P 언어를 이용한 한글 프로그래밍

최시영
싸이브레인 연구소
sychoi2100@gmail.com

Programming with Korean Vocabulary by Using P Language

Sea-Young Choi
Cybrain Laboratory

요 약

본 논문에서는 모국어를 이용한 프로그래밍을 위한 지원 방법으로서, 모국어로 된 데이터의 표현, 변수의 모국어 표현, 문법 키워드의 모국어 표현, 모국어 병행표현 등에 대하여 알아본다. 그리고 임의의 다국어를 지원하도록 설계된 P 언어를 이용하여 한글 프로그래밍을 하는 방법을 알아본다. 구체적으로 한글 프로그래밍 환경을 구축하는 방법, 한글 프로그램을 위한 어휘의 선정에 대하여 알아보고, 이를 이용한 간단한 알고리즘의 구현과 art 모듈을 이용하여 그래픽 프로그래밍의 예를 들어보겠다. 그리고 한글 프로그래밍을 위해 P 언어를 사용한 경우의 장점과 단점에 대하여 알아보겠다. 끝으로 한글 프로그래밍의 발전을 위해서는 표준 한글어휘 선정을 위해 학계와 산업계의 통일된 노력이 필요하다는 점과 한글 프로그래밍이 가져다 줄 수 있는 영향과 한글 프로그래밍의 바른 사용 방법으로서 영문과 한글의 병용사용을 제안한다.

주제어: 모국어 프로그래밍, 자국어 프로그래밍, P 언어, 한글 프로그래밍

1. 서론

자연어 분야에서 모국어가 아닌 외국어를 배우는 과정은 오랜 시간과 노력이 필요하다. 이와 비슷하게 프로그래밍 언어의 영역에서도 모국어에 기반하지 않은 프로그래밍 언어를 새로 배우는 과정도 오랜 시간과 노력을 요구한다. 이런 추가적인 학습비용에도 불구하고 자신의 모국어에 기반하지 않는 프로그래밍을 배우는 이들은 해당 프로그래밍 언어가 기반하는 자연어를 모국어로 하는 문화권의 사람들보다 기술적으로 높은 성과를 내기에는 여러 가지 어려움이 존재한다. 비모국어 사용자에게는 프로그래밍에 대한 입문 자체를 어렵게 하고, 숙련된 소프트웨어 개발자들에게도 외국어로 작성된 프로그래밍 소스와 기술문서에 대한 이해에 많은 어려움이 존재하게 한다.

1946년 미국에서 최초의 전자식 일반목적(general purpose)용 컴퓨터인 ENIAC(Electronic Numerical Integrator And Computer)[1]이 만들어진 이후로 많은 후속 컴퓨터 모델들이 미국에서 만들어졌다. 프로그래밍 언어는 이들 컴퓨터 하드웨어를 조작하기 위한 도구로서 탄생된 것이기 때문에, 프로그래밍 언어는 이들 하드웨어들이 제조된 미국에서 많이 만들어지게 되었다. 이런 이유로 대부분의 프로그래밍 언어는 영어에 기반하여서 만들어져 왔다. 프로그래밍 언어는 그 언어를 만든 개인 또는 회사의 모국어에 자연스레 영향을 받게 된다. 구체적으로 프로그래밍 언어 속의 문법어, 데이터 타입, 함수 이름, 언어 기술서등이 모두 해당 언어로 만들어지기 때문이다. 예를 들어 한국인 프로그래머는 영어로 작성된 문서상의 'concat' 라는 함수가 'concatenate' 라는 단어에서 나왔으며, 그 함수의 역할이 문자열을 연결시킨다는 것을 예상하기는 쉽지 않다.

영어에 기반하고 있는 프로그래밍 언어들

초기의 프로그래밍 언어들인 IPL(1954), FORTRAN(1955), LISP(1958)[2], ALGOL(1958), COBOL(1959) 들과, 그 후에 나온 RPG(1959), APL(1962), Simula(1962), SNOBOL(1962),CPL(1963) 등의 언어들은 모두 영어에 기반하고 있다. 역사적으로 학계에서 많이 사용되거나 산업계에서 대중화된 언어들인 BASIC, C, C++, Smalltalk, Prolog, ML, Pascal, Forth, SQL, Ada, Modula, Java, Erlang, Perl, Tcl, Delphi 등의 언어들도 영어에 기반하고 있다. 최근에 나온 Haskell[3], Python[4], Ruby[5], Lua[6], Scheme[7], Javascript, PHP[8], D[9], C# (Microsoft Inc., ISO/IEC/2327)[10] 등의 언어들도 영어에 기반하고 있다. Ruby 언어는 1990년경에 일본의 마츠모토 유키히로가 만든 언어이지만 영어에 기반하고 있다. Prolog, Python, Erlang 등도 비영어권 나라에서 개발되었지만 영어에 기반하고 있다.

본 논문에서는 프로그래머 또는 사회제도에 각각 초점을 맞춘 표현들인 '모국어' (mother language) 와 '자국어' (national language)를 같은 의미로 혼용해서 사용하겠다. 이에 따라 '모국어 프로그래밍', '모국어를 이용한 프로그래밍', '자국어 프로그래밍', '자국어를 이용한 프로그래밍' 들을 모두 같은 의미로 사용하겠다.

2. 관련 연구

많은 프로그래밍 언어가 영어에 기반하고 있기는 하지만, 산업계 중심으로 모국어로 프로그래밍을 하고자 하

는 노력이 부단하게 이루어져 왔다.

2.1. 소프트웨어에 있어서 자국어 지원을 위한 기술들

영어가 아닌 자국어에 기반한 소프트웨어를 사용하고자 하는 기술은 실용적인 이유에서 학계보다는 산업계에서 많이 연구되어온 주제이다. 소프트웨어의 일반 사용자들에게는 무엇보다도 자국어를 포함한 데이터를 표현하는 것이 가장 시급한 문제이었기 때문에 자국어 문자를 표현할 수 있는 문자집합에 대한 기술이 가장 먼저 개발되었다. 그리고 일반 사용자들이 자국어에 기반하여 소프트웨어를 사용할 수 있도록 메뉴, 시간, 통화 등을 표현하는 기술인 소위 '국제화와 지역화'에 관련된 기술이 발달되었다.

2.1.1. 문자집합(character set)

문자집합(character set, charset)은 문자들의 집합을 정의한 것을 말한다. 이와 비슷한 개념으로서 문자인코딩(character encoding)이라는 것이 있는데, 문자인코딩은 이러한 문자들을 부호화하는 방식이다. 그러므로 문자가 한 바이트에서 어느 영역에 할당되는 가 하는 것과 한 개의 문자가 몇 개의 바이트로 표현되는가 하는 문제들은 모두 문자인코딩의 문제이다. 그러나 이 두 가지는 실제에서는 거의 비슷한 의미로 사용되고 있다. 예를 들어 일부 문헌(대표적으로 미국 마이크로소프트사의 문서)에서 사용하는 다중바이트문자셋(multibyte character set) 이라는 표현은 부적절한 표현(misnomer)이다.

컴퓨터 초창기에는 4 bit로 이루어진 이진수를 묶어서 10진법 한자리수를 표현하는 BCD (Binary-Coded Decimal)방식[11] 을 개량해 IBM 회사가 6 bit를 사용해 숫자뿐만 아니라 모든 알파벳 문자와 특수기호를 표현하는 BCDIC (Binary-Coded Decimal Interchange Code) 방식을 개발하여 사용하였다. BCDIC 방식은 통일된 방식이 없이 각 회사별로, 각 제품 별로 다르게 정의되어서 사용되고 있었다. 그 후 미국국립표준협회(The American National Standards Institute, ANSI)가 1963년에 ASCII 표준[12]를 만들어서 이 방식이 널리 사용되게 되었다. 그러나 ASCII 는 영문자만을 지원해서, 한글문자, 한자, 일본문자 등의 동양권 언어들의 문자를 지원할 수 없어서, IBM 과 마이크로소프트사에서는 소위 multi-byte character set을 만들어 우리나라의 KSC 5601-1987 등을 지원하였다. 그러나 동양 3국의 문자뿐만 아니라 현존하는 모든 문자들을 표현하기 위한 필요성이 제기되면서, 마침내 1991년 유니코드협회(Unicode Consortium, Unicode Inc.)에서 유니코드(unicode) 표준[13]이 나왔으며, 이것이 널리 사용되게 되면서 현재로서는 산업계의 표준이라고 할 수 있게 되었다. 유니코드 표준은 1991년8월에 버전1.0 이 나온 이후로 현재까지 2012년9월에 버전 6.2 가 나왔다. 유니코드는 문자셋에 대한 이름이며, 유니코드를 부호화하는 유니코드 인코딩 방식으로서는 UTF-7, UTF-8, UTF-16, UTF-32 등이 있으며, 그 중에서도 UTF-8 이 널리 사용되고 있다.

2.1.2. 국제화와 지역화(Internationalization and localization)

국제화와 지역화는 소프트웨어를 여러 지역 환경에 공통적으로 사용할 수 있도록 만들어 각 지역 환경에 부합하여 변경하는 것을 말한다. 국제화는 보통 I18N 으로 표기하는데, I18N은 'internationalization' 의 첫 글자 'I' 와 마지막 글자 'N' 과 그 사이의 'nternationalizatio' 의 글자수인 18을 결합해서 만든 신조어이다. 이 국제화와 현지화는 주로 응용소프트웨어에 관하여 발전된 기술이며, 각 지역의 문자집합, 날짜/시간형식, 통화표시 방법 등에 관한 내용을 다루고 있다. 국제화와 지역화는 별개로 존재하는 것이 아니라 서로가 상대를 가정하여 존재하며 서로 영향을 미치는 상호 순환하는 과정이다(그림 1 참조)

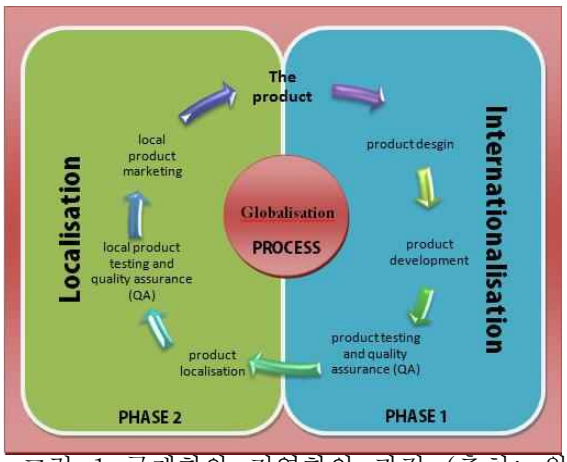

그림 1 국제화와 지역화의 과정 (출처: 위키피디어, IaT_vicky[14])

2.2. 프로그래밍 언어에서의 자국어 지원을 위한 연구

프로그래밍 언어 수준에서 자국어를 지원하고자 하는 필요성과 노력은 많이 있었으나, 명확한 규격이나 구현까지 이루진 것은 많지 않다.

2.2.1. ALGOL 68 의 키워드의 자국어표현

ALGOL 언어는 소위 GAMM-ACM 위원회를 통하여 미국과 유럽의 양 대륙 컴퓨팅 학계의 공동노력으로 이루어진 것이어서, 영어뿐만 아니라, 독일어, 불어, 러시아, 불가리어 같은 유럽의 언어로도 표현될 수 있는 것이 필요하였다. 특히 ALGOL 68 은 ALGOL 전통에 따라 여러 가지 형식의 표현법을 지원하였다. 즉, 표현 언어(representation language) 로서 참조언어(reference language), 출판언어(publication language),하드웨어언어(hardware language)를 지원하고 있다[15][16]. 영어가 아닌 다른 자국어에 기반한 문법 키워드로 구성되는 언어 표현을 지원할 수 있는 여지를 두었고, 실제로 러

시아어로 된 ALGOL 68 규격도 나왔다[17].

2.2.2. 프로그래밍 언어의 특정 자국어 판(localized version)

이들 언어들은 영어에 기반한 특정 프로그래밍 언어의 키워드를 자국어로 변경한 언어들이다. 이들 언어들은 원본 언어들과 분리되어 개발과 배포되어, 원본 언어들의 발전사항을 반영하지 못하는 제약점이 있고, 해당 자국어에 고정화(hardcoded) 되어 해당 자국어 이외로의 번역이 불가하여 해당 자국어를 사용하는 프로그래머만 이해할 수 있는 단점이 있었다. 각 나라별로 여러 가지 언어들이 등장하였다. 중국어의 경우, BASIC 언어에 기반한 中文培基(Chinese BASIC), Python 언어에 기반한 中文파이션[18]등이 있다.

2.2.3. 관련 개념들

Source-to-source translation: 한 프로그래밍 언어로 된 소스를 다른 프로그래밍 언어로 된 소스로 바꾸는 것을 말한다. 예를 들면 Ada 언어로 작성된 소스를 Pascal 언어 또는 C 언어로 바꾸는 경우를 말한다.

Natural Language Processing: 인간이 사용하는 자연어에 대하여 프로그래밍 기법을 통하여 분석하는 것을 말하며, NLP 라고 자주 통칭된다.

2.3. 전산용어의 한글화 노력

국립국어원에서는 1991년부터 2002년까지 각 분야별 “국어 순화 자료집”을 발간하였다. 그중에 1996년도에는 전산 관련 한글 순화 자료집을 발간하였는데, 이것은 “전산기 용어”라는 분야에서 전산 관련 일반적인 용어에 관하여 정하였으며 프로그래밍 언어에 특화되지는 않았다. 2003년에는 각 분야별 국어 순화 자료집을 모아서 “국어 순화 자료집 합본”으로 출간하였다[19]. 이러한 국어 순화의 노력에도 불구하고, 일반 국민에게는 제대로 알려지지 않았으며, 일반 국민들 또한 이에 대한 관심이 극히 낮았다고 한다[20]. 그 이유는 국어순화가 관련 몇몇 전문가 중심으로 이루어져 일반 국민들의 반응을 전혀 살펴보지 않은 채 순화어를 대량으로 만들어 일방적으로 보급하여 일반 국민의 언어의식에 합치되지 않았기 때문이라고 한다[20].

3. 모국어 프로그래밍의 지원방법

모국어 프로그래밍을 지원하기 위한 영역은, 언어 내적인 부분과 언어 외적인 부분으로 나뉜다. 언어 외적인 부분으로서는 해당 언어를 실행하는 실행기(인터프리터 또는 컴파일러)의 사용과, 언어 매뉴얼 등의 언어관련 문서에서 해당 언어를 지원하는가의 여부이다. 언어 실행기의 사용자 메뉴와 오류 메시지 또는 디버그 메시지 등이 해당 언어를 지원하여야 한다. 언어 내적인 부분은 다음과 같은 영역을 지원하여야 한다.

3.1. 모국어 데이터의 표현(입력과 출력)

모국어로 된 데이터를 표현할 수 있는가 하는 점이다. 예를 들어 문자열(string)에서 모국어 문자열을 사용할 수 있는가 또는 모국어로 된 파일이름을 지정할 수 있는가 하는 점이다. 이를 위해서는 모국어 문자집합(character set) 의 구현뿐만 아니라, 모국어 문자의 입력과 출력을 포함한다. 예를 들어 키보드 상에서 직접 한글을 입력시킬 수 있어야 하며, 모니터 화면상으로 한글이 표현될 수 있어야 한다. 다음은 모국어 문자열의 예이다.

“ครสิมาสตหกีำลงจะมา”, “圣诞节快到了”

3.2. 모국어 이름의 변수 사용

모국어에 기반한 문자로 바로 식별자(identifier)를 만들 수 있어야 한다. 예를 들면 다음과 같은 방식의 프로그래밍이 가능하여야 한다.

```
金額 = 100 ;
병의크기 = 갑의크기 + 을의크기 ;
私のリスト = [ 1,2, 3, 4] ;
```

3.3. 모국어로 된 문법어(키워드)

문법어를 이루는 키워드가 모국어로 표현될 수 있어야 한다. 문법 키워드는 파서를 만들 때 파싱(parsing)의 중요한 기준점이 되며 각 언어의 핵심에 해당하는 부분이다. 그러므로 각각의 모국어에 따른 다른 문법 키워드를 지원하기 위해서는 원칙적으로 각각의 모국어에 따른 개별적인 파서를 만들어야 한다는 어려움이 존재한다. 모국어로 된 문법어가 지원된다면, 예를 들어 다음과 같은 방식의 표현이 가능하다.

```
만일에( 갑의수학점수 > 90 )
{
   반복하기( 3번, 출력하기(“축하! ”) ) ;
   출력하기( “참 잘 했습니다.”);
}
```

이것에는 한 개의 모국어만을 지원하는 경우(이글 2.2.2.참조)나, 특별히 지정된 몇 개의 모국어가 지원 가능한 경우(ALGOL 68), 임의의 모국어를 지원 가능한 경우(P 언어)로 나누어 볼 수 있다.

3.4. 모국어 병행표현(모국어전환)

병행표현은 특정 모국어에 기반한 프로그래밍 소스를 다른 모국어로 변경할 수 있는 가 하는 점이다. 모국어 프로그래밍의 가장 큰 장점이자 단점은 해당 모국어로 작성된 프로그래밍 소스가 해당 모국어 사용자에게는 판

독성이 높아지나, 외국어 사용자에게는 이해가 불가한 소스가 된다는 점이다. 모국어 간의 전환은 이러한 단점을 보완해주는 기능을 하게 된다. 예를 들어, 병행표현이 지원된다면, 아래의 중국어로 된 소스는 한글로 된 다른 소스로 변경할 수 있게 되고, 프로그램 실행시 동일한 결과가 나와야 한다.

```
函数定義 新增评论(自己, 稿件)
{
    寫( "<!--" + 自己 + "-->" );
    寫( 稿件 );
}

함수정의하기 주석달기(내이름, 문건)
{
   쓰기( "<!--" + 내이름 + "-->" );
   쓰기( 문건 );
}
```

4. P 언어를 이용한 한글 프로그래밍

4.1. P 언어란

P 언어는 일반목적(general-purpose)용 동적인 스크립팅(scripting)형 언어이다. P 언어는 멀티패러다임을 지원하며, 특히 객체지향과 함수형 프로그래밍을 동시에 지원한다. P 언어의 특징[21] 으로서는 a) 간결한 문법을 사용하면서도 표현력이 뛰어나고, b) 복잡한 알고리즘을 쉽게 표현할 수 있으며, c) 모든 데이터는 함수 호출을 통해 생성되며, d) 모든 제어구조는 함수로 표현되고, e) 모든 연산자는 함수의 이름에 불과하며, f) 이름과 값이 분리돼 있는, g) 식 중심의 언어(expression-oriented programming language)이다.

4.2. P 언어의 한글 프로그래밍의 지원

P 언어는 자연어 중립적인 언어이며 어느 자연어에 구속된 언어는 아니다. 그러나 P 언어는 처음부터 모든 개별 자연어를 지원하기 위해서 디자인이 된 언어로서, 개별 자연어에 대한 설정 파일만 만들어 주면, 해당 자연어에 기반한 프로그래밍이 가능하게 설계되었다. 그리고 위에서 서술한 모국어 프로그래밍의 4 가지 지원분야를 비교적 충실히 지원한다.

4.3. 한글프로그래밍의 환경구축

ㄱ) 개별 프로그래머가 정한 한글 어휘를 지원하는 방법: P 언어가 실행되는 인터프리터 상에서 P 언어 소스를 입력하는 것과 동일한 방식을 사용하여, 프로그래머가 선정한 한글을 지원할 수 있다. 인터프리터 상에서 다음과 같은 형식의 문장을 입력한다. 여기서 '새로운 이름'은 한글 어휘를 말한다.

새로운이름 = 기본이름 ;

예를 들어, 영어 'if' 라는 표현대신에 한글로 '만약에' 이라는 표현을 사용하고자 하는 경우에는 인터프리터 상에서 다음과 같이 입력한다.

```
만약에 = if ;
```

ㄴ) 설정 파일에 의한 환경 구축: 매번 인터프리터를 실행할 때 마다 프로그래머가 한글 환경을 일일이 구축하는 방식은 효율적이지 못하다. P 언어와 함께 배포되는 각 언어별 설정파일을 이용하여 한글 프로그래밍 환경을 구축할 수 있다. 이 방법을 사용하면 표준화된 한글어휘를 정할 수 있는 장점이 있다.

4.4. 한글 프로그래밍의 모습

P 언어를 사용해서 재귀 함수를 사용하는 '팩토리얼' 이라는 함수를 정의해보면 다음과 같다.

```
함수정의하기(팩토리얼, 숫자,
             만약에( 숫자 == 0 ,
                     1,
                     숫자*팩토리얼(숫자-1) )
 ) ;
 팩토리얼(100);   // 함수사용법
```

객체지향 프로그래밍을 지원하는 파일 사용법은 다음과 같다.

```
철수파일 = 파일( "내파일" ) ;
철수파일.쓰기( "작성자:철수" ) ;
철수파일.닫기( ) ;
```

4.5. art 모듈의 사용

P 언어에서 기본적으로 제공하는 art 모듈 속의 간단한 함수를 이용하여 다음과 같은 그래픽 프로그래밍을 할 수 있다.

```
그림판(400,300);
반복하기( 10,  ( 앞으로(10),
                 원그리기(30),
                 왼쪽으로(65))) ;
```

표 1. art 함수

한글함수	영어함수	하는 일
그림판	canvas	그림판 만들기
앞으로	forward	앞으로 가기

뒤로	backward	뒤로 가기
오른쪽으로	right	오른쪽 방향전환
왼쪽으로	left	왼쪽 방향전환
원그리기	circle	원거리기

그림 2 art 모듈을 사용한 예 (출처 [22])

5. P 언어를 사용한 한글 프로그래밍의 장단점

5.1. P 언어를 사용한 한글 프로그래밍의 장점

ㄱ) 자유로운 한글 어휘의 선택가능: P 언어에서는 문법 키워드(또는 문법 키워드 역할을 하는 함수)가 syntax 수준이 아니라 실행시 이름공간(name space)상의 이름으로 존재하므로, 프로그래머가 임의적으로 원하는 한글 어휘를 사용하여 한글 프로그래밍을 할 수 있다.

ㄴ) P 언어는 다국어 지원이 언어 내재적으로 이루어지므로 다른 지역화 된 언어(이글 2.2.2.참조)들과 다르게, 각 다국어 버전이 독립적으로 존재하지 않고 하나의 버전 안에 존재하게 된다. 그러므로 P 언어의 발전(업그레이드)이 모든 다국어 버전에 그대로 반영되게 된다.

ㄷ)임의의 자국어 지원 가능 및 자국어간의 전환 가능

프로그래머가 표준화된 자국어 어휘를 사용하는 경우, 다른 자국어으로의 전환이 가능하여, 각 자국어 버전간의 소스전환이 이루어질 수 있다.

5.2. P 언어의 단점 및 미비점

P 언어는 위에서 서술한 바와 같이 한글 프로그래밍을 비교적 충실히 지원하지만, P 언어 디자인상의 자유로움으로 인해 언어 내적인 단점과 신규로 만들어진 언어로 인한 언어 외적인 미비점이 존재한다.

ㄱ) 프로그래머의 비표준화된 한글 어휘의 사용 : P 언어는 제어구조가 특정 키워드에 고정화된 언어가 아니므로 프로그래머가 P 언어에서 제공하는 표준화된 한글 어휘를 사용하지 않고 독자적인 한글 어휘를 사용하는 경우에는 가독성이 떨어질 수 있다. 예를 들어 프로그래머는 'if' 기능을 하는 한글어휘인 '만약에' 대신에 '만약', '만일', 참인경우' 등의 표현을 사용하는 경우에는 다른 외국어 버전으로 전환이 어려울 수 있다.

ㄴ) 충분한 라이브러리의 부족: P 언어는 문자열, 함수, 리스트, 배열 등의 다양한 데이터 타입들을 지원하고, 파일, 네트워크, 윈도우를 위한 프로그래밍을 지원하지만, 다양한 소프트웨어를 손쉽게 만들기 위해서 필요한 여러 가지 라이브러리(language libraries)가 아직 지원되고 있지 않다.

ㄷ) 실행파일의 생성 지원 미비: 영리 목적의 소프트웨어 제작사의 경우에는 자신의 프로그래밍 소스를 공개하기보다는 기술 보호를 위해서 바이너리로 된 실행파일 형태의 소프트웨어 제작을 선호하는 편이다. P 언어는 현재로서는 스크립트 기반의 언어로서 모든 P 소스는 텍스트 파일 형태로 존재하여 공개되게 되어있다. 그러나 추후 P 언어를 지원하는 컴파일러가 제작이 되면, P 언어에서 바로 바이너리 코드로 만들어진 소프트웨어 제작이 가능할 것이다.

5. 결론

P 언어는 한글만을 위한 한글 전용 프로그래밍 언어는 아니지만, 임의의 다국어를 지원할 수 있는 언어 설계에 의하여 한글 프로그래밍이 가능하다는 것을 알 수 있었다.

한글 프로그래밍이 가능하다고 하여도 구체적으로 프로그래밍에 사용될 한글 어휘에 대한 결정은 언어 외적인 영역에 속한다. 위의 관련 연구(이글 2.3.참조)에서 언급하였던 한글 순화의 미비한 효과에서 알 수 있듯이 언어의 사용자인 일반 국민의 참여 없이 몇몇 전문가들에 의한 작업은 큰 성과를 내기가 어렵다. 프로그래밍 언어의 주된 사용자는 산업계이므로 한글 프로그래밍 어휘의 선정을 위해서는 산업계의 참여와 협조가 절대적으로 필요하다. 그러므로 한글 프로그래밍의 개척과 발전을 위해서는 한글 어휘에 대한 학계와 산업계의 통일된 표준이 존재하여야 할 것이다. 한글 프로그래밍의 발전은 프로그래밍 인구의 저변 확대뿐만 아니라, 더 중요한 영향으로서, 영어적 사고 외에 한국어라는 새로운 언어적 사고의 도입으로 인하여 프로그래밍 이론 분야에서 창의적이고 혁신적인 이론 개발에 기여할 것이다.

한글과 한자를 병행하여 사용할 때 우리말의 표현력과 정확성이 높아지듯이, 프로그래밍 분야에서도 한글 전용 또는 영어 전용보다는 한글과 영어를 적절히 배합하여 사용하는 경우에 좀 더 표현력과 정확성이 높아질 수 있다. 한글 프로그래밍은 기존의 영어 전용 환경에서의 불편함과 비효율성을 해소 하고 모국어의 활용성과 사고적 다양성을 높이기 위한 것이지, 한글 프로그래밍 그 자체를 목적으로 하는 것은 아니다.

참고문헌

[1] http://en.wikipedia.org/wiki/ENIAC
[2] McCarthy, John. "Recursive Functions of Symbolic Expressions and Their Computation by Machine,

Part I", 1960.
[3] http://www.haskell.org
[4] http://www.python.org
[5] http://www.ruby-lang.org
[6] http://www.lua.org
[7] Sussman, Gerry, Hal Abelson, and Julie Sussman, "Structure and interpretation of computer programs." The Massachusetts Institute of Technology 10, 1985.
[8] http://www.php.org
[9] http://www.dlang.org
[10] Information technology-Programming languages-C#, http://standards.iso.org/ittf/PubliclyAvailableStandards/c042926_ISO_IEC_23270_2006(E).zip
[11] Eric Fisher, "The Evolution of Character Codes, 1874 - 1968", http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.678&rep=rep1&type=pdf, Accessed 2013
[12] American Standards Association, "American Standard Code for Information Interchange", ASA X3.4-1963, June 1963.
[13] Unicode Consortium, "The Unicode Standard, volume 1.0." Addison-Wesley, Reading, MA, 1991.
[14] http://en.wikipedia.org/wiki/File:Globalisation chart.jpg
[15] Mailloux, Barry James, J. EL Peck, and C. HA Koster. "Report on the algorithmic language ALGOL 68." Numerische Mathematik 14.2, pp. 79-218, 1969.
[16] Van Wijngaarden, A., et al. "Revised Report on the Algorithmic language." Acta Informatica 5, pp. 1-236, 1975.
[17] "Programming language ALGOL 68 - Язык программирования АЛГОЛ 68", GOST 27974-88, 1988.
[18] http://www.chinesepython.org
[19] 국립국어원, "국어 순화자료집 합본", 2003.
[20] 고성환, "국어 순화의 역사와 전망", 국립국어원 새국어생활 Vol.181, pp. 5-6, 2011
[21] 최시영, "새로운 범용 프로그래밍 언어 data-p", 마이크로소프트웨어, pp. 254-255, 2012.10
[22] 최시영, "예제로 배우는 data-p 실전 프로그래밍", 마이크로소프트웨어, p. 257, 2013.2

부록 1. 데이터 타입의 한글이름 예

한글표현	영어표현	데이타유형
없음	none	null
진리값	bool	boolean
숫자	num	number
문자열	str	string
목록	list	list
배열	array	array
레코드	record	record
함수	func	function
매크로	macro	macro
변수	var	variable
상수	const	constant
클래스	class	class
인스턴스	instance	instance
모듈	module	module
자료집	file	file 객체
창	win	window 객체
망연결기	socket	sockect 객체

부록 2. 제어구조의 한글이름 예

한글표현	영어표현	하는 일
만약에	if	조건
동안에	while	조건부 반복
아닌동안에	unless	부조건부 반복
각각에	foreach	순환
경우에	case	택일 선택
계속하기	for	반복
반복하기	loop	단순반복
반환하기	return	함수결과 반환
중단하기	break	순환구조내 중단
다음하기	continue	순환구조내 다음과정
종료하기	exit	전체프로그램중단