COM+를 이용한 WebProgramming 오류검출의 시각화 설계

박 성 준*, 전 인 오**, 양 해 술*
*호서대학교 벤처전문대학원
**호서대학교 글로벌창업대학원
e-mail : popo8402@hanmail.net, shinjin88@hotmail.com
hsyang@office.hoseo.ac.kr

Design about Visualization of Error Detection on WebProgramming Using COM+

Sung June Park,*, Hae Sool Yang*, In ho Jeon*
*Dept. Application of Computer Technology, Hoseo Graduate School of Venture
** Hoseo Graduate School of Global Enterprenuership

요 약

웹구현을 위한 통합솔루션 환경과 재사용성·확장성의 확실한 장점을 지닌 COM+ 기술은 많은 연구를 통해 계속된 진화가 이루어지고 있다. 한편, COM+의 유용성을 이해하면서도 제어하고 관리할 수 있는 프로그램 개발과 유지보수 단계에서의 어려움은 상존한다. 본 논문에서는 CR다이어그램을 통해 운용을 모니터링 화하여 디버깅시 구조파악을통한 오류검출이 가능하도록 하였으며, 세부적으로 COM+를 그 실체인 Entity와 속성(Attribute)과 관계(Relationship)로 표현하여 구체화 하였다.

Ⅰ. 서 론

개발 환경에서의 모니터링화 연구는 소프트웨어 공학 분야에서 계속적으로 이루어지고 있는 분야이다. 소스 분석을 위한 모니터링화 구현 등이 대표적이며, 본 논문은 최근까지 이슈로 통용되고 있는 COM(Common Object Model)에서의 디버깅시 보다 편리한 방법론을 제안하였으며, COM+를 이용한 WebProgramming 오류 검출의 모니터링화 설계를 하고자 한다.

최근까지 고급기술로서 많이 다루어지는 COM+는 COM 기술에 추가된 강력한 기술이며, 소프트웨어 공학 분야에서 강조되는 재사용성(Reusing)과 확장성(Extended)의 내용을 반영한 기술이며, 지속성 있는 데이터와(Presistent) 공용 애플리케이션을 가지고 캡슐화 되어 있다는 점이 있다. 그리고 또 COM+는 이미 존재하는 소스 코드에 커다란 변화 없이 수명 관리, 트랜잭션 지원, 보안 그리고 COM 오브젝트에서의 트랜잭션 같은 서비스를 추가함으로써, 마이크로소프트는 COM을 보다 확장 가능한 기술이자 커다란 스케일의 분산 개발(Distributed Development)에서의 보다 적합한 기술로 끌어올렸다. 이 COM+기술은 복잡한 구조상의 관계를 가지기 때문에 구조를 이해하기도 어려운 점이 많으며, 더 나아가서 디버깅(Debugging)시에 구조상 파악의 이유로 디버깅 작업의 어려움까지 이어지게 된다. 이러한 배경으로 본 연구는 WebProgramming에서의 COM+의 오류 검출 모니터링화 모델인 'CR-Diagram' 모델을 제안함으로써 기존의 개발 환경에서의 복잡하게 얽혀있는 구조상의 문제를 보다 쉽게 표현을 하였다.

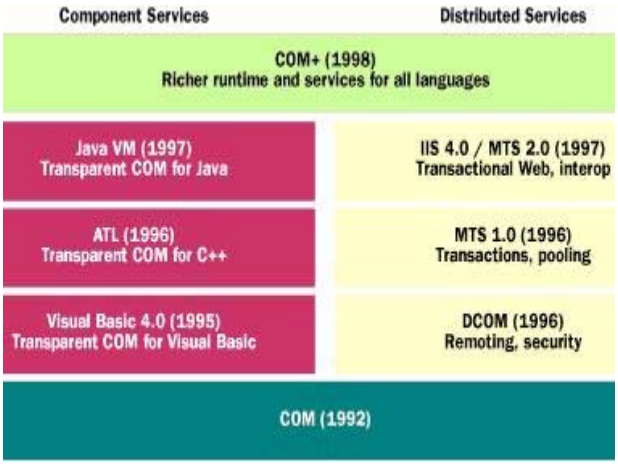
Ⅱ. COM+ Error Detection

1. COM+의 이해

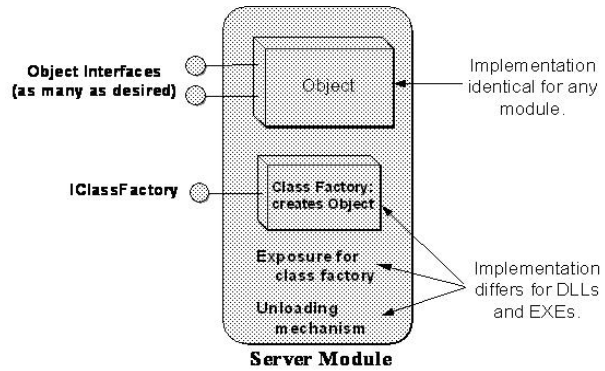
COM+(Component Object Model)는 서버군인 윈도우즈 2000의 출시와 함께 COM이 OLE 2.0에 대한 보강 책으로

처음 발표되고, 많은 관심을 모아오는 고급기술이다. COM+는 쉽게 말해서 COM의 발전체라고 생각하면 이해하기 쉽다. 또한 COM+는 마이크로소프트 트랜잭션 서버(Microsoft Transaction Server, MTS)와 마이크로소프트 메시지 큐(Microsoft Message Queue)가 결합되고. 몇 가지 특징이 추가된 것이라고 다시 표현할 수 있다.

COM+의 이해를 더 돕기 위하여 COM+의 원조격인 COM의 간단한 특징을 살펴보자. 첫째, 프로젝트 개발기간을 단축하여 준다. 둘째, 프로젝트 개발기간이 단축됨에 따라 자연히 비용도 절감된다. 셋째, COM은 프로그래밍언어에 종속적이지 않다. COM+로 발전된 이유 중 하나는 수많은 트랜잭션을 요구하는 다수의 사람들을 처리하기 어렵다는 것이었는데, COM+는 이러한 내용을 보완함으로써, 더욱 고급화되고, 체계화되었다. (그림 1)은 분산 환경에서의 COM+의 진화과정을 보여주는 그림이며, (그림 2)는 COM+의 기초적인 내용을 도식화한 그림을 나타낸 것이다.



(그림1) 분산 환경에서의 COM+ 진화과정

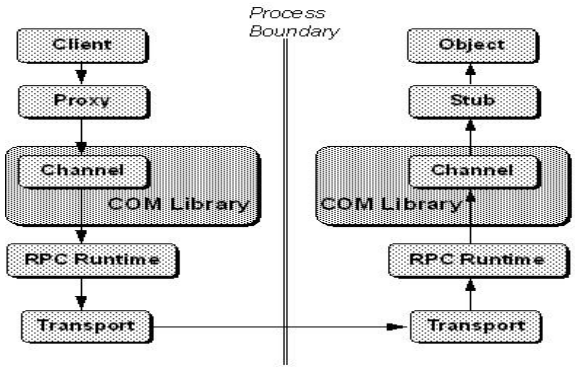


(그림2) Fundamental of COM+

2. COM+간의 관계 및 WebPage와의 관계

COM+는 복잡하고 정규화된 내용을 담고 있는데, 모듈 처리가 많이 이루어지는, 규모가 작지 않은 프로젝트에서 많이 사용되어지는 기술이다.

단순히, WebPage와의 COM+간의 관계는 서로와 서로를 연결해주는것에서 벗어나 하나의 WebPage가 다른 COM+로 이어지게 되고, 이어진 COM+는 또다른 COM+로 이어질 수 있는 등 관계적으로 보았을 때 많은 복잡함을 내포하고 있는 것이 사실이다.



(그림 3) COM+와 COM+간의 1:1처리내용

설명을 보충하기 위해서 하나의 도식을 더 살펴보면, COM+와 COM+는 Process Boundary를 기준으로, 서로 독립된 객체로 존재하나 Client에서 Proxy로 Proxy에서 Chanel을 거쳐 RPC Runtime을 거쳐 Transport까지 다다르게 되면, 다른 COM+와 통신을 하는 즉, 독립된 객체로 구성되어있지만, Processing시에는 서로 데이터를 주고받는 것을 볼 수 있다.

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using System.Configuration;

using UIS.Registration ;
using UIS.Lesson ;
using UIS.User;
```

(그림 4) .NET환경에서의 COM+ 사용

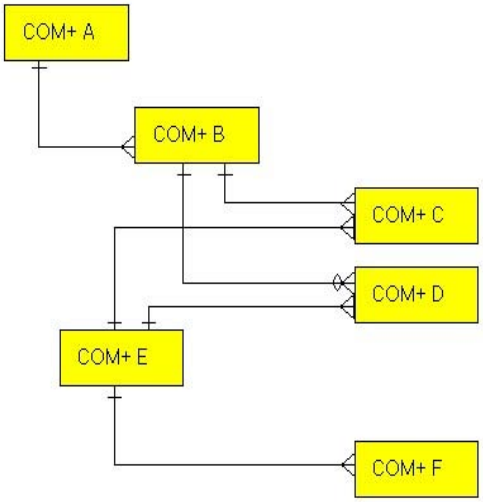
위의 (그림 4)는 UIS라는 하나의 프로젝트에서 Lesson COM+, Registration COM+, Staff COM+, User COM+를 선언한 것이다.

3. COM+오류검출 모니터링화의 필요성

앞에서도 언급하였듯이 COM+는 일반적으로 프로그램의 설계에 근거한 복잡한 관계를 가지고 있다. 일반 소스의 디버깅(Debugging)과 달리 COM+는 복잡한 관계 구조에 의해 디버깅시 많은 어려움을 야기 한다. WebPage와 COM+간의 관계, 복잡한 트랜잭션 관계를 가진 COM+와 COM+ 모델간의 관계를 모니터링화(Monitoring)을 하여 표현한다면, 프로그래머는 전체적인 프로젝트의 윤곽을 모니터링 화에 의해 더 명확하게 이해함과 동시에 관계와 관계간의 선언된 내용을 쉽게 파악함으로써 디버깅하는 시간을 줄이고, 관계를 파악이 용이한 일석이조의 효과를 가져온다.

Ⅲ. COM+ 오류검출의 모니터링화

1. 모니터링화의 구성



(그림 5) 모니터링화의 구성

COM+로 표기가 된 Entity(의미 있는 유용한 정보를 제공하기 위하여 기록, 관리하고자 하는 프로그램의 유형으로 객체, 컴포넌트 등이 될 수 있음)와 각COM+들의 Attribute로 구성이 되며, RelationShip(관계)으로 구성되어있으며, Reflexive(재귀적 관계)와 Dependent(종속적 관계)등으로 구성되어진다.

2 각 구성의 정의

의미 있는 정보이며, 다른 Entity와 관계가 존재되어야하는 Entity는 '의미 있는 유용한 정보이기 위하여 기록, 관리 하고자 하는 프로그램의 유형이다.

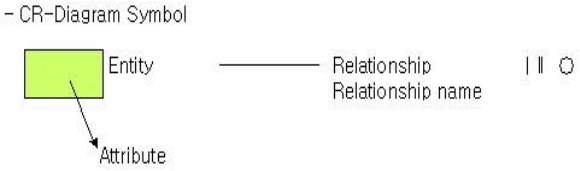
Relationship은 Entity간의 존재하는 상호간의 연관성으로, 해당 실체와 관련된 작업이 수행된다.

Relationship의 관계 형태의 결정을 살펴보면 관계는 기수성(Cardinality)과 선택성(Optionality)으로 이루어져 있으며, 작업 흐름상의 두개의 Entity가 관련된 작업 규칙을 추출하여 기수성과 선택성을 조사한 후 두가지를 통합하여 관계를 정한다'

Attirubte는 Entity에 통합되는 구체적인 데이터 항목으로 더 이상 분리될 수 없는 최소의 프로그램의 단위'로 정의가 가능하며, Attribute추출 -> Attribute배치 -> Attribute검증의 세단계로 정의절차가 이루어지며, Attribute추출은 ① 정보 분석 단계에서 수립된 각종 속성 ② Entity정의 시에 파악된 것 ③ 프로세스 모델링 ④ 기존 정보 시스템 분석의 내용으로 추출될 수 있다.

3. CR-Diagram의 설계

CR-Diagram의 기본설계는 (그림 6)와 같다.



(그림 6) CR-Diagram의 기본설계

Cardinality of Relationship(Degree)은 (그림 7)과 같다.

(그림 7) Cardinality of Relationship

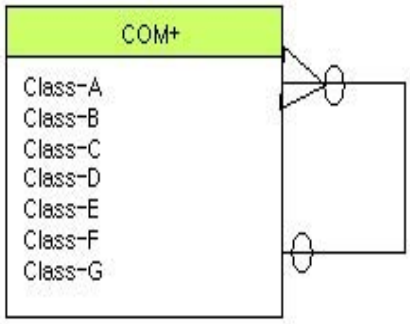
Optionality of Relationship은 (그림 8)과 같다.

(그림 8) Optionality of Relationship

관계의 완성표현은 (그림 9)과 같다

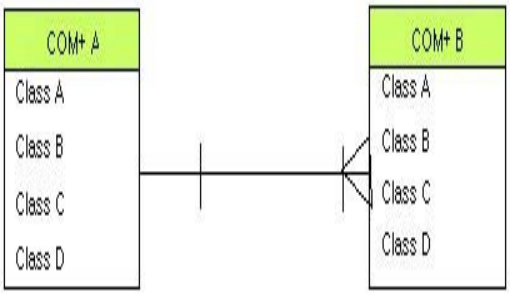
(그림 9) 관계의완성표현

재귀적 관계(Reflexive)의 표현은 (그림 10)과 같다



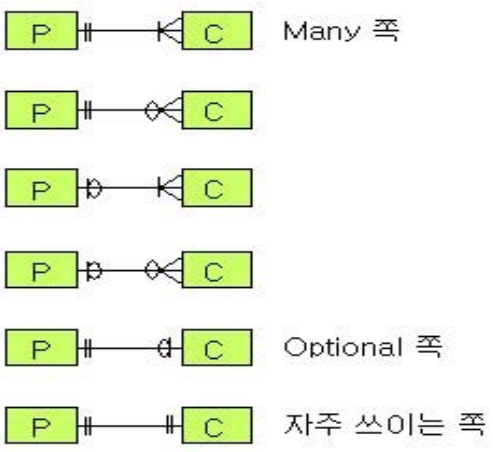
(그림 10) 재귀적관계(Reflexive)

종속적 관계(Dependent)의 표현은 (그림 11)과 같다



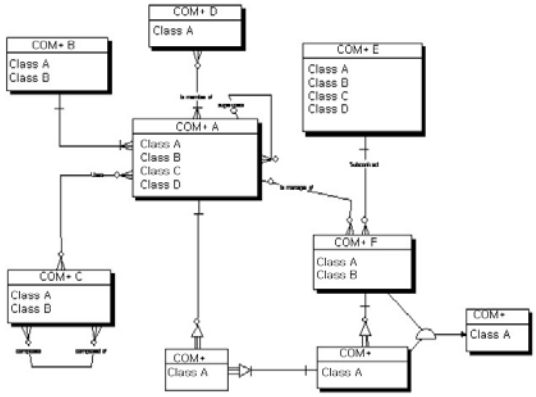
(그림 11) 종속적관계(Dependent)

ParentEntity와ChildEntity식별 법은(그림 12)과 같다.



(그림 12) ParentEntity 와 ChildEntity식별법

완성된 CR-Diagram의 설계표현은 (그림 13)과 같다.



(그림 13) CR-Diagram의 표현

Ⅵ. 결론 및 향후 연구 과제

웹구현을 위한 COM+ 기술은 많은 연구를 통해 계속된 진화가 이루어지고있다. 한편, COM+의 유용성을 이해하면서도 제어하고 관리할 수 있는 프로그램 개발과 유지보수 단계에서의 어려움은 상존한다. 이러한 어려움을 해결하기위하여 본 논문에서는 CRDiagram설계를 통해 운용을 모니터링화하여 디버깅시 구조파악을통한 오류검출이 가능하도록 설계 하였으며, 세부적으로 COM+를 그 실체인 Entity와 속성(Attribute)과 관계(Relationship)로 표현하여 구체화 하였다. 향후 본 설계를 기초하여 실제적으로 시뮬레이션화 할 수있도록 구현한다면 실시간 오류수정이 요구되는 웹환경에서의 중요한 기술로 인식될 수 있을 것이며, 좋은 자료로 활용되어 질 것이다.

[참 고 문 헌]

[1] Gyu Eddon, Inside COM+ Base Services
[2] 데이비드 플랫, 닷넷 기반 기술에 대한 이해2
[3] 마틴파울러, 엔터프라이즈 애플리케이션 아키텍처 패턴
[4] 아처 · 화이트채플 InSide C#2
[5] 카스 밸린저, 닷넷 웹 서비스(원리와 구현)
[6] 정주현, 홍찬기 “컴포넌트기반의 웹 기반 교육시스템 설계에 관한 연구”,한국정보처리학회지, 제8-D권 제6호, pp.674, 2001.
[7] Alan W. B개주, “Component-Based Software Engeering