自動化船의 衝突豫防專門家시스템 開發에 관한 硏究

김 시 화*

On the Development of Prototype Expert Collision Avoidance System of Automated Ship

Si-Hwa Kim

〈목 차〉

Abstract

This paper intedns to develop a Prototype Expert Collision Avoidance System by introducing expert system techniques into the decision block of anti-collision loop. The problem domain of this study is characterized and specified by combining the concepts of anti-collision loop and knowledged-based system for collision avoidance.

Domanin knowledge which may oriiginates from the appropriate sources such as the International Regulations for Preventing Collision at Sea 1972, Marine Traffic Laws, and many texts on the subject of anti-collision navigation and good seamanship, is acquired and formalized into the knowledge-base system using production rule.

Finally, a Prototype Expert Collision Avoidance System is built by using the CLIPS, developed by AIS

* 正會員, 韓國海洋大學

at NASA written in and fully integrated with the C language, and some test-and-run results of the system are demonstrated and examined. The author considers the proposed system which is named PECAS to be meaningful as a test bed for a further refined Expert Collision Avoidance System on board the Automated Ship.

1. 서 론

해상의 선박충돌사고는 인명 및 재화의 손실은 물론 해양자연환경에도 심각한 위협을 초래한다. 국제해상충돌방지규칙 제7조(b)는 레이다가 설치되어 이를 사용할 수 있는 선박에 대하여, 충돌사고를 방지하기 위해 레이다 장치를 적절히 사용하여야 할 것(Proper use shall be made of radar equipment ..)을 주의의무로 명시하고 있다.[1] Anna Salen호의 선박충돌사고에 관한 판례에서도 레이다는 지적으로 사용하고 또한 지적으로 해석하는 경우에만 효용성이 있다고 지적하여, 레이다의 「지적인 사용(Intelligent use)」의 중요성을 부각하였다.[2] 종종 최악의 레이다 원조 충돌사고(The worst radar-assisted collision)로 인용되는, 멕시코만 북위 26° 18′, 서경 91° 26′ 해상에서 발생한 바 있는 미국의 두 유조선 간의 충돌사고는, 39명의 선원의 생명을 앗아갔으며 그 충돌사고에서 겨우 살아남은 항해사들은, 세번에 걸친 충돌회피 조선 후에 일어난 당시의 사고에 관하여 이렇게 진술하고 있다. "I just stood there and looked at the oncoming echo and could not move." 대부분의 이러한 선박의 해상충돌사고는 불과 수십분 내에 벌어지고 만다.[3]

레이다는 선박에 탑재된 충돌예방을 위한 부적이 아니다. 레이다로 부터 얻은 데이타를 지적으로 사용하여, 이를 바탕으로 충돌예방 조선법을 숙지하고 있는 항해사가 적절히 조선할 때에 비로소 레이다는 유효한 항해장비(Navigational aid)가 된다. 하지만 항해사의 전문적인 지식과 경험의 차이에 따르는 human factor는 여전히 문제가 된다. 이러한 human error에 의한 사고를 줄이려는 노력의 결과로 개발된 것이 ARPA(Automatic Radar Plotting Aids)이다. 하지만 레이다에 부착, 설치되는 ARPA의 역할은, 레이다로 부터 여타 선박의 항행에 관한 데이타를 신속, 정확하게 추출하고 분석하여 그 출력정보를 표시하여 줌으로써, 시간이 소요되는 성가신 플롯팅 작업을 해소하여 주는 것이며, 항해사의 의사결정에 관한 임무까지 대신하는 것은 물론 아니다.[4]

컴퓨터의 기술이 발전하면서 한걸음 더 나아가, 충돌예방조선에 관한 의사결정까지 포함시켜 충돌예방 문제를 해결하고자 하는 전산 프로그램의 개발에 관하여, 지금까지 주로 알고리즘적인 접근법에 의한 연구가 이루어져 왔다.[5] 그러나 여러가지 제약조건 하의 다양한 복수선박의 조우상황을 만나게 되면 충돌예방 문제의 수학적 모형이 대단히 복잡해지는 까닭에, 알고리즘적인 접근법은 비효율적이다. 최근에는 이러한 문제점를 전문가시스템(Expert system) 기법으로 해결하고자 하는 몇몇 연구가 이루어지고 있다.[6),7]

이 논문은 충돌예방시스템에 전문가시스템의 기법을 도입하여, 하나의 프로토타이프 충돌예방 전문가시스템을 개발하는 것이 그 목적이다. 연구방법으로는, 먼저 연구대상인 충돌예방시스템의 개요를 살펴보고 이 논문에서 다룰 문제의 범위를 설정하여, 그에 따른 충돌예방전문가시스템의 개념을 제시한다. 다음에는 다양한 유형의 해상교통상황에 적용될 충돌예방조선법의 의사결정규칙들을, 국제해상충돌방지규칙(1972), 해상교통법론, 충돌예방 레이다항법 및 기타 충돌예방 조선법에 관한 전문 항해사의 지식과 경험을 바탕으로 개발하고, 이러한 의사결정규칙들을 적절히 분류하여 충돌예방조선을 위한 지식베이스를 구축한다. 이러한 지식베이스를 바탕으로, 전문가시스템 개발을 위한 도구로써 전문가시스템 쉘(Shell) 중의 하나인 CLIPS를 이용하여 PC 상에서 실행가능한 프로토타이프 충돌예방전문가시스템(Prototype Expert Collision Avoidance System)을 개발하고자 한다.

2. 충돌예방전문가시스템의 개요

2.1 충돌예방시스템의 개념과 ARPA의 역할

그림 1은 충돌예방시스템의 충돌예방 루프를 개념적으로 보여주고 있다. 레이다 상의 상대선들(Targets)과 자선(Own ship)이 결합하여 기하학적으로 선박조우상황(Ship encounter situation)을 형성한다. 항해사는 시각관측이나 레이다를 통하여 상대선들에 관한 데이타를 거리, 방위 및 관측시각 등과 같이 평가할 수 있는 데이타로 정리하여 플롯터 상의 전형적인 작도(plot)등과 같은 형태로 데이타베이스에 저장한다. 이렇게 저장된 데이타를 분석하여 상대선의 상대운동(Relative motion), 최근접점(CPA : Closest Point of Approach), 최근접시각(TCPA : Time of CPA) 및 최근접거리(DCPA : Distance at CPA) 등을 구한다. 상대선의

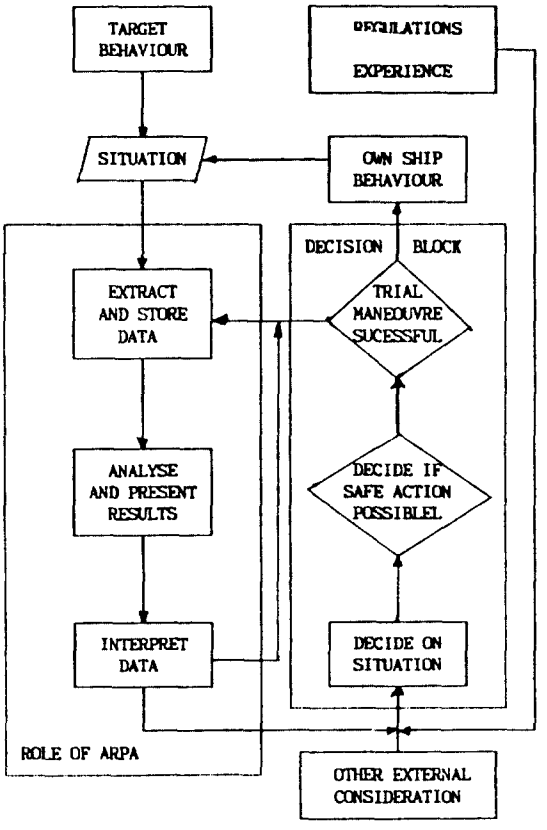

Fig. 1 The Anti-collision Loop

진운동(True motion)을 구하려면 자선의 침로와 속력을 도입하여 벡타 삼각형을 해석하면 된다.

이러한 정보들이 플롯터 상에 적절한 형태로 표시되면 선박조우상황의 해석이 끝난 것이다. 이제 이러한 정보들을 바탕으로 의사결정단계에 들어갈 준비가 되었다. 항해사는 자신의 전문적인 경험과 국제해상충돌방지규칙의 제규정들, 그리고 자선의 조선 및 지리적 제약등을 통합적으로 고려하여 충돌예방을 위하여 필요할 경우 취하여야 할 조선동작을 결정한다. 마지막으로 충돌예방조선을 실시하기 전에 데이타 분석을 위한 블록에 다시 들어가, 조선동작의 실시 후의 상황을 분석하는 과정을 거친다. 마침내 충돌예방조선을 실시하게 되고 충돌예방 루프는 새로이 반복된다.

ARPA의 역할은 그림 1의 점선 내부에 해당한다. 이미 관찰한 바 대로 ARPA시스템이 하는 역할의 전부는 레이다로부터 상대선들의 데이타를 추출하여 데이타베이스에 저장하고 이 데이타베이스를 바탕으로 관련정보들을 구하여 항해사에게 벡타 형태와 같은 요약된 정보를 표시하여 주는 일이다. 대개의 ARPA시스템은 시험조선의 조건(Trial condition)을 입력으로 받아들여 그 조선동작으로 인한 결과를 보여준다. ARPA시스템은 항해사의 충돌예방조선에 관한 의사결정의 임무까지 대신하는 것은 물론 아니지만, 상대선들에 관한 레이다 데이타에 대하여 시간이 소요되는 성가신 플롯팅 작업을 해소하면서 신속하고 정확하게 분석된 출력정보를 표시하여 줄 뿐만 아니라, 시험조선동작에 대한 신속한 재평가 결과를 제공함으로써 항해사들이 충돌예방조선을 위한 의사결정을 내리는 데 매우 큰 도움을 주는 항해장비라 하겠다.[8),9)]

2.2 전문가 시스템의 개요

산업혁명이 인간의 노동을 기계로 대체한 기술혁신이라 한다면, 컴퓨터 혁명은 인간의 두뇌를 기계로 대체하고자 하는 기술혁신이라 할 수 있다. 가장 낙관적인 과학자들의 예측도 훨씬 못 미칠 정도로 발전한 컴퓨터의 기술-비유컨데, 만약 자동차의 가격과 기술발전이 컴퓨터 하드웨어 분야의 경우와 보조를 같이 하였다면 우리는

오늘날 Rolls-Royce를 미화 2.75달러에 살 수 있고, 1 갤론의 가솔린으로 3백만 마일을 달릴 수 있다.[10]-에 힘입어 1970년대 초 전문가 시스템이 태동하였다. 컴퓨터 하드웨어 전문가들이 마이크로칩의 기술을 발전시키는 동안 소프트웨어 전문가들은 Artificial Intelligence(AI)로 알려진 컴퓨터 과학의 신생분야의 획기적인 개념전환의 초석을 닦아 온 것이다.

AI과학자들의 목표는 어떤 면에서 지적인 인간이 하듯이 문제를 생각하고 해결할 수 있는 컴퓨터 프로그램을 개발하는 데 있다. 초기의 AI과학자들은 광범위한 유형의 문제를 해결하는 일반적인 방법을 발견하여, 그 복잡한 사고의 과정을 범용 컴퓨터 프로그램으로 재현하여 보려고 노력하였지만 거의 무의였다. 단일 프로그램이 처리하여야 할 문제의 유형이 많을수록, 프로그램의 수행능력은 훨씬 더 빈약해진다는 사실 때문에, **지적인 컴퓨터 프로그램(Intelligent** computer program)을 개발하는 또 다른 방법을 모색하게 된 것이다. 최근의 AI 연구는 매우 좁은 문제영역에 관한 고도의 전문적 지식을 사용하여 대단히 전문화된 컴퓨터 프로그램을 개발하는 데 그 촛점을 두게 되었다. 전문가시스템이란 바로 좁은 문제영역에서 높은 수행능력을 달성하기 위해 고도의 전문가 지식을 사용하는 컴퓨터 프로그램이다.[11]

T. C. Chang은 이러한 전문가시스템을 재래의 컴퓨터 프로그램과 비교하여 다음과 같이 정의하고 있다.[12]

1) 전문가시스템은 지식자체와 지식을 사용하는 방법을 구분한다.

2) 전문가시스템에서 지식은 부호화되며, 시스템은 기호로 추리할 수 있는 능력이 있어야 한다.

3) 전문가시스템은 자신의 추리과정을 설명할 수 있어야 한다.

4) 전문가시스템은 인간 전문가와 동등한 기능을 수행할 수 있어야 한다.

그림 2는 이러한 전문가시스템의 구조를 보여주고 있다.[13] 그림에서 보듯이 전문가시스템은 지식베이스와 추론기관의 주요한 두 구성요소를 가지고 있다.

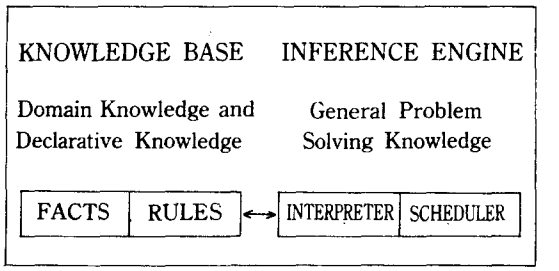

Fig. 2 The Structure of Expert System

1) 지식베이스(Knowledge base)

해당 문제영역의 선언적 지식(Declarative knowledge)을 포함하여 문제의 해결을 가능하게 하는 영역지식(Domain knowledge)들로 이루어지며, 전문가시스템에서 가장 중요한 부분으로 '사실(facts)' 및 '규칙(rules)'으로 구성되어 있다. 이러한 지식들은 완전하고(complete), 애매모호하지 않으며(unambiguous), 효율적이고(efficient), 모순되지 않아야(compatible)하며, 사용가능한 형태로 표현되어야 한다.

2) Inference engine(Control system)

추론기관(Inference engine)은 일반적인 문제해결지식(general problem-solving knowledge)으로 이루어져 있으며, 해석기(Interpreter)와 Scheduler라 부르는 내장 탐색알고리즘으로 이루어져 있다.

2.3 충돌예방 전문가시스템 (Prototype Expert Collision Avoidance System)

이상에서 우리는 충돌예방 루프의 개념도와 함께 충돌예방시스템의 개요를 살펴보았으며, 또한 전문가시스템의 구조를 이해함으로써 이 논문에서 다룰 충돌예방 전문가시스템의 개념을 그림 3과 같이 제시할 수 있게 되었다. 그림 3은 그림 1의 충돌예방 루프 중에서 의사결정 불록이 지식베이스 시스템(Knowledge-based system)으로 대체되었음을 알 수 있다. 이 지식베이스시스템은 ARPA 시스템으로 부터 분석된 선박조우상황을 서술해 주는 사실(facts)과 국제해상충돌방지규칙(1972), 해상교통법론, 충돌예방 레이다항법 및 기

타 충돌예방 조선법에 관한 전문 항해사의 지식과 경험을 바탕으로 개발된 충돌예방조선법의 의사결정규칙들, 그리고 이 지식베이스 시스템을 제어하는 추론기관으로 이루어져 있다.

그림 3에서 제시한 시스템은 ARPA 시스템에서 분석된 모든 정보들이 지식베이스에 직접 인터페이스 되도록 되어있으나, 이러한 데이타의 인터페이스 문제는 별도의 과제로 남겨두고, 이 논문에서는 ARPA 시스템이 제공해 주는 선박조우상황에 관한 제반 정보를 적절한 서술자(Descriptor)로 나타내어, 지식베이스 시스템에 대화형으로 입력할 수 있도록 프로토타이프 충돌예방 전문가 시스템을 구축하고자 한다.

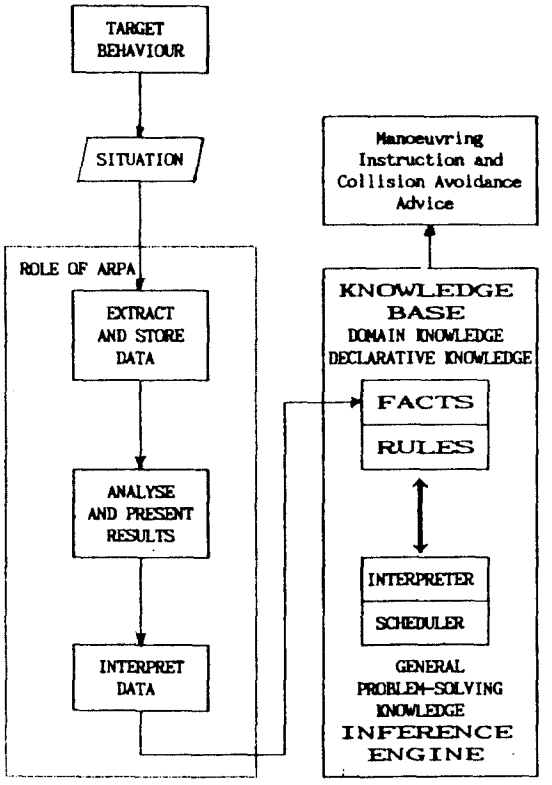

Fig. 3 Block Diagram of Prototype Expert Collision Avoidance System.

3. 충돌예방조선을 위한 의사결정규칙의 개발

3.1 선박조우상황 및 조선동작의 세분

(1) 기본적인 선박조우형태의 구분

충돌예방조선을 위한 의사결정규칙을 개발하기 위하여 우선 해상교통상황에서 있을 수 있는 선박조우상황을 간명하게 구분할 필요가 있다. 그림 4는 Keystone system에서 구분한, 레이다에 기록될 수 있는 8가지 기본적인 선박조우상황을 나타내고 있다.[14]

이러한 8가지 가능한 선박조우 상황 중에서 자선이 항행중일 때만 생각하면 6가지로 압축된다. 한편, 국제해상충돌방지규칙 상의 피항선과 유지선의 관계를 고려하면 교차하여 오는 상대선은, 우현으로부터 좌현으로 교차하여 오는 경우와 좌현으로부터 우현으로 교차하여 오는 경우로 구분할 필요가 있다. 그리고 자선과 동일침로, 동일속력으로 나란히 항행하는 상대선의 경우는, 서로 상대선의 진로를 방해할 수 없으므로 자선의 입장에서 추월 당하는 상대선에 포함시킬 수 있다. 이상의 내용을 정리하면 다음과 같은 6가지 유형으로 선박조우상황을 구분할 수 있으며 레이다 데이타를 서술할 때 이러한 분류가 유효함을 알 수 있다.

1) 우현에서 좌현으로 교차하여 오는 상대선 [XSP]
   (Target crossing from starboard to port)
2) 우현에서 좌현으로 교차하여 오는 상대선 [XPS]
   (Target crossing from port to starboard)
3) 반대침로로 마주오는 상대선[THO]
   (Target on reciprocal course)
4) 자선을 추월하는 상대선[TOO]
   (Target overtaking ownship)
5) 자선이 추월하는 상대선[OOT]
   (Ownship overtaking target)
6) 정지하여 있는 상대선[TST]
   (Target stopped)

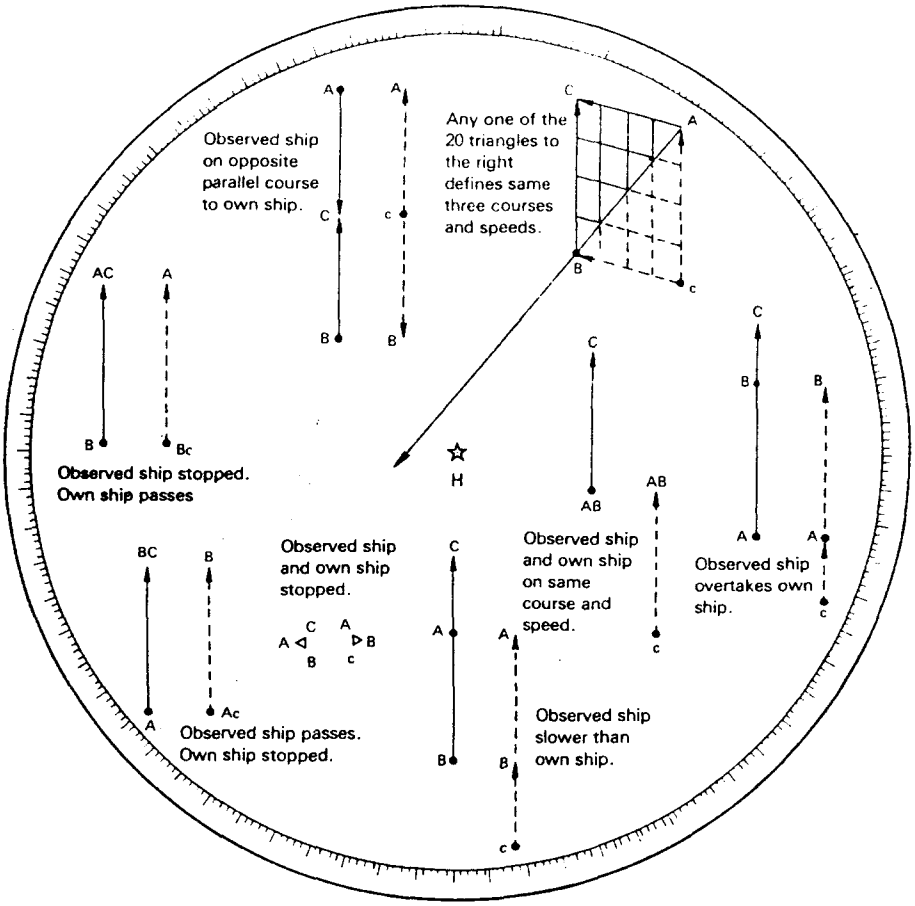

Fig. 4 Eight Basic Situations Recorded on the Radar

(2) 레이다 데이타의 서술

레이다 데이타는 Keystone system의 벡타 삼각형 해석에 의해 시스템에 활용할 수 있는 표준정보로 변환한다. 벡타 삼각형 해석에 의하면 상대선의 진침로, 진속력, 상대침로, 상대속력, 최근접시각 및 최근접거리 등에 관한 정확한 수치정보를 얻을 수 있으며, 전술한 바 대로 ARPA는 이 작업을 신속, 정확하게 처리하여 준다. 그러나 국제해상충돌방지규칙의 선박조우상황 서술방법 뿐만 아니라, 전문항해사들이 실제로 익숙해 있는 선박조우상황의 표현방법은 상대선 A의 진침로 000°, 진속력 17kt 등의 표현보다는 『2시 방향으로 보이는 상대선 A는 우현에서 좌현으로 교차하여 오고 있으며, 최근접시각은 여유가 있는 편이나, 본선 선수방향으로 다소 위험한 최근접거리를 가지며 통과한다』와 같은 표현들이다.

이 논문에서는 충돌예방조선 전문가시스템을 개발하면서, 지식베이스를 구축하기 위한 의사결정규칙들은 If-then 형식의 Production rule을 사용하고자 하므로, ARPA가 제공해 주는 정보들을 적절한 서술자로 표현하고자 한다. 다음은 레이다 데이타를 서술자를 이용하여 표현한 실례이다.

[레이다 데이타의 서술자표현 실례]

[XSP 1 c1 danger hurry up no ahead]
[XPS 1 c10 safe easy no astern]

[THO 1 p0 danger hurry up yes none]
[TOO 1 c5 safe easy no ahead]
[OOT 1 c0 danger hurry up yes none]
[TST 1 c1 safe easy yes none]

이상의 예들은 [선박조우형태, 식별번호, 인지방향, DCPA, TCPA, 추가서술]의 형태로 구성된 것이며, 이러한 서술은 일의적이기 보다는 전문가시스템을 개발하기 위해 사용할 도구가 무엇인가에 따라 달라질 수 있다.

(3) 충돌예방 조선동작의 세분

해상에서의 여러가지 유형의 선박조우상황에 대처하여 적절한 충돌예방 조치를 취하려면, 국제해상충돌방지규칙(1972) 및 해상교통법의 내용을 숙지하고 있어야 할 뿐만 아니라, 충돌예방 레이다항법 및 기타 충돌예방 조선법에 관한 전문 항해사의 지식과 경험을 요한다. 1958년 "The Ten-Second Anti-Collision Radar Navigation System" 이라는 이름의 충돌예방을 위한 간결한 레이다항법시스템이 창안되기 이전에는 사실상 체계적인 충돌예방절차가 알려져 있지 않았다.[15] 해상에서 항해사가 충돌예방을 위해 취할 수 있는 가능한 조선동작을 적절히 세분하려면, Keystone system과 같은 체계적인 레이다 항법을 명확하게 파악할 필요가 있다.

여기서는 항해사가 취할 수 있는 가능한 충돌예방 조선동작에 대하여 Coenen(1989) 등이 제안한 내용을 보완하여 다음의 6가지 유형으로 크게 구분하였으며,[16] 각 유형의 조선동작에 대하여 상황의 완급 기타의 제조건들을 고려한 세분된 조선동작으로 분류하였다. 그 중에서도 신속충돌예방조선으로 구분한 조선동작은 DeWit(1981)가 제안한 충돌예방조선을 원용하여 지칭한 것이다.[17]

1) 침로유지조선(The Stand-on Action) ;
   Stand on(stand-on)
2) 우현변침조선(Starboard Alterations)
   Alter Course to Stb'd (alt__stbd)
   Alter Course to Stb'd and parallel the target (stbd__parallel)
   Alter Course to Stb'd to place stern on target (stbd__stern__on)
   Alter Course 40 degrees to Stb'd immediately (stbd__40)
   Alter Course 90 degrees to Stb'd immediately (stbd__90)
3) 좌현변침조선(Port Alterations)
   Alter Course to Port (alt__port)
   Take a Round Turn to Port (port__round__turn)
   Alter Course to Port to place stern on target (port__stern__on)
   Alter Course 40 degrees to Port immediately (port__40)
   Alter Course 90 degrees to Port immediately (port__90)
4) 신속충돌예방조선(Fast and Clear Collision Avoidance Actions)
   Fast and Clear Collision Avoidance Action 1 (fcca__action__1)
   Fast and Clear Collision Avoidance Action 2 (fcca__action__2)
5) 감속조선(Speed Reductions)
   Speed Reduction to Half Ahead(reduce__spd__to__half)
   Speed Reduction to Slow Ahead(reduce__spd__to__slow)
6) 긴급조선(Emergency Action)
   Emergency (emerg)

3.2 의사결정규칙의 개발과 지식베이스 구축

(1) 충돌예방 의사결정규칙의 개발

전문가 시스템이 통상의 컴퓨터 프로그램과 구별되는 기본적인 차이점은, 재래의 프로그램이 데이타를 조작하고 처리하는 반면에 전문가 시스템은 지식을 처리한다는 점이다.[18] 지식공학(Knowledge engineering) 또는 지식처리기술(Knowledge processing technology) 등의 연구분야가 이러한 문제를 다룬다.

충돌예방 전문가 시스템을 개발하려면, 선박조우상황에서 대처하는 전문항해사들의 사고의 흐름, 의사결정과정 및 처리해 내는 입출력정보들을 조사하고 분석하여야 할 뿐만 아니라, 국제해상충돌방지규칙(1972), 해상교통법론 등의 참고문헌

및 해난사고의 판례, 그리고 충돌예방 레이다항법 및 기타 충돌예방조선법에 관한 제반 전문지식을 출처로 하여 지식을 획득하고, 이들을 지식공학적인 기법으로 구조화하여 지식베이스를 구축하여야 한다. 이러한 지식획득의 과정을 예시하여 주는 것이 그림 5이다.[19]

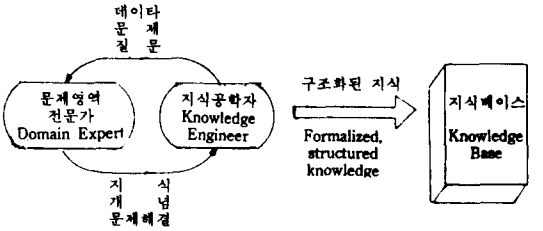

Fig. 5 Typical Knowledge Acquisition Process for Building an Expert System.

이와 같은 지식공학적인 접근법으로 충돌예방 전문가시스템의 지식베이스를 구축할 경우, 충돌상황에 대처하는 전문항해사들의 사고의 흐름이나 의사결정과정 및 입출력정보의 처리방법들에 관한 면담도 중요하겠지만, 이외의 방법으로 해난사고의 판례 및 해상교통법론 등의 출처에서도 중요한 지식들을 획득할 수 있다. 이렇게 획득한 지식들을 바탕으로, 충돌예방을 위한 조선동작과 요주의 사항을 선정하는 규칙을 Production rule 시스템에 적합한 If-then 형식으로 표현한 예를 소개하면 다음과 같다.

[충돌예방 의사결정 규칙의 예]

If (Stbd-Port Crossing 1시, 2시 방향의 상대선들이 있다.) and
(각각의 선박의 DCPA has-danger) and
(각각의 선박의 TCPA
XSP T-1 take-it easy
XSP T-2 hurry-up) and
(Target-Overtaking from stbd-quarter, Alt-stbd has-danger) and
(Port-Stbd Crossing 10시 방향의 상대선이 있다.) and
(XPS DCPA has-danger)
then (Fast-and Clear Port-roun-turn) with caution to
"Target-Overtaking from stbd-quarter, Alt-stbd has-danger"
Or,
(Reduce-speed until Alt-stbd has-safe) and
(Alt-stbd 90 immediately) with caution to
"XSP T-2 hurry-up and
Target-Overtaking from stbd-quarter, Alt-stbd has-danger")

(2) 충돌예방 지식베이스 구축

전문가시스템에서 지식을 표현하는 가장 보편적인 방법으로 Rules, Frames 및 Sematic nets 등을 들 수 있는데, 이 연구에서 충돌예방 전문가시스템을 개발하기 위해 사용하는 전문가시스템 쉘 CLIPS는 Production rule의 형식으로 지식을 표현한다.[20] NASA/Johnson Space Center(JSC)의 Artificial Intelligence Section(AIS)은 NASA의 사용자들에게 LISP-based 전문가 시스템을 공급하여 오면서, 수많은 문제점을 겪어 왔다. 그 대표적인 세가지 문제점으로 일반 컴퓨터들에의 낮은 응용성, 현용 LISP 툴(tool) 및 하드웨어가 고가라는 점, 그리고 LISP와 타언언와의 빈약한 통합성 등을 들 수 있다. 이러한 문제점들을 해결하기 위하여 AIS에서 C언어로 작성하여 C언어와의 완벽한 통합성을 갖도록 개발한 전문가시스템 툴(tool)이 곧 CLIPS('C' Language Integrated Production System)이다.[21]

규칙은 Ada 또는 Pascal 등과 같은 절차적 언어의 If-then 문장과 유사하다. 이러한 규칙들은 조건들이 만족되면 실행되는 수행의 모임이라 할 수 있다. 여기서는 PECAS의 지식베이스 구축에 사용된 몇몇 규칙들이 CLIPS로 표현된 예를 보인다.

1) 최우선으로 고려해야 할 선박조우형태의 결정규칙 표현

```
(defrule select-XSP-module
        (declare (salience -1))
        (select-module)
        (a XSP $?)
=>
```

```
        (assert (module 1))
)
(defrule select-OOT-module
        (declare (salience-1))
        (select-module)
        (not (a XSP $?))
        (not (a THO $?))
        (a OOT $?)
=>
(assert (module 5))
```

2) 충돌예방 조선동작의 결정규칙 표현

```
(defrule XSP-facc-stbd
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (not (a TOO ?i c5 d h $?))
        (not (a TOO ?i c5 $? h $?))
=>
        (assert (XSP-facc-stbd))
)
(defrule XSP-alt-port-stern-on
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (a TOO ?i c5 $? h $?)
        (not (a XPS $?))
        (not (a TOO $?))
=>
        (assert (XSP-alt-prot-stern-on))
)
(defrule XSP-alt-port-round-turn
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (a TOO ?i c5 $? h $?)
        (a XPS $?)
        (not (a TOO ?i c7 d h $?))
        (not (a TOO ?i c7 $? h $?))
=>
        (assert (XSP-alt-port-round-turn))
)
```

4. 충돌예방조선 전문가시스템의 개발

4.1 시스템 소프트웨어의 구성

이 연구에서 개발된 프로토타이프 충돌예방 전문가시스템 PECAS는, 항해하는 선박이 해상에서 접하는 여러가지 유형의 선박조우상황에 대하여, 행해사가 취하여야 할 적절한 충돌예방조선법과 그에 따르는 요주의 사항들을 자동적으로 출력해 주는 대화형 충돌예방 전문가시스템이다.

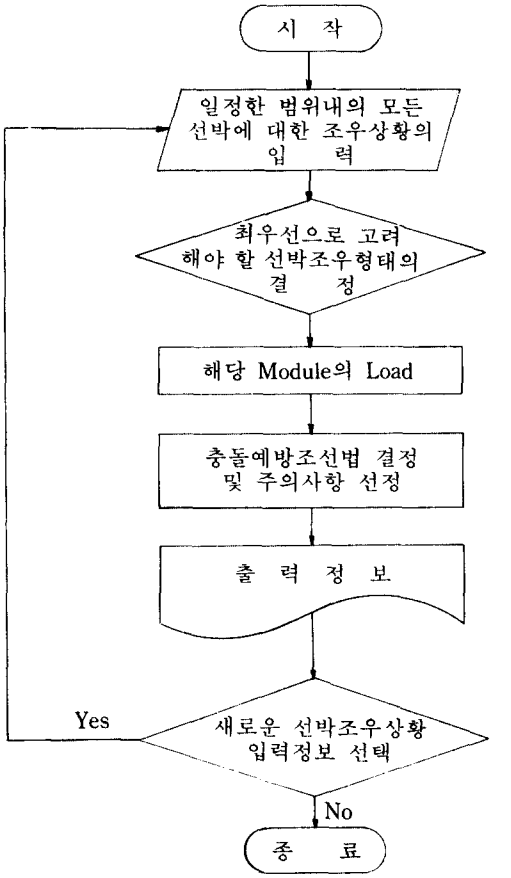

Fig. 6 Flow Chart of Interactive PECAS

그림 6에서 예시하는 바와 같이 PECAS는 일정한 거리내의 모든 선박들에 대한 ARPA데이타를 대화형으로 입력 받아서 이들 중 충돌예방을 위해 최우선으로 고려해야 할 선박조우 형태를 결

정하여 그에 해당되는 Module를 load한다. 이렇게 선정되는 해당 Module은 기타 선박들의 모든 조우상황을 고려하여 실시하여야 할 충돌예방조선법을 결정하는 규칙들을 포함하고 있다.

4.2 시스템 실행 예

이상에서 설명한 대화형 PECAS는 XSP.CLP, XPS.CLP, THO.CLP, TOO.CLP, OOT.CLP, TST.CLP 등의 6개의 Module을 가지고 있다. 개발한 시스템에 대하여 5척의 복수선박 조우상황을 입력한 결과, 그 중에서 충돌예방을 위해 최우선으로 고려해야 할 선박조우 형태가 우현에서 좌현으로 교차하여 오는 상대선(Target crossing from starboard to port)으로 나타난 경우의 스템의 실행예를 여기에 나타내 보인다.

(1) 선박조우형태의 입력부분

아래에서 보이는 대화형 입력 부분은 설명이 없어도 이해할 수 있으리라 생각한다.

```
CLIPS (run)
==================================================

Prototype Expert Collision Avoidance System
1991.  6.  29

PECAS : Edited by Kim, Si-Hwa
This system generates collision avoidance
maneouvring instruction, if any, and then
gives output to the navigator with advices.

To execute, merely load, reset and run.
==================================================

PECAS CLASSIFY SHIP ENCOUNTER TYPES
AS FOLLOWS

[XSP] Target Crossing from stbd to port
[XPS] Target Crossing from port to stbd
[THO] Target Heading On
[TOO] Target Overtaing Ownship
[OOT] Ownship Overtaing Target
[TST] Target Stopped

==================================================
ANY TARGET CROSSING FROM STBD TO
PORT ? 〈y/n〉y
ANY TARGET CROSSING FROM PORT TO
STBD ? 〈y/n〉y
ANY TARGET HEADING ON ? 〈y/n〉y
ANY TARGET OVERTAKING OWNSHIP ? 〈y/n〉y
IS THE OWNNSHIP OVERTAKING ANY TAR-
GET ? 〈y/n〉n
ANY TARGET STOPPED ? 〈y/n〉n

--------------------------------------------------
How many targets are crossing from stbd to port ? 2
--------------------------------------------------

How's the apparent aspect of XSP target 1
1, 2, or 3 o'clock ? [c1/c2/c3] ? c1
How's the DCPA of XSP target 1
safe or dangerous ? [s/d]d
How's the TCPA of XSP target 1
easy or hurry up ? [e/h]h
Is XSP target 1 restricted ? [y/n]n
How's the passing aspect of XSP target 1
passing ahead or astern ? [ah/as] ah
```

(2) XSP Module의 Loading 및 의사결정 부분

XSP Module이 load되면서 f-0, f-1…으로 표시된 사실(facts)들과 Matching이 되어 실행된 규칙들이 Agenda에 대기한다.

```
--------------------------------------------------
DO MODULE XSP
--------------------------------------------------

==〉 f-0    (initial-fact)
==〉 f-1    (Do-Module XSP)
==〉 f-2    (XSP-encounter y)
==〉 f-3    (XPS-encounter y)
==〉 f-4    (THO-encounter y)
==〉 f-5    (TOO-encounter y)
==〉 f-6    (OOT-encounter n)
==〉 f-7    (TST-encounter n)
==〉 f-8    (number-of-XSP 2)
==〉 f-9    (a XSP 1 c1 d e n ah)
==〉 f-10   (a XSP 2 c2 d h n ah)
==〉 f-11   (number-of-XPS 1)
==〉 f-12   (a XPS 1 c10 d h n ah)
==〉 f-13   (number-of-THO 1)
==〉 f-14   (a THO 1 p0 d h n 0)
==〉 f-15   (number-of-TOO 2)
```

==> f-16 (a TOO 1 c5 d h n ah)
==> f-17 (a TOO 2 c7 d e n ah)
==> f-18 (values-for id 1 2 3 4)
==> Activation 5 phase-id : f-1, f-18

CHECK BAD INFORMATION이라 되어 있는 부분은 현재의 사실(facts)들 중 입력의 오류등으로 인해 생긴 정보들이 없는지 검정하는 부분이다.

CHECK BAD INFORAMTION

FIRE 1 phase-id : f-1, f-18
<== f-18 (values-for id 1 2 3 4)
==> f-19 (values-for asp c0 c1 c2 c3 c4 ...)
==> Activation 5 phase-asp : f-1, f-19
FIRE 2 phase-asp : f-1, f-19
<== f-19 (values-for asp c0 c1 c2 c3 c4 ...)
==> f-20 (values-for dcpa s d)
==> Activation 5 phase-dcpa : f-1, f-20
FIRE 3 phase-dcpa : f-1, f-20
<== f-20 (values-for dcpa s d)
==> f-121 (values-for tcpa e h)
==> Activation 5 phase-tcpa : f-1, f-21
FIRE 4 phase-tcpa : f-1, f-21
<== f-21 (values-for tcpa e h)
==> f-122 (values-for res y n)
==> Activation 5 phase-res-1, f-22
FIRE 5 phase-res : f-1, f-22
<== f-22 (values-for res y n)
==> f-123 (values-for pass 0 ah as)
==> Activation 5 phase-pass : f-1, f-23
FIRE 6 phase-pass : f-1, f-23
<== f-23 (values-for pass 0 ah as)

이러한 검정이 끝나면 의사결정 단계로 들어가 의사결정을 위한 규칙들이 실행된다.

DECISION PHASE

==> f-24 (decision-phase-0)
==> Activation-1 change-to-decision-phase-1 : f-1, f-24
==> Activation 0 XSP-alt-stbd : f-1, f-24, f-10,
==> Activation 0 XSP-alt-stbd : f-1, f-24, f-10,
==> Activation 0 XSP-alt-stbd : f-1, f-24, f-9,
==> Activation 0 XSP-alt-stbd : f-1, f-24, f-10,
FIRE 7 XSP-alt-stbd : f-1, f-24, f-9
==> f-25 (XSP-alt-stbd)
FIRE 8 XSP-alt-stbd : f-1, f-24, f-9,
FIRE 9 XSP-alt-stbd : f-1, f-24, f-10
FIRE 10 XSP-alt-stbd : f-1, f-24, f-10,
FIRE 11 change-to-decision-phase-1 : f-1, f-24
<== f-24 (decision-phase-0)
==> f-26 (decision-phase-1)
==> Activation -1 change-to decision-phase-2 : f-1, f-26
==> Activation 0 XSP-alt-port-round-turn : f-1, f-26, ...
FIRE 12 XSP-alt-port-round-turn : f-1, f-26, f-25, f-16, f-12,,
==> f-27 (XSP-alt-prot-round-turn)
FIRE 13 change-to-decision-phase-2 : f-1, f-26
<== f-26 (decision-phase-1)
==> f-28 (decision-phase-2)
==> Activation -1 go-to-print-phase-main : f-1, f-28
FIRE 14 go-to-print-phase-main : f-1, f-28
<== f-28 (decision-phase-2)
==> f-29 (print-info)
14 rules fired

마침내 XSP-alt-port-round-trun의 충돌예방조선법이 결정되고 이러한 결과를 출력으로 보내주기 위해 프린트 단계로 들어가게 된다. 곧 f-29라는 사실(fact)이 (print-info)이며 이 f-29와 Matching이 되어 실행될 규칙들이 출력을 위한 규칙들이다.

4.3 시스템의 유용성

이상에 살펴본 프로토타이프 충돌예방 전문가시스템의 유용성은 전문가시스템 소프트웨어의 개발 싸이클을 예시하여 주고 있는 그림 7에서 이해할 수 있다.

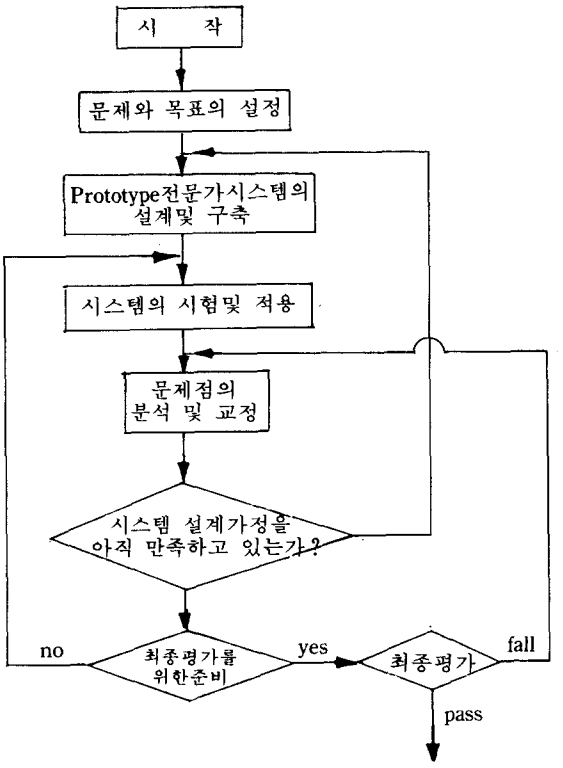

Fig. 8 Exploratory Development Cycle for Rule-based Expert System.

프로토타이프 전문가시스템이 구현되면 지식공학자와 문제영역의 전문가(Domain expert)는 시스템에 문제를 주고 그 결과를 시험, 분석하여 시스템의 지식베이스를 수정, 보완함으로써 완전한 시스템으로 구축될 때까지 시스템의 문제점을 해결하여 나간다. 이러한 접근법은 LOGO 언어를 개발한 Seymour Papert에 의해 제안된 바 있다. 이러한 관점으로 보면 지식베이스는 구축된다('constructed')기 보다는 성장된다('grown')고 할 수 있다. 그러므로 어떤 전문가시스템도 종결된 것으로 생각할 수 없는 것이다. 실제로 미국의 DEC사의 컴퓨터시스템의 구축을 위한 전문가시스템 XCON의 경우, 1981년에 약 500 여개의 규칙(rules)으로 VAX 780시스템을 구축할 수 있었으나, 1984년에는 약 4000 여개의 규칙(rules)을 가진 시스템으로 수정, 보완되었고 이후에도 매년 50% 이상의 규칙들이 수정되고 있다고 한다.[22]

이 연구에서는 우선 전술한 바와 같은 프로토타이프 충돌예방 전문가시스템(PECAS)을 대화형으로 구축하였다. 시스템을 개발하는 도구로 사용한 CLIPS는 C 언어와의 완벽한 통합성을 가지고 있을 뿐만 아니라, Production rule에 기초를 둔 시스템이기 때문에 지식베이스의 수정, 보완이 용이하며 시스템의 확장성이 뛰어난 점 등의 장점을 가지고 있으므로, PECAS는 보다 나은 시스템을 개발하기 위한 시험대(test bed)로 유용하게 사용되리라 생각한다. 그리고 대화형으로 입력되는 ARPA로 부터의 모든 정보는 파일로 처리하여 입력할 수 있으므로, 레이다의 데이타를 PECAS에 입력할 수 있는 형태의 정보로 변환해 주는 알고리즘을 C 언어로 작성하여 주면, RADAR/PECAS 인터페이스의 문제도 해결할 수 있으리라 생각한다.

5. 결 론

컴퓨터의 기술이 발전하면서 다양한 분야의 문제해결에 응용되고 있는 전문가시스템의 기법을 자동화선의 충돌예방 시스템에 적용하여 보았다. 종전까지 주로 알고리즘적인 접근법을 사용하여 자동충돌예방시스템을 개발하고자 하는 시도가 이루어져 왔다. 그러나 여러가지 제약조건 하의 다양한 복수선박의 조우상황을 만나게 되면 충돌예방 문제의 수학적모형이 대단히 복잡해지는 까닭에, 알고리즘적인 접근법 보다는 발견적(Heuristic) 접근법으로 보다 나은 결과를 얻고자 하는 연구가 이루어지고 있다.

이 논문에서는 충돌예방시스템에 전문가시스템 기법을 도입하여, 국제해상충돌예방규칙, Keystone 충돌예방 레이다항법 및 기타 충돌예방 조선법에 관한 전문적인 지식과 경험을 바탕으로 지식베이스를 구축하고, 이를 바탕으로 NASA의

AIS에서 개발한 Production rule에 기초를 둔 전문가시스템 쉘 CLIPS를 사용하여, 우선 전술한 바와 같은 프로토타이프 충돌예방 전문가시스템(PECAS)을 대화형으로 구축하였다.

시스템을 개발하는 도구로 사용한 CLIPS는 C 언어와의 완벽한 통합성을 가지고 있을 뿐만 아니라, Prodcution rule에 기초를 둔 시스템이기 때문에 지식베이스의 수정, 보완이 용이하며 시스템의 확장성이 뛰어난 점 등의 장점을 가지고 있으므로, PECAS는 보다 나은 시스템을 개발하기 위한 시험대(test bed)로 유용하게 사용되리라 생각한다. 그리고 대화형으로 입력되는 ARPA로 부터의 모든 정보는 파일로 처리하여 입력할 수 있으므로, 레이다의 데이타를 PECAS에 입력할 수 있는 형태의 정보로 변환해 주는 알고리즘을 C 언어로 작성하여 이를 시스템과 통합시키면, RADAR/PECAS 인터페이스의 문제도 해결할 수 있으리라 생각하며, 이 문제는 다음의 과제로 남겨 둔다.

REFERENCE

1) Internationl Regulations for Preventing Collision at Sea, 1972.
2) 박용섭 저, 해상교통법론, 한국선원선박문제연구소, p. 237, 1986.
3) Captain C. Hengst et al., Keystone International Anti-Collision Navigation System, Codan Marine Inc., pp. 11～12, 1977.
4) A. G. Bole et al., Automatic Radar Plotting Aids Manual, London, p. 23, 1984.
5) W. P. Cannel, "Collision Avoidance as a Game of Co-ordination," Journal of Navigation, Vol. 34, 1981.
6) F. P. Coenen, G. P. Smeaton and A. G. Bole, "Knowledge-based Collision Avoidnace," Journal of Navigation, Vol. 42, 1989.
7) Martha Grabowski, "The Piloting Expert System : Transition from Off-Line Prototype to On-Line Decision Support," Navigation, Vol. 37, No. 1, Spring, 1990.
8) ibid 4) pp. 22～24.
9) 德田迪夫, 上田一郎 共著, 1971, 超自動化船とコソピュータ, p. 211, 海文堂.
10) Donald H. Sanders, Computers Today, McGraw-Hill Book Co., p. 27, 1988.
11) Donald A. Waterman, A Guide to Expert System, Addison-Wesley Pub. Co. pp. 3～11, 1986.
12) T. C. Chang, Expert Process Planning for Manufacturing, Addision-Wesley Pub. Co. pp. 145～148, 1990.
13) ibid 11) p. 19.
14) ibid 3) p. 15.
15) ibid 3) p. 1.
16) ibid 6) pp. 111～112.
17) Cornelis Dewit, "A Fast and Clear Collision Avoidance Manoeuvre," Proceedings of 6th SCSS, Vol. 1, pp. C1－1～C1－15, Oct. 1981.
18) Paul Harmon and David King, Expert systems : Artificial Intelligence in Business, John Wiley & Sons, Inc., pp. 7～9, 1985.
19) ibid 11) p. 153.
20) ibid 18) pp. 34～48.
21) CLIPS Reference Manual : Version 4. 2, AIS, NASA/JSC, April, 1988.
22) Goerge F. Luger and William A. Stubblefield, Artificial Intelligence and the Dsign of Expert Systems, The Benjamin/Cummings Pub. Co. Inc., pp. 297～301, 1989.

[주 프로그램 PECAS. CLP 리스트]

```
(defrule BEGIN
        (initial-fact)
        (not (BEGIN))
=>
        (system "cls")
        (printout t "" crlf crlf crlf crlf)
        (printout t "*================================================*" crlf)
        (printout t "*  Prototype Expert Collision Avoidance System   *" crlf)
        (printout t "*                                                *" crlf)
        (printout t "*                1991. 6. 29                     *" crlf)
        (printout t "*                                                *" crlf)
        (printout t "*        PECAS: Edited by Kim, Si-Hwa            *" crlf)
        (printout t "* This system generates collision avoidance      *" crlf)
        (printout t "* maneouvring instruction, if any, and then      *" crlf)
        (printout t "* gives output to the navigator with advices.    *" crlf)
        (printout t "*                                                *" crlf)
        (printout t "* To execute, merely load, reset and run.        *" crlf)
        (printout t "*================================================*" crlf)
        (assert (ask-overall-situation))
        (assert (BEGIN))
        (system "pause")
)
;*************************************************
;*      QUERY PHASE 0 ASK OVERALL SITUATION      *
;*************************************************
(defrule ask-overall-situation
        (declare (salience -1))
        ?rem <- (ask-overall-situation)
=>
        (system "cls")
        (retract ?rem)
        (printout t "" crlf crlf)
        (printout t "*    PECAMS CLASSIFY SHIP ENCOUNTER TYPES      *" crlf)
        (printout t "*                AS FOLLOWS                      *" crlf)
        (printout t "*                                                *" crlf)
        (printout t "*  [XSP] Target Crossing from stbd to port       *" crlf)
        (printout t "*  [XPS] Target Crossing from port to stbd       *" crlf)
        (printout t "*  [THO] Target Heading On                       *" crlf)
        (printout t "*  [TOO] Target Overtaking Ownship               *" crlf)
        (printout t "*  [OOT] Ownship Overtaking Target               *" crlf)
        (printout t "*  [TST] Target Stopped                          *" crlf)
        (printout t "*                                                *" crlf)
        (printout t "*                                                *" crlf)
        (printout t "*================================================*" crlf)
        (printout t "" crlf crlf)
        (printout t "ANY TARGET CROSSING FROM STBD TO PORT? [y/n]")
        (assert (XSP-encounter =(read)))
        (printout t "ANY TARGET CROSSING FROM PORT TO STBD? [y/n]")
        (assert (XPS-encounter =(read)))
        (printout t "ANY TARGET HEADING ON? [y/n]")
        (assert (THO-encounter =(read)))
        (printout t "ANY TARGET OVERTAKING OWNSHIP? [y/n]")
        (assert (TOO-encounter =(read)))
        (printout t "IS THE OWNSHIP OVERTAKING ANY TARGET? [y/n]")
        (assert (OOT-encounter =(read)))
        (printout t "ANY TARGET STOPPED? [y/n]")
        (assert (TST-encounter =(read)))
        (assert (ask-type-info))
        (system "cls")
```

```
;*********************************************
;*      QUERY PHASE 1 ASK TYPE SITUATION      *
;*********************************************
(defrule query-for-XSP-1
        ?rem <- (ask-question)
        (XSP-encounter y)
        (not (number-of-XSP ?))
=>
        (retract ?rem)
        (printout t "How many targets are crossing from stbd to port ?")
        (assert (number-of-XSP =(read)))
)
(defrule query-for-XSP-2
        ?rem <- (ask-question)
        (XSP-encounter y)
        (number-of-XSP ?noXSP)
        (not (a XSP ?id ?asp ?dcpa ?tcpa ?res ?pass))
=>
        (bind ?k (+ 1 0))
        (while (>= ?noXSP 1)
               (printout t "How's the apparent aspect of XSP target" ?k crlf
               (printout t "1, 2, or 3 o'clock? [c1/c2/c3] ?")
               (bind ?asp (read))
               (printout t "How's the DCPA of XSP target" ?k crlf)
               (printout t "safe or dangerous ? [s/d]")
               (bind ?dcpa (read))
               (printout t "How's the TCPA of XSP target" ?k crlf)
               (printout t "easy or hurry up ? [e/h]")
               (bind ?tcpa (read))
               (printout t "Is XSP target " ?k " restricted? [y/n]")
               (bind ?res (read))
               (printout t "How's the passing aspect of XSP target" ?k crlf)
               (printout t "passing ahead or astern ? [ah/as]")
               (bind ?pass (read))
               (assert (a XSP ?k ?asp ?dcpa ?tcpa ?res ?pass))
               (bind ?noXSP (- ?noXSP 1))
               (bind ?k (+ ?k 1))
        )
)
(defrule query-for-XPS-1
        ?rem <- (ask-question)
        (XPS-encounter y)
        (not (number-of-XPS ?))
=>
        (retract ?rem)
        (printout t "How many targets are crossing from port to stbd ?")
        (assert (number-of-XPS =(read)))
)
(defrule query-for-XPS-2
        ?rem <- (ask-question)
        (XPS-encounter y)
        (number-of-XPS ?noXPS)
        (not (aspect-XPS ?asp))
        (not (a XPS ?id ?asp ?dcpa ?tcpa ?res ?pass))
=>
        (bind ?k (+ 1 0))
        (while (>= ?noXPS 1)
               (printout t "How's the apparent aspect of XPS target" ?k crlf)
               (printout t "9, 10, or 11 o'clock ?[c9/c10/c11]")
               (bind ?asp (read))
               (printout t "How's the DCPA of XPS target" ?k crlf)
               (printout t "safe or dangerous ? [s/d]")
               (bind ?dcpa (read))
```

```
                (printout t "How's the TCPA of XPS target" ?k crlf)
                (printout t "easy or hurry up ? [e/h]")
                (bind ?tcpa (read))
                (printout t "Is XPS target " ?k " restricted? [y/n]")
                (bind ?res (read))
                (printout t "How's the passing aspect of XPS target" ?k crlf)
                (printout t "passing ahead or astern ? [ah/as]")
                (bind ?pass (read))
                (assert (a XPS ?k ?asp ?dcpa ?tcpa ?res ?pass))
                (bind ?noXPS (- ?noXPS 1))
                (bind ?k (+ ?k 1))
        )
)
(defrule query-for-THO-1
        ?rem <- (ask-question)
        (THO-encounter y)
        (not (number-of-THO ?))
=>
        (retract ?rem)
        (printout t "How many targets are heading on ?")
        (assert (number-of-THO =(read)))
)
(defrule query-for-THO-2
        ?rem <- (ask-question)
        (THO-encounter y)
        (number-of-THO ?noTHO)
        (not (a THO ?id ?asp ?dcpa ?tcpa ?res ?pass))
=>
        (bind ?k (+ 1 0))
        (while (>= ?noTHO 1)
                (printout t "How's the apparent aspect of THO target" ?k crlf)
                (printout t "on-port or on-stbd near 0 o'clock or c0?[p0/c0/s0]
")
                (bind ?asp (read))
                (printout t "How's the DCPA of THO target" ?k crlf)
                (printout t "safe or dangerous ? [s/d]")
                (bind ?dcpa (read))
                (printout t "How's the TCPA of THO target" ?k crlf)
                (printout t "easy or hurry up ? [e/h]")
                (bind ?tcpa (read))
                (printout t "Is THO target " ?k " restricted? [y/n]")
                (bind ?res (read))
                (assert (a THO ?k ?asp ?dcpa ?tcpa ?res 0))
                (bind ?noTHO (- ?noTHO 1))
                (bind ?k (+ ?k 1))
        )
)
(defrule query-for-TOO-1
        ?rem <- (ask-question)
        (TOO-encounter y)
        (not (number-of-TOO ?))
=>
        (retract ?rem)
        (printout t "How many targets are going to overtake ownship ?")
        (assert (number-of-TOO =(read)))
)
(defrule query-for-TOO-2
        ?rem <- (ask-question)
        (TOO-encounter y)
        (number-of-TOO ?noTOO)
        (not (a TOO ?id ?asp ?dcpa ?tcpa ?res ?pass))
```

```
=>
        (bind ?k (+ 1 0))
        (while (>= ?noTOO 1)
          (printout t "How's the apparent aspect of TOO target" ?k crlf)
          (printout t "on which direction of the clock?[c5/c6/c7]")
          (bind ?asp (read))
          (printout t "How's the DCPA of TOO target" ?k crlf)
          (printout t "safe or dangerous ? [s/d]")
          (bind ?dcpa (read))
          (printout t "How's the TCPA of TOO target" ?k crlf)
          (printout t "easy or hurry up ? [easy/hurry]")
          (bind ?tcpa (read))
          (printout t "Is TOO target " ?k " restricted? [y/n]")
          (bind ?res (read))
          (printout t "How's the passing aspect of TOO target" ?k crlf)
          (printout t "passing ahead/astern or none of the two [ah/as/0]")
          (bind ?pass (read))
          (assert (a TOO ?k ?asp ?dcpa ?tcpa ?res ?pass))
          (bind ?noTOO (- ?noTOO 1))
          (bind ?k (+ ?k 1))
        )
)
(defrule query-for-OOT-1
        ?rem <- (ask-question)
        (OOT-encounter y)
        (not (number-of-OOT ?))
=>
        (retract ?rem)
        (printout t "How many targets are to be overtaken by ownship ?")
        (assert (number-of-OOT =(read)))
)
(defrule query-for-OOT-2
        ?rem <- (ask-question)
        (OOT-encounter y)
        (number-of-OOT ?noOOT)
        (not (a OOT ?id ?asp ?dcpa ?tcpa ?res ?pass))
=>
        (bind ?k (+ 1 0))
        (while (>= ?noOOT 1)
           (printout t "How's the apparent aspect of OOT target" ?k crlf)
           (printout t "on which direction of the o'clock?[c1/c0/c11]")
           (bind ?asp (read))
           (printout t "How's the DCPA of OOT target" ?k crlf)
           (printout t "safe or dangerous ? [safe/danger]")
           (bind ?dcpa (read))
           (printout t "How's the TCPA of OOT target" ?k crlf)
           (printout t "easy or hurry up ? [easy/hurry]")
           (bind ?tcpa (read))
           (printout t "Is OOT target " ?k " restricred? [y/n]")
           (bind ?res (read))
           (assert (a OOT ?k ?asp ?dcpa ?tcpa ?res 0))
           (bind ?noOOT (- ?noOOT 1))
           (bind ?k (+ ?k 1))
        )
)
(defrule query-for-TST-1
        ?rem <- (ask-question)
        (TST-encounter y)
        (not (number-of-TST ?))
=>
        (retract ?rem)
        (printout t "How many targets stopped are there?")
        (assert (number-of-TST =(read)))
)
```

```
(defrule query-for-TST-2
        ?rem <- (ask-question)
        (TST-encounter y)
        (number-of-TST ?noTST)
        (not (a TST ?id ?asp ?dcpa ?tcpa ?res ?pass))
=>
        (bind ?k (+ 1 0))
        (while (>= ?noTST 1)
               (printout t "How's the apparent aspect of TST target" ?k crlf)
               (printout t "on-stbd, on-port or on-ahead? [s/p/a]")
               (bind ?asp (read))
               (printout t "How's the DCPA of TST target" ?k crlf)
               (printout t "safe or dangerous ? [safe/danger]")
               (bind ?dcpa (read))
               (printout t "How's the TCPA of TST target" ?k crlf)
               (printout t "easy or hurry up ? [easy/hurry]")
               (bind ?tcpa (read))
               (printout t "Is TST target " ?k " restricted?[y/n]")
               (bind ?res (read))
               (assert (a TST ?k ?asp ?dcpa ?tcpa ?res 0))
               (bind ?noTST (- ?noTST 1))
               (bind ?k (+ ?k 1))
        )
)
(defrule ask-type-info ""
        (ask-type-info)
        (not (ask-question))
=>
        (assert (ask-question))
)
;*****************************************
;*      PHASE 1 SELECT MODULE            *
;*****************************************
(defrule select-XSP-module
        (declare (salience -1))
        (select-module)
        (a XSP $?)
=>
        (assert (module 1))
)
(defrule select-THO-module
        (declare (salience -1))
        (select-module)
        (not (a XSP $?))
        (a THO $?)
=>
        (assert (module 3))
)
(defrule select-OOT-module
        (declare (salience -1))
        (select-module)
        (not (a XSP $?))
        (not (a THO $?))
        (a OOT $?)
=>
        (assert (module 5))
)
```

```
(defrule select-XPS-module
        (declare (salience -1))
        (select-module)
        (not (a XSP $?))
        (not (a THO $?))
        (not (a OOT $?))
        (a XPS $?)
=>
        (assert (module 2))
)
(defrule select-TOO-module
        (declare (salience -1))
        (select-module)
        (not (a XSP $?))
        (not (a THO $?))
        (not (a OOT $?))
        (not (a XPS $?))
        (a TOO $?)
=>
        (assert (module 4))
)
(defrule select-TST-module
        (declare (salience -1))
        (select-module)
        (not (a XSP $?))
        (not (a THO $?))
        (not (a OOT $?))
        (not (a XPS $?))
        (not (a TOO $?))
        (a TST $?)
=>
        (assert (module 6))
)
(defrule select-NONE
        (declare (salience -1))
        (select-module)
        (not (a XSP $?))
        (not (a THO $?))
        (not (a OOT $?))
        (not (a XPS $?))
        (not (a TOO $?))
        (not (a TST $?))
=>
        (printout t "Nothing but Own ship exists at the moment!!" crlf)
        (reset)
)
;*****************************************
;*      PHASE 2 LOAD MODULE              *
;*****************************************
(defrule load-XSP-Module
        (declare (salience -2))
        (load-module)
        ?rem <- (module 1)
=>
        (retract ?rem)
        (printout t crlf "Crossing-stbd-port Module Loaded." crlf)
        (load "XSP.clp")
        (assert (Go-to-Module XSP))
)
(defrule load-XPS-Module
        (declare (salience -2))
        (load-module)
        ?rem <- (module 2)
```

```
=>
        (retract ?rem)
        (printout t crlf "Crossing-port-stbd Module Loaded." crlf)
        (load "XPS.clp")
        (assert (Go-to-Module XPS))
)
(defrule load-THO-Module
        (declare (salience -2))
        (load-module)
        ?rem <- (module 3)
=>
        (retract ?rem)
        (printout t crlf "Head-on-target Module Loaded." crlf)
        (load "THO.clp")
        (assert (Go-to-Module THO))
)
(defrule load-TOO-Module
        (declare (salience -2))
        (load-module)
        ?rem <- (module 4)
=>
        (retract ?rem)
        (printout t crlf "Target-Overtaking Module Loaded." crlf)
        (load "TOO.clp")
        (assert (Go-to-Module TOO))
)
(defrule load-OOT-Module
        (declare (salience -2))
        (load-module)
        ?rem <- (module 5)
=>
        (retract ?rem)
        (printout t crlf "Ownship-Overtaking Module Loaded." crlf)
        (load "OOT.clp")
        (assert (Go-to-Module OOT))
)
(defrule load-TST-Module
        (declare (salience -2))
        (load-module)
        ?rem <- (module 6)
=>
        (retract ?rem)
        (printout t crlf "Target-Stopped Module Loaded." crlf)
        (load "TST.clp")
        (assert (Go-to-Module TST))
)
;*****************************************
;*      PHASE 3 DO MODULE                *
;*****************************************
;       ..................
;*****************************************
;*      PHASE 4 PRINT-INFO PHASE         *
;*****************************************
(defrule print-info-1
        (declare (salience -4))
        (print-info)
        (XSP-encounter y)
        (number-of-XSP ?n)
=>
        (assert (noXSP ?n))
)
;       ........................
;
```

```
;*****************************************
;*      PHASE CONTROL RULES              *
;*****************************************
(defrule change-to-phase-1
        (declare (salience -10))
        ?rem <- (ask-type-info)
        ?rem1 <- (ask-question)
=>
        (retract ?rem ?rem1)
        (assert (select-module))
)
(defrule change-to-phase-2
        (declare (salience -10))
        ?rem <- (select-module)
=>
        (retract ?rem)
        (assert (load-module))
)
(defrule change-to-phase-3
        (declare (salience -10))
        ?rem <- (load-module)
        ?rem1 <- (Go-to-Module ?mod)
=>
        (retract ?rem ?rem1)
        (assert (Do-Module ?mod))
)
(defrule change-to-phase-4
        (declare (salience -10))
        ?rem <- (Do-Module ?mod)
        ?rem1 <- (print-info)
=>
        (retract ?rem ?rem1)
        (printout t "PECAS is peacefully terminated !!!")
)
```

[부 프로그램 XSP. CLP 리스트]

```
;;;*************************************************
;;;*                                               *
;;;*   TARGET CROSSING FROM STARBOARD TO PORT      *
;;;*                                               *
;;;*************************************************
;;;*         EDITED BY KIM, SI-HWA AND             *
;;;*************************************************
(defrule check-info
        (Do-Module XSP)
=>
        (assert (values-for id 1 2 3 4))
)
(defrule bad-info-id
        (declare (salience 10))
        (Do-Module XSP)
        (values-for id $?list)
        ?fid <- (a ?type ?value ?a ?d ?t ?r ?p)
        (test (! (member ?value $?list)))
=>
        (retract ?fid)
        (printout t "Target ID information bad!!" crlf)
        (printout t "(" ?type " " ?value " " ?a " " ?d " "
                        ?t " " ?r " " ?p ")")
        (system "pause")
)
(defrule bad-info-asp
        (declare (salience 10))
        (Do-Module XSP)
        (values-for asp $?list)
        ?fasp <- (a ?type ?i ?value ?d ?t ?r ?p)
        (test (! (member ?value $?list)))
=>
        (retract ?fasp)
        (printout t "Target Aspect information bad!!" crlf)
        (printout t "(" ?type " " ?i " " ?value " " ?d " "
                        ?t " " ?r " " ?p ")")
        (system "pause")
)
(defrule bad-info-dcpa
        (declare (salience 10))
        (Do-Module XSP)
        (values-for dcpa $?list)
        ?fdcpa <- (a ?type ?i ?a ?value ?t ?r ?p)
        (test (! (member ?value $?list)))
=>
        (retract ?fdcpa)
        (printout t "Target DCPA information bad!!" crlf)
        (printout t "(" ?type " " ?i " " ?a " " ?value " "
                        ?t " " ?r " " ?p ")")
        (system "pause")
)
```

```
(defrule bad-info-tcpa
        (declare (salience 10))
        (Do-Module XSP)
        (values-for tcpa $?list)
        ?ftcpa <- (a ?type ?i ?a ?d ?value ?r ?p)
        (test (! (member ?value $?list)))
=>
        (retract ?ftcpa)
        (printout t "Target TCPA information bad!!" crlf)
        (printout t "(" ?type " " ?i " " ?a " " ?d " " ?value
                        " " ?r " " ?p ")")
        (system "pause")
)
(defrule bad-info-res
        (declare (salience 10))
        (Do-Module XSP)
        (values-for res $?list)
        ?fres <- (a ?type ?i ?a ?d ?t ?value ?p)
        (test (! (member ?value $?list)))
=>
        (retract ?fres)
        (printout t "Target Restriction information bad!!" crlf)
        (printout t "(" ?type " " ?i " " ?a " " ?d " " ?t
                        " " ?value " " ?p ")")
        (system "pause")
)
(defrule bad-info-pass
        (declare (salience 10))
        (Do-Module XSP)
        (values-for pass $?list)
        ?fpass <- (a ?type ?i ?a ?d ?t ?r ?value)
        (test (! (member ?value $?list)))
=>
        (retract ?fpass)
        (printout t "Target Passing information bad!!" crlf)
        (printout t "(" ?type " " ?i " " ?a " " ?d " " ?t
                        " " ?r " " ?value ")")
        (system "pause")
)
;;*********************************************
;;*   PHASE CONTROL FOR CHECKING BAD-INFO      *
;;*********************************************
(defrule phase-id
        (declare (salience 5))
        (Do-Module XSP)
        ?rem <- (values-for id $?list)
=>
        (retract ?rem)
        (assert (values-for asp c0 c1 c2 c3 c4 c5 c6
                        c7 c8 c9 c10 c11 p0 s0))
)
(defrule phase-asp
        (declare (salience 5))
        (Do-Module XSP)
        ?rem <- (values-for asp $?list)
=>
        (retract ?rem)
        (assert (values-for dcpa s d))
```

```
(defrule phase-dcpa
        (declare (salience 5))
        (Do-Module XSP)
        ?rem <- (values-for dcpa $?list)
=>
        (retract ?rem)
        (assert (values-for tcpa e h))
)
(defrule phase-tcpa
        (declare (salience 5))
        (Do-Module XSP)
        ?rem <- (values-for tcpa $?list)
=>
        (retract ?rem)
        (assert (values-for res y n))
)
(defrule phase-res
        (declare (salience 5))
        (Do-Module XSP)
        ?rem <- (values-for res $?list)
=>
        (retract ?rem)
        (assert (values-for pass 0 ah as))
)
(defrule phase-pass
        (declare (salience 5))
        (Do-Module XSP)
        ?rem <- (values-for pass $?list)
=>
        (retract ?rem)
        (assert (decision-phase-0))
)
;;*************************************************
;;*     XSP DECISION PHASE-0                      *
;;*************************************************
(defrule XSP-alt-stbd
        (Do-Module XSP)
        (decision-phase-0)
        (or (a XSP ?i ?a d ?t ?r ah)
            (and (a XSP ?i ?a d ?t ?r ah)
                 (not (a XSP ?i ?a d ?t ?r as))
            )
        )
=>
        (assert (XSP-alt-stbd))
)
;
;       ......................
;
;
(defrule XSP-alt-port
        (Do-Module XSP)
        (decision-phase-0)
        (a XSP ?i ?a d ?t ?r as)
        (not (a XSP $? ah))
        (not (a THO $?))
        (not (a XPS $?))
=>
        (assert (XSP-alt-port))
)
;       ......................
;
```

```
;;*********************************************
;;*      XSP DECISION PHASE 1                 *
;;*********************************************
(defrule XSP-alt-stbd-90
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (a TOO ?i ?a s e n $?)
=>
        (assert (XSP-alt-stbd-90))
)
;       ........................
;
(defrule XSP-facc-stbd
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (not (a TOO ?i c5 d h $?))
        (not (a TOO ?i c5 $? h $?))
=>
        (assert (XSP-facc-stbd))
)
;       ........................
;
(defrule XSP-alt-port-stern-on
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (a TOO ?i c5 $? h $?)
        (not (a XPS $?))
        (not (a TOO $?))
=>
        (assert (XSP-alt-port-stern-on))
)
;       ........................
;
(defrule XSP-alt-port-round-turn
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (a TOO ?i c5 $? h $?)
        (a XPS $?)
        (not (a TOO ?i c7 d h $?))
        (not (a TOO ?i c7 $? h $?))
=>
        (assert (XSP-alt-port-round-turn))
)
;
;       ........................
;
(defrule XSP-reduce-speed
        (Do-Module XSP)
        (decision-phase-1)
        (XSP-alt-stbd)
        (a TOO ?i c5 $? h $?)
        (a XPS $?)
        (and (or (a TOO ?i c7 d h $?)
                 (a TOO ?i c7 $? h $?)
             )
        )
=>
        (assert (XSP-reduce-speed))
;       ........................
```

```
;;*********************************************
;;*      XSP DECISION PHASE 2                 *
;;*********************************************
;
;
(defrule XSP-emergency
        (Do-Module XSP)
        (decision-phase-2)
        (XSP-alt-stbd)
        (a TOO ?i c5 $? h $?)
        (a XPS ?i ?a d h $?)
        (and (or (a TOO ?i c7 d h $?)
                 (a TOO ?i c7 $? h $?)
             )
        )
        (a TOO ?i c6 d $?)
=>
        (assert (XSP-emergency))
)
;       ..............................
;
;;*********************************************
;;*      XSP DECISION PHASE CONTROL RULES     *
;;*********************************************
(defrule change-to-decision-phase-1
        (declare (salience -1))
        (Do-Module XSP)
        ?rem <- (decision-phase-0)
=>
        (retract ?rem)
        (assert (decision-phase-1))
)
(defrule change-to-decision-phase-2
        (declare (salience -1))
        (Do-Module XSP)
        ?rem <- (decision-phase-1)
=>
        (retract ?rem)
        (assert (decision-phase-2))
)
(defrule go-to-print-phase-main
        (declare (salience -1))
        (Do-Module XSP)
        ?rem <- (decision-phase-2)
=>
        (retract ?rem)
        (assert (print-info))
)
```