원격 분산 환경에서의 소프트웨어 개발을 위한 통합 정보 객체 관리

한 관 희†

요 약

원격 분산 환경에서의 효과적인 소프트웨어 개발 관리를 위해서는 각종 정보 객체들을 체계적으로 분류하고 정보 객체들 간의 연관 관계를 구조적으로 정립하여 통합 관리하는 기능이 가장 기본이 된다고 할 수 있다. 본 논문에서는 통합 정보 객체 관리를 위해 첫째, 분산 소프트웨어 개발에 필요한 각종 정보 객체들의 구조 및 관계를 통합관리하는 틀로서 BOC(Bill Of Class)를 제안하며 BOC를 이용한 정보 객체 관리 시스템의 구조와 기능을 제시한다. 둘째, BOC 구조내의 부품을 구성하고 있는 데이터들을 표준화하여, 파트 사전을 구성하여 프로그램 작성시 표준 파트(데이터)만을 사용하게 함으로써 개발 생산성과 유지 보수 생산성을 향상시키는 방안을 제시한다. 셋째, 제시한 정보 객체 관리 구조를 활용하여 분산 환경에서의 효과적인 소프트웨어 개발을 지원하는 통합 정보 객체 관리 시스템을 설계 및 구현하고 이의 유용성을 보인다.

An Integrated Information Object Management for Distributed Software Development

Kwan-Hee Han†

ABSTRACT

For effective distributed software development, integrated information object management functions that manage the structures and relationships among information objects are most required. Presented in this paper is a managerial framework comprised of BOC (Bill Of Class) and part dictionary scheme for integrated information object management in distributed software development processes. Based on the proposed BOC and part dictionary scheme, an integrated information object management system is designed and implemented. As a result of this implementation work, the usefulness and benefit of proposed framework are also shown.

키워드 : 정보 객체 관리(Information Object Management), 분산 소프트웨어 개발(Distributed Software Development), BOC(Bill Of Class), 데이터 표준화(Data Standardization)

1. 서 론

소프트웨어 개발은 개발에 관계하는 구성원들이 하나의 작업 단위인 프로젝트를 구성하여 적합한 개발 프로세스를 수행하여 목표로 하는 소프트웨어를 생산해내는 일련의 과정이라고 정의할 수 있는데, 소프트웨어의 대형화 추세와 제품 출시 기간의 단축 요구 등과 같은 최근의 개발 환경 변화는 소프트웨어 개발 관리에 복잡성을 더해주고 있는 실정이다. 특히, 개발 팀원들이 지역적으로 서로 떨어진 장소에서 협동 작업에 의해 소프트웨어를 개발해야 할 경우, 효과적인 소프트웨어 개발 관리의 필요성은 매우 크다고 하겠다. 실제로 국내 소프트웨어 개발업체의 경우에도-특히 SI(System Integration) 회사의 경우-개발 팀의 일부는 국내나 국외의 고객사에 상주하여 개발 작업을 수행하고 일부는 본사에서 작업을 하면서 하나의 소프트웨어를 개발하는 사례가 증가하고 있다.

이와 같이, 최근 들어 분산 개발팀에 의한 소프트웨어 개발이 가속화되면서 기존의 CASE(Computer Aided Software Engineering) 도구에 의한 단위 개발 업무 자동화를 넘어선 통합 개발 관리 시스템에 대한 요구가 높아지고, 또 이에 대한 연구도 활발히 이루어지고 있다. 분산 환경에서의 통합 소프트웨어 개발 관리를 효과적으로 지원하기 위해 필요한 기능 요건으로는 1)원격지 개발팀원들 간의 협동 작업 지원 기능, 2)설계 문서, 소스 코드 등과 같은 개발 과정에서 산출되는 각종 정보 객체(Information Object)들의 저장 및 공유를 지원하는 정보 객체 관리 기능과, 3)개발에 필요한 절차 및 단위 업무를 정의하고 이를 자동 실행하는 프로세스 관리 기능 및 4)개발 일정계획, 자원 할당 및 상

† 정 회 원 : 경상대학교 산업정보공학과/공학연구원 교수
논문접수 : 2002년 1월 19일, 심사완료 : 2002년 4월 11일

태 모니터링 등과 같은 프로젝트 관리 기능 등을 들 수 있다[1, 2].

이 중에서 개발 과정에서 산출되는 각종 정보 객체들을 체계적으로 분류하고 정보 객체들 간의 연관 관계를 구조적으로 정립하여 통합 관리하는 기능은 위의 네 가지 기능 요건 중에서 가장 기본이 되는 기능이라 할 수 있다. 소프트웨어 개발 과정에서 관리 대상이 되는 정보 객체들로는 각종 문서와 클래스로 구현된 소스 코드 및 클래스내의 속성명/메서드명이나 데이터베이스 테이블명/컬럼명 등을 정의하는 데이터들로서, 이들 정보 객체들은 상호 여러 가지 형태로 복합적인 연관 관계를 맺고 있기 때문에 개발 단계가 진행될수록 정보 객체들 간의 상호 관계를 파악하기 어렵게되어 효율적인 소프트웨어 개발을 저해하는 요인이 된다. 그러므로, 이러한 복잡한 연관 관계들을 구조화하여 관련된 정보 객체들을 일관되게 관리할 수 있는 통합적인 관리 틀을 제공할 필요가 있다. 또, 분산 개발팀에 의한 소프트웨어 개발시에는 개발팀원간 협동 작업이 필수적인데, 이를 위해서는 정보 객체 자체의 컨텐츠(contents) 정보뿐만 아니라 컨텍스트(context) 정보에 대한 관리가 필요하며 컨텍스트 정보는 정보 객체간 구성 관계, 참조 관계, 종속 관계 등 주로 관계 정보에 의해 표현되므로[3], 통합 정보 객체 관리에서는 정보 객체들 사이의 연관 관계를 일관된 방법으로 구조화하는 것이 중요하다.

그리고, 특히 분산 개발팀에 의한 소프트웨어 개발시 소프트웨어를 구성하는 부품이라고 할 수 있는 데이터를 개발팀원들 사이에 제약 없이 사용함으로써 동일한 데이터 이름이 서로 다른 용도로 사용되거나, 동일한 엔티티(entity)를 서로 상이한 데이터 이름으로 정의하는 등의 사례가 빈번하여 개발 생산성이나 유지 보수 생산성에 악영향을 주고 있어서 데이터 공용화 및 표준화 방안이 마련되어야 한다.

위에서 언급된 분산 환경에서의 소프트웨어 개발을 위한 통합 정보 객체 관리에 대한 요구사항을 정리하면 다음과 같다. 첫째, 소프트웨어 개발에 필요한 각종 정보 객체들의 구조와 관계를 통합적으로 관리할 수 있는 틀을 제공하여야 한다. 둘째, 정보 객체 각각의 컨텐츠 관리 뿐 아니라 정보 객체간의 다양한 관계를 구조화된 방법으로 일관되게 관리할 수 있어야 한다. 셋째, 개발 생산성 및 유지 보수 생산성 향상을 위해 정보 객체 관리 대상의 범위는 관련 문서나 소프트웨어를 구성하는 패키지, 클래스 뿐 아니라 구성 부품인 데이터 등 개발에 필요한 정보 객체들을 모두 포괄하여 관리할 수 있어야 한다.

본 논문은 분산 개발팀에 의한 객체지향 소프트웨어 개발시의 효과적인 정보 객체 관리를 위한 통합 관리 틀을 제시하고 이를 시스템으로 구현하는 것을 그 범위로 하며, 논문의 목적은 다음과 같다. 첫째, 일반 제조업에서 제품의 구조를 표현하는 BOM(Bill Of Material)이 제품의 설계 및 생산 과정에서 관련 정보들을 통합하는 기본 틀로 사용되는 것[4]과 유사하게 소프트웨어 개발에 필요한 각종 정보 객체들의 구조 및 관계를 통합 관리하는 틀로서 BOC(Bill Of Class)를 제안하며, BOC를 이용한 정보 객체 관리 시스템의 구조와 기능을 제시한다. 즉, 소프트웨어 개발에 관련된 각종 정보 객체들을 BOC에 의해 구조 및 관계를 일관된 시각으로 제공함으로써 각 정보 객체들의 종속, 참조, 구성 및 파생 관계를 쉽게 파악할 수 있도록 한다. 둘째, BOC 구조내의 부품을 구성하고 있는 데이터들을 표준화하여 부품 사전을 구성하여 프로그램 작성시 표준 부품(데이터)만을 사용하게 함으로써 개발 생산성과 유지 보수 생산성을 향상시키는 방안을 제시한다. 셋째, 제시한 정보 객체 관리 구조를 활용하여 정보 객체들의 공유, 유통 및 저장을 촉진시켜 분산 환경에서의 효과적인 소프트웨어 개발을 지원하는 통합 정보 객체 관리 시스템을 설계 및 구현하고 이의 유용성을 보인다.

2. 관련 연구

분산 환경에서의 통합 소프트웨어 개발 관리를 위한 연구로서, 미국의 ARCADIA 프로젝트[5]의 목표는 대형 복합 소프트웨어의 분석, 개발, 유지 보수를 위한 소프트웨어 엔지니어링 환경을 구축하는 것으로 프로젝트 결과물 중의 하나인 Endeavors 시스템은 개발팀원들 사이의 프로세스 관리 및 조정 행위를 지원하기 위한 '개방형 분산 프로세스 실행 환경'으로 동적인 프로세스 변경과 사건 모니터링 및 외부 개발 도구와의 통합 기능 등을 그 특징으로 하고 있다. Ozweb[6]은 프로세스 정의 및 실행을 위해 규칙 기반의 고유한 프로세스 모델링 언어를 사용하고 있으며 전후진 연쇄 추론을 특징으로 하는 프로세스 엔진을 가지고 있다. Ozweb은 소프트웨어 개발 과정에서 생성되는 각종 산출물들의 저장소로서의 웹 기능을 제공하는 것에 초점을 맞추었다. MILOS 프로젝트[1]는 '분산 소프트웨어 개발을 위한 지원 시스템'을 목표로 하고 있으며 개발 프로세스 계획, 프로세스 실행 엔진, 프로세스 실행 조정, 변경 사항에 대한 통지 메카니즘 등을 제공하고 있으며 프로젝트 관리 기능의 강화를 특징으로 들 수 있고, 최근에는 개발팀원의 경험이나 지식을 개발 조직 내에서 효과적으로 활용하기 위한 지식 관리 시스템으로 확장하고 있다[7].

분산 환경에서의 정보 객체 관리는 소프트웨어 형상 관리 분야에서 활발히 연구되고 있으며[8], 최근에는 기존의 변경 관리 기능에 부가해서 소프트웨어 개발 프로세스 및 프로젝트 관리 기능을 통합하려는 연구들이 이루어지고 있다[9, 10]. [11]과 [12]에서는 하나의 정보객체에 대한 동시적 변경이 발생했을 때 이를 처리하는 방법에 대한 연구가 수행되었는데 단위 정보 객체의 변경과 그 처리 프로세스에

연구 대상을 제한하였다.

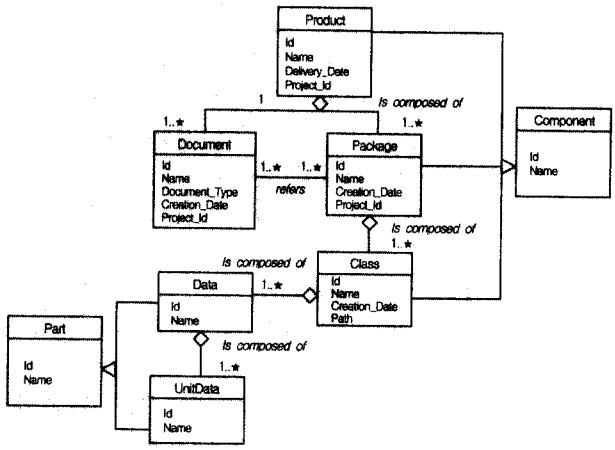
문서 관리를 중심으로 하는 공동 작업장 시스템인 BSCW (Basic Support for Cooperative Work)[13]에서는 정보 객체들의 상태 변화를 초래하는 사건 감지 및 팀원들 사이의 인지(awareness) 기능의 효과에 대한 연구가 이루어졌다. 즉, 협동 작업을 위해서는 공유 대상이 되는 문서에 대해 누구에 의해 언제 무엇(문서의 생성, 삭제, 수정, 이동 등)이 이루어졌는지를 팀원들에게 알리는 인지 기능의 중요성을 강조하였다. DHT(Distributed semantic HyperText)접근 방법[14]에서는 의미론적 하이퍼텍스트(semantic hypertext) 모델에 기반하여 정보 객체들의 관계를 모델링하고 이를 분산 환경에서의 가상 저장소로 구현하였으나 단위 클래스의 형상 관리에 중점을 두었다. [15]에서는 소프트웨어의 재사용성을 향상시키고 다양한 산출물을 효율적으로 관리하기 위해 소스 코드로부터 구문 분석에 의해 각 클래스의 클래스 부품 정보(클래스명, 멤버 함수, 속성)를 추출하고 클래스간 관계를 형성함으로써 효율적인 객체 관리 저장소를 설계하였으나 관리 대상을 클래스로만 한정하였다. [16]에서는 소프트웨어 부품 관리 측면에서 소프트웨어 제품 개발에 사용되는 부품인 데이터들을 표준화하여 여러 개발 프로젝트에서 표준화된 부품(데이터)만을 사용함으로써 개발 생산성 제고와 유지 보수를 용이하게 하려는 연구가 수행되었는데, 이 연구에서는 관리 대상을 소프트웨어를 구성하는 부품 단위인 데이터와 데이터를 구성하는 약어로만 국한하였다.

위에서 살펴본 바와 같이 기존 연구의 대부분은 분산 환경에서의 정보 객체 관리를 위해 필요한 기능 중 일부분만을 다루고 있어서 통합된 정보 객체 관리에 대한 전체적인 요구사항을 만족시키지 못하고 있는 실정이다.

3. 정보 객체 관리 프레임워크

3.1 BOC(Bill Of Class) 관리

하나의 객체지향 소프트웨어 제품은 하나 이상의 패키지(Package)와 분석, 설계 및 구현 단계에서 작성되는 하나 이상의 문서로 이루어져 있다. 여기서 패키지는 상호 관련성이 높은 클래스들의 묶음을 의미한다. 그리고, 하나의 패키지는 하나 이상의 클래스로 구성되어 있고, 하나의 클래스는 클래스명, 속성명, 메서드명 등 복수 개의 데이터로 구성된다. 여기서 하나의 데이터는 데이터를 구성하는 기본단위인 단위 데이터(Unit Data)의 조합으로 구성된다(예를 들면, 재고 수준을 나타내는 데이터인 InventoryLevel은 Inventory와 Level이라는 단위 데이터들의 조합으로 구성됨). 단위 데이터와 데이터는 소프트웨어를 구성하는 최하위 부품으로서 이는 부품(Part) 클래스로 추상화될 수 있고 식별자와 이름을 공통 속성으로 갖는다. 또, 클래스와 패키지 및 제품은 소프트웨어를 구성하는 중간품이나 최종품으로서 이는 조립품(Component) 클래스로 추상화될 수 있고 식별자와 이름을 공통 속성으로 갖는다. 위에서 설명한 소프트웨어 제품의 논리적 구조를 UML(Unified Modeling Language) 클래스 다이어그램으로 표시하면 (그림 1)과 같다. 본 연구에서는 소프트웨어 제품의 논리적 구조 중에서 클래스-문서/패키지-제품 관계를 'BOC(Bill Of Class)'라 칭하고 소프트웨어 개발시의 각종 산출물을 BOC의 관점에서 관리한다.



(그림 1) 소프트웨어 제품의 논리적 구조

각 정보 객체간의 관계는 하나의 제품 내에 어떤 패키지들이 소속되어 있고 또 하나의 패키지 내에 어떤 클래스가 소속되어 있는가를 표현하는 구성(structure) 관계, 특정한 클래스에 관한 설계 정보를 담고 있는 설계 문서는 어느 것인가를 표현하는 참조(reference) 관계, 클래스간의 슈퍼-서브 관계를 표현하는 종속(dependency) 관계 및 하나의 객체 내에서의 변경 이력을 표현하는 파생(derivation) 관계를 정의하고 관리하게 된다. 즉, 소프트웨어 개발 과정이 진행되면서 발생하는 산출물을 저장소에 저장하거나 변경할 때는 BOC 구조에 따라 등록/수정이 될 수 있도록 함으로써 단위 산출물과 산출물 사이의 관계를 일관성 있게 관리할 수 있다. 정보 객체간 관계는 크게 문서-클래스 관계, 패키지-문서 관계, 클래스-패키지 관계 및 슈퍼-서브클래스 관계로 분류된다. 문서-클래스 관계에서는 해당 클래스 정보가 어느 문서에 포함되어 있는지(클래스-문서)와 해당 문서는 어떤 클래스 정보를 포함하고 있는지(문서-패키지-클래스)를 표현한다. 패키지-문서 관계에서는 해당 패키지 정보가 어느 문서와 관련이 있는지를 나타내며(패키지-문서), 클래스-패키지 관계에서는 해당 클래스가 어느 패키지에 포함되어 있는지를 나타낸다(클래스-패키지-제품). 슈퍼-서브클래스 관계에서는 하나의 슈퍼클래스에 종속된 서브클래스들을 표시하며 동시에 각 서브클래스가 어느 패키지에 소속되어 있는지를 나타낸다(슈퍼클래스-서브클래스-

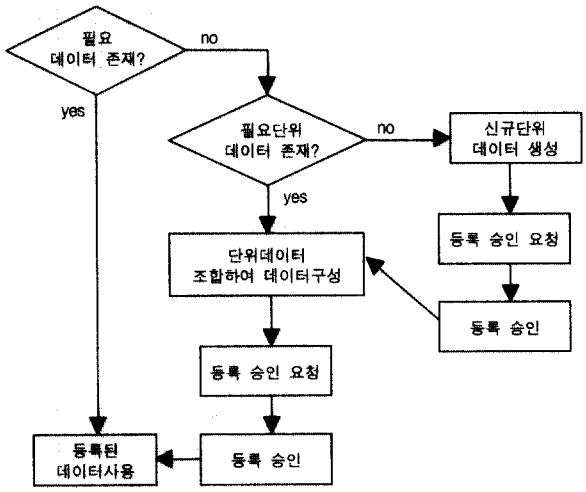
패키지). 그리고, 하나의 정보 객체 내에서의 변경 이력에 대한 파생 관계는 버전 트리(version tree)로 표현한다.

단위 정보 객체 중에서 문서는 그 성격에 따라 관리용과 기술용으로 구분하는데, 관리 문서는 프로젝트 착수 보고서, 진도 보고서, 일정 계획서, 예산 계획서 등과 같은 프로젝트 관리에 필요한 문서를 의미하며 기술 문서는 실제 제품 개발 과정에서 발생한 요구사항 분석서, 시스템 분석/설계 문서, 시험 문서 등을 의미한다. 주요 기술 문서는 프로젝트 착수시 프로젝트 스텝 등록 과정에서 각 스텝에서 산출되어야 하는 베이스라인(baseline) 문서로 정의함으로써 프로젝트의 진도 관리를 위한 마일스톤으로 사용된다. 개발 과정에서 소스 코드의 형태로 산출되는 클래스는 그 용도에 따라 크게 세 가지로 나누어 사용자 인터페이스 부분인 프리젠테이션(presentation) 부분과 응용 로직(application logic) 부분 및 데이터베이스 연결이나 보안관리 등과 같은 서비스(service) 부분으로 분류하여 관리한다.

3.2 파트사전(Part Dictionary) 관리

효율적인 정보 객체 관리를 위해 (그림 1)의 논리적인 소프트웨어 제품 구조를 두 부분으로 나누어 제품-패키지-클래스 관계는 BOC 기능에서 관리하고, 데이터-단위 데이터 관계는 파트 사전 기능에서 분리하여 관리한다. 파트 사전 관리에서는 소프트웨어 개발에 사용되는 각종 데이터를 파트 사전에 미리 정의된 단위 데이터의 조합만으로 구성하게 하고, 단위 데이터의 조합으로 이루어지는 데이터도 파트 사전에 등록하여 관리함으로써 데이터 표준을 꾀할 수 있다. 즉, 클래스 이름이나 데이터베이스 테이블 이름 등에서 사용되는 각종 데이터명을 표준화된 단위 데이터들의 조합으로 구성하는 것만을 허용함으로써 동일한 데이터가 서로 다른 용도로 사용되거나 동일한 사물을 칭하는 데이터가 개발자들간에 서로 상이한 데이터 이름으로 사용되는 것을 방지하여 개발과 유지 보수 생산성을 향상시킬 수 있다.

그러므로, 개발자가 개발 과정에서 데이터를 정의하여 사용하고자 할 때에는 (그림 2)와 같이 우선 파트 사전에 원하는 데이터가 있는지를 찾아보고, 있으면 등록된 데이터를 사용하며 없는 경우에는 파트 사전에 등록된 단위 데이터의 조합으로 구성할 수 있는지를 살펴보아, 가능한 경우에는 단위 데이터의 조합으로 데이터를 구성하여 승인 과정을 거쳐서 파트 사전에 등록하고, 원하는 단위 데이터가 존재하지 않을 때는 먼저 단위 데이터를 생성하여 승인 과정을 거쳐 승인 후 단위 데이터들을 조합하여 원하는 데이터를 생성한다. 이와 같이 단위 데이터나 단위 데이터의 조합으로 구성된 데이터를 파트 사전에 등록할 경우에는 항상 정해진 승인 과정을 거치도록 함으로써 데이터 남용을 방지할 수 있다. 데이터에 관한 신규 등록뿐만 아니라 모든 정보 객체에 대한 변경(등록/수정/삭제) 사항은 항상 정해진 승인 절차를 필요로 하며 이는 정보 객체 변경 관리에서 관리한다.

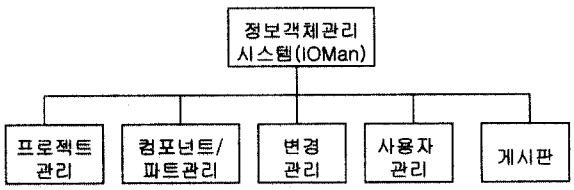


(그림 2) 파트사전 등록 과정

4. 시스템 구현

구현된 통합 정보 객체 관리 시스템의 구성은 1) 원격지에서의 분산 개발 프로젝트를 효율적으로 관리하기 위한 프로젝트 관리 기능, 2) BOC를 중심으로 한 정보 객체 구성 및 관계 관리와 데이터의 표준화 사용을 위한 파트 사전 관리를 수행하는 컴포넌트/파트 관리 기능, 3) 각종 정보 객체의 등록과 변경 사항을 관리하는 변경 관리 기능, 4) 특정 정보 객체 및 특정 기능에 대한 접근 가부 판단이나 사용자 권한을 관리하기 위한 사용자 관리 및 5) 게시판 기능으로 구성되어 있다. (그림 3)에 통합 정보 객체 관리 시스템(IOMan)의 기능 구조도를 보인다.

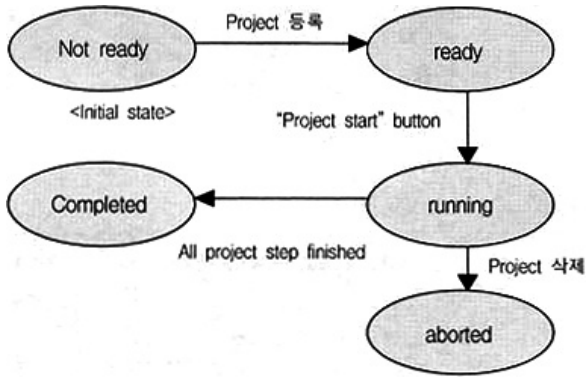
시스템의 구현은 원격지에서의 용이한 접근이 보장되도록 웹 기반 시스템으로 구성하였고 시스템 구조의 개방성과 재사용성을 향상시키기 위해 3 계층 CS(Client-Server) 구조로 설계하였다. 이를 위해 사용자 인터페이스 부분은 JSP(Java Server Page)[17]와 자바 애플릿(applet)을 사용하였고, 응용 로직 부분은 자바 클래스와 자바빈즈(Java Beans)을 이용하였다. 웹서버로는 Apache 1.3.19[18]과 서브릿(Servlet) 엔진으로 Tomcat3.2[19]를 사용하였으며 정보 객체들의 영속적인 저장 관리를 위해서 MS SQL2000[20]을 사용하였다.



(그림 3) 통합 정보객체 관리시스템 기능구조도

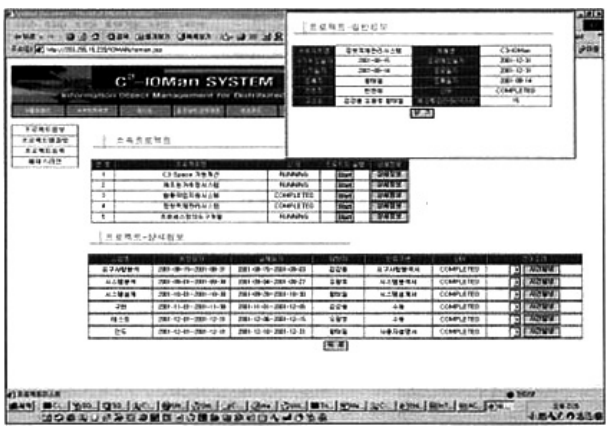
4.1 프로젝트 관리

프로젝트 관리는 분산 환경에서의 소프트웨어 개발 프로젝트 수행에 필요한 세부 활동들과 그 순서들을 정의하는 작업 템플리트 등록 기능과 등록된 작업 템플리트를 이용하여 해당 프로젝트 세부 스텝과 일정을 등록하여 진도 관리를 하는 기능으로 나누어져 있다. 프로젝트 스텝을 등록할 때는 해당 스텝에서 요구되는 베이스라인 문서를 등록할 수 있게 하여 이를 이용해 프로젝트 진도 관리를 용이하게 하였다. 하나의 프로젝트는 (그림 4)에서와 같이 5가지 상태를 가질 수 있으며 시스템 내외부 사건에 의해 상태 전이가 발생한다. 프로젝트 내의 각 세부 활동들인 프로젝트 스텝도 프로젝트 상태와 동일한 5가지 상태를 갖는다. 프로젝트 스텝 등록시 베이스라인 문서가 등록된 작업에 대해서는 해당 베이스라인 문서가 승인 절차를 거쳐 시스템에 등록되면 자동적으로 해당 스텝의 상태가 'Running'에서 'Completed'로 전이된다. 현재 정의되어 있는 베이스라인 문서는 요구사항 분석서, 시스템 분석서, 시스템 설계서, 테스트 결과서, 사용자 설명서 등이 있다.



(그림 4) 프로젝트 상태전이

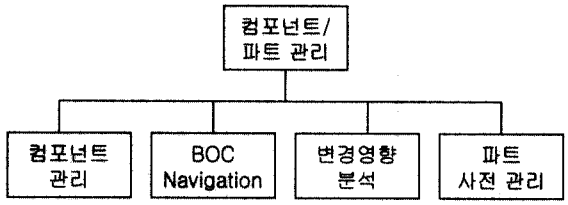
(그림 5)는 프로젝트 정보를 보여주는 화면으로 중앙 프레임에는 로그인한 팀원이 참여하거나 참여 중인 프로젝트 리스트를 보이며, 상단 프레임에는 특정 프로젝트의 구성원, 일정, 현재 상태 등과 같은 일반적인 정보를 보여준다. 하단 프레임에서는 특정 프로젝트를 구성하는 상세 스텝의 일정 계획/실적 및 상태 정보를 나타낸다.



(그림 5) 프로젝트 정보

4.2 컴포넌트/파트 관리

컴포넌트/파트 관리는 (그림 6)에서 보는 바와 같이 크게 네 가지 기능으로 분류되며, 이 중에서 컴포넌트 관리는 제품, 문서, 패키지, 클래스와 같은 단위 컴포넌트들의 상세 정보를 유지 관리하는 '단위 컴포넌트 관리'와 컴포넌트간 관계를 등록하는 '컴포넌트 관계 관리' 기능을 수행한다. (그림 7)에 정보 객체 유형 중 하나인 '패키지'에 대해 객체 저장소에 저장되어 있는 패키지에 대한 전체 목록과 특정 패키지 정보객체에 대한 상세 정보와 관계 정보를 예로 보여준다.



(그림 6) 컴포넌트/파트 관리 기능

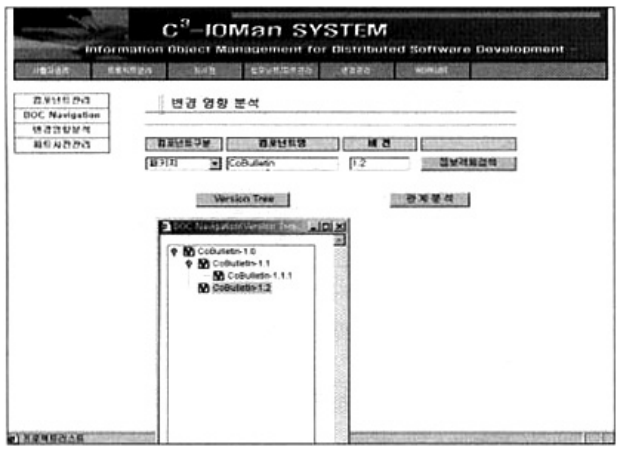
(그림 7) 컴포넌트 정보

BOC 네비게이션에서는 하나의 제품에 대한 구조 정보를 트리 형태로 그래픽하게 보여주며, 변경 영향 분석에서는 특정한 정보 객체를 변경하고자 했을 경우 해당 정보 객체의 변경이 다른 정보 객체에 어떤 영향을 주는지를 분석하기 위해 정보 객체간의 구성, 참조, 종속, 파생 관계를 그래픽하게 보여준다. (그림 8)에서 특정 제품(이 예에서는 C3-Space)의 BOC 구조를 보여주는데 제품, 문서, 패키지, 클래스가 서로 다른 아이콘으로 구분되어 있으며 문서는 기술문서와 관리 문서가 색깔로 구분되어 표현된다.

하나의 정보 객체 내에서의 변경 이력에 대한 파생 관계는 버전 트리로 표현되는데, (그림 9)에서와 같이 변경 영

향 분석 기능에서 분석하고자 하는 정보 객체명을 기입하고 '버전 트리' 버튼을 누르면 지정된 정보 객체가 하이라이트 되면서 해당 정보 객체의 버전 이력을 트리 형태로 나타낸다. (그림 10)에는 특정한 정보 객체에 관한 여러 가지 관계를 분석한 화면으로, 이 예에서는 하나의 제품에 포함되어 있는 '클래스' 정보 객체의 구성 정보를 BOC 형태로 가장 좌측 프레임에 나타내며, 하이라이트된 해당 정보 객체의 세 가지 관계(클래스-문서 관계, 클래스-패키지-제품 관계, 슈퍼클래스-서브클래스-패키지 관계) 정보를 별도의 프레임으로 구분하여 나타내고 있다.

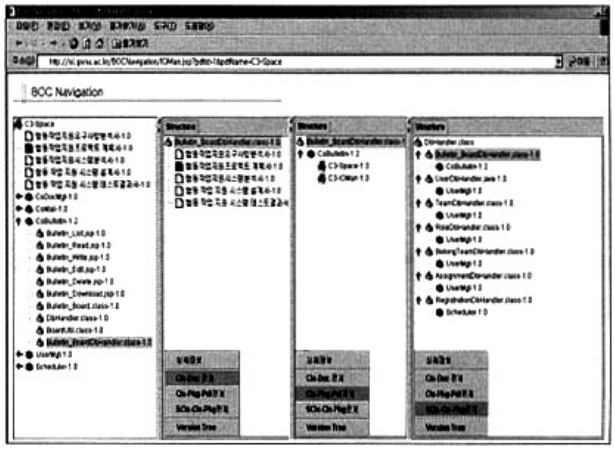
(그림 8) Bill Of Class



(그림 9) 버전 트리

파트 사전 관리에서는 데이터의 표준화 사용을 위해 단위 데이터와 데이터의 정보를 일관되게 유지 관리하여 프로그램 작성시 객체 저장소에 등록된 데이터만을 사용하도록 제한함으로써 개발 및 유지 보수 생산성 향상을 꾀한다. 데이터의 유형으로는 클래스명, 클래스 속성명, 클래스 메서드명, 패키지명, DB 테이블명, DB 테이블 컬럼명이 있다. (그림 11)에 단위 데이터와 데이터 목록을 나타내는데, 데이터의 경우에는 이름, 한글 명칭, 영문 명칭, 데이터 유형, 길이, 설명, 작성자, 등록일 등과 같은 정보를 가지고 있다.

모든 정보 객체가 객체 저장소에 체크인(check-in)될 경우에는 사전에 정해진 승인 절차를 거치도록 하며, 체크아웃(check-out)될 경우에는 단순 사용인지 변경인지를 구분하여 변경을 위한 체크아웃일 경우에는 해당 정보 객체에 잠금 기능을 설정함으로써 하나의 정보 객체가 동시에 여러 곳에서 변경될 수 없게 제어한다. 그리고, 현재 잠겨있는 정보 객체를 체크아웃 하고자 할 경우에는 현재 소유자와 체크아웃 일자를 보여줌으로써 개발자간에 상호 조정을 할 수 있는 정보를 제공한다.



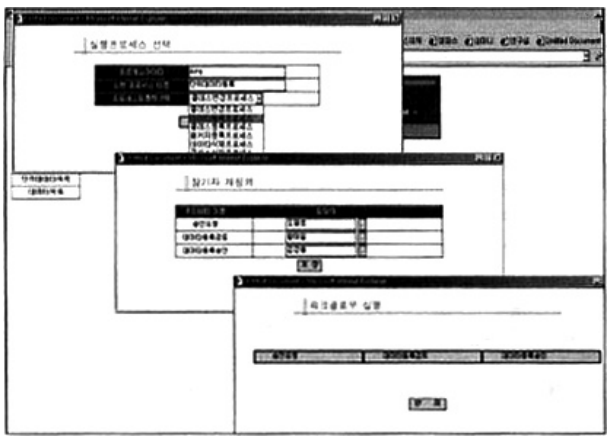
(그림 10) 정보 객체 관계 분석

(그림 11) 단위데이터/데이터 목록

4.3 변경 관리

각 정보 객체에 대한 변경 사항은 버전 번호에 의해 관리되며 변경 이력은 버전 트리에 의해 관리된다. 변경 관리는 정보 객체에 대한 등록, 수정, 삭제 등 각종 변경 사항이 발생했을 때 정해진 승인 절차를 관리하는 워크플로우(work-

flow) 관리와 버전 관리로 구성되어 있다. 워크플로우는 "전체적인 조직의 목표를 달성하기 위해 정해진 규칙들에 의거하여 참여자들 사이의 정보 및 업무가 전달되는 절차와 과정들을 자동화하는 것"이라고 정의되며[21], 본 연구에서는 정보 객체들의 변경 사항에 대한 승인 절차를 위해 워크플로우 관리를 적용한다. 예를 들면, 신규 데이터 등록에 관한 승인 절차는 승인 요청 -> 검토 -> 승인의 과정을 거치게되며 각 단계에서 구체적으로 수행해야 하는 업무 내용 및 담당자 등의 할당이 사전에 정의되고 실행시에는 이를 기반으로 승인 절차가 자동적으로 실행된다. (그림 12)에서는 데이터 신규 등록 승인 절차를 처리하기 위해 사전에 정의된 프로세스 중에서 하나를 선택해서 실행 프로세스 이름을 기입하고 참여자를 재정의 함으로써 실행 프로세스를 구동시키는 과정을 나타내고 있다. 이와 같이 워크플로우 관리는 필요한 변경 승인 절차를 사전에 정의하는 '프로세스 정의' 기능과 정의된 프로세스를 기반으로실제 작업 흐름을 관리하는 '프로세스 실행' 기능 및 각 개발팀원들이 수행해야 할 작업 내용과 상태 등을 관리하는 '작업목록(worklist) 관리' 기능으로 이루어져 있다. (그림 13)의 작업 목록에는 수행해야 할 작업 내용, 해당 작업의 현재 상태, 도착일 및 작업 수행에 사용되는 응용 시스템 등의 정보가 나타나 있다.



(그림 12) 워크플로우 실행

(그림 13) 작업목록과 게시판

4.4 사용자 관리

사용자 관리는 분산 개발팀원들의 소속팀과 역할 등을 정의하고 접속 승인이나 접근 권한 여부 등을 관리하는데, 구성원들은 하나 이상의 프로젝트 팀에 소속될 수 있으며 하나 이상의 역할을 가질 수 있도록 설계하여 조직이나 역할 변화에 유연성을 갖도록 하였다. 각 팀원들의 역할 정의는 워크플로우 관리에서 적합한 업무 담당자를 선택하는 기준의 하나로 사용된다.

4.5 게시판

게시판은 원격지에 떨어져 있는 프로젝트 팀원들에게 알려야 할 사항을 전체적으로 공지하거나 특정한 안건에 대해 상호 의견을 교환할 때 사용된다. (그림 13)은 개발팀원들이 시스템에 로그인 한 후에 나타나는 메인 화면으로 본인이 수행해야 할 작업 목록과 게시판이 나타나 있다.

5. 결론 및 향후 과제

현재 급속히 확산되고 있는 원격 분산 환경에서의 소프트웨어 개발을 지원하기 위해서는 효과적인 정보 객체 관리가 매우 중요하나 기존의 연구에서는 1장에서 제시한 통합 정보 객체 관리 요구사항의 일부분만을 충족하고 있는 실정이다. 본 연구에서는 효과적인 통합 정보 객체 관리에 대한 요구사항을 세 가지로 정리하고, 이를 충족시키기 위해 BOC라는 관리 틀을 제안하여 소프트웨어 개발에 필요한 각종 정보 객체들을 하나의 틀로 구조화하였고, 정보 객체간의 관계를 구성, 참조, 종속, 파생 관계 등으로 정의하여 각종 정보 객체들의 관계를 일관되게 관리할 수 있는 방법을 제시하였다. 그리고, 소프트웨어를 구성하는 부품이라고 할 수 있는 데이터들에 대해 파트 사전 관리를 통해 데이터의 표준화 사용을 꾀함으로써 개발 및 유지 보수 생산성을 향상시키는 방안을 제시하였다. 마지막으로, 제안한 통합 정보 객체 관리를 정보 시스템으로 구현하여 그 기능성과 구조를 보였다. 본 연구에서 구현한 통합 정보 객체 관리 시스템은 현재 급속히 확산되고 있는 분산 개발팀에 의한 소프트웨어 개발과 컴포넌트 기반 소프트웨어 개발을 효과적으로 지원하는 유용한 개발 관리 도구가 되리라 판단된다.

본 연구의 향후 과제로는 첫째, 현재는 각 정보 객체들의 메타 정보나 정보 객체간의 관계 정보를 사용자가 수동으로 지정해주어야 하기 때문에 시간이 많이 걸리고 번거로운데 이를 시스템에서 소스 코드 등을 읽어서 자동으로 정보를 추출하는 방법에 대한 연구가 필요하고, 둘째로 구현된 시스템과 외부 상업용 개발 도구들과의 통합이 필요하며, 셋째로 현재 본 연구는 정보 객체 관리에 중점을 두고

개발되었는데 1장에서 언급한 분산 환경에서의 통합 소프트웨어 개발 관리 시스템에 대한 네 가지 요구 조건을 모두 충족시키기 위한 기능 구조 설계 및 시스템 구현에 관한 연구가 있다.

참 고 문 헌

[1] Frank Maurer, Barbara Dellen *et al.*, "Merging Project Planning and Web-Enabled Dynamic Workflow Technologies," IEEE Internet Computing, Vol.4, No.3, pp.65-74, 2000.

[2] Jarir Chaar, Santanu Paul and Ram Chillarege, "Virtual Project Management for Software," NSF Workshop on Workflow & Process Automation, University of Georgia, Athens, GA, May, 1996.

[3] James A. Highsmith III, Adaptive Software Development, pp.261-293, Dorset House Publishing, 2000.

[4] 한관희, 박찬우, "제품 정보관리시스템 개발을 위한 기능 분석에 관한 연구", 한국 CAD/CAM 학회논문집, Vol.7, No.1, pp. 42-56, March, 2002.

[5] Gregory Alan Bolcer and Richard N. Taylor, "Endeavors : A Process System Integration Infrastructure," Proceedings of the 4th International Conference on the Software Process, Brington, England, December, 1996.

[6] G. E. Kaiser *et al.*, "WWW-based Collaboration Environment with Distributed Tool Services," World Wide Web Journal, Vol.1, No.1, pp.3-25, 1998.

[7] Frank Maurer and Harald Holz, "Process-Oriented Knowledge Management for Learning Software Organizations," Proceedings of 12th Knowledge Acquisition Workshop (KAW '99), Banff, Canada, 1999.

[8] Stephen A. MacKay, "The State of the Art in Concurrent, Distributed Configuration Management," Proceedings of the 5th International Workshop on Software Configuration Management, Seattle, WA, April, 1995.

[9] Andre van der Hoek, Dennis Heimbigner and Alexander L. Wolf, "Does Configuration Management Research Have a Future?," Proceedings of the 5th International Software Configuration Management Workshop, Seattle, WA, April, 1995.

[10] Rational Software, "Simplifying the Process of Change-Rational ClearCase," www.rational.com, Cupertino, CA, 2001.

[11] James J. Hunt, Frank Lamers, Jurgen Reuter, and Walter F. Tichy, "Distributed Configuration Management via Java and the World Wide Web," Proceedings of the 7th International Workshop on Software Configuration Management, Boston, MA, pp.161-174, May, 1997.

[12] Hal Render and Roy Campbell, "An Object-Oriented Model of Software Configuration Management," Proceedings of the 3rd International Workshop on Software Configuration Management, Trondheim, Norway, pp.127-139, June, 1991.

[13] Wolfgang Appelt, "WWW Based Collaboration with the BSCW System," Springer Lecture Notes in Computer Science 1725, Berlin, pp.66-78, 1999.

[14] John Noll and Walt Scacchi, "Supporting Software Development in Virtual Enterprise," Journal of Digital Information, Vol.1, No.4, http : //jodi.ecs.soton.ac.uk, December, 1998.

[15] 선수균, 송영재, "통합 관리 모델을 이용한 효율적인 객체 관리 저장소 설계", 정보처리학회논문지, Vol.8-D, No.2, pp. 166-174, 2001.

[16] 이병엽, 박준기, 양성모, 김현정, "다중 프로젝트간의 데이터 표준화 지원을 위한 도구의 설계 및 구현", 한국정보과학회 '98 추계학술대회논문집, pp.535-537, 1998.

[17] 공성현 외 6명, JSP 레퍼런스 바이블, 베스트북, 2001.

[18] www.apache.org.

[19] jakarta.apache.org.

[20] www.microsoft.com/sql/.

[21] Object Management Group, "Workflow Management Facility Specification : Version 1.2," http : //www.omg.org, April, 2000.

한 관 희

e-mail : hankh@nongae.gsnu.ac.kr

1982년 아주대학교 산업공학과(학사)

1984년 한국과학기술원 산업공학과(석사)

1996년 한국과학기술원 산업공학과(박사)

1984년~1989년 대우전자 MIS실

1990년~1999년 대우정보시스템(주) 기술연구소

2000년~현재 경상대학교 산업정보공학과 조교수

관심분야 : 객체지향 기술, workflow, CSCW, KMS