한국 CAD/CAM 학회 논문집
제 5권 제 3 호 2000년 9월 pp. 224-231

학술논문

제품구조 및 구성을 위한 옵션조합관리 기능 개발

김선호*, 정병용**, 주경준***, 정석찬****

Development of Option Specification Management Function for Product Structure and Configuration

Kim, S. H.*, Jeong, B. Y.**, Joo, K. J.*** and Jeong, S. C.****

ABSTRACT

In order to satisfy customers' various needs in the market, manufacturing companies tend to add similar variants to an existing product model. This variety causes immense growth of product configuration and makes data management uncontrollable. In order to resolve this problem, we propose a method to efficiently represent the variants in a single schema of product structure and configuration and manage the product configuration. First of all, the product structure which adopts the concept of features and options is suggested. Second, the method to represent option specifications which restricts option configurations is proposed. Finally, the prototype module, which inspects if a product configuration violates the option specifications, is introduced.

Key words : Product Structure, Configuration, BOM, PDM, Feature, Option, Option Specification

1. 개 요

PDM(product data management)에서 중요한 기능 중의 하나는 *제품구조 및 구성*(product structure and configuration) 관리 모듈이다. 제품구조는 부품간의 상하 조립관계를 나타내는 것으로서 상위부품, 하위부품, 소요 수량 등의 정보를 가지고 있다. 제품구성은 한 제품의 구성 또는 조립 사양을 나타내는 것으로서 옵션(option)과 같은 변형품(variant) 정보, 대체품 및 유효성(effectivity) 정보 등을 포함하고 있다[1,2].

제품구조는 일반적으로 *BOM(bill of material) 구조*, *부품목록(parts list) 구조*로 구분된다[3]. 한 상위부품의 바로 아래에 동일한 하위부품이 여러 개 있을 경우, BOM 구조에서는 동일한 여러 개의 하위부품들을 하나의 하위부품과 그 동일한 하위부품의 수량으로 표현하게 된다. 그러나, 부품목록 구조에서는 동일한 하위부품들을 각각 나타내고 있다. 또한, 부품목록 구조에서는 하나의 부품과 바로 상위 부품간의 1단계 조립관계(immediate parent-child relationship)뿐만 아니라 BOM구조에는 없는 specified_higher_usage_occurrence와 같은 속성을 이용하여 더 위의 상위부품과의 조립관계도 표현된다[3].

PDM에서 이용되는 제품구조는 BOM구조를 다루고 있으므로 여기서는 BOM에 대해 집중적으로 설명하기로 한다. BOM을 현장에서 제대로 활용하기 위해서는 제품구성까지 효율적으로 관리할 수 있어야 한다. 즉, 업체에서 관리하는 모델이 다양해지고 한 모델 내에서도 변형품이 많아지면 관리해야 할 제품구성의 종류는 기하급수적으로 많아진다. 그리고, 이와 관련하여 BOM정보 역시 제품구성별로 모두 관리해야 하므로 중복되는 데이터가 많아진다. 또한, 한 부품에 대한 설계가 변경되면 이와 관련된 모든 BOM정보를 찾아 전부 수정하여야 하는 불편함이 따르게 된다. 따라서 이러한 문제점을 해결하기 위하여 제품구조와 제품구성을 모두 고려하여 하나의 데이터 구조로 표현할 수 있는 BOM이 필요하다.

지금까지 제품구조 및 제품구성을 고려하는 BOM에 대한 연구가 많이 진행되어 왔다. 고석완[4], Trappey,

*정회원, 명지대학교 산업공학과
**대우정보시스템(주) 기술연구소
***ETRI 컴퓨터소프트웨어기술연구소 전자상거래연구부
****동의대학교 경영정보학과

et al.[5], Chung, *et al.*[6]는 BOM정보와 제품정보를 동적으로 표현하기 위하여 Relational DB대신에 객체지향 개념을 이용하여 BOM시스템을 설계하였다. 그러나 이러한 연구에서는 제품구조에 대한 내용을 중심으로 설계하였으며 제품구성의 개념은 별로 고려되지 않고 있다.

제품구성의 기본인 변형품을 표현하는 방법으로 지금까지 modular BOM, family BOM, matrix BOM, add/delete BOM 등이 소개되어 있다[7,8,9]. Modular BOM은 제품구조 상위의 주요부품들을 공통부품 그룹과 변형품 그룹으로 구성하고 상위 레벨에서 변형품 조합의 BOM을 구성하는 방식이다. Family BOM은 modular BOM과 유사하나 최상위 부품을 유사한 그룹으로 분류하는 것이 다르다. 이 방식들에서는 제품구조 상위레벨의 주요한 부품들의 소요량만 파악할 수 있고 하위레벨의 단위부품까지 소요량을 파악하기 위해서는 별도의 그룹별 BOM구조를 다시 만들어야 하는 불편함이 있다. matrix BOM은 부품의 수량을 제품별로 테이블의 형태로 나열하고 변형품도 명시하는 방식이나 제품의 구조가 표현되지 않아 중간 조립품의 정확한 소요량을 파악하기 어려운 단점이 있다. add/delete BOM은 변형품이 생길 때마다 부품을 해당부품을 교체, 추가, 삭제하는 방식이다. 그러나 부품을 추가, 삭제하면서 제품구조를 변형해야 하는 번거로움이 있다.

최근에는 변형품을 표현하는 방법으로서, 변형품 그룹에 generic code number를 주고, 변형품을 구분하는 색깔이나 기능 등의 특성에는 parameter 값을 주는 generic BOM이 제시되었다. 여기서는 parameter가 상하 부품간에 상속되도록 정의하여 상위에서 변형품을 정의하면 하위부품에서 그 변형품이 선택되도록 구성하였다[10,11]. McKay, *et al.*[12]는, 제조의 관점(assembly view)에서 개별적인 제품들로 표현된 기존의 데이터베이스 모델과, 영업의 관점(sales view)에서 변형품이 많은 product family를 나타내는 모델을 데이터의 중복 없이 통합하여 표현할 수 있는 framework-based product data modeling 기법을 제시하였다. 김선호[8]은 피쳐(feature)와 옵션을 고려하여 제품구조 및 구성을 표현하는 방법을 제시하였다. 여기서는 피쳐는 옵션부품들을 대표하는 이름으로서 실제부품이 아닌 가상부품으로 처리된다. 그래서 사용자는 피쳐별로 옵션부품을 선택하여 제품구성을 하도록 되어 있다.

그러나 이렇게 변형품을 제품구성에 반영하면서 생기는 문제가 있다. 기능상의 제약사항이나 조립상의 문제점으로 인하여 같이 사용할 수 없는 옵션이 존재하거나 반드시 같이 사용해야 하는 옵션이 존재할 수 있다. 예를 들어 자동차에서 3.0 L 엔진 옵션에는 반드시 대형 배터리 옵션을 사용해야 하는 기능상의 제약사양이 발생할 수 있다. 이러한 제약사항을 표현하는 방법으로 generic BOM에서 옵션을 선택할 때 제약사항을 주어 제품구성을 표현하는 방법이 제시되었다[9]. 이 방법은 모든 1단계 상하위 부품그룹(product item) 간에 제약사항(conversion rules)들을 주는 방식이다. 최근에 PDM Enabler[13]에서는 IDL (Interface Definition Language)을 이용하여 이러한 제약사양에 대해 PDM간에 교환할 수 있는 인터페이스 표준을 제시하여 이것의 중요성을 보여주고 있다.

본 연구에서는, 변형품을 표현하는 방법으로 본 저자들이 제시한 피쳐 및 옵션 표현 방법[2]을 기본으로 하여 제품구성의 제약사양을 표시하는 방법을 제시한다. 이 방법은 generic BOM[14]과 유사하나 두 가지 측면에서 다르다. 첫째, generic BOM에서는 BOM구조의 모든 구성품들을 부품그룹으로 정의하였으나 여기서는 옵션이 있는 부품 그룹만을 피쳐로 정의하였다. 또한 제약사양이 모든 1단계 상하위 부품그룹(product item)간에 주어지나 현실적으로 모든 1단계 상하위 부품그룹(product item)간이 아니더라도 제약사양이 주어지지 않는 경우도 있다. 예를 들면 "엔진과 베터리는 상하위 부품그룹 관계가 아니나 엔진에 3.0 L이상이면 베터리는 대용량을 사용해야 한다"는 경우가 발생한다. 본 연구에서는 상하위 조립에 관계없이 제약사항을 줄 수 있도록 하였다.

연구내용은 다음과 같다. 첫째, 제품구조 및 구성관리 모델로서 피쳐와 옵션을 이용한 옵션관리 모델을 제시한다. 둘째, 제품구성의 제약사양을 표현하는 옵션조합식을 제시한다. 마지막으로, 제시된 옵션조합식을 이용하여 옵션조합식을 등록 및 관리하고, 제품구성을 검사하는 옵션조합관리기능을 개발하였으며 그 개발 내용을 설명한다.

2. 옵션관리 모델

2.1 옵션을 고려한 제품구조

제품의 일반적인 BOM 구조는 Fig. 1에서와 같이 부품의 상하 관계와 하위부품의 수량으로 표현된다. 제품들 중에서 판매를 목적으로 제조되는 최상위 부품을 일반적으로 모델이라고 부르며 하나의 모델에는 사용자의 요구에 따라 변형품이 존재하게 된다. 즉, 선택할 수 있는 부품 사양이 존재하여 한 모델 내에서 선택적으로 제품을 구성할 수 있게 된다. 이

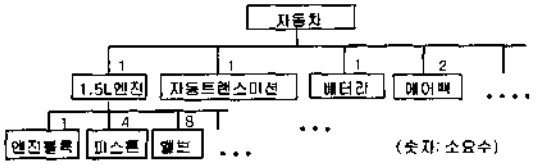


Fig. 1. A typical BOM structure.

때 이러한 부품을 옵션이라고 한다. 이러한 옵션이 많아질 경우 가능한 제품구성의 수가 옵션 수에 비례해서 증가하게 된다. 예를 들어, 자동차의 한 모델에, 차체 색깔 10종류, 엔진배기량 3종류, 트랜스미션 4종류, 브레이크 2종류의 옵션이 있을 때, 240종류의 제품구성이 존재하게 된다. 이러한 변형품을 고려하지 못하는 일반적인 BOM 구조로 저장할 경우 각 제품구성별로 BOM 데이터를 저장 관리해야 한다. 이때, 중복되는 데이터의 양이 매우 많아지고, 한 부품을 수정할 경우 각 제품구성별로 그 부품을 찾아 수정해야 하는 불편함이 따르게 된다.

이러한 문제점을 해결하기 위하여 여기서는 하나의 모델에 여러 가지의 옵션을 표현할 수 있는 *옵션 BOM*을 제시한다. 여기서 하나의 모델은 옵션으로 구성이 가능한 변형품의 집단으로 정의하며 옵션 이외의 구성 부품들이 달라지면 다른 모델로 간주한다. 옵션 BOM에 대한 예제가 Fig. 2(편의상 수량은 생략함)에 나타나 있다. 이 예에서는 최상위 제품인 자동차는 엔진, 기어박스, 배터리 등으로 구성된다. 여기서 여러 개의 옵션들을 대표하는 품목명을 나타내기 위하여 피쳐(feature) 라는 가상부품이 도입되었다. 엔진, 트랜스미션, 배터리, 에어백, 에어컨은 가상부품에 해당되는 피쳐들이다. 이 피쳐는 그 아래에 실제부품인 옵션들로 구성된다. 여기서 엔진 피쳐는 1.5(E15), 2.0(E20), 3.0리터(E30)의 옵션으로, 트랜스미션 피쳐는 4단(G4), 5단(G5), 자동(GA) 옵션으로 구성된다. 피쳐는 다시 *필수피쳐와 선택피쳐*로 구분된다. 필수피쳐는 하나의 옵션을 의무적으로 택해야 하는 경우이고, 선택피쳐는 옵션을 택하지 않아도 관계없는 경우이다. 이 예에서 엔진, 트랜스미션, 배터리는 옵션을 반드시 하나 택해야 하는 필수피쳐이며 에어백, 에어컨은 택하지 않아도 관계없는 선택피쳐이다.

2.2 옵션관리를 위한 데이터 모델

이러한 옵션 개념을 나타내는 E-R 다이어그램이 Fig. 3에 요약되어 있다. 하나의 모델은 여러 개의 피쳐를 가지게 되므로 이것을 표현하기 위하여 모델-피쳐 엔티티가 존재한다. 모델별로 피쳐의 구성이 달라질 수 있으므로 여기서는 모델번호와 피쳐부품번호가 키(key)가 된다. 그리고 이 엔티티에는 각 피쳐가 필수인지 선택인지를 판단하는 *피쳐종류* 속성이 포함된다. 또한 각 피쳐는 여러 가지 옵션들을 가지고 있으므로 이것을 표현하는 피쳐-옵션 엔티티가 존

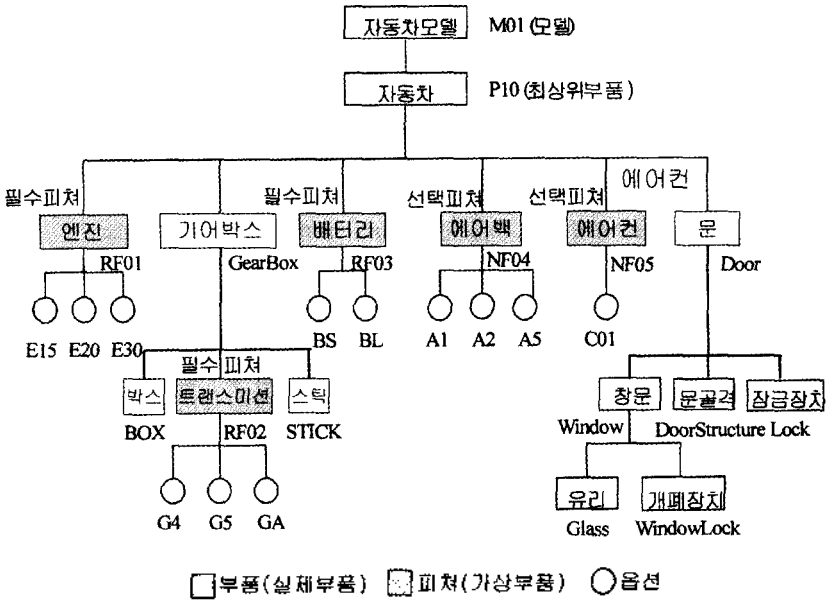


Fig. 2. An example of option BOM.

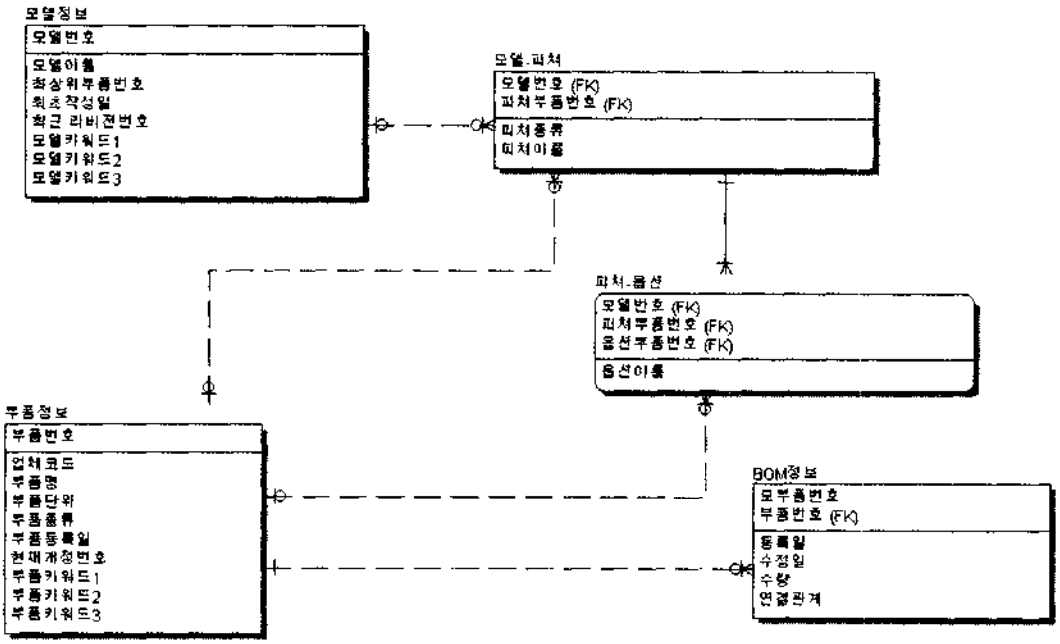


Fig. 3. ER diagram for option.

재한다. 여기서 피처-옵션 엔티티의 키는 모델번호, 피처부품번호, 옵션부품번호가 되며 모델번호, 피처부품번호가 모델-피처 엔티티로 연결시켜 주는 외부 키(foreign key)역할을 하게 된다. 그리고, BOM 구조를 나타내기 위하여 BOM정보 엔티티가 모부품번호, 자부품번호, 수량 등의 속성을 가지게 된다. 피처가 고려된 데이터 구조하에서 일반적인 방식으로 BOM을 전개할 경우 가상부품인 피처가 포함되어 전개되는 문제점이 있다. 이 문제를 해결하기 위하여 우선, 피처를 실제부품과 구별하는 속성을 부품정보 엔티티에 포함하여야 한다. 여기서는 이 속성을 *부품종류* 라고 칭하였으며 최상위부품, 중간조립품, 부품, 원자재, 부자재, 가상부품, 등으로 구분된다. 그래서, 부품의 종류가 가상부품이면 피처로 간주하고 BOM 전개시 제외할 수 있도록 한다. 여기서 부품종류에 옵션 부품을 별도로 정의하지 않는 이유는 한 모델에서 피처 내의 옵션이었던 부품이 다른 모델에서는 일반부품으로 사용될 수도 있기 때문이다.

BOM 전개시 피처를 제외하기 위해서는, BOM정보 엔티티의 모든 자부품번호마다 피처인지를 확인하기 위하여 부품정보 엔티티를 탐색하게 되므로 전개시간이 매우 길어지는 단점이 있다. 즉, 전개과정을 거치면서 많은 수의 transaction을 유발시키므로 성능저하의 요인이 된다.

그래서 전개 시간을 줄이기 위하여 BOM 엔티티에, 모부품과 자부품간에 관계를 나타내는 *연결관계* 속성을 추가하였다. 연결관계 속성은 상하관계로서 부품과 부품, 부품과 피처, 피처와 옵션 등의 관계를 나타내어 BOM엔티티의 레코드마다 피처가 포함되어 있는지를 즉시 알 수 있다. 이 속성은 다음과 같은 값을 갖게 된다.

MtT–모부품이 모델이고 자부품이 최상위부품인 경우, 자부품이 최상위부품인 경우
모부품은 실제로 존재하지 않으며 모델과 최상위부품은 동일한 부품을 나타낸다.

PtP–모부품과 자부품이 모두 부품인 경우.

PtF–모부품이 부품이고 자부품이 피처인 경우.

FtO–모부품이 피처이고 자부품이 옵션인 경우.

위의 연결관계속성 값에서 가운데의 "t"를 중심으로 좌측이 모부품, 우측이 자부품를 나타내며, "M", "T", "P", "F", "O"는 각각 모델, 최상위부품, 부품, 피처, 옵션을 나타낸다. 이러한 연결관계 속성은 BOM 등록시에 부품정보 엔티티의 부품종류를 참조하여 자동 생성된다.

3. 옵션조합관리 모델

제품구조 및 구성 모듈에서 옵션들을 정의할 때

특별한 제약사양을 가지는 옵션들이 존재할 수 있다. 예를 들어 "A/C와 자동변속기 옵션을 택할 경우 6기통 엔진 옵션을 사용해야 한다"는 제약사양이 있을 수 있다. 제약사양은 설계나 제조과정에서 성능을 향상시킬 목적이나 제조의 편의를 위해 생길 수도 있으며 또한 고객의 요구에 따라 생길 수도 있다. 이러한 제약사양에 따라 옵션의 조합을 올바르게 구성할 수 있도록 하는 것이 옵션조합관리 기능이다. 여기서는 옵션 BOM과 연관하여 제약사양을 표현하는 옵션조합식의 문법을 제시하고 제약사항에 따라 BOM을 처리하는 기능을 설명한다.

3.1 옵션조합식

옵션조합식은 제품구조 및 구성에서 각 옵션들이 가지게 되는 제약사양들을 If... then...의 형태로 표현한 것이다. 여기서는 If부분을 *조건식*(Condition)이라 하고 then부분을 *제약식*(Restriction)이라고 정의한다. 이 옵션조합식은 제품구성에 포함된 옵션이 조건식에 포함되어 있으면 제약식을 만족하는지 검사하게 된다. 조건식과 제약식은 AND(&), NOT(~), OR(+), XOR(@)의 네 개의 논리기호와 두 개의 구분자(쉼표, |)를 이용하여 표현된다. 각 논리기호 뒤에는 옵션부품번호들이 오게 되며, 옵션부품번호들은 해당하는 논리기호에 의해 논리연산 된다. Table 1은 Fig. 1의 예에 대한 옵션조합식을 표현한 예를 보여주고 있다.

3.1.1 AND (&)

이 논리기호는 반드시 같이 사용되야 할 옵션들을 명시한다. Fig. 1에서 엔진 E15를 선택할 경우 Table 1의 옵션조합식 G001에 해당하는 제약식에 따르면 배터리는 반드시 BS를 사용해야 한다는 뜻이다. 또는 엔진 E30을 선택하였을 경우 배터리는 반드시 BL을 사용해야 한다. 이와 같이 AND는 주로 제품 성능 향상이나 다른 옵션을 보조하기 위해 반드시 사용돼야 하는 부품들을 명시하는데 이용된다.

3.1.2 NOT (~)

이 논리기호는 같이 사용될 수 없는 옵션들을 명시한다. Fig. 1에서 엔진을 E15를 선택할 경우 Table 1의 제약식을 보면 에어컨 C01은 선택할 수 없다. NOT이 사용되는 경우는 AND가 사용되는 경우와 유사하며 부분적으로 AND를 대신하여 사용될 수도 있다. 예를 들면, Fig. 1의 예에서 엔진 E15를 선택할 경우 배터리 BS를 선택하기 위해 &BS 대신에 ~BL을 사용할 수도 있다. 여기서 배터리는 필수피쳐이고 옵션이 두 가지 밖에 존재하지 않으므로 이와 같은 사용이 가능하지만, 추후에 배터리 피쳐에 대용량 배터리인 BXL이라는 옵션이 추가되었을 경우에는 문제의 소지가 있으므로 그리 바람직한 방법이라고 볼 수는 없다.

3.3.3 OR (+)

이 논리기호는 하나 이상의 옵션 부품을 선택할 수 있을 때 사용된다. 예를 들어 Fig. 1에서 엔진을 E30을 선택했을 경우, Table 1에서 옵션조합식 번호 G003의 제약식에 의해 에어백 A1은 반드시 선택해야 한다. 그리고 에어백 A2 와 A3의 경우에는 둘 중의 하나를 사용하거나 두 가지를 동시에 사용할 수도 있다.

3.1.4 XOR (@)

이 논리기호는 복수의 부품 중에서 반드시 하나만을 선택할 수 있을 때 사용된다. 이것은 주로 서로 상호 배타적인 관계에 있는 옵션 부품들에 사용된다. 예를 들어 Fig. 1에서 엔진 E20을 선택했을 경우 Table 1의 옵션조합식 번호 G002에 의해 트랜스미션은 G4, G5, GA 중에서 반드시 하나만을 선택하도록 되어있다. 이러한 특성때문에 XOR는 선택피쳐를 택할 경우에는 이용되지 않고 필수피쳐를 택할 때 많이 이용된다.

3.1.5 구분자

옵션조합식은 앞서 설명한 4개의 논리기호(AND, NOT, OR, XOR)와 구분자인 쉼표(,)와 수직선(|), 각 논리기호에 할당되어지는 옵션부품번호들로 구성된다. 구분자는 옵션부품번호와 옵션부품번호, 논리기호와 논리기호 사이를 구분하여 옵션조합식을 해석하기 쉽도록 도와준다. 쉼표(,)는 하나의 논리기호에 할당된 옵션부품번호들을 분리하여 주며, 수직선(|)은 전체 제약식에서 하나의 논리기호가 효력을

Table 1. An application example of option specifications

옵션조합식 번호	모델번호	조건식	제약식
G001	M01	&E15	&BS|@G4,G5|~C01
G002	M01	&E20	&BL|@G4,G5,GA|&A1
G003	M01	&E30	&BL|@G5,GA|&C01|&A1|+A2,A3
G004	M01	&E20,GA	&A1,A2

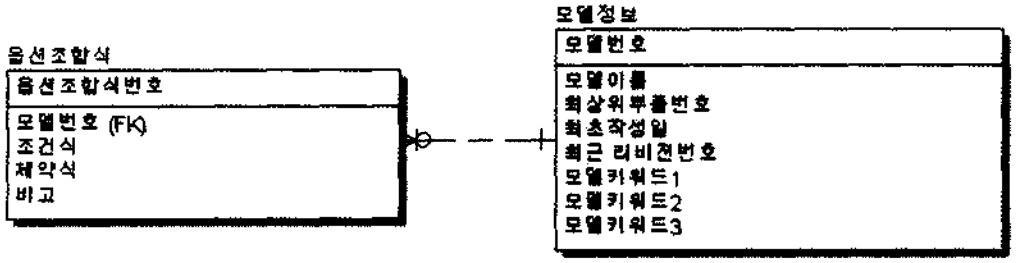


Fig. 4. ER diagram for option specifications

가지는 범위를 표시한다. Table 1에서 엔진 E20을 선택했을 때의 제약식인 "&BLI@G4.G5,GAI&A1"에서 보듯이 쉼표는 옵션부품 G4, G5, GA간을 구분하며 수직선으로 이 세가지 옵션부품들이 XOR 논리기호 범위 내에 있는 것을 나타낸다.

3.1.6 기타 규칙

-옵션조합식은 네 개의 논리기호와 두 개의 구분자, 복수의 옵션부품번호로 구성된다.

-논리기호 간의 연산우선순위는 없으며 논리기호는 중복되어 사용될 수 있다.

-하나의 옵션조합식에서 옵션부품번호는 중복되어 사용될 수 없다.

-옵션조합식 안에 공백은 허용되지 않는다.

3.2 옵션조합식을 위한 데이터 모델

옵션조합식은 모델별로 존재하므로 Fig. 4와 같이 옵션조합식 엔티티는 모델정보 엔티티와 연결된다. 옵션조합식의 조건식과 제약식은 각각 속성으로 정의된다. 그러나, 조건식과 제약식의 길이가 튜플(tuple)별로 달라지므로 데이터 타입은 길이를 가변적으로 수용할 수 있는 "VARCHAR"로 정의하였다.

4. 옵션조합관리 기능 개발

옵션조합관리 기능은 Fig. 5와 같이 옵션조합식 등록 및 관리, 옵션조합 검사 모듈로 구성되어 있다. 옵션조합식 등록 및 관리 모듈은 BOM정보, 모델-피처, 피처-옵션 엔티티에 저장된 데이터를 이용하여 옵션조합식을 등록하고, 등록된 옵션조합식들을 조회, 수정, 삭제하는 기능을 가지고 있다. 그리고, 옵션조합 검사 모듈은, 사용자가 옵션을 선택하여 제품구성을 하였을 때, 선택한 옵션 목록이 옵션조합식에 위배되는지를 검사한다. 이 모듈은 다른 모듈에 비하여 복잡하므로 여기서 더 상세히 설명하기로 한다.

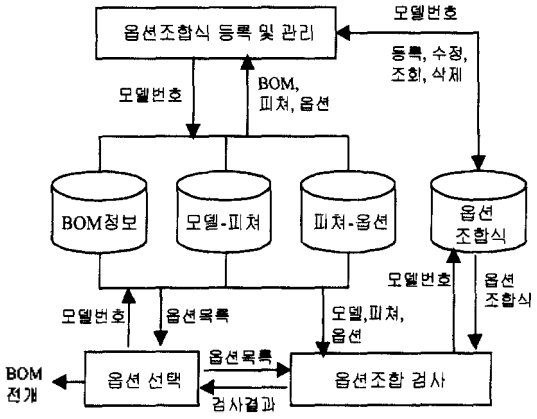


Fig. 5. Data flow diagram of the option specification management module.

4.1 옵션조합 검사 모듈

이 기능은 선택한 옵션목록에 따라 BOM을 전개할 때 실행하게 된다. 이 기능은 옵션조합식에 주어진 조건식과 제약식을 해석하고, 사용자가 한 모델 또는 부품의 BOM을 전개하기 위해 선택한 옵션목록이 옵션조합식에 위배되는지를 비교 검사한다. 이때, 옵션목록이 옵션조합식에 위배되지 않을 때 BOM이 전개된다.

옵션조합 검사모듈의 검사과정을 Flow Chart의 형식으로 나타낸 것이 Fig. 6에 나타나 있다. 이 모듈은 조건식을 옵션제약식 엔티티에서 가져온 후 옵션목록과 비교한다. 이때, 비교하기 위해서 조건식뿐만 아니라 모델, 피처, 옵션들의 데이터가 함께 이용된다. 만일 옵션목록이 조건식에 있으면 다시 제약식을 불러와 옵션목록과 비교한다. 이때, 선택된 옵션목록에서 제약식에 위배되는 옵션이 있는지 검사한다. 위배되는 옵션이 있으면 그 옵션과 옵션조합식을 보여주며, 없을 경우 모든 조건식과 계속해서 비교하게 된다. 비교할 조건식이 더 이상 존재하지 않을 때 이 모듈은 실행을 종료하게 된다.

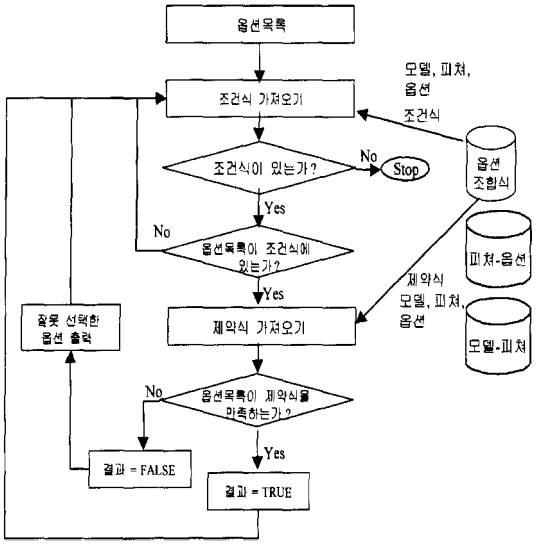


Fig. 6. Procedure of the option specification inspection module.

4.2 옵션조합관리 모듈 개발

옵션조합관리 모듈은 옵션 BOM의 구조를 생성한 후 옵션조합식을 입력할 때, 그리고 BOM 전개를 위해 옵션부품을 구성하였을 때 제대로 구성하였는지를 검사하기 위해 이용된다. 개발된 모듈중 옵션조합식을 입력하는 모듈이 Fig. 7에 나타나 있다. 화면 왼쪽에는 한 모델에 해당하는 피쳐와 옵션들이 전개되어 있다. 옵션조합식을 입력할 때 이 옵션부품들이 이용된다. 오른쪽 상단에는 옵션부품을 선택했을 경우 그 옵션부품에 해당하는 옵션조합식을 보여준다. 그리고 화면 우측 하단에는 옵션조합식을 입력하는 부분을 보여주고 있다. 여기서는 화면 좌측에서 한 옵션부품번호을 선택했을 때, 옵션부품번호란에 선택한 부품의 부품번호가 표시되며 옵션조합식에 입력할 수 있다. 옵션부품이 아닌 부품을 선택했을 때는 옵션부품번호란에 "옵션을 선택해야 합니다." 란 메시지가 표시된다. 옵션조합식의 종류(조건식, 제약식), 논리기호는 사용자가 메뉴에서 선택하게 되어 있다. 구분자는 옵션조합식의 상태에 따라 자동으로 입력된다.

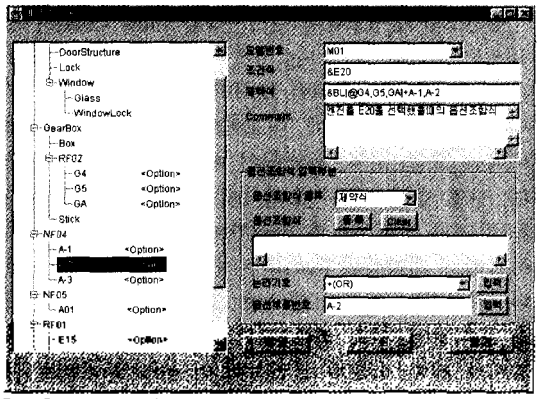


Fig. 7. Display of the input module of option specification.

4. 결 론

대부분의 PDM시스템이나 ERP시스템들은 제품구성을 표현하는 기능들이 다양하지 못한 단점들을 가지고 있다. 이러한 문제점을 해결하기 위하여, 본 논문에서는 옵션을 고려하여 제품구조 및 구성을 표현하는 방법과, 옵션에 주어지는 제약사양을 표현하고 검사하는 옵션조합관리 방법을 제시하였다. 이 방법은 자동차, 항공, 조선 등과 같이 한 모델에 변형품이 많을 경우 효율적으로 활용될 수 있는 특징이 있다. 이 방법은 영업의 관점(sales view)이나 제조의 관점(manufacturing view)중 어느 한 관점에서 옵션을 고려하면서 제품구성에 활용할 수 있다. 왜냐하면 영업의 관점에서 보는 옵션과 제조의 관점에서 보는 옵션이 다를 수 있기 때문이다. 그러나, 기업에서는 하나의 제품구조 및 구성으로 운용되어야 하므로 영업의 관점과 제조의 관점을 모두 고려하여 제품구조 및 구성을 표현할 수 있는 방안이 필요하다.

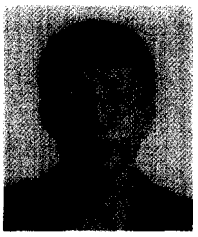
그리고 이러한 옵션을 표현함으로써 제품구조가 일반적인 BOM구조에 비하여 복잡해지게 되며 BOM 전개 시간이 길어지는 단점이 있다. 예를 들어 정전개시 부품이 피쳐인지를 파악해야 하며 피쳐일 경우 옵션목록에서 선택된 옵션을 파악하여야 하는 절차를 거치게 된다. 또한 옵션조합식에 의해 옵션의 제약조건들을 점검해야 하므로 연산시간이 증가하게 된다. 여기서는 이러한 BOM전개 방법과 효율성에 대해서는 본 논문에서 언급하지 않았으나 앞으로 이에 대해 좀 더 연구되어야 할 것으로 예상된다. BOM구조가 복잡해지면 효율성을 높이는 하나의 방안으로는 객체지향 개념의 데이터 구조와 전개방식을 적극적으로 연구할 필요가 있다고 생각된다.

한편, 여기서는 옵션조합식을 문자열 형태로 저장하여 이를 파싱(parsing)하는 방법을 사용하였는데 이러한 식이 많아질 경우 if-then의 production rule과 추론엔진을 이용하는 것도 고려해 볼 수 있다. 또한, 옵션조합식이 많아지거나 수정을 할 때 옵션조합식간에 상충되는 현상이 나타날 가능성이 있다. 현

재로는 작업자가 주의해서 옵션조합식을 만들어야 하나 앞으로 효율성을 높이기 위해서는 상충방지 기법이 별도로 연구되어야 할 것이다.

참고문헌

1. 권용성, 김선호, "이종분산환경하에서의 CORBA기반의 Product Structure 및 Configuration Management 시스템 개발", *IE Interface산업공학*, vol. 13, no. 3, 2000.
2. 김선호, *제품구성 관리기 개발*, ETRI 위탁연구보고서, 1999년 5월.
3. ISO 10303-44, *Industrial Automation Systems and Integration - Product Data Representation and Exchange, Part 44: Integrated Generic Resources: Product Structure Configuration*, 1994.
4. 고석완, 김선호, 정석찬, "Option을 고려한 객체지향형 Product Structure 시스템 설계", *대한산업공학회지*, vol. 24, no. 3, pp. 457-473, 1998. 9.
5. Trappey, A. J. C., Peng, T. K. and Lin, H. D., "An Object-Oriented Bill-Of-Materials System For Dynamic Product Management", *the 3rd International Conference on Computer Integrated Manufacturing*, vol. 1, pp. 629-636, 1995.
6. Chung, Y. and Fischer, G. W., "A Conceptual Structure and Issue for an Object-Oriented Bill of Materials (BOM) Data Model", *Computers and Industrial Engineering: An International Journal*, vol. 26, no. 2, pp. 321-339, 1994.
7. Tersine, R. J., *Principles of Inventory and Materials Management*, 4th Edition, PTR Prentice Hall, 1994.
8. 김선호, 문희석, "Family BOM을 통한 효율적인 설계정보 관리에 대한 연구", *대한산업 공학회/경영과학회 '96 춘계학술대회 논문집*, pp. 346-351, 1996.
9. 이한표, 이춘열, 이국철, Family BOM데이타베이스 구조에 대한 대안: "목적별 BOM 연결 구조의 간접표현 방안", *대한산업공학회 95 추계학술대회 논문집*, pp. 195-202, 1995.
10. Erens, F. J., *The Synthesis of Variety Developing Product Families*, Eindhoven University of Technology, 1996.
11. Hegge, H. M. H. and Wortmann, J. C., "Generic bill-of-material: a new product model", *International Journal of Production Economics*, 23, pp. 117-128, 1991.
12. McKay, A., Erens, F. and Bloor, M. S., "Relating Product Definition and Product Variety", *Reseach in Engineering Design*, pp. 63-80, 1996.
13. *PDM Enablers Joint Proposal to the OMG in Response to OMG Manufacturing Domain Task Force RFP1*, mfg/98-01-01, Submitted by Digital Equipment, Fujitsu, IBM, Matrix One, SDRC, and Sherpa, 1998.
14. Van Veen, E. A. and Wortmann, J. C., "New developments in generative BOM processing systems", *Production Planning and Control*, vol. 3, no. 3, pp. 327-355, 1992.



김 선 호

1979년 서울대학교 산업공학과 학사
1989년 Pennsylvania State University 산업공학 석·박사
1992년~현재 명지대학교 산업공학과 교수
관심분야: PDM, workflow management, B2B EC

정 병 용

1999년 명지대학교 산업공학과 학·석사
2000년~현재 대우정보시스템(주) 기술연구소 재직 중
관심분야: PDM, scheduling

주 경 준

1976년 고려대학교 산업공학과 학·석사
1978년~1991년 삼성전자 생산관리부장
1991년~현재 ETRI 컴퓨터소프트웨어기술연구소 전자상거래연구부 팀장
관심분야: 전자상거래, SCM

정 석 찬

1987년 부산대학교 기계설계학과 학사
1993년 일본 오사카 부립대학 경영공학과 석·박사
1993년~1999년 ETRI 컴퓨터소프트웨어기술연구소 전자상거래연구부 선임연구원
1999년~현재 동의대학교 경영정보학과 조교수
관심분야: 전자상거래, SCM