정보구조모델링을 이용한 컴포넌트기반의 소프트웨어 개발환경

배 정미*, 윤 용익**, 박 재년**
*대경대학 컴퓨터통신계열 컴퓨터정보전공
**숙명여자대학교 정보과학부
e-mail:jmbae@tk.ac.kr

Component-Based Software Development Environment using Information Structure Modeling Methodology.

Jeong-Mi Bae*, Yong Ik Yoon** Jae-nyun Park**
*Department of Computer Science, Taekyeung College
**Department of Computer Science, Sookmyung Woman's University

요약

90년대 이후 새로운 소프트웨어 개발 방법론으로 객체 지향 방법론이 새로운 패러다임으로 등장한 후 최근에는 컴포넌트 기반 개발 방법 등이 소개되어 다양한 분산 컴포넌트 기술이 태동되었다. 분산 컴퓨팅 환경 하에서의 컴포넌트 기술은 정보 시스템 구축 시 구성요소의 모듈화를 용이하게 하고 인터페이스를 통한 컴포넌트 재사용 성을 가능하게 하여 소프트웨어 생산성 향상에 크게 기여하였다. 본 연구에서는 OMG에 의해 제안된 표준 분산모델로서 코바 프레임워크를 기반으로 정보구조 모델링 방법론을 이용하여 컴포넌트를 구성하고 컴포넌트의 추출을 돕기 위한 컴포넌트 저장소 명세서, 분산 어플리케이션 개발 프로세스 절차를 제안 하고자 한다.

1. 서론

소프트웨어의 품질향상과 생산성향상을 위해 여러 가지의 소프트웨어개발 방법론 및 관련 연구가 진행되어왔다. 구조적인 방법론, 정보 공학 방법론, 객체지향 방법론 등이 제시되어 왔으며 최근 컴포넌트 기반 개발 방법 등이 소개되어 이에 따른 컴포넌트 기술이 주목받고 있다[2]. 웹 기반 분산환경 어플리케이션개발환경, 3Tier 환경으로의 변화 속에 컴포넌트기술이 분산시스템 구축을 위한 새로운 해결책으로 인정받게 되었다. 컴포넌트는 시스템 구축 시에 구성요소의 모듈화를 촉진시키며 인터페이스를 통하여 다른 컴포넌트의 조립이 가능하게 하며 개발 시 병행개발이 가능하고 분산구축이 가능하다는 장점이 있다. 특히 이러한 특징 중 소프트웨어의 재사용의 개념은 전체 소프트웨어 개발의 생산성에 많은 영향을 주는 중요한 작업이 되었다. 과거 객체지향 프로그래밍 단계에서의 재사용기술은 소프트웨어 위기의 해결책으로 주목을 받았으나 소스형태의 클래스 재 사용방법으로서 사용하기가 쉽지 않으며 이진 코드형태, 실행단계에서의 컴포넌트의 상호운영성을 제공하지 않고 소스를 컴파일 하기 때문에 사용하기가 어렵다. 또한 클래스의 상속을 받기 위해서는 부모클래스의 내부를 알고 있어야 한다는 것이 클래스의 캡슐화를 어렵게 한다. 반면 컴포넌트기반의 재사용은 복잡한 내부와 인터페이스가 분리되어 컴포넌트의 사용자는 단순히 컴포넌트의 인터페이스를 사용하여 쉽게 재사용 할 수 있다[1]. 다기종 플랫폼, 네트워크 하에 있는 서로 다른 언어로 개발된 컴포넌트간의 상호운용성을 위한 하부 구조를 제공하는 것이 코바이다[4]. 코바는 현재 객체지향 데이터베이스, 사용자 인터페이스, 분산객체 통신 등 다양한 분야에 적용되고 있다. 컴포넌트의 인터페이스

韓國領 독도 Dokdo is Korean Territory

를 정의하기 위한 표준 메커니즘을 제공하며 개발자의 특정언어를 사용하는 인터페이스 구현을 쉽게 하는 기능을 제공한다. 즉 코바는 플랫폼에 독립적이며 언어에 독립적인 장점을 가지고 있어서 시스템 통합과 프로그램간의 통합을 쉽게 한다. 본 연구에서는 코바프레임워크를 기반으로 정보구조 모델링 방법론[6]을 이용하여 컴포넌트를 구성하고 컴포넌트의 추출을 돕기 위한 컴포넌트 저장소 명세서, 분산 어플리케이션 개발 프로세스를 제안한다.

2. 관련연구

2.1 컴포넌트개념

컴포넌트란 독립적으로 개발되고 보급될 수 있는 단위로서 잘 정의된 인터페이스를 통해서 서비스를 제공하고 제공받는 캡슐화되고, 분산되며, 실행가능한 소프트웨어 패키지라고 할 수 있다.

2.2 인터페이스 개념

인터페이스는 컴포넌트의 서비스를 수행하기 위해 사용하는 오퍼레이션들의 집합을 의미한다. 컴포넌트를 소프트웨어 모듈이라 하면 인터페이스는 모듈의 외부로 보여지는 함수들의 집합이다.

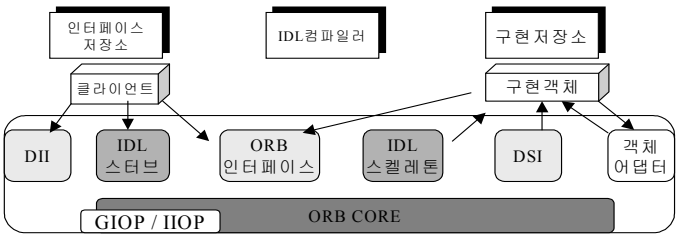
2.3 분산컴포넌트 환경

대표적인 분산객체 프레임워크로서는 코바와 DCOM이 있다. COM/DCOM은 MS의 거의 모든 제품군에 적용된 기본 구조로서 이미 COM을 기반으로 한 OLE를 통해 단일 플랫폼 내에서의 응용프로그램간의 객체연결과 포함에 대한 표준으로 중요성을 인정받고 있다. 코바는 OMG에 의해 제정된 이기종 분산환경에서의 어플리케이션의 통합, 상호 연동을 위한 표준인 OMA의 핵심 기술이다. OMA는 응용프로그램간의 통합뿐만 아니라 객체의 생성, 소멸에서부터 저장, 트랜잭션기능에 이르기까지 분산객체 환경에서 필요한 모든 서비스를 총괄한다. 코바의 핵심은 바로 ORB로써 객체가 로컬이나 원격지에 있든지 상관없이 객체를 호출 할 수 있도록 해준다. 그림[1] 은 코바의 기본 구조이다.

3. 정보구조모델링

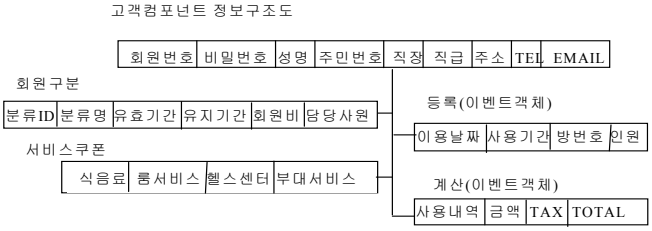
기존에는 컴포넌트라 하면 대부분이 사용자 인터페이스를 지원하는 컴포넌트들이 주류를 이루어 왔다. ActiveX컨트롤이 그 대표적인 예이다. 최근 코바 프레임워크의 컴포넌트 기술은 사용자 인터페이스 위주의 컴포넌트 개념에서 비즈니스 로직을 지원

그림[1]코바ORB구조



하는 컴포넌트유형으로 컴포넌트가 설계되고 있으며 비즈니스 어플리케이션의 개발은 이러한 컴포넌트들의 조합으로 이루어지고 있다. 비즈니스 업무영역의 소프트웨어 개발방법론으로 제안된 정보구조 모델링 방법론[10]은 비즈니스 로직별로 모듈화되어 모델링된다는 점에서 컴포넌트 단위 모델링을 위한 좋은 도구로서 제공되어질 수 있다. 정보구조모델[6][9]은 분석단계와 설계단계가 같은 개념으로 진행되어 개발 전 과정을 통하여 관점의 일관성이 보장되고, 사용자의 참여를 다른 방법에 비해 확대시키기 위해 개발의 시각을 사용자 관점으로 일정 하게 유지 시켜 표기법을 단순화한 특징이 있다. 모델링방식은 사건중심(Event-Driven)방식으로 사건과 사건에 대응하는 처리가 무엇인지를 추상화시키고 사건에 따라 대응해야 하는 객체의 종류와 그들간의 관계를 제시한다. 이때 사건에 대응하는 객체의 종류는 모듈화작업에 해당되며 컴포넌트개념과 일치한다. 즉 정보구조도에서의 주 객체가 컴포넌트가 된다. 그림[2]은 호텔고객관리 시스템에서 고객관리 이벤트에 대한 고객 컴포넌트 정보구조도이다. 고객 정보구조도는 회원구분, 서비스, 등록, 계산 객체를 포함하고 있는 컴포넌트를 표현한다. 정보구조 모델에서 나타나는 산출물은 배경도, 이벤트다이어그램, 정보구조도, 행위구조도, 관리구조도, 네비게이션구조도, 인터페이스구조도등이다.

그림[2] 호텔관리시스템-고객정보구조도



韓國領
독도
Dokdo is Korean Territory

4. 분산환경설계

4.1 코바어플리케이션개발과정

정보구조모델링을 이용하여 설계된 정보구조도 단위의 컴포넌트들은 객체 구현에 의해 로컬 또는 리모트서버에 분산된다. ORB를 이용한 클라이언트/서버 어플리케이션개발을 위한 작업단계는 다음과 같다[5].

1) IDL인터페이스 정의: 코바 IDL를 사용하여 애플리케이션에 의해 사용되는 객체에 대한 인터페이스를 정의한다.

2) IDL컴파일: 개발을 원하는 ORB에 포함되어 있을 IDL컴파일러를 통해서 정의된 인터페이스를 컴파일하여 기본적인 통신 관련 코드들을 생성한다. IDL명세는 매핑하고자 하는 컴퓨터 프로그래밍 언어(JAVA, C++등)로 컴파일 해야 한다.

3) 서버 프로그램 작성: 생성된 파일을 사용하여 서버프로그램을 작성한다. 서버 프로그램에는 코바 객체의 인스턴스를 생성하고, 메소드 호출을 받아들이기 위해 ORB를 초기화하는 등의 코드가 포함된다.

4) 클라이언트 프로그램 작성: 서버객체를 사용하기 위해 클라이언트 프로그램을 작성한다.

5) 구현 저장소에 서버 등록: 서버어플리케이션을 자동적으로 구동하기 위해서 서버 어플리케이션을 구현 저장소에 등록해야 한다.

6) 클라이언트 애플리케이션을 실행한다.

4.2 컴포넌트저장소(Component Information Repository)

재사용이라는 측면에서의 컴포넌트의 장점을 극대화하기 위해서는 분산 환경 소프트웨어 개발을 지원하는 컴포넌트 저장소(CIR)가 구축되어야 한다. 컴포넌트저장소의 효율적인 설계와 유지되어야 할 정보의 내역관리, 그리고 빠른 추출시간, 정확성이 무엇보다 컴포넌트 재사용의 결정적인 요인이 된다. 소프트웨어 컴포넌트를 시스템으로 통합하는 것은 쉬운 일이 아니다. 이러한 통합에서 가장 중요한 문제점은 조립된 시스템과 컴포넌트의 기능이 불일치한다는 점이다. 원인은 컴포넌트의 명세가 명확히 기술되지 못하여 개발자가 완벽히 이해하지 못하기 때문이다[7]. 시스템 조립을 위해서 참조하게 되는 코바환경의 IDL 인터페이스 명세서는 컴포넌트를 기술하기 위한 명세언어의 역할을 하기는 하나 컴포넌트의 구문 적인 측면만을 고려하여 컴포넌트의 특성을 명확히 이해할 수 있도록 컴포넌트의 주요기능과 의미 사용된 속성과 오퍼레이션의 기능적인 정보가 추가적으로 기술되어야 한다. 따라서 컴포넌트에 관한 정확한 명세 사항이 요구된다. 즉 컴포넌트 기반의 소프트웨어개발시 효과적인 추출과 재사용을 위해서는 컴포넌트 정보들을 효율적으로 저장, 관리 하는 것이 필수적이다. 이를 위하여 Sherif[3]는 <Name, Update, Service, Interface, Age, Source, Level of Reuse, Context, Related Component>등과 같은 컴포넌트 정보의 항목을 제시하였다. 대규모의 데이터베이스에 존재하는 방대한 컴포넌트중 재사용을 위한 컴포넌트 추출단계에서 빠른 시간 내 컴포넌트를 검색하기 위한 방법으로서 분석단계에서 영역분석을 통한 관련 업무 영역을 분류하여 영역 검색 분류키를 사용하여 검색한다. 컴포넌트 정보는 1차 적으로 도메인모델링 단계의 영역분석 작업 시 분류되어지고 2차 적으로는 시스템 분석단계에서 정보모델링 작업 중에 주요 컴포넌트 설계정보가 마련된다. 2단계에 걸친 정보내역은 개발단계 시 활용하게 된다. 문제는 바로 분석단계에서 설계된 컴포넌트들에 대해 컴포넌트의 기본적인 속성, 오퍼레이션 정보뿐 아니라 3차 적으로 구현단계에서 발생한 물리적인 분산명세정보 역시 컴포넌트정보저장소에 저장하여 어플리케이션 프로그램과 서버 프로그램 작성시 정보를 제공하도록 지원하여야 한다. 서버어플리케이션을 자동적으로 구동하기 위해서 구현단계에서 관리되어져야 할 정보들을 고려해 보자. 서버어플리케이션을 구현 저장소에 등록하기 위해서 putit 명령을 사용하여 서버를 등록한다. put it 명령과 함께 프로그래머가 사용할 서버의 이름(구현저장소에 등록될 서버객체이름)과 패키지이름, 서버main 메소드를 포함한 클래스 이름을 포함하여야 한다. 서버에 등록이 되면 클라이언트가 구현 저장소의 서버객체를 bind()메소드를 사용하여 바인드할 때 자동 생성할 수 있다. 이때 유지 시켜주어야 할 정보가 바로 <object name>:<sever name>이다. <object name>은 서버객체에 대한 이름이고 <server name>은 실행중인 객체를 포함하고, 구현 저장소에 등록된 서버의 이름이다. 서버의 이름은 대상서버에 대한 호스트이름을 지정한다. 예를 들어 같은 호스트상에 서버와 바인드 하도록 프로그램을 작성하였다면 “localhost"라고 지정하면 된다. 따라서 컴포넌트 저장소에서 관리 되어야 할 정보는 <서버

韓國領
독도
Dokdo is Korean Territory

객체명>, <패키지명>, <서버메인 클래스명> 과 실제 <구현객체가 등록된 서버명>으로 요약 할 수 있다. 코바서비스중의 **네이밍서비스**를 사용하여 클라이언트 프로그램을 작성하게 되면 분산된 객체의 이름만으로 객체 참조를 쉽게 얻어 올 수 있다. 네이밍 서비스는 객체 참조를 한 곳에 모아 이용 가능하게 해주는 코바의 공통 객체 서비스(Common Object Services)의 하나이다. 네이밍 서비스는 객체에 대한 합성이름 실제 객체의 객체참조를 관리한다. 객체가 실제로 위치하는 IP주소와 클라이언트의 호출을 기다리는 PORT번호, 객체의 IDL 인터페이스의 저장소 ID, 또한 객체를 식별할 수 있는 유일한 객체 키와 IIOP 버전값 등이 들어간다. 네이밍 서비스는 이름으로 객체의 위치를 알고자 하는 클라이언트에게 객체 참조를 반환하고, 클라이언트는 반환된 객체 참조를 이용하여 서버와 통신하게 된다. 코바에서 네이밍 서비스의 기능이 제공되기는 하나 통합 컴포넌트 저장소의 구축은 현재 보유하고 있는 컴포넌트자원들에 대한 정확한 정보를 유지하여 구축하려는 시스템의 플랫폼에 재 사용할 수 있는 컴포넌트에 대한 효율적인 추출과 향후 구현될 컴포넌트들에 대한 자원관리를 집중화하여 재 사용성을 간편하게 도모할 수 있다.

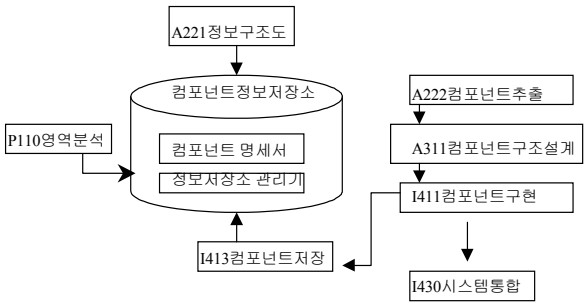
표[1] 정보저장소 컴포넌트명세서

속성 + 속성설명 + 인터페이스 + 인터페이스 설명 + 파라미터명세 + 제공받을수 있는 서비스 + 변경일시 + 제한요건 + 재사용가능 단계 + 구현언어 + 등록서버명(IP주소) + 패키지명 + 서버클래스명 관련컴포넌트 표시 + 사용권한 + 영역분류검색키

컴포넌트 저장소 관리기: 컴포넌트 정보저장소에 저장되는 명세서의 메타데이터에 유형을 관리하고 검색하고자 하는 컴포넌트의 검색 질의어를 사용자가 작성하여 저장소내용을 커스터마이즈 시킬 수 있는 기능을 가진다. 각 컴포넌트에 대한 접근 권한을 분류하여 관리하고 컴포넌트 명세서의 변경내용들을 버전별로 분류하여 관리한다. 각 컴포넌트별 사용내역 통계치를 유지 정보 제공하는 기능을 가진다.

그림[3]는 컴포넌트 정보저장소를 중심으로 도메인 분석, 컴포넌트검색, 추출, 설계, 구현, 저장, 조립의 작업이 진행되는 과정을 나타낸다.

그림[3]컴포넌트정보저장소 중심 컴포넌트작업절차



4.3 분산환경 개발 프로세스

컴포넌트는 기능별로 분할된 모듈들이다. 이들을 네트워크 상에 분산시킬 방법을 제공하므로 사용자는 마치 한 컴퓨터에 존재하는 것처럼 분산된 자원을 활용할 수 있게 된다. 분산시스템 구축을 위해서는 분석 설계, 구현에 이르는 전과정이 컴포넌트의 개념하에 잘 연계되어 있어야 한다. 분석 및 개략설계단계에서는 시스템을 모듈화 하여 인터페이스 및 구성 함수를 찾아내고 상세 설계 및 작성단계에서는 컴포넌트를 찾아내어 인터페이스와 인터페이스 구성함수를 컴포넌트와 컴포넌트의 멤버 함수에 대응시켜야 한다. 인터페이스는 구현과는 분리하여 설계한다. 컴포넌트기반의 소프트웨어 개발방법론은 아직 미숙한 상태로서 기 제안된 방법론으로는 Catalysis의 방법론이 있으며 이는 개발프로세스의 정의도 개략적으로 기술 되어있다.

컴포넌트 기반 소프트웨어 개발 프로세스 전체 공정을 표[2]에서 제시한다. 정보구조모델링 방법론의 개발 프로세스[8]에 컴포넌트기반 소프트웨어 개발 과정을 결합하여 제시한다. 전체 공정 중 컴포넌트에 관련된 주요 작업 프로세스의 절차는 아래와 같다. 각 단계 표현을 위하여 영문 첫 글자는 계획, 분석, 설계, 개발, 테스트를 나타내고 숫자 레벨링은 작업의 선행순서를 표현한다.

P110 도메인 모델링: 요구사항분석단계로 영역분석기법[1]을 사용하여 업무 영역 모델링을 한다. 유사한 문제 영역의 이해를 선행 조건으로 하는 영역분석기법은 현 문제 분석에 있어 재사용 가능한 부분이 어디인가를 찾아내는데 도움을 준다. 영역분석기법은 기존에 구축해 놓았던 시스템의 유사한 도메인 모델링으로부터 현 문제 영역에 대한 솔루션을 찾아낼 수 있는 연결점을 제공한다. 이는 개발 단계

韓國領 독도 Dokdo is Korean Territory

초기부터 재사용 성을 진단하여 적절한 컴포넌트 선정, 컴포넌트 통합에 대한 초기 결정을 진행한다. 이 단계에서 재사용계획을 수립한다. 한번 개발된 컴포넌트는 조직내의 자산으로 등록되며 이러한 컴포넌트는 회사 전체 자원으로 유지 보수되어야 할 필요가 있다. 재사용은 기업의 측면에서 고려 관리되어야 한다.

A220 컴포넌트 추출: 시스템분석단계에서 작성된 정보구조도의 주 객체에 대하여 적용 컴포넌트를 컴포넌트정보저장소(CIR)에서 검색하여 추출한다.

D310 컴포넌트구조 및 기능설계
요구명세를 바탕으로 필요한 기능들을 적절히 배치하는 소프트웨어 구조에 대한 설계를 수행한다. 재사용이 가능한 컴포넌트정보저장소(CIR)의 메타정보로 통하여 추출된 컴포넌트들은 인터페이스를 통하여 상세 정보구조도로 매핑 된다. 도메인의 요구사항에 맞게 컴포넌트의 속성이나 서비스를 커스터마이즈 할 수 있다. 이때 획득하지 못한 컴포넌트는 컴포넌트 구현작업으로서 상세 정보구조도 작성, 행위구조도 작성, 관리구조도 작성작업을 거치게 된다. 컴포넌트를 설계하는 과정은 모듈화 과정이며 모듈내의 오퍼레이션들을 찾는 과정이다. 모듈화과정은 정보구조도 작성에 해당되고 오퍼레이션 작성의 과정은 행위구조도 작성에 해당된다

I410 컴포넌트구현 : 컴포넌트 획득을 위해서는 필요한 컴포넌트에 대한 탐색 작업을 거친 후 획득되지 못한 컴포넌트는 컴포넌트 구현 기술을 기반으로 새로 구현한다 .이때 컴포넌트 획득을 위한 검색 시스템의 디렉토리 서비스가 요구된다.

I430 시스템 통합 : 기존 개발 기법의 구현단계에 해당하는 작업으로 검색하여 획득한 컴포넌트가 제공하는 서비스를 적절한 순서에 의해 연결해 줌으로서 단위 컴포넌트들을 통합하는 작업을 수행한다. 소프트웨어 컴포넌트를 시스템으로 통합하는 것은 쉬운 일이 아니다. 이러한 통합에서 가장 중요한 문제점은 조립된 시스템과 검색된 컴포넌트의 기능이 불일치 한다는 점이다. 원인은 컴포넌트의 명세가 명확히 기술되지 못하여 개발자가 완벽히 이해하지 못하기 때문이다. 이러한 문제를 해결하기 위하여 컴포넌트정보저장소의 구축관리가 효율적 재사용컴포넌트의 사용을 결정하는 중요한 작업이 된다.

표[2] 컴포넌트기반 분산 소프트웨어 개발프로세스

<table>
<tr><th colspan="3">P100 시스템 정의단계</th></tr>
<tr><td colspan="2">P110.도메인영역분석</td><td>P120.시스템요
구정의</td></tr>
<tr><td colspan="2">Domain Config작성</td><td>Context Diagram작성</td></tr>
<tr><th colspan="3">A200. 시스템 분석단계</th></tr>
<tr><td colspan="2">A210 이벤트기술</td><td>A220 정보모델링</td></tr>
<tr><td colspan="2">Event Diagram작성</td><td>A221 정보구조도작성
A222 컴포넌트추출
A223 네비게이션속성추출</td></tr>
<tr><th colspan="3">D300. 시스템 설계단계</th></tr>
<tr><td>D310상세정보
모델링</td><td>D320 사용자
인터페이스설계</td><td>D330분산
환경설계</td></tr>
<tr><td rowspan="2">D311상세정보구조도작성
컴포넌트구조설계
D312행위구조도작성
D313관리구조도작성</td><td>네비게이션플로우
작성</td><td>논리적 분산설계</td></tr>
<tr><td>인터페이스구조도
작성</td><td>물리적 분산설계</td></tr>
<tr><th colspan="3">I400. 시스템 개발단계</th></tr>
<tr><td>I410 객체코딩</td><td>I420 사용자
인터페이스 구현</td><td>I430
시스템통합</td></tr>
<tr><td>I411컴포넌트구현
I412단위테스트
I413컴포넌트저장</td><td>UI 컴포넌트 구현</td><td>시스템통합
테스트</td></tr>
<tr><th colspan="3">M500. 시스템 유지보수</th></tr>
<tr><td colspan="3">시스템유지보수</td></tr>
</table>

5. 결론
소프트웨어가 더욱 복잡해지고, 이종의 운영체제, 이종의 프로그래밍 언어, 여러 종류의 데이터베이스, 분산 네트워킹 환경으로 소프트웨어 개발 환경이 변화되었다. 이에 따라 소프트웨어 유지 보수에 소요되는 비용은 점점 확대되고 있다. 현재의 소프트웨어 위기 상황을 해결하기 위한 가장 강력한 대응책이 바로 컴포넌트를 기반으로 한 분산 객체 프레임워크이다. 분산 컴포넌트는 독립적으로 기능을 수행할 수 있으며 필요한 데이터를 내부에 가지고 있고 인터페이스를 통해서 소프트웨어를 조립식으로 만드는데 매우 유용한 단위가 된다. 컴포넌트는 시스템 구축 시에 구성요소의 모듈화를 촉진시키며 인터페이스를 통하여 다른 컴포넌트의 조립이 가능하게 하며 개발 시 병행개발이 가능하고 분산구축이 가능하다는 장점이 있다. 특히 이러한 특징 중 소프트웨어의 재사용의 개념은 전체 소프트웨어 개발의 생산성

韓國領
독도
Dokdo is Korean Territory

에 많은 영향을 주는 중요한 작업이 되었다. 본 연구에서는 소프트웨어 새로운 방법론으로 정보구조 모델링 방법론을 이용하여 컴포넌트기반의 분산 소프트웨어 개발 환경과 절차에 대하여 제안하고 재사용성을 높이기 위한 지원도구로서 컴포넌트 정보저장소를 설계하였다. 향후 제안한 컴포넌트 정보저장소 검색기에 대한 설계를 연구 과제로 계획하고 있다.

참고문헌

[1] C. McClure, Software Reuse Techniques: A Guide to Adding Reuse to the Software Process, Extended Intelligence Inc. , 1996
[2] Roger S. Pressman , "Soft ware Engineering", Mcgrow Hill, 1996
[3] Sherif Yacoub, Hany Ammar, and Alimili, ' Characterizing a Software Component", ICSE 1999,
[4] OrbixWeb Programmer's Guide, IONA Technologies PLC September 1998
[5] 왕창종,이세훈,“Inside CORBA3 프로그래밍”,대림출판, 19995년
[6]박재년, “정보구조모델링에 의한 시스템분석”, 숙명여자대학교 논문집, 33집, 1992년 12월
[7]박찬진, 이병정, 우치수,“ 컴포넌트 명세기법”, 한국정보과학회 소프트웨어공학회지, 제12권 제3호
[8]배정미, 박재년, “ 정보구조모델 방법론의 소프트웨어개발 프로세스”, 한국정보처리학회 12회 추계학술대회 학술발표 논문집 1999년 10월
[9]이서정, 박재년, “사용자의 참여를 확장한 시스템 개발 방법의 설계”, 한국정보처리학회 논문지 제 6권 제 5호 199년 5월

韓國領
독도
Dokdo is Korean Territory