컴포넌트 기반개발에 의한 망관리시스템 에이전트의 인터페이스 스펙 모델링

박 수 현*°, 이 광 형**, 백 두 권***

* 동의대학교 컴퓨터공학과
** 동서대학교 컴퓨터공학과
*** 고려대학교 컴퓨터학과 소프트웨어시스템 연구실

Interface Specification Modeling of Network Management System Agent by the Component Based Development

Soo-Hyun Park*, Kwang-Hyung Lee**, Doo-Kwon Baik***

* Department of Computer Engineering, Dongeui University
** Department of Computer Engineering, Dongseo University
*** Software System Lab., Dept. of Computer Science & Engineering, Korea University

Abstract

Farmer 모델에 의하여 정의된 망관리 시스템 에이전트의 구성모델을 인터페이스 명세모델로의 매핑에 대하여 서술하였다. 인터페리스 명세모델은 컴포넌트 기반 개발 (Component Based Development) 에서 구현과 설계를 구분하기위하여 제시하는 모델로서 본 논문에서는 TMN 에이전트 설계의 예로서 설명하였다. 특히 Farmer 모델에서의 측면의 개념을 반영하기 위하여 측면 인터페이스(Aspect Interface)의 개념을 도입하였다.

I. 서론

Farmer 모델[1][2]은 실제세계의 사물을 다중측면에 의하여 분석한후 개체노드, 측면개체노드, 관일성 개체 및 다중화 추상화라는 여러 개념을 이용하여 디자인할 수 이론적 모델이다. Farmer 모델에 의하여 디자인된 시스템은 ADL(Aspect_Object Definition Language)의 형태로 저장된다. ADL내의 ILB / OLB 는 동적/정적로딩 컴포넌트로서 ADL의 구현을 위하여는 CBD 개념을 이용하는 것이 바람직하다고 생각된다. 하지만 기존의 CBD로는 Farmer 모델에서 제시하는 다양한 개념(예를 들어 측면개체, 다중성)을 제대로 반영할 수 없다. 본 논문에서는 이와 같은 여러 문제점에 대한 해결방안을 제시하고 Farmer 모델에 의하여 디자인된 망관리 시스템 에이전트의 구성모델을 컴포넌트 기반 개발 방법론(Component Based Development[3]) 에서 제시하는 컴포넌트-인터페이스 메타모델로의 매핑에 대하여 서술하였다.

II. 컴포넌트 기반 개발(CBD : Component Based Development)

2.1 컴포넌트의 개념

컴포넌트는 단일기능(atomic function)을 수행하는 소프트웨어의 단위로서 인터페이스를 통하여서만 외부에 모습을 드러내며 그 기능의 구현은 사용자에게 철저히 베일에 가려지는 특징이 있다. 이렇게 컴포넌트의 스펙(specification)과 구현(implementation)의 분리는 객체지향 패러다임에서의 거의 완벽한 캡슐화(encapsulation)를 제공함으로서 스펙은 컴포넌트의 제공자와 사용자 사이에서 울타리의 역할을 함으로서 사용자는 단지 컴포넌트의 스펙만을 보고 판단하여 컴포넌트를 사용하게 된다. 컴포넌트의 구현에 대하여는 관심을 둘 필요가 없게된다. 컴포넌트의 정의는 다음과 같이 내릴 수 있다.

" *컴포넌트란 플랫폼에 독립적으로 생성과 유지보수를 보다 용이하게 수행할 수 있는 일종의 작은 소프트웨어 조각으로서 상호간의 표준화 된 인터페이스를 갖는다.*"[4][5][6]

컴포넌트는 컴퓨터 언어에 중립적인 클라이언트/서버 상호작용 모델에 기반하여 상호동작(interoperation)을 수행한다. 이제까지의 객체와는 달리 컴포넌트는 컴퓨터언어 패키지 및 운영체제와 무관하게 네트워크 내에서 상호작용을 수행할 수 있다. 그러나 컴포넌트는 상속성, 폴리모피즘 그리고 인캡슐레이션을 지원한다는 점에서 객체와 유사한 특성을 가지고 있다. Ivar Jacobson은 컴포넌트를 블랙박스의 속성을 갖는 객체로 보았다. 블랙박스 컴포넌트의 경우는 상속성을 통한 컴포넌트의 확장이 불가능하다. OLE 컴포넌트는 바로 이러한 블랙박스의 범주에 속하나 CORBA 및 OpenDoc에서 사용하는 컴포넌트들은 상속성 개념을 지원한다. 화이트박스 컴포넌트는 고전적인 개념의 객체와 같이 행동하는 컴포넌트를 의미한다. 이와 같이 여러 정의가 존재하는 관계로 여기서는 컴포넌트가 지원해야하는 가장 최소의 기능 및 특성에 대하여 서술하기로 한다.

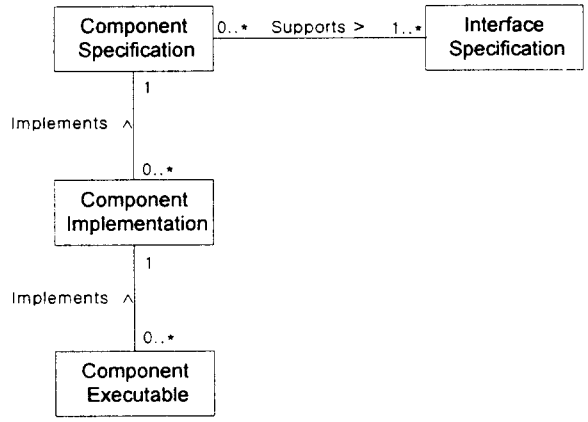

그림 2.1 컴포넌트와 인터페이스사이의 메타모델

■ *개방된 컴포넌트(Component)는 특정기능을 수행하는 소프트웨어의 단위(unit)를 의미한다. 이러한 소프트웨어 시장에서 구매가 가능해야 한다.*

컴포넌트는 다른 컴포넌트에 종속적이지 않아야 하며 개방된 시장에서 마치 상품과 같이 구매가 가능해야 한다.

■ *완벽한 응용 프로그램이 아니다.*

컴포넌트는 다른 컴포넌트와 연동하여 완벽한 응용프로그램을 형성한다. 따라서 컴포넌트는 응용프로그램 영역내의 한정된 역할을 수행한다. 컴포넌트는 다음과 같은 3가지 종류로 구분할 수 있다.

- Fine-grained object : 간단한 C++ 클래스 등
- Medium-grained object : GUI 컨트롤 등
- Coarse-grained object : applet 등

■ *어떠한 조합으로도 사용할 수 있어야 한다.*

컴포넌트는 P&P(Plug & Play) 개념에 의하여 다른 어떠한 컴포넌트와 조합을 이루어 동일한 family를 이루어야 한다.(이를 suite라 함).

■ *컴포넌트는 잘 서술된(well-specified) 인터페이스를 가져야 한다.*

고전적 개념의 객체와 같이 컴포넌트는 자신이 갖는 인터페이스를 통하여 통제를 할 수 있다. CORBA/OpenDoc의 IDL(Interface Definition Language)이 이에 해당한다.

■ *컴포넌트는 상호호환가능한 객체이다.*

컴포넌트는 주소공간, 네트워크, 컴퓨터 언어, 운영체제, 그리고 툴의 한계를 뛰어넘어 상호호환하여 사용할 수 있는 시스템 독립적인 개체로 볼 수 있다.

Farmer 모델에서는 컴포넌트웨어 형태로 변형된 컴포넌트요소들에 대하여 이미 다음과 같은 정의를 내린바 있다.[2]

【 정의 】 컴포넌트 요소

컴포넌트요소의 집합 C는 다음과 같은 구조(structure)로 정의된다.

$\forall c \in C,\ c = <$ Cid, G, SZ, L_AT $>$

where, Cid : 컴포넌트의 이름
G : 컴포넌트가 속해있는 PICR내의 그룹이름
SZ : 컴포넌트의 크기(화일 크기 (단위 : byte))
L_AT : 컴포넌트의 로딩과 관련된 속성집합
L_AT = { DYNAMIC, STATIC, DYNAMIC-STATIC } ■

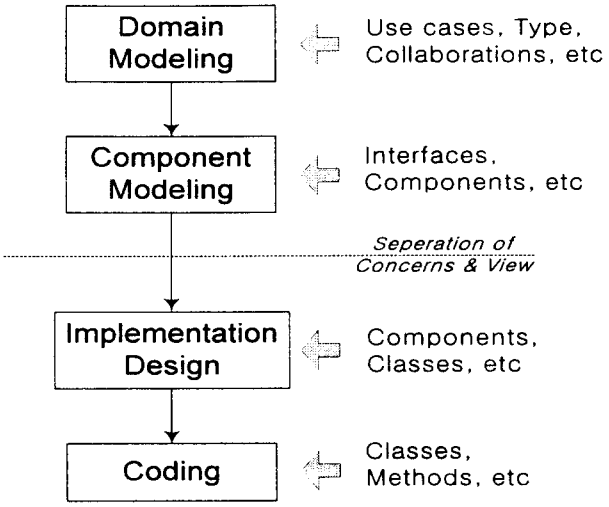

그림 2.2 프로세스관점에서의 도메인 모델링과 컴포넌트 모델링의 관계

2.2 컴포넌트 기반 개발(CBD : Component Based Development)의 개념

CBD[3]는 "미리 제작되어있는 소프트웨어 컴포넌트를 조립한다라는 개념에 기반한 소프트웨어 개발공정의 산업화과정"라고 정의할 수 있으며 다음과 같은 두 가지 기본개념에 기본을 두고 있다. 첫 번째로 응용프로그램을 개발시에 미리제작된 소프트웨어 컴포넌트들을 순서에 맞추어 조립함으로서 개발속도를 크게 향상시킬 수 있다. 두 번째로 상호호완가능한 소프트웨어 컴포넌트가 급속히 증가함에 이미 개발된 소프트웨어 컴포넌트들을 카탈로그로 만들어 놓음으로서 응용프로그램의 개발자로 하여금 쉽게 접근할 수 있도록 한다. CBD는 기제작되어 사전테스트를 거친 컴포넌트를 사용함으로서 개발단가를 낮출 수 있고 부가가치를 손쉽게 추가할 수 있고 차별화함으로서 최종고객의 요구사항에 보다 손쉽게 접근할 수 있는 장점이 있다.

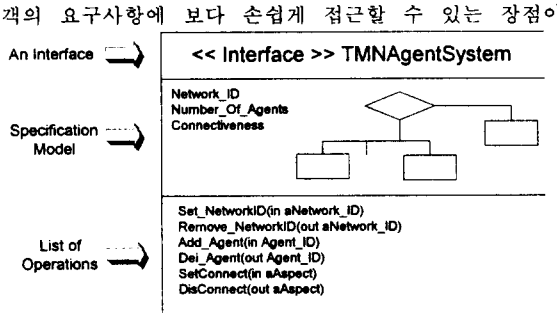

그림 2.3 인터페이스 개념

CBD는 시스템 통합 분야에서부터 새로운 개념의 통신 시스템인 지능망 분야에 이르기까지 이미 다양한 분야에서 활용되고 있다. CBD는 객체지향프로그래밍(OOP)의 개념에서 출발하였지만 OOP의 한계점인 코드재사용(code reusability)시의 상호호환성(interoperability)의 한계를 극복하였다 OOP 언어를 이용하여 생성된 클래스의 경우 실행코드(이진코드)차원에서는 클래스 상호간의 호환성을 제공하지 못할 뿐 아니라 및 런타임시에도 동적인 상호호환성을 제공하지 못하는 단점이 있다. CBD를 이용함으로서 얻을 수 있는 장점은 상호호환 가능한 코드재사용이라는 장점이외에도 클래스 카탈로그나 인터페이스 카탈로그를 이용함으로서 개발상의 복잡도를 감소시킬 수 있다. 또한 대형 프로젝트를 진행시 여러 팀이 동시에 병렬로 개발할 수 있기 때문에 생산성을 향상시킬 수 있을 뿐만 아니라 동일한 기

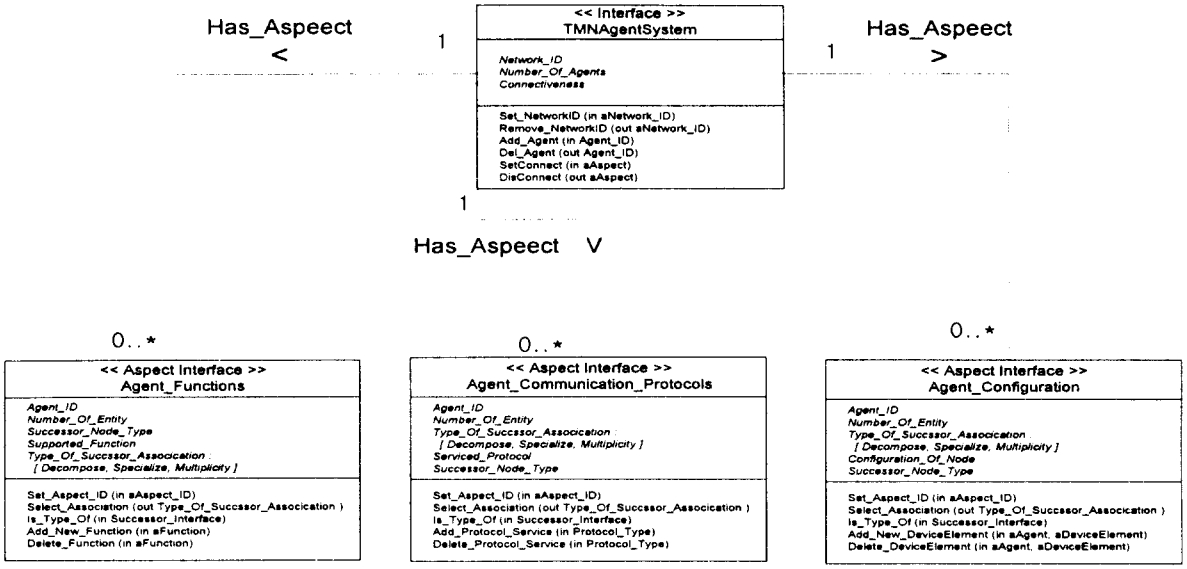

그림 2.5 Farmer 모델에 의한 TMN 에이전트 모델의 인터페이스 명세모델로의 매핑 (1)

능을 수행하는 여러 컴포넌트들 중에서 보다 저렴하고 효율적인 컴포넌트를 구매 또는 사용함으로서 코드선택의 신축성(flexibility)을 제공한다. 또한 이미 테스트를 거친 코드인 관계로 신뢰성을 확보할 수 있다는 장점이 있다.

2.2.1 CBD 활용 시나리오

Sterling Software Co.의 백서(White Paper)[3]에서는 CBD의 유용하고도 효율적인 활용을 위하여 다음과 같은 시나리오를 제안하고 있다.

■ 비즈니스 프로세스 모델의 재사용
이미 이전에서 사용하였던 비즈니스 프로젝트 모델의 어휘(vocabulary)와 프로세스(process, practice)를 유사한 기능을 수행하는 프로젝트에 적용

■ 컴포넌트 카탈로그에서 비즈니스 요구사항과 일치하는 컴포넌트를 선택
명세 카탈로그(specification catalog)에서 비즈니스 요구사항과 일치하는 소프트웨어 컴포넌트를 선택

■ 더욱 저렴한 컴포넌트로 대체
동일한 기능을 수행하는 컴포넌트를 여러 벤더들이 제공시 가격이 저렴한 컴포넌트를 사용

■ 기존의 시스템, 패키지, 사용자가 제작한 응용프로그램과의 인터페이스고려

■ 신속한 응용프로그램의 조립

■ 비즈니스 사용자의 신규 요구에 대한 신속한 처리 및 필요로 하는 부가 컴포넌트의 조립서비스 제공 (Business User Smart Assembly)
비즈니스 사용자가 지금까지의 루틴들에서 벗어난 비정규적인 데이터가 발생하여 이를 신속하게 처리할 필요가 발생할 경우 고객의 요구에 맞는 새로운 컴포넌트를 신속히 제공하여 처리해주는 기능

2.2.2 컴포넌트의 구성 및 컴포넌트/ 인터페이스 모델링

소프트웨어 컴포넌트는 다음과 같이 3 측면으로 볼 수 있다.

1) 명세(specification)

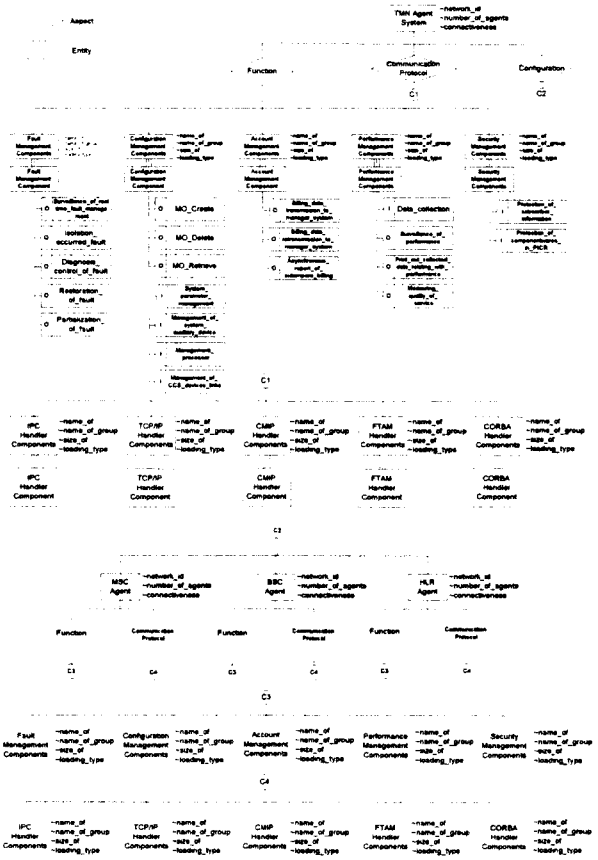

그림 2.4 Farmer 모델에 의한 TMN 에이전트 시스템의 디자인

시멘틱(semantics)에 대해 서술. 컴포넌트의 주요 기능 및 고객이 이를 사용할 수 있는 방법 등에 대하여 서술한다. 컴포넌

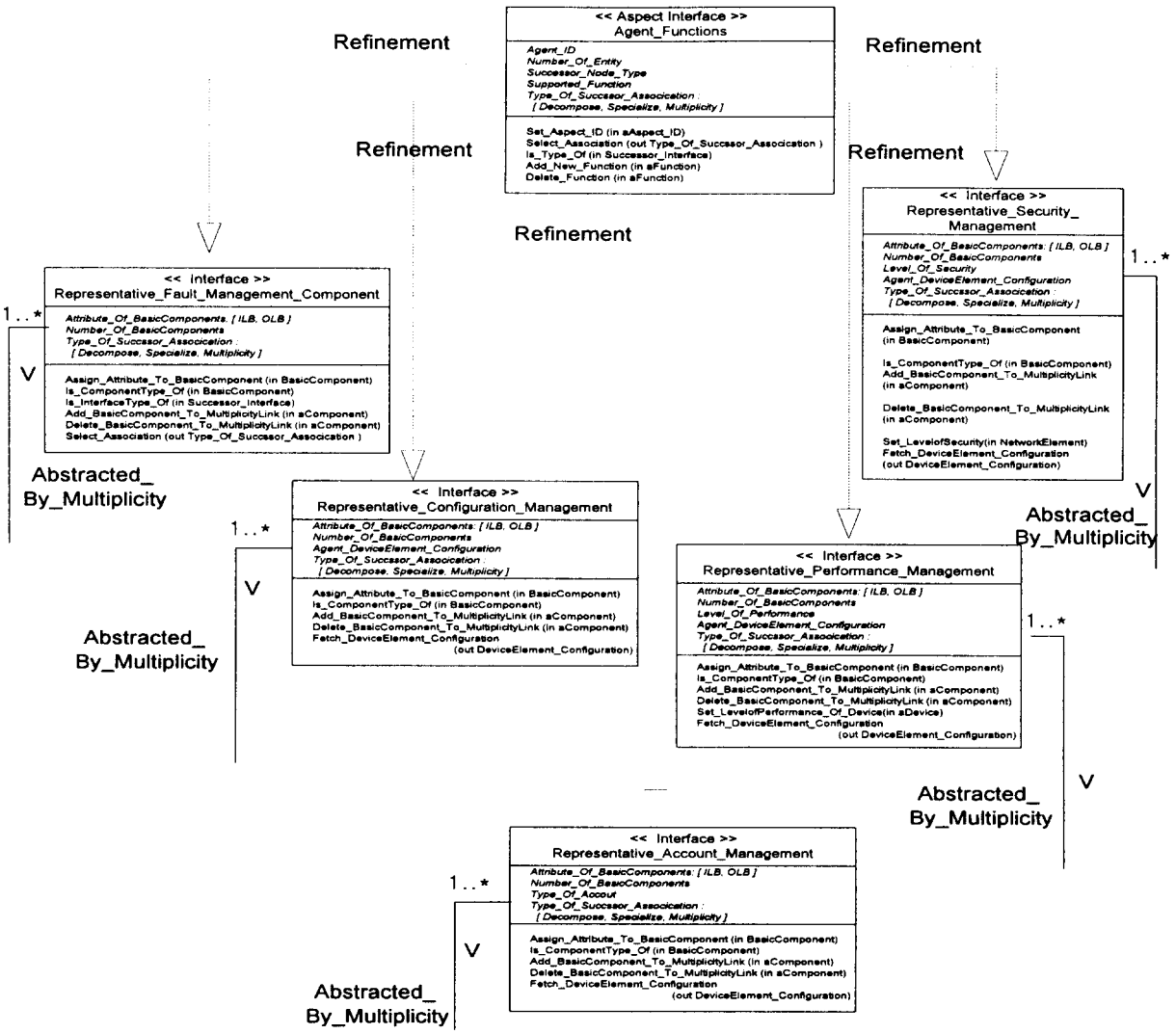

그림 2.6 Farmer 모델에 의한 TMN 에이전트 모델의 인터페이스 명세모델로의 매핑 (2)

트의 외부 뷰(external view)를 제공

2) 구현 디자인(implementation design)
명세에서 서술한 컴포넌트의 의도를 만족시키기 위하여 소프트웨어와 데이터저장의 디자인 및 구축에 대한 방안에 대하여 서술

3) 실행(executable)
컴포넌트 상호간에 공동으로 동작할 수 있는 플랫폼에 대하여 서술해준다. 예를 들어 철자검색 기능 컴포넌트(spell checking component)는 사전관리 컴포넌트(dictionary management component)와 상호동작 해야지만 제대로 역할을 수행할 수 있다. 이와같이 컴포넌트의 실행에 필요한 제반조건들에 대하여 서술해주는 역할을 담당한다.

명세(specification)는 컴포넌트의 최종사용자로 하여금 보다 손쉽게 빠르고 정확하게 해당 컴포넌트가 제공하는 서비스를 이해할 수 있도록 도와주는 역할을 담당한다. 명세는 자연어 형태나 형식모델(formalized model)로서 주로 서술된다. 명세는 다음과 같은 2 부분으로 구성된다.

1) 서비스(service)
컴포넌트에 의해 제공되는 동작의 리스트(list of operations)를 제공한다.

2) 명세타입 모델(specification type model)
동작(operation)의 행위(behavior)를 보여주는 어휘(vocabulary)를 제공한다. 어휘는 각 동작들이 행위를 서술한 문장내의 명사들을 나타내는 데 이들 명사는 객체로서 나타내어진다. 이러한 객체를 어휘라 한다.

명세는 인터페이스들로서 구성되는 데 인터페이스는 컴포넌트에 의해 제공되는 서비스(동작의 리스트)들의 클러스터들이라 말할 수 있다.

컴포넌트는 하나이상의 인터페이스로서 구현될 수 있으며 동일한 인터페이스가 다른 여러 컴포넌트에 의하여 지원될 수 있다. 그림 2.1은 컴포넌트와 인터페이스사이의 관계를 보여주는 메타모델이다. 그림 2.2는 도메인 모델링과 컴포넌트 모델링의 관계를 프로세스의 일부분으로 보여주고 있다. 그림 2.3은 인터페이스의 개념 및 구성에 대한 예를 보여주고 있다.

III. Farmer 모델의 기본개념

Farmer 모델[1][2]의 주요 목적은 실제 디자인 하고자 하는

에이전트를 분석하여 에이전트를 구성하고 있는 컴포넌트 요소들을 측면[7]에 따라 분리, 추출해 내는 데 있다. 이러한 결과 추출된 컴포넌트 요소들은 Farmer model tree의 leaf 노드에 위치하게 되며 이러한 컴포넌트 요소들은 network를 통하여 최종적으로 PICR에 저장이 된다. 또한 Farmer 모델은 플랫폼독립형 클래스저장소(PICR)[1][2]에서 컴포넌트 요소를 에이전트로 동적 또는 정적으로 다운로딩하는 Farming[1]의 개념을 추가한 형식모델이다. Farmer 모델은 실세계의 객관적, 추상적 대상체를 표현하는 개체노드 타입 (entity node type), 이러한 개체를 표현하는 관점인 측면노드 타입 (aspect node type), 개체와 측면간의 관계성을 나타내는 링크타입 (link type) 그리고 개체가 가지는 성질을 나타내는 속성타입 (attribute type), 동일한 이름을 가지는 2 노드는 동일한 속성과 동일한 서브트리를 갖는다는 균일성의 원리에 의해 정의되는 균일성 개체노드 타입 (uniformity entity node type)과 균일성 측면노드 타입 (uniformity aspect node type) 그리고 Farming 시 ILB 컴포넌트에 해당하는 속성을 지니는 IM-컴포넌트 타입 노드 (IM component type node : ILB Multiplicity Component type node), OLB 컴포넌트의 속성을 지니는 OM-컴포넌트 타입 노드(OM component type node: OLB Multiplicity component type node) 등의 구성요소들과 generalization, aggregation, multiplicity 등의 3가지 추상화 개념들로 구성된다. 그림 2.4는 TMN (Telecommunication Management Network) 에이전트 시스템을 Farmer 모델 다이어그램을 이용하여 디자인한 예이다.

IV. Farmer 모델에서 인터페이스 명세모델(Interface Specification Model)로의 매핑

실세계의 사물을 다중측면에 의하여 분석한 후 다중성 추상화와 같은 여러 추상화개념을 이용하여 모델링하는 Farmer 모델은 OLB/ILB와 같은 컴포넌트의 정적/동적로딩을 이론상으로 정립하였다. Farmer 모델에 의하여 설계된 시스템을 실제로 구현하기 위하여는 개체노드, 측면노드 및 ILB/OLB 등과 같은 객체들이 수행하는 기능에 대한 명세를 나타내는 인터페이스 명세 모델이 반드시 필요하게 된다. 본 절에서는 Farmer 모델에 의하여 디자인된 TMN 에이전트 시스템 설계 (그림 2.4)를 인터페이스 명세모델로 매핑하는 예를 보여주고 있다.

본 연구에서는 측면노드의 개념을 지원하기 위하여 Aspect Interface의 개념을 두었으며 Aspect Interface에 대하 상세정보를 보관하기 위하여 Aspect Interface 카탈로그를 별도로 유지하고 있다. 그림 2.5와 2.6은 이러한 매핑과정의 일부를 보여주고 있다.

V. 결론

ADL내의 ILB / OLB는 동적/정적로딩 컴포넌트의 개념을 이용하여 주요 시스템을 설계하는 데 사용되는 Farmer 모델의 개념을 지원하기 위하여 컴포넌트를 기반으로 시스템을 설계하고 구현하는 CBD의 개념을 도입하였다. 본 논문에서는 Farmer 모델의 다양한 개념을 반영하기 위하여 기존의 CBD 인터페이스 명세 모델을 그대로 사용하지 않고 측면 인터페이스 등의 개념을 도입함으로서 Farmer 모델이 갖는 특징을 반영하였다. 본 논문에서는 Farmer 모델에 의하여 디자인된 망관리 시스템 에이전트의 구성모델을 컴포넌트 기반 개발 방법론에서 제시하는 컴포넌트-인터페이스 메타모델로의 매핑에 대하여 서술하였다.

참고문헌

[1] Soo-Hyun Park, Doo-Kwon Baik, "Platform Independent TMN Agents Based on the Farming Methodology", The IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, The Institute of Electronics, Information and Communication Engineers (IEICE), pp.1152 - 1163, Japan, 1998

[2] Soo-Hyun Park, Doo-Kwon Baik, "Adaptive Modeling of Distributed Agents by the Farmer Model", *In Proceedings of The 6th IEEE International Workshop on Intelligent Signal Processing and Communication Systems (ISPACS'98), IEEE Computer Society,* Volume 2, pp.928 - 932, Melbourne, Australia, 1998

[3] Keith Short, "Component Based Development and Object Modeling", *White Paper,* Sterling Software, 1997

[4] Robert Orfali, Dan Harkey, and Jeri Edwards, *The Essential Distributed Objects, Survival Guide,* John Wiley & Sons, Canada, 1996.

[5] ComponentWare Consortium, "ComponentWare ArchI tecture : A technical product description", *I-Kinetics, Inc,* 1995.

[6] Robert Orfali and Dan Harkey, *Client/Server Programming with JAVA and CORBA,* John Wiley & Sons, Canada, 1996.

[7] Zeigler B. P, *Multifaceted Modeling and Discrete Event Simulation,* Academic Press, 1984.