Borland®

C++개발에서의 선택

볼랜드 C++Builder와 마이크로소프트 비주얼 스튜디오.NET에 대한 고찰

Jeremy McGee

Borland Software Corporation

2002년 7월

목차

서론

볼랜드(Borland Software Corporation)는 C++이 아직 일반적인 용도의 고성능 컴퓨팅을 위한 언어로 가장 적절한 언어라고 믿고 있습니다. 더욱이 이 언어는 서로 다른 플랫폼에서 기능에 제약을 받지 않고 작동되어야 하는 클라이언트 기반과 서버 기반의 양쪽 모두의 애플리케이션에 있어서 이상적입니다. 볼랜드 C++Builder 6는 ANSI/ISO 표준 컴파일러이며, 윈도우 및 리눅스 호환성을 제공합니다.

C++Builder의 새로운 기능은 개발자들에게 최신의 요구 사항도 지원할 수 있도록 시스템을 쉽게 확장할 수 있게 해줍니다. 특히, C++Builder는 웹 서비스를 통한 애플리케이션의 통합, 인터넷 서버 개발, 직접적인 데이터베이스와의 작업 등의 기능들을 갖추어 오늘날의 개발자에게 이상적인 선택입니다.

이 백서는 소프트웨어 개발 프로젝트의 관리 및 프로그래밍에 관련된 모든 사람들, 특히 C와 C++를 이용한 개발에 있어 기존 시스템에 대한 투자를 보존하면서도 효율성, 품질 그리고 속도를 향상시키고자 하는 사람들을 위한 내용입니다.

C++Builder™

white paper

C++는 가장 인기 있는 언어

객관적인 조사에 의하면, 개발자들은 다른 어떤 언어보다도 C 언어군을 가장 많이 사용한다고 합니다. 어떤 다른 주요 언어도 강력함, 제어 수준, 성능에 있어서 C++과 비교가 되지 않습니다. 이러한 이유로 인해 오늘날의 대부분의 상업용 패키지 소프트웨어 개발은 C 또는 C++를 사용합니다.

C와 C++ 언어는 여러 가지 장점이 있습니다. C/C++ 언어는 정의된 표준을 가지고 있으므로 어느 한 시스템에서 C/C++에 익숙한 개발자는 다른 시스템에서도 C/C++를 사용할 수 있습니다. 두 가지 모두 광범위한 종류의 시스템에서 사용 가능합니다. C++는 C의 저수준의 이점을 가진 유일한 현대적 객체 지향 언어로서, 이로 인해 메모리 점유량, 성능, 하드웨어 및 운영 시스템 레벨의 액세스가 중요한 임베디드, 모바일, 산업용 애플리케이션에서 표준 언어가 되어왔습니다.

아마도 가장 중요한 것은, C++ 언어가 컴퓨터 언어와 컴파일러 디자인에서 최신의 객체 지향적 개발을 고려하면서 발전해왔다는 점입니다. 템플릿, 다중 상속 및 객체 지향에 대한 기본적 접근 등은 C++를 통하여 일반 대중들이 처음으로 광범위하게 사용할 수 있게 되었습니다.

이제 C++는 다시 진화하고 있습니다. 이번에는, 더 높은 생산성을 약속하는 발전된 프로그래밍 환경을 통해서 입니다. 목표는 C와 C++ 개발자들이 더 좋은 코드를 더 빨리 작성하도록 돕는 것입니다.

특히, 개발자들은 웹 서비스를 통하여 시스템에 통합 기능을 더 쉽게 추가할 수 있는 툴을 강하게 요구하고 있습니다. 이러한 시스템은 웹 서버를 통한 브라우저 기반 사용자 인터페이스도 제공해야 할 것입니다. 이것이 볼랜드와 마이크로소프트 모두가 최근에 출시한 C++ 개발 환경의 초점입니다.

마이크로소프트가 최근에 출시한 개발 세트인 비주얼 스튜디오.NET에는 비주얼 C++와 MFC의 최신 버전이 포함되어 있습니다. 여기에는 또한 간결한 C 스타일의 구문을 사용하고, 특히 웹 서비스를 위하여 고안된 새로운 언어인 C#도 포함되어 있습니다.

볼랜드 또한 자사의 C++ 개발 환경을 개선했습니다. 볼랜드 C++Builder 6 Enterprise는 기존 응용 프로그램에 직접 통합시킬 수 있는 미리 개발된 컴포넌트를 사용함으로써 개발자가 웹 서비스를 더욱 간편하게 생성할 수 있습니다. C++Builder를 사용하면 단일 소스로 현재 가장 많이 사용되는 운영 시스템인 윈도우와 리눅스를 위한 크로스플랫폼 애플리케이션을 구축할 수 있습니다.

C와 C++ 애플리케이션을 개발하고 유지/보수하는 개발자들은 결정을 내릴 때가 되었습니다. 어떻게 하면 시간과 지식 면에서 투자했던 것들을 잃지 않고, 코드를 재작성할 필요 없이, 웹 서비스, 그래픽 폼 디자이너 및 기타 생산성 이점을 최대한의 활용할 수 있을 것인가?

C++ 애플리케이션 개발

C와 C++ 개발자들은 애플리케이션 개발에 유용한 다양한 솔루션을 가지고 있습니다. 일반적으로, 무에서 새로운 애플리케이션을 생성하는 경우는 매우 드뭅니다. 그러나 개발자는 단지 이러한 경우에만 기존 코드를 유지하는 데에 관계 없이 가장 최근의 기술을 선택할 사치를 누릴 수 있습니다.

더 전형적인 개발 사례의 경우에는 기존의 애플리케이션을 확장하거나 재작업하게 됩니다. 기존의 코드는 보통 안정되어 있고, 업계에서 신중한 테스트와 검증을 거친 것이기 때문에 특별한 이유가 없으면 변경하지 않는 것이 좋습니다.

윈도우 C와 C++ 개발자들이 처리할 수 있어야만 하는 주요한 애플리케이션의 변경 사항은 다음과 같습니다:

- **웹 서비스 기능 추가.** 웹 서비스는 기업 내에서 그리고 기업들 사이에서 애플리케이션 사이의 상호 작용을 현격하게 간소화합니다. 많은 대형 시스템들이 다른 시스템에 쉽게 접속할 수 있도록 XML 인터페이스를 추가하도록 수정되고 있습니다.
- **웹 기반 사용자 인터페이스 생성.** 사용자의 데스크톱 PC에 모든 애플리케이션이 설치되어 있기를 요구하는 것은 대부분의 경우 현실적이 아닙니다. 그래서 많은 개발자들은 사용자의 데스크톱 PC의 웹 브라우저를 중앙 애플리케이션 서버에서 현재 실행되는 애플리케이션의 인터페이스로 사용합니다. 이는 운영 및 보안 측면을 크게 향상시킵니다. 트랜잭션 중심의 애플리케이션에 있어서는 성능을 향상시킬 수도 있습니다.
- **새로운 비즈니스 프로세스 지원을 위한 비즈니스 로직 확장.** 대형 애플리케이션의 핵심 트랜잭션 논리는 거의 변하지 않지만, 이전에 수작업으로 처리하였던 새로운 프로세스들이 시스템에 추가되어야 할 수도 있습니다.

따라서, 대다수 개발자들의 초점은 기존 코드를 유지한 상태에서 필요 부분만 추가하는 것입니다. 이것은 개발도구가 기존 시스템을 변화시키지 않고 컴파일 해야 함을 의미합니다. 또한, 새로운 개발 작업은 기존 코드의 대규모 변화를 최소화하며 수행되어야 함을 의미합니다.

개발자들이 C++ 언어에 바라는 것은 무엇인가?

윈도우 개발자들은 여러가지 이유로 C++를 선택합니다. 다음과 같은 이유들을 들 수 있습니다.

- **시스템에 대한 세부적인 제어.** C++보다 하드웨어 더 저수준으로 제어할 수 있는 것은 어셈블러 뿐 입니다. C++의 완전한 유연성으로 인해 PC와 그 작동에 대한 거의 전적인 제어를 할 수 있습니다.
- **높은 성능.** C++ 언어는 고성능 실행 코드를 염두에 두고 설계되었고 수십 세대에 걸쳐 최적화된, 대단히 성숙한 컴파일러를 보유하고 있습니다. 프로세서들도 C++ 프로그램의 실행을 감안하여 설계되어 왔습니다.
- **ANSI/ISO/IEC 표준 기반 개발.** C와 C++ 언어는 어떤 한 공급사에 의해 제어되는 언어가 아니므로, 따라서 개발자들은 자신의 프로그램이 표준에 호환되는 차세대 컴파일러에서도 실행될 것임을 확신할 수 있습니다.
- **짧은 대기 시간.** 많은 시스템들은 사용자 인터페이스, 객체 로직 및 의료, 증권, 군사 및 제조 솔루션과 같은 전용 하드웨어의 경우에 대해 빠른 응답을 필요로 합니다.
- **다른 시스템으로의 이식성**. 현재 사용되고 있는 거의 모든 하드웨어 플랫폼은 C/C++ 컴파일러를 사용할 수 있습니다. 이것은 다른 하드웨어를 사용하는 경우에 있어서도 C와 C++ 애플리케이션의 작성과 디버깅에 대한 투자를 다시 할 필요가 거의 없음을 의미합니다.
- **완벽한 메모리 제어권.** 가상 머신을 사용하는 언어와는 달리, C와 C++ 개발자들은 스스로 메모리를 관리합니다. 이것은 디자인 패턴에 직접 매핑되는 복잡한 데이터 구조를 구축하는 데에 있어 상당한 유연성을 제공합니다. 또한, 백그라운드에서 실행되는 외부 메모리 관리자가 없으므로 최종 프로그램의 속도 역시 향상됩니다.
- **윈도우와 리눅스 API에 대한 직접적인 액세스.** C와 C++ 개발자에게 익숙한 개발 기술과 스타일을 사용합니다.
- **양보 없는 생산성.** C++는 객체 지향적이므로 잘 디자인된 클래스 프레임워크는 개발 속도를 향상시키는 다양한 종류의 고수준 객체를 제공합니다. C++ 표준 템플릿 라이브러리(STL)는 다양한 일반적인 데이터 구조와 알고리즘을 제공합니다.
- **C 언어 사용 가능성.** 수많은 라이브러리가 C 언어로 작성되어 있고, 극도로 효율적인 프로그램을 생성할 수 있습니다.

이러한 이유로 C++은 다양한 범위의 애플리케이션을 위해 선택되어 왔습니다.

- **대량의 데이터를 빠르게 처리하는 애플리케이션.** 특정한 단일 애플리케이션을 위한 데이터 구조를 생성하는 능력으로 인해 C++는 자연스레 선택됩니다. 신속함과 효율성이 필요한 실시간 시스템의 경우에 있어 C++은 사실상 표준 언어입니다.
- **다양한 하드웨어에서 실행되어야만 하는 대형 비즈니스 애플리케이션.** C와 C++의 이식성은 서로 다른 시스템간에 이식되어야 하는 고성능 시스템에는 이상적입니다. POSIX와 등에 대한 지속적인 주도권은 이런 개발에 시사점을 제공합니다.
  - **하드웨어를 제어하는 애플리케이션**. 윈도우 API를 통하여 작업할 때조차 시간상 제약은 하드웨어를 처리하기 위해서는 애플리케이션을 매우 세심하게 디자인해야 함을 의미할 수 있습니다. C 또는 C++와 같이 저수준 액세스와 제어 기능을 가진 언어는 제조업 공장 제어 시스템, 게임 및 멀티미디어 시스템 등에 이상적입니다.

다음과 같은 다른 많은 업계에서도 C++의 능력을 활용해왔습니다.

- **통신업계**에서는 빠른 속도를 요구하는 네트워크 라우팅 및 레이팅 시스템과 같은 OSS 어플리케이션에 널리 사용되고 있습니다. C와 C++로 작성된 애플리케이션은 서로 다른 하드웨어 기술 및 소프트웨어 플랫폼에서 사용할 수 있으므로 기존 코드에 대한 투자 비용을 보존할 수 있습니다. 언어에 대한 정의된 표준이 있으므로, 커스텀 하드웨어로의 이식도 용이합니다. C와 C++는 모두 AT&T Bell Labs 에서 개발한 것으로서, 이 분야에서의 위상은 확고합니다.
- **금융업계**에서 사용되는 시스템의 개발자에게 있어서 C++ 애플리케이션의 높은 성능은 시장의 변동 상황에 연동할 수 있고 대규모의 데이터를 처리할 수 있음을 의미합니다.
- **제조업계**는 로봇 공학과 제어 공정의 개발이 필요하며, 이것은 C/C++와 같은 저수준 시스템 개발 언어만이 가능합니다.

대규모의 복잡한 데이터를 신속하게 처리하는 능력은 많은 IT 부서에서 C++를 비즈니스 애플리케이션을 위한 범용 프로그래밍 언어로 사용하고 있음을 의미합니다. 어떤 사람들은 기획 단계의 클라이언트 사이드 개발을 위하여 볼랜드 Delphi와 마이크로소프트 비주얼 Basic 와 같은 RAD (Rapid Application Development) 도구를 사용하는 혼성(hybrid) 접근 방식을 취합니다. C++Builder는 프로그래밍 언어로서의 C++와 기획 단계의 클라이언트 사이드 개발을 위한 RAD 양쪽 모두를 제공합니다. 애플리케이션에 웹 인터페이스를 제공해야 하는 필요성 때문에 많은 사람들이 웹 프리젠테이션을 위한 Active Server Pages(ASP) 또는 자바Server Pages (JSP)와 같은 기술을 수용하게 되었습니다. 수많은 상황에서 C 또는 C++는 여전히 핵심 프로세싱 엔진으로 사용되고 있습니다.

C++의 인기는 여전합니다. 명목상의 이식성 있는 환경인 자바,를 비롯하여 비주얼 베이직과 같은 RAD 도구 등의 등장에도 불구하고, 설문조사에 의하면 C와 C++는 광범위한 개발자 기반을 가지고 있으며 계속 성장중이라고 합니다. 예를 들어, 2001년에 실시된 IDC의 설문조사에 의하면 C++ 프로그래머는 자바 개발자의 3배에 이르고, C++는 여전히 세계 제일의 언어입니다. 다른 조사에 의하면 윈도우 플랫폼에만 약 170만명의 C++ 개발자들이 있다고 합니다.

비주얼 스튜디오.NET: 관리 C++와 CLR

윈도우 플랫폼상에서의 주요 C++ 공급자인 볼랜드와 마이크로소프트는 웹 서비스와 인터넷 개발의 중요성을 인식하고 이러한 새로운 기술을 이용한 개발을 지원할 수 있도록 툴들을 업그레이드하였습니다.

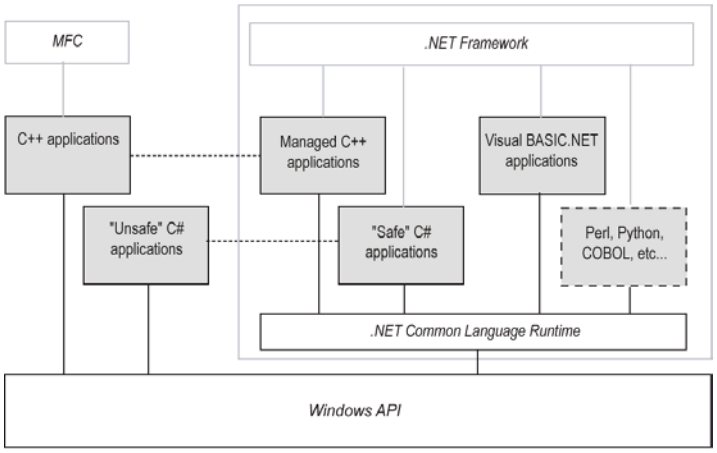

그림 1. *비주얼 C++.NET 애플리케이션*

마이크로소프트의 경우, 비주얼 스튜디오.NET은 개발에 대한 새로운 접근방법을 소개하고 있습니다. 두 가지 새로운 기술이 포함되어 있는데, 그것은 운영 시스템과 작동하기 위한 표준 시스템 호출 라이브러리인 .NET 프레임워크와 프로그램 실행을 위한 인터프리터인 CLR (Common Language Runtime)입니다.

.NET 프레임워크에는 사용자 인터페이스, 웹 서비스와의 작업 및 웹 서버와의 통신을 위한 고수준의 기능들이 포함되어 있습니다. CLR에서 실행되는 프로그램은 런타임 라이브러리로 .NET 프레임워크를 이용하여 COM과 윈도우 API를 캡슐화합니다.

윈도우 애플리케이션의 보안과 관리를 둘러싼 많은 문제를 피하기 위하여 비주얼 스튜디오.NET을 이용하여 생성된 프로그램들은 기본적으로 더 이상 원시 이진 애플리케이션으로 실행되지 않습니다. 그보다는, 중간 언어(MSIL ; Microsoft Intermediate Language)로 CLR에서 동작되도록 컴파일된 후에 CLR위에서 실행됩니다.

마이크로소프트는 윈도우 개발자들이 CLR의 혜택을 많이 볼 수 있다고 주장합니다. CLR은 메모리 요구를 자동으로 관리함으로써 프로그램 불안정의 주요 원인이 되는 메모리 누출과 버퍼 오버런을 제거합니다. 그리고, 라이브러리의 버전을 확인함으로써 DLL 버전의 불일치로 인한 문제들을 제거합니다. 또한, 프로그램의 인증 여부를 검증하는 데에 사용되어 신뢰성을 높여 줍니다.

이 모델은 자바의 "sandbox"와 매우 유사하며, 자바와 같이 이런 방식으로 실행되는 프로그램은 잠재적으로 위험한 방식으로 실행되는 것으로부터 보호를 받습니다.

그러나, 자바와는 달리, 마이크로소프트는 여러 프로그래밍 언어가 마이크로소프트 중간 언어(MSIL: Microsoft Intermediate Language)로 컴파일되는 것을 직접적으로 지원합니다. 처음 출시된 비주얼 스튜디오.NET에 포함된 언어는 비주얼 베이직.NET, 비주얼 C++.NET, J#과 새로운 언어인 C#입니다.

모두 MSIL 형식으로 컴파일되고, .NET 프레임워크를 사용할 수 있습니다. 모든 언어가 런타임 라이브러리의 모든 기능을 동일한 방법으로 액세스할 수 있다는 사실은 윈도우 프로그래밍을 간단하게 하는 주요 요인입니다.

그러나, 모든 언어가.NET 프레임워크와 CLR의 제한 사항을 준수해야 하므로, C++와 같이 유연한 언어에 있어서는 일종의 타협이라고도 할 수 있습니다.

C#이란 무엇인가?

.NET 프레임워크는 새로운 C# 언어로 작성되어 있습니다. C#은 C++보다 사용하기 쉽고 생산성을 높이는 반면 비즈니스 애플리케이션을 위한 유연성이 충분한 객체 지향적 언어로 개발되었습니다.

C#의 기본적인 언어 요소는 C++와 비슷합니다. 많은 특징들은 코드의 가독성을 높이도록 고안되었고, 따라서 유지보수가 더 용이합니다. 헤더 파일은 더 이상 사용되지 않지만 메인 소스의 일부로 들어가고, "->"와 "::" 연산자는

간단하게 "."로 바뀌는 등의 변화가 있었습니다.

다른 특징들은 코드를 더 논리적이 되도록 고안되었습니다. 예를 들어, switch 구문은 더 이상 각 각 절마다 명시적인 "break"를 필요로 하지 않습니다. 정수 타입은 더 이상 Boolean 형식으로 묵시적 형변환이 되지 않으므로, 코드를 더 읽기 쉽고 이해하기 쉽게 해 줍니다.

그러나, C#의 특성 가운데 C++ 개발자에게 덜 매력적인 면도 있습니다. C++와 달리, C#은 진정한 다중 상속을 지원하지 않습니다. 그 대신, 클래스는 다중 인터페이스를 구현할 수 있으며, 이것은 순환 참조의 복잡성을 피하려는 의도이지만 언어의 유연성을 감소시킵니다.

또한, C++와 달리, C#은 핵심 언어의 일부로서 포인터를 지원하지 않습니다. 그 대신, 객체와 그 인스턴스를 참조하기 위하여 "reference" 데이터 형식을 사용하는데, 이것은 CLR 메모리 관리자가 관리합니다. 이것은 프로그래밍 실수로 인한 시스템의 불안정을 예방하지만, 복합 데이터 구조를 가지고 작업하기 어렵게 만듭니다.

C#은 소위 "불안전한 코드"에서 포인터를 지원하는 기능이 있습니다. 불안전한 코드 블록은 전통적인 C 스타일의 포인터 수식을 실행하는 문장을 포함할 수 있습니다. 이것은 하드웨어를 직접 조작하거나 더 복잡한 알고리즘에 대하여 때로 유용합니다. 그러나, CLR이 메모리를 안전한 코드 블록에 있는 데이터에 대하여 할당하기 때문에 안전한 코드 불록과 불안전한 코드 블록 사이의 데이터 교환은 어렵습니다. 이런 이유로 불안전한 쪽의 코드를 프로그램하는 일은 더 어렵습니다.

C#의 고수준 기능은 .NET 프레임워크에서 제공합니다. 여기에는 COM 객체의 C# 클래스에 대한 매핑과, 구조에 대하여 객체에 질의하는 오버헤드의 제거가 포함됩니다. 나아가 XML, SOAP와 같은 프로토콜에 대한 직접적 통합을 통해 개발자들이 웹 서비스를 통해 노출된 외부 객체와 작업하는 것을 이전의 마이크로소프트 언어에서보다 더 간단하도록 만들었습니다.

새로운 개발에 있어서 C#은 그래픽 폼 디자이너, 웹 서비스의 용이한 사용, 메모리 보호 등의 강점을 지니고 있습니다. 그러나, C++ 개발자에게 있어서 이것은 덜 유연한 새로운 언어로의 이전과 최종 프로그램의 성능 저하를 감수해야만 함을 의미합니다.

C#은 .NET 프레임워크와 매우 밀접하게 연관되어 있습니다. .NET 프레임웍은 비 Intel구조에 대한 외부의 개발이나 지원이 없는 윈도우만의 폐쇄적이고 소유권이 걸린 부분입니다. 결국, C#를 사용하는 개발자들은 윈도우 서버 기술을 사용하는 데 묶여버릴 수밖에 없습니다.

마지막으로, C# 자체는 다른 RAD 도구에 비하여 단점이 있습니다. 새로운 것이기 때문에 자바, 볼랜드 Delphi 또는 볼랜드 C++Builder와 같은 언어와 관련된 지식이나 기반이 없습니다. 처음 출시된 제품으로는 상당히 포괄적이기는 하지만 .NET 프레임워크는 자바 Swing이나 볼랜드 VCL (Visual Component Library) 수준으로 클래스 라이브러리가 성숙하지는 않았습니다. 끝으로, 마이크로소프트 C#은 윈도우에서만 실행되고, 애플리케이션을 다른 운영 체제로 이동시킬 수 없습니다.

비주얼 C++.NET과 관리 C++이라는 타협안

비주얼 C++는 처음부터 MFC(Microsoft Foundation Classes)를 중심으로 디자인되었습니다. MFC는 윈도우 API에 대한 래퍼(wrapper)로서, C++ 개발자가 윈도우를 구성하는 30,000개 이상의 API를 가지고 작업하는 것을 더 쉽게 만드는 고수준 객체를 제공합니다.

이것은 비주얼 C++ 개발자에게 주 라이브러리이지만, 비주얼 스튜디오.NET에 포함된 버전에는 새로운 MFC 기능이 거의 없습니다. 웹 서비스와 그래픽 폼을 생성하는 기능은 관리 C++ 환경에서만 사용할 수 있습니다.

웹 서버 애플리케이션이나 웹 서비스를 생성하기 위하여 .NET을 사용하는 개발자는 관리 C++(Managed C++)로 작성된 새로운 객체를 디자인해야 합니다. 개발자는 유연성이 줄어든 C++ 구현과 CLR에서의 실행

으로 인한 속도 감소를 감수해야 합니다. 관리 C++ 섹션은 윈도우 API의 일부를 호출할 수도 있지만, 애플리케이션 내의 MFC 클래스들에는 액세스할 수 없습니다. 비관리 C++과 관리 C++에 있는 클래스 사이에는 상속관계가 없습니다.

비관리 C++와 관리 C++ 사이의 메모리 관리는 특히 어렵습니다. 왜냐하면, 관리 C++에는 프로그래머의 통제를 받지 않고 메모리를 언제라도 정리할 수 있는 고유한 메모리 관리자가 있기 때문입니다. 두 개의 시스템을 합치는 것은, 시스템에서 변경하지 않는 것이 바람직한 비관리코드의 변경을 포함하여 세심한 주의가 필요합니다.

볼랜드 C++Builder

C와 C++ 개발자를 위한 이상적인 프로그래밍 환경은 기존 코드에 대하여 새로운 윈도우 기술을 적용하는 것입니다. 그러면 언어 기능의 절충이란 없을 것이며, 최근의 표준에 가장 근접하게 부합될 것입니다. 특히, MFC를 사용하는 기존 코드와 시스템은 그대로 보존될 것입니다. 또한, 윈도우 뿐 아니라 리눅스와 같은 다른 플랫폼에서도 작동하는 애플리케이션을 만들 수도 있을 것입니다.

사용자 인터페이스을 위해서든 웹 서버를 위해서든 애플리케이션을 위한 프리젠테이션 레이어를 빠르게 생성하는 능력은 C++ 언어 환경에서 모두 가능해야 하며, RAD 기술을 사용할 수 있어야 합니다.

2001년 12월에 실시된 평판 있는 Netcraft 조사에 의하면, 일반 인터넷 웹 사이트의 57%가 공개 소스인 Apache 서버에서 실행되는데 반하여 윈도우는 31%에 불과하였습니다. Apache의 높은 신뢰성과 성능 뿐만 아니라, 리눅스, 윈도우 및 솔라리스에서 실행되는 능력은 Apache가 탁월한 선택이라는 것을 말해줍니다. 웹 서버 애플리케이션을 생성해야 하는 C++ 개발자들은 여러 운영 시스템에서 다양한 서버 애플리케이션에 대하여 쉽게 디버깅할 수 있는 이식성 있는 코드를 선호합니다.

볼랜드 C++Builder 6는 이러한 조건을 모두 만족합니다. 이것은 C 및 MFC 기반 C++ 애플리케이션을 작업하기 위한 종합적이고도 유연하며 강력한 개발 환경입니다. C++Builder 6는 최신 C++ 언어 표준을 지원하고, 대단히 고속으로 최적화된 코드를 생성합니다.

C++Builder 6는 볼랜드 컴포넌트 라이브러리를 사용하는 RAD 환경을 포함하며, C++ 언어의 강력함을 손상시키지 않으면서도 생산성이 높은 진정한 비주얼 시스템입니다. COM 객체와 기존 윈도우 API를 C++Builder 6에서 직접 사용할 수 있으며, 관리 C++로 옮길 필요가 없습니다.

C++Builder 6는 윈도우 IIS(Internet Information Server) 및 Apache 서버를 위한 웹 서버 애플리케이션 개발을 가능하게 함으로써 선도적인 서버 플랫폼과의 상호 연동을 보장합니다. C++Builder는 리눅스에서 프로그램을 실행시키는 기술을 포함하고 있기 때문에 이런 웹 서버 애플리케이션은 가장 인기 있고 지원을 많이 받는 웹 서버 시스템을 목표로 할 수 있습니다.

C++Builder는 쉽고 빠르게 웹 서비스 작업을 할 수 있는 능력이 뛰어나므로, 애플리케이션이 다른 시스템과 쉽고도 간단하게 통합되도록 해줍니다. 여기에는 복합적인 시스템을 조립하는 시간을 크게 줄일 수 있도록 여러 웹 서비스 인터페이스의 데이터를 변환하는 고수준 기능이 포함됩니다. 결정적으로, 이 웹 서비스 지원은 C++Builder 클래스 라이브러리의 최신 버전에 포함되어 있습니다.

이런 고수준 프레임워크와 도구뿐만 아니라, 이 개발 환경은 웹 서버 애플리케이션과 웹 서비스 양쪽 모두를 위한 강력한 분산 디버깅을 지원합니다. 이 기능은 대규모의 분산 시스템 개발을 도와줍니다.

이러한 최신 표준들 외에, C++Builder는 DCOM과 CORBA 명세를 지원함으로써, 아직 웹 서비스를 이용하지 않는 다른 플랫폼에서 구동되는 애플리케이션과의 상호 연동을 용이하게 합니다. 게다가, C++Builder는 이러한 다른 시스템으로부터 웹 서비스를 생성하는 "래퍼(wrapper)"를 생성하기가 아주 편리합니다.

새로운 비즈니스 로직의 개발을 더욱 빠르게 하기 위해, 이 RAD 환경은 데이터베이스에 직접 액세스하는 강력한 컴포넌트 프레임워크를 가지고 있습니다. 비주얼 디자이너는 매력적인 사용자 인터페이스를 쉽고 빠르게 만들수 있도록 해주며, 사용자 수용성에 도움이 됩니다. 이것은 기존의 코드와 함께 사용할 수 있습니다.

마지막으로, 마이크로소프트의 개발 시스템과는 달리, 볼랜드 C++Builder는 단일 소스 코드와 단일 크로스 플랫폼 프레임워크로 윈도우와 리눅스 양쪽 모두를 위한 시스템의 개발을 지원합니다. 이것은 더 넓은 범위의 대상 운영 플랫폼을 지원한다는 것 뿐만 아니라, 특정 공급사의 운영 시스템에의 의존없이 미래에도 시스템이 계속 운영될 것이라는 보증을 해줍니다.

이러한 빠른 개발 기능은 VCL(Visual Component Library) 및 그 윈도우/리눅스 크로스플랫폼용 사촌격인 CLX (Component Library for Cross-platform)에서 비롯된다는 점이 중요합니다. C++Builder에는 마이크로소프트 MFC 코드를 변환하는 위저드가 포함되어 있습니다.

올바른 개발툴의 선택

.NET과 .NET 프레임워크의 도입은 윈도우 플랫폼에서의 소프트웨어 개발에 대한 전망을 현격하게 바꾸어 놓고 있습니다.

마이크로소프트가 C와 C++ 개발자들에게 웹 서비스를 위한 강력한 개발툴을 제공했지만, 이것은 .NET 프레임워크의 일부로서만 이용 가능합니다. .NET은 개발자들이 배포 환경으로 윈도우를 사용하도록 단단히 구속하고 있습니다.

반대로, 볼랜드 C++Builder는 C++ 개발자들에게 윈도우 98, 2000, XP 및 리눅스에서 작동하는 웹 서비스와 그래픽 UI 개발 기능을 제공하는 환경을 제공합니다. 볼랜드는 또한 앞으로 .NET 프레임워크에서의 배포도 지원한다고 발표하였습니다. 분명히 이것은 개발과 배포 플랫폼에 대한 훨씬 더 넓은 선택으로써, 선택의 자유를 넓혀줍니다.

C++Builder는 신뢰성 있고 포팅가능한 코드 등 .NET의 많은 핵심적인 이점을 제공합니다. WSDL, XML 및 SOAP에 대한 네이티브하고 투명한 지원을 통해 웹 서비스 개발을 지원합니다. 웹 서비스 개발을 위한 탁월한 생산성을 포함합니다. 무엇보다도, ANSI 표준 C++ 언어의 저수준 기능을 상실하지 않고 이러한 기능들을 제공하고 있습니다.

새로운 기술을 수용해야 하는 C와 C++ 개발자들은 어떤 전략을 선택할지 신중하게 결정해야 합니다. 생산성을 기대하는 C++ 개발자는 언어를 바꿀 필요가 없습니다. C++Builder는 진정한 C++의 RAD를 가능하게 함으로써 과거를 포기하지 않고도 개발자들을 미래로 안내해 줍니다.

C++Builder를 평가해보려면 다음 주소를 방문해보십시오.
http://www.borland.co.kr/cbuilder

Borland®

100 Enterprise Way
Scotts Valley, CA 95066-3249
www.borland.com | 831-431-100C