論 文

大 韓 造 船 學 會 論 文 集
第 33 卷 第 1 號 1996年 2月
Transactions of the Society of
Naval Architects of Korea
Vol. 33, No. 1, February 1996

외부 시스템과의 접속을 통한 통합 구조설계 전문가 시스템 개발

이경호*, 이동곤*

A Structural Design Expert System with Integrated Design Environments

by

K. H. Lee* and D. K. Lee*

요 약

본 논문에서는 범용 전문가 시스템 쉘을 이용하여 구조설계, 특히 선박의 중앙 횡단면 구조설계를 지원할 수 있는 전문가 시스템 (ESMID)을 개발하였다. 기존의 전문가 시스템과는 달리 여기서는 효율적인 설계 전문가시스템을 구현하기 위하여 지식베이스와 공학 데이타베이스, 사용자 인터페이스, 그리고 공학 계산 프로그램들이 통합된 시스템을 구현하였다. 본 시스템의 유효성을 검증하기 위하여 64K 살물선에 적용하여 구조설계를 수행하였다. 기존의 전문가시스템에 데이타베이스, 그래픽 사용자 인터페이스 등의 환경들이 접목됨으로써 초보자도 쉽게 구조설계를 수행할 수 있게 된다.

Abstract

In this article, an Expert System for MIDship section design (ESMID) is developed based on a general-purpose expert system shell. The system focuses on the integration of knowledge-based system and design resources such as an engineering database, graphical user interface, and engineering application programs for the calculation of section modulus and longitudinal strength of ship structure. The validation of the ESMID system was examined and verified by applying the system to the 64K bulk carrier. The developed expert system can help a novice engineer in designing the midship section of ship.

접수일자 : 1994년 8월 11일, 재접수일자 : 1995년 12월 1일
*정회원, 한국기계연구원 선박해양공학연구센터

1. 서 론

설계 분야에서 효율적인 전문가 시스템을 개발하기 위해서는 많은 양의 설계정보들을 어떻게 효율적으로 이용하느냐 하는 것과 기존에 사용되고 있는 각종 설계 프로그램들과 어떻게 유연하게 접속할 것인가 하는 두가지 문제가 해결되어야 한다 [1].

본 연구에서는 범용 전문가 시스템 쉘인 Nexpert Object를 이용하여 범용 쉘이 어떻게 선박설계, 특히 중앙단면 구조설계에 이용될 수 있는가를 검토하였다.

즉, 지식베이스와 데이타베이스의 통합을 통한 설계정보의 이용과 시스템 통합 관점에서 선박 구조설계를 지원하는 전문가 시스템을 개발하였다.

특히, 객체지향 개념에 의한 지식베이스의 구현과 지식베이스와 데이타베이스의 통합 기법, API (Application Programming Interface)를 통한 시스템 통합 기법을 연구하였다.

먼저, 지식 표현에 있어서 객체지향 개념에 의해, 사용되는 부재를 클래스로 정의하고, 이들의 그룹핑과 계층화를 시도하였다. 또한 이를 통한 규칙의 구성, 동적객체, 메타슬롯등의 구현을 통해 지식베이스 구축의 효율성을 검토하였다. 지식베이스 전문가 시스템의 개발에 있어서 지식베이스만을 통한 시스템 구현은 그 한계를 가지고 있다. 따라서 데이타베이스의 접속을 통한 시스템의 확장이 설계 전문가 시스템 구현을 위해서는 아주 중요한 요소로 자리잡고 있다. 구조설계 분야에서도 많은 양의 데이타가 사용되며, 추론시에 이런 데이타가 이용될 뿐 아니라 추론된 데이타가 신속히 저장되어 설계의 다음 단계에서 이용되어야 한다. 따라서 지식베이스와 데이타베이스의 외부형태의 단순한 접속이 아니라 지식베이스 내에서 직접 데이타베이스를 접근하는 기법을 연구하였다. 기존의 이 분야에 대한 연구[13]에 비해 본 연구에서는 지식베이스와 데이타베이스의 접속 및 외부환경과의 접속을 통하여 좀더 실제적이고 효율적인 설계 전문가 시스템을 구축할 수 있었다. 또한 전문가 시스템의 사용자는 일반적으로 쉘의 기능 및 사용법을 잘 알지 못하기 때문에 시스템 개발자는 사용자 인터페이스를 통해 사용자가 쉽게 시스템을 사용할 수 있도록 하여야 한다. 이를 위해서는 외부 프로그램에서 전문가 시스템 쉘의 기능을 제어할 수 있어야 한다. 따라서 전문가 시스템 쉘이 제공하는 외부 프로그램과의 접속기능 (Callable Interface)을 이용하여 GUI(Graphical User Interface)프로그램을 바탕으로 한 구조설계용 API 프로그램을 구현하였다.

특히, Motif를 이용한 그래픽 사용자 인터페이스 (GUI)를 도입하여 사용자 지향의 시스템을 추구하였다.

2. 설계 지식의 분석

설계란, 요구된 기능을 제품이라는 실체로 구체화하는 것이라고 정의된다 [2]. 또한 설계는 기능정의 (Function)를 구성원들 간의 상관관계를 규정짓는 구조 (Structure)로 변환하는 것으로서, 그들 사이를 매개하는 것은 설계 대상물의 거동 (Behavior)이라고 할 수 있다 [3]. 이러한 설계의 개념을 바탕으로 선박 중앙단면 구조설계의 작업과정을 정보의 흐름 (Information Flow)과 작업의 흐름 (Work Flow)으로 구분하였다. 여기서 설계정보는 주로 구조부재에 대한 치수 등을 나타내는 속성정보를 의미하며, 이 부재들의 속성정보는 설계작업 과정중에서 계속적으로 생성되고 변경되지만 부재대상 자체는 정적인 지식, 즉 클래스나 객체로 간주할 수 있다. 따라서 선박 중앙단면 구조설계에 이용되는 설계 데이타를 분석한 후 구조부재를 객체들로 정의하고 이들의 속성정보를 데이타베이스화 하여 정보의 흐름을 표현하였다. 또한 설계되어야 할 구조부재들에 대해 객체와 이들의 상관관계로 구성되는 사실 (Fact)을 정의하였다 [4].

또한 일련의 설계과정, 즉 설계 대상물의 거동을 나타내는 작업의흐름을 규칙 (Rule)을 이용하여 표현하였고, 지식의 정적인 면과 동적인 면을

고려하여 3장과 4장 에서와 같이 설계지식을 표현하였다.

3. 객체지향 개념에 의한 구조부재의 정적 지식표현

3.1 클래스 (Class) 의 정의

일반적으로 특정 분야의 지식은 객체 (Object), 클래스 (Class), 속성(Property)들에 의해서 모델링 될 수 있다 [5].

즉, 객체를 이용하여 실세계를 표현할 수 있고, 이렇게 정의된 객체들은 공통된 특성을 가질 수 있다. 이를 클래스라 하며 클래스는 객체들의 모임으로 일반화 (Generalization)된 개념이라 할 수 있다.

Fig.1 은 선체 구조설계에 사용되는 부재들을 클래스로 정의한 일부이다. 그림에서 볼 수 있듯이 Plate, Stiff 등과 같이 일반화된 개념을 추상화하여 클래스로 정의하였다.

3.2 객체 (Object) 의 정의

객체는 정보의 가장 작은 단위이다. 즉, 지식표현에 있어서 어떠한 대상을 묘사하기 위하여 사용하는데, 클래스의 "Instances" 라고 볼 수 있다.

Fig.2 는 선체 중앙부 구조부재의 객체를 정의한 일부이다. 그림에서 keel_plate, inner_bottom_plate 등과 같은 객체는 고유의 속성을 가지고 있으면서, 이것이 Plate 클래스의 인스턴스이므로 Plate 클래스의 속성들을 상속받게 된다.
일반적으로 객체는 그들의 값을 갖기위해 속성(Property)을 가지고 있다.

3.3 메타슬롯 (Meta-Slot) 의 정의

메타슬롯은 객체의 거동 (Behavior)을 나타내는데 사용되며, 객체의 속성값 및 객체의 속성값을 얻기 위한 수식을 정의해 두는 포인터(Pointer)를 갖고 있다.

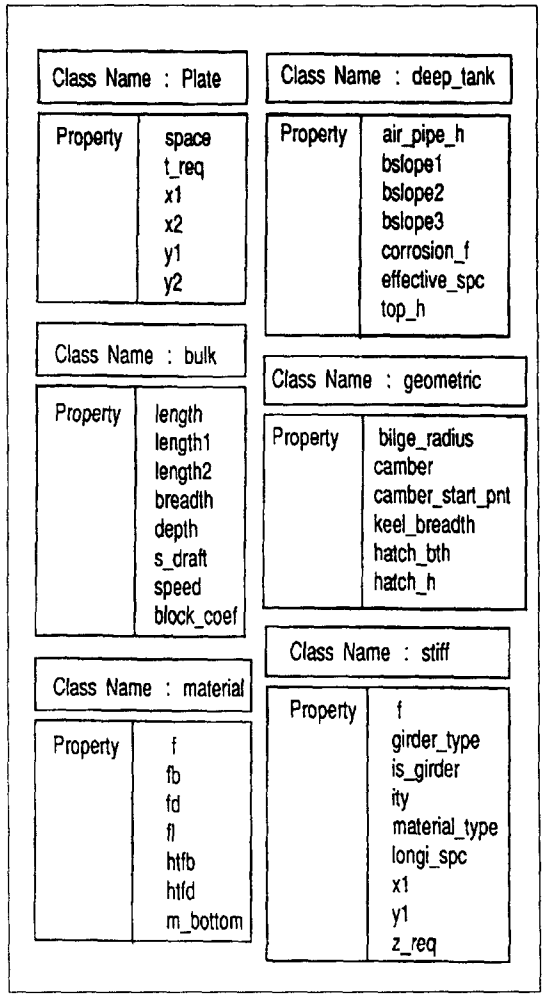

Fig. 1 Definition of classes

속성값을 얻기위한 복잡한 계산식을 메타슬롯에 정의함으로써 규칙의 복잡성을 줄일 수 있다. 즉 메타슬롯은 복잡한 계산식을 표현하기 위해 여러 규칙들을 정의하여야만 표현할 수 있었던 과거의 전문가 시스템들과는 달리, 복잡한 계산식을 메타슬롯에 정의해 놓으므로써 보다 현실적인 지식의 표현은 물론, 이러한 지식의 표현 및 유지를 쉽게 하며 지식베이스의 복잡성을 줄여준다. Fig.3 는 메타슬롯의 구현예를 보인것이다.

예를 들어, 규칙을 추론할때 bottom_plate.ttl의 값이 사용되면 이 슬롯의 값을 얻기위해 다른 규

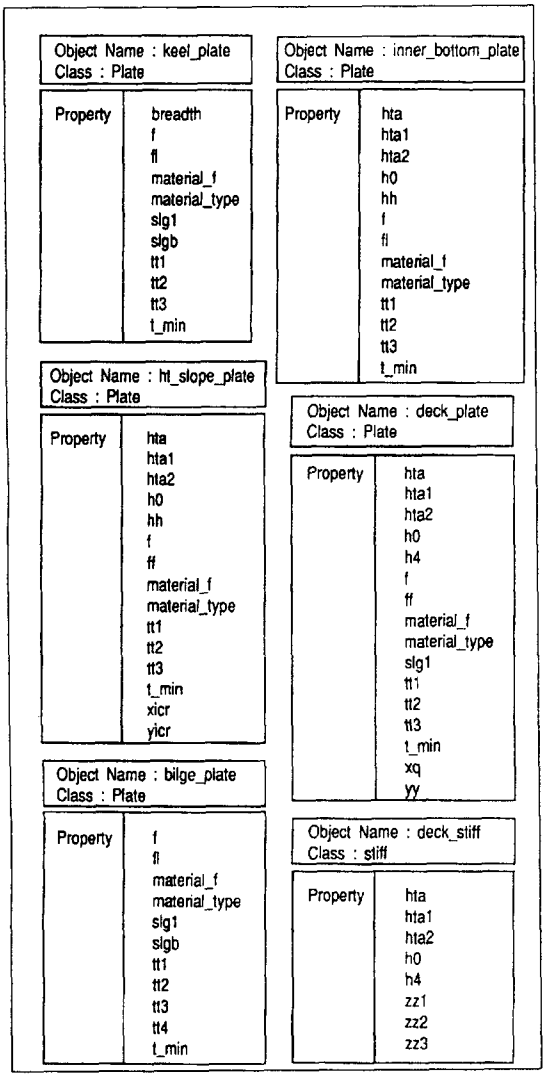

Fig. 2 Definition of objects

칙을 정의하지 않고 메타슬롯에 계산식을 기술해 놓으므로써 추론시 쉽게 값을 얻어 올 수 있으며, 이 슬롯에 대한 지식이 변경될때도 메타슬롯의 내용만 수정하면 되므로 유지보수 측면에서도 유용한 방법이라 할 수 있다.

이 메타슬롯 개념은 객체지향 개념의 Method와 유사한 개념이다.

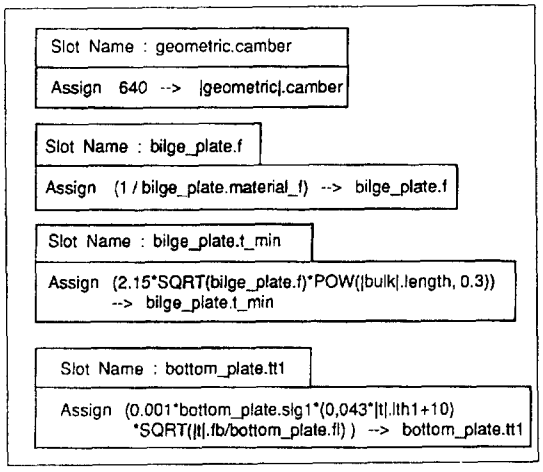

Fig. 3 Definition of meta-slots

4. 구조설계용 데이타베이스 구현

4.1 ORACLE DBMS를 이용한 테이블 스키마 구현

Table 1 은 실적선 데이타 테이블을 구성하기 위한 데이타 목록이다.

Table 1 Schema table for existing database

테이블 스키마		
과제명 : ED5380		테이블명 : SHIPS
데이타베이스 분류 : 실적선 DB		작성일 : 199 . 8
작성자 : 이 경 호		
ATTRIBUTE	DATA TYPE	DESCRIPTION
DB_SHIPID	char (10)	실적선 Identifier
DB_TYPE	char(8)	Type (BULK,...)
DB_DWT	number(10)	Deadweight
DB_LBP	number(10,2)	Ship Length
DB_DL_DAY	date	인도 날짜

본래 ER 모형도를 구성한 다음 이를 바탕으로 테이블간의 상관관계를 분석하여 테이블을 구성[6]해야 하지만, 여기서는 테이블간의 상관관계가 거의 없는 간단한 형태이므로 이전단계를 생략하고 테이블에 수록될 데이타 목록만을 표현하였다.

Table 1 에서 구성한 데이타 목록을 바탕으로 테이블을 상용 관계형 데이타베이스 시스템인 ORACLE DBMS 에서 구성하기 위해서는 다음과 같은 방법으로 작업을 수행한다.

(1) ORACLE DBMS에서 작업을 위한 Account 를 생성

```
# su - oracle
% sqlplus system/manager
SQL> grant connect, resource
      to nexpert identified by nexpert ;
SQL> exit
```

(2) 데이타 목록에 따른 SHIPS 테이블 구현

```
% sqlplus nexpert / nexpert
SQL> create table ships (
       DB_SHIPID          char ( 10 ),
       DB_TYPE            char ( 8 ),
       DB_DWT             number( 10),
       DB_LBP             number(10,2),
       DB_DL_DAY          date ) ;
```

(3) 데이타 입력

```
SQL> insert into ships values
    ( '65KBC', 'BULK', '65700', '220.23',
'22-FEB-90' ) ;
```

실재로 선급 규정에 의한 선박 중앙단면 설계를 위해서는 선급 규정에 의해서 계산된 보강재 (Stiffeners)의 단면 계수에 따른 보강재의 형상이 데이타베이스에 저장되어 있어서, 단면 계수에 따라 적합한 보강재를 선택하게 된다. 단면 계수에 따른 보강재 규격의 데이타베이스 구축을 위한 데이타 목록은 Table 2와 같다. 이 테이블은 I 형강재의 속성을 표현하기 위한 것이다. 다른 종류의 보강재도 이와 유사하게 표현된다.

4.2 SQL을 통한 ORACLE RDBMS 접근

SQL은 "Structured Query Language" 를 의미하며, 데이타베이스를 검색하기 위한 언어의 일종이다. 여기서는 SQL*Plus 를 사용하였으며, 이것은 다음과 같은 기능을 수행한다 [7].

- 데이타베이스내에서 테이블 생성
- 테이블내에 정보를 저장
- 테이블내의 데이타 변경
- 원하는 데이타의 검색
- 데이타베이스 자체의 유지보수

Table 2 Schema table for stiffener database

테이블 스키마		
과제명 : ED5380 데이타베이스 분류 : 보강재 DB 작성자 : 이 경 호		테이블명 : stiffener 작성일 : 199 . 8
ATTRIBUTE	DATA TYPE	DESCRIPTION
DB_A	number (10)	Web 길이 (mm)
DB_B	number (10)	Flange 길이 (mm)
DB_T1	number (10,1)	Web 두께 (mm)
DB_T2	number (10,1)	Flange 두께 (mm)
DB_AREA	number (10,2)	단면적 (Cm**2)
DB_Z	number (10,1)	단면계수 (Cm**3)

5. 데이타베이스와 지식베이스의 인터페이스

5.1 DB Bridge를 통한 인터페이스

본 연구에서 사용된 전문가시스템 쉘의 지식베이스에서 데이타베이스를 접근하기 위해서는 Retrieve 나 Write 연산자를 사용하게 된다.

Retrive 연산자는 데이타베이스로 부터 필요한 데이타를 추출한 후 이것을 전문가시스템 쉘의 작업 기억공간 (Working Memory)로 넘겨준다. 반대로 Write 연산자는 전문가시스템 쉘의 작업 기억공간에 있는 데이타를 데이타베이스로 저장시키게 된다 [8].

5.2 ORACLE DBMS와 지식베이스의 인터페이스

전문가시스템 쉘에서 데이타베이스로 접근하기

위해서는 원하는 데이타베이스를 연결시켜 주는 DB Bridge가 필요하다.

DB Bridge는 일반적으로 아래와 같은 3가지 개념의 방법을 통해 지식베이스와 데이타베이스를 연결시켜 준다.

- Atomic operation
- Sequential operation
- Grouped operation

5.2.1 Atomic operation

Atomic operation 은 단지 한개의 레코드를 retrieve 하거나 write하는 것을 말하며, 데이타베이스와 전문가시스템 쉘 작업 기억공간 사이를 하나의 레코드만을 전송시킨다. 특히 Atomic retrieve는 retrive 윈도우의 Query의 제한 조건에 따라 데이타베이스로 부터 전문가시스템 쉘의 작업 기억공간으로 데이타를 넘겨주게 된다. 이때 Query가 데이타베이스로 부터 한개의 레코드를 retrieve 하기에 충분하지 않더라도 첫번째 레코드를 가져오게 된다.

5.2.2 Sequential operation

Sequential operation은 한번에 하나씩 여러개의 레코드를 retrieve 하거나 write 하는 것을 말하며, 데이타베이스와 전문가시스템 쉘 작업 기억공간 사이를 한번에 한 레코드씩 여러개의 레코드를 전송시킨다. 특히 Sequential retrieve 는 데이타베이스의 한 레코드를 지식베이스의 슬롯에 값으로 받아 작업을 처리한 다음, 다음 레코드를 같은 슬롯에 받아 다시 작업을 수행하게 된다.

5.2.3 Grouped operation

Grouped operation은 한번의 동작으로 많은 레코드를 한꺼번에 retrieve 하거나 write 하는 것을 말하며, 특히 Grouped retrieve는 데이타베이스로 부터 추출한 여러개의 레코드를 지식베이스에 정의된 클래스 아래 동적 객체를 생성하여 한 객체에 한 레코드씩 객체 슬롯에 데이타가 저장된다.

여기서는 위의 3가지 Operation 중에서 가장 많이 사용하는 Atomic Operation을 이용하였다.

5.3 Atomic Retrieve

이것은 하나의 데이타베이스 레코드로 부터 값을 추출하여 지식베이스의 객체의 속성 슬롯에 값을 넘겨주는 것이다.

4.1절에서 구현한 SHIPS 테이블에서 DWT가 100,000 ton 이상인 배를 검색하려면 우선 지식베이스는 아래와 같이 나타낼 수 있다.

```
IF    Reset      dummy_object. dummy_cursor
      Retrieve   "nexpert/nexpert"
Hypo  atomic_start
```

여기서 이 검색이 Atomic retrieve가 되기 위해서는 cursor 슬롯이 UNKNOWN 상태로 되어야 한다. 즉, Reset을 이용하여 cursor 슬롯의 값을 UNKNOWN 상태로 지정하였다. Retrieve 다음의 문자열은 ORACLE 데이타베이스를 접근하기 위한 것으로서, 그 구문은 "username/password" 형태로 표현된다. Fig.4 는 Atomic retrieve를 위해 데이타베이스의 필드와 지식베이스의 객체슬롯을 매핑시키는 작업 윈도우이다.

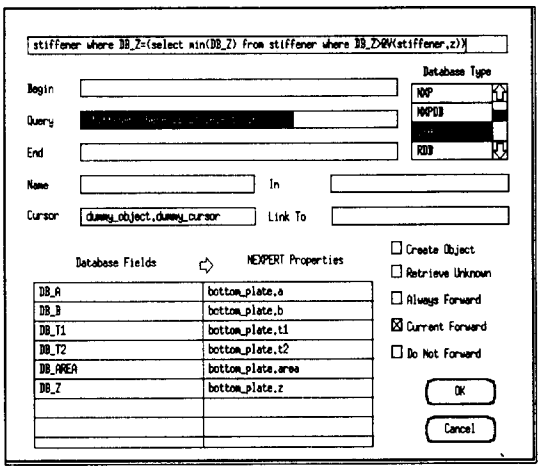

Fig. 4 Atomic Retrieve Window

이 연산이 하나의 레코드의 값을 얻기 위한 연산이라는 것을 나타내기 위해 cursor 부분에 cursor 슬롯을 정의하고 이를 UNKNOWN 값으로

고정시켰다. 다음으로 관련되는 데이타베이스의 필드와 지식베이스의 객체슬롯을 연결하여 1:1로 값을 가져오도록 정의한다.

Fig.5 는 데이타베이스의 레코드와 지식베이스의 객체슬롯과의 관계를 도식화한 그림이다.

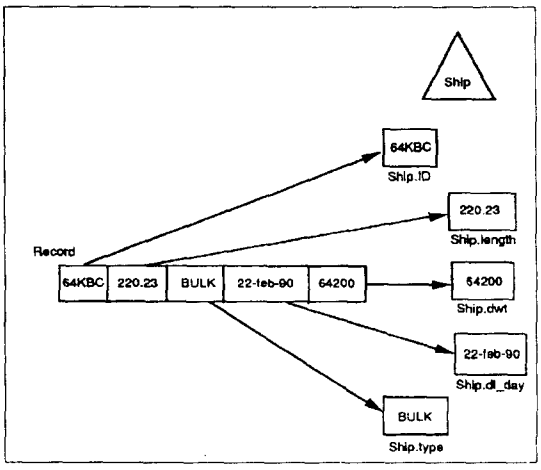

Fig. 5 Relation between fields of database and objects

일반적으로 Atomic retrieve에서 Query에 적합한 데이타베이스의 레코드가 여러개 존재할때는 맨 먼저 선택된 레코드 값이 선택된다.

여기서, Query 의 조건 부분의 값을 고정시키지 않고 다른 객체 슬롯의 값을 이용할 수 있도록 동적 값(Dynamic values)을 부여할 수 있다. 이때는 @V(Slotname) 의 형태로 나타낸다.

본 논문에서는 Table 2 에의해 구현된 보강재 DB를 이용하여 Scantling 에 의해 구해진 보강재의 단면계수에 따른 적합한 보강재를 검색하여 설계에 이용된다. 이때 사용되는 지식베이스의 규칙은 다음과 같다.

```
IF    Reset      dummy_object.dummy_cursor
      Yes        get_section_modulus
      Retrieve   "nexpert / nexpert"
Hypo  get_bottom_stiff
```

여기서도 Scantling 을 통해 단면계수 값을 얻은 다음 Atomic retrieve 를 이용하여 데이타베이스의 보강재 DB로 부터 적합한 보강재를 선택하게 된다.

Fig. 6은 지식베이스의 추론 후 얻어진 bottom_stiff 객체의 값을 보여주고 있다.

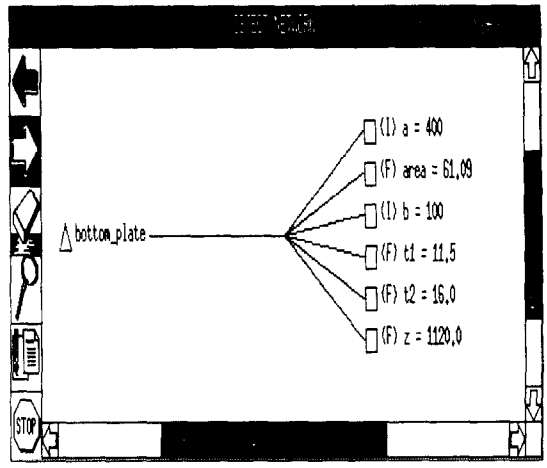

Fig. 6 Retrieval of stiffener data from database

6. API 프로그래밍을 통한 시스템 통합 기법

6.1 Call-In, Call-Out 개념을 이용한 구조설계용 API 구현

대부분의 범용 전문가시스템 쉘에서는 개방형 설계를 지원하기 위하여 Call-In, Call-Out 개념을 제공하여, 사용자가 필요에 따라 사용자 프로그램에서 지식베이스를 제어하고, 지식베이스에서도 외부 환경들을 제어할 수 있도록 하고 있다 [9]. Call-In, Call-Out을 이용하여 외부 환경과의 접속을 수행할 수 있도록, Nexpert Object에서는 API (Application Programming Interface)를 제공하여 C나 FORTRAN 프로그램을 이용한 응용프로그램을 생성할 수 있도록 지원하고 있다.

Call-In, Call-Out 의 Callable Interface를 이용한 응용 프로그램의 개발에 대한 개념을 Fig.7 에

나타내었다.

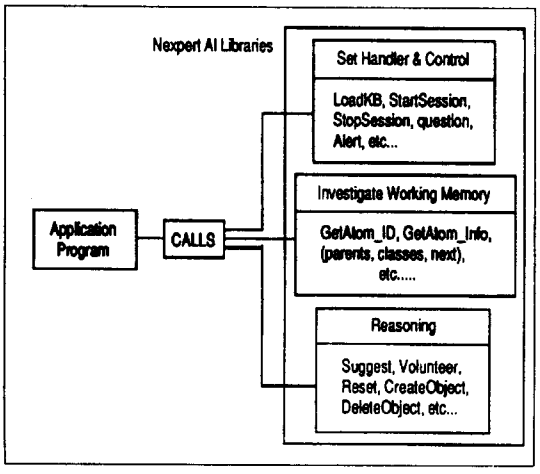

Fig. 7 Concept API using callable interface

Nexpert Object 는 라이브러리 형태로 제공되므로 응용 프로그램내에서 함수를 호출하듯이 호출만 하면 추론을 위한 여러가지 기능을 수행할 수 있다. 응용 프로그램에는 다음과 같은 항목들이 포함된다.

- 전문가시스템 쉘에서 제공되는 라이브러리를 호출하기 위한 헤더화일 선언
  : <nxpdef.h>
- C 프로그램에 관계되는 헤더화일 선언
  : <stdio.h> ...
- Main() 함수 : 각종 함수 호출
- Call-Out 지원을 위한 Handler 선언
  : NXP_SetHandler ( )
- 지식베이스에서 호출된 사용자 정의 루틴

6.2 동적객체의 생성과 API의 구현

일반적으로 지식베이스를 구현함에 있어서 객체를 먼저 정의하고 이들을 이용하여 규칙을 구성하게 된다. 그러나 때로는 이러한 객체의 숫자가 일정치 않아 미리 정의할 수 없을 경우가 생긴다. 선급 규정을 이용하여 선박의 중앙단면의 Scantling을 수행할 때 Plate 를 공작상의 이유로 분할하게 되며, 이들 각각의 Scantling이 달라진다. 또한 몇개로 분할할 것인가는 설계자가 설계 도중에 정하게 되어 미리 이것을 객체로 정의할 수 없다. 따라서 주어진 side plate 분할 입력 데이타에 따라 이를 동적객체로 생성하여, 이들 각각에 대해 사용자가 정의해 놓은 calcu_side_p()라는 함수를 실행시키도록 하는 규칙을 구성하였으며,

```
/*************************************************
API for creation of dynamic object
Knowledge base : dynamic.kb
Key concepts    : Call-In, Call-Out
Data            : 199 . .
*************************************************/

#include <stdio.h>
#include <nxpdef.h>
void calcu_side_p ( ) ;
float side_t [10] ;

main ( )
{
AtomId testHypo, no_plate_atom;
KBId testKB;
float plateno=3.0;

NXP_Control (NXP_CTRL_LNIT) ;
NXP_SetHandler (NXP_PROC_EXECUTE, calcu_side_p,
                (char*) 0 ) ;
NXP_LoadKB ("dynamic.kb",&testKB) ;
printf ("KB loading. .\n" ) ;
NXP_GetAtomId ("side_shell_result_r",&Hypo,
               NXP_ATYPE_SLOT ) ;
NXP_Suggest (testHypo, NXP_SPRIO_SUG) ;
NXP_GetAtomId ("no_plate",&no_plate_atom,
               NXP_ATYPE_SLOT) ;
NXP_Volunteer (no_plate_atom, NXP_DESC_FLOAT,
           (char*) &plateno,
          NXP_VSTRAT_SET | NXP_VSTRAT_CURFWRD ) ;
NXP_Control (NXP_CTRL_KNOWCESS) ;
NXPGFX_Control (NXPGFX_CTRL_INIT) ;
NXPGFX_Control (NXPGFX_CTRL_START) ;
}
void calcu_side_p ( )
{
. .
. .
}
```

Fig. 8 API program for dynamic objects

Fig.8 은 이 지식베이스를 외부에서 제어하고자 하는 API 프로그램이다.

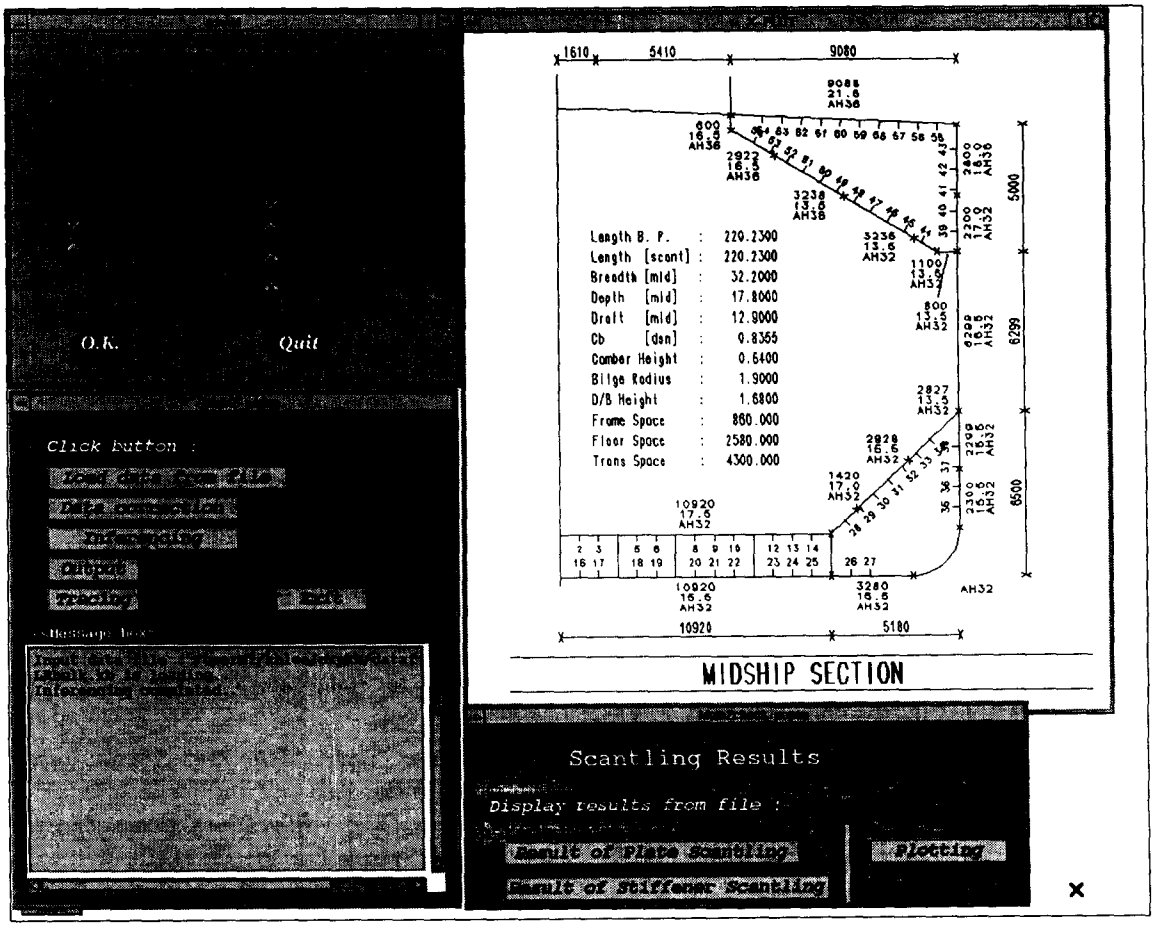

Fig. 9 Scantling results of 64K Bulk Carrier

6.3 선박 중앙단면 Scantling을 위한 API 구현

본 연구에서는 Nexpert Object와 외부환경 (사용자 정의 함수, 데이타베이스, 그래픽 라이브러리, GUI 등) 과의 인터페이스에 초점을 두었다.

CI (Callable Interface)를 이용한 사용자 정의 함수의 인터페이스, 데이타베이스와의 인터페이스에 대해서는 앞에서 언급하였고, 여기서는 GUI (Graphical User Interface) 를 통한 API의 구현 및 Nexpert Object 에로의 접근을 시도하였다.

GUI를 통한 API의 구현은 API 프로그램을 GUI 프로그램의 하나의 Callback Function으로 사용하게 된다 [10].

Fig.9는 64K Bulk Carrier에 대하여 Lloyd 선급을 적용하여 Scantling 을 수행하여 얻은 결과를 가시화한 것이다 [11,12].

7. 결 론

본 연구에서는 범용의 전문가 시스템 개발도구를 이용하여 선박설계, 특히, 중앙단면 구조설계에의 응용 가능성을 검토해 보았으며, 특히 객체지향 개념에 의한 지식표현 방법, 지식베이스와 데이타베이스의 인터페이스 기법, API 를 통한 시스

템의 통합에 관해 알아 보았다.

본 논문의 의의 및 결론은 다음과 같다.

첫째, 구조설계 작업 과정의 지식을 효과적으로 표현하고자 클래스, 객체, Method 등을 이용하여 설계대상을 객체지향 개념으로 표현하고, 설계 자체에 관한 지식을 규칙으로 표현하는 Hybrid형 지식표현을 도입함으로써 설계지식을 효과적으로 표현할 수 있었다.

둘째, 지식베이스에서 추론 도중 필요한 데이터를 데이타베이스로 부터 Retrieve 하고, 추론된 결과를 다시 데이타베이스에 Write 하게 하는 지식베이스와 데이타베이스의 통합 기법을 연구함으로써 단편적인 시험용 전문가시스템이 아니라, 설계정보의 효과적인 활용을 통한 설계 전문가시스템의 구현이 가능하게 되었다.

세째, Call-In, Call-Out 개념에 의한 API 구현을 통해 설계자가 쉽게 전문가 시스템의 지식베이스를 사용할 수 있으며, 공학설계에 필요한 모든 환경이 통합화된 효율적인 전문가시스템의 구현이 가능하다.

네째, 그래픽 사용자 인터페이스를 도입하여 입출력의 가시화, 결과의 가시화, 결과의 검증을 위한 지식베이스의 Trace-back 기능 등을 통해 효율적인 설계를 지원할 수 있었다.

즉, 문제 해결에 있어서 전문가 시스템이 단독으로 사용되지 않고, 외부 프로그램, 데이타베이스 기술, 가시화 기술이 API를 통하여 함께 접목됨으로써 효율적인 시스템을 구성할 수 있었다.

본 연구 결과의 활용 방안으로서는 과학기술처의 지원을 받아 산학연 공동으로 개발중인 선박설계 생산 전산시스템 개발 (CSDP) 연구사업의 일환으로 수행되고 있는 설계 전문가 시스템의 지식베이스화 기초 기술로 활용할 예정이다.

후 기

본 연구는 1993년도 과학기술처에서 시행한 기본연구 사업 결과의 일부이며, CSDP 개발 사업의 기반기술로 활용되고 있다.

참 고 문 헌

[1] 이경호, "Nexpert Object 를 이용한 전문가시스템 개발", 한국과학기술원 산학협동공개강좌 Proc., 1994. 1.

[2] Agaki 외, "설계 Expert System의 기초와 응용 (일본어)", 코로나사, 1990년 5월.

[3] J. S. Gero, *"Expert System for Design : A Framework"*, pp.721-727, Proc. of The World Congress on Expert Systems, Dec. 1991.

[4] R. Coyne, M. Rosenman, et al., *"Knowledge-Based Design Systems", Addison-Wesley, 1990.*

[5] *"Nexpert Object Version 2.0 - Introduction Manual"*, Neuron Data Inc. Oct. 1990.

[6] 문송천 역, "데이타베이스 시스템 총론", 형설출판사, 1988.

[7] Jonathan Sachs, *"SQL*Plus User's Guide"*, Oracle Co., 1986.

[8] *"Nexpert Object Version 2.0 - Database Integration Guide"*, Neuron Data Inc., Oct., 1990.

[9] *"Nexpert object version 2.0 - Application Programming Interface Programmer's Reference"*, Neuron Data Inc., Oct. 1990.

[10] Douglas A. Young, *"The X Window System Programming and Applications with Xt : OSF/Motif Edition"*, Prentice-Hall, 1990.

[11] 윤덕영 외, "Bulk Carrier 중앙단면 종강도 구조의 최적설계 System 개발", 대우조선 기술연구실, 1989년 2월.

[12] 이경호, "선체 구조설계용 Rule의 지식기반 시스템 개발", 한국기계연구원 선박해양공학연구센터 연구보고서, UCE538-1747.D, 1993년 12월.

[13] 이경호 외, "객체지향적 지식표현과 개방형설계에 의한 구조부재 치수 결정 지원 시스템 개발", 대한조선학회논문집, 제30권 제2호, 1993.5