애플리케이션 기반에서 Punycode를 적용한 다국어 이메일 주소 제안 및 구현

박민수*, 이희찬, 송관호, 신용태
숭실대학교
e-mail : {mspark*, hclee}@cherry.ssu.ac.kr
{khsong, shin}@ssu.ac.kr

The Proposal and Implementation of The Internationalized Email Address applying Punycode in Application Layer

Min-Su Park*, Hee-Chan Lee, Kwan-Ho Song, Yong-Tae Shin
Soong-Sil University

요 약

새로 제정된 이메일 프로토콜 표준에 따라 사용자는 이메일 주소에 한글과 같은 다국어를 포함할 수 있다. 다국어 이메일 주소를 사용하여 송수신자가 메일을 전송하기 위해서는 양측 메일 서버가 모든 문자를 표현할 수 있는 UTF-8 인코딩 방식을 지원하도록 정의하고 있지만 현재의 네트워크 상에는 UTF-8 인코딩 방식을 지원하는 메일 서버와 지원하지 않는 메일 서버 모두가 존재하고 있다. 이는 곧 메일이 정상적으로 수신자에게 전송이 되지 못하는 결과를 발생시킨다. 본 논문에서는 UTF-8 을 지원하는 서버와 지원하지 않는 서버의 기존 상태를 유지하면서 메일을 정상적으로 송수신을 하기 위해 애플리케이션에서 퓨니코드 변환을 적용시킨 전달 기법을 제안하였으며 직접 구현하여 정상적으로 작동하는 것을 확인 하였다.

1. 서론

IETF 의 EAI(Email Address Internationalization) 워킹 그룹은 다국어를 적용한 이메일 프로토콜 표준 제정 활동을 진행 하여 2010 년까지 차례로 기본적인 이메일 프로토콜인 SMTP 와 POP3 그리고 IMAP 의 RFC 국제 표준을 새롭게 제정 하였다[1][2][3].

해당 국제 표준들은 기존의 SMTP 와 POP3 등의 프로토콜을 확장한 형태로서 다국어를 표현하기 위해서 UTF-8 인코딩 방식을 적용하였다. 따라서 새로 제정된 표준에 따라 다국어 이메일 서비스를 제공하기 위해서 메일 서버는 UTF-8 인코딩이 가능한 프로그램을 설치하거나 교체 해야 하는 요구사항이 있다. 이러한 방식은 사용자 측면에서 봤을 때 어떠한 애플리케이션 변경이 없이 다국어 이메일을 사용할 수 있지만 서비스 제공자 측면에서는 프로그램 혹은 서버 교체시 사용자 정보 손실 또는 그로 인한 예측할 수 없는 문제점이 발생할 수 있다. 따라서 다국어 이메일을 제공하는 메일 서버의 교체는 장기적이 될 수 밖에 없으며 네트워크에는 UTF-8 인코딩 방식을 지원하는 메일서버와 지원하지 않는 메일 서버가 동시에 존재할 수 있게 된다.

따라서 본 논문에서는 UTF-8 을 지원하는 메일 서버와 지원하지 않는 메일 서버가 네트워크에 동시에 존재함으로써 발생할 수 있는 전송 실패 가능성의 문제를 해결하기 위한 방안으로 메일 서버를 기존 상태로 유지 시키고 신뢰적인 다국어 이메일 전송을 하기 위해 애플리케이션 기반에서 퓨니코드 변환 기법을 적용시킨 다국어 이메일 전송 기법을 제안한다.

2. 관련 연구

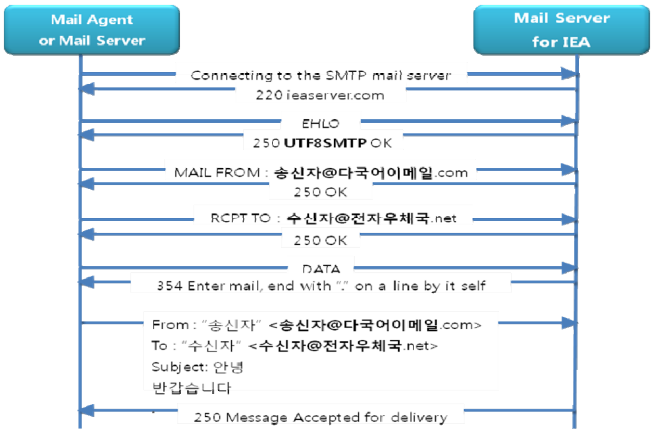
2.5 IEA 지원 SMTP 프로토콜

IEA 지원 SMTP 프로토콜 표준은 RFC 5336 에서 명세하고 있으며 기존의 SMTP 프로토콜과의 가장 큰 차이점은 ASCII 가 아닌 UTF-8 형식의 다국어 문자열을 메일 메시지의 Envelope, Mail header, 그리고 Body 에 직접 입력할 수 있다는 점이다. 다만 UTF-8 을 지원하는 서버는 접속을 요청하는 상대 서버나 사용자 에이전트에게 UTF-8 지원 여부를 알려야 한다.

IEA 지원 SMTP 프로토콜은 ELHO 키워드를 받으면 8BITMIME 등을 포함하여 UTF8SMTP 키워드를 접속을 요청한 사용자 에이전트나 메일 서버에게 반환하여 UTF-8 인코딩 방식 지원 여부를 알린다.

<그림 2>는 IEA 지원하는 SMTP 프로토콜 수행과정을 나타낸다. 기존 SMTP 프로토콜과 비교 했을 때 Envelope 뿐만 아니라 DATA 에 해당하는 메일 해더와 본문에 직접 다국어를 입력 할 수 있으며 별도의

처리 없이 UTF-8 방식으로 처리 되는 차이점을 가지고 있다.



<그림 1: IEA 지원 SMTP 프로토콜 동작 과정>

2.6 퓨니코드(Punycode)[4]

퓨니코드는 ACE(ASCII Compatible Encoding)형식 중 하나로서 기본적으로 비 ASCII 문자에서 ASCII 문자로의 변환을 수행하며 문자열 앞에 "xn--"접두사를 가지고 있는 특징을 가지고 있다.

퓨니코드 인코딩이 수행되기 위해서는 특정 국가의 표준 문자 코드로 입력된 문자열은 전부 유니코드로 전환되어야 한다. 그리고 퓨니코드 인코딩이 수행되기 전에 정규화 과정[5]을 통해 인코딩이 올바르게 수행 될 수 있기위한 처리를 한다. 정규화 과정은 마지막으로 유니코드는 퓨니코드 표준 문서인 RFC 3492 에 명세되어 있는 toASCII() 작업을 통해 아스키로 변환되며 toUNICODE()작업을 통해 다시 유니코드로 복구 시킬 수 있다.

3. 퓨니코드 적용 다국어 이메일 서비스

현재의 다국어 이메일 서비스의 가장 큰 이슈는 네트워크 상에 UTF-8 를 지원하는 메일서버와 지원하지 않는 메일서버가 동시에 존재할 수 있다는 점이다. 따라서 메일이 정상적인 방법으로 전송되지 않을 수 있다. 본 절에서는 이 같은 문제를 완벽하게 해결 하기 위하여 메일 서버를 기존상태로 유지하고 애플리케이션에서 퓨니코드를 적용하여 다국어 이메일 서비스를 제공할 수 있는 기법을 제안하고 실제적으로 구현하여 이를 실행 결과를 통해 증명한다.

3.1 제안 기법 개요

제안하는 기법은 이러한 다국어 도메인 서비스의 구조적인 특징을 다국어 이메일 서비스에 적용시켜 UTF-8 지원 메일서버와 ASCII 전용 메일 서버간에 발생할 수 있는 메일 전송실패의 대안을 마련한다.

따라서 제안 기법은 기본적인 구조는 애플리케이션에 속하는 사용자 에이전트를 제외하고 서버와 사용하는 프로토콜 모두 동일하다. 사용자 에이전트는 기존의 메일 작성 및 전송 기능에 로컬 문자(해당 국가의 국가코드를 사용하는 문자 ex)EUC-KR)를 유니코드로 변환시키는 기능과 정규화, 그리고 퓨니코드 인코딩 및 디코딩 기능이 추가되었다.

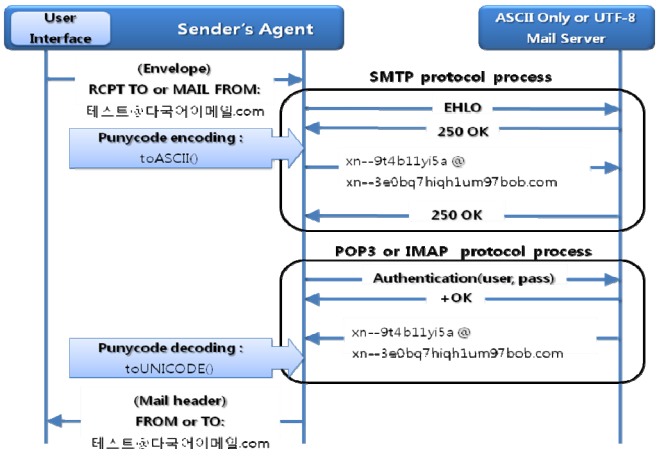
ASCII 전용 메일 서버는 기존의 기능을 가지며 UTF-8 지원 메일 서버는 수신한 이메일 주소가 퓨니코드 형태일 경우 이를 toUNICODE() 작업을 통해 원래의 UTF-8 형태의 메일 ID 와 매칭할 수 있다고 가정한다.

그리고 기존의 SMTP 프로토콜은 RFC5336 을 따르며 나머지는 동일하다.

3.2 제안 기법 동작 과정

애플리케이션에서 모든 다국어 도메인 처리가 이루어지기 때문에 메일 서버간의 트랜잭션 처리는 중요하지가 않다. 사용자 에이전트 중심으로 처리 흐름을 보면 먼저 송신자가 UI(User Interface)를 통해서 Envelope 에 주소를 입력한다. 만약 주소가 다국어 이메일 주소일 경우 에이전트는 송신자 메일 서버에 전송하기 전에 퓨니코드의 toASCII()을 통하여 ASCII 형태의 주소로 변환시킨다. 그 다음 다국어 메일은 송신자 메일서버의 UTF-8 지원 여부에 상관없이 전송이 된다(송신자 메일 서버가 UTF-8 을 지원하는 경우 서버는 UTF-8 형태의 계정 ID 를 얻기 위해 임시적으로 toUNICODE()를 사용한다). 이 후의 과정은 기존 SMTP 프로토콜과 동일하다.

사용자 에이전트가 메일 서버의 메시지를 수신하는 경우에는 POP3 또는 IMAP 프로토콜을 사용한다. 에이전트는 수신한 메일의 헤더에 퓨니코드 형태의 이메일 주소가 존재 할 경우 퓨니코드의 toUNICODE()작업을 통해 원래의 다국어 이메일 주소로 변환 시킨다. 최종적으로 에이전트는 화면상에 다국어 이메일 주소를 출력시켜 사용자에게 다국어 이메일 서비스를 제공한다.



<그림 2: 퓨니코드 인코딩 기반 SMTP 확장 기법 동작 과정>

4. 제안 기법 구현

전체 시스템은 MUA main 모듈과 Punycode 변환 모듈 그리고 Display 모듈의 핵심 요소로 구성되어 있다. MUA main 모듈은 일반적인 메일 클라이언트 작업을 수행하며 내부적으로 전달되는 이메일 주소값을 검사하여 변환을 위해서 Punycode 변환 모듈을 호출 한다.

만약 입력된 이메일 주소가 비 아스키 문자일 경우 Punycode 변환 모듈의 toASCII() 함수를 호출 하여 접두사 xn- 형태를 가진 주소로 변환 시키고 메시지 보관함 열람 또는 Reply 와 같은 작업을 통해 출력하려는 이메일 주소가 xn- 형태를 가진 주소일 경우에 toUNICODE() 함수를 사용하여 다시 원래의 다국어 주소로 변환 시킨다.

이때 메일 서버에 생성한 아이디를 이미 toASCII() 함수 처리 이후의 xn- 형태로 등록을 하였기 때문에 MUA main 모듈은 메일 서버와 클라이언트 사이에서의 통신은 UTF-8 을 지원하지 않는 메일 서버와 기존의 SMTP 프로토콜을 사용하여 수행 할 수 있다.

4.3 실행 결과

먼저 제안 서비스를 제공 하기 위해 다음의 순서에 따라 환경을 구축한다.

① 메일 서버와 데이터 베이스 설치

메일서버로 현재 윈도우 기반 공개 소프트웨어인 hMailServer 를 설치하고 데이터 베이스로 MySQL 을 설치 한다.

② 메일 서버에 다국어 계정을 생성

메일 서버에는 미리 xn--zb0b93vlxt@frars.co.cc 와 xn--989aj00b65m@frars.co.cc 의 두개의 다국어 이메일을 생성한다. 이 계정들은 각각 '관리팀' 과 '설계팀' 이라는 다국어와 맵핑된다.

③ 퓨니코드 적용 MUA 설치

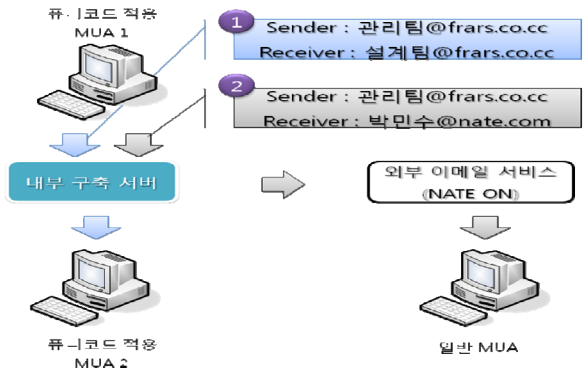
생성한 계정에 각각 접속하여 메일을 확인 할 수 있도록 두 개의 클라이언트에 퓨니코드 적용 MUA 를 설치한다. MUA 의 초기화면은 <그림 : >과 같다.

④ 외부 메일 서비스 계정 생성

제안 서비스가 다른 외부 포탈에서 제공하는 메일 서비스와 상호 작동이 가능한지 알아보기 위해 xn-lg3bvb24y 의 이름으로 계정을 생성한다. 해당 계정은 다국어로 '박민수' 와 맵핑 된다. (외부 메일 서비스 중 계정 명에 하이픈 문자를 지원하는 nate.com 서비스를 사용)

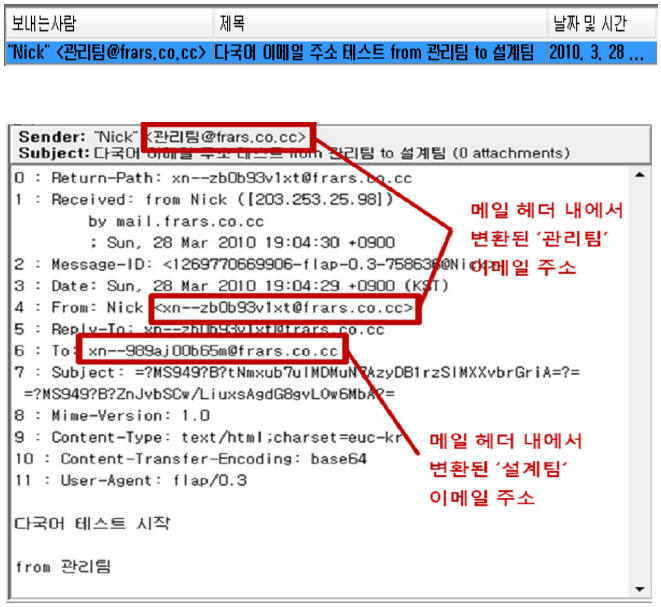
테스트 시나리오로 다음 그림 과 같다. 첫번째, 관리팀 계정에서 다국어 주소를 사용하여 마찬가지로 설계팀의 다국어 주소로 메일을 전송 하여 전송 여부를 확인 한다.

두 번째로 외부 메일 서비스와 동작 여부를 확인하기 위해서 관리팀 계정으로 직접 '박민수' 형태의 다국어로 메일을 전송 후 수신 확인 한다.



• 관리팀의 계정을 사용하여 설계팀으로 메일 전송

<그림 8:다국어 이메일 주소 사용 메일 전송 테스트>



<그림 9: 전송 테스트 실행 결과>

아이디 '설계팀' 계정의 메시지 함에 메일이 전송된 것을 확인 할 수 있다. 6 번의 메일 헤더 필드에서 확인 할 수 있듯이 설계팀@frars.co.cc 이메일 주소가 내부적으로 xn--989aj00b65m@frars.co.cc 주소로 변환되어 처처리된 것 알 수 있다. 또한 프로그램이 이메일 주소를 출력하기 위해 xn- 형태를 디코딩 하

여 Sender 의 상태 창에 정상적으로 원래의 다국어 형태인 '관리팀' 이 표시 되는 것을 알 수 있다.

· 외부 메일 서비스 전송 테스트

<그림 10: 외부 메일 서비스로 메일 전송 테스트>

<그림 11: 외부 메일 서비스 전송 테스트 결과>

외부 포탈로 전송하기 위해 다국어 이메일 주소가 정확하게 xn- 형태로 인코딩 되어서 전송된 것을 확일 할 수 있다. 단 포탈 서비스에는 xn- 으로 인코딩된 형태가 순수하게 계정 아이디로 사용되기 때문에 변환하지 않고 직접적으로 출력하고 있다.

5. 결론

본 논문에서는 애플리케이션 기반인 다국어 도메인 서비스의 특징을 이용하여 다국어 이메일 서비스를 제공하는 기법을 제안 하였다. 또한 제안된 정의와 설계를 바탕으로 직접 시스템을 구현하여 구축된 메일 서버뿐만 아니라 외부 메일 서버를 통해서 시스템이 정상적으로 동작하는지 확인 하였다. 테스트 결과에서는 직접적으로 xn- 형태로 인코딩된 주소가 노출 되었으나 이는 올바른 결과를 확인 하기 위해 제거 하지 않았으며 사용자의 혼란을 막기 위해서 메일 주소가 다국어 하나만을 출력 하는 것이 요구 된다.

기본적으로 기존의 메일 서버와 프로토콜을 현 상태로 유지한 체 사용자 에이전트에서 모든 다국어 처리를 수행하는 특징을 가진다. 또한 사용자 에이전트는 메일 서버로 이메일 메시지를 전달하기 전에 다국어 이메일 주소가 있는지 검사하여 존재할 경우 이를 퓨니코드 인코딩 방식을 사용하여 ASCII 형태로 변환 시킨다.

이러한 특징은 네트워크 상에 UTF-8 을 지원하는 메일 서버와 지원하지 않는 메일 서버가 존재하여도 메일이 정상적으로 전송될 수 있도록 해준다. 즉 사용자 에이전트는 접속하려는 메일 서버의 UTF-8 지원여부에 영향 받지 않으며 메일 서버도 다국어 이메일 주소에 대한 인지 없이 기존 방식 그대로 운영될 수 있다.

그러나 결국 완전한 다국어 이메일 서비스 제공을 위해서는 UTF-8 을 지원하지 못하는 서버를 업데이트 하여 어떠한 부가적인 작업을 필요하지 않고 다국어 이메일 서비스를 제공해야 할 것이다. 그렇지만 메일 서버 업데이트 작업은 데이터 손실과 여러 위험요소의 이유로 더디게 진행될 수 밖에 없다. 따라서 본 논문에서 제안한 기법은 이런 과도기 시점에서 좀더 일찍 사용자들에게 다국어 이메일 서비스를 제공할 수 있는 발판이 될 수 있을 것이며 ASCII 문자가 아닌 다국어를 이메일에 사용 가능하게 함으로써 비영어권 국가가 가진 불리한 요소를 극복하는데 적지 않은 영향을 줄 것이다.

참 고 문 헌

[1] J. Klensin, "Simple Mail Transfer Protocol", RFC 2821, 2001.04.

[2] Abel, Y., Ed., "Internationalized Email Headers", RFC 5335, 2008.09.

[3] J. Yao and W. Mao, "SMTP Extension for Internationalized Email Address", RFC 5336, 2008.09.

[4] A. Costello, "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications(IDNA)", RFC 3492, 2003.03.

[5] P. Hoffman, M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)", RFC 3491, 2003.03.