유즈케이스 모델링을 위한 시나리오 근간의 목표(Goal)지향 분석 방안

(A Scenario-based Goal-oriented Approach for Use Case Modeling)

이 재 호 † 김 재 선 †† 박 수 용 †††
(Jea ho Lee) (Jaeseon Kim) (Soo yong Park)

요 약 소프트웨어 시스템이 대형화되고 복잡화해짐에 따라 사용자의 요구사항을 올바로 분석하고 서술하는 것이 중요시되고 있다. 이중 유스케이스 분석 방법은 요구사항 분석에서의 복잡도를 해결해 주는 장점 때문에, 객체지향의 분석 설계와 컴포넌트 기반의 개발에서 많이 이용되고 있다. 그러나 이러한 유스케이스 분석 방법은 흩어진 유스케이스들의 단순한 집합이어서 유스케이스들을 구조화하기 어렵고, 유스케이스들간의 영향분석을 하기 어려우며, 비기능적인 요구사항을 표현하기 어렵다. 이러한 문제를 해결하기 위해서, 본 논문에서는 목표지향의 분석 방법을 이용한 유스케이스 모델에의 적용방안을 제안하였다. 현재 연구되고 있는 목표지향의 분석은 요구사항으로부터 목표를 추출하기 어렵고, 분석 방법이 분석가의 경험적 근거에 의존적이다. 따라서 본 논문에서는 요구사항으로부터 목표를 직관적으로 식별하는 것이 어렵기 때문에 기초자료로 시나리오를 이용하여 그것으로부터 목표를 추출하는 시나리오 근간의 목표지향 분석 방법을 제안했다. 마지막으로 제안된 방안을 검증하기 위해 ITS의 시내버스정보 서브시스템에 적용하였다. 이 제안된 방안을 통해서 소프트웨어 분석가들은 유스케이스들간의 영향분석을 쉽게 하여 소프트웨어 개발 초기에 유스케이스들간의 불일치(inconsistency)를 찾을 수 있고, 비기능적인 요구사항을 표현할 수 있다.

키워드 : 요구공학, 유스케이스, 목표지향분석방법, 시나리오

Abstract As system become larger and more complex, it is important to correctly analyze and specify user's requirements. Use case modeling is widely used in Object-Oriented Analysis and Design(OOAD) and Component-Based Development(CBD). It is useful to mitigate the complexity of the requirements analysis. However, use cases are difficult to be structured, to explicitly represent non-functional requirements, and to analyze what is affected by changes of use cases. To alleviate these problems, we propose scenario-based goal-oriented approach for use case modeling. The approach is to apply goal-oriented analysis method to use case model. Since goal-oriented analysis method is not systematic and heuristics is considerably involved, we adopted scenarios as the basis for the goal extraction. The proposed method is applied to City Bus Information Subsystem(CBIS) in Intelligent Transport Systems(ITS) domain. The proposed approach helps software engineer to analyze what is affected by use case's changes and represent non-functional requirements.

Key words : Requirement Engineering, Use case, Goal-oriented Analysis, Scenario

1. 서 론

· 본 연구는 한국과학재단 특정기초연구 No.2000-1-303-001-3 지원에 의한 것임.

† 비 회 원 : 삼성전자 통신연구소 연구원
jhlee@samsung.com
†† 학생회원 : 서강대학교 컴퓨터학과
jskim@selab.sogang.ac.kr
††† 정 회 원 : 서강대학교 컴퓨터학과 교수
sypark@ccs.sogang.ac.kr

논문접수 : 2001년 7월 9일
심사완료 : 2002년 1월 16일

과거 25년 동안 소프트웨어의 요구사항은 소프트웨어를 개발하는데 있어서 중요한 현실적인 문제로 인식되어 왔다. Boehm은 소프트웨어 라이프사이클에서 후반부에 발생되는 요구사항의 오류를 수정하는 비용은 라이프사이클의 개발초기 단계인 요구공학 단계에서 오류를 검출하고 수정하는 비용의 200배가 소요된다고 언급했다[1].

미국의 350개의 회사를 대상으로 8000개 이상의 프로젝트에 대한 조사에서 프로젝트의 3분의 1은 완성되지

못했고, 2분의 1은 단지 부분적(부분적 기능의 완성, 비용의 초과, 시간의 지연)으로 완성되었다. 대부분의 개발 관리자들은 이러한 실패에 대한 주된 원인으로 빈약한 요구사항을 뽑았다. 그 내용을 간략히 살펴보면, 사용자와 의사소통 부족이 13%, 요구사항의 불완전성이 12%, 요구사항 변경이 11%, 비현실적인 기대가 6%, 명백하지 않은 목표가 5%로 나타났다[2]. 유럽의 17개 나라 3800개 이상의 조직에 대한 조사에서도 비슷한 결과를 이끌었는데, 인식된 소프트웨어 문제점 중 약 50%이하가 요구사항 명세에 있었고, 약 50%정도가 요구사항 관리에 있었다[3].

또한 최근의 소프트웨어의 동향을 보면 점차 사용되는 분야가 복잡해지고 대형화됨에 따라 소프트웨어에 대한 사용자의 요구사항의 분석과 올바른 서술이 점차 복잡하고 어려워지기 시작하였다. 때문에 사용자의 요구사항을 올바로 분석하고 명세화하는 요구 공학이 중요한 분야로 부각되고 있다. 이러한 요구공학의 주된 목표는 개발될 소프트웨어에 어떤 기능과 특징이 있으며 제약 요소에는 무엇이 있는지를 파악하고 명확하게 명세화하는 것이다.

요구공학의 과정에서 핵심 중의 하나는 시스템을 요청하는 사용자 관점에서 동의를 얻는 것이다. 이 중에서 유스케이스 중심의 분석은 사용자 중심으로 가시화되어 있어서 매력적인 접근으로 주목받고 있으며, 현재 객체지향 분석설계와 컴포넌트기반의 개발에서 요구사항을 추출, 분석, 명세화하는데 널리 사용되어지고 있다. 그러나 유스케이스는 요구사항을 구조화하기 어렵고, 요구사항간의 영향관계를 분석하기 어려우며, 요구사항에 대한 정당한 근거를 제시하기 어렵고 비기능적인 요구사항을 가시화가 어렵다. 또한 경험이 풍부하지 않는 분석가에 의해 분석되어지면 유스케이스의 추상화 단계 문제(유스케이스의 쪼개지는 정도에 관한 문제)를 발생시킬 수도 있다. 이유는 다음과 같다[20].

- 유스케이스는 독립적이지 않다. 그들간에 겹쳐질 수 있고 동시에 발생될 수도 있다. 또는 서로간에 영향을 미칠 수도 있다.
- 유스케이스는 어떤 조건이 만족되었을 때에 발생된다. 상황을 발생시키고 종결시키는 상황을 가지고 있다.
- 유스케이스의 추상화의 레벨과 기술되는 문장의 길이는 임의로 정해지게 된다.
- 유스케이스는 실제로 시스템을 사용하는 시나리오의 모든 가능한 부분들을 다룬다는 보장을 하지 못한다.

이를 보완하기 위한 대안으로써 본 논문에서는 요구사항을 구조화하기 용이하고, 비기능적 요구사항을 가시화할 수 있고, 요구사항들간의 영향분석을 할 수 있는 목표지향 분석을 이용한 유스케이스 모델에의 적용 방안을 제안한다. 그러나 현재 진행되고 있는 목표지향의 분석은 요구사항으로부터 목표(Goal)를 식별하는 명백한 공정을 제시하고 있지 않고 경험적 근거에 의존하는 것에 그치고 있다. 따라서 본 논문에서는 현재 목표지향의 분석을 보완하기 위해, 사용자로부터 요구사항을 추출하는데 용이한 시나리오를 기반으로 목표(Goal)를 식별하여 분석하는 공정을 제시하고, 요구사항으로 대응되는 목표들간의 영향관계를 분석하는 방안을 제시한다.

제시된 시나리오 기반의 목표지향 분석이 적용된 유스케이스 모델은 기능적, 비기능적 요구사항을 표현할 수 있으며, 요구사항을 구조화하여 요구사항간의 영향관계를 가시화할 수 있어 시스템 설계시에 발생할 수 있는 모듈간의 충돌을 사전에 예방할 수 있고, 컴포넌트 기반의 개발시 컴포넌트가 되는 근거가 될 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구에 대한 배경지식과 관련연구들에 대해 살펴보고, 3장에서는 시나리오 근간의 목표(Goal)지향 분석 방법을 제시하고, 4장에서는 시나리오 근간의 목표(Goal)지향 분석을 이용한 유스케이스(Use Case) 모델에의 적용을 제시한다. 5장에서는 3장, 4장에서 제시된 분석방안과 유스케이스(Use Case) 모델의 확장을 시내버스 정보 서브시스템(CBIS)에 적용하여 그 타당성을 검증해 보고, 6장에서는 결론을 맺고자 한다.

2. 관련연구

2.1 요구공학(Requirement Engineering)

1990년대에 들어오면서 요구사항에 대한 체계적이고 총괄적인 접근에 대한 필요성이 인식되어 요구사항의 분석과 서술 및 이들의 추출, 관리, 검증 등을 포함한 요구사항에 관한 모든 활동과 원칙을 요구공학(Requirements Engineering)이라 하고 소프트웨어 공학에 있어서 중요한 분야로 부각되기 시작하였다. 즉 요구공학은 소프트웨어 요구사항과 관련된 모든 활동들, 즉 요구사항 추출, 분석, 기술 및 관리 등에 대한 공학적 접근 방법이다[4].

어떤 기능과 특징을 갖는 소프트웨어를 개발할 것이며, 제약 요소는 무엇인지 등을 파악하고 분명히 하여 명세화하는 것은 소프트웨어 개발시 요구분석 단계에서 가장 큰 목표라 할 수 있다. 그러나 소프트웨어가 사용되는 분야가 점점 복잡해지고 대형화됨에 따라 이러한 소프트웨어의 개발에 있어서 사용자의 요구분석과 올바

른 서술이 점차 복잡하고 어려워지기 시작하였다.

예전에 조사된 바로는 문제가 발생된 국방에 관련되는 소프트웨어 중 약 60% 정도가 요구사항에 그 원인이 있다는 것이다.[6] 그 외에도 몇 가지 보고서들이 위의 사실을 확인시켜 주고 있다[6][7]. 이러한 문제들의 원인은 이해부족, 의사소통의 부족, 관점의 차이, 시간·예산의 부족이 있다. 이러한 문제점들을 해결하기 위해서는 좀더 체계적이고 공학적 접근이 필요한데 이를 위하여 최근 들어 요구공학 분야에서는 요구사항 분석 및 서술뿐 아니라 이들의 추출, 관리, 검증, 유지 등을 포함하여 요구사항에 관계되는 모든 활동과 원칙들에 관한 연구가 진행되고 있다.

2.2 유스케이스 중심의 분석(Use Case Driven Analysis)

유스케이스 중심의 분석 상에서 기본적인 개념은 액터(Actor)와 유스케이스(Use Case)로 정리될 수 있다. 액터는 시스템의 사용자에 의해 수행되는 특정한 역할을 말하는 것으로 시스템을 이용할 때 비슷한 행위를 하는 사용자들의 범주를 나타낸다. 여기에서 사용자는 실제로 사람들을 가리킬 수 있고 시스템과 통신하는 다른 장치나 외부 시스템을 가리킬 수도 있다. 유스케이스는 특정 액터가 시스템을 사용하는 시나리오의 특징이다. 분석 과정에 따라 모든 액터에 대해서 많은 수의 유스케이스를 발견하고 기술한다. 유스케이스는 시스템의 도메인에서 사용되어지는 언어들을 이용해서 자연어로 기술되어 진다. 이러한 액터와 유스케이스의 명세화를 통해서 유스케이스 모델(Use Case Model)을 얻게 된다.

유스케이스 중심의 분석은 요구사항 분석과정의 복잡성을 다루는 데에 도움을 준다. 액터의 사용적 측면에서 다른 유스케이스들에 대해 독립적으로 발견하고 분석함으로서 하나의 유스케이스에 초점을 둘 수 있기 때문이다. 유스케이스 중심 분석의 개념은 간단하고 유스케이스에 기술할 내용이 시스템의 문제 도메인에서 쉽게 발견되어 질 수 있기 때문에 고객과 시스템의 사용자가 의욕적으로 참여할 수 있다. 따라서 개발자들은 잠재적으로 존재할 수 있는 사용자들, 실제로 그들이 지닌 요구들과 행위들을 발견할 수 있다.

유스케이스 중심 분석의 가장 주된 단점은 구조화의 지원이 약하다는 점과 이들 유스케이스들 간의 영향관계를 알 수 없다는 것이고 유스케이스가 추출되는 근거가 제시되지 못하다는 것, 즉 "왜"에 대한 질문을 통해서 요구사항의 추출이 이루어지지 못한다는 것이다. 유스케이스 모델은 단지 유스케이스들의 흩어져있는 모임일 뿐이다.

유스케이스의 수는 복잡한 시스템의 경우 아주 클 수 있다. 독립적으로 만들어졌기 때문에 유스케이스의 기술들간의 불일치가 발생될 수 있다. Jacobson의 A Use Case Driven Approach[8]에서는 이와 같은 문제들을 해결하기 위한 지원책은 서술되어 있지 않고 있다.

특정 유스케이스는 모든 경우에 발생될 수 없다. 각 유스케이스는 어느 특정 상황에서 시작되고 성공적으로 완수되는 것을 기술하기 위한 것이다. 이러한 이슈는 유스케이스를 이용한 분석에서 다루어지지 않는다. 일반적으로 유스케이스를 이용한 분석에서는, Jacobson의 A Use Case Driven Approach[8]에서 정의된 바와 같이, 다음과 같은 이슈들에 대해서 충분히 다루어지지 않고 있다.

- 유스케이스는 독립적이지 않다. 그들 간에 겹쳐질 수 있고 동시에 발생될 수도 있다. 또는 서로간에 영향을 미칠 수도 있다.
- 유스케이스는 어떤 조건이 만족되었을 때에 발생된다. 상황을 발생시키고 종결시키는 상황을 가지고 있다.
- 유스케이스의 추상화(Abstraction)의 레벨과 기술되는 문장의 길이는 임의로 정해지게 된다.
- 유스케이스는 실제로 시스템을 사용하는 시나리오의 모든 가능한 부분들을 다룬다는 보장을 하지 못한다.

본 논문에서는 이러한 유스케이스를 이용한 분석에서 발생하는 문제점을 보완하기 위해 목표지향의 요구사항 분석을 이용해 유스케이스에 적용하는 방안을 제시하려고 한다.

2.3 목표(Goal)기반의 분석

지금까지 요구공학에서는 "무엇을", "어떻게"의 범주에서 주로 연구가 이루어졌다. 즉 요구사항을 데이터와 오퍼레이션을 중심으로 분석하였다. 이 때문에 그러한 데이터와 오퍼레이션이 왜 필요한지와 그것들이 요구공학 프로세스에서 자연적으로 발생하게 되는 상위단계의 요구사항을 성취하는 데에 충분한 지에 대한 검증이 어려웠다[9]. 이를 해결하는 방안으로 요구사항을 구조화하는 것을 지원하는 목표기반의 분석이 제안되고 있다.

Yue는 요구사항 모델을 명백한 목표의 표현으로 만드는 것이 요구사항의 완전성에 대한 기준을 제공한다고 말했다. 즉 요구사항이 목표로 잘 구조화되어 있다면 요구사항은 완전성을 가진다는 것이다[10]. 일반적으로, 목표는 환경과 시스템이 될 소프트웨어에서 에이전트의 상호협력을 통해서 성취되는 실재의 시스템에 대응된다.

요구사항 모델에서 목표의 통합과 목표 정제에 대한

두 개의 보완적인 프레임워크, Formal프레임워크[11]와 Qualitative프레임워크[12]가 존재한다. Formal 프레임워크에서는 목표 정제는 인공지능분야의 문제 축소에서 가져온 AND/OR 그래프 구조를 통해서 인식되어지고 AND연결, OR연결, Conflict연결, Operationalization연결이 존재한다. 이 프레임워크에는 KAOS(Knowledge Acquisition in autOmated Specification)방법론이 있다. Qualitative프레임워크에서 목표가 성취된다는 것은 아래 단계의 목표로 정제해 나아가는 것이거나 목표를 완전히 성취하는 것이 아니라 제한된 범위 내에서 성취하는 것을 의미한다. 이 프레임워크에는 NFR(Non-Functional Requirement)방법론이 있다. 이러한 목표기반의 프레임워크의 장점은 기능적인 요구사항뿐만 아니라 비기능적인 요구사항도 분석하기에 용이하다는 것이다.

Lamsweed는 Goal-directed requirement acquisition[13]에서 상위수준의 개념에 의해서 요구사항을 획득하는 방안을 제시하였다. 이 요구사항 획득 방안은 meta-model 그래프를 순회하면서 요구사항을 획득하는 특별한 방안이다. Goal-directed requirement acquisition[13]에서 목표를 중심으로 요구사항를 획득하는 절차를 제시하고 있지만, KAOS *meta-model*에 매우 의존적이어서 분석가 meta-model에 대한 이해가 없으면 목표를 중심으로 요구사항을 추출하기가 쉽지 않다.

2.5 시나리오 기반의 요구사항 분석

시스템을 개발하고자 할 때 시스템의 요구사항을 두고 사용자와 시스템 분석가간에는 원활한 의사소통이 무엇보다도 중요하다. 시나리오 기반의 요구사항 분석은 시스템의 사용자 요구사항을 사용자와 분석가 사이에서 사용자 입장에서는 이해하기 쉽고 분석가 입장에서는 보다 구체적인 단계의 요구사항 명세가 가능하도록 하기 위해서 이야기 예제 중심의 형식이 보강된 시나리오를 통하여 그 역할을 가능하게 한다. 시나리오를 통해서 사용자와 요구사항 분석가간의 효과적인 의사소통을 가능하게 함으로서 실세계의 경험적인 측면에서 요구사항 분석을 진행한다.

시나리오(Scenario)란 속성을 가지고 있는 에피소드의 선형적 시퀀스를 말한다. 즉 시스템을 명확하게 기능적 측면에서 그의 운용체계를 서술하기 위하여 각 입력과 출력의 시퀀스와 상태(state)를 보여준다. 즉, 구현하고자 하는 시스템의 요구사항을 스토리가 있는 형식을 빌어 서술할 수 있는 방법이다. 이것은 사용자로 하여금 시나리오를 통하여 자신이 필요로 하는 요구사항들을 추출하는 것을 가능하게 한다.

시나리오의 장점은 시스템의 초기 요구사항 분석시 시스템 관련자들 간의 원활한 의사소통이 주요한 것이다. 이를 보다 구체적으로 서술하면 첫째, 사용자의 관점에서 외부적인 시스템의 행위(behavior)를 직접적으로 서술할 수 있다. 둘째, 초기에 지속적인 사용자와의 상호작용을 요구사항 분석 동안에 지원한다. 셋째, 효율적인 비용의 프로토타입을 만들기 위한 지침을 제공한다. 마지막으로 요구사항 명세서의 확인을 도와준다.

시스템 요구사항으로부터 바로 목표(Goal)를 추출하는 것이 용이하지 않기 때문에, 본 논문에서는 이러한 시나리오 기반의 분석 기법의 시나리오를 목표(Goal)지향의 분석에서 목표(Goal)를 추출하는데 이용하게 되고, 각 목표(Goal)를 명세화 하는데 구성요소로 사용한다.

3. 시나리오 기반의 목표(Goal)지향 분석 방법

3.1 목표의 개념

요구공학에서 목표는 여러 응용분야에서 사용되고 있다. 목표가 사용되어지는 분야를 다음과 같이 요약할 수 있다.

- 목표는 요구사항의 추출과 정제를 위한 핵심적인 의미로 사용한다. 목표를 중심으로 요구사항을 분석할 때, 스테이크홀더에게 "왜", "어떻게", "그 밖에 어떻게"에 대한 질문을 사용하게 된다.
- 목표는 조직과 비즈니스 상황을 시스템과 연결하는데 사용된다.
- 목표는 자세하게 진행할 필요 없이 요구사항과 스테이크홀더의 목적을 명백히 하는 데 사용된다.
- 목표는 요구사항간의 충돌을 다루는데 사용된다.

이렇게 목표는 요구사항을 분석하는데 있어서 여러 측면에서 유용하게 사용되어진다. 이 목표는 어느 측면을 강조하는가와 목표를 주장하는 사람들에 따라 여러 가지 정의가 있다. 그 중에서 목표에 대한 몇 가지의 정의를 살펴보면 다음과 같다.

"Something that some stakeholder hopes to achieve in the future[14]."

"Nonoperational objective to be achieved by the composite system[13]."

"*High-level objective of the business, organization* or system[15]."

여기에서 "stakeholder"란 해당 프로젝트에 관여하는 모든 사람을 뜻하는 말이다.

본 논문에서는 목표를 다음과 같이 정의한다. 목표란 "소프트웨어 시스템에 의해 성취되어야할 사용자의 요구사항을 나타낸 실재"이다. 본 논문에서 제안한 목표의 정의에 기초하여, 목표를 추상화의 정도에 따라 3단계로

나누고, 바라보는 초점에 따라 3가지 측면으로 분류된다. 추상화라는 것은 핵심적인 면에 초점을 두기 위해서 자세한 부분을 감추는 구조를 말한다. 본 논문에서 추상화 단계를 나누어 놓은 것은 각 추상화 단계별로 목표를 기술하여 분석가로 하여금 문제영역으로부터 목표를 이끌어 내고 그 목표로부터 조작 가능한 목표로 정제해 가는데 유용하게 하기 위함이다. 따라서 문맥 목표로부터 시스템 상호작용 목표를 거쳐서 시스템 내부 목표가 되면 이 시스템 내부 목표는 조작 가능한 상태가 되어서 설계에 반영되게 되는 것이다. 아래의 그림 1은 목표의 추상화 단계를 표현한 것이다.

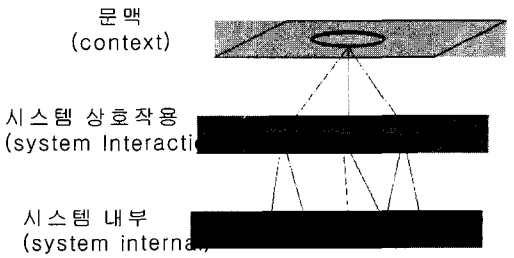

그림 1 목표의 추상화 단계

● 문맥 목표(context goal)

시스템이 궁극적으로 제공할 서비스를 식별한 것이 문맥 목표가 된다. 이 문맥 목표는 비즈니스 목표를 만족하게 하는 하나의 수단이 되는 것이다. 예를 들면, 비즈니스 목표가 "학사 행정 서비스를 향상시킨다"이면 이 비즈니스 목표를 만족시키기 위한 가능한 하나의 방법인 문맥 목표는 "웹을 통해서 수강등록 서비스를 제공한다"가 된다. 이 문맥 단계에서 가장 중요한 것은 시스템이 비즈니스 목표를 이루기 위해서 조직에게 도움이 되는 가능한 대안을 가시화 하는 것이다.

● 시스템 상호작용 목표(system interaction goal)

시스템과 사용자간의 상호작용에 초점을 두는 것으로 여기에서 상호작용 목표라는 것은 문맥 단계에서 식별된 문맥 목표(시스템이 제공할 서비스)를 성취하기 위해서 필요한 시스템과의 상호작용을 나타낸다. 따라서 시스템 상호작용 목표는 이전 단계인 문맥 단계에서 식별된 문맥 목표를 이루기 위해 시스템과의 상호작용을 추출하는 것이다. 예를 들면, 문맥 목표 "웹을 통해서 수강등록 서비스를 제공한다"에 대한 시스템 상호작용 목표 중의 하나는 "학생은 웹을 통해서 수강신청을 한다"가 된다.

● 시스템 내부 목표(system internal goal)

시스템 상호작용에서 식별된 시스템 상호작용 목표를 수행하는데 필요한 행위에 초점을 둔다. 이 단계에서 표현되는 시스템 내부 목표는 시스템 내부에서 수행되는 시스템의 행위형태로 표현되어진다. 시스템 내부 목표는 시스템 상호작용 목표의 기술에서 식별된 목표를 수행하기 위한 행위와 상태를 나타내게 된다. 예를 들면, 시스템 상호작용 목표 "학생은 웹을 통해서 수강신청을 한다"에 대한 시스템 내부 목표 중의 하나는 "학생은 로그인을 한다"가 된다.

본 논문에서 목표는 바라보는 초점에 따라 3가지 측면인 행위, 관점, 내용으로 분류되어진다. 이것은 목표를 추출함에 있어서 유용함을 제공하는 것으로 행위의 측면은 목표를 "성취해야하는 것(Achieve)"과 "유지해야 하는 것(Maintain)"으로 분류되어지고, 관점의 측면은 "액터 중심적인(Actor-specific goal)"과 "시스템 중심적인 것(System-specific goal)"으로 분류되어지며, 내용의 측면은 "기능적인"과 "비기능적인"으로 분류되어진다. 아래의 그림 2는 목표의 3가지 측면을 나타낸 것이다.

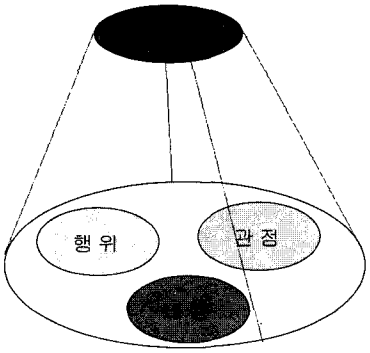

그림 2 목표 3가지 측면

● 행위 측면 분류 : Achieve, Maintain
● 관점 측면 분류 : Actor-specific, System-specifi
● 내용 측면 분류 : Functional, Nonfunctional

3.2 목표의 명세화

요구사항 명세는 분석된 요구사항을 명확하고, 정확하게 기록하여 문서화하는 것이다. 잘 정의된 요구사항 명세는 정확성(correct), 명확성(unambiguous), 완전성(complete), 입증가능성(verifiable), 일관성(consistent), 수정용이성(modifiable), 연계성(traceable) 등의 속성을

갖추어야 한다.

목표 분석을 통해서 추출된 각 목표는 내부 구성요소를 가지고 있다. 이 내부 구성요소는 요구사항을 목표로 분석한 내용을 올바로 명세화하기 위해 필요한 요소이고, 본 논문에서 제시한 목표의 개념을 기반으로 하고 있다. 이러한 목표에 대한 명세는 이후 단계에서 수행되어지는 프로젝트 계획, 설계, 코딩, 테스팅에 기초자료로 사용되어진다. 다음의 그림 3은 목표를 명세화하는 구성요소를 표현한 것이다.

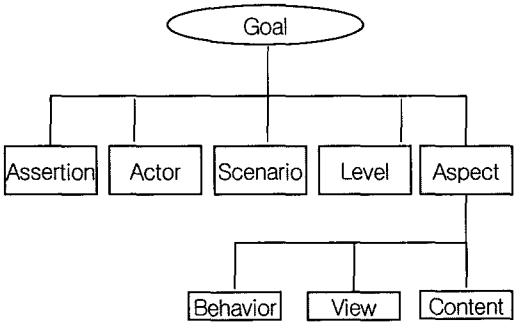

그림 3 목표의 내부 구성 요소

- 단언(Assertion) : 목표를 하나의 단문으로 표현한 것이다.
- 액터(Actor) : 목표와 연관된 액터를 식별하여 기술하는 것이다.
- 시나리오(Scenario) : 목표를 성취하는 측면에서 그것에 대한 운용체계를 서술하기 위해서 각 입력과 출력의 시퀀스와 상태를 보여 주는 것이다. 이 시나리오는 목표를 각 단계별로 분해하는데 이용되어진다.
- 단계(Level) : 목표가 어느 단계에 속해 있는가를 표현해 주는 것이다.
- 측면(Aspect) : 목표가 3가지 측면별로 어디에 분류되는가를 표현해 주는 것이다.

본 논문에서는 목표를 중심으로 분석된 요구사항을 명세화하는 형식을 정의하였다. 요구사항을 목표중심으로 분석한 전체구조는 목표 계층다이어그램으로 표현되어지며, 각 목표에 대해서는 본 논문에서 제시한 형식에 따라 명세화되어 진다. 명세에 포함되는 내용은 목표의 정의를 기반으로 하고 목표를 이루고 있는 내부구성요소를 중심으로 기술되어진다. 본 논문에서 제시한 목표 명세화 형식은 아래의 그림 4과 같다.

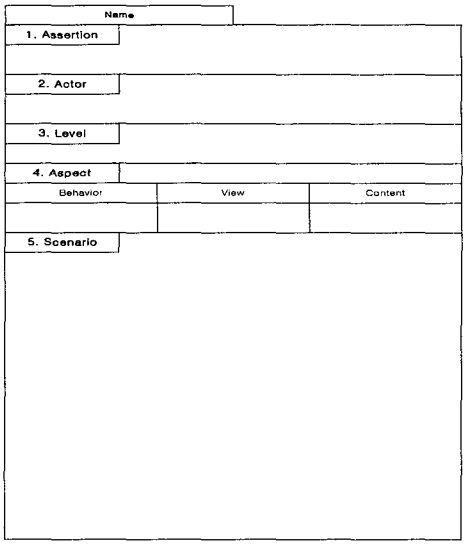

그림 4 목표 명세화 형식

3.3 목표의 관계

목표를 성취하는데 있어서 목표들간의 관계를 분석한다. 아래의 그림 5은 목표들간의 관계가 표현된 목표 계층 다이어그램이다.

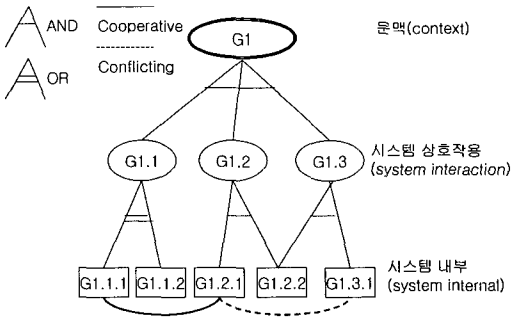

그림 5 목표들간의 관계

3.3.1 같은 부모노드를 가진 목표들간의 관계

하나의 목표에 대한 하위 목표들의 관계는 다음의 관계로 분석할 수 있다.

- AND : 상위 목표를 달성하기 위해서 하위 목표들이 모두 만족되어야 한다는 것을 의미한다. 즉 목표를 성취하는데 하위 목표들이 서로에게 요청되어지는 관계를 나타낸다.
- OR : 상상위 목표를 달성하기 위해서 하위 목표들 중의 하나 이상이 만족되어야 한다는 것을 의미한다. 즉 하나의 같은 목표를 성취하는데 있어서의 또 다

른 방법을 제공해 주는 관계를 나타낸다.

3.3.2 목표들간의 영향관계

목표를 성취하는데 있어서 목표들간의 영향관계를 분석하는 것이다. 하나의 목표를 성취하는데 있어서 다른 목표가 협력적인(cooperative) 영향을 미칠 수도 있으며 대립적인(conflict) 영향을 미칠 수도 있다. 이러한 요구사항간의 영향관계를 분석하여 시스템을 설계하는데 있어서 컴포넌트화하는 근거를 얻을 수 있고, 시스템 내부 기능간의 충돌을 개발초기에 인식하여 설계에 반영할 수 있다.

목표들의 관계를 설정하는 것은 두 단계로 이루어진다. 아래의 그림 6은 목표간의 관계설정 공정과 산출물을 표현한 것이다.

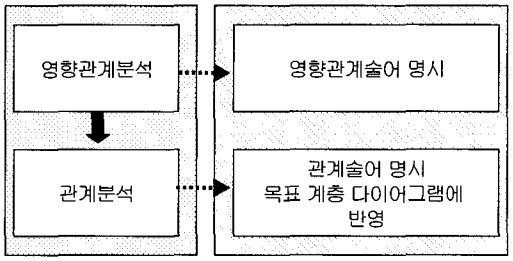

그림 6 목표들간의 관계설정 공정

하위 목표간의 영향분석을 하는데 있어서 그것이 소속된 상위 목표를 성취하는 측면에서 목표들 간의 영향을 분석할 수 있다. 이것을 술어명제로 표현하면 아래와 같다.

● $CP_{G1.1}(G1.1.1, G1.2.1)$
목표 G1.1을 성취하는 측면에서 목표G1.1.1과 목표 G1.2.1의 관계는 협력적인 이다.
● $CF_{G1.2}(G1.2.1, G1.3.1)$
목표 G1.2을 성취하는 측면에서 목표G1.2.1과 목표 G1.3.1의 관계는 대립적인 이다.

위에서 제시한 술어명제를 바탕으로 각각의 상위 목표를 성취하는데 있어서 하위 목표들이 어떠한 영향을 미치는가에 대한 분석한다. 이것에 대한 목표들간의 협력영향관계와 대립영향관계를 술어명제로 표현하면 아래와 같다.

서로 다른 상위 목표를 가진 하위 목표들에 대해서 영향을 분석할 때, 하위 목표는 소속된 각각의 상위 목표를 중심으로 분석을 하게 된다. 따라서 *CP(G1.1.1, G1.2.1)*는 상위 목표 G1.1을 성취하는데 하위 목표 G1.1.1와 G1.2.1이 협력적인 이거나, 상위 목표 G1.2을 성취하는데 하위 목표 G1.1.1와 G1.2.1이 협력적일 때, 협력영향관계가 성립하게 된다. *CF(G1.2.1, G1.3.1)*은 상위 목표 G1.2을 성취하는데 하위 목표 G1.2.1와 G1.3.1이 대립적이거나, 상위 목표 G1.3를 성취하는데 하위목표 G1.21와 G1.3.1이 대립적일 때, 대립영향관계가 성립된다.

● $CP(G1.1.1, G1.2.1)$
$=CP_{G1.1}(G1.1.1, G1.2.1)$
$\vee CP_{G1.2}(G1.1.1, G1.2.1)$
목표G1.1.1과 목표G1.2.1의 관계는 협력영향관계이다.
● $CF(G1.2.1, G1.3.1)$
$= CF_{G1.2}(G1.2.1, G1.3.1)$
$\vee CF_{G1.3}(G1.2.1, G1.3.1)$
목표G1.2.1과 목표G1.3.1의 관계는 대립영향관계이다.

목표 Gi와 목표 Gj간의 관계 *R(Gi, Gj)*는 위에서 제시한 영향관계*CP(Gi, Gj)*와 *CF(Gi,Gj)*의 쌍으로 정의되어지며, *<CP(Gi,Gj), CF(Gi,Gj)>*으로 표현된다. 여기에서 Gi와 Gj가 협력영향관계에 있으면 *CP(Gi,Gj)*가 *True*이고, Gi와 Gj가 대립영향관계에 있으면 *CF(Gi,Gj)*가 *True*이다. 목표 Gi와 목표 Gj간의 가능한 4가지의 관계는 아래와 같다.

● $R(Gi, Gj) = <FALSE, FALSE>$
목표 Gi와 목표 Gj는 관계가 없다.
● $R(Gi, Gj) = <True, FALSE>$
목표 Gi와 목표 Gj는 협력적인 관계이다.
● $R(Gi, Gj) = <FALSE, True>$
목표 Gi와 목표 Gj는 대립적인 관계이다.
● $R(Gi, Gj) = <True, True>$
목표 Gi와 목표 Gj는 균형적인 관계이다.

협력적인 관계에 있는 요구사항들은 시스템 설계시 같이 컴포넌트화 할 수 있는 근거를 제시할 수 있으며, 대립적인 관계에 있는 요구사항들은 요구사항들간의 불일치(inconsistency)를 나타내는 것으로, 이러한 불일치(inconsistency)를 초기에 발견하여 해결하거나 시스템 설계시에 반영하면 시스템 개발의 비용을 절감할 수 있다.

3.4 시나리오 근간의 목표지향 분석 공정

시나리오 근간의 목표지향 분석은 문제영역의 요구사항으로부터 목표를 중심으로 요구사항을 추상화하고 특징화하여 분류하고 분해함으로서 요구사항의 구조화를 얻을 수 있다. 목표지향의 분석방법의 공정과 산출물은 다음의 그림 7와 같다.

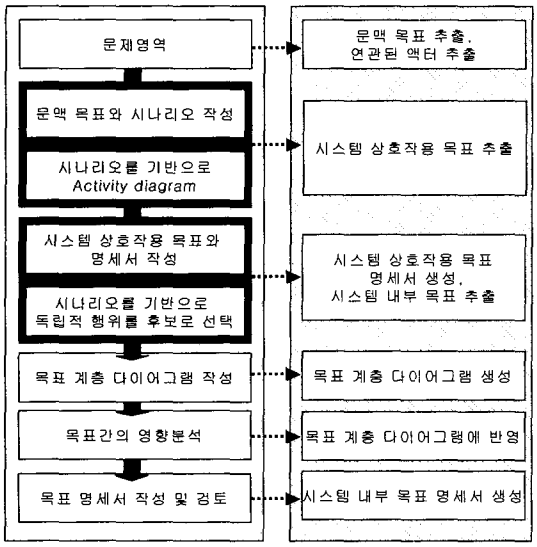

그림 7 목표지향의 분석 전체 공정

문제 영역과 사용자와의 인터뷰를 통해서 비즈니스 목표를 만족하는 문맥 목표를 추출한다. 이 문맥 목표는 비즈니스 목표를 만족하는 하나의 수단이다. 예를 들면, 비즈니스 목표가 "대중교통활성화 서비스"일 때, 이것을 만족시키는 하나의 수단인 문맥목표는 "시내버스 정보제공"이라는 것이 된다. 이것에 연관된 액터는 이용자, 시내버스운행관리 서브시스템(CBOMS), 권역교통정보 서브시스템(RTICS), GPS위성, 시스템이 된다.

다음 단계에서는 이 추출된 문맥 목표를 기준으로 요구사항에 적합한 시나리오를 작성하게 되고, 이 시나리오를 바탕으로 Activity diagram을 작성하게 된다. 이 Activity diagram은 요구사항에서의 실제로 존재하는 것(객체)들간의 상호작용과 작업의 흐름을 보여준다. 따라서 Activity diagram은 다음 단계의 시스템 상호작용 목표를 추출하는데 기초자료로 사용되어진다.

다음 단계에서는 추출된 각각의 시스템 상호작용 목표에 대한 명세를 작성하게 된다. 작성되는 명세서는 본 논문에서 제시한 목표 명세화 형식으로 작성되게 된다. 여기에 작성된 명세서 내용 중의 시나리오에서 독립적인 행위로 분류될 수 있는 것을 후보로 추출하여, 이 후보들 중에서 시스템 내부 목표를 추출하게된다.

이 각각의 시스템의 상호작용 목표에 대해서 시스템 상호작용 목표 명세서가 작성된다. 여기에서 작성된 명세서의 시나리오를 바탕으로 독립적인 행위로 분류될 수 있는 것을 후보로 하여 시스템 내부 목표를 추출하게 된다. 각각의 시스템 상호작용 목표에 대한 명세서의 시나리오는 시스템 내부 목표를 추출하는 기초자료가 된다. 각각의 시스템 상호작용 목표에 대한 명세서를 바탕으로 시스템 내부 목표가 작성되게 된다.

다음 단계는 이렇게 추출된 시스템 내부 목표를 목표 계층 다이어그램에 반영하여 목표 계층 다이어그램을 작성하게 된다. 이렇게 요구사항을 목표 지향으로 분석하여 목표를 추출하여 요구사항을 구조하여 가시화 할 수 있다. 또한 이렇게 구조화된 요구사항들을 표현한 목표 계층 다이어그램은 요구사항들간의 관계를 분석하는데 도움을 준다.

다음 단계는 목표 계층 다이어그램을 기반으로 하여 요구사항들간의 관계를 분석하는 것이다. 이 단계에서는 먼저 두 목표간의 영향관계를 설정한 후에, 두 목표간의 관계를 설정하게 된다. 분석된 관계는 목표 계층 다이어그램에 반영하게 된다.

위에서 제시한 관계분석 공정에 따라서 각각의 시스템 내부 목표들에 대해서 관계를 분석하여 설정하게 되고, 설정된 목표들간의 관계는 목표 계층 다이어그램에 반영되게 된다.

다음 단계는 추출된 시스템 내부 목표에 대한 시스템 내부 목표 명세서를 작성하게 된다. 시스템 내부 목표 명세서는 시스템 상호작용 목표 명세서와 마찬가지로 본 논문에서 제시한 목표 명세서 형식으로 작성하게 된다.

추출된 각각의 시스템 내부 목표에 대해서 시스템 내부 목표 명세서를 작성하게 된다. 시스템 내부 목표 명세서까지 완성이 되면, 시스템 분석가는 수집된 요구사항을 기반으로 목표 계층 다이어그램, 목표들간의 관계 설정, 시스템 상호작용 목표 명세서, 시스템 내부 목표 명세서를 모두 검토한다. 이것으로 시나리오 기반의 목표지향 분석이 완성되는 것이다.

4. 시나리오 근간의 목표(Goal)지향 분석을 이용한 유스케이스(Use Case)에의 적용 방안

4.1 목표(Goal)와 유스케이스(Use Case)의 관계

유스케이스를 목표 분석을 적용하기 위해서 목표와 유스케이스간의 대응되는 관계를 살펴보도록 하겠다.

액터는 하나 이상의 목표를 가지고 있다. 즉 액터는

소프트웨어 시스템을 통해서 목표로 하는 업무를 처리하게 된다. 목표지향의 요구사항 분석은 이러한 액터가 가지고 있는 목표를 중심으로 추출하여 분석한다. 반면에 유스케이스 분석은 액터가 가지는 목표를 성취하기 위해서 필요한 특정의 사용자 측면인 기능을 추출한다. 따라서, 액터를 중심으로 목표와 유스케이스를 연관시킬 수 있다. 그림 8은 목표와 유스케이스간의 관계를 표현한 것이다.

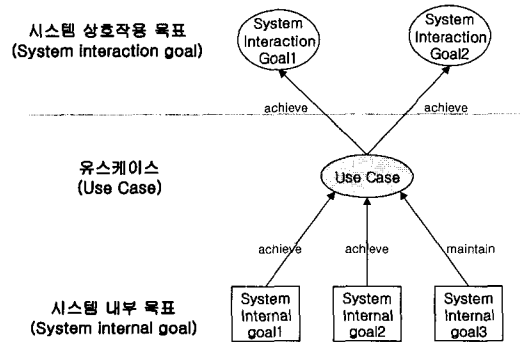

그림 8 목표와 유스케이스의 관계 다이어그램

목표지향의 분석을 통해서 산출된 목표 계층 다이어그램과 목표 명세서와 유스케이스 분석을 통해서 산출된 유스케이스 모델을 가지고, 각 유스케이스에 대해서 목표들간의 관계를 설정한 목표와 유스케이스 관계 다이어그램을 작성하게 된다.

4.1.1 시스템 상호작용 목표(System interaction goal)와 유스케이스(Use Case)

시스템 상호작용 목표는 문맥 목표(Context goal)로부터 액터를 중심으로 액터가 시스템과 상호작용을 통해서 성취하고자 하는 목표를 분석하여 추출한 것이다. 유스케이스는 액터가 시스템을 사용하는 측면에서 추출한 것이다. 따라서, 시스템 상호작용 목표를 성취하는데 사용되는 사용적 측면이 유스케이스가 되는 것이다. 유스케이스를 수행하는 것은 시스템 상호작용 목표를 성취하는 것이다. 시스템 상호작용 목표과 유스케이스를 연관시킴으로서 요구사항에서 유스케이스가 추출된 정당한 근거를 보여줄 수 있다.

4.1.2 유스성 명케이스(Use Case)와 시스템 내부 목표(Syste부 서m internal goal)

각각의 유스케이스 분석을 분석할 때 유스케이스는 수행할 시스템의 내부 행위단위로 표현된다. 시스템 내부 목표는 시스템 내부에서 수행되는 시스템의 행위형태로 표현된 것으로 식별된 시스템 상호작용 목표(System interaction goal)를 수행하기 위한 행위와 상태이다. 따라서, 시스템 내부 목표들을 성취(achieve)하거나 유지(maintain)함으로서 유스케이스를 수행하게된다. 유스케이스와 시스템 내부 목표를 연관시킴으로서 유스케이스에서 생략되었던 비기능적인 요소를 첨가할 수 있으며 구체적인 요구사항을 첨가할 수 있다.

4.2 시나리오 근간의 목표지향 분석을 이용한 유스케이스 모델에의 적용 공정

앞서 서론에서 언급되었던 유스케이스 모델에서의 단점들을 보완하기 위해서 목표지향의 분석을 이용한 유스케이스 모델에의 적용방안을 제시한다. 그림 9는 이러한 방법에 대한 공정과 산출을 표현한 것이다.

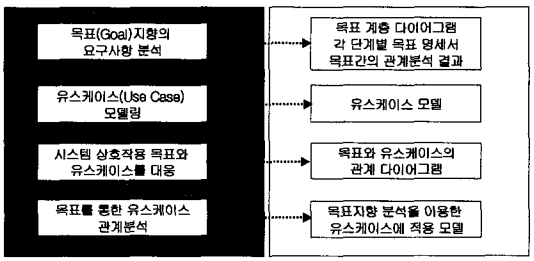

그림 9 목표지향의 분석을 이용한 유스케이스 모델에 적용 공정

시나리오 근간의 목표지향 분석을 이용한 유스케이스 모델에의 적용 공정은 처음에 시나리오 근간의 목표지향 요구사항 분석을 하게된다. 이 시나리오 근간의 목표지향 요구사항 분석은 본 논문 3장에서 제시한 방안으로 요구사항을 분석하게 된다. 이렇게 분석된 산출물은 목표 계층 다이어그램, 각 단계별 목표 명세서, 목표간의 관계분석 결과이다.

다음 단계는 유스케이스를 이용해서 요구사항을 분석하는 유스케이스 모델링을 하게된다. 분석한 결과로서의 산출물은 유스케이스 모델이다.

다음 단계는 시스템 상호작용 목표(System interaction goal)과 유스케이스를 대응시키는 것이다. 이 단계에서는 이전 단계에서 산출된 산출물인 목표 계층 다이어그램, 각 단계별 목표 명세서, 목표간의 관계분석, 유스케이스 모델이 사용되어진다. 시스템 상호작용 목표과 유스케이스는 액터를 중심으로 추출되어진다. 따라서, 본 논문 4.1에서 제시한 것과 같이 시스템 상호작용 목표와 유스케이스를 연관된 액터를 중심으로 대응시킬 수 있다. 그림 10은 시스템 상호작용 목표와 유스케이스

를 대응시키는 방안을 그림으로 표현한 것이다.

각각의 유스케이스를 시스템 상호작용 목표에 대응시키게 된다. 유스케이스는 대응되는 시스템 상호작용 목

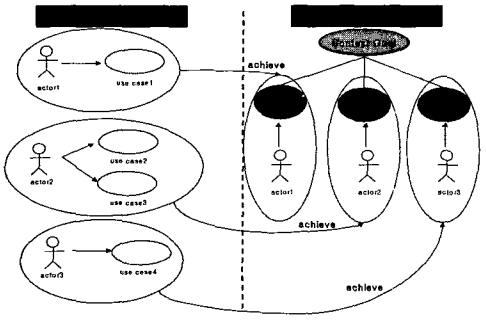

그림 10 시스템 상호작용 목표와 유스케이스의 대응관계

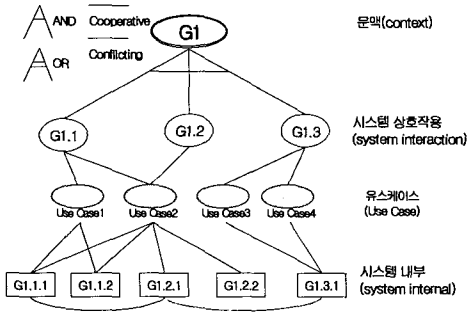

그림 11 시나리오 근간의 목표지향 분석을 이용한 유스케이스 적용 모델

표에 속하는 시스템 내부 목표와도 연관을 시킨다. 이렇게 해서 작성되는 것이 4.2절에서 제시된 목표와 유스케이스의 관계 다이어그램이다.

다음 단계는 목표를 통한 유스케이스 관계분석을 하는 것으로서 각 목표와 유스케이스 관계 다이어그램을 통해서 목표 분석을 이용한 유스케이스 적용 모델을 작성하게 된다. 그림 11은 시나리오 근간의 목표지향 분석을 이용한 유스케이스 적용 모델을 표현한 것이다.

그림 11에서의 목표지향 분석을 이용한 유스케이스 적용 모델로 Use Case1와 Use Case2는 협력관계에 있으며, Use Case2와 Use Case4는 대립적인 관계에 있다는 것을 가시적으로 알 수 있다. 또한 각 유스케이스에 대해서 요구사항의 정당한 근거가 되는 목표와 연관시켜서 구조화적으로 표현되며, 시스템 내부 목표에서 나타날 수 있는 비기능적인 요소가 첨가되어 가시화 된다. 즉, 그림 11에서 시스템 내부 목표들과 유스케이스와의 연결, 그리고 시스템 상호작용 목표들과 유스케이스와의 연결을 통해서 비기능적인 요소를 명시적으로 가시화하는 것이다.

본 논문에서 제안한 목표지향의 분석을 이용한 유스케이스의 적용 모델에서는 유스케이스에 대한 요구사항의 정당한 근거를 제공할 수 있도록 목표와 연관시켜서 구조화하였다. 또한 이 구조를 기반으로 목표들간의 관계를 분석하여 유스케이스들 간의 관계를 설정하였으며 유스케이스에서 가시화하기 어려운 비기능적인 요소를 시스템 내부 목표와 연관시킴으로서 가시화하여 표현하였다.

5. 시내버스정보 서브시스템(CBIS)에제 적용

본 장에서는 제안한 목표지향의 분석과 이 분석을 이용한 유스케이스 모델에의 적용 방안을 지능형 교통정보 시스템의 시내버스정보 서브시스템에 적용한다.

5.1 문제영역

본 논문에서 제시하는 방안을 검증하기 위해 현재 정부에서 추진하고 있는 지능형 교통 시스템(ITS) 도메인을 선택하였다. 지능형 교통 시스템은 교통체계의 효율성과 안정성을 제고하기 위하여 기존의 교통체계에 전자·정보·통신·제어 등의 기술을 접목시킨 차세대 교통체계이다. ITS는 크게 교통관리 최적화, 전자지불처리, 교통정보유통 활성화 여행자정보 고급화, 대중교통 활성화, 화물운송 효율화, 차량 및 도로의 첨단화의 7개 분야로 구성된다[16a]. 이 중 대중교통 활성화의 시내버스정보 서브시스템이 본 논문의 적용 도메인이다.

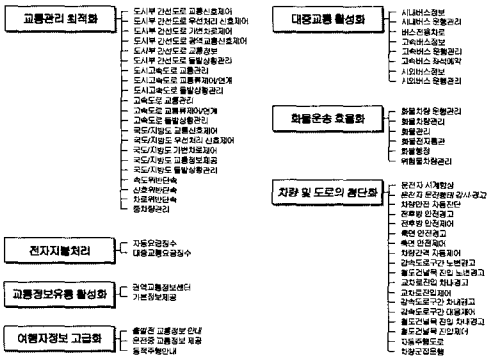

그림 12 ITS 도메인의 구조

시내버스정보 서브시스템은 안내전광판, 단말기, PC통신, 인터넷 등을 사용하는 시내버스 이용자에게 버스운행계획정보, 버스운행상태정보, 버스정적/동적 통행정보

의 서비스를 제공하는 시스템이다.

5.2 문제 도메인에 적용

5.2.1 시나리오 근간의 목표(Goal)지향 요구사항 분석

시나리오 근간의 목표지향 분석은 본 논문의 3장에서 제시한 공정에 의해서 분석하게 된다. 시나리오 근간의 목표지향 분석은 추출되는 요구사항에 대한 정당한 근거와 동기를 제공하면서 분석되어지고, 요구사항을 목표로 구조화하여 요구사항간의 영향분석을 할 수 있다. 이 단계에서는 목표 계층 다이어그램, 각 단계별 목표 명세서, 목표간의 관계분석 결과가 산출물로 나타나게 된다. 시내버스정보 서브시스템의 도메인을 분석하여 도출된 목표 계층 다이어그램은 아래의 그림 13과 같다.

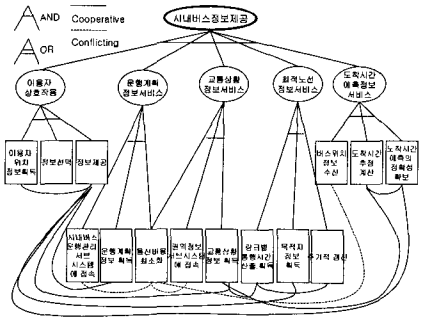

그림 13 문제영역에 적용한 목표 계층 다이어그램

각각의 추출된 목표에 대해서 목표 명세서를 작성하게 된다. 추출된 목표 중에서 시스템 상호작용 목표 "도착시간예측정보 서비스"에 대한 시스템 상호작용 목표 명세서를 살펴보면 아래의 그림 14과 같다.

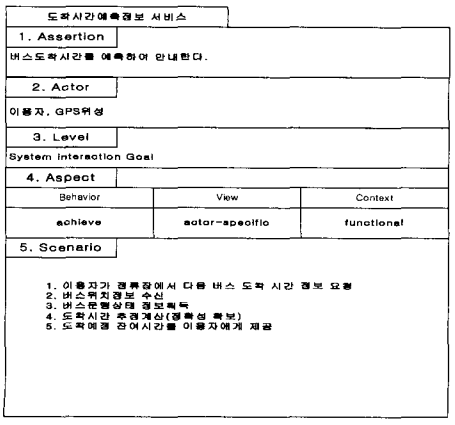

그림 14 "도착시간예측정보 서비스" 시스템 상호작용 목표 명세서

추출된 목표들간의 영향을 분석하기 위하여 목표들간의 관계를 설정하게 된다. 목표들간의 관계를 설정하기 위해서 추론되는 과정이 목표간의 관계분석 결과가 된다. 다음의 예는 관계 분석 중에서 목표 "버스위치정보수신",과 "통신비용최소화"에 대한 관계를 분석하는 것을 살펴본 것이다.

● *CP(G버스위치정보수신, G통신비용최소화)*
= $CP_{G운행계획정보서비스}$*(G버스위치정보수신, G통신비용최소화)*
∨ $CP_{G도착시간예측정보서비스}$*(G버스위치정보수신, G통신비용최소화)*
= *FALSE*

● *CF(G버스위치정보수신, G통신비용최소화)*
= $CF_{G운행계획정보서비스}$*(G버스위치정보수신, G통신비용최소화)*
∨ $CF_{G도착시간예측정보서비스}$*(G버스위치정보수신, G통신비용최소화)*
= *TRUE*

두 목표 "버스위치정보수신"과 "통신비용최소화"에 대한 영향관계는 "*CP(G버스위치정보수신, G통신비용최소화) = FALSE*"으로 협력영향관계에 있지 않고, "*CF(G버스위치정보수신, G통신비용최소화) = TRUE*"으로 대립영향관계에 있다. 이것을 기반으로 두 목표간의 관계는 아래와 같이 분석될 수 있다.

● *R(G버스위치정보수신, G통신비용최소화)*
= *<FALSE, TRUE>*
목표 G버스위치정보수신과 G통신비용최소화는 대립적인 관계이다.

따라서 두 목표 "버스위치정보수신"과 "통신비용최소화"의 관계는 대립적인 관계이다. 이렇게 분석된 관계는 목표 계층 다이어그램에 반영하게 된다. 위에서 두 목표간의 관계가 분석된 결과는 그림 23에서 보는 것과 같이 목표 계층 다이어그램에 반영된다.

5.2.2 유스케이스(Use Case) 분석

UML의 유스케이스를 이용하여 문제영역을 분석하여 유스케이스 모델을 작성한다. 유스케이스 중심의 분석은 문제영역의 사용적 측면을 중심으로 분석하게 된다. 아래의 [그림 15]은 시내버스정보 서브시스템에 대한 유스케이스 모델이다.

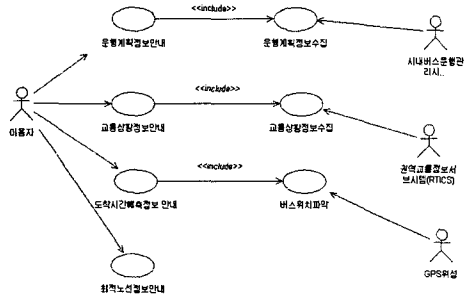

그림 15 시내버스정보 서브시스템 유스케이스 다이어그램

5.2.3 시스템 상호작용 목표(System interaction goal)과 유스케이스(Use Case)를 대응

시나리오 근간의 목표지향 분석의 결과로서 목표 계층 다이어그램과 유스케이스 다이어그램을 이용하여 시스템 상호작용 목표와 유스케이스를 연관되는 것끼리 대응시킨다. 시스템 상호작용 목표와 유스케이스의 대응은 연관된 액터를 중심으로 이루어진다. 시내버스정보 서브시스템에서 시스템 상호작용 목표와 유스케이스를 대응시킨 것은 아래의 그림 16과 같다.

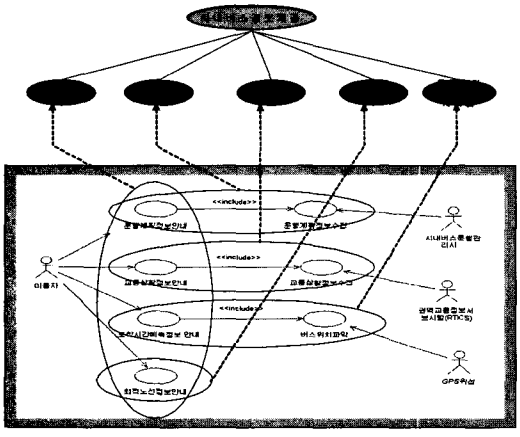

그림 16 시스템 상호작용 목표와 유스케이스 대응

시스템 상호작용 목표와 유스케이스가 대응된 후에는 목표와 유스케이스 관계 다이어그램을 작성하여 시스템 내부 목표와도 연관시킨다. 유스케이스 "운행계획정보안내", "운행계획정보수집"에 대한 목표와 유스케이스 관계 다이어그램은 그림 17과 같다.

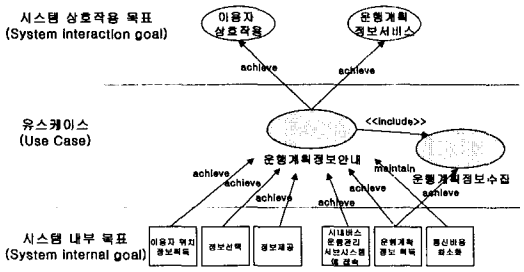

그림 17 목표와 유스케이스 관계 다이어그램

"운행계획정보" 유스케이스를 수행하는 것은 "이용자 상호작용", "운행계획정보서비스"의 시스템 상호작용 목표를 성취하는 것이다. 시스템 내부 목표 "사용자 위치 정보획득", "정보선택", "정보제공", "시내버스운행관리 서브시스템에 접속", "운행계획정보 획득"는 성취(achieve)함으로서 "운행계획정보" 유스케이스를 수행하게 되고, "통신비용 최소화"는 유지(maintain)함으로서 유스케이스를 수행하게 된다.

독립적인 유스케이스별로 목표와 연관시키고, 그것에 대한 목표와 유스케이스 관계 다이어그램이 작성된다.

5.2.4 목표(Goal)를 통한 유스케이스(Use Case) 관계 분석

독립적인 유스케이스별로 목표와 연관시키고, 그것에 대한 목표와 유스케이스 관계 다이어그램이 작성된다. 이것을 기반으로 시나리오 근간의 목표지향 분석을 이용한 유스케이스 적용 모델을 작성하게 된다. 그림 18은 문제 도메인을 적용한 시나리오 근간의 목표지향 분석을 이용한 유스케이스 적용 모델이다.

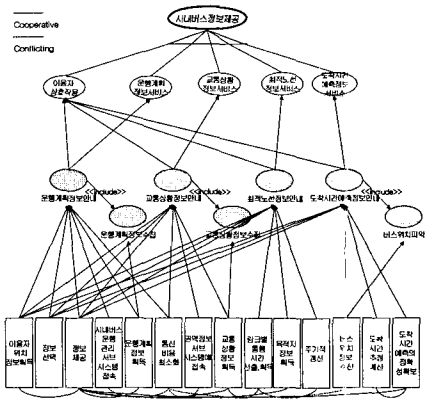

그림 18 시나리오 근간의 목표지향의 분석을 이용한 유스케이스 적용 모델

이 시나리오 근간의 목표지향 분석을 이용한 유스케이스 적용 모델을 통해서 유스케이스들 간의 영향관계를 알 수 있다. 아래의 표 1와 표 2는 유스케이스 적용 모델을 기반으로 시내버스정보 서브시스템의 유스케이스들간의 영향관계를 나타낸 것이다.

표 1 협력적인 관계를 가지는 유스케이스들

시스템 내부 목표 (System internal goal)	대응되는 유스케이스 (Use Case)
교통상황정보획득	교통상황정보안내
링크별통행시간 산출, 획득	최적노선정보안내

시스템 내부 목표 "교통상화정보획득"과 "링크별통행시간 산출, 획득"의 관계는 *R(G교통상화정보획득, G링크별통행시간 산출, 획득) = (TRUE, FALSE)*로 협력적인 관계에 있다. 따라서 대응되는 유스케이스 "교통상황정보안내"와 "최적노선정보안내"는 협력적인 영향관계를 가진다.

표 2 대립적인 관계를 가지는 유스케이스들

시스템 내부 목표 (System internal goal)	대응되는 유스케이스 (Use Case)
통신비용최소화	운행계획정보안내 교통상황정보안내
버스위치정보수신	도착시간예측정보안내

시스템 내부 목표 "통신비용최소화"와 "버스위치정보수신"의 관계는 *R(G통신비용최소화, G버스위치정보수신) = (FALSE, TRUE)*로 대립적인 관계에 있다. 따라서 대응되는 유스케이스 "운행계획정보안내"와 "도착시간예측정보안내"는 대립적인 영향관계를 가지고 또한 유스케이스 "교통상황정보안내"와 "도착시간예측정보안내"는 대립적인 영향관계를 가진다.

6. 결론

현재 객체지향 개발과 컴포넌트기반의 개발에서 요구사항을 분석하는데 유스케이스 중심의 분석을 많이 사용하고 있다. 이러한 유스케이스는 특정 사용자를 중심으로 사용적 측면인 기능을 위주로 분석하기 때문에 복잡도를 해소한다는 장점을 지니고 있다. 그러나, 소프트웨어 아키텍처를 구성할 때 중요한 요소가 되는 비기능적인 요구사항을 가시화할 수 없고, 요구사항들간에 구조화하는 것이 어려워 요구사항에 대한 근거를 제공하지 못한다. 그리고 요구사항간의 관계를 분석하기 어렵기 때문에 요구사항에서 발생할 수 있는 충돌을 찾기 어렵다. 이러한 문제점을 보완하기 위해서, 본 논문에서는 목표지향의 분석을 이용한 유스케이스에의 적용방안을 제안하였다. 그리고 여기서 이용되는 기존의 목표지향의 분석방법은 목표를 추출하는데 있어서, 분석가의 경험적 근거에 의해서 이루어져 일반 분석가들이 목표를 추출하는데 어려웠다. 이를 위해서, 본 논문에서는 목표를 추출하기 용이하게 하기 위해서 시나리오를 기반으로 해서 목표를 추출하는 시나리오 근간의 목표지향 분석 방법을 제안하였다.

시나리오 근간의 목표지향 분석 방법에서는 시나리오를 근간으로 목표를 추출하고 구조화하는데 사용되어지는 목표에 대한 개념을 목표에 대한 추상화 단계와 목표의 측면으로 설명하였고, 목표를 통해서 요구사항을 분석한 후에 목표를 명세화하는 방안과 추출되고 명세화된 목표들 사이에서 충돌을 찾기 위한 방안으로 목표들 간의 영향관계를 분석하는 방안을 설명하였다. 마지막으로, 이 시나리오 근간의 목표지향 분석 방법을 어떻게 수행하는지에 대한 전체 공정을 설명하였다. 이 시나리오 근간의 목표지향 분석 방법을 제안함으로써 일반 분석가들이 목표지향의 분석 방법을 이용하는데 있어서, 목표를 추출하고 구조하는데 용이성을 제공하여 줄 것이며, 시나리오 근간의 목표지향 분석 방법에 대한 지침을 제공할 것이다.

시나리오 근간의 목표지향 분석을 이용한 유스케이스 모델에의 적용에서는 목표와 유스케이스간의 대응관계와 유스케이스 모델에의 적용 공정에 대해 설명하였다. 이를 통해서 유스케이스에서 나타낼 수 없었던, 비기능적인 요구사항을 가시화할 수 있었고, 유스케이스를 구조화하여 유스케이스에 대한 정당한 근거를 나타낼 수 있었고, 유스케이스들간의 관계분석을 통해서 요구사항들간에 발생하는 협력영향관계와 대립영향관계를 찾을 수 있었다. 이 시나리오 근간의 목표지향 분석 방법을 유스케이스 모델에 적용을 제안함으로써 유스케이스들간의 영향관계 분석을 통해서 시스템 설계시에 컴포넌트화하는 근거를 제시할 수 있고, 시스템 내부 모듈간의 발생할 수 있는 충돌을 개발초기인 요구사항 단계에서 인식함으로서 설계에 반영하여 비용을 절감할 수 있을 것이다. 요구사항들간의 불일치(inconsistency)를 가능한 초기에 발견하는 것은 소프트웨어를 개발하는데 있어서 그것을 해결하는 방안에 대해 좀더 자유로움을 제공할 수 있다.

마지막으로 제안한 방안을 검증하기 위해서 지능형 교통시스템(ITS)의 시내버스정보 서브시스템(CBIS)에 적용하였다. CBIS에 적용하여, 유스케이스들간의 대립영향관계를 3개, 협력영향관계를 2개 찾았고, 유스케이스에서 표현할 수 없었던 비기능적인 요구사항 3개를 표현할 수 있었다.

향후 연구로 시나리오 근간의 목표지향 분석을 지원하는 도구의 프레임웍의 연구와 실제로 도구의 구현 개발이 필요하다. 이를 통해서 보다 효과적으로 본 논문에서 제시된 공정에 따른 이득을 최대화할 수 있을 거라 예상된다. 그리고 본 논문에서 제안한 시나리오 근간의 목표지향 분석 방법을 유스케이스 모델에 적용하여 유스케이스들 간에 발생하는 대립적인 영향관계, 즉 불일치(inconsistency)를 찾았다. 이 연구를 기반으로 유스케이스간의 대립적인 영향관계, 즉 불일치(inconsistency)가 발생했을 때, 요구사항단계에서 해결하는 방안을 연구할 필요성이 있다. 이것은 요구사항을 구현하는 소프트웨어를 성공적으로 개발하는데 필수적인 요소가 될 것이다.

참 고 문 헌

[1] B.WBoehm, "Software Engineering Economics", Prentice Hall, 1981.

[2] The Standish Group, "Software Choas", http://www.standishgroup.com/chaos.html.

[3] European Software Institute, "European User Survey Analysis", Report USV_EUR 2.1, ESPITI project, January 1996.

[4] 강기선, 김진태, 박병철, 박수용, "Requirement engineering: Overview," 소프트웨어공학회지, 1998

[5] US General Accounting Office, "Mission Critical Systems: Defense Attempting to Address Major Software Challenges", GAO/IMTEC-93-13, December 1992

[6] Richard H. Thayer, Merlin Dorfman, "Software Requirements Engineering", 2nd Edition. IEEE Computer Society Press. 1997

[7] Ian Sommerville, Pete Sawyer, "Requirements Engineering", Wiley, 1997

[8] Jacobson, I., et al. "Object-Oriented Software Engineering; A Use Case Driven Approach", Addison-Wesley, 1992

[9] G.F.Hice, W.S. Turner, L.F.Cashwell, "System Development Methodology." North Holland, 1974.

[10] K. Yue, "What Dose It Mean to Say that a Specification is Complete?", Proc. IWSSD-4, Fourth International Workshop on Software Specification and Design, Monterey, 1987.

[11] A. Dardenne, A. van Lamsweerde, "Formal Refinement Patterns for Goal-Driven Requirements Elaboration", Proc FSE'4-Fourth ACM SIGSOFT Symp. on the Foundations of Software Engineering, San Francisco, October 1996, pp.179-190

[12] J. Mylopoulos, "Information Modeling in the Time of the Revolution", Invited Review, Information System Vol. 23 No.3/4, 1998, pp.127-155

[13] Anne Dardenne, Axel van Lamsweerde, stephen fickas, "Goal-directed Requirement acquisition", science of Computer Programming, Vol. 20, 1993, pp.3-50.

[14] Colette Rolland, Carine Souyet, Camille Ben Achour Guiding, "Goal Modeling Using Scenarios", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 24, NO. 12, 1998, pp.1055-1071

[15] Annie-I.-Anton, "Goal-Based Requirements Analysis", Proceedings of ICRE 96, 1996, pp.136-144

[16] 건설교통부, "국가 ITS 아키텍처 확립을 위한 연구(II)", 국토연구원, 1999년 12월.

[17] Kenha Park, Jintae Kim, and Sooyong Park. Goal Based Agent-Oriented Software Modeling, Seventh Asia pacific Software Engineering Conference (APSEC 2000), pp. 320-324, December 2000.

[18] 김재선, 김종원, 홍태기, 박수용, "웹 기반 요구사항 관리도구의 구현", 소프트웨어공학회지, 제 13권 제 4호, pp. 16-29, 2000. 12.

[19] Taeki Hong, Jaeho Lee, Sooyong Park, "Implementation of Web-based Scenarios Management System Supporting Architecture Tradeoff Analysis Method", JWSE(Joint Workshop on Software Engineering) 2001, April 2001.

[20] Jacobson, I., et al. "Object-Oriented Software Engineering, A Use Case Driven Approach", Addison-Wesley, 1992

이 재 호

1997년 순천향대학교 공과대학 전산학과 학사. 2001년 서강대학교 공과대학 컴퓨터학과 소프트웨어 공학 석사. 2001년 ~ 현재 삼성전자 통신연구소 연구원. 관심분야는 요구공학, 소프트웨어 아키텍처, 컴포넌트 기반 개발(CBD), Embedded systems software

김 재 선

1999년 서강대학교 컴퓨터학과 공학사. 현재 서강대학교 컴퓨터학과 석사과정. 관심 분야는 소프트웨어 아키텍처, 에이전트 프레임웍, 웹서비스

박 수 용

1986년 서강대학교 전자계산학과 공학사. 1988년 후로리다 주립대 전산학 석사. 1995년 ~ 1998년 TRW Senior Software Engineer. 1998년 ~ 현재 서강대학교 컴퓨터학과 조교수. 관심분야는 요구공학, 에이전트 지향의 소프트웨어공학, 소프트웨어 아키텍처