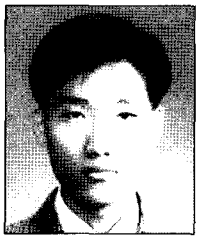
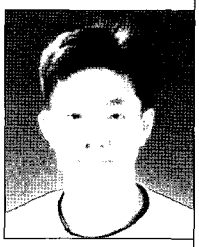
공학계산용 프로그래밍언어

Programming Languages for Engineering Computation

김 두 기* 김 동 현** 구 기 영***

*군산대학교 토목환경공학부 교수
**군산대학교 해양시스템공학전공 교수
***한국과학기술원 건설 및 환경공학과 박사수료

1. 서 론

1950년대 FORTRAN이 개발됨으로써 본격적으로 시작된 공학계산용 프로그래밍언어는 지난 반세기동안 컴퓨터 관련 산업의 발전과 함께 급속한 진보를 거듭해왔다. 간단한 공학연산만이 가능했던 초창기의 프로그래밍 언어와는 달리, 현대의 프로그래밍언어는 복잡하고 대용량인 수치연산을 매우 빠르게 처리할 수 있을 뿐만 아니라, 기호연산(symbolic computation), 물리량 계측(measurement), 신호처리(signal processing), 정보의 시각화(visualization) 및 인터넷을 통한 정보의 제공 및 교환 등을 가능하게 하였으며, 더 나아가 시공간적 제약을 뛰어넘는 유비쿼터스(ubiquitous) 환경을 만들어 가고 있다. 전산구조공학은 이러한 발전된 공학용 프로그래밍 언어를 바탕으로 복잡한 대규모의 작업을 보다 원활이 수행할 수 있게 되었다.

현대 전산구조공학에서 주로 사용되는 프로그래밍언어와 이를 통해 수행할 수 있는 주요한 작업은 다음과 같다. 전통적으로 사용되어온 FORTRAN은 잘 알려진 바와 같이 대규모의 수치연산을 매우 빠르고 안정적으로 수행할 수 있는 언어이며, Visual C++/Visual Basic은 GUI (graphic user interface)를 사용하여 사용자와 상호작용을 반복적으로 요구하는 응용프로그램을 개발할 수 있으며, Maple이나 Mathematica는 기호연산을 수행하여 이론적인 유도과정을 편리하게 해주며, LABVIEW나 MATLAB은 물리량의 계측과 신호처리를 수행할 수 있으며, 최종적으로 가공된 결과는 인터넷 언어를 통하여 많은 사용자들이 접속과 자료공유가 가능할 수 있도록 웹(web)에 게시될 수 있다.

본고에서는 제시된 각각의 프로그래밍언어를 간략히 소개하고, 각각의 특징을 비교함으로써, 공학적 문제에 당면한 독자가 주어진 작업환경에서 어떤 프로그래밍언어를 선택하여 사용하는 것이 가장 적합한지에 대한 개략적인 가이드라인을 제공하고자 하였다.

2. 기본언어

본고에서 기본언어란 일반적으로 고급언어(high-level programming language)로 분류되는 프로그래밍언어로, (Visual) FORTRAN, C/(Visual) C++/C#, (Visual) BASIC 등을 말하며, 사용자는 당면한 공학문제의 종류와 성격에 따라서 그에 적합한 언어를 다양하게 선택할 수 있다. 여기서 프로그래밍언어를 선택할 때 고려해야할 중요한 고려요소는 프로그래밍언어에 관한 학습의 편의성, 코딩 및 디버깅의 편의성, 적절한 Math-Library의 가용성, 수치연산의 속도 및 안정성, 연산결과 시각화의 편의성, 더 나아가 개발된 코드의 재사용의 편의성(이식성) 등이 있다.

일반적인 공학 문제는 선형대수/미적분/미분방정식 등

의 문제를 해결하고 결과를 시각화하는 과정이라고 볼 수 있는데, 문제가 복잡하고 대규모의 문제라면 수치연산의 속도가 가장 중요한 측면이 되며, 통상 이에 가장 적합한 언어가 바로 FORTRAN이다. FORTRAN은 BASIC이나 C언어 계열에 비해서 수치연산에 초점이 맞추어져 개발된 프로그래밍언어로써, 빠른 연산속도가 특징이고 대용량의 문제도 안정적으로 다룰 수 있으며, IMSL™ Numerical Libraries를 포함한 Math-Library들이 널리 개발되어 있으며, 특히 지난 수십 년간 다양한 공학적 문제에 적용되어 온 많은 코드들이 가용하다는 장점이 있다.

한편, 수치적 문제를 포함한 공학의 일반적인 문제 외에도 연산의 결과를 사용자에게 시각화하거나, 사용자의 요구를 받아들여 그에 적합한 결과를 다시 보여주는 과정을 지속적으로 반복하는 문제의 경우에는 GUI가 필수적으로 요구되고, 이를 위하여 통상 Visual C++이나, Visual Basic등을 선택할 수 있다.

다음에서는 각각 언어의 주요한 특징을 약술하였다.

2.1 FORTRAN (The IBM Mathematical FORmula TRANslating System)

FORTRAN은 범용의 순차적 컴퓨터 언어로써 자연과학과 공학 영역에서의 수치계산을 위해 설계되었다. 1950년대에 IBM에서 자연과학과 공학의 수치연산문제에 적용하기 위하여 개발하였는데, FORTRAN은 개발이후 현재까지 지난 50년 동안 기후예측과 유한요소 구조해석을 포함한 다양한 공학문제를 풀기 위한 프로그래밍언어로써 지배적인 위치를 차지해 왔다.

FORTRAN은 이전버전과의 호환성을 유지하면서 기능을 추가적으로 확장하면서 발전해 왔는데, FORTRAN 77은 문자열 연산 기능을 포함하였고, FORTRAN 90은 동적할당배열 (dynamically allocated array), module based programming, objected-based programming을 지원하였고, FORTRAN 2003은 object-oriented programming과 generic programming을 지원하게 되면서, 기본적인 수치연산의 기능 외에도 현대 프로그래밍언어어의 특징인 데이터 추상화, 모듈화 기능들을 포함하게 되었고, 이를 통해 코딩의 편의성, 이식성 등을 개선시켜왔다.

FORTRAN은 자연과학과 공학의 대부분의 분야에서 오랫동안 사용되어 왔는데, 이는 많은 공학적 문제를 해결하기 위한 FORTRAN 코드가 이미 전 세계적으로 작성되어 왔다는 것을 의미하며, 많은 경우 기존의 코드를 이용하여 빠르고 신속하게 직면한 문제를 해결 할 수 있다는 장점을 가지고 있다. 더욱이 FORTRAN은 계산량이 매우 큰 대규모 공학계산을 슈퍼컴퓨터를 사용하여 해결하는데 있어서 독보적인 위치를 차지하고 있다. 예를 들어 1993년에는 기존 FORTRAN의 성능을 더욱 향상시키기 위하여 High Performance Fortran Forum(HPFF)이 조직되었는데, 이들은 고성능 Fortran의 규약을 제정하기 위한 산학연 협동 조직으로써, 1994년에 version 1을 1997년에는 version 2를 발표하였으며, 현재에도 새로 개발된 슈퍼컴퓨터의 성능을 측정하기 위한 기준으로 FORTRAN 연산속도를 사용하고 있다.

2.2 C/C++/C# 및 Visual C++

C는 1970년대 초에 Unix OS(operating system)를 위해서 개발된 범용의 절차적 컴퓨터 언어로써 현재에는 모든 OS에 적용 가능하며, 범용 프로그래밍언어 중에서 전 세계적으로 가장 널리 사용되고 있다. C의 확장으로써 1983년 개발된 C++은 데이터 추상화(data abstraction), 객체지향 프로그래밍(object-oriented programming), generic programming 등의 현대적인 프로그래밍 기법을 최초로 도입한 프로그래밍언어이며, C#은 모든 것을 객체로 취급하는 컴포넌트 프로그래밍언어로 2000년 6월 마이크로소프트가 닷넷(.NET) 플랫폼를 위해 개발하였다. 웹을 통해 정보와 서비스를 교환하고, 개발자들이 이식성 높은 응용프로그램들을 만들어 낼 수 있게 고안되었는데, 대대적인 코드의 수정 없이도 하나 이상의 OS에서 사용될 수 있는 응용프로그램들을 만들어낼 수가 있다.

이러한 프로그래밍언어는 주로 일반적인 프로그램 개발 분야에서 활발히 사용되어 왔다. 자연과학과 공학의 수치적 문제에 초점이 맞추어져 개발되지 않아서 수치연산속도가 느리고 수치적 안정성이 나쁘며, Math-Library등이 풍부하게 개발되어 있지 않는 등의 이유로 공학계산용으로 널리 사용되고 있지는 못한 것이 현실이다. 그러나 최근에 개발된 C/C++/C# 컴파일러(compiler)들은 FORTRAN에 버금가는 수치연산속도를 보여주고 있다고 알려져 있다. 최근에는 FORTRAN과 C언어의 장점을 합쳐서, 수치연산의 핵심코드는 FORTRAN으로 작성하고 사용자 GUI는 Visual C++/C#으로 작성하는 mixed-language programming이 많이 선택되어 사용되고 있다.

2.3 BASIC 및 Visual BASIC

BASIC(beginner's all-purpose symbolic instruction

code)은 1963년에 교육용 프로그래밍언어로 개발되었는데, 인터프리터 형식으로써 코드작성이 편리한 장점이 있으나, 연산속도가 느리고 기능상 제약이 많은 단점으로 인해 공학용 프로그래밍언어로 사용되기 힘들었다. 그러나 최근 알려진 Visual BASIC은 BASIC이 가지는 코드작성의 용의함을 유지하면서 실행속도도 개선되었다. 특히 간단하고 편리하게 GUI 응용 프로그램을 작성할 수 있다는 장점을 가지므로, 작은 규모의 공학계산용 프로그램을 작성할 때에는 매우 유리하다.

3. 응용언어

공학분야에서 많이 사용되고 있는 응용언어로는 Mathematica, Maple, Matlab 등이 있다. 이들 응용언어들은 C나 FORTRAN과 같은 개발자 전용 언어(본고에서의 기본언어)가 가지는 사용상의 단점을 극복하고, 일반 사용자들도 간편하고 쉽게 코딩(coding)하고, 계산결과를 확인할 수 있도록 개발되었다. 응용언어들은 대규모 공학문제에서는 그 계산속도가 느리다는 단점이 있으나, 사용성과 응용성에서의 획기적 장점으로 인해 자연과학, 통계학, 공학 등의 분야에서 폭 넓게 이용되고 있다.

이들 응용언어가 가지는 공통적인 특징들을 간략히 살펴보면 다음과 같다.

첫째, 별도의 컴파일링(compiling) 과정 없이 코딩 결과를 바로 확인할 수 있다. 이것은 기존의 언어들에서 필수적으로 거쳐야하는 컴파일링, 링킹(linking) 등의 다소 지루한 과정을 생략하고, 곧바로 결과를 확인할 수 있는 기능으로, 사용자의 입장에서는 시간을 매우 효율적으로 사용할 수 있는 기회를 제공한다.

둘째, 기호연산이 가능하다. 미분방정식, 대수방정식 등과 같이 복잡한 수식전개가 필요한 경우 이들을 간단히 수학적으로 표현한 후 solver를 구동시킴으로써 그 해를 얻을 수 있으므로 매우 복잡한 비선형방정식을 접하는 경우나 혹은 수학적 유도가 용이치 않은 경우 해결과정에 소요될 수 있는 많은 노력을 절감시켜 준다.

셋째, 방대한 수량의 내장 및 외장함수를 제공한다. 물론, 기본언어인 C나 FORTRAN의 경우에도 그 내장함수가 다양하지만, 응용언어들은 길고 복잡한 알고리즘이 소요되는 프로세스를 간단한 함수가 대신하므로 그 장점을 특별히 부각시킬 수 있다. 이것이 가능하게 된 것은 각 분야의 저명한 학자들이 프로그램의 제작과정에 참여할 수 있는 기회가 제공됨으로써 최신의 함수들을 매우 빠른 속도로 제공해주기 때문이다. 이를 위해 정기적인 Workshop 등을 통해 다양한 분야의 연구자들이 각자의 연구결과를 이들 응용언어의 형태로 공유할 수 있는 장이 마련되어 있기 때문이다.

넷째, 해석결과를 가시화할 수 있는 그래픽기능이 강력하다. 기본언어에서는 프로그램, 컴파일, 결과정리, 가시화의 단계별 작업이 구분되어 있지만, 응용언어에서는 이러한 모든 작업이 한 번에 가능하다. 특히, 최근에 제공하고 있는 그래픽 결과들은 Origin, Grapher 등을 사용한 별도의 그래픽작업이 필요가 없을 정도로 매우 높은 수준에 올라와 있다.

다섯째, 기존 언어 등과의 인터페이스가 가능하다. 최근에는 Java, Visual Basic, Visual C++ 등의 언어와 공유가 가능하도록 인터페이스를 개발하여 출시하고 있는 추세이다.

3.1 Mathematica

개별 언어에 대해 좀더 자세히 알아보면 우선, Mathematica는 Wolfram research(http://www.wolfram.com) 사에서 개발한 것으로 대수, 미적분, 통계 등 수학적 연산식의 해법에 중점을 두고 있으며, 기호연산의 강력한 기능을 갖추고 있다. 또한 노트북(notebook) 기능을 갖추고 있어 수식, 그림 및 문자열이 포함된 문서작성을 가능하게 해 준다. 그림 1은 노트북을 이용한 문서작성의 한 예를 보여주고 있고, 그림 2는 식 (1)의 문자열 방정식의 해석과 식 (2)의 적분을 수행한 결과를 보여주고 있다. 해석하고자 하는 문제를 대괄호로 묶고 Solve, Integrate와 같은 명령어 앞에 씀으로써 간단하게 결과를 얻을 수 있다.

$$x^2+x+\alpha \tag{1}$$

$$I=\int\{\sqrt{x}\,\sqrt{1+x}\}dx \tag{2}$$

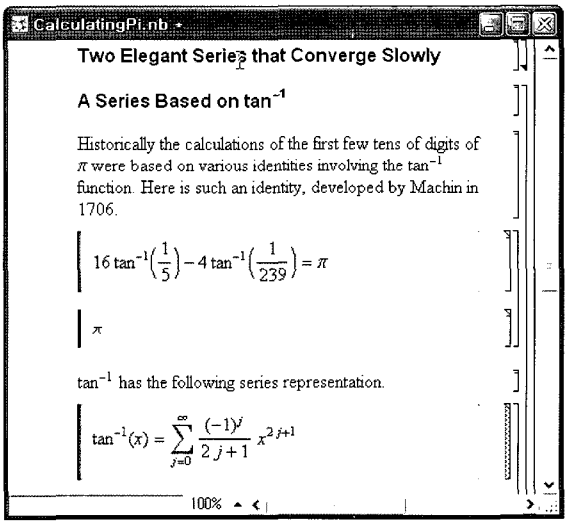


그림 1 Mathematica의 노트북 기능

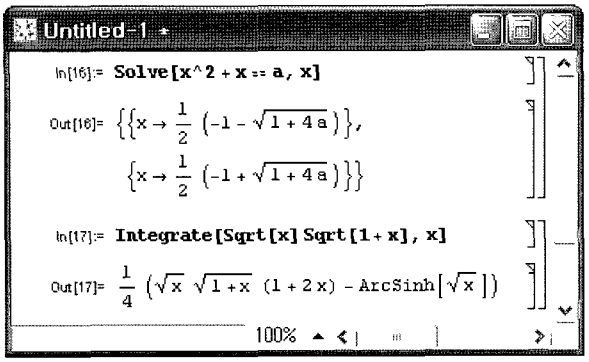


그림 2 Mathematica의 연산사례

3.2 Maple

Maple은 Maplesoft(http://www.maplesoft.com)사에서 개발하였으며, Mathematica가 가지고 있는 대부분의 연산 능력을 보유하고 있다. 특징적인 것으로는 문서작성(documentation) 기능이 매우 강력하여, 그림 3과 같이 Mathematica보다 더욱 섬세한 문서편집이 가능하다. 또한, 강력한 code generation 기능으로 Maple에서 작성한 모듈을 C, Fortran, Java, Matlab, Visual Basic 등으로 변환할 수 있다. 최근에는 Matlab과의 연계를 강화하기 위해 Toolbox for Matlab를 출시했고, 미분방정식을 이용해 Simulink block을 손쉽게 만들어 주는 Blcok builder를 발표한 바 있다.

3.3 Matlab

Matlab은 Mathworks(http://www.mathworks.com)사에서 개발한 것으로 Mathematica 및 Maple과 같이 기본적 연산을 대부분 포함하고 있으며, 문서작성 기능은 최근

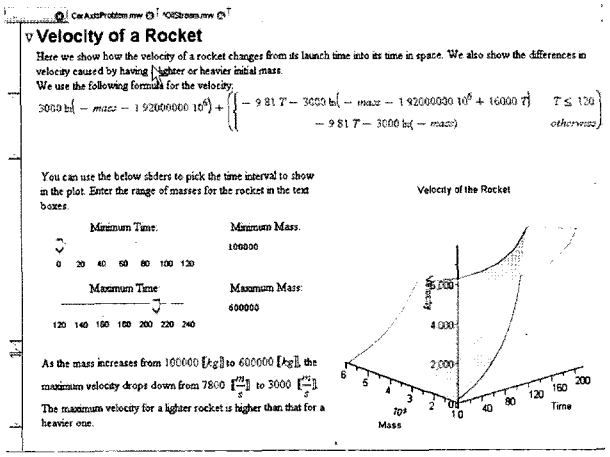


그림 3. Maple의 문서작성 화면

들어 그 기능이 강화되고 있다. Mathematica 및 Maple과 다른 Matlab만의 특징을 나열하면 다음과 같다.

첫째, Simulink라고 하는 block diagram을 이용한 시스템 정의 기법이 있다. 이것은 복잡한 미분방정식을 일련의 block diagram으로 표시하여 신호의 흐름과 구성을 가시적으로 확인하고 해석할 수 있도록 하는 큰 장점을 가지고 있다. 특히, Simulink는 회로, 동적시스템, 그리고 제어시스템의 설계와 해석에 매우 유용하게 사용할 수 있다.

둘째, Toolbox라고 하는 외장함수를 이용할 수 있다. 이러한 방식은 공학의 각 분야별로 학술적 성과를 공유할 수 있게 하고 최신의 이론을 많은 연구자들이 함수형태로 공유할 수 있도록 하는 매우 강력한 기능이다. Neural Network, Fuzzy Logic, Wavelets, Genetic Algorithm 등은 각계의 저명학자들이 참여해 개발한 Matlab만의 강력한 Toolbox로서 최신공학기법을 누구나 손쉽게 활용할 수 있도록 해주고 있다.

셋째, GUI 프로그래밍을 활용하여 응용프로그램을 개발할 수 있다. GUI환경의 event based programming과 Matlab compiler를 이용하여 Matlab 플랫폼 없이도 단독실행이 가능한 배포용 응용프로그램도 개발할 수 있다.

기본언어인 C, Fortran, C++, Visual C++, Visual Basic 등은 종합적인 기능의 프로그램 개발에 초점이 맞춰진 언어들이기 때문에, 많은 경우에 학술연구 및 공학계산에는 적합하지 않을 수 있다. 무엇보다도 매우 빠른 연산속도를 자랑하는 기본언어들은 프로그램의 완성단계에서는 그 효용가치가 매우 높다고 할 수 있으나, 새로운 이론의 개발이나 수치해석 기법의 개발 등 학술적 측면에서는 프로그램의 신속하고 반복적인 수정과 작성한 프로그램이 주는 결과를 손쉽게 가시화할 수 있는 기능에 대한 요구가 높다. 따라서 이러한 목적에 충실한 Mathematica, Maple, Matlab 등과 같은 응용언어의 수요는 앞으로도 계속될 것이며, 과거에는 학술적 목적의 이용에만 국한되었

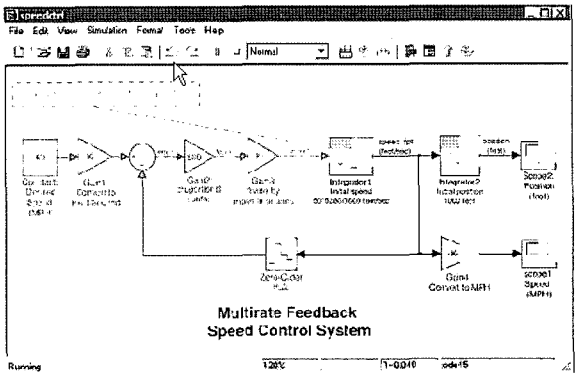


그림 4. Matlab Simulink 화면

던 이러한 응용언어들이, C나 FORTRAN으로의 변환을 가능하게 하는 Code generation기능이나 플랫폼에 무관하게 독립적으로 사용할 수 있게 해주는 stand alone application 기능들의 개발로 인하여, 개발용 언어영역으로의 입지를 확대해 나가고 있기 때문에 그 발전 가능성이 무궁무진하다고 볼 수 있다.

4. 인터넷 언어

2005년 상반기 정보화실태조사 결과, 우리나라의 인터넷 이용자수는 3,257만명(이용률 72.9%)이며, 초고속인터넷 보급률은 OECD국가 중 1위를 차지하였다. 즉, 인터넷이 없는 일상생활은 상상할 수도 없을 만큼, 인터넷은 광범위하고도 밀접하게 우리와 연결되어 있으며, 공학을 포함한 모든 학문분야에서도 그 적용범위와 분야가 나날이 증가하고 있다.

FTP(file transfer protocol) 및Telnet(telecommunication network)에서 시작된 공학계산에서의 이러한 인터넷의 활용은 WWW(world wide web)가 대중화됨에 따라 점차 큰 비중을 차지하게 되었으며, 관련 인터넷 언어들의 응용성은 무궁무진한 잠재력을 갖게 되었다. 본 기사에서는 통상적으로 공학계산에 응용되는 인터넷 언어들의 특징과 장단점에 대하여 알아보았다.

4.1 HTML

HTML(hypertext makeup language)이란 하이퍼텍스트(hypertext) 기능을 가진 문서를 만드는 인터넷 언어이다. 하이퍼텍스트란 하나의 텍스트(text)로 두는 것이 아니라 텍스트와 텍스트를 서로 연결시켜, 도시의 복잡한 도로처럼 상호 연결하여 접근 방식을 다양화시킨 텍스트를 말한다.

간단히 웹(web)이라고도 부르는 WWW는 HTML을 이용하여 작성한 홈페이지들이 인터넷을 통하여 서로 유기적으로 연결된 상태를 의미한다. 이러한 홈페이지들은 이 문서가 존재하는 서버(server)에 접속한 사용자(client)가 웹 브라우저(web browser)를 통하여 볼 수 있다.

HTML은 SGML(standard generalized markup language)에서 하이퍼텍스트를 강조하여 만들어진 언어이고, 별도의 컴파일러(compiler)가 필요치 않으며, 웹 브라우저에서 해석이 가능한 사용하기 쉬운 언어로 각광을 받고 있다. HTML에서 사용하는 명령어를 태그(tag)라고 하는데, 태그는 시작(<tag>)과 끝(</tag>)을 표시하는 2개의 쌍으로 이루어져 있다. 참고로 인터넷에서 웹을 통해 접근되는 대부분의 웹 페이지들(web pages)은 HTML로 작성되며, HTML은 문서 이외에도 이미지, 동영상, 소리 등을 다른 도구 없이 웹 브라우저가 해석할 수 있도록 지원해 준다.

4.2 CGI

HTML에 기반을 둔 많은 언어들이 출현하였으며, 이 새로운 언어들은 인터넷을 통한 웹의 활용을 더욱 다양하고 충실하게 만들고 있다. HTML은 정적 언어이기 때문에 제작함과 동시에 끝이 나버리는 페이지이다. 즉, 페이지를 수정할 땐 HTML을 편집기(editor) 등으로 수정하여 서버에 다시 올려야 한다. 이런 정적 언어인 HTML로는 지금의 거대한 웹 시장에서 자리를 잡기가 힘들다. 그래서 그 단점을 보완하기 위해 나온 것이 CGI, ASP, JSP, PHP 등 대화형 언어(또는 동적인 언어)이다.

CGI(common gateway interface)란 브라우저와 서버 그리고 응용프로그램간의 인터페이스를 의미한다. 즉, 서버는 브라우저로부터 서버에 전달된 사용자의 요구를 응용프로그램에 전달하고, 서버에서 응용프로그램을 수행한 후 그 결과를 다시 브라우저를 통해 사용자에게 되돌려 주는 표준적인 방법이며, 웹 HTTP(hypertext transfer protocol)의 일부이다.

CGI는 사용자의 요구가 서버로부터 응용프로그램으로 전달되고, 서버에서 응용프로그램을 사용하여 가공된 정보를 사용자에게 되돌려주는 단순하고 일관적인 기술이다. 이것은 응용프로그램을 작성하는 사람 입장에서는, 서버에서 어떤 운영체계를 쓰든 상관없이 그 프로그램이 운영될 것이라는 확신을 가질 수 있다는 것을 의미한다. 즉, 인터페이스가 일관되기 때문에, 프로그래머는 CGI 응용프로그램을 여러 가지 다른 언어로 작성할 수 있다. CGI 프로그램 작성에 가장 보편적으로 사용되는 언어로는 1) 인터프리터 언어: UNIX shell script, Perl(practical extraction and report language) 등, 2) 컴파일 언어: C/C++, visual Basic 등, 3)

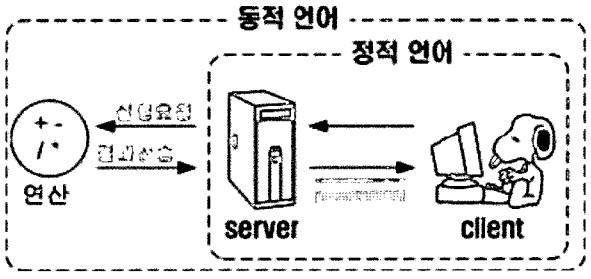


그림 5 Static and Dynamic Languages

컴파일 및 인터프리터 언어: JavaScript, VBScript 등이 있다.

Sun Microsystem사의 JavaScript는 표준 아스키 텍스트 스크립트(script) 형태로 직접 HTML문서 내에 프로그램되어 웹 브라우저에 의해 직접 실행된다. 즉, 사용자의 브라우저에서 실행되므로 진정한 의미의 CGI언어라고 할 수는 없으나, 클라이언트에서 데이터를 미리 처리하게 되므로 서버의 부담을 덜어주게 된다. Microsoft사의 VBScript는 JavaScript와 매우 유사한 특성을 지니며 주로 비디오나 사운드 파일을 처리해 주는 기능을 수행한다. 브라우저는 웹서버로부터 VBScript를 받는 동안 코드를 바로 해석, 컴파일 하여 실행된다. 웹클라이언트와 웹서버 환경에서 보면 펄은 웹서버에서, 자바스크립트는 클라이언트에서, VBSript는 서버와 클라이언트에서 동시에 실행된다.

CGI는 응용프로그램을 실행하는 표준 방식으로 자리잡았으며, 사용이 편리하다는 장점이 있지만, 일반적으로 서버에서 많은 프로그램을 수행해야 하므로 서버에 무리를 준다는 단점도 지니고 있다. 따라서 CGI를 대체할 수 있는 기술로 ASP, JSP, PHP와 같은 기술이 나타나게 되었다.

4.3 ASP

ASP(active server pages)는 CGI보다도 서버의 부담을 덜 수 있다는 측면에서 각광 받고 있는 서버에서 해석되는 스크립트 언어이며, CGI에 대응하기 위한 Microsoft사의 서버기반기술이고, Microsoft사의 IIS(internet information server)에서 쉽게 프로그래밍 할 수 있게 고안된 언어이다.

ASP는 CGI의 서버에서 수행되는 성질과 JavaScript의 성질을 동시에 가진 것이라고 보면 된다. ASP는 데이터베이스와 연동되는 프로그램의 작성이 쉽고 객체 기반의 프로그램이 되며, 동시 접속 시 스레드(thread)로 처리되어 서버의 부하가 적다는 특징을 가진다. 서버에서 작동하므로 속도는 서버의 사양에 따라 다르고, 클라이언트측은 인터넷을 사용하는 사용자의 사양에 따라 다르다. 지금까지 많이 사용하던 CGI 프로그램보다 작성하기가 쉬워 빠르게 확산되고 있으나, 윈도우 운영체제에서만 사용할 수 있다는 단점도 있다.

4.4 JSP

JSP(java server page)는 웹전용 스크립트 언어로써, PHP, ASP와 비슷하지만, Java의 막강한 기능을 그대로 가지고 있기 때문에 더욱 강력하다. JSP는 HTML, DHTML, XML과 같은 동적 컨텐츠(contents)를 처리하는 Java 플랫폼 기술이다. HTML코드에 Java 코드가 포함되어, JSP 탐지 서버가 이를 컴파일하고 스스로 해당 서브릿(servlet, 서버에서 수행되는 소형 프로그램)을 작동/삭제시킨다.

JSP는 웹 서버에 있는 서브릿을 사용해 웹 페이지의 내용과 모양을 제어하는 기술이다. 서브릿 API(application program interface)로써 ASP와 유사하나, JSP는 Java 프로그램을 호출하고, ASP는 스크립트(VBScript나 JScript)를 사용한다.

JSP는 CGI의 경우 발생하는 사용자 수의 증가에 따라 성능이 감소되는 현상이 상당 부분 없어지며, Java를 기반으로 만들어져 Java의 가장 큰 장점인 플랫폼의 독립성이 그대로 적용되어, 모든 웹 서버, 웹 어플리케이션 서버에서 실행시킬 수 있으며, 어떠한 웹 브라우저에서도 접근이 가능하다는 장점이 있다. JSP는 Java를 기반으로 하므로, 시스템에 자바가상머신(java virtual machine)이 구축되어 있어야 한다.

4.5 PHP

PHP(personal hypertext preprocessor)는 HTML에 내장되어 동작하는 스크립트 언어(HTML-embedded scripting language)이다. 별도의 실행 파일을 만들 필요 없이, HTML 문서 안에 직접 포함시켜 사용하며, C, Java, Perl 언어 등과 유사한 문장 형식이 많아, 이에 대한 기본 지식이 있는 사람은 웹 문서를 빠르고 쉽게 작성할 수 있다. ASP와 같이 스크립트에 따라 내용이 다양해서 동적 HTML 처리 속도가 빠르며, 처음에는 'Personal Homepage Tools (PHP)'이라 불렸으며, 공개된 무료 소스이다.

서버용 스크립트 언어인 PHP가 해석되는 과정은 우선 사용자가 웹서버에 특정 URL을 요청하면 웹 서버가 요청받은 URL을 찾아 어떤 언어로 만들어졌는지 확인한다. 만일 PHP로 만들어져 있다면, 서버에서 가지고 있는 PHP해석 엔진으로 보내고, 해석 엔진에서는 해당 코드를 해석해서, 결과를 서버로 보내게 되고, 서버는 전달 받은 결과를 그대로 사용자(클라이언트 또는 웹 브라우저)로 전달한다.

PHP의 특징으로는 우선 데이터베이스의 쉬운 연결을 들 수 있다. PHP에서 지원하는 데이터베이스에는 MySQL, dBase, ODBC 등을 포함함 거의 모든 데이터베이스를 지원한다. 또한 PHP는 UNIX 운영체제를 비롯하여 윈도우

운영체제하의 IIS나 PWS(personal web server), 윈도우용 Apache 웹 서버에서도 돌아간다.

4.6 Java

Java는 미국의 Sun Microsystems사가 개발한 객체 지향 프로그래밍언어로, 1995년 5월에 발표, 1996년 1월부터 정식 버전을 배포하였다. Java는 C++를 바탕으로 언어 규격을 규정하였다. 버그의 원인이 되기 쉬운 기능, 예를 들면 포인터(pointer) 및 연산자의 중복 등을 생략하였다. 또 C++에서는 사용이 끝난 객체(object)를 명시적으로 폐기하도록 프로그램에 기술하였으나, 그 대신 garbage collection(자동 폐영역 회수) 기능을 추가하였다. Java는 객체 클래스의 계승(inheritance) 관계를 실행할 때에 확정하는 동적 모델을 채용하였다. C++에서는 컴파일 시에 확정하기 때문에 상위 클래스에 새로운 기능을 추가하면 그 클래스와 계승 관계가 있는 모든 클래스를 재컴파일해야 하지만, Java에서는 클래스의 계승 관계를 실행할 때 확정하기 때문에, 하위의 클래스를 재컴파일할 필요가 없다. Java는 TCP/IP 프로토콜을 처리하므로 네트워크 분산 환경에 유리하고, 멀티 스레드 기능을 언어 자체에서 지원하여 대화형 인터넷 프로그램의 개발에도 유리하다. 반면에 Java는 프로그램 실행 시 실행형태에 관한 많은 정보를 동반하므로 실행속도가 떨어질 수 있으나 프로그램의 작성에는 융통성이 생긴다.

Java로 작성한 프로그램의 원시 코드는 Java 컴파일러로 컴파일한다. Java는 인터넷에 연결된 다양한 운영체제와 CPU를 가지는 컴퓨터에서 모두 실행되는 언어가 되기 위하여, 플랫폼에 독립적인 중간 코드인 바이트코드(bytecode)를 생성하여 이를 만족시킨다. 이 코드는 각 각의 컴퓨터에서 실행되는 기계어 코드가 아니며 자바가상머신이라는 인터프리터를 통하여 실행이 되며, 자바가상머신을 실장한 컴퓨터라면, 컴퓨터의 명령 집합 아키텍처나 운영 체계(OS)에 관계없이 같은 바이트코드를 변경하지 않고, Java를 실행할 수 있다. 즉, Java의 원시 코드를 고쳐 쓰거나 재컴파일할 필요가 없다. 그렇기 때문에 Java는 기종이나 운영 체계와 무관한 응용 프로그램의 개발 도구로 각광받고 있다.

5. 참고문헌

- C# (http://msdn.microsoft.com/vcsharp/)
- Computer language history (http://www.levenez.com/lang/)
- Intel Visual FORTRAN(http://www.intel.com/cd products/asmo-na/eng/compilers/278834.htm)
- Maplesoft Inc.(URL: http://www.maplesoft.com)
- Mathworks Inc.(URL: http://www.mathworks.com)
- The C Programming Language 2nd ed., B. Kernighan and D. Ritchie, Prentice Hall, 1988

4.7 각 언어의 비교

구 분	CGI	ASP	PHP	Java(JSP)
구동가능 웹서버	모든 서버에서 기본적으로 지원하므로, 모든 환경에서 동작 가능.	IIS, PWS의 NT/Windows 환경에서만 작동하며, 웹서버는 IIS, DB는 Ms-SQL을 주로 사용.	플랫폼이 독립적이므로 아무 환경에서나 동작하지만, Apache웹서버와 MySQL을 주로 사용.	모든 환경에서 동작하고, 특별히 최적화되는 환경이 없음.
시스템 자원 효율성	Java(Jsp) > PHP > ASP > CGI Java는 스레드를 통해서 프로세서를 실행시키기 때문에 자원을 적게 쓰고, 반면 CGI는 요청이 있을 때마다 프로세서를 실행시키기 때문에 자원의 낭비가 제일 심하다.			
플랫폼 독립성	Java(Jsp) = CGI = PHP > ASP Java, CGI, PHP는 객체지향언어로서 플랫폼의 독립성이 뛰어나 어느 환경에서나 어느 환경에서나 실행이 가능하나, ASP는 Window환경에서만 실행이 가능하다.			
확장성	Java(Jsp) > PHP > ASP > CGI			
데이터 접근성	Java(Jsp) = PHP > ASP > CGI			
프로그램 난이도	ASP > PHP > Java(Jsp) > CGI CGI 및 Jsp는 기존의 C언어 및 Java를 알아야 사용할 수 있는 언어이기 때문에 스크립트 중심의 ASP 및 PHP에 비해 프로그래밍 난이도가 높다.			
프로세스 처리속도	PHP > Java(Jsp) > CGI > ASP 단독 처리속도는 스크립트 해석능력이 가장 뛰어난 PHP가 단연 빠르고, CGI기반의 ASP 및 CGI같은 경우 프로세서의 지속된 생성으로 처리속도가 비교적 느리다. 특히 ASP는 CGI로의 컴파일 과정이 포함되어 더욱 느리다. 그러나 실제적으로는 거의 차이가 없다.			

• Visual C++ (http://msdn.microsoft.com/visualc/)
• Wolfram Research Inc.(URL: http://www.wolfram.com)
• 백광현, 인터넷을 이용한 진동/소음 관련 기술: 인터넷 언어와 공학계산에의 응용, 한국소음진동공학회지, Vol.12, No.3, pp.163~171, 2002.6.
• 정보통신부, 뉴스 및 보도자료, http://www.mic.go.kr.