของกรณีที่ทำการเพิ่ม หรือแก้ไข ในกรณีที่เกิดความผิดพลาดขึ้นกับระบบ (เช่น ปัญหาไฟฟ้า หรือเครื่องแฮงค์)

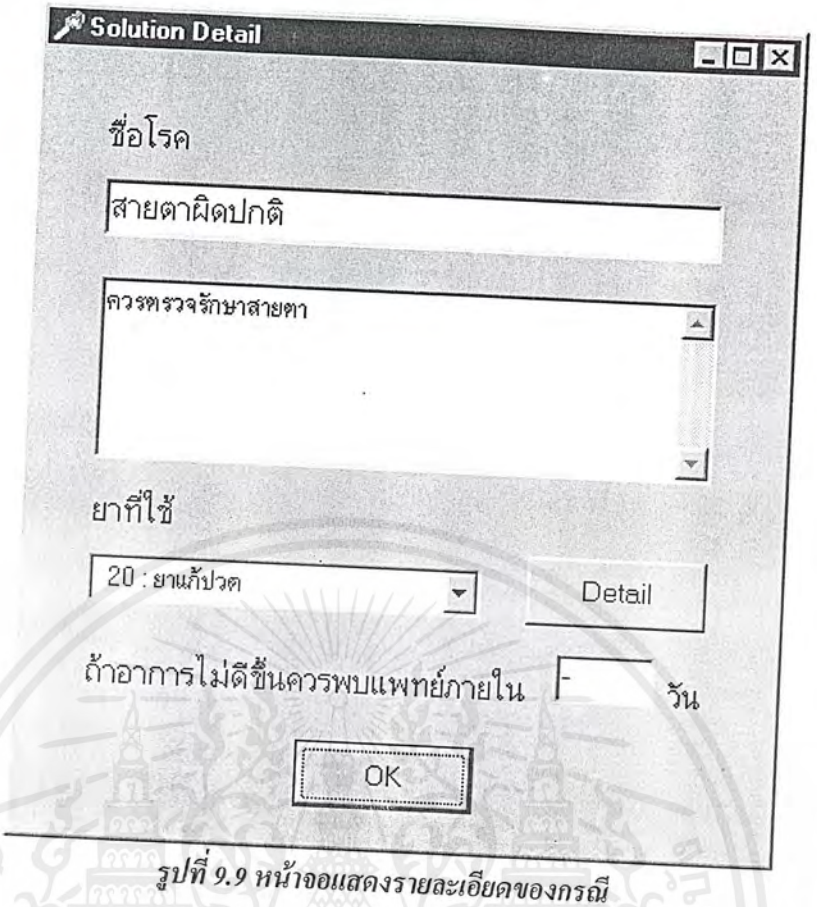
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



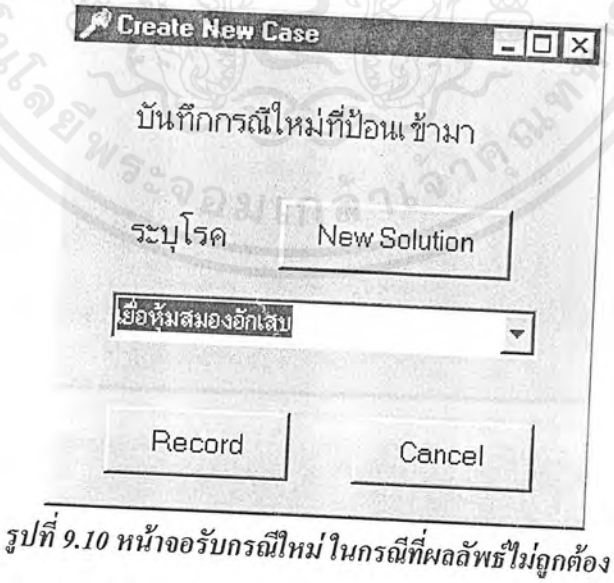
รูปที่ 9.7 หน้าจอแสดงข้อมูลยา

รูปที่ 9.8 หน้าจอแสดงกรณีอื่นๆ ที่อาจเป็นได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

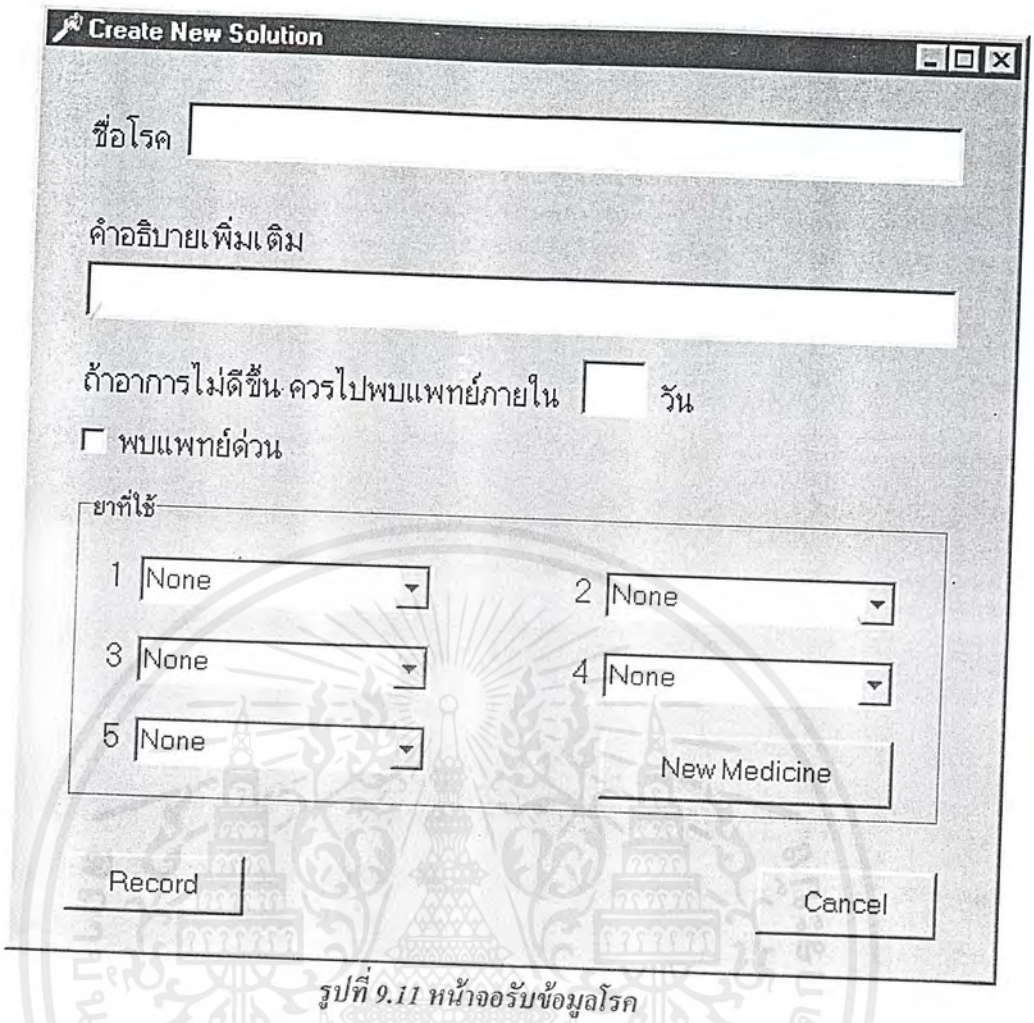


รูปที่ 9.9 หน้าจอแสดงรายละเอียดของกรณี



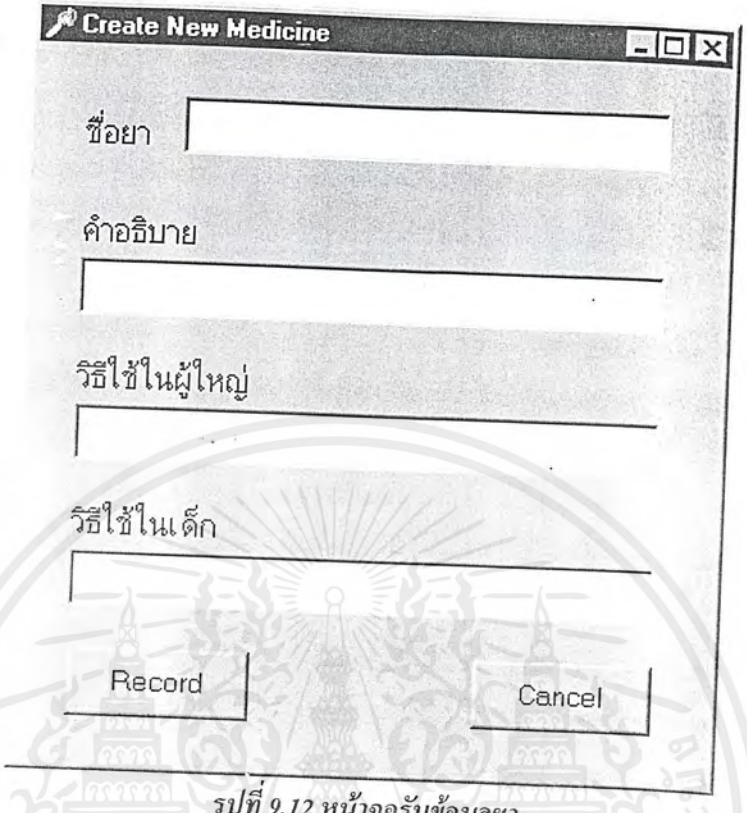
รูปที่ 9.10 หน้าจอรับกรณีใหม่ ในกรณีที่ผลลัพธ์ไม่ถูกต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9.11 หน้าจอรับข้อมูลโรค

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9.12 หน้าจอรับข้อมูลยา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 10
ผลการทดสอบ

หลังจากที่ได้ทำการพัฒนาระบบจนเสร็จสมบูรณ์ตามที่ได้ทำการออกแบบมาแล้ว ก็ได้ทำการนำระบบไปทำการทดลองผลการใช้งาน ณ โรงพยาบาลกรุงเทพ โดยคุณหมออัครพันธ์ ซึ่งทำการทดสอบโดยการอาการของโรคที่คุณหมอนำมา มาป้อนลงระบบ แล้วเปรียบเทียบผลที่ได้จากระบบกับผลของคุณหมอ

ผลการทดสอบปรากฏว่าครึ่งหนึ่ง หรือร้อยละ 50 ของการทดสอบได้ผลตรงตามความเป็นจริงทุกประการ ตัวอย่างเช่น นำอาการของไข้หวัดใหญ่มาป้อน ก็ได้ผลเป็นไข้หวัดหรือไข้หวัดใหญ่ ส่วนอีกครึ่งหนึ่งนั้น ผลที่ได้ค่อนข้างจะเป็นผลที่ไม่แน่นอน คือ ในบางกรณีจะได้ผลที่ใกล้เคียงกับผลลัพธ์จริง คือเป็นโรคในกลุ่มเดียวกัน แต่เป็นโรคที่รุนแรงกว่า เช่น ไข้หวัดใหญ่ , วัณโรค และ มะเร็งปอด ซึ่งมีอาการคล้ายกัน และอยู่ในกลุ่มโรคที่คล้ายกัน แต่มีความแตกต่างกันที่ระดับความรุนแรงของโรคและอาการ

ส่วนผลการทดสอบที่นอกเหนือจากนี้ไป เป็นผลที่ค่อนข้างผิดพลาด คือ ให้ผลของโรคที่มีอาการใกล้เคียงกับโรคที่ถูกต้อง แต่เป็นโรคที่ต่างกับโรคที่ถูกต้อง ทั้งในด้านสาเหตุของโรค วิธีปฏิบัติในการรักษา เช่น ป้อนอาการ ปวดท้อง และ ปวดจุกเสียดบริเวณลิ้นปี่ ซึ่งคุณหมอให้ความเห็นว่าน่าจะเป็นอาการของโรคกระเพาะ แต่ระบบให้ผลลัพธ์ออกมาเป็นอาการปวดท้องธรรมดา ซึ่งมีสาเหตุต่างกัน และวิธีรักษาที่ต่างกัน คือปวดท้องธรรมดาให้กินยาแก้ปวดท้องบรรเทาอาการ แต่โรคกระเพาะห้ามกินยาแก้ปวดเพราะจะยิ่งทำให้อาหารยิ่งแย่ลง ซึ่งความผิดพลาดที่เกิดขึ้นในกรณีนี้เนื่องมาจากอาการที่ป้อนเข้ามาไม่ครบถ้วนตามลักษณะอาการของโรคกระเพาะ ซึ่งประกอบด้วย ปวดท้อง , ปวดเป็นๆหายๆ และปวดจุกบริเวณลิ้นปี่ ซึ่งถึงแม้อาการจะไม่ครบถ้วนสมบูรณ์ก็น่าจะสรุปได้ว่าเป็นโรคกระเพาะ ความผิดพลาดที่เกิดขึ้นนี้อาจแก้ไขได้โดยการถ่วงน้ำหนักของแต่ละอาการในลักษณะอาการในไลบรารีของกรณีให้ไม่เท่ากัน ซึ่งในปัจจุบันใช้วิธีให้ถ่วงน้ำหนักเท่ากันทุกอาการ

การถ่วงน้ำหนักอาการในลักษณะอาการให้ไม่เท่ากัน ตามความสำคัญของอาการในแต่ละกรณีจะช่วยให้สามารถชี้บ่งถึงลักษณะเด่นของแต่ละกรณี และลักษณะสถานการณ์ที่จะนำกรณีไปใช้ประโยชน์ได้ดียิ่งขึ้น แต่การที่จะถ่วงน้ำหนักให้สามารถใช้งานได้อย่างมีประสิทธิภาพ จะต้องอาศัยความรู้จากผู้เชี่ยวชาญ ซึ่งต้องทำการประเมินกรณีทุกๆ กรณี และระบุระดับการถ่วงน้ำหนักออกมาเป็นตัวเลขสำหรับทุกๆ อาการในทุกกรณี ซึ่งนอกจากจะต้องการเวลาในการวิเคราะห์ข้อมูลมากแล้ว ยังต้องอาศัยความรู้ และความร่วมมือจากคณะผู้เชี่ยวชาญ ตลอดเวลาที่จะทำการวิเคราะห์ข้อมูลเหล่านี้ด้วย ซึ่งจุดนี้เป็นจุดด้อยของการถ่วงน้ำหนัก เพราะไม่สามารถจัดหาแหล่งข้อมูลในระดับนี้ได้ (ไม่เคยมีหนังสือเล่มใดที่ระบุข้อมูลเชิงตัวเลข) และไม่สามารถจัดหาความร่วมมือจากกลุ่มผู้เชี่ยวชาญที่พร้อมจะสละเวลาจำนวนมากให้ได้ เพราะงานนี้มิใช้งานเชิงพาณิชย์

เงื่อนไขที่อาจทำให้เกิดความผิดพลาดในตัวผลลัพธ์ มักจะเกิดกับโรค หรือกรณีที่มีลักษณะจำเพาะบังคับ ซึ่งเมื่อขาดอาการใดอาการหนึ่งไปก็ถือว่าไม่ตรงกับโรคหรือกรณีนั้น เช่น ครรภ์เป็นพิษ จะต้องมีเกิดกับสตรีที่มีครรภ์เท่านั้น หากไม่มีครรภ์ก็ไม่สามารถเป็นโรคครรภ์เป็นพิษได้ หรือ โรค หรือกรณี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่มีลักษณะลำดับความสำคัญของแต่ละอาการไม่เท่ากัน นั่นคือ มีส่วนที่เป็นอาการหลัก และส่วนที่เป็นอาการเพิ่มเติม ที่อาจไม่มีก็ได้ เช่น โรคกระเพาะ ดังที่ได้กล่าวมาแล้ว อาการหลักคือ อาการปวดท้อง และปวดจุกบริเวณลิ้นปี่ ส่วนอาการปวดเป็นๆ หายๆ อาจไม่มีก็ได้ (โดยเฉพาะในรายที่เพิ่งจะเป็นไม่นาน จะไม่มีอาการนี้) ซึ่งการแก้ปัญหาทั้ง 2 กรณีนี้ อาจทำได้โดยการถ่วงน้ำหนักตามที่ได้กล่าวมาแล้ว หรืออาจจะใช้การทำพาร์ทิชัน ที่เคยได้กล่าวมาแล้วในเรื่องของการจัดโครงสร้างของกรณี มาช่วยในการแบ่งกลุ่มของโรคหรือกรณีที่ชัดเจน โดยจะต้องมีการทำดัชนีที่มีประสิทธิภาพ ที่จะสามารถแบ่งแยกกลุ่มของกรณีได้อย่างถูกต้อง และยังคงสามารถคงไว้ซึ่งความสามารถในการอ้างอิงถึงกลุ่มโรคที่ใกล้เคียงได้อย่างมีประสิทธิภาพ ซึ่งก็ต้องอาศัยข้อมูลที่ได้วิเคราะห์เป็นอย่างดีจากผู้เชี่ยวชาญ เช่นเดียวกับวิธีอื่นๆ ที่กล่าวมาแล้วข้างต้น

สรุปโดยรวมแล้ว ผลการทดสอบด้วยการใช้งานจริงค่อนข้างเป็นที่น่าพอใจของผู้ทำการทดสอบ (คุณหมออัครพันธ์) เพราะสามารถให้คำตอบเกินกว่าครึ่งที่ถูกต้อง ถึงแม้บางครั้งจะไม่ถูกต้องทุกประการหรือไม่ตรงตามความเห็นของคุณหมอก็ตาม คุณหมอได้ให้ความเห็นว่า ถึงแม้บางครั้งผลลัพธ์ที่ได้จะไม่ถูกต้องในระดับที่จะถือว่าเป็นคำวินิจฉัยที่ดี แต่ผลลัพธ์ก็อยู่ในเกณฑ์ที่ดี เมื่อเทียบกับขนาดและขอบเขตของความรู้ที่ระบบนี้มีอยู่ รวมทั้งในการพัฒนาครั้งนี้เป็นงานเชิงการศึกษา มิใช่งานเชิงพาณิชย์ จึงไม่สามารถจัดหาคณะผู้เชี่ยวชาญที่จะมาให้ข้อมูลได้ และไม่สามารถดำเนินการได้ในอีกหลายๆ ด้าน เนื่องจากข้อจำกัดทางด้านงบประมาณ ดังนั้นถ้าพิจารณาเฉพาะในสิ่งที่มีอยู่ ก็ถือว่าคำตอบที่ได้น่าจะเป็นที่พอใจสำหรับการใช้งานเป็นเครื่องมือช่วยแพทย์ในการวินิจฉัยโรคแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 11
บทวิจารณ์และสรุป

จากที่ได้ทำการออกแบบ และพัฒนาระบบผู้เชี่ยวชาญ โดยการอ้างเหตุผลเชิงกรณี ตามทฤษฎีที่ได้ทำการศึกษามาทั้งหมด จะเห็นได้อย่างชัดเจนเลยว่า การตัดสินใจในการเลือกที่จะออกแบบ และพัฒนาระบบไปในแนวทางใดก็ตาม ย่อมมีทั้งผลดีและผลเสียในขณะเดียวกัน ไม่ว่าจะใช้วิธีทั่วไป หรือวิธีที่เฉพาะเจาะจง ย่อมจะเกิดปัญหาทั้งสิ้น โดยเฉพาะในการออกแบบ เพราะสามารถเลือกลักษณะของการออกแบบของแต่ละส่วนได้หลายแบบ ซึ่งบางส่วนสมารถเลือกได้อย่างเป็นอิสระต่อกัน บางส่วนต้องพิจารณาถึงผลกระทบของการเลือกโครงสร้างในแต่ละแบบ ส่วนที่เห็นได้ชัดเจนที่สุดคือ โครงสร้างการจัดรวบรวมกรณีขึ้นเป็นไลบรารีของกรณี และวิธีที่ใช้ในการเรียกค้นและจับคู่กรณี ซึ่งถ้าหากการออกแบบในทั้งสองส่วนนี้ไม่สอดคล้องกัน กระบวนการในการค้นหาผลลัพธ์ย่อมไม่เกิดประสิทธิภาพ

นอกจากจะต้องพัฒนาระบบที่สามารถทำงานได้ตามหลักการแล้ว ส่วนที่สำคัญอีกส่วนหนึ่งก็คือการสร้างไลบรารีของกรณี ซึ่งถือว่ามีความสำคัญพอๆ กันกับระบบที่ใช้ในการอ้างเหตุผล เพราะไลบรารีกรณีก็ถือเป็นส่วนที่จะใช้วัดประสิทธิภาพของระบบเช่นกัน ดังนั้นอุปสรรคสำคัญก็คือ การขาดความเชี่ยวชาญในสาขาที่ทำการพัฒนา ซึ่งจะนำไปสู่ความเข้าใจผิดในโครงสร้างของปัญหา และอาจนำไปถึงการออกแบบระบบที่ไม่เหมาะสมต่อขอบเขตของปัญหาที่ต้องการได้

ในการศึกษาทฤษฎีเพื่อที่จะนำมาออกแบบและพัฒนาเป็นระบบ ล้วนเป็นทฤษฎีในเชิงบรรยายทั้งสิ้น นั่นหมายความว่าในการออกแบบโครงสร้างข้อมูล รวมไปถึงอัลกอริธึมในการทำงานของระบบต้องคิดเอาเองจากสิ่งที่หลักการได้กล่าวไว้ทั้งสิ้น เนื่องจากในหนังสือที่ค้นคว้าด้านทฤษฎีไม่ได้ระบุตัวอย่างที่อยู่ในรูปแบบของโปรแกรม หรือสูโดโค๊ด (Pseudo Code) ไว้ ดังนั้นในการออกแบบและพัฒนาจะได้ผลเป็นอย่างไร ย่อมขึ้นอยู่กับความเข้าใจของผู้ที่ทำการศึกษา เนื่องจากไม่มีแบบอย่างในการทดสอบความสำเร็จหรือความความล้มเหลวของโครงสร้าง และวิธีการแต่ละแบบ ในขอบเขตของปัญหาที่เราต้องการ

แนวทางที่อาจนำไปทำการพัฒนาต่อได้ อาจเป็นการขยายขอบเขตของระบบเดิม โดยเพิ่มคุณลักษณะที่ยังขาดไปในระบบนี้ เช่น การจัดเรียงกรณีในไลบรารีของกรณีแบบที่เป็นโครงข่าย (ควรใช้แบบรีดันแดนซี ดิสคริมเนชันเน็ทเวิร์ค จึงจะสามารถเพิ่มประสิทธิภาพในการเรียกค้นได้อย่างแท้จริง) การจับคู่ในการเรียกค้นโดยใช้ฮิวริสติกช่วยในการจับคู่ข้ามชนิดของฟีเจอร์ หรือใช้ฮิวริสติกช่วยในการประเมินระดับความเป็นประโยชน์ของกรณีต่อสถานการณ์ใหม่ (ซึ่งในปัจจุบันใช้วิธีเชิงจำนวน สามารถคำนวณได้ง่ายกว่า สร้างผลลัพธ์ ที่ค่อนข้างจะเป็นที่น่าพอใจ แต่ถ้าสามารถใช้ฮิวริสติกเข้ามาช่วย จะสามารถทำการประเมินได้มีประสิทธิภาพมากกว่า) หรือแม้แต่การนำวิธีการดัดแปลงผลลัพธ์ของกรณีที่ไม่ได้มีใช้ในระบบนี้เข้ามาเพิ่มเติม ซึ่งจะช่วยเพิ่มความหลากหลายให้กับการดัดแปลงมากขึ้น (เมื่อเพิ่มทางเลือกให้มากขึ้น ย่อมจะมีโอกาสในการสร้างคำตอบที่ดีได้มากขึ้น)

เมื่อทำการพัฒนาระบบจนเสร็จแล้ว ผลที่ได้คือระบบที่สามารถทำงานตามหน้าที่ ที่ได้ออกแบบโดยเลือกใช้หลักการบางส่วนในการพัฒนา เพราะไม่สามารถใช้หลักการทั้งหมดในระบบเดียวกันได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพราะบางหลักการก็เป็นหลักการที่ค่อนข้างจะขัดแย้งกัน ต้องเลือกอย่างใดอย่างหนึ่ง บางหลักการก็ไม่เหมาะสมกับขอบเขตของปัญหา ทำให้ระบบทำงานได้ตามหลักการส่วนหนึ่งซึ่ง หลักการที่ว่านี้ก็ได้พยายามออกแบบให้ตรงกับขอบเขตของปัญหามากที่สุด ผลที่ได้จากการพัฒนาระบบ ก็จะขึ้นอยู่กับการนำไปใช้อีกส่วนหนึ่ง เพราะการอ้างเหตุผลเชิงกรณีส่วนที่จะพัฒนาประสิทธิภาพของระบบได้ จะอยู่ที่การรับค่าป้อนกลับจากการใช้งานจริง ดังนั้นหากไม่ได้นำไปใช้งาน และป้อนกลับผลการใช้งาน ระบบก็จะไม่สามารถทำงานได้อย่างมีประสิทธิภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก
ตัวอย่างกรณีที่ใช้ในระบบ

กรณีที่ 1

ชื่อโรค เยื่อหุ้มสมองอักเสบ

รหัสโรค 1

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- คอแข็งเกร็ง

กรณีที่ 2

ชื่อโรค เยื่อหุ้มสมองอักเสบ

รหัสโรค 1

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- กระหม่อมโป่งตึง
- อายุต่ำกว่า 2 ปี

กรณีที่ 3

ชื่อโรค เยื่อหุ้มสมองอักเสบ

รหัสโรค 1

อาการ

- มีไข้
- มีผื่น หรือ มีตุ่มใสขึ้นเป็นบริเวณ
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- คอแข็งเกร็ง

กรณีที่ 4

ชื่อโรค เยื่อหุ้มสมองอักเสบ

รหัสโรค 1

อาการ

- มีไข้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- มีจุดแดงหรือจ้ำเขียว
- คอแข็งเกร็ง

กรณีที่ 5

ชื่อโรค เยื่อหุ้มสมองอักเสบ

รหัสโรค1

อาการ

- มีไข้
- มีอาการชัก
- คอแข็งเกร็ง

กรณีที่ 6

ชื่อโรค เยื่อหุ้มสมองอักเสบ

รหัสโรค1

อาการ

- มีไข้
- มีอาการชัก
- กระหม่อมโป่งตึง
- อายุต่ำกว่า 2 ปี

กรณีที่ 7

ชื่อโรค เลือดออกในสมอง

รหัสโรค2

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- คอแข็งเกร็ง

กรณีที่ 8

ชื่อโรค เลือดออกในสมอง

รหัสโรค2

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- กระหม่อมโป่งตึง
- อายุต่ำกว่า 2 ปี

กรณีที่ 9

ชื่อโรค ฝีในสมอง

รหัสโรค3

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- คอแข็งเกร็ง

กรณีที่ 10

ชื่อโรค ฝีในสมอง

รหัสโรค3

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- กระหม่อมโป่งตึง
- อายุต่ำกว่า 2 ปี

กรณีที่ 11

ชื่อโรค มาลาเรียขึ้นสมอง

รหัสโรค4

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- เข้าดงมาลาเรีย หรือทำการถ่ายเลือดในช่วง 1 เดือนที่ผ่านมา

กรณีที่ 12

ชื่อโรค มาลาเรียขึ้นสมอง

รหัสโรค4

อาการ

- มีไข้
- ชัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- เข้าดงมาลาเรียในช่วงระยะเวลา 1 เดือน

กรณีที่ 13

ชื่อโรค พิษสุนัขบ้า หรือเหตุร้ายแรงอื่นๆ

รหัสโรค5

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง
- ถูกสุนัขหรือแมวกัด

กรณีที่ 14

ชื่อโรค อาจเป็นสมองอักเสบ

รหัสโรค6

อาการ

- มีไข้
- ไม่ค่อยรู้สึกตัว หรือ อาเจียนรุนแรงหรือเป็นเลือด หรือ ปวดหัวรุนแรง

กรณีที่ 15

ชื่อโรค โปลิโอ

รหัสโรค7

อาการ

- มีไข้
- อ่อนแรงหรืออ่อนเพลีย หรือ เป็นอัมพาต

กรณีที่ 16

ชื่อโรค ไขสันหลังอักเสบ

รหัสโรค8

อาการ

- มีไข้
- อ่อนแรงหรืออ่อนเพลีย หรือ เป็นอัมพาต

กรณีที่ 17

ชื่อโรค โลหิตเป็นพิษ

รหัสโรค9

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อาการ

- มีไข้
- มีอาการช็อก
- มีไข้นานกว่า 1 สัปดาห์ หรือมีอาการติดเชื้อซัดเจน

กรณีที่ 18

ชื่อโรค โลหิตเป็นพิษ

รหัสโรค9

อาการ

- มีไข้
- ผิวซีด (โลหิตจาง)
- ม้ามโต (สามารถคลำได้)
- มีไข้นานกว่า 1 สัปดาห์
- มีอาการติดเชื้อซัดเจน

กรณีที่ 19

ชื่อโรค โลหิตเป็นพิษ

รหัสโรค9

อาการ

- มีไข้
- ผิวซีด (โลหิตจาง)
- ม้ามโต (สามารถคลำได้)
- ตับโต (สามารถคลำได้)
- มีไข้นานกว่า 1 สัปดาห์
- มีอาการติดเชื้อซัดเจน

กรณีที่ 20

ชื่อโรค โลหิตเป็นพิษ

รหัสโรค9

อาการ

- มีไข้
- มีจุดแดงหรือจ้ำเขียว
- ตับโต (สามารถคลำได้) หรือ ม้ามโต (สามารถคลำได้)
- มีไข้นานกว่า 1 สัปดาห์ หรือ ผิวซีด (โลหิตจาง) หรือ ผิวเหลือง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กรณีที่ 21

ชื่อโรค โลหิตเป็นพิษ

รหัสโรค9

อาการ

- มีไข้
- ตาเหลือง
- มีอาการติดเชื้อชัดเจน

กรณีที่ 22

ชื่อโรค ช็อกเนื่องจากไข้เลือดออก

รหัสโรค 10

อาการ

- มีไข้
- ช็อก
- มีไข้มานาน 3-4 วัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1. Elias M. Awad , " Building Expert System : Principle , Procedures , and Application " , West Publishing Company , 1995
2. James Martin , Steven Oxman " Building Expert Systems : The Development and Implementation of Rule - Based Expert Systems " , Prentice - Hall International , Inc
3. Dmitry Korenkov , Http://www.eas.asu.edu/~drapkin/556/mycin.html , January 29 , 1996
4. Janet Kolodner, "Case-Based Reasoning" : Georgia Institute of Technology, Morgan Kaufmann Publishers , Inc.,1993
5. Joe Phillips , Http://krusty.eecs.umich.edu/people/josephp/quals/systems/mycin
6. นายแพทย์สุรเกียรติ อาชานานุภาพ ,"ตำราการตรวจรักษาโรคทั่วไป เล่ม 1 : หลักการวินิจฉัยและรักษาโรค " , Folk Doctor Publishers , 1988

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้