북한 한글 순서를 지원하는 EUC-KR 기반의 로캘과 응용 프로그램 개발

정 일 동[†] · 이 중 화[††] · 김 용 호[†††] · 김 경 석[††††]

요 약

UCS(= ISO/IEC 10646, = Unicode)는 국제화에 따라 앞으로 점점 더 많이 쓰게 될 것이고, 일단 정착되고 나면 한참 동안 쓸 것으로 예상하고 있다. 하지만, UCS로는 남한과 북한의 같은 글자를 쓰면서도 다른 사전 순서로 쓰는 상황을 해결할 수 없다. 국제 표준 ISO/IEC 14651 : 2000(International String Ordering)은 여러 나라 글자계(script)가 섞여 있을 때, 모든 글자의 차례를 정하고 간추리는 틀에 관한 표준이다. ISO/IEC 14651을 이용하면 간추리는 차례가 공통 틀 표(Common Template Table)에 들어 있기 때문에 글자의 간추리는 차례를 쉽게 바꿀 수 있으며, 남한과 북한의 가나다 차례를 통일하지 않고 글자 순서가 다른 문제를 해결할 수 있다. ISO/IEC 14651 관련 함수는 리눅스와 솔라리스, FreeBSD와 같은 유닉스 기반 운영체제의 최신 라이브러리에 포함되어 있다. 본 논문에서는 북쪽의 한글 가나다 차례를 남쪽에서 활용할 수 있도록 하기 위해서 북쪽의 한글 가나다 차례를 포함하는 북쪽 로캘을 리눅스 시스템에서 만들고, 입력된 문자열(=글자떼)을 남쪽 혹은 북쪽의 한글 가나다 차례에 따라 간추리기 할 수 있는 프로그램을 개발하였다.

Development of EUC-KR based Locale and Application Program Supporting North Korean Collating Sequence

Il-dong Jung[†] · Jung-hwa Lee[††] · Yong-ho Kim[†††] · Kyongsok Kim[††††]

ABSTRACT

UCS (=ISO/IEC 10646, =Unicode) will be used widely as globalization. If UCS is used for official purpose in Koreas, UCS solves a problem in different hangeul code between South and North Korea. But, UCS is not a solution for problems in unequal order with the same character. ISO/IEC 14651 : 2000 (International String Ordering), which is a international standard for string ordering, defines a framework sorting all character strings consisting multi-national scripts. Because the Common Template Table in ISO/IEC 14651 defines orders of characters, we can change orders of characters without changes of characters sequences in programs. Therefore, we can solve a ordering problem without unifying order of hangeul in South and North Korea. Functions related ISO/IEC 14651 are contained by system librarys in unix-based operating system such as Linux, Solaris and FreeBSD. We implement EUC-KR-based North Korean locale, which includes North Korean hangeul order, in Linux in order to use North Korean locale in South Korea. And we develop a program ordering strings with South and North Korean hangeul order.

키워드 : ISO/IEC 14651, 한글공학(Hangeul Engineering), 한국어정보처리(Hangeul Information Processing), 한글 정렬(Hangeul Ordering), UCS

1. 서 론

"한민족 언어 정보화"는 국내외에서 국어정보화의 인구가 확대되고, 올바른 국어 생활을 영위할 수 있는 여건을 마련하는 것을 목적으로 한다. 이를 위한 과제로서 남북한으로 분단되어 있는 우리 나라의 현 상황에서, 남과 북이 다르게 추진하고 있는 국어 정보화를 조화시키고 통일시킬 필요가 있으며, 더 나아가서는 우리말과 우리 글을 사용하는 전세계의 해외 동포들에게 국어정보화의 틀을 마련하여 주어야 할 필요가 있다[1].

우리 글의 국어정보화의 기초는 한글 부호계의 통일에서 시작한다. 남한의 EUC-KR, 북한의 EUC-KP, 해외 동포들이 각자 쓰고 있는 부호계가 다르다면 이미 만들어진 정보를 통합할 수가 없다. 게다가 남한과 북한, 그리고 해외 동포가 작성한 문서를 서로 나누어 볼 수가 없다. 이는 정보의 분열뿐만 아니라 동포의 분열까지 초래할 것이다.

※ 본 논문은 2002년 학술진흥재단 남북학술협력지원사업에 의하여 연구되었음(2002-전-B-5).
† 정 회 원 : LG전자 DAC연구소 주임 연구원
†† 정 회 원 : 동의대학교 소프트웨어공학과 조교수
††† 정 회 원 : 한국기계연구원 자본재정보기술그룹 선임연구원
†††† 정 회 원 : 부산대학교 전자전기정보컴퓨터공학부 교수
논문접수 : 2004년 8월 10일, 심사완료 : 2004년 12월 7일

UCS(= ISO/IEC 10646, = Unicode)는 국제화에 따라 앞으로 점점 더 많이 쓰게 될 것이고, 일단 정착되고 나면 한참 동안 쓸 것으로 예상하고 있다. 따라서 얼마 지나지 않아서 남북도 모두 UCS를 쓰게 될 것이다. 현재의 상황으로 보아서는, 남북이 각각 UCS를 쓰다가 나중에 통일이 되면서 계속하여 UCS를 쓰게 될 것으로 보인다.

여러 한글 부호계를 사용하는 문제는 UCS를 사용하면서 해결할 수 있지만, 아래와 같이 남한과 북한의 한글 순서가 다른 문제는 해결할 수가 없다.

- **남쪽 가나다 차례**
  - 첫소리 글자 : ㄱ, ㄲ, ㄴ, ㄷ, ㄸ, ㄹ, ㅁ, ㅂ, ㅃ, ㅅ, ㅆ, ㅇ, ㅈ, ㅉ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ
  - 가운뎃소리 글자 : ㅏ, ㅐ, ㅑ, ㅒ, ㅓ, ㅔ, ㅕ, ㅖ, ㅗ, ㅘ, ㅙ, ㅚ, ㅛ, ㅜ, ㅝ, ㅞ, ㅟ, ㅠ, ㅡ, ㅢ, ㅣ
  - 끝소리 글자 : ㄱ, ㄲ, ㄳ, ㄴ, ㄵ, ㄶ, ㄷ, ㄹ, ㄺ, ㄻ, ㄼ, ㄽ, ㄾ, ㄿ, ㅀ, ㅁ, ㅂ, ㅄ, ㅅ, ㅆ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ

- **북쪽 가나다 차례**
  - 첫소리 글자 : ㄱ, ㄴ, ㄷ, ㄹ, ㅁ, ㅂ, ㅅ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ, ㄲ, ㄸ, ㅃ, ㅆ, ㅉ, ㅇ
  - 가운뎃소리 글자 : ㅏ, ㅑ, ㅓ, ㅕ, ㅗ, ㅛ, ㅜ, ㅠ, ㅡ, ㅣ, ㅐ, ㅒ, ㅔ, ㅖ, ㅚ, ㅟ, ㅢ, ㅘ, ㅝ, ㅙ, ㅞ
  - 끝소리 글자 : ㄱ, ㄳ, ㄴ, ㄵ, ㄶ, ㄷ, ㄹ, ㄺ, ㄻ, ㄼ, ㄽ, ㄾ, ㄿ, ㅀ, ㅁ, ㅂ, ㅄ, ㅅ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ, ㄲ, ㅆ

UCS에 들어있는 한글 첫가끝 조합형 글자와 완성형 글자마디의 차례는 남쪽의 것을 따랐는데, 이를 북쪽 차례 또는 제3의 가나다 차례에 따라 글자와 글자마디의 차례를 바꾸는 것은 UCS 범위 밖이다. 따라서 남북 한글 가나다 차례가 다른 문제를 해결할 방법이 필요한데, WG2에서는 한글 가나다 차례의 해결 방안으로 북쪽에 ISO/IEC 14651 을 활용할 것을 제안하였다. 국제 표준 ISO/IEC 14651 : 2000(International String Ordering)은 여러 나라 글자계(script)가 섞여 있을 때, 모든 글자의 차례를 정하고 간추리는 틀에 관한 표준이다.

ISO/IEC 14651 관련 함수는 리눅스와 솔라리스, FreeBSD와 같은 유닉스 기반 운영체제의 최신 라이브러리에 포함되어 있다. 현재 리눅스에는 ISO/IEC 14651을 지원하는 strcoll()/strxfrm() 함수가 있다. 이 함수는 로캘(Locale)을 바탕으로 작동하는데, 로캘은 프로그램 속으로 하드코딩을 하기 어려운 언어/문화와 관련한 사항을 다룬다. 리눅스에서 다양한 로캘을 만들고 설치하여 여러 나라의 언어에 관한 설정을 할 수가 있다.

본 논문에서는 북쪽의 한글 가나다 차례를 남쪽에서 활용할 수 있도록 하기 위해서 북쪽의 한글 가나다 차례를 포함하는 북쪽 로캘을 만들고, 입력된 글자떼를 남쪽 혹은 북쪽의 한글 가나다 차례에 따라 간추리기 할 수 있는 프로그램을 개발하였다. 2장에서는 ISO/IEC 14651에 대해서 소개하고, 3장에서는 본 연구에서 개발한 시스템을 서술한다. 4장에서는 ISO/IEC 14651을 사용한 관련 연구와 관련 문서를 살펴보고, 5장에서 결론을 맺는다.

2. ISO/IEC 14651

2.1 ISO/IEC 14651의 소개

ISO/IEC 14651의 제목은 International String Ordering and Comparison Method for Comparing Character Strings and Description of the Common Template Tailorable Ordering 인데, 2001년의 투표에서 국제 표준으로 통과되었다.

이제 ISO/IEC 14651의 내용을 보기를 보면서 살펴보자.

ISO/IEC 14651은, 주어진 문자열(= 글자떼, character string) 두 개가(보기 : s1, s2) 있을 때, 그 두 개의 관계(한글의 경우 가나다 차례라고 볼 수 있다)가 다음 세 가지 가운데 하나라고 결정한다.

가) s1 이 s2 보다 앞선다.
나) s1 과 s2 는 차례가 같다.
다) s1 이 s2 보다 뒤선다.

구체적인 보기를 들어보자.

(보기 1)
첫가끝 조합형으로 s1="간다"(U+1100 1161 11AB 1103 1161)와 UCS 완성형으로 s2="가다"(U+AC00 B2E4)

가 주어졌다면, "s2가 s1 보다 앞선다"는 결과가 나와야 한다. 그런데, 이 때에는 부호값을 그냥 견주면, 부호값 1100이 AC00 보다 작기 때문에, "간다"가 "가다" 보다 앞선다고 잘못 판단할 수 있다. 따라서 부호값의 크기만으로 어느 문자열이 앞선다고 결정할 수 없다는 점을 알게 된다.

(보기 2)
첫가끝 조합형으로 s1="가다"(U+1100 1161 1103 1161)와 UCS 완성형으로 s2="가다"(U+AC00 B2E4)

가 주어졌다면, "s1과 s2는 차례가 같다"는 결과가 나와야 한다. 그런데, 이 때에는 부호값을 그냥 견주면, 부호값 1100과 AC00 이 다르기 때문에, "간다"와 "가다"의 차례가 다르다(또는 첫가끝 조합형의 "가다"가 UCS 완성형 "가다" 보다 앞선다)고 잘못 판단할 수 있다. 따라서 부호값의 크기만으로 두 문자열의 차례가 다르다고 결정할 수 없다는 점을

알게 된다.

ISO/IEC 14651은 부호값을 사용하지 않고, 각 글자의 가중치를 사용해서 문자열을 간추린다. 그러므로 위의 보기와 같은 문제를 쉽게 해결할 수 있다. 2.2절과 2.3절에서는 ISO/IEC 14651을 이해하는데 필요한 개념을 자세히 살펴보자.

2.2 관련 용어

2.2.1 collating symbol

ISO/IEC 14651 본문에 보면 collating symbol 은 다음과 같이 정의되어 있다.

4.3 collating symbol : a symbol used to specify weights assigned to a collating element

collating symbol은 ISO/IEC 14651에서 아주 중요한 개념으로, 글자를 비교하여 순서를 결정할 때 사용하는 상징(기호)이다. collating symbol은 실제 입력된 글자를 대신해서 비교할 때 사용하는 상징인데, 이 상징의 순서가 결정되어 있기 때문에 글자의 부호값과 무관하게 글자를 비교할 수 있다.

이제 각 수준 (ISO/IEC 14651에 나오는 수준 수는 1에서 4까지 모두 네 개이다)에서 쓰는 collating symbol 보기를 살펴보자.

- **첫째 수준** : 보통 글자가 나온다. 현재 Common Template Table 에 있는 첫째 수준의 collating symbol 은 모두 <Sxxxx> 꼴이다. 아마도 Sxxxx에서 S는 symbol에서 온 듯하다. 아래에서 <Sxxxx>..<Syyyy>는 범위를 나타낸다. 한글의 경우, 첫가끝 조합형, UCS 완성형 등이 모두 들어가 있다.
- **둘째 수준** : 보통 글자의 발음 구별 기호(Diacritical mark)가 나온다. 현재 Common Template Table에 있는 둘째 수준의 collating symbol 은 모두 <Dxxxx>꼴이다.
- **셋째 수준** : 보통 글자의 모양을 나타내는 기호가 나온다.
- **넷째 수준** : 보통 글자 그 자체 혹은 글자 전체 기호가 나온다.

첫째 수준의 collating symbol 보기가 아래에 나온다.

```
% First-level collating symbols
<S0009>..<S327F> % Alphabetics, syllabics, general symbols
<S4E00>..<S9FA5> % Symbols for Han
<SAC00>..<SD7A3> % Symbols for Hangul(weights must be
  constructed)
<SFA0E>..<SFA29> % Symbols for Compatibility Han
<SFFFC>..<SFFFD> % Special symbols
<SFFFF> % Guaranteed largest symbol value. Keep at end of
  this list
```

한글의 경우에는 첫째 수준을 제외한 나머지는 대부분의 경우에 의미가 없기 때문에 이 연구에서는 첫째 수준만 정의하여 로캘을 만든다.

2.2.2 collation (weighting) table

ISO/IEC 14651 본문에 보면 collation(weighting) table 는 다음과 같이 정의되어 있다.

4.4 collation (weighting) table : a mapping from collating elements to weighting elements

collation table 은 글자의 차례와 문자열에서 글자를 읽어 들이는 방향(보기를 들어 "보기"라는 단어에서 글자 "보"를 먼저 혹은 "기"를 먼저 읽는지의 순서)을 기술하는 부분이다. 남쪽 혹은 북쪽의 가나다 차례를 지원하기 위해서는 로캘 중에서 collation table에 해당하는 부분의 내용을 수정하여 collating symbol의 순서를 바꾸면 된다. 왜냐하면 ISO/IEC 14651을 지원하는 라이브러리는 부호계와 collation table을 분리하므로 부호계는 문제가 되지 않기 때문이다. 한글의 경우 네 수준을 모두 쓰지 않고 제 1 수준만 비교하여 순서를 정할 수 있기 때문에 collation table이 아주 단순하다.

2.2.3 ordering = collation

ordering과 collation은 뜻이 같으며, ISO/IEC 14651 본문에 보면 다음과 같이 정의되어 있다.

4.8 ordering : a process by which two strings are determined to be in exactly one of the relationships of less than, greater than, or equal to one another

사실 ISO/IEC 14651의 근본적이고도 마지막 목표는, 주어진 문자열 두 개의 순서를 정하는 것이다.

2.2.4 (collation) level((견줌) 수준)

어떤 글자(U+xxxx)가 있으면 ISO/IEC 14651에서는 이를 수준의 수만큼의 subkey(견줌 수준에 대응하는 부분 키(=열쇠))를 만든다. level의 수는 ISO/IEC 14651에서 현재 default 값이 4이며, 일반적으로 적어도 3이어야 한다.

ISO/IEC 14651 본문에 보면(collation) level은 다음과 같이 정의되어 있다.

4.7 (collation) level : the sequence number for a subkey

현재 Common Template Table에는 수준이 네 개 있으며, 각 수준은(level) 1, 2, 3, 4로 부른다. 견줌 수준에 대한 설명은 2.3절에서 다시 하겠다.

2.2.5 (collation) weight

ISO/IEC 14651 본문에 보면 weight(또는 collation weight

라고도 함)는 다음과 같이 정의되어 있다.

(collation) weight : a positive integer value, used in subkeys, reflecting the relative order of collating elements

2.2.6 collating element

ISO/IEC 14651 본문에 보면 collating element 는 글자의 순서를 결정할 때 한 개의 글자로 사용할 글자의 묶음을 정의한다. 현재 배포되는 collation table에서는 collating element로 <Uxxxx>, <Uxxxx_yyyy>, <Uxxxx_yyyy_zzzz> 꼴 세 가지가 있다.

그런데 Common Template Table에 보면 <Uxxxx> 꼴은 collating element로 나와 있지 않으며, <Uxxxx_yyyy>, <Uxxxx_yyyy_zzzz> 두 가지 꼴만 collating element로 나와 있다. <Uxxxx> 꼴은 굳이 Common Template Table에 적지 않아도 알 수 있는 것이기 때문에 나타내지 않는다.

Common Template Table에서 <Uxxxx_yyyy>, <Uxxxx_yyyy_zzzz> 꼴의 collating element 보기 몇 개가 아래에 나와 있다.

```
<U0413_0301> from "<U0413><U0301>" % decomposition of
  CYRILLIC CAPITAL LETTER GJE
<U0406_0308> from "<U0406><U0308>" % decomposition of
  CYRILLIC CAPITAL LETTER YI
<U0CC6_0CC2_0CD5> from "<U0CC6><U0CC2><U0CD5>" %
  decomposition of KANNADA VOWEL SIGN OO
```

한글 경우 collating element와 관련 있는 것은 겹글자(두 겹글자, 세 겹글자)이다.

영어의 경우, aerobic 과 arm 두 낱말의(사전에 나오는) 차례를 정하는 과정을 보면, 처음 a-a는 같고, 그 다음 e-r 에서 e가 r 보다 앞서기 때문에 aerobic이 arm 보다 앞선다.

그러나 한글의 경우, "가다"와 "까다"에서 "ㄲ"은 "ㄱ + ㄱ" 임에도 우리는 "ㄲ"을 하나로 보고 차례를 정하게 된다. 우리 한글의 첫소리 글자 가나다 차례는 다음과 같다.

ㄱ - ㄲ - ㄴ - ㄷ - ㄸ - ...

만일 "가다"와 "까다"를 영어처럼 비교한다면, 처음 ㄱ-ㄱ 은 같고, 그 다음 ㅏ-ㄲ에서 ㄱ이 앞서기 때문에 "까다"가 "가다" 보다 앞서야 한다. 그러나 한글에서는 실제로 ㄱ-ㄲ 을 견주어 ㄱ이 ㄲ 보다 앞서기 때문에, 우리는 "가다"가 "까다" 보다 앞세운다.

한글 겹글자(특히 옛한글 겹글자) 처리를 위하여 collating element를 활용하는 가능성에 대하여는 앞으로 더 연구가 필요한 부분이다.

2.3 비교 과정 설명

ISO/IEC 14651은

① 비교할 문자열을 간추리기 요소(collating element) 로 바꾸고,
② 그 정의에 따라 각 견줌 수준별로 간추리기 상징(collating symbol)을 이어서 부분 키를 만들고,
③ 부분 키를 이어서 간추리기 키를 만들어 낸다.

이 개념은 이렇게 개념만 듣고는 이해하기가 힘들기 때문에 보기를 들어서 알아보도록 하자.

두 문자열 "가다", "간다"의 순서를 결정한다고 가정하자. 그러면 아래와 같이 두 부호값 뭉치가 입력될 것이다(입력되는 문자열은 EUC-KR 부호계를 사용한다고 가정한다. 각 부호값에 대한 간추리기 요소는 부호값 옆의 괄호 안의 내용과 같다고 정의한다.).

"가다" : 0xB0A1 0xB4D9 (<UAC00> <UB2E4>)
"간다" : 0xB0A3 0xB4D9 (<UAC04> <UB2E4>)

각 간추리기 요소는 간추리기 표에서 미리 정의된 내용에 따라 간추리기 상징으로 바뀌어서 견줌 수준별로 부분 키를 생성한다. 한글은 첫째 견줌 수준만을 사용하기 때문에 다음과 같이 키를 생성한다.

"가다" : <SAC00><SB2E4>
"간다" : <SAC04><SB2E4>

각 키의 간추리기 상징은 다시 가중 요소(Weight Element) 로 바뀌며, 이 가중 요소를 비교하여 문자열의 순서를 결정한다.

참고. 각 가중 요소의 값은 ISO/IEC 14651 문서에는 정의되어 있지 않으며 구현할 때 알맞게 정하면 된다. 지은이가 ISO/IEC 14651 문서를 검토해본 결과, 실제 구현에서는 보통 정수를 쓸 것으로 보인다.

개념상 전체적인 흐름은 위에서 본 바와 같지만, 실제 함수 구현은 각자 알아서 하게 된다. 구현을 어떻게 하든 문자열을 다른 상징 혹은 수로 변환하여 비교할 것으로 보인다. 또한 collation table에서 글자의 차례만 바꿈으로서 한글 가나다 차례를 쉽게 바꿀 수 있기 때문에 간추리는 프로그램을 고치지 않아도 된다는 장점이 있다.

3. 문자열 간추리는 시스템 개발

3.1 북쪽 가나다 차례 지원하는 LC_COLLATE 로캘 만들기

locale(로캘)은 "지역에 따라 바뀌는 사항"을 정의한다. 보

기를 들어, 우리는 보통 2001.8.10 처럼 연월일 순으로 적지만, 영국은 10 Aug., 2001처럼 일월년의 차례로, 미국은 Aug. 10, 2001 월일년의 차례로 적는다. 만일 이와 같이 지역에 따라 다르게 나타내는 사항 때문에 각 지역에서 쓸 프로그램을 일일이 바꾼다면(작성한다면) 엄청나게 일이 많아진다. 이런 불편을 피해 가기 위하여, 지역에 따라 다른 사항은 로캘(변수 또는 파일이라고 볼 수 있다)에 넣어두고, 프로그램은 하나만 만든 뒤, 사용자가 로캘을 지정하면, 프로그램은 그 로캘에 따라 그 곳의 관습에 맞게 처리하게 된다. 따라서 곳에 따라 프로그램을 일일이 만드는 것이 아니라, 프로그램은 하나만 있되 필요한 경우 로캘만 바꾸면 된다.

이제 남북의 한글 가나다 차례가 다른 문제도 로캘 정의하여 어떻게 해결할 수 있는지 살펴보자. LC_COLLATE는 글자의 순서를 정의하는 로캘인데, 남쪽 가나다 차례를 원하면 남쪽 가나다 차례를 LC_COLLATE 로캘에 넣어둔 뒤 ISO/IEC 14651에 따른 간추리기를 지원하는 strcoll()/strxfrm() 함수를 부르면 쓰면 남쪽 가나다 차례대로 간추릴 수 있으며, 북쪽 가나다 차례를 원하면 북쪽 가나다 차례를 LC_COLLATE 로캘에 넣어둔 뒤 strcoll()/strxfrm() 함수를 부르면, 프로그램을 전혀 바꾸지 않고 북쪽 가나다 차례대로 간추릴 수 있다.

이 연구는 간추리기에 대한 것이기 때문에 로캘 가운데 간추리기에 관련된 LC_COLLATE 로캘에 대해서만 살펴보겠다.

참고 : LC_COLLATE 말고도 다음과 같은 로캘이 있다.

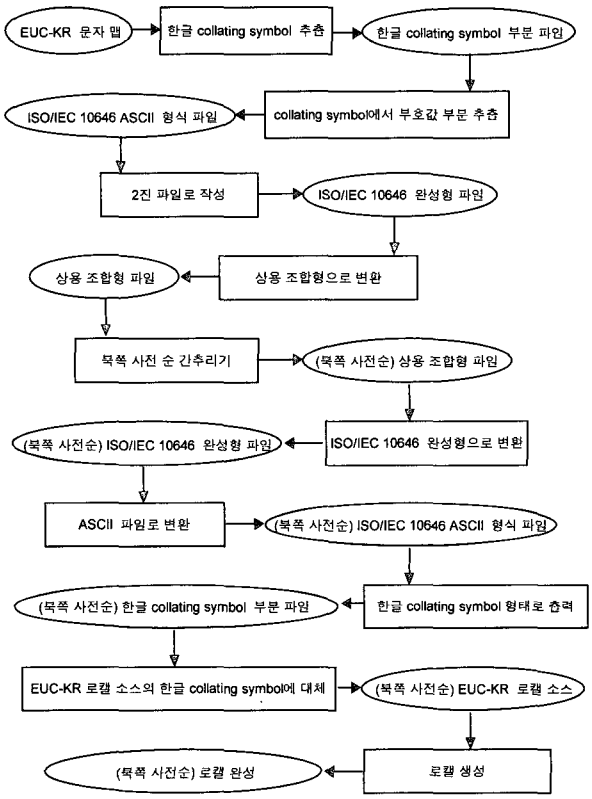
- *LC_ADDRESS*
- *LC_IDENTIFICATION*
- *LC_MONETARY*
- *LC_PAPER*
- *LC_MEASUREMENT*
- *LC_NAME*
- *LC_TELEPHONE*
- *LC_CTYPE*
- *LC_MESSAGES*
- *LC_NUMERIC*
- *LC_TIME*

북쪽의 가나다 차례를 지원하는 로캘을 만들기 위해서는 북쪽의 부호계인 EUC-KP를 사용해서 간추리기 상징을 정의하는 것이 자연스러운 방법이다. 그러나 남쪽에서는 북쪽의 EUC-KP 부호계를 지원하는 프로그램이 잘 없고, 또한 EUC-KP 폰트가 잘 없기 때문에 EUC-KP 부호계로 된 결과를 쉽게 볼 수 있는 방법이 마땅치 않다.

따라서 이 연구에서는 EUC-KP 부호계에 있는 한글 글자마디 2679개를 다 지원하지 않고, EUC-KR 부호계와 EUC-KP 부호계에 공통으로 있는 2350개의 한글 글자 마디만 지원하는 북쪽 로캘을 만들었다. 요약하면, 남쪽에서 사용하는 EUC-KR 부호계를 기반으로 북쪽의 가나다 차례를 반영한 로캘을 만들었다.

3.2 북쪽 가나다 차례를 지원하는 로캘 소스 만들기

로캘 소스를 손으로 모두 입력할 수 있지만, 부호계 관련 프로그램을 사용하면 쉽고 빠르게 개발할 수 있다. 북쪽 가나다 차례를 지원하는 로캘을 개발하는 과정을 그림으로 요약하면 (그림 1)과 같다.



(그림 1) 북쪽 로캘 소스 생성 과정의 개요

로캘 소스에서 한글 간추리기 상징은 <Uxxxx> 꼴로 표현되는데, 이 중 xxxx 부분은 ISO/IEC 10646의 한글 완성형 부호 값과 일치한다. 그래서 이전에 개발된 한글 부호계 변환 프로그램과 간추리기 프로그램을 사용한다. 로캘을 만들 때는 로캘의 내용을 정의하는 로캘 소스와 로캘에서 사용하는 문자를 정의하는 문자 맵을 사용하는데, 부호계는 EUC-KR을 사용하기로 하였으므로 북쪽 사전 순서에 맞는 로캘 소스를 만들도록 한다.

① 한글 간추리기 상징(collating symbol) 추출

ko_KR 로캘 소스에서 한글 간추리기 상징을 추출하는 방

법과 EUC-KR 문자 맵에서 한글 간추리기 상징을 추출하는 방법이 있는데, 본 논문에서는 후자의 방법을 사용한다. 왜냐하면 ko_KR 로캘 소스에는 한글 부분이 범위로 지정되어 있어서 추출할 수 없기 때문이다.

```
<UAC00> /xb0/xa1 HANGUL SYLLABLE KIYEOK-A
<UAC01> /xb0/xa2 HANGUL SYLLABLE KIYEOK-A-KIYEOK
<UAC04> /xb0/xa3 HANGUL SYLLABLE KIYEOK-A-NIEUN
.....
<UD799> /xc8/xfc HANGUL SYLLABLE HIEUH-I-PIEUP
<UD79B> /xc8/xfd HANGUL SYLLABLE HIEUH-I-SIOS
<UD79D> /xc8/xfe HANGUL SYLLABLE HIEUH-I-IEUNG
```

(그림 2) EUC-KR 문자맵 중 한글 부분

```
<UAC00>
<UAC01>
<UAC04>
.....
<UD799>
<UD79B>
<UD79D>
```

(그림 3) 한글 간추리기 상징 부분 추출

② collating symbol에서 부호값 추출과 2진 파일로 작성

collating symbol은 ISO/IEC 10646 완성형 한글 부호값을 사용하기 때문에 이를 이용하기 위해 한글 간추리기 상징 부분에서 부호값 부분을 추출한다. 한글 부호계 관련 연구에서 이미 제작한 부호계 변환 프로그램을 사용하기 위해서 2진 파일로 저장한다. 2진 파일로 저장할 때 IBM PC에 맞게 Little Endian으로 저장해야 한다.

③ 상용조합형으로 변환

북쪽 사전 순으로 부호계를 간추리기 위해서는 낱글자 단위로 글자를 비교해야 하므로, 낱글자 부호값의 조합으로 완전한 글자의 부호값을 만들어 내는 상용 조합형으로 바꾼다. 참고로 상용 조합형 부호계는 2바이트로 구성하는데, 2바이트 중 첫 번째 비트는 한글임을 표시하고 나머지 15비트를 첫 소리, 가운데 소리, 끝소리로 5비트씩 사용하여 글자를 표현한다.

④ 북쪽 사전 순 간추리기

한글 낱글자 부호값을 북쪽 사전 순으로 저장한 배열을 이용해서 각 완성된 글자의 순서를 결정하였다. 글자 순서를 결정하는 방법은 (그림 4)과 같다.

```
int init = { "ㄱ"의 부호값, …, "ㅇ"의 부호값} ;
int mid = { "ㅏ"의 부호값, …, "ㅣ"의 부호값} ;
int end = { "  ㄱ"의 부호값, …, "ㅆ"의 부호값} ;

int compare( unsigned short c1, unsigned short c2 )
{
    if( find_init(c1) > find_init(c2) )
       return 1 ;
    else if( find_init(c1) < find_init(c2) )
       return -1 ;
    else if( find_mid(c1) > find_mid(c2) )
       return 1 ;
    else if( find_mid(c1) < find_mid(c2) )
       return -1 ;
    else if( find_end(c1) > find_end(c2) )
       return 1 ;
    else if( find_end(c1) < find_end(c2) )
       return -1 ;
    else
       return 0 ;
}

int find_init(unsigned short c1)
{
     return find_idx(init 배열, c1의 초성) ;
}

int find_mid(unsigned short c1)
{
     return find_idx(mid 배열, c1의 중성) ;
}

int find_end(unsigned short c1)
{
     return find_idx(end 배열, c1의 종성) ;
}
```

(그림 4) 한글 간추리기 Pseudo 코드

⑤ ISO/IEC 10646 완성형으로 변환 및 ASCII 형태로 저장

한글 부호계 변환 프로그램을 이용해서 ISO/IEC 10646 완성형으로 바꾼 후, ASCII 형태로 저장한다.

⑥ 한글 collating symbol 형태로 출력하고 EUC-KR 로캘 소스의 한글 collating symbol 대체

로캘 소스에서 사용하는 <Uxxxx> 꼴의 한글 간추리기 상징으로 변환하여 파일로 저장한다. 이후에 EUC-KR 로캘 소스에 포함된 한글 간추리기 상징 부분을 새로 생성한 한글 간추리기 상징으로 대체한다.

⑦ 로캘 생성

로캘 생성 프로그램을 사용해서 생성한다. 로캘 생성 프로그램 사용법은 리눅스의 맨 페이지(man page)를 참고한다.

본 연구에서 생성한 전체 로캘 소스는 http://sinsi.cs.pusan.ac.kr/hangeul/linux-14651-sort/north_locale.src에서 얻을 수 있다.

3.3 글자 간추리는 프로그램 만들기

3.3.1 개발 환경

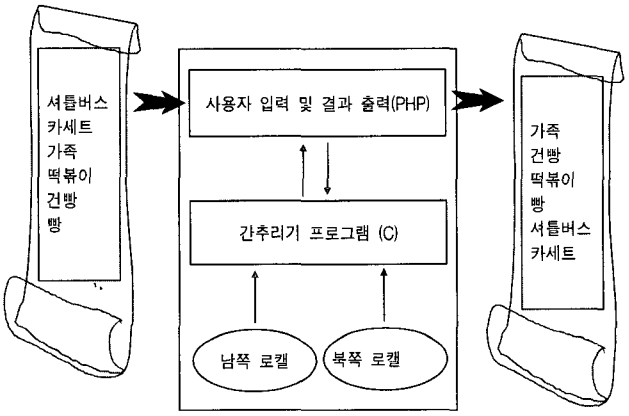
- 플랫폼 : 인텔 펜티엄3
- 운영체제 : 맨드레이크 리눅스 8.0

- 언어 : C, PHP
- 남쪽 또는 북쪽 파일 가나다 차례대로 간추리고자 하는 입력 파일 제약 사항 :
  ㄱ) 입력 파일 부호계 : EUC-KR
  ㄴ) 입력 파일 크기 : 최대 5000줄, 한 줄당 최대 200바이트

입출력 조건을 다양하게 해서 프로그램을 작성하는 것이 좋지만, 프로그램을 쉽게 작성하기 위해서 입력 파일에 위와 같은 제약을 두었다.

3.3.2 시스템 구성

전체 시스템은 (그림 5)와 같이 사용자의 입력 파일을 처리하는 부분, 입력 파일을 간추리는 부분, 남쪽과 북쪽의 로캘로 구성된다.



(그림 5) 전체 시스템 구성

① 사용자 입력 및 결과 출력(PHP)

사용자는 간추리기를 해야 할 입력 파일을 지정하고, 남쪽과 북쪽 중 원하는 가나다 차례를 정한 후 프로그램의 시작을 명령하면, 사용자가 입력한 파일을 올려서(upload) 시스템의 임시 디렉토리(유닉스 사용자 권한이 777)에 저장한다. 그리고 나서 간추리기 프로그램을 호출하여 간추릴 입력 파일을 전달하고, 간추리기 프로그램의 실행이 끝나면 간추려진 결과(출력) 파일을 브라우저에 보여준다.

② 문자열을 간추리는 프로그램(C)

ISO/IEC 14651 관련 함수인 setlocale()과 strcoll()을 이용해서 입력된 파일의 각 줄을 남쪽 혹은 북쪽 가나다 차례에 맞게 간추린 후 결과(출력) 파일로 저장한다.

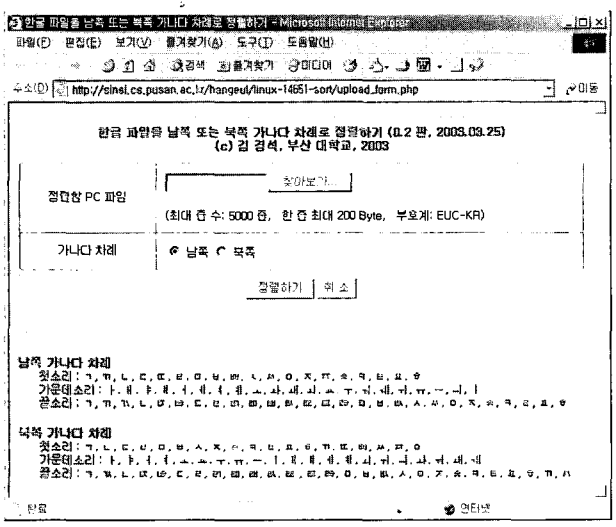
③ 남쪽과 북쪽의 로캘

리눅스 운영체제가 남쪽 로캘은 이미 만들어 공급하므로 남쪽 가나다 차례대로 간추리려면 운영 체제의 남쪽 로캘을 그대로 쓰면 되며, 북쪽 로캘은 위의 절에서 만든 것을 사용한다.

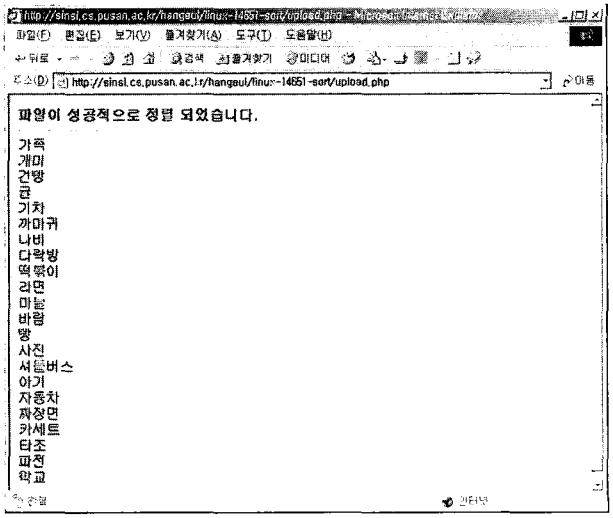
3.3.3 실행

① 웹 브라우저를 실행하고, 주소 입력창에 프로그램의 URL을 입력한다(http://sinsi.cs.pusan.ac.kr/hangeul/linux-14651-sort/upload_form.php).
② 간추릴 파일을 선택한다.
③ 찾아보기 버튼을 눌러 남쪽 또는 북쪽 가나다 차례를 선택한다.
④ 정렬하기 버튼을 누른다.
⑤ 일정 시간이 지난 후에 선택한 가나다 차례에 따라 간추려진 파일이 보일 것이다.

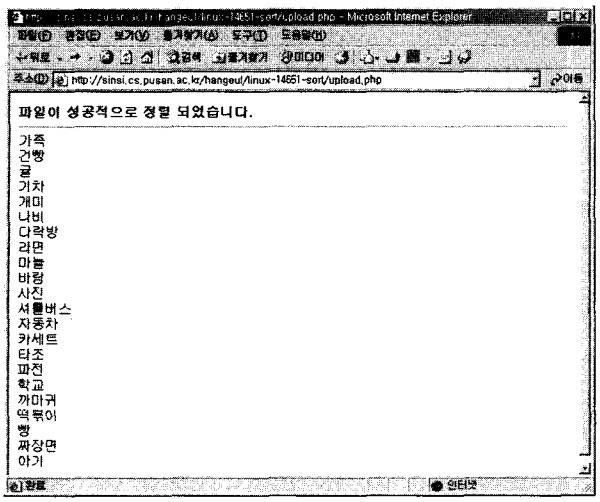
그림으로 프로그램의 실행을 실행하는 과정과 결과를 살펴보자.



(그림 6) 프로그램 실행 화면



(그림 7) 남쪽 가나다 차례로 간추린 결과



(그림 8) 북쪽 가나다 차례로 간추린 결과

3.3.4 실행 결과

3.3.3절에서 프로그램의 실행 모습과 실행 결과 화면을 보았다. 본 절에서는 시험 데이터와 남쪽과 북쪽 가나다 차례에 따른 간추린 결과를 <표 1>에서 비교하도록 하겠다.

〈표 1〉 시험 데이터와 가나다 차례에 따른 실행 결과

Input file	Output	
	남쪽 가나다 차례로 정렬	북쪽 가나다 차례로 정렬
셔틀버스	가족	가족
카세트	개미	건빵
가족	건빵	귤
떡볶이	귤	기차
건빵	기차	개미
빵	까마귀	나비
타조	나비	다락방
나비	다락방	라면
파전	떡볶이	마늘
사진	라면	바람
개미	마늘	사진
학교	바람	셔틀버스
아기	빵	자동차
자동차	사진	카세트
까마귀	셔틀버스	타조
마늘	아기	파전
기차	자동차	학교
귤	짜장면	까마귀
다락방	카세트	떡볶이
라면	타조	빵
짜장면	파전	짜장면
바람	학교	아기

4. 관련 연구

현재 ISO/IEC 14651에 관한 연구가 많지 않다. 영어권 국가에서 적용하고 있는 ISO/IEC 14651을 한글에 어떻게 적용할지에 관한 연구만 진행되었다.

[2]은 첫가끝 한글과 완성형 한글이 섞여 있는 문자열을 간추리면 첫가끝 조합형 한글이 완성형 한글보다 먼저 나오는 결과가 올바르지 않음을 지적하고, 완성형 한글을 첫가끝 조합형으로 나타낼 것을 제안하였다. 완성형 한글과 첫가끝 조합형 한글을 견줌 테이블(collating table)을 이용해서 변환하기 때문에 ISO/IEC 14651의 기본 틀을 벗어나지 않으면서 문제를 해결하고 있다. 하지만, 한글과 여러 나라 글자가 섞여 있는 상황은 고려하지 않았다. 따라서 한글과 영문자를 섞어서 사용하는 경우에 문자열을 제대로 간추리지 못하는 한계가 있다.

[3]은 끝소리 글자가 없는 첫가끝 조합형 한글과 여러 나라의 글자가 섞여 있는 경우에 문자열이 제대로 간추려지지 않는 문제를 해결하였다. "자干"와 "잔"은 첫가끝 조합형 한글로 표현하면 "ㅈ ㅏ 干"와 "ㅈ ㅏ ㄴ" 가 되는데, "ㅈ ㅏ"가 비교된 이후에 "干"와 " ㄴ"이 비교된다. 첫가끝 한글의 부호값이 한자의 부호값보다 앞서기 때문에 "잔"이 "자干"을 앞선다. 이에 전처리 과정을 추가하여 끝소리 글자가 없는 첫가끝 조합형 한글에 가상의 값을 넣어서 한글 낱자와 다른 나라 글자가 비교되지 않도록 하였다.

5. 맺음말

5.1 이번 연구의 성과

국제 표준 ISO/IEC 14651 : 2000(International String Ordering)은 여러 나라 글자계(script)가 섞여 있을 때, 모든 글자의 차례를 정하고 간추리는 틀에 관한 표준이다. 남쪽과 북쪽의 다른 가나다 차례 문제를 국제 표준을 활용하여 해결할 수 있는 좋은 방안이다.

이 연구에서 사용한 ISO/IEC 14651 관련 함수는 리눅스와 솔라리스, FreeBSD와 같은 유닉스 기반 운영체제의 최신 라이브러리에 포함되어 있으며, 오라클과 같은 데이터베이스 관리 체제에도 포함되어 있다. 이 연구에서는 오픈 운영체제인 리눅스에서 제공하는 ISO/IEC 14651 관련 함수인 strcoll()과 strxfrm()을 사용해서 남쪽과 북쪽의 가나다 차례에 따라 문자열을 간추리는 프로그램을 만들어 보았다. 개발한 프로그램을 사용해서 시험 문서를 실제로 북쪽 가나다 차례에 따라 간추려 보았으며, 그 결과를 보였다.

이 연구에서 개발한 북쪽 가나다 차례를 지원하는 로캘은, 복사하여 일반인들의 리눅스 머신에 설치하여 사용할 수 있다. 개발 환경인 인텔 기반 리눅스에서는 바이너리 호환이 되므로 로캘을 그대로 복사해서 사용할 수 있지만, 개발 환경과 다른 플랫폼의 리눅스나 유닉스 운영체제에서는 로캘 소스를 이용해서 새로 컴파일해야 하리라고 본다. 리눅스가 데스크톱 및 서버의 운영체제로 많이 보급되고 있는 시점이기 때문에 일반인들이 쉽게 사용할 수 있을 것으로 예상된다.

또한 이 연구에서 개발한 북쪽 가나다 차례에 따라 간추리는 프로그램은 웹에서 작동하기 때문에 누구나 쉽게 사용

할 수 있다.

이번 연구에서는 실제 리눅스 운영체제에서 EUC-KR을 지원하는 북쪽 로캘을 만들고 그것을 활용하는 프로그램을 개발하였다. 시험 결과 북쪽 가나다 차례를 지원할 수 있음을 보았다.

5.2 앞으로 더 연구가 필요한 사항

이번의 연구는 남쪽과 북쪽의 로캘을 개발하여 활용한 기초 연구이며, 앞으로 ISO/IEC 14651과 로캘에 대한 지속적이 연구가 필요하다고 본다. 특히 다음과 같은 사항을 연구해야 할 것이다.

① ISO/IEC JTC1/SC22/WG20(Internationalization)에서 남북이 같이 만나서, ISO/IEC 14651에 대한 공동 검토 및 연구를 수행한다.
② ISO/IEC 14651의 남북 공동 규격화를 추진한다.
③ 한글의 정규화(NF : normalization form) 및 준비(prep)에 대한 연구가 아직 많이 모자라므로, 앞으로 남북이 이에 대한 연구를 같이 추진한다.
④ 북쪽의 가나다 차례를 지원하는 로캘 ko_KP.UTF-8과 옛 한글까지 지원하는 로캘 ko_KR.UTF-8@old1을 남북이 같이 개발한다.
⑤ 겹글자(특히 옛한글 겹글자)를 지원하기 위하여 collating element를 활용하거나, collating element가 수천일 경우, 일일이 이를 적지 않고 간단하게 나타낼 수 있는 새로운 개념을 ISO/IEC 14651에 넣는 방안을 연구해볼 필요가 있다.

(후 기)

저자(김 경석)는 1994년부터 약 10년 동안 10여 차례에 걸쳐서 북쪽 전문가들을 만나보았다. 다른 일부 분야에서는 공개적으로 공동 연구를 하기도 하지만 아직 정보 기술 표준 분야에서의 남북 교류는 걸음마 수준이다.

저자는 처음에는 중국에서 열린 남북 한글 전산화 관련 학자 모임에서 북쪽 전문가들을 만나다가, 2000년대에 와서는 국제 표준화 기구(ISO) 회의에서도 북쪽 전문가들을 만날 수 있었다.

북쪽은 국제 표준화 기구 안에서도 JTC1/SC2(글자 부호계) 분야에 주로 참여하며, 국제화(JTC1/SC22/WG20), 로마자 적기(TC46)에도 관심은 있지만 현재 참여는 하지 않고 있다.

북쪽 전문가들을 만나서 얘기를 할 때는 서로에게 어려움이 생기지 않는 범위 안에서만 서로 협조할 수 있기 때문에 늘 조심스럽다.

이 논문에서 다룬 문제는 북쪽이 많은 관심을 가지고 있는 주제이다. ISO/IEC 10646 국제 표준에 한글 글자마디와 한글 글자의 차례가 모두 남쪽 가나다 차례대로 들어가 있는데, 북쪽은 이를 북쪽 차례 또는 제3의 차례대로 다시 배열하자고 주장하였다. 그러나 이는 받아들여지지 않았고, 그 대신 북쪽 가나다 차례대로 간추리는 것은 ISO/IEC 14651을 활용할 수 있다는 결론이 나왔다.

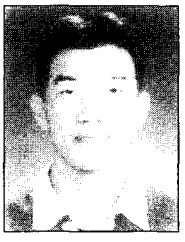
이에 저자는 ISO/IEC 14651 규격에 따라 북쪽 가나다 차례대로 간추릴 수 있는 방안에 대하여 북쪽과 토의하였고, 또한 실제로 프로그램을 개발하게 된 것이다.

현재 정보 기술 표준화 분야에서는 북쪽 전문가들을 국제 회의장에서 만나서 의견을 나누는 것이 최대한의 교류 수준으로 보인다. 저자는 국제 규격을 남북이 공동으로 한글로 번역하여 남북 공동 표준을 만들자고 여러 차례 제안하였지만 아직까지 아무런 답변이 없다.

저자는 앞으로도 지속적으로 북쪽의 정보 기술 표준 전문가들과 접촉하여 5년 또는 10년 뒤에는 정보 기술 분야에서 남북 공동 표준을 만들어 보겠다는 꿈을 가지고 연구하고 또한 지속적으로 접촉하고 있다.

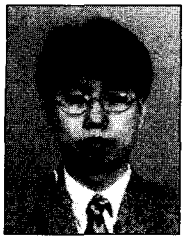
참 고 문 헌

[1] 최유경, 황호전, 안동언, 정성종, "남북한 언어 비교 사전 검색 시스템의 설계 및 구현", 정보처리학회 2001년 춘계학술대회, Vol.08, No.01, pp.0797-0800, April, 2001.
[2] 김종휘, 김경석, "ISO 14651에 의한 한글 ordering의 문제점과 그 해결 방안", 한국정보과학회 2001 가을 학술 발표논문집 (II), 제28권 제2호, pp.187-189, 2001.
[3] 옥제영, 정일동, 김경석, "ISO/IEC 14651 틀에서 한글 간추리기 문제점에 대한 해결 방안", 한국 멀티미디어 학회 추계 학술 발표 논문집, 제5권 제2호, pp.59-64, Nov., 2002.
[4] 한국표준협회, "국제 표준에 따른 남북 가나다 차례 지원 방안 연구", 한국표준협회, 2001.
[5] 정 일동, 김 경석, "북쪽 가나다 차례를 지원하는 로캘과 활용 프로그램 개발", 정보과학회 2003년 가을 학술발표논문집(I), pp.523-525, Oct., 2003.



정 일 동

e-mail : idjung@asadal.cs.pusan.ac.kr
2000년 부산대학교 전자계산학과(이학사)
2002년 부산대학교 대학원 전자계산학과 (이학석사)
2002년~현재 부산대학교 대학원 전자계산학과 박사과정
2003년~현재 LG전자 DAC연구소 주임 연구원
관심분야 : 정보가전, 인터넷 컴퓨팅, 임베디드 시스템 등

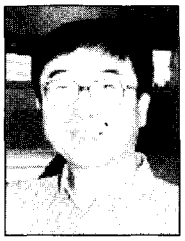


이 중 화

e-mail : jhlee@asadal.cs.pusan.ac.kr
1992년 부산대학교 전자계산학과(이학사)
1995년 부산대학교 전자계산학과(이학석사)
2001년 부산대학교 전자계산학과(이학박사)
2002년~현재 동의대학교 소프트웨어공학과 조교수
관심분야 : 데이터베이스, XML, 시맨틱 웹 등

김 용 호

e-mail : yhkim@asadal.cs.pusan.ac.kr
1993년 부산대학교 전자계산학과(이학사)
1995년 부산대학교 전자계산학과(이학석사)
2001년~현재 부산대학교 전자계산학과 박사과정
1995년~현재 한국기계연구원 자본재정보기술그룹 선임연구원
관심분야 : 인터넷 응용, 홈 네트워크, 정보가전, 데이터베이스 등



김 경 석

e-mail : gimgs2004@asadal.cs.pusan.ac.kr
1977년 서울대학교 무역학과(경제학사)
1979년 서울대학교 전자계산학과(이학석사)
1988년 일리노이 주립대 (어바나-샴페인) 전자계산학 박사
1988년~1992년 미국 노스다코타 주립대학교 전자계산학과 조교수
1992년~현재 부산대학교 전자전기정보컴퓨터공학부 교수
관심분야 : 데이터베이스, 멀티미디어, 한글/한말 정보처리, 인터넷 컴퓨팅 등