개발도구·언어·환경 등에 응용되면서 급속 확산

마이크로소프트의 Active-X 및 선마이크로시스템즈의 Java관련 기술, 인터넷 기반의 컴포넌트 소프트웨어 기술등 개방형, 분산, 객체 기술을 활용한 정보시스템 구축이 일반화될 전망이다. 이 글에서는 객체기술을 이용한 정보시스템 구축에 관련된 개발환경 및 개발도구 등을 소개하고, 그 활용방법 및 업무분석/설계방법(객체지향 분석/설계 방법)을 논의하고 그 활용사례 등을 6회에 걸쳐 기술한다.

이동진/ 한국CIM 대표이사

연·재·순·서

1. 객체기술 개요

객체기술이란 소프트웨어 시스템을 구성함에 있어서 객체를 활용하는 전체적인 기법을 말하며, 객체기술은 시스템을 객체들의 집합 및 이들의 상호 작용으로 규정한다. 이를 위해서는 기존의 객체를 활용하거나 혹은 새로 정의된 객체를 사용하며 특히 분산 환경에서 존재하는 객체들간의 원활한 통신 및 서비스를 지원하기 위한 모든 제반 환경들도 그 대상으로 삼는다.

객체기술이 가져다주는 혜택으로는 기존의 객체를 활용함으로써 얻을 수 있는 구현 경비의 절감 및 객체단위로 이해됨으로써 얻을 수 있는 이해 용이성, 유지 보수의 용이성과 경비 절감을 들 수 있다.

1.1 객체기술의 중요성

객체기술은 새로운 기술이라거나 혹은 단일 성격의 기술이 아니라 오히려 기존에 개발된 여러 기술의 복합체이며, 특히 분산 환경이 가져다주는 특수성으로 인해 이러한 기술들이 독특한 시각에서 재구성된 것으로 이해할 수 있다. 이를 구성하는 요소 기술들로는 객체지향 프로그래밍을 포함하여 객체지향 분석/설계, 통신 프로토콜, 트랜잭션 처리기술, 클라이언트/서버 구축 기술, 분산처리 등을 들 수 있다.

이들이 통합되는 과정에서 일관된 개념들이 있다면 시스템을 구성하는 기본 단위는 객체라는 것이고 객체의 이용 및 통신을 위해 미들웨어 레벨에서 객체의 투명성을 보장한다는 점이다. 객체지향 기법이 제공하는 특성으로는 상속성, 캡슐화, 재사용성 등이 있으며 분산 환경에서 작용 시는 IDL(Interface Definition Language)의 사용, 투명성 등이 있다. 이러한 특성들을 적절히 이용함으로서 얻을 수 있는 이점들은 다음과 같다.

첫째, 이해의 용이성을 들 수 있다. 시스템의 분석/설계에서 시작하여 코딩,

유지 보수에 이르는 SDLC(System Develop-ment Life Cycle) 전체 단계에서 일관성 있게 나타나는 개념이 객체이다.

따라서 시스템이 어떤 계층에서 이해되든지 한번 파악된 객체는 동일한 개념으로 시스템의 각 단계에 나타나므로 다른 기법에 비해 이해가 용이하다.

둘째, 빠른 프로토타입(Prototype)의 개발을 들 수 있다. 객체는 독립적으로 이해되며 구현이 가능하다. 이것은 객체가 갖는 자율성에 기인한 특성이며 시스템 구성에 있어서 중요하다고 생각되는 부분에 있어서는 독립적인 서브 시스템의 구성 및 테스팅이 가능하다.

셋째, 유지 보수의 편리성을 들 수 있다. 응용 시스템을 유지 보수함에 있어서 객체 단위로 관리된다. 따라서 기능에 대한 변경의 파급 효과가 적절히 유지되고 추적이 용이하다.

마지막으로, 재사용 성을 들 수 있다. 한번 개발된 객체의 코드는 재사용이 가능하다. 객체의 재사용은 집짓기 블록과 유사한 개념으로 이해될 수 있으며, 새로운 시스템을 구성할 때 기존에 개발된 객체 단위의 컴포넌트들을 적절한 순서로 배열하여 원하는 기능을 수행하는 방식의 재사용이 가능하다. 이것은 코딩에 대한 시간 및 비용을 절감해 줄뿐만이 아니라 테스팅이 완료된 컴포넌트를 사용함으로써 품질이 우수한 제품을 생산하도록 유도한다.

이상에서 살펴 본 것과 같이 객체기술은 다른 기술에 비하여 많은 이점을 가져다 줄 수 있는 가능성을 가지고 있으며, 인트라넷, 통합 패키지 소프트웨어 등 실제로 최근의 정보시스템 구축에 사용되는 개발 도구들이나 개발 언어와 개발 환경들이 객체기술을 응용한 것들이고 그 사용은 급속히 확산되고 있는 중이다.

1.2 객체기술의 범위 및 분류

객체기술을 이해하기 위하여 우선적으로 객체기술에 대한 요소 기술들을

〈표〉 객체지향 기술 분류표

대구분	중구분	소구분	세구분
1. 분산 시스템 환경	분산 운영체계 기술	마이크로 커널 기술	- 데이터 암호화 기술 - 사용자 인증 기술
		보안 기술	
		프로세스 통신 기술	- 메시지 기반 통신 - 원격 프로시저 호출
		Naming 기술	- Name 정의 기술 - Routing 기술 - Name Resolution 기술
		프로세스 관리 기술	- 스케줄링 기술 - 이전(Migration) 기술
		동기화 및 병렬제어 기술	
		자원관리 기술	- 자원 부하 공유 기술 - 자원 부하 균형 기술 - Deadlock관리 기술
		보호 기술	- 자원 부하 공유 기술 - 자원 부하 균형 기술 - Deadlock관리 기술
	응용서비스 지원기술	파일 서비스 기술	
		객체 트랜잭션 관리 기술	
		API(Application Program Interface) 기술	
	표준화 기술	데이터 통신 프로토콜 기술	(TCP/IP, UDP)
		응용 플랫폼 표준화 기술	
	시각 프로그래밍 환경	프로그래밍 기술	- 비 절차 프로그래밍 기술 - 오류처리 기술 - 퍼지 프로그래밍 기술 - Interactive 프로그래밍 기술 - 절차 추상화 기술 - 아이콘 정렬 최적화 기술 - 병렬 프로그래밍 기술

대구분	중구분	소구분	세구분
		언어 정의 기술	- 시각언어 문법 정의 기술 - 아이콘 자료형 정의 기술 - 아이콘 의미 정의 기술 - Graphic 연산자 아이콘 정의 기술 - 문법에 따른 시각언어 생성 기술
2. 프로그래밍 언어	시각 프로그래밍 환경	컴파일러 구축 기술	- Syntax 정의 기술 - 파싱 기술 - Semantic action 루틴 기술 - 아이콘 사전 기술 - 중간코드 생성 기술 - 동적/정적 타입 검사 기술 - 시각언어 컴파일러-컴파일러 기술 - 오류 처리 기술
		스크린 표현 기술	- 정적 표현 기술 - Interactive 표현 기술 - 2차원 표현 기술 - 3차원 표현 기술 - 미세부분 은닉 기술
		사용자 접속 및 정보검색 기술	- 하이퍼미디어 접근 기술 - 네비게이션 탈피 기술 - 멀티미디어 접속 기술 - 시각 질의 번역 기술 - WWW 브라우저 접속 기술
	재사용 컴포넌트공학	컴포넌트 정의 기술	
		컴포넌트 분류 기술	
		콤포넌트 저장 기술	
		컴포넌트 검색 기술	
		컴포넌트 인터페이스 기술	
	프로토타이핑 언어 기술	명세 언어 정의 기술	
		명세 언어 검증 기술	
		프로그래밍 코드 생성 기술	
		명세 언어 번역 기술	
		실행과정 시각화 기술	- 정적 표현 기술 - Interactive 표현 기술 - 2차원 표현 기술 - 3차원 표현 기술 - 애니메이션 기술
3. 객체지향 DB 기술	DB Modeling	Entity 객체 모델 표준화	
		정형화된 질의어 개발	
	DB 설계 지원 도구	DB 설계를 위한 시각 설계 환경	
		DB 지원 라이브러리 구축	
	분산 DB 기술	자료의 병렬 제어 기술	
		자료 접근 경로 최적화 기술	
		복구 기술	
		카탈로그 기술	
	표준화 기술	DB 엔진을 지원하는 표준화 기술	
		데이터 보호를 위한 데이터 형식 표준	
		트랜잭션을 위한 Customization 표준	
4. Graphic User Interface	이미지 기술	한글 모델	
		한글 Postscipt	
	GUI 도구	Toolkit 개발을 지원하는 Rapid Prototyping 도구	
		코드 생성기	
		스크립트 언어 생성 기술	

대구분	중구분	소구분	세구분
	지능형 사용자 인터페이스	음성	
		터치 스크린	
		전자 펜	
		그래픽스	
	지식베이스	구문에 관한 지식	
		사용자 특징 평가 및 추론 모델	
		대화에 관한 지식 모델	
		작업 및 문제 영역 지식 모델	
		인터페이스 지식 모델	
5. 객체 지향 방법론	프로젝트 관리 기술	소프트웨어 개발 주기 모델	
		버전 및 형상 관리 기술	
	객체 모델링 연구	분석/설계 방법론	- Business/Enterprise 모델
			- 실시간 응용 모델
			- 일반 응용 모델
			- Agent 기반 모델
			- Pattern 기반 모델
		영역 분석/모델	
		메타 방법론	
		재공학 서비스	
	개발 지원 기능	개발 지원 Toolkit	
		객체 검증 및 디버깅 도구	
		GUI 도구	
6. 객체 지향 CASE 도구 개발 기술	통합 CASE 도구	모형 편집 도구	
		일관성 검증기	
		정보 저장소 접속 기술	
		자동 코드 생성기	
		문서 생성을 위한 스크립트 언어	
	재사용 기술	Template을 이용한 재사용 기술	
		Pattern을 이용한 재사용 기술	
		Transformation을 이용한 재사용 기술	
	정보 저장소 구축 기술	정보 저장소 구축을 위한 메타 모델링	
		정보 저장소 인터페이스 라이브러리	
		정보 저장소 표준화 기술 (IRDS, PCTE)	
7. 분산 응용 지원 기술	분산 소프트웨어 개발 방법론	분산 및 병렬 시스템 모델링	
		분산 시스템을 위한 정형 명세 기법	
		분산 시스템 성능 평가 기법	
		분산 시스템 테스팅 기술	
	분산 프로그래밍 지원 도구	CASE 도구	
		정형 명세	
		시각화	
		디버거	
	분산 객체 표준화	ISO/Open Distributed Processing(ODP)	
		OMG CORBA	
		Microsoft OLE/Component Object Model(COM)	
		IC Lab. OpenDoc	
	분산 시스템을 위한 미들웨어	객체 지향 분산 운영 체제	
		객체 지향 네트워크 공학	
		분산 자원 관리 시스템	
		분산 멀티미디어 통합	
8. OMA(Object Management Architecture) 구축 기술	언어 바인딩 기술	IDL 컴파일러	
		Binding 라이브러리	
		Interface Repository	
	ORB 구축 기술		

대구분	중구분	소구분	세구분
	Dynamic Interface Invocation 기술	Implementation Repository	
		Parameterization	
	Object Adater 구축 기술	Basic Object Adapter	- 절차적 언어 (C, COBOL) object adapter - 객체 지향 언어 (C++, Smalltalk) object adapter - 함수 언어 (ML, Haskell) object adapter
		Database object adapter	
		Library object adapter	
	객체 서비스 구축 기술	객체 서비스 I (표준화 작업 완료)	- Naming service - Event service - Lifecycle service - Event service - Persistence service - Concurrency control service - Externalization service - Relationship service - Transaction service
		객체 서비스 II (표준안 진행 중)	- Security service - Time service - Licensing service - Properties service - Query service - Change management service - Trader service - Collections service
	공용 부대 시설 구축 기술	Cataloging Browsing Text editor Printing/Spooling Error reporting Help Tutorial Common access to remote information repository Internationalization	
	응용 객체 구축 기술	OS components E-mail process Signal process Banking classes Inventory classes	
9. 개방형 객체 응용 기술	Business Process Reengineering(BPR) 지원 기술	비지니스 객체 생성 기술	
		비지니스 객체 표현 기술	
		비지니스 시스템 모델	
	CASE 도구 통합 기술	데이터 통합 기술	
		자료 통합 기술	
	멀티미디어 응용 기술	멀티미디어 자료 표현 기술	
		멀티미디어 저장/표현 기술	

파악하는 것이 중요하다. 〈표〉은 STEPI의 기술 분류표와 중앙대학교의 객체지향 기술 분류표를 바탕으로 한 것으로써 객체기술을 이해하는 데에 도움이 될 것이다.

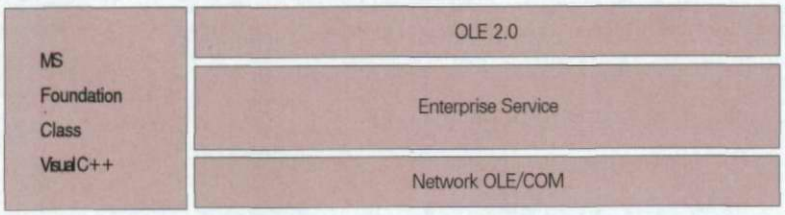
2. 기술 현황 분석

분산 환경하의 객체 기술을 통합 구현한 대표적인 세 부류인 Microsoft의 OLE/COM 계열, IC Lab의 OpenDoc 계열, 그리고 OMG CORBA (Common Object Request Broker Architecture) 접근 방식을 위주로 이들의 구조적 차이점과 유사점 그리고 이를 바탕으로 한 관련 제품들의 개발 동향을 분석한다.

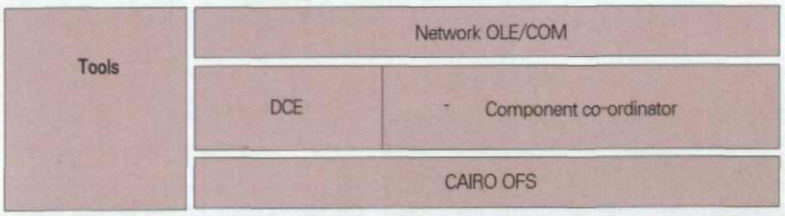
2.1 COM(Component Object Model)과 OLE(Object Linking and Embedding)

Microsoft사는 자사에서 개발한 응용 프로그램들 간의 데이터 공유를 위해 90년도 초에 OLE 기술을 개발하였다. 이후 PC들간의 네트워크 연결이 보편화되고 네트워크상의 자원 공유에 대한 필요성이 증가함에 따라 PC레벨에서 OMG(Object Management Group) CORBA와 같은 분산 객체 공유 기술의 확보가 필요하게 되었다.

이에 Microsoft사는 COM 표준을 개발하게 되었다. COM에 있어서 객체는 컴포넌트라 명칭 되며 COM내에서는 객체가 네트워크 상에 존재하던 데스크톱 상에 존재하던 동일한 방식으로 호출된다. COM 객체는 강한 타입 결합을 지원하는 표준 인터페이스에 의해 통신한다. 이들 인터페이스는 이진 코드로 되어 있으며 COM 객체를 부를 수 있는 매소드에 대한 외부 정의부를 갖는다.



〈그림 1〉 응용 모델 측면의 COM 구조



* OFS : Object Flexible System ,
DCE : Distributed Computing Environment
〈그림 2〉 시스템 관리 측면의 COM 구조

인터페이스는 분리가 불가능하며 따라서 기존의 컴포넌트가 변경되었을 경우는 새 인터페이스를 만들어야 한다. COM 객체 인터페이스의 중요한 점은 객체 인터페이스의 내용은 이를 호출하는 객체에 의해 동적으로 불려질 수 있다는 점이다.

COM에서는 응용 모델의 구성과 시스템 관리라는 두 가지 측면에서 각각의 모델을 제공하며 이들 모델의 대략적인 모습이 〈그림 1〉과 〈그림 2〉에 나타나 있다.

2.2 IC Lab의 OpenDoc 계열

Component Integration(IC) Lab은 비영리 단체로서 1994년 Apple과 IBM Novel/WordPerfect사의 합작으로 설립되었으며 이후 Lotus사의 참여가 있었다. IC Lab은 OpenDoc 구조를 위주로 컴포넌트 소프트웨어의 개발, 이들의 통합에 대한 명세 작성을 목표로 한다.

이들의 연구 대상이 되는 OpenDoc은 문서 위주의 처리를 지양하는 개념으로서 복합적, 협동적, 그리고 맞춤 가능한 문서를 지원하는 구조이다. OpenDoc 컴포넌트 소프트웨어는 Compound-document service, Component service, Storage service, Object-management service, Interoperability service, Automation service의 6가지서비스로 구성되는데 〈그림 3〉은 OpenDoc의 구조이며, 〈그림 4〉는 응용 모델 측면의 OpenDoc 구조이다.

2.3 OMG CORBA 계열

CORBA 계열을 지지하는 회사들은 IBM, Apple, HP, Taligent 및 Novell 사 등이며 모델 역시 〈그림 4〉와 〈그림 5〉의 구조를 갖는다.

3. 객체기술 향후 전망

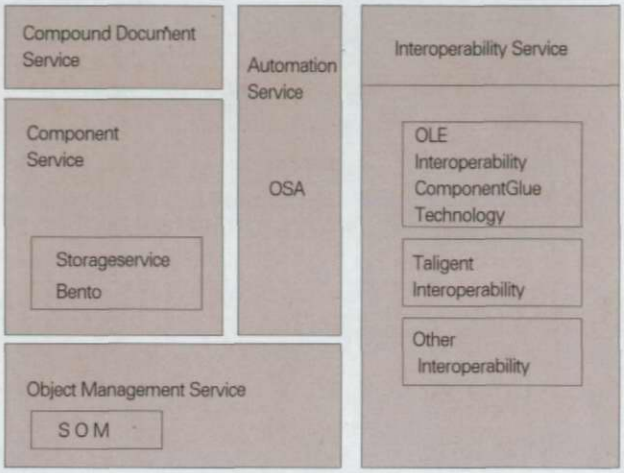
3.1 패키지에서 컴포넌트로 변화

기존의 소프트웨어들은 기능들이 통합된 패키지 형태로 제품화되거나

Customize되어 왔다. 객체기술의 발달에 따라 이러한 추세는 점차 조직의 분화와 함께 이에 상응하는 기능의 분화 과정을 밟아 발전될 것으로 예상된다.

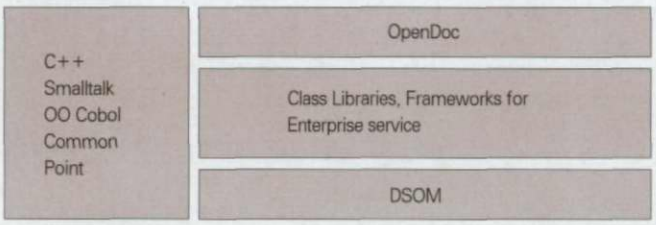
즉, 중앙 집중식 업무 구조가 기능 분화되어 분산 환경에 맞도록 분할됨에 따라 소프트웨어 역시 지리적으로 독립된 노드 상에서 특정한 업무를 수행한다. 이때 유지보수의 단위가 되는 것은 객체이며, 이러한 객체들로부터 새로운 시스템의 구성이 가능할 것이다.

따라서 기존의 소프트웨어 개발이 완성된 패키지 형태의 결과물을 대상으로 하는 형태에서 차후에는 소단위 혹은 특정 기능 단위로 분화된 객체들이 개발의 단위가 될 것으로 기대되며 정보 시스템 개발형태의 주된 형태가 될 것이다.



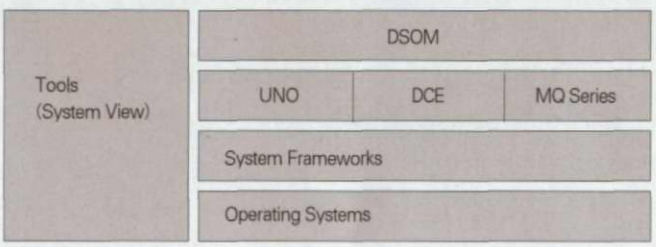
* SOM : System Object Model

〈그림 3〉 OpenDoc의 구조



* DSOM : Distributed System Object Model

〈 그림 4 〉 응용 모델 측면의 OpenDoc 구조



〈그림 5〉 시스템 모델 측면의 OpenDoc 구조

3.2 CASE 도구에서 BPR로

정보기술(IT:Information Technology) 업계에서 개발지원도구라 함은 CASE(Computer-Aided Software Engineering) 도구 환경으로 통합되어 왔었다. 객체지향 CASE 도구 역시 기존에 존재하는 네트워크 상에서 구축하는 소프트웨어 분석 및 설계에 객체 지향 시각으로부터 접근한다는 점에서 큰 차이는 없어 왔다.

그러나 객체를 지원하는 미들웨어가 보편화됨에 따라 이러한 응용 영역은 객체 단위로 변환되고 따라서 기존의 IT 관련 업무가 재 구축되어야 할 필요성이 대두되게 되었다. 객체지향 재공학은 이러한 요구를 수용할 수 있는 방법론과 개발 환경을 제공해주는 중요한 기술이다. 따라서 향후에는 CASE 도구의 모습이 객체지향 비즈니스 재공학을 지원하는 형태로 발전되어 갈 것이다.

3.3 미들웨어의 형태

지금까지 개발된 ORB(Object Request Broker)들은 그 자체가 독립적인 패키지로서 개발, 판매되어 왔다. 하지만 조만간 다른 ORB들간의 상호 운용성이 보장되고 표준화 노력이 정착될 경우 이들은 독립적으로 존재하기 보다 시스템 공급자들에 의해 운영체제의 일부처럼 제공될 것이다. 따라서 사용자는 추가의 노력 없이 자연스럽게 분산 환경 하에서 서비스를 제공받거나 작업할 수 있게 될 것이다.