내장형 리눅스 환경의 전자책 리더 용 자바 클래스 라이브러리 개발

이 은 정† · 조 수 선††

요 약

본 논문에서는 리눅스 환경에서의 전자책 리더를 위한 자바 라이브러리 Xeni의 개발을 소개한다. Xeni는 XML 기반의 전자책 표준을 따르는 컨텐츠를 해석하여 랜더링하는 기능과 전자책의 네비게이션 및 북마크 기능 등을 제공하는 API 라이브러리다. 본 라이브러리는 내장형 리눅스 기반의 기기에서 동작하는 자바 가상 기계 환경을 목표로 하여, 자바 언어로 개발하였다. 개발된 라이브러리의 설계와 구조를 소개하고 이 라이브러리를 이용하여 구현된 전자책 리더의 구현 예를 살펴본다.

Development of Java Class Library For E-Book Reader Systems on Embedded Linux Environment

Eunjung Lee† · Soosun Cho††

ABSTRACT

We developed a Java library Xeni for e-book reader systems on embedded Linux environment. Xeni is an API library providing functions such as rendering for XML-based e-book contents, navigation mechanism for readers and bookmark management. This library is developed in Java language, targeting java virtual machines on embedded Linux systems. We discuss design and structure of the developed library, and introduce the reference implementation of a reader system using this library. Also, virtual machines on Linux environment are briefly surveyed.

키워드 : 전자책 리더(E-book reader), 자바 API(Java API), 퍼스널자바 실행환경(Java runtime environment), XML

1. 서 론

전자책은 책이라는 인류 유산의 한 형태를 디지털 컨텐츠로 제작, 유통, 보급하기 위한 프레임워크이다. 전자책의 보급과 사용을 위해서는 전용 리더 시스템이 필수적인데, 전자책은 제작자에 의해서 생성된 컨텐츠를 리더 시스템을 통해서만 접근 가능하도록 허용함으로써 컨텐츠의 저작권을 보호하고 유통을 위한 인프라를 제공하게 된다[6, 7].

최근 전자책의 제작 및 보급이 활기를 띄면서 사용자들을 위한 전자책 리더 시스템들이 다수 개발되었다. 특히 PDA 기반의 전자책 전용 단말이 많이 보급될 것으로 기대되는데, 이런 단말 환경에서 내장형 운영체제로서 리눅스 시스템에 대한 관심이 높아지고 있다. 현재 많이 사용되고 있는 운영체제는 PalmOS나 Windows CE 등인데 이들 제품의 높은 비용 부담으로 인해 보급형 저가 단말기 제작을 위해 공개 소스인 리눅스 시스템이 큰 매력을 가지게 되었다.

한편 리눅스 시스템은 책임있는 기술지원이 힘든 공개 소스이기 때문에 버전이나 제품 간 호환이 잘 안 되고, 개발 API의 일관성이 부족하다는 점 등이 문제로 지적된다. 이렇게 어플리케이션 개발을 위한 환경이 상대적으로 열악하여 응용 프로그램의 개발이 활발히 진행되지 못하는 주요한 원인이 된다. 자바 언어 및 자바 가상 기계는 이러한 일관된 API와 실행 환경의 부재를 극복할 수 있는 좋은 대안이 될 수 있다. 자바 언어는 체계적인 언어 및 API에 대한 표준의 유지관리와 플랫폼 독립적이고 객체 지향적이라는 언어적 특성으로 인해 리눅스 환경의 개발 언어로 많이 채용되고 있으며 공통된 실행환경으로써의 가상 기계는 여러 플랫폼에 성공적으로 이식되어 그 활용도가 더욱 높아지고 있다[16-18].

※ 이 논문은 전자통신연구원 인터넷정보가전연구부의 지원을 받았음.
† 정 회 원 : 경기대학교 정보과학부 전임강사
†† 정 회 원 : 전자통신연구원 인터넷정보가전연구부 선임연구원
논문접수 : 2001년 10월 4일, 심사완료 : 2001년 12월 24일

본 연구팀에서는 리눅스 상의 자바 가상기계 환경에서 전자책 리더 시스템을 개발하는데 사용할 수 있는 자바 라이브러리인 (가칭) Xeni를 소개한다. Xeni 라이브러리는 전자책 컨텐츠의 해석 및 랜더링을 위한 기능과 네비게이션, 북마크 등의 기본적인 사용자 인터페이스를 지원하는 클래스들로 구성되며, 사용자 인터페이스 계층을 추가하여 쉽게 리더 시스템을 구축할 수 있다.

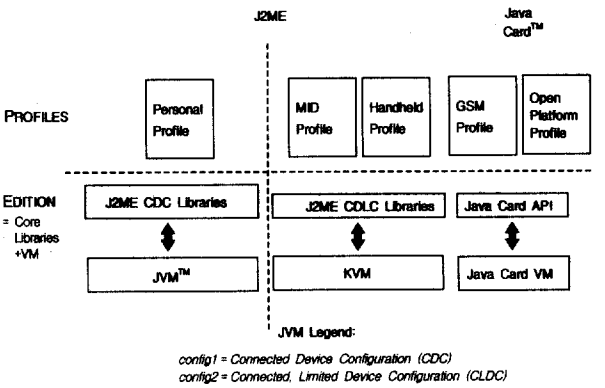
Xeni 라이브러리는 리눅스 시스템에서 동작하는 자바 가상 기계인 Kaffe 환경을 목표로 개발되었다[17]. Kaffe는 내장형 시스템에서 동작하는 자바 가상 기계로 가장 널리 사용되는 오픈 소스 중 하나이다. 본 라이브러리는 자바 1.1.8에 준한 자바 언어의 기본 API만 사용하여 Kaffe 환경뿐 아니라 여타의 내장형 자바 시스템에도 쉽게 이식 가능하다. 본 라이브러리는 미국을 중심으로 제정된 전자책 표준인 Open E-Book Forum에서 제시한 OEB 1.0 표준에 기반한 전자책 컨텐츠의 랜더링 기능을 제공한다. OEB 1.0은 XML 기반의 전자책 컨텐츠 표준으로 내용은 HTML과 유사한 XML 문서에 의해 기술되고 스타일 표기는 CSS(Cascading Style Sheet) 방식으로 주어진다[19, 21].

2절에서는 리눅스 상의 자바 가상 기계들을 소개하고 비교하면서 본 연구에서 목표로 하고 있는 Kaffe 환경에 대하여 소개한다. 3절에서는 전자책 유통 구조와 OEB 1.0 표준에 대해 살펴보고 4절에서는 본 라이브러리의 설계와 구조를 소개한다. 그리고 5절에서는 이 라이브러리를 이용하여 구현된 리더 시스템에 대해 살펴보고 6절에서 결론을 맺는다.

2. 리눅스 용 자바 실행 환경

자바 언어는 클래스 바이트코드 및 API에 관한 표준 명세의 집합이고, 가상 기계란 이 언어의 실행을 가능하게 하는 소프트웨어 모듈이라고 볼 수 있다. 적은 의미의 자바 가상 기계는 자바 클래스 파일을 실행할 수 있는 바이트코드 인터프리터와 기본적인 클래스 및 네이티브 라이브러리를 포함한다. 여기에 클래스 파일 로더와 성능 향상을 위한 JIT(Just In Time) 컴파일러, 메모리 관리를 위한 가비지 콜렉션 기능, 그리고 스래드 관리 기능 등이 추가되어 독자적으로 동작할 수 있는 운영체제의 기능을 모두 포함한 제품들도 다수 등장하였다[16, 17, 18].

한편 자바 2 표준에서는 자바 언어의 실행 환경을 몇 가지로 구분하고 있는데, 크게 데스크탑 환경에서 실행 가능한 표준 본(Standard Edition : J2SE)과 내장형 환경을 위한 마이크로 본(Micro Edition : J2ME)으로 나눌 수 있고, J2ME는 다시 (그림 1)과 같이 일반 JVM 상의 CDC(Connected Device Configuration) 라이브러리를 가지는 유형과 킬로바이트 VM(KVM)이라는 내장형을 위한 특수한 가상 기계에 CLDC(Connected Limited Device Configuration) 라이브러리를 가진 환경을 들 수 있다. 여기서 CDC를 가지는 내장형 가상 기계를 구현한 시스템의 예가 퍼스널 자바 실행 환경이라고 할 수 있다[11, 20]. KVM은 핸드폰 등의 무선 환경이나 초소형 드바이스에 적합하고 퍼스널 자바는 PDA나 정보 가전 등의 내장형 시스템에 적합한 환경이라 할 수 있다. 전자책은 퍼스널 자바 응용 분야에 속한다고 볼 수 있다.



(그림 1) 내장형 자바 가상 기계의 종류

리눅스 상에서의 자바 가상 기계는 몇몇 공개 소스 개발팀에 의해 개발 또는 이식되었는데[16], 우선 블랙다운 그룹에서 선 마이크로시스템 사의 자바 실행 환경(JRE : Java Runtime Environment)을 리눅스에 이식한 Blackdown이라는 가상 기계가 있다[18]. 이것은 현재 JDK 1.3버전까지 이식에 성공하였으며, 선 마이크로시스템 사의 소스 라이센싱을 따르게 된다. 반면 카페는 GNU 소스 라이센싱 방식을 따르는 공개 VM인데 선 마이크로시스템즈 사와 독립적으로 개발되었으며(clean implementation) 카페 그룹에 의해 1996년에 처음 발표된 비교적 오랜 역사를 가지는 시스템이다[17]. 내장형 자바 환경 중 퍼스널 자바 실행 환경을 구현한 것이라고 볼 수 있는 카페는 광범위한 플랫폼에 이식되었으며 가장 널리 알려진 가상 기계 중 하나이고 트랜스버츄얼 사에 의해 상용화를 위한 지원이 이루어지고 있다. 이외에도 Japhar라는 공개 VM이 제한적인 기능으로 발표되었다. 한편 상용화된 VM으로 가장 대표적인 것은 인시그니아 사에서 개발한 제오드(Jeode)를 들 수 있는데, 이 가상 기계는 퍼스널 자바 및 KVM을 구현한 것으로 성능과 안정성 면에서 매우 뛰어난 것으로 알려져 있다. 특히 제오드의 컴파일 방식인 동적인 적응형 컴파일 방식은 JIT (Just In Time) 방식에 비해 메모리 요구량을 획기적으로 줄인 새로운 기술이라고 한다[8]. 국내의 제품으로는 XCE 사에서 개발한 SK-VM이 현재 핸드폰에 탑재되어 출시되고 있다.

〈표 1〉 자바 실행 환경의 비교[8, 16, 17, 18]

	홈 페 이 지	라 이 센 스	특 징	버전 및 크기
Kaffe	kaffe.org	GPL	JDK1.1(AWT 포함) 구현, 안정적, 대부분 플랫폼에 이식 가능	Version 1.0.6, 다운로드 크기 3.4M, JRE 1.1.8
Blackdown	Blackdown.org	Sun Liscenced	JDK 1.2 까지 구현 선 소스의 리눅스 이식	다운로드 크기 9.6M, JRE 1.1.8
Japhar	Japhar.org Hungry.com	LGPL	Netscape등에서 연결하여 사용 가능 (LGPL) 제한적인 기능만 가능	버전 0.1.0, 다운로드 크기 200K
Jeode	Insignia.com	유 료	내장형 환경에서 성능 우수 퍼스널 자바 및 내장형 자바 구현 제품	

본 연구팀에서는 Kaffe를 일차적인 실행 환경으로 보고 Xeni 라이브러리를 개발하였는데 이 VM의 선정 이유는 소스 및 실행 파일의 사용이 자유롭다는 점과 안정성, 그리고 여타 플랫폼으로의 높은 이식성 등을 들 수 있다. 또한 카페 환경은 자바 실행 환경 중에서 전자책이라는 응용분야를 포함하기에 적합한 내장형의 퍼스널 자바 환경에 속한다고 볼 수 있다. 구체적으로 카페는 JDK 1.1.8 버전에 AWT(Abstract Windows Toolkit)을 지원하는데, 이 버전이 현재 많이 사용되고 있는 기본적인 자바 표준 버전일 뿐 아니라 이후 버전과의 호환성도 높아 클래스 라이브러리의 개발을 위해 가장 적합할 것으로 생각된다.

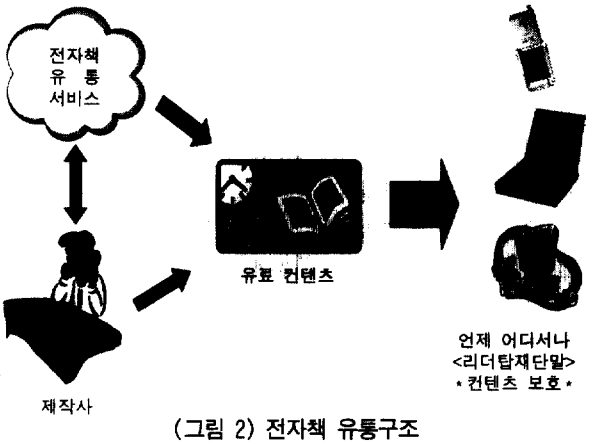
3. 전자책 표준

3.1 전자책 유통 구조

책이라는 인류 유산의 한 형태를 이용하여 인터넷 상에서 컨텐츠 유통 프레임워크를 구축하기 위한 노력이 전자책 표준의 형태로 국내외에서 활발하게 진행되고 있다. 전자책 표준은 저작권 보호가 가능한 컨텐트를 생성, 보급, 이용할 수 있도록 제작자와 서비스, 그리고 사용자 간에 필요한 컨텐트 제작과 유통을 위한 인프라를 제공한다. 전자책 표준에서는 컨텐트 보호를 위해 전용 리더 시스템에 의해서만 컨텐츠 접근을 허용함으로써, 허용된 사용자만이 허용된 컨텐트 부분에 접근할 수 있고 또한 내용을 복제하거나 수정하는 등의 행위를 방지한다.

이와 같이 전자책이 리더 시스템에 의해서만 접근 가능하게 하려면 여러 리더 시스템들이 동일하게 동작하도록 보장되어야 한다. 많은 경우 리더 시스템은 무료로 배포되는 경우가 많은데 한 리더 시스템은 모든 편집자(출판사나 개인 저자가 될 수도 있음)가 만든 책을 의도한 대로 보여줄 수 있어야 한다. 즉 책의 내용 뿐 아니라 배치나 편집 방법 등이 리더에 의해 그대로 보여져야 하는데, 이와 같은 상호 호환성을 보장하기 위해서는 전자책의 내용 작성 표준이 정해지고 리더 시스템에 의해서 사용자에게 보여지는 방식이 정의되어야 한다. 미국을 중심으로 하는 오픈이북 표준[19]과 일본의 JepaX, 그리고 국내에서 최근 발표된 EBKS 표준[2]등이 이러한 목적으로 제정되었다.

(그림 2)는 전자책의 제작과 사용에 이르기까지 관련 주체들의 역할과 관계를 보여준다. 전자책 리더 시스템의 기본적인 기능은 전자책 컨텐츠의 내용을 표준에 준하여 사용자에게 보여주는 기능(이하 "랜더링"이라 함)과 컨텐츠 보호를 위한 인증 기능이다. (그림 2)에서와 같이 편집자에 의해 만들어진 컨텐츠는 서비스 사이트나 다른 유통 채널을 통해서 사용자에게 판매 또는 배포되는데 이 때 암호에 의해 보안 처리되어 권한을 가진 사용자만이 볼 수 있도록 처리된다. 제공된 컨텐츠는 리더 시스템이 설치된 컴퓨터나 이동 단말기에서 읽혀지는데, 리더 시스템은 암호를 해석하여 컨텐츠를 사용자에게 보여준다. 컴퓨터나 단말기에 다운로드된 컨텐츠는 그 자체로서는 해석되어질 수 없으며, 리더 시스템에 의해서 읽어질 때 암호가 해석된다. 그러므로 컨텐츠의 재사용이나 복제가 방지된다.



(그림 2) 전자책 유통구조

2.2 OEB 1.0 : XML 기반 전자책 표준

Open E-Book 포럼은 1998년 설립되어 1999년 처음으로 전자책 표준을 발표하였다. 이 포럼은 미국 내의 주요 출판사와 마이크로소프트를 비롯한 소프트웨어 개발업체, 그리고 PDA 등의 단말기 제조 업체들이 같이 참여하고 있다.

한편 e-Book exchange 워킹 그룹에서는 전자책 유통을 위한 EBX(E-Book Exchange) 표준을 발표하였는데, 여기서는 구매, 대여, 반납 등에 관련된 제반 프로토콜을 정의하고 있다.

OEB 표준에서는 전자책의 내용이 XML 기반의 텍스트 문서와 이미지 파일을 포함할 수 있다고 보는데, 기타 멀티미디어 컨텐츠는 브라우저에서의 플러그인 같은 형태로 리더 시스템의 처리 능력에 맡겨져 있다. XML은 HTML을 뒤이은 인터넷 언어로 태그를 필요에 따라 새로 정의하여 사용할 수 있고, 문서의 형태(스타일 정보)를 별도로 제공함으로써 데이터를 자동으로 처리하는 것이 가능하여 다양한 분야에서 차세대 인터넷 언어로 주목 받고 있다. OEB 표준에서 사용하는 XML은 기존의 HTML과 유사한 구조를 가지며, 태그도 유사하여 기존의 컨텐츠를 변환하는 것이 용이하다.

한편 XML 언어의 또 하나 특징은 내용과 스타일을 분리할 수 있다는 점인데, OEB 표준에서는 HTML 태그를 그대로 사용하여 기본적인 스타일 정보를 표현 가능하도록 하였고 CSS를 이용하여 추가적으로 스타일을 기술할 수 있다. CSS는 태그의 종류나 내용 부류에 따라 적용할 스타일을 기술할 수 있는 방법으로 여기서 사용 가능한 스타일은 다음과 같이 나눌 수 있다.

- 블록(문단)에 관련된 속성 : 여백, 줄 간격, 들여쓰기, 배경색, 테두리, 정렬 등
- 텍스트 부분에 관련된 속성 : 글자 색/크기/모양, 상하위치, 배경색, 테두리 등
- 이미지나 표 등의 배치 방법 : 좌우 위치 지정, 페이지 시작/끝 등
- 페이지에 관련된 정보(확장기능) : 페이지 머리말, 꼬리말, 페이지 나누기, 컬럼 수 지정 등

OEB 표준에서 전자책을 이루는 단위는 하나의 패키지로 묶여진 여러 개의 파일이다. 패키지에는 하나의 책으로 묶여질 수 있는 모든 정보들이 모두 포함되어 있는데, 전체를 대표하는 패키지 파일에는 전자책을 이루는 구성요소(모든 파일 정보)들과 책에 관련된 일반적인 정보(제목, 저자, 출판사 등)를 담을 수 있다. 이 정보는 리더에 의해 책을 읽어 들이는데 사용되며, 그 외에 책을 구성하는 순서나 출판사에서 리더에게 알리는 추가적인 정보를 담을 수 있다. (그림 3)의 소스 부분은 이러한 패키지 파일의 한 예를 보여준다.

```
<?xml version = "1.0"?>
  <package unique-identifier = "xyz">
    <metadata>
      <dc-metadata>
       <dc:title>춘향전</dc:title>
       <dc:type>Novel</dc:type>
       <dc:identifier id = "chunhyangjun"
         scheme = "ISBN">123456789</dc:identifier>
       <dc:publisher>남원시홍보실</dc:publisher>
       <dc:contributor role = "ann">
         남원시 홍보전산실 - 한문해설 담당자
       </dc:contributor>
       <dc:creator>불명</dc:creator>
       <dc:date>2001- 04-23</dc:date>
       <dc:language>kr</dc:language>
      </dc-metadata>
</metadata>
```

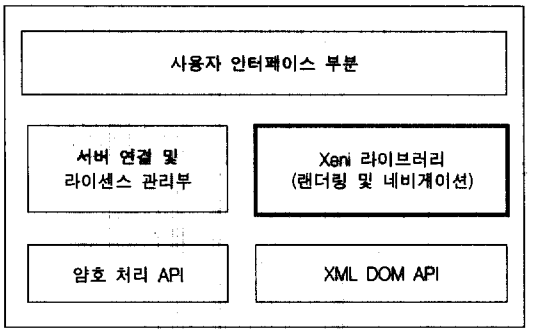
```
    b<menifest>
      <item id = "표지" href = "title.html"
          media-type = "text/x-oeb1-document"/>
      <item id = "광한루원구경" href = "광한루원구경.xml"
          media-type = "text/x-oeb1-document"/>
      <item id = "이도령과의만남"
          href = "이도령과의만남.xml"
          media-type = "text/x-oeb1-document"/>
      … 이하 중략 …
      <item href = "style.css" media-type = "text/css"/>
    </menifest>
    </package>
```

(그림 3) OEB 패키지 파일의 예

4. 전자책 리더 라이브러리 설계 및 개발

Xeni 라이브러리를 이용하는 시스템의 개발 모델은 (그림 4)와 같다. 리더 시스템의 기능은 전자책의 유통을 위한 서버 연결 및 라이센스 관리부와 책을 읽기 위한 랜더링 및 네비게이션 관리부로 나누어 볼 수 있다. 사용자 인터페이스 계층은 제품의 외관인 윈도우와 버튼 등을 관리하고 사용자 입력의 처리를 담당하게 되는데, 시스템의 논리적 계산과 처리를 담당하는 하부 계층과 구분되도록 설계할 수 있다. 하부의 처리부는 암호 처리부와 XML DOM API 부를 포함하는 하부 라이브러리와 이를 이용하는 중간 계층으로 나눌 수 있다. 중간 계층을 독립된 API 라이브러리 형태로 제공하게 되면 사용자 인터페이스의 수정이 용이하며, 개발된 라이브러리의 재사용으로 리더 시스템을 구축하는데 드는 시간과 노력이 절감되는 등 여러 가지 장점이 있다. 여기서 소개하는 Xeni 라이브러리는 그 중에서 랜더링 및 네비게이션을 위한 기능은 지원하는 부분이다.

Xeni 라이브러리를 전자책 리더 시스템의 구축을 위해 재사용 가능한 API로 개발하기 위해서는 전자책 리더의 랜더링 및 네비게이션 관련 기능이 명확히 정의되어야 하고 사용자 인터페이스 계층에서 사용할 수 있는 API 인터페이스를 효과적으로 설계하는 것이 핵심적인 부분이 될 것이다.



(그림 4) 라이브러리 활용 시스템 구성도

4.1 기능 설계

본 연구팀에서는 전자책 리더의 기능을 랜더링 부분과 네비게이션 부분, 그리고 추가적인 인터페이스 부분으로 나누었다.

본 연구에서 개발한 라이브러리는 위 3절에서 소개한 OEB 표준을 따르는 컨텐츠를 랜더링하는 기능을 제공한다. 그러므로 파일의 형식 및 랜더링에 관한 기능 명세는 OEB 표준을 따르게 된다. 특기할 점은 전자책 리더는 전용 단말을 사용하고 스크린의 크기가 작은 경우를 가정하므로 윈도우 크기가 바뀌는 것을 고려하지 않고 스크롤 기능도 제공하지 않는 것이 일반적이다. 그러므로 종이책을 읽을 때와 같이 항상 같은 페이지에 같은 내용이 배치되는 것을 보장할 수 있다.

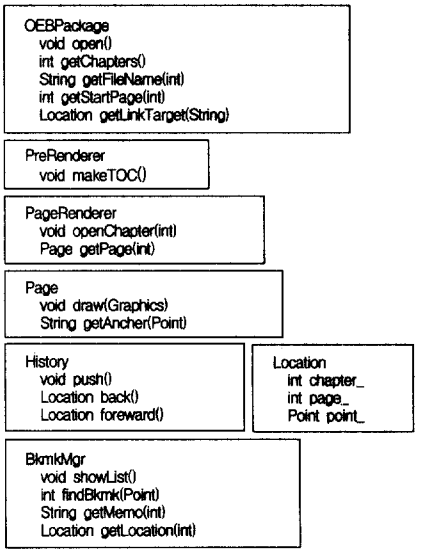
한편 네비게이션 부분은 다음페이지, 이전페이지, 첫페이지, 끝페이지로의 이동이 필요하고 임의의 페이지로 바로 이동하는 것도 가능하다. 또한 원하는 챕터로 바로 이동하기와 링크에 의해 지정된 곳으로 이동하기 등이 가능하여야 할 것이다. 이동을 위한 사용자 상호 작용은 사용자 인터페이스 층에서 구현되므로 라이브러리에서는 지정된 위치로의 이동 기능, 즉 지정된 챕터를 읽어들여 지정된 페이지를 보여줄 수 있는 기능을 제공해야 할 것이다.

추가적인 전자책 리더의 기능으로 Xeni 라이브러리에서는 히스토리에 의한 네비게이션 기능과 북마크에 의한 위치 표시 및 메모 기능, 그리고 특정 북마크로의 이동 기능을 제공하고 있다.

본 라이브러리의 특징으로는 리더 시스템이 전자책의 내용을 읽어들이는 방식을 두가지 모드로 구분하였다는 점이다. 전자책은 여러 개의 파일(챕터)로 구성되고 전자책에서 보통 사용되고 있는 방식은 사용자가 요청한 챕터에 해당하는 파일을 읽어들이는 것인데[13, 15], 그때그때 한 챕터씩 읽어들이게 되면 책 전체의 정보를 알 수 없으므로 다른 챕터로의 링크를 처리할 수 없다는 점과 전체 책의 페이지 정보를 이용할 수 없다는 제약 사항을 가지게 된다. 챕터 단위로 랜더링을 하는 경우에는 책 전체에서의 페이지 번호 대신 [챕터, 챕터 내 페이지]의 형태를 가지는 위치 정보를 기반으로 네비게이션하게 된다. 그러나 책 전체에 대해 일련번호로 매겨지는 페이지 번호는 오프라인에서 책을 읽을 때 가장 기본적인 네비게이션 정보이며, 책 전체에 대한 일관된 위치 및 방향을 제공하는 역할을 한다. 본 라이브러리에서는 책 전체를 미리 다 읽어 랜더링하는 기능을 선택적으로 제공한다. 이 전처리 과정은 상당히 많은 계산양이 요구되지만, 사용자는 페이지 번호 기반 네비게이션을 사용할 수 있게 된다.

4.2 API 인터페이스 설계

위의 기능을 제공하기 위해서 아래와 같이 몇 가지 중요한 클래스의 인터페이스를 설계하였다. (그림 5)는 이들 클래스의 인터페이스를 요약하여 보여준다.



(그림 5) 클래스 인터페이스

① 패키지 클래스(OEBPackage)

패키지 파일을 읽어들이고 책 정보와 챕터에 대한 정보를 유지 관리하는 기능을 가진다. 이외에도 CSS 스타일을 비롯한 전자책의 정적인 정보들을 보관하는 정보 저장 및 관리 클래스이다.

② 렌더러 클래스(PageRenderer)

랜더링 엔진 역할을 하는 클래스로 요청된 챕터의 파일을 읽어 페이지를 계산한다. 생성된 페이지를 보관하다가 네비게이션에 의해 이동할 페이지가 요청되면 제공한다.

③ 전처리 랜더링 클래스(PreReader)

전체 책을 구성하는 모든 파일을 읽어 랜더링 계산을 수행하는 부분이다. 여기서는 페이지 수의 계산과 링크 타겟의 계산을 수행한다. 페이지 수는 OEBPackage 파일의 각 챕터에 대한 정보로 저장되고 링크 타겟은 Location으로 역

시 OEBPackage에 저장된다.

④ 페이지 클래스(Page)

화면에 그려지는 하나의 단위가 되는 페이지의 내용을 담고 있는 클래스로 줄 정보, 페이지의 레이아웃 등의 정보를 모두 가진다. 페이지 클래스에서 해당 페이지를 그리기 위한 메소드를 제공한다.

페이지 클래스는 이 외에 그 페이지에서 링크가 설정되어 있는 위치를 알고 있어서 주어진 마우스의 좌표에 의해 링크의 목적위치를 돌려준다. 링크는 랜더링하기 위해 파일을 읽어들일 때 <a href=""> 태그에 의해 표시된 자리를 페이지 클래스에서 기억하게 된다.

⑤ 위치 정보 클래스(Location)

위치 정보를 가지는 클래스로 챕터, 페이지, 그리고 페이지 내에서의 좌표 값을 가진다. 위치 정보는 리더의 네비게이션에서 표시 가능한 모든 위치를 나타내는데 사용되며, 링크에 의해 이동할 위치, 히스토리에서 이동한 위치 표시, 그리고 북마크의 위치 표시 등에 쓰인다.

⑥ 북마크 관리자 클래스(BkmkMgr)

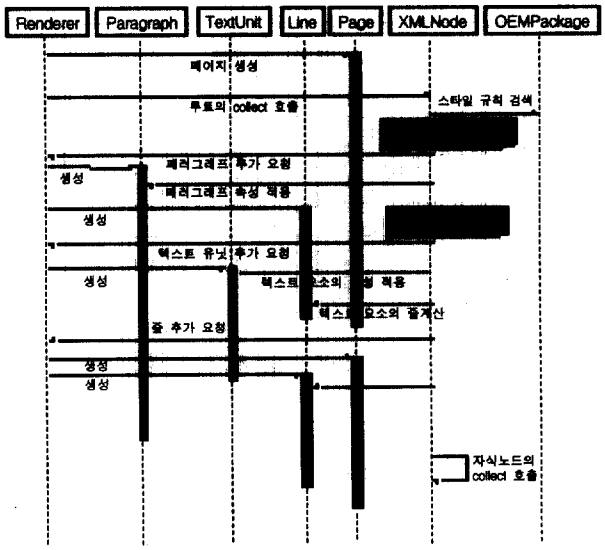
북마크는 사용자가 책 속의 특정 지점을 표시하고 메모를 달 수 있는 기능으로 각 북마크는 Location 클래스를 상속한 북마크 클래스에 의해 나타내진다. 북마크 기능에서는 메모 편집 기능과 리스트 관리 기능을 제공해야 하는데 이를 위한 별도 윈도우를 가진다. 북마크의 위치는 페이지 별로 관리되며, Page 클래스에서 페이지를 화면에 그릴 때 북마크를 표시해 준다. 또 마우스의 위치에서 해당 북마크가 있으면 메모의 내용을 돌려주어 툴팁으로 표시 가능하게 제공한다. 북마크 관련 기능을 이 클래스에 모두 모아서 북마크 기능을 포함하지 않을 경우 이 클래스를 제외시킬 수 있게 처리하였다.

⑦ 히스토리 클래스(History)

이 클래스는 사용자가 히스토리를 기반으로 네비게이션할 수 있게 해주는 back(), foreward(), push() 메소드를 가진다. 이동 위치는 Location 클래스를 사용하여 저장한다.

4.3 Xeni 라이브러리의 구현

Xeni 라이브러리에서 랜더링 기능은 PageRenderer클래스에 의해서 수행된다. 이 클래스는 DOM API의 Element에서 상속한 XMLNode 클래스의 랜더링 함수 collect를 호출한다. 이 함수는 DOM 트리의 루트 노드로부터 재귀적으로 자식들을 호출하여 패러그래프, 텍스트 부분, 이미지, 표 등의 모든 요소들을 수집한다. XMLNode 클래스가 요소들을 수집하는 과정은 (그림 6)과 같다.



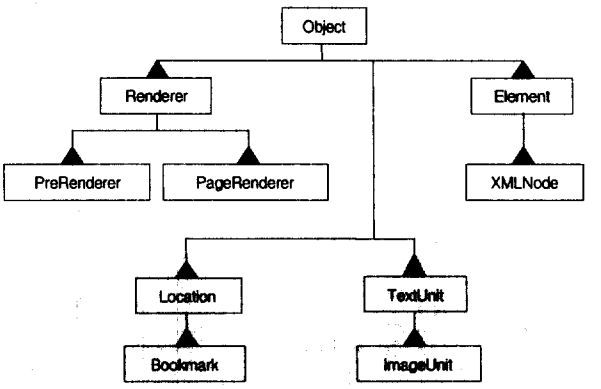
(그림 6) XMLNode 클래스의 랜더링 요소 수집 과정

랜더러 객체는 페이지와 줄 객체를 생성한 후 DOM 트리의 루트노드인XMLNode객체의 collect함수를 호출한다. 생성된 줄과 페이지는 랜더링 과정에서 최근에 계산된 줄과 페이지를 저장하는 객체가 된다. XMLNode는 태그의 종류에 따라 적용할 스타일을 OEBPackage에서 찾아오고, 블록 요소이면 패러그래프 객체를 생성하고 PCDATA 노드이면 텍스트 유닛 객체를 생성한다. 이들 객체에 스타일 규칙에 따라 속성을 반영하고 현재 줄 객체에게 텍스트 유닛을 추가하도록 요청한다. 줄 객체는 현재까지 보태진 유닛들 다음 위치에 새로운 유닛을 추가하고 줄이 넘어가는 경우는 거짓을 반환한다. 줄이 종료하기 않은 경우는 랜더러에게 새로운 줄의 생성을 요청하고 새 줄에게 텍스트 유닛의 나머지 부분을 추가하여 계산하도록 요청한다. 이와 같이 렌더러는 패러그래프, 텍스트 유닛, 줄, 페이지 등의 정보를 생성, 관리하는 역할을 하고 XMLNode는 자신이 가진 노드의 종류에 따라 속성을 적용하고 줄 객체에게 랜더링 계산을 요청한다. XMLNode는 이상의 과정이 끝나면 자식들의 collect 함수를 차례대로 호출하여 재귀적으로 부분트리의 요소들을 모두 수집한다.

랜더링은 PreRenderer와 PageRenderer의 두가지 클래스에서 지원되고 이들 클래스의 부모로 Renderer 클래스가 있다. PreRenderer는 화면에 보여주기 위한 것이 아니라 페이지 계산과 링크 정보의 수집을 하는 부분이고 PageRenderer는 화면에 보여줄 페이지의 계산을 수행한다. 위의 (그림 6)은 페이지 랜더러 부분의 수행과정을 보여주는데, PreRenderer는 PageRenderer와 달리 패러그래프와 텍스트 유닛, 줄, 페이지 요소를 생성하지 않고 한 개의 인스턴스를 이용하여 정보를 수정기록하여 메모리의 낭비를 줄였다. 또 PreRenderer는 링크 타겟이 되는 모든 지점을 계산하여 수집해 둔다. 반면 PageRenderer는 링크가 걸려 있는 지점

이 어디인가를 페이지마다 기억하여 두고 사용자가 마우스를 이동하다가 그 지점에 오면 링크임을 표시해 줄 수 있어야 한다.

한편 XMLNode는 DOM API의 Element 클래스를 상속하여 파싱할 때 트리의 노드를 XMLNode의 객체로 생성하도록 처리하였다. 그리하여 랜더링의 계산이 트리 상에서 재귀적으로 호출된다. 이외에 클래스들 간의 주요한 상속관계를 (그림 7)에서 요약하여 보여준다.



(그림 7) 랜더링 부분의 클래스 관계도

현재 Xeni 라이브러리는 자바 MSXML에 기반하여 개발되었다. 그러나 DOM API에 대한 의존도는 그리 높지 않아 다른 API로의 대체가 어렵지 않을 것으로 예상된다[12].

4.4 내장형 가상 기계 환경의 고려사항

Xeni 라이브러리는 내장형 리눅스 시스템 상에서 동작하는 자바 가상 기계 환경을 주요 목표로 개발되었다. 대표적인 가상 기계 환경으로 Kaffe와 Blackdown 자바 실행 환경에서 개발 및 테스트하였다.

내장형 시스템의 특징은 CPU의 성능과 메모리 크기 등의 사양에서 제약을 가질 수 있다는 점이다. 특히 메모리는 데스크탑 환경과 비교했을 때 가장 큰 차이이자 제약 사항이라고 할 수 있다. 이러한 내장형 시스템의 환경을 고려하여 Xeni 라이브러리의 설계에서는 XML의 파서를 DOM API를 사용하되 이후에 SAX로 변환을 고려하여 설계하였다. 아래 표는 DOM과 SAX API를 비교한 것이다. 또한 확장성(scalability)를 높이기 위해 전자책을 읽어들이는 전처리 과정에서 전체 책을 읽어들일 것인가 현재 챕터만 읽어들일 것인가를 실행시간 아규먼트로 지정할 수 있게 하였으며, 북마크 기능을 독립적으로 설계하여 구현 시스템에서 선택적으로 추가할 수 있게 설계하였다. 4.1절에서 언급한 바와 같이 페이지 기반의 네비게이션 지원을 위해서는 전체 책을 읽어들이는 전처리 과정을 필요로 하지만 이것은 제한된 메모리와 계산 성능을 가진

환경에서는 너무 많은 시간을 요하므로 이 과정을 지원하되 사용자 인터페이스를 개발하는 과정에서 생략 가능하도록 설계하였다. 본 연구팀에서 개발한 리더 시스템은 전처리 여부를 실행 시간 아규먼트로 선택할 수 있게 하였다.

〈표 2〉 XML 파싱 API 비교

	SAX	DOM
파싱 처리 방법	이벤트 처리 방식	파스 트리 생성 / 반환
메모리 사용	적 음	많음(파스 트리 전체를 구축)
유효성검증	불 가	가 능
API 크기	적 다	크 다
처리속도	느리다	빠르다

구현 단계에서의 고려사항으로는 텍스트, 문단, 줄 정보, 페이지 정보 등 랜더링 객체들을 가능한 한 새로 생성하지 않고 하나의 메모리 인스턴스를 사용하였으며, Font 객체를 자주 생성하는 것에 의해 메모리에 부담을 주지 않기 위해 Font객체를 생성 관리하는 팩토리 클래스를 추가하였다.

또한 임베디드 환경의 자바 가상 기계의 공통적인 지원 범위인 1.1.8 버전에 준하여 모든 라이브러리 클래스들을 구현하였고 GUI를 위한 API에서도 자바 언어의 기본 라이브러리인 AWT(Abstract Window Toolkit) 만을 사용하고, 자바 1.2 이후의 기능과 스윙 등의 추가 API는 사용하지 않았다. 이러한 기본적인(pure) 자바 언어로 개발하여, 퍼스널 자바 환경에 준하는 카페 실행 환경에서 동작할 뿐 아니라 다양한 플랫폼의 가상 기계들에 활용될 수 있는 가능성이 높아질 것으로 기대된다. 그 외에도 정보 가전 등 내장형 리눅스의 응용분야를 고려하여 마우스와 함께 키보드를 이용한 인터페이스가 가능하도록 설계하였다.

5. 라이브러리를 이용한 전자책 리더의 구현

본 연구팀에서는 Xeni 라이브러리를 이용하여 전자책 리더의 랜더링 부분을 구현하였다. 이 시스템은 카페 상에서 동작하며 현재 프로토타입이 개발되었다.

5.1 사용자 인터페이스 부분의 구성

전체 시스템은 위에서 소개된 Xeni 라이브러리에서 제공되는 기능을 이용하며, 프레임을 담당하는 메인 클래스 Reader와 버튼 관리 패널, 그리고 내용을 그려주는 ViewPane, 목차를 그리는 TocPane으로 이루어진다. 이하에서는 시스템의 동작을 단계별로 살펴본다.

5.1.1 시작 단계

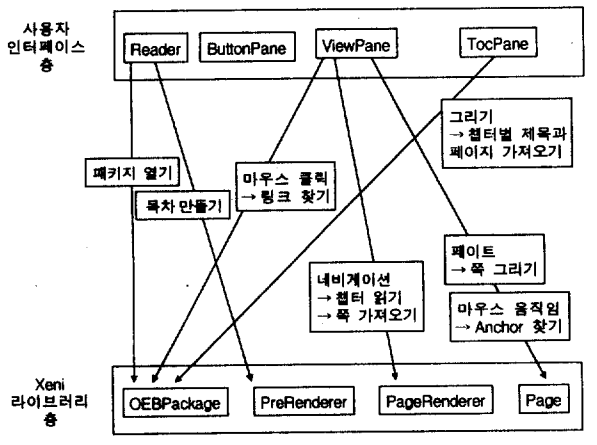
프레임을 생성하고 버튼과 패널을 생성한다. 파일 대화상자에서 읽을 전자책 파일을 선택한다.

5.1.2 전처리 단계

패키지 파일을 읽어들이고(Package.open()), 전자책을 구성하는 모든 챕터를 읽어들인다(PreRenderer. makeTOC()).

5.1.3 페이지 생성 단계

ViewPane에서 해당 챕터를 읽는다(PageRenderer. getChapter(0)). 읽어들인 챕터에서 해당 페이지를 현재 페이지로 설정한다. 그러면 ViewPane의 paint 함수에 서 현재 페이지의 draw 함수를 호출한다(Page.draw()).



(그림 8) 사용자 인터페이스 층과 Xeni 라이브러리의 인터페이스

5.1.4 이 동

이동은 같은 챕터 안에서 다른 페이지로 이동하기와 다른 챕터로 이동하기로 나누어진다. 우선 같은 챕터 안에서의 이동은 ViewPane의 현재 페이지를 이동할 페이지로 설정해 주면 된다. 다른 챕터로 이동하는 경우는 위의 페이지 생성 단계와 같이 해당 챕터를 읽어들이고 읽어들인 챕터에서 이동할 페이지를 현재 페이지로 설정한다.

5.1.5 목차 생성

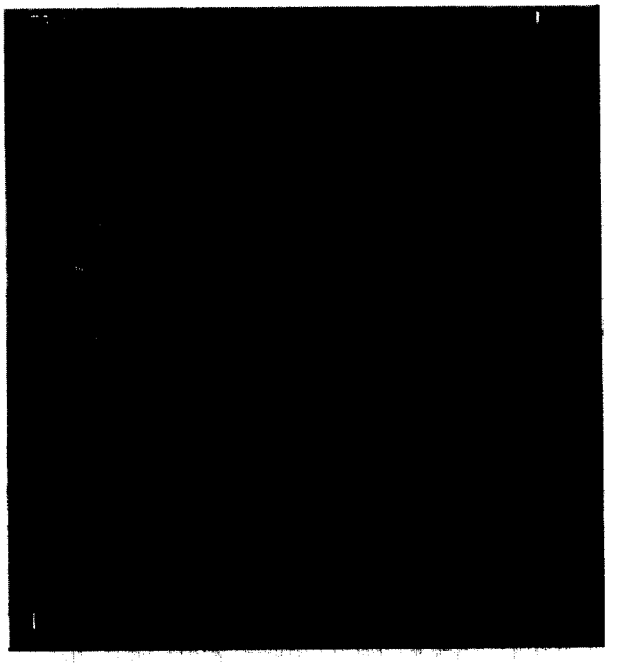
TocPane은 목차를 보여주는 부분으로 아래 (그림 9)와 같다. 여기서는 챕터의 순서와 제목, 그리고 각 챕터가 시작하는 페이지 수를 보여준다. 이상의 정보는 OEBPackage에서 전처리 단계에서 모아진 내용을 제공하는 인터페이스를 가진다.

5.1.6 링 크

링크는 링크가 걸려있는 곳(앵커)과 이동할 곳(타겟)으로 구분되는데, 앵커 정보를 관리하는 것은 Page 클래스이고 타겟 정보는Package 클래스에서 관리한다. 링크에 의한 이동은 ViewPane의 마우스 이벤트 처리부에서 담당하게 된다. 본 시스템은 마우스의 움직임에 따라 링크의 내용을 툴팁으로 보여주고 마우스가 클릭되면 해당 위치로 이동한다. 툴팁을 보여주기 위해서 ViewPane에서 마우스가 움직이는 위치를 가지고 Page 클래스에게 getAncher()를 요청한다. 반환된 스트링으로 툴팁을 보여주고, 마우스가 그 위치에서 클릭되면 패키지에서 이동할 타겟 위치를 찾는다(Package. getLinkTarget()). 이 함수의 반환값은 위치 정보를 가진 Location 객체이다. 여기에서 챕터와 페이지, 그리고 페이지 안에서의 좌표를 구할 수 있다. 해당 챕터와 페이지로 이동하는 것은 위의 (4)의 이동 단계에서와 같다.

5.1.7 북마크

사용자는 페이지에서 북마크가 걸린 위치에 마우스를 가져다 놓고 메모 내용을 툴팁으로 볼 수도 있고 그 위치를 클릭하여 메모 내용을 바꿀 수 있다. 또 새로운 곳에서 팝업 메뉴를 이용하여 북마크를 추가할 수 있다. 그 외에 북마크 리스트에서 해당 북마크로 바로가기, 삭제하기, 파일에 저장/읽어오기도 가능하다.



(그림 9) 목차 화면

5.2 구현 결과 화면

이 절에서는 구현된 시스템의 수행 화면을 통해 기능을 소개한다. 여기서 사용하는 전자책 컨텐츠는 남원시 전산실에서 작성한 춘향전 HTML 파일을 본 연구팀에서 수정하여 작성한 전자책 컨텐츠이다. 수행 화면은 윈도우에서 스크린 캡춰한 것이고, 리눅스 상에서는 Kaffe 1.0.6 버전과 Blackdown JDK 1.1.8 버전에서 테스트하였다.[1)]

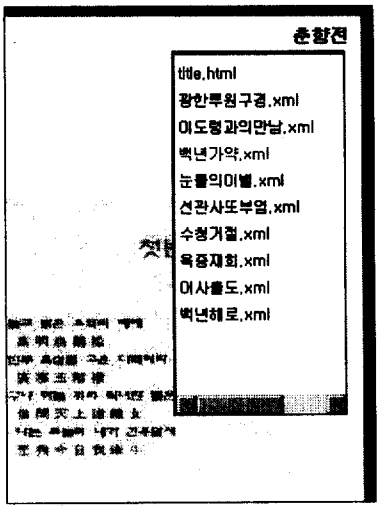
(그림 9)는 전자책을 읽어 들였을 때 나타나는 초기 화면이다. 이 페이지에서는 구성하는 챕터들과 해당 페이지를 보여준다. 전자책 전체를 읽어들이는 전처리 과정은

1) 실행 환경 : 펜티엄 233, 메모리 128, 리눅스 커널 2.4.2. Gnome 데스트 탑 환경, KAFFE OpenVM 1.0.5 for i386, Blackdown jre1.1.8, ver. 3.0

비교적 많은 시간이 걸리는데, 펜티엄 233MHz 환경의 리눅스 카페에서 160페이지의 춘향전 파일을 모두 읽어들이는데 약 20초 정도가 소요된다. 이 시간 동안 목차 페이지에서 현재 몇 페이지까지 읽었는가를 표시해 준다. 또한 읽기가 완료되면 버튼이 활성화된다. 이 화면은 또한 네비게이션 기능을 제공하는데, 목차에서 제목 부분에 마우스를 가져가면 제목이 노란색으로 바뀌고 그 상태에서 마우스를 클릭하면 해당 챕터의 첫번째 페이지로 이동한다.

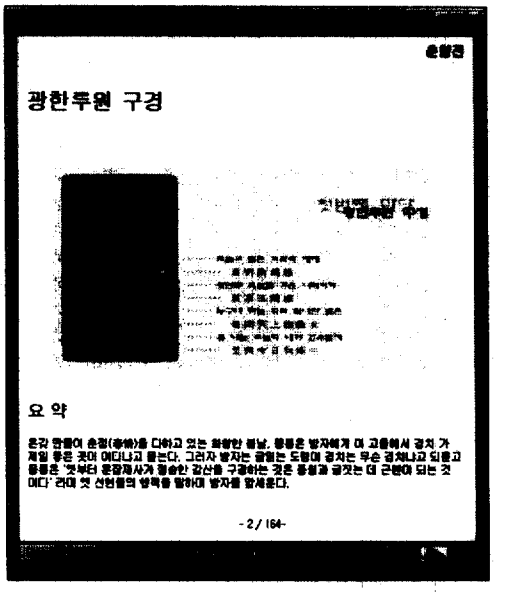
현재 출시되어 있는 전자책 리더들은 목차생성 기능을 가지지 않는 경우도 있다. 실시간 내장형 환경에서 동작하는 전자책 리더는 전체 파일을 모두 읽어 들여 랜더링하는데 너무 많은 시간이 걸리고 메모리의 제약으로 그 자체가 불가능한 경우도 있다. 본 시스템은 이러한 내장형 시스템의 환경을 고려하여 전처리 단계를 생략 가능하도록 구현하였다. 가상 기계의 실행 명령에서 프로그램 아규먼트를 받아 목차 생성 여부를 사용자가 실행시 결정할 수 있게 구현하였으며 목차를 생성하지 않는 경우는 목차 페이지에 챕터 별 페이지 수가 보여지지 않고 제목만 나타난다. 그리고 책 전체의 링크 타겟을 미리 모아두는 기능, 지정된 페이지로 바로 가기 기능은 지원되지 않는다.

(그림 10)은 본문 내용을 보는 화면이다. 본문의 그림이나 제목 등은 스타일 파일에서 정의된 스타일 규칙을 따라 랜더링 된 것이며, 하단에는 전체 페이지 중에 현재 페이지 번호를 보여준다. 상단 우측에는 책의 제목이 보여진다.



(그림 10) 본문의 목차창 띄우기

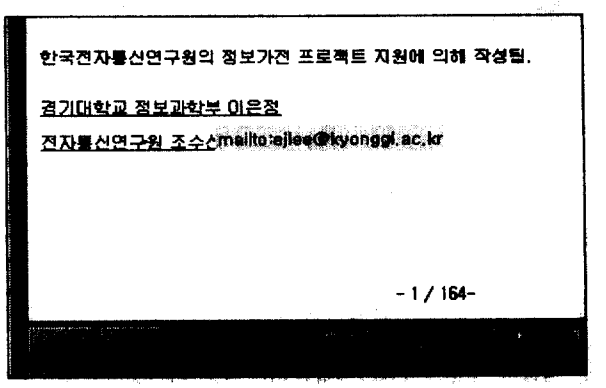
(그림 11)은 본문 내용에서 작은 윈도우로 목차를 볼 수 있는 기능이다. 여기서는 챕터 별 페이지 번호는 보여주지 않고 마우스 선택에 의한 이동은 가능하다.



(그림 11) 본문 보기 화면

(그림 12)는 리더 시스템 하단의 툴바를 자세하게 보여준다. 툴바에서 제공되는 기능은 열기, 지정된 페이지로 바로 가기, 목차 내용을 번갈아 보기, 현재 챕터의 제일 앞, 제일 뒤 페이지로 가기, 다음 페이지, 이전 페이지로 가기 기능이 제공되며, 화살표는 히스토리의 back, foreward 버튼이다. 마지막 버튼은 북마크를 표시할 것인가 숨길 것인가를 정하는 버튼이다. 숨기기를 선택한 경우 북마크 표시 부분을 보여주지 않고 북마크 추가나 리스트 보기 메뉴도 제공되지 않는다.

(그림 12) 네비게이션 툴바



(그림 13) 링크 연결부의 표시 및 툴팁 기능

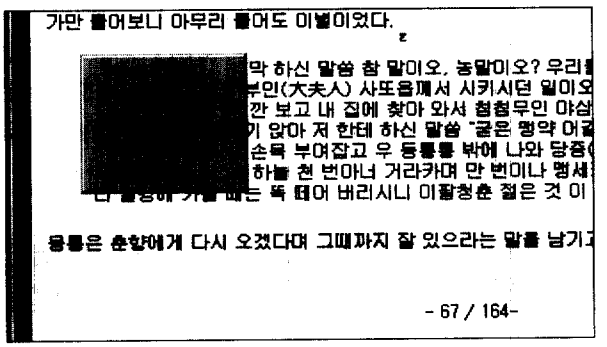
(그림 13)은 링크 부분에 마우스를 가져갔을 때 툴팁으로 이동할 타겟을 보여주는 화면이다. 여기서는 <a> 태그에

설정되어 있는 링크의 내용이 mailto : 인 경우이다. 본 시스템은 현재 전자책 내부에서의 링크 이동만 지원하고 있다. 링크 부분에 설정된 내용이 현재의 전자책이면 마우스를 클릭했을 때 해당하는 위치로 이동한다.

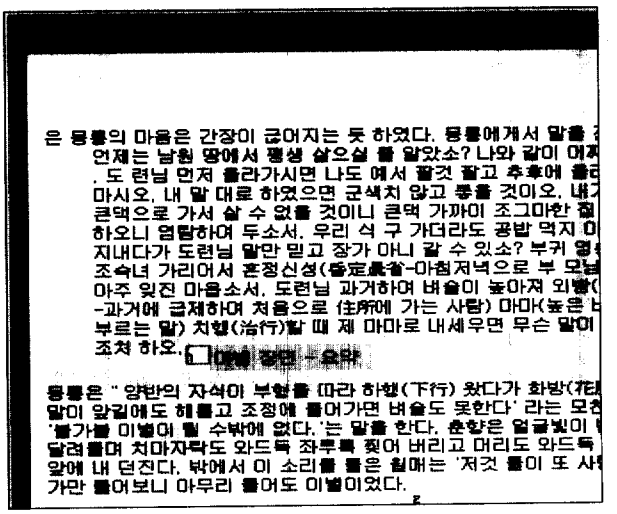
책을 읽으면서 이동하는 경우에는 이동할 위치가 히스토리에 기록되어, back, foreward로 히스토리를 따라 네비게이션하는 기능이 제공된다.

(그림 14)에서 (그림 16)까지는 북마크에 관련된 기능을 보여주는 화면이다. (그림 14)는 북마크를 추가하는 메뉴가 제공되는 팝업 메뉴이다. 팝업 메뉴에서 북마크 기능을 선택하면 마우스가 있는 위치에 새로 북마크가 추가된다. 북마크 기능을 선택하면 아래 (그림 17)과 같은 편집창이 열리고 여기에 메모를 입력할 수 있다. 메모가 입력된 위치는 (그림 15)와 같이 뷰어에서 빨간 색 숫자로 표시되어 북마크를 보여준다. 북마크 위치에 마우스를 갖다대면 툴팁으로 메모의 내용을 보여준다.

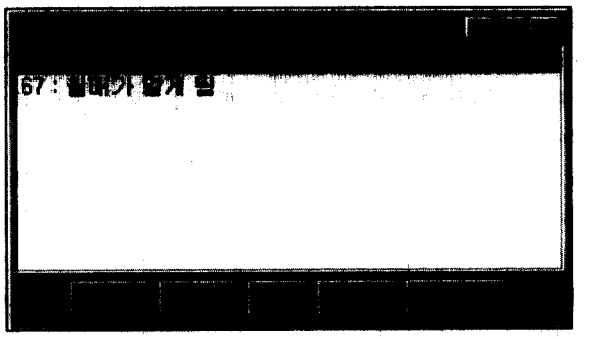
또한 현재 위치의 북마크는 테두리를 해서 보여준다. 현재 위치의 북마크란 북마크를 이용한 네비게이션에서 다음 또는 이전 북마크로 이동할 때 현재 위치가 된다. 북마크가 설정된 위치에서 마우스를 클릭하면 그것이 현재 위치가 되고 메모 편집창이 뜬다.



(그림 14) 팝업 메뉴의 북마크 기능

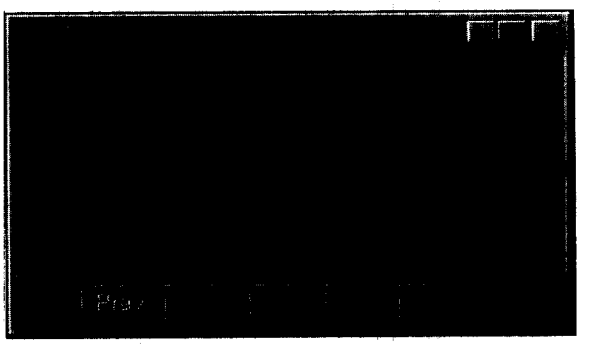


(그림 15) 북마크의 메모 보기 기능(툴팁)



(그림 16) 북마크 리스트 보기

(그림 17)은 북마크의 내용을 편집하는 편집창인데, 이 창은 편집과 북마크 리스트 보기((그림 16) 참조)를 겸하는 별도의 프레임 윈도우로 나타난다. 편집 상태에서는 추가할 북마크나 현재의 북마크의 메모를 편집할 수 있고 리스트 상태에서는 전체 리스트를 보여주고, 다른 북마크로 이동하는 기능이나, 저장/읽어오기 등의 기능을 제공한다.



(그림 17) 북마크 메모 편집 화면

6. 결 론

본 논문에서는 내장형 리눅스 환경에서 전자책 리더의 랜더링 기능을 제공하는 자바 클래스 라이브러리 Xeni를 소개하였다. 본 라이브러리는 국제적인 전자책 표준인 OEB 1.0 표준에 따라 작성된 전자책의 컨텐츠를 읽어 유효한 XML 문서를 해석하고 CSS 표준에 따른 스타일 정보를 처리하여 랜더링하는 기능을 제공한다. 또한 기본적인 네비게이션 기능과 북마크를 이용한 메모 기능을 제공한다.

리눅스 시스템은 저가형 전자책 리더 단말기를 제작하기 위해서 사용할 수 있는 공개 소스 기반 내장형 운영체제이며, 리눅스 상에서 동작하는 자바 가상 기계는 일관된 실행 및 개발 환경을 제공할 수 있어 전자책 리더 시스템을 위한 좋은 플랫폼을 제공한다. 특히 내장형 자바 실행 환경을 구현한 카페 시스템은 PDA나 정보 가전 등의 제품을 위한 퍼스널 자바 환경을 구현한 공개 소스 기반의 실행환경으로 본 연구에서는 내장형 리눅스 시스템에서의 전자책 리더를 위한 일차적인 개발환경으로 카페 시스템을 선택하였다.

Xeni 라이브러리는 리눅스 시스템 상의 카페 환경에서 사용될 수 있는 API로써, JDK 1.1.8에 준하는 순수한 자바 언어와 기본 라이브러리인 AWT만을 사용하였으며, 기능을 부가적으로 추가할 수 있는 조립형으로 설계하여 작고 확장가능한(scalable) 클래스 라이브러리다.

본 논문에서는 개발된 라이브러리를 이용하여 사용자 인터페이스 계층을 구현한 참조 시스템을 개발하였으며, 개발된 시스템을 카페, 블랙다운, 선 마이크로시스템즈 사의 자바 실행환경 등에서 테스트하였다.

6.1 이후 연구 방향

Xeni 라이브러리는 내장형 자바 가상 기계 환경을 위한 기본 클래스 라이브러리로 개발되었으며 현재 프로토타입을 개발 완료한 상태이다. 이 라이브러리를 이용한 리더 시스템은 현재 상당량의 실행메모리(3메가바이트 정도)를 필요로 하는데, 이를 1/3 이하로 줄이기 위해서는 랜더링 엔진의 상당한 개선이 필요할 것이다. 구체적으로 XML 파서 API를 SAX로 바꾸고 현재의 파일 단위 랜더링을 줄 단위의 완전히 동적인 랜더링으로 바꿔야 할 것이다. 이외에도 개발 API의 활용도를 테스트하여 수정 보완하고, 추가적으로 필요한 기능의 보강과 조립형 API의 특성을 살린 추가적인 모듈화 등이 이후의 과제가 될 것이다.

한편 Xeni 라이브러리는 현재 랜더링 기능만을 제공할 뿐 아니라OEB 표준에 따른 전자책 컨텐츠만 처리 가능하다. 이를 다른 표준이나 스타일 방식을 지원하도록 확장할 수 있을 것이다. 특히XSL-FO 기반의 EBKS 표준에 대해서도 지원하는 것을 고려 중이다[2]. XSL-FO는 CSS에 비해 매우 강력한 포매팅 기능을 가져 현재 출판물에서 사용하는 모든 배치와 편집 방식을 그대로 지원 가능하다. 그러나 XSL-FO는 매우 방대한 표준으로 모든 기능을 구현하는 것은 많은 개발 노력이 필요할 것으로 보인다[21].

또한 전자책 리더 시스템 구축을 위해 중요한 기능으로 서버와의 통신을 비롯한 유통기능과 컨텐츠 보호를 위한 암호화 및 복호화 기능이 있다((그림 3)에서 서버 연결 및 라이센스 관리부). 본 연구팀은 Xeni 라이브러리를 확장하여 교환 프로토콜인 EBX(E-Book eXchange) 프로토콜을 지원하는 기능을 포함시킬 계획이다[19].

다음으로 컨텐츠 제작을 위한 소프트웨어의 개발이 필요할 것이다. 리더에서의 랜더링 결과를 보면서 WYSIWYG으로 편집할 수 있는 편집기의 제공은 컨텐츠의 활발한 제작을 위해서는 반드시 필요하다. 현재 전자책 제작에 사용가능한 HTML 기반의 편집기 제품이 많이 출시되어 있으며, XSL 기반의 편집기들도 다수 상용화되어 있다[5, 14]. 이러한 기술들을 이용하면 전자책 표준을 위한 편집기의 개발도 가능할 것으로 보인다.

참 고 문 헌

[1] 윤지헌 외, "리눅스에서의 한글 인터페이싱", *정보처리학회지*, Vol.6, No.6, pp.71-77, 1999.

[2] 손원성 외, "XML 에 기반한 한국 전자책 문서 표준", *정보처리학회지*, Vol.8, No.3, pp.27-37, 2001.

[3] 이은정, 조수선, "Q+ 플랫폼을 위한 전자책 리더 시스템 개발", 2001년 한국전자거래학회 학술대회, 서울, 2001.

[4] 임영태, 한우용, 정회경, "XML에 기반한 EDI 문서 교환 시스템 설계 및 구현", *정보처리학회논문지*, Vol.7, No.11, 2000.

[5] 정채영 외, "XML 에디터", *정보처리학회지*, Vol.8, No.3, pp.10-16, 2001.

[6] 조기원, 최성, "e-book(전자책)", *정보처리학회지*, Vol.7, No.5, pp.96-102, 2000.

[7] 하순회, 박근수, "전자책 단말기 기술의 현황과 전망", *정보과학회지*, Vol.18, No.9, pp.4-12, 2000.

[8] Insignia Solutions, "Jeode Platform White Paper," http : //www.insignia.com, April, 2001.

[9] Peter C. Mehlitz, "Kaffe - one for all.. or .to have a free Java," http : //www.transvirtual.com/presentations/one4all/ , 1998.

[10] Steven Holzner, *XML Inside*, 디지털북스, 2001.

[11] Sun Microsystems, "Personal Java Technology White Paper," http : //java.sun.com/j2me, 1998.

[12] 마이크로소프트 xml 홈, http : //msdn.microsoft.com/xml/.

[13] 마이크로소프트 리더 홈, http : //www.microsoft.com/reader.

[14] OCI 정보통신의 WiseForm 에디터 홈페이지, http : //www.wiseform.co.kr.

[15] 에이원 프로사의 전자책 홈페이지, http : //www.aonepro.co.kr/ebook/kor_sol.html.

[16] 자바가상 기계 디렉토리, http : //www.geocities.com/SiliconValley/Lakes/6686/java.html.

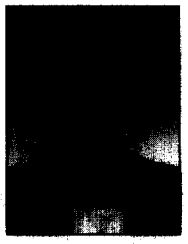
[17] 내장형 자바 가상기계 Kaffe 홈페이지, http : //www.kaffe.org .

[18] 블랙다운 자바 가상 기계 홈페이지, http : //www. blackdown.org.

[19] Open E-Book 컨소시엄 홈페이지, http : //www. open-ebook.org.

[20] Sun Microsystems J2ME 홈페이지, http : //java.sun.com/j2me.

[21] W3C 홈페이지, www.w3.org.



이 은 정

e-mail : ejlee@kyonggi.ac.kr

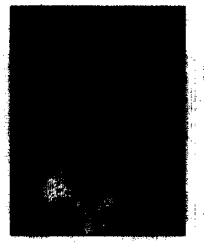
1990년 한국과학기술원 전산학과 석사

1994년 한국과학기술원 전산학과 박사

1994년~2000년 전자통신연구원 선임연구원

2001년~현재 경기대학교 정보과학부 전임강사

관심분야 : 프로그래밍 언어론, XML 미들웨어 등



조 수 선

e-mail : scho@etri.re.kr

1987년 서울대학교 계산통계학과 졸업 (학사)

1989년 서울대학교 대학원 계산통계학과 졸업(석사)

1989년~1994년 (주)웅진미디어 CBE개발부 연구원

1994년~현재 한국전자통신연구원 컴퓨터 소프트웨어기술연구소 선임연구원

관심분야 : 실시간 시스템 응용, 웹 기술 등