CORBA를 사용한 이미지 필터 분산 엔진의 개발

정명진[0] 최유주 김경희
이화여자대학교 공과대학 컴퓨터학과
(eastland[0], choirina, mhkim)@ewha.ac.kr

Implementation of Distributed Image Filter Engine By CORBA*

Myung-Jin Jung[0] Yoo-Joo Choi Myung-Hee Kim
Dept. of Computer Science & Engineering, Ewha Womans University

요 약

본 논문에서는 이미지 필터 모듈을 CORBA를 사용하여 분산 엔진으로 구현하는 방법에 대한 연구를 수행하였다. 이미지 필터 모듈들은 영상 처리 과정에서 기본적으로 응용되는 전처리 모듈로서 원 영상을 목적에 따라 변형시키는 기능을 수행하며, 다양한 어플리케이션에서 여러 플랫폼으로 빈번하게 구현되곤 한다. 이 연구에서는 각각 독립적으로 구현된 이미지 필터 모듈을 CORBA를 사용한 분산 서버로 구축함으로서, 다양한 종류의 플랫폼에서 구현된 여러 어플리케이션들이 이미지 필터 기본 모듈들을 별도의 구현 없이 쉽게 사용할 수 있도록 하였다.

1. 서 론

영상 처리 기술은 목적에 따라 응용 범위가 매우 넓고 다양하며, 따라서 본격적인 영상 처리 작업을 수행하기 전에 원 영상은 용도에 맞게 전처리 과정을 거쳐 적합하게 수정된 후 사용되는 경우가 대부분이다.

이미지 필터(image filter)란 이렇듯 영상에 적용하여 특정 목적에 보다 적합하게 만들어주는 전처리(preprocessing) 기술을 말한다. 전처리 단계는 영상 복구(image restoration)와 영상 강화(image enhancement)로 분류될 수 있다. 영상 복구는 오류가 포함된 영상의 오류를 없애는 작업이며, 영상 강화는 원 영상에 대해 이진 출력 혹은 특정 목적에 맞게 변조나 조작을 가하는 작업이다.

이러한 필터 모듈들은 영상 처리의 기본 모듈이라 할 수 있으며 따라서 다양한 어플리케이션에서 여러 플랫폼으로 빈번하게 구현되어지곤 한다. 이렇듯 분산되어 있는 이미지 필터 모듈을 통합하기 위해 CORBA(Common Object Request Broker Architecture)를 사용한 분산 시스템을 구현하여 보았다. CORBA는 객체지향 기술을 바탕으로 응용 프로그램들을 결합하기 위한 미들웨어로 개발 언어나 네트웍, 운영체제 등 이기종 시스템간의 상호 연동이 가능하고 개방적인 표준을 지향한다.

본 논문에서는 이미지 필터 모듈을 CORBA로 통합한 이미지 필터 엔진을 구성하여 PC와 UNIX 시스템에서 접속하여 사용할 수 있는 서버를 구현하여 보았다.

또한 Client가 Server에 손쉽게 접근하고 사용할 수 있도록 unix기반 client에서는 QT를, PC기반 client에서는 MFC를 사용한 GUI를 제작하여 이미지 필터의 적용 및 파라미터 지정, 결과 영상 확인 작업들을 쉽고 편리하게 하였다.

이 논문은 다음과 같이 구성된다. 2장에서 본 이미지 필터 분산 엔진을 구현한 중심 기술인 CORBA에 대해 소개하였으며, 3장에서는 이미지 필터 분산 엔진의 구현 과정, 4장에서 사용자 인터페이스에 대해 설명하고 5장에서 본 논문의 결론을 맺는다.

2. CORBA의 구조

본 이미지 필터 분산 엔진은 C, C++로 구현되어 서로 다른 시스템에 분산되어 있는 이미지 필터 모듈들을 통합하는데 CORBA(Common Object Request Broker Architecture)를 사용하였다.

CORBA는 개발 언어나 네트웍, 운영체제 등 이기종 시스템간의 상호 연동을 가능하게 하는 미들 웨어로서, 약 800여 개 이상의 컴퓨터 관련 단체들이 참여한 OMG(Object Management Group)가 제정한 표준인 OMA(Object Management Architecture)에 의해 표준화되어 있다. OMA는 응용 프로그램간의 통합 뿐 아니라 객체의 생성, 소멸과 저장, 트랜잭션(transaction) 기능에 이르기까지 분산 객체 환경에서 필요한 모든 서비스를 총칭하며, CORBA는 OMA의 가장 중요한 요소가 된다.

CORBA는 컴퓨터 내부의 버스처럼 서로 다른 프로그램 들 사이의 Bus 역할을 하는 모듈로서 각기 다른 언어

* 본 논문은 부분적으로 정보통신부 대학정보통신연구센터(ITRC) 육성 지원 사업과 한국과학재단 가상현실 연구센터 지원사업, 과학기술부 국가지정연구실 사업의 지원에 의해 수행되었음.

로 구현되고, 또 분산되어 있는 모듈들에 대해 마치 로컬에 있는 것처럼 사용할 수 있도록 하는 기능을 제공해 준다. 최근 OMA 2.0의 구조는 그림 1과 같다.

클라이언트에서 서버(구현 객체)로의 요청을 중개하는 것이 ORB(object request server)의 역할이며, ORB는 ORB core, Object Adapter, ORB Interface, Static IDL stub, Dynamic Invocation Interface(DII), Static IDL Skeleton, Interface Repository 및 Implementation Repository로 구성된다.

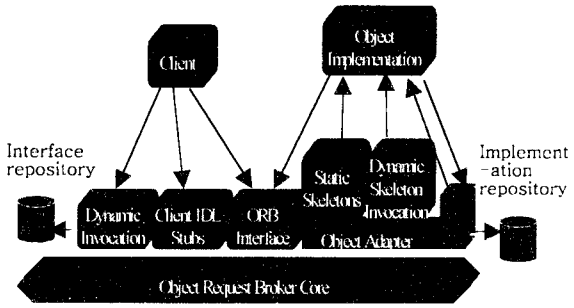

그림 1 CORBA 2.0의 구성

3. 이미지 필터 분산 엔진의 구성

본 논문에서 구현한 이미지 필터 분산 엔진은 크게 CORBA 서버와 CORBA 클라이언트 모듈로 나뉘어 진다. CORBA 서버는 펜티엄III 750MHz cpu, 128M memory, Windows NT server를 사용하는 PC 플랫폼상에서 구현되었으며, 클라이언트는 펜티엄III 750MHz cpu, 128M memory, windows xp 기반 PC와 UNIX IRIX 6.5 시스템에서 각각 구현되었다.

CORBA 엔진을 구성하기 위한 첫 단계로 C와 C++ 및 PASCAL의 서로 다른 언어로 구현된 이미지 필터 모듈을 통합하기 위한 IDL(Interface Definition Language) 파일을 작성한다. IDL이란 인터페이스를 기술하기 위한 정의 언어이며, 인터페이스를 구현 언어로 작성하기위해 IDL 컴파일러를 사용한다. 그림 2는 각기 다른 언어의 모듈이 어떻게 CORBA에 매핑되는가를 보여준다.

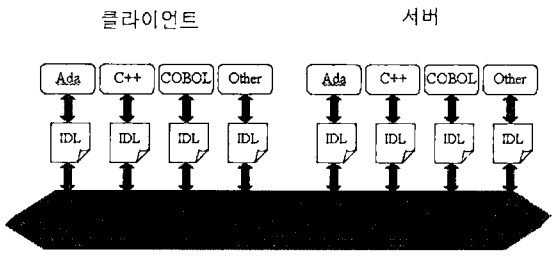

그림 2 CORBA IDL 바인딩

IDL 파일을 만드는 과정에서 CORBA와 기존 구현 언어 간의 데이터 타입의 변환이 이루어 지며, 데이터의 메모리 관리 소유권을 서버, 클라이언트 혹은 양방 모두에게 줄 것인가를 지정해준다. 본 시스템에서는 CORBA와 C++, unix C를 매핑하였다.

IDL 파일은 컴파일 되면서 자동으로 헤더파일을 생성하고 interface 모듈 선언부가 만들어지며, 이 부분이 서버파일에서 구현된다 .

구현 객체를 정의하기위한 헤더 파일을 생성하고, 여기에 IDL에서 선언한 인터페이스의 실제 구현 클래스가 포함되며, ORB를 위한 스켈레톤 코드를 정의한다.

마지막으로 서버 파일의 main() 함수에서 ORB통신을 하기 위해 ORB를 초기화 하고, 클라이언트에게 서비스하기 위해 BOA에 등록하게 되는데, BOA(Basic Object Adapter)는 클라이언트 프로그램과 서버 객체가 사용하는 주요 기능들을 제공하는 어댑터이다.

CORBA 클라이언트는 커맨드 방식 및 GUI 방식으로 작성될 수 있다. 기본적으로 클라이언트에서는 서버 프로그램에서 작성한 스텁 코드를 참조하여 ORB를 초기화 하고 서버의 구현 객체를 호출하게 된다.

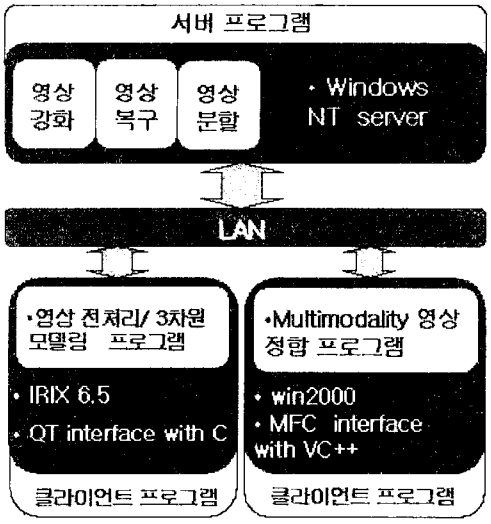

그림 3 이미지 필터 분산 엔진의 구성

그림 3은 이미지 필터 분산 엔진의 전체 구성을 보여준다. 기존의 unix 기반 영상 처리 응용 프로그램 및 PC 기반 응용 프로그램에서는 전처리 작업에서 공통된 이미지 필터 모듈들을 필요 한다. 이러한 각기 다른 언어로 구현된 기존 환경을 클라이언트 인터페이스로 구현하여 서버로 구현된 기본 전처리 모듈들을 호출하는 모습을 보여주고 있다.

4. 사용자 인터페이스

직관적이고 사용하기 편리한 사용자 Interface의 구현을 위해 GUI를 사용한 클라이언트 interface를 구현하였다. PC 플랫폼에서는 MFC를, UNIX 플랫폼에서는 QT를

사용한 인터페이스를 구현하였다. 그림 4는 PC 플랫폼에서 작동하는 MFC 인터페이스를, 그림5는 UNIX 플랫폼에서 작동하는 QT 인터페이스의 모습을 보여준다.

MFC로 구현한 클라이언트는 독립적인 볼륨 영상 처리 모듈로서 영상 전처리 작업에 CORBA 엔진의 이미지 필터를 적용하여 사용한 예이다 . QT로 구현된 UNIX 기반 interface는 PC 플랫폼의 필터 엔진을 유닉스에서 사용하기 위해 작성된 것으로서 볼륨 영상을 불러들이고 윈도우 파라미터를 바를 사용하여 결정한 후 이미지 필터를 적용한 결과를 보여주고 있다.

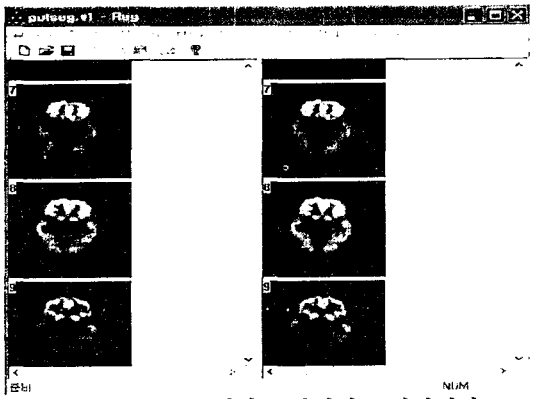

그림 4 MFC로 구현한 클라이언트 인터페이스

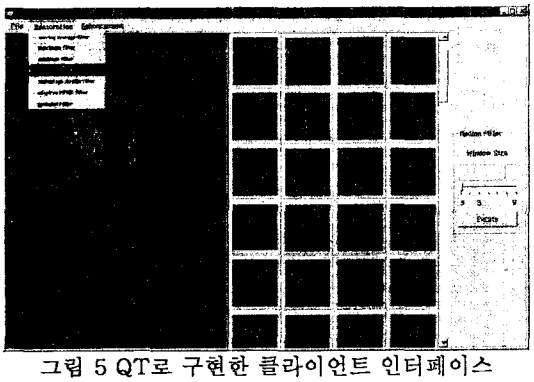

그림 5 QT로 구현한 클라이언트 인터페이스

5. 결론

본 논문에서는 서로 다른 시스템간의 연동을 도와주는 CORBA의 특성을 매우 빈번하게 사용되는 영상 전처리 모듈에 적용하여, 서로 다른 플랫폼상에서 효율적으로 호출할 수 있는 이미지 필터 분산 엔진을 구현하였다.

통합된 이미지 필터 분산 엔진은 영상 처리의 대부분의 전처리 과정에 적용될 수 있으므로 응용 범위가 다양하며, 확장 및 수정이 자유롭기 때문에 새로운 시스템에도 쉽게 적용할 수 있다. 그리고 한 개의 독립적인 시스템으로 존재함으로써 계속적으로 사용될 수 있고, 불필요한 반복 구현의 부담을 덜어줄 수 있다는 점에서도 유용하다고 할 수 있다.

참고 문헌

[1] CORBA Programming, 문왕식 저, 도서출판 대림,2000

[2] www.omg.org

[3] Digital Image Processing Algorithms, Ioannis Pitas, Prentice Hall, 1995

[4] Digital Image Processing, Rafael G. Gonzalez, Richard E. Wood, Addison-Wesley Publishing Company, 1992