

Fluid Link를 이용한 웹 문서의 효율적인 브라우징

송승헌^{*)} · 윤보열^{*)} · 전형민^{*)} · 정선인^{*)} · 곽상호^{*)} · 김응곤^{*)}

순천대학교 정보과학대학원^{**)} · 순천대학교 컴퓨터학과^{*)}

Web Document's Efficient Browsing Using Fluid Link

Sung-hun Song^{*)} · Bo-yul Yoon^{*)} · Hyong-min Joun^{*)}
Sun-in Jung^{*)} · Sang-ho Gwak^{*)} · Eung-gon Kim^{*)}

Graduate School of Information Science, Suncheon National University^{**)}
Department of Computer Science, Suncheon National University^{*)}

E-mail : song9232@cs.suncheon.ac.kr

요 약

대부분 문서의 정보는 주 내용이 있고 여기에 보충하는 자료가 있어서 주석을 달든지 여백에 설명이나 그림을 넣는데 웹 문서에서는 이 자료를 하이퍼텍스트로 연결한다. 그러나 현재 사용하는 웹브라우저에서는 하이퍼텍스트의 링크를 통해 새로운 문서로 완전히 화면이 바뀌게 된다. 이때 사용자는 원래의 문서에서 떠나게 되어 전체의 흐름을 잃게 되고, 더 이상 관심이 없을 때 되돌아와야 하는 번거로움이 있다. Fluid link 기법을 이용하여 웹 문서 하이퍼텍스트 부분의 마우스 이동만으로 행간 혹은 옆 여백에 연결된 문서의 내용을 동시에 보여 줌으로써 원래 문서와 연결 문서로 불필요하게 오고가는 문제를 해결할 수 있다. 본 논문에서는 Fluid 링크를 이용한 여러 가지 브라우징 기법을 소개하고, 그 형태 및 장단점을 비교하며 Java 언어를 이용하여 실제 구현하는 방법에 대하여 논한다.

ABSTRACT

Most document consists of primary content and supporting material, such as footnotes, detailed explanations, and illustrations, and the related supporting materials are linked as hypertext on web document. However the content of hypertext links is appeared in the new windows on present web browser. Then the user will leave the primary material, may lose the entire context, and must have some difficulties to return to the primary context when the interest disappears. Using the technique for fluid links, we can solve these problems easily. If the mouse is putted on the link, the related material is presented in between lines or at margin maintaining the context of primary material. In this paper, we introduce the various browsing techniques using fluid links, analyze the forms and the features, and then we propose the best way to implement inJava.

1. 서 론

WWW의 출현으로 인해 인터넷이 대중화되었으며 그 사용자의 수는 폭발적으로 증가하게 되었다. 이 때 인터넷 정보를 효과적으로 이용하기 위하여 사용되는 소프트웨어인 웹 브라우저는 기능이 꾸준히 향상되어 왔으며[1], 많은 사람들에게 멀티미디어의 방대한 정보를 편리하게 제공해주었다.

웹 브라우저에서 사용되는 문서의 가장 큰 특징으로 하이퍼텍스트 기능이 있는데, 이는 일반 문서와 달리 기본 문서에 연결된 문서를 가진 입체적인 구조를 하고 있다. 이 하이퍼텍스트는 연결 고리를 통해 원하는 보충 자료로 바로 연결되

어 그 내용을 볼 수 있는 편리함이 있으나, 하이퍼텍스트의 링크를 통해 새로운 문서로 완전히 화면이 바뀌어 버리는 방향 상실(Disorientation)을 가져오며, 원래 문서의 흐름을 잃고 또 연결된 문서와의 관계를 떨어뜨려 파악해야 하는 인지적 부담(Cognitive Overhead)을 갖게 한다[2].

이는 정보의 제공과 습득 측면에 있어서 웹 페이지 제공자와 이용자 사이에 능률적으로 연결될 수 있는 매우 강력한 수단이 더욱 절실해졌으며, 이제까지와는 전혀 다른 새로운 형태의 접근이 제시될 필요하다는 것을 의미한다.

예를 들면, 현재의 웹 페이지 상에서는 어떤 하이퍼텍스트나 단어에 대해 그 내용을 파악하거

나 상세한 정보를 얻기 위해서는 그 하이퍼텍스트로 직접 이동하거나 설명을 위해 따로 마련된 공간으로 이동하여야 한다. 이러한 동작은 링크된 문서의 내용과 약이나 몇몇 단어의 이해를 위해서, 보고있는 원래 페이지에서 여러 군데의 HTML 문서로 옮겨 다니면서 여러 차례의 Forward와 Previous 버튼을 누르는 작업을 수행해야 한다는 것을 의미한다. 이러한 과정은 사용자에게 비능률적이며 정보제공자는 공유를 위해 링크시킨 정보에 대해 또 다른 링크를 연결한다.

따라서 본 논문에서는 이러한 문제를 해결하기 위하여 Fluid 링크를 이용한 여러 가지 브라우징 기법을 소개하고, 그 형태 및 장단점을 비교하여 Java 언어를 이용하여 실제 구현하는 방법에 대하여 논한다.

본 논문의 구성은 다음과 같다. 2장에서는 여러 가지 효율적인 Fluid 링크와 관련된 연구에 대해 서술하고, 3장에서는 Fluid 링크를 이용한 여러 가지 브라우징 기법들을 제시하며, 4장에서는 Java언어를 통해 Fluid 링크 기법을 실제 구현하는 방법에 대해 논의하며, 5장에서는 결론을 맺으면서, 앞으로의 과제를 소개한다.

II. 관련 연구

WWW을 비롯한 여러 일반적인 다른 시스템에서 문서나 정보를 브라우징 하기 위한 여러 기법이 발표되었다.

그 중 몇 가지 두드러진 예를 든다면, 첫째, Hypertext기법을 들 수 있다. 1954년 처음 개념이 소개된 이래[4], 오늘날 웹 페이지 상에서 서로 다른 물리적 위치의 정보를 연결시키는 고리로운 위치에서 다른 물리적 위치로 사용자를 이동시키는 중요한 역할을 수행해 왔다.

하지만 어떤 Hypertext의 내용을 알기 위해선 그 anchor의 위치로 반드시 이동을 해야 한다는 한계를 가지고 있다. 비록, 인터넷 익스플로러 버전 4.0[5] 이후에 마우스를 위에 가져다 놓는 것으로 anchor에 대한 간략한 정보를 풍선 도움말 형식으로 표현하는 것이 가능해 졌지만, 이것은 HTML의 제목이나 헤더에 간단한 단어의 삽입을 통한 제한적인 요소만을 나타낼 수 있을 뿐이므로 anchor의 본문 내용이라든가 그림이나 표 등 상세한 주석과 같은 자세한 정보를 표시하지 못한다. Fluid 링크 기법은 이러한 Hypertext의 제한된 기능을 개선할 수 있다.

Overlapping의 기법은 정보공간의 개별적인 요소들 사이에 시각적 연관성을 역동적이고 도식적인 방법으로 나타내는 기법이다. The NiF Elastic Catalog[6]의 수많은 목록들 중에서 단지 관련된 몇 줄만을 시각적으로 표시하기 위해 사용되었지만 Fluid 링크 기법에서는 단순한 시각적 표시만이 아닌 연결된 주석문들을 원래의 본문과 내용 상으로나 시각적으로 구분짓기 위해 사용된다.

Zooming 기법은 Pad++ 브라우저[4] 처음의 본문에서 사용자가 원하는 정보가 나타날 때, 그 선택되어진 정보를 화면의 앞쪽으로 확대시킴과 동시에 처음의 본문을 화면의 뒤편으로 축소시킴으로써 구현된다. 하지만 이 기법 역시 Hypertext와 마찬가지로 처음의 원문이 다른 한편에 가려지고 다른 링크나 주석문이 나타났다가 다시 사라지는 등의 반복적인 작업을 하는 형태를 갖고 있다.

가장 진보적이라 할 수 있는 Focus+context 기법은 전체 정보의 개략적인 내용을 한 눈에 파악하면서 동시에 사용자가 원하는 부분을 자세히 볼 수 있도록 하는 기법[7]으로서 화면에 나타내는 정보를 의도적으로 변화를 주어 실질적으로 나타내어야 할 정보와 더불어 전체적인 내용의 배치나 상태를 포괄적으로 표현하는 방법이다.

하지만 Fluid 링크 기법에서는 표현 형태상의 왜곡이나 공간을 집약시키는 일이 없이도 주석문의 크기를 직접적으로 변화시킴으로 Focus + context 기법과 동일하거나 더 나은 결과를 얻을 수 있다.

III. Fluid 링크를 이용한 브라우징 기법

Fluid 링크 기법을 시각화하여 브라우징에 실질적으로 적용하기 위해서 여러 가지 다양한 기법이 제시되어 왔다.[3]

이러한 기법들로는 화면의 남은 여백을 이용하여 주석을 표기하거나, 웹 문서 본문에 주석을 겹쳐 보이게 하기, 고해상도의 주석 매김 기법 등을 대표적인 예로 들 수 있다.

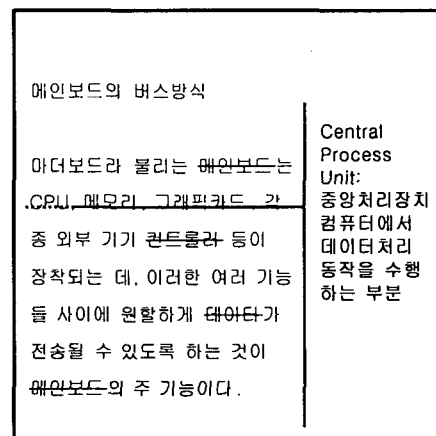


Fig. 1 화면의 여백에 주석을 표기하는 기법 (1)

화면의 여백에 주석을 표기하는 기법은 브라우저의 오른쪽 또는 왼쪽, 혹은 anchor의 바로 아래

쪽에 주석을 위한 공간을 따로 할당하는 방법을 사용한다. 단지 마우스 포인터를 가져다 대거나, 클릭하는 간단한 동작만으로도 anchor에 그어진 밑줄이 차츰 늘어나 문서의 오른쪽이나 왼쪽 여백에 anchor의 본문이나 내용을 설명하는 주석문을 [Fig.1]처럼 펼쳐지게 할 수 있으며, 시각적으로 본문 전체의 균형을 위해 양쪽의 여백을 사용하는 방법이 합당치 않다면 열과 열 사이의 여백을 늘려서 그 사이에 본문이나 주석을 위치시키는 방법을 [Fig.2]처럼 사용할 수 있다.

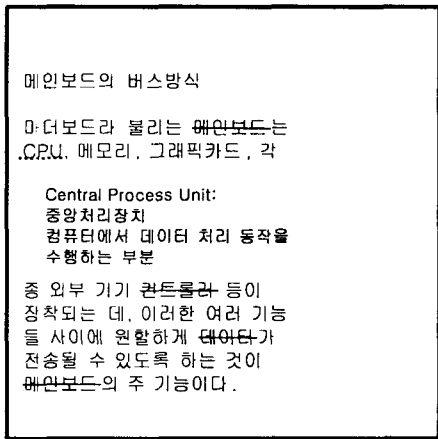


Fig. 2 화면의 여백에 주석을 표기하는 기법 (2)

다음으로, 웹 문서에 주석을 겹쳐 보이게 하는 기법은 여타 그래픽 프로그램에서 쓰이는 레이어

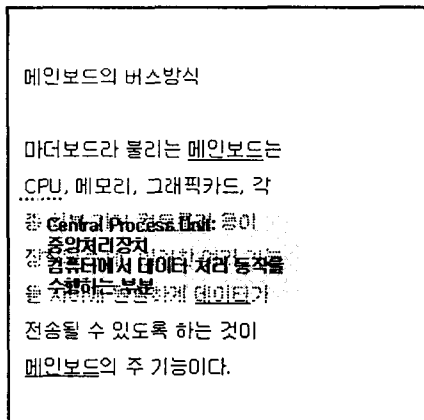


Fig. 3 웹 문서에 상에 주석을 겹쳐 보이게 하는 기법

개념과도 일맥상통한 것으로써 마우스 포인터를 anchor 위에 올려놓음으로도 [Fig. 3]처럼 원문의 단어에 링크된 anchor의 본문이나 주석문이 반투명의 셀 형태로 겹쳐져서 나타나는 형태를 가지고 있다. 이때 겹쳐진 본문과는 약간 다른 반투명의 색상을 가진 셀은 원문의 변화없이 충실하게 여다보여 짐으로 문서가 들쭉, 날쭉 혼란스럽게 변하는 상황을 피할 수 있으며 약간 돌출 되는 형태를 가지게 됨으로써 입체감을 부여하는 효과를 기대할 수 있다.

다음 설명할 고해상도의 시스템을 이용한 주석 매김의 기법인 [Fig. 4]는 일반적으로 사용되어지는 화면 해상도 값의 2-3배의 화면 해상도를 제공하는 시스템에서 구현될 수 있는 기법으로, 웹 문서가 원본 그대로 불러들여진 상태에서 화면상의 크기나 배열의 변화없이, 열과 열 사이에 3-4 point 정도의 작은 크기로 주석문을 표시함으로써 시각화하는 기법이다.

이 기법은 그 시스템의 지원만 따라준다면 원문의 시각적 변화 없이 직접적으로 주석을 제시하므로 사용자에게 더할 나위 없이 이상적인 방법이 될 것이다.

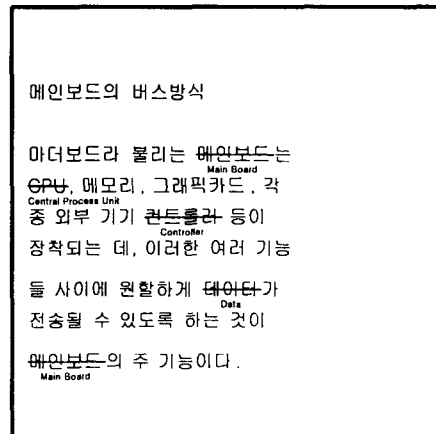


Fig. 4 고해상도의 시스템을 이용한 주석 매김의 기법

더 나아가, 여러 단계의 계층화로 표현되는 주석문들을 사용자에게 시각적으로 면밀히 묘사하기 위하여 Fig. 3의 주석을 겹쳐 보이게 하는 기법과 Fig. 1의 여백을 이용하는 기법을 사용, 동시에 표현하는 방법을 제안한다.

주석을 겹쳐 보이게 하는 기법으로써 [Fig. 5]처럼 원문의 형태를 유지하여 주석문을 첨부하며, 그 주석문의 주석을 위해 문서의 여백(엄밀히 말한다면 주석문의 여백)에 다시 주석을 덧붙인다면, 시각적 겹침이나 혼란없이 내용을 계층적으로

전달할 수 있다. 이 기법은 본문에 위치하게 될 셀 형태의 자세한 좌표지정과 주석문에 덧붙여질 새로운 주석 셀의 여백 위치 지정에 대한 일관성 있는 기준의 제시가 되어야만 한다.

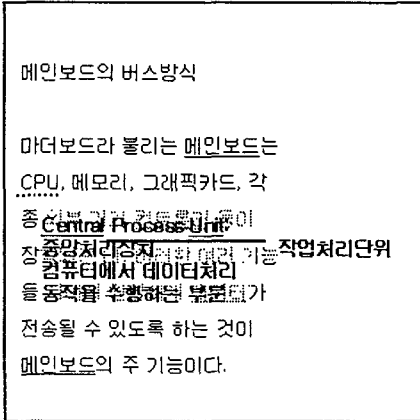


Fig. 5 주석문에 주석을 붙이기 위해 여백과 겹쳐 보임을 복합한 기법

지금 까지 제시된 기법들은 주석문이 붙은 상태에 주석문들을 다시 덧붙임으로써 각 단계별로 좀더 상세하고 세분화된 정보를 얻을 수 있게 한다.

그리고 이러한 주석문에는 문서의 저자, 웹 문서의 생성날짜, 위치한 URL, 관련된 여러 형식의 파일들에 대한 요약과 같은 메타 정보[8]들이 첨부됨으로써 제작자에 의해 제공되어진 정보가 새로운 정보로 가공되어 질 수 있는 2차 부산물로 유도되어지는 부가적인 효과를 얻을 수 있다.

IV. Fluid 링크의 구현

이 논문에서 Fig. 3과 같은 Fluid 링크 기법을 구현하기 위해 사용한 프로그래밍 언어는 Java이다.[9]

Java가 바이트 코드를 채택함으로써 Java는 구조적으로 중립성을 가지기 때문에 순수한 라이브러리로 작성된 프로그램은 어떤 종류의 컴퓨터에서도 실행될 수 있으며, Java의 애플릿 프로그램은 WWW와 연동되어 서버에서 전송받아 클라이언트에서 실행되는 형태를 취하기 때문에 네트워크 분산 환경에서 원격지의 파일이나 객체들을 직접 사용할 수도 있고, 원격지의 프로그램들과 상호 동작할 수 있는 여러 기법을 제공한다.

하나의 프로그램에 다 수의 컴퓨터가 접속하는 웹 환경에서 효율적으로 수행되기 위해서는 높은 신뢰성을 가져야 하는데 Java는 명시적인 포인터를 제공하지 않음으로 포인터에 기반한 C나 C++과 같은 언어에서 발생하는 많은 문제점들을 배

제하였으며, 할당된 메모리의 해제를 위해 자동 쓰레기 수집(garbage collection)의 기능을 수행하고, 엄격한 형 검사 수행 등과 같은 일련의 조치를 취함으로써 기존의 언어에서 나타나는 많은 에러원인의 요소들을 제거한다.

주석문의 겹쳐 보임을 시각화하기 위해서 웹 브라우저 상에서 나타내는 본문의 표현을 Java 애플릿 상에서 이루어지도록 하였으며 주석문의 표현을 위한 <Fluid>라는 새로운 형식의 태그를 삽입하고, HTML 문서와 애플릿의 상호작용을 묘사하기 위하여 FHTML이라는 새로운 문서형식을 제시하였다.

애플릿은 실행명령에 의해 실행되지 않고 WWW의 HTML 코드내에서 호출되어 실행되는 형태를 가지고 있으며, 애플릿을 해석하여 실행할 수 있는 인터프리터를 가진 브라우저에 애플릿 호출을 포함하고 있는 HTML 문서가 로드되면 브라우저는 HTML 문서가 로드된 서버에 접근하여 HTML에서 호출한 애플릿을 컴퓨터로 로드한다.

Java 애플릿을 사용하기 위해서 Java에서 제공하는 소켓과 서버 소켓인 java.net.Socket, java.net.ServerSocket 클래스를 사용하여 클라이언트-서버 환경을 구축하였다. 여기에서 소켓 클래스는 클라이언트가 서버에게 요청을 하기 위해 필요한 소켓을 생성하며, 데이터를 기록한 것과 같은 순서로 상대에게 전달되는 순차적 스트림을 제공한다.

```
s= new Socket(getCodeBase().getHost(), 2504)

/** 클라이언트에서 소켓을 생성하여 지정된 호스트 이름과 포트(2504)에 접속하는 메소드 사용의 예 */
```

서버 소켓 클래스는 통신 시 사용되는 소켓타입을 나타낸다.

```
ServerSocket ss = (ServerSocket)null;

:
ss= new ServerSocket(2504)

:
soc = ss.accept();
createSocketThread();
soc.close();

/** 서버측에서 클라이언트의 요청이 들어오는 지를 감시하기 위한 소켓을 생성하는 서버 소켓 클래스의 예 */
```

이러한 클라이언트 - 서버 환경은 연결된 컴퓨터간에 데이터를 전송시키는데, 보안성을 구축하는데 사용된다.

Fig. 6에서는 HTML 문서에서 Applet을 구현하기 위한 HTML 문서 형식을 보이고 있으며 애플릿을 사용하기 위해서는 HTML 문서내에 단지 <APPLET> </APPLET> 태그를 사용함으로써 포함되어진다.

```
<html>
<head>
<meta name="generator" content=" ModelWorks IDE">
<title>Fluid</title>
</head>
<body>
<h2>Test page for Fluid Class</h2><hr>
<applet code=Fluid.class width=300 height=300>
</applet>
<hr></body>
</html>
```

Fig. 6 HTML 문서에서 애플릿을 사용하기 위해 추가된 태그

```
<FHTML>
<head>
<title>Fluid Link Document Example</title>
</head>
<body>
<h2>메인보드의 버스방식</h2> <BR>
마더보드라 불리는 메인보드는<BR>
<Fluid Type=overlay Src=introduce.fhtml>CPU</Fluid>.
메모리, 그래픽카드, 각<BR>
중 외부 기기 컨트롤러 등이<BR>
장착되는데, 이러한 여러 기능<BR>
들 사이에 원활하게 데이터가<BR>
전송될 수 있도록 하는 것이<BR>
메인보드의 주 기능이다 <BR>
</body>
</FHTML>
```

Fig. 7 FHTML 문서의 형식

Fig. 7에서는 Fluid 태그를 도입한 FHTML 문서 형식을 나타내고 있다. FHTML은 Java 애플릿에서 HTML 문서를 인식시키기 위해 새롭게 구상한 문서형식으로, 새로운 형식의 <Fluid> 태그를 HTML 문서에서 적용시키기 위해 고안하였다.

Fluid 태그는 애플릿을 사용하는 방법과 비슷하게 <Fluid> </Fluid> 태그를 삽입하여 나타낸다.

```
<Fluid Type=overlay Src=introduce.fhtml>
CPU </Fluid>
/** Fluid 태그의 정리
overlay - 문자접침을 나타냄,
Src - 연결될 주석문서를 지정함.
이곳에서는 FHTML 문서로써 지정한다.
**/
```

Java 애플릿 내에서 마우스 이벤트로 Fluid 링크를 시동시키기 위해 사용되는 메소드 가운데 mouseEnter는 마우스 커서가 애플릿의 범위로 들어오는 경우 호출되는 메소드이다. 이 경우 좌표값의 입력이 필요하다.

```
public boolean mouseEnter
(Event e, int x, int y) {
display Fluid(this.type, this.fhtml)
return true;
}
/** 마우스 메소드 중 마우스를 가로지르는
이벤트로 작동하는 메소드 **/
```

Java에서는 GUI(Graphic User Interface)의 구축을 위해 java.awt(Abstract Window Toolkit) 패키지를 제공한다. 그중 [Fig. 8] 에서와 같이 패키지 중 PixelGrabber 클래스를 사용함으로써 Java 애플릿 상에 주석문의 위치와 크기를 지정하여 출력해 줄 수 있게 하였다.

```
import java.awt.*;
import java.awt.image.PixelGrabber;
public class Fluid extends java.applet.Applet
{
private int[] resolveImage (Image cpu, int width, int height)
{
int pixel [] = new int[300 * 100];
PixelGrabber pg =
new PixelGrabber(cpu, 0,0, width, height, pixel, 0, 300);
try {
pg.grabPixels();
}
catch(Exception e) {System.out.println(e.toString());}
return pixel;
}
```

Fig. 8 cpu 에 관한 주석문을 표현하기위해 PixelGrabber 클래스가 적용된 예

본 연구에서는 Windows 95 운영체제에서 JDK 1.2.2를 사용하여 개발하였으며, Fig. 9는 Fluid 링크 기법이 실지 적용되어 시각화된 구현 결과를 보여주고 있다.

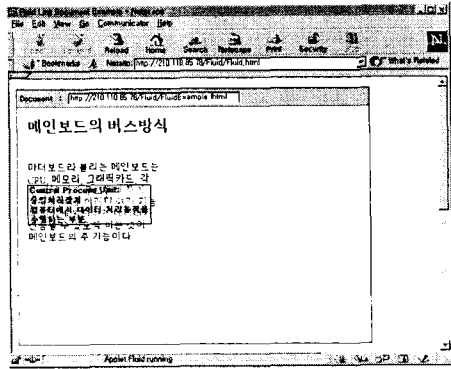


Fig. 9 Fluid 링크의 구현 결과

V. 결 론

본 논문에서는 웹을 향해서 사용하는 사용자들을 위한 보다 효율적인 웹 브라우징을 위하여 Fluid 링크를 이용한 여러 가지 브라우징 기법을 소개하고, 그 형태 및 장단점을 비교하여 Java 언어를 이용하여 실제 구현하는 방법에 대해 논하였다.

이제까지의 WWW 상에서는 자신이 필요로 하는 정보를 얻는 과정에서 정보제공자의 일방적인 게시로 웹을 탐색하는 이용자에 대한 배려가 거의 도외시되어 왔다. 본 논문에서는 웹 상에서 Java 언어를 이용한 Fluid 링크 기법의 적용으로 여러 단계의 불필요한 계층적 이동을 거치지 않고도 처음 검색된 기본적인 틀에서 링크된 문서들이나 주석처리된 자료들을 보게 함으로써 정보를 찾으려는 사람들에게 원하는 정보로의 더욱 쉬운 접근의 기회와 양질의 다양한 정보를 제공할 수 있다. 또한 정보를 제공하는 사람들에게 있어서는 일관성있는 자료들을 자신이 의도된 바에 따라 명확히 전달할 수 있게 되었다.

앞으로의 과제는 이미지나 사운드 등의 멀티미디어 정보가 들어 있는 문서를 Fluid 링크를 이용하여 효율적으로 처리하도록 하는 것이다.

감사의 글

본 연구는 과학기술부, 한국과학재단지정 여수대학교 "설비자동화 및 정보시스템연구개발센터"의 지원에 의한 것입니다.

참 고 문 헌

- [1] Nielsen, J, "The tyranny of the page : continued lack of decent navigation support in version 4 browsers", <http://www.useit.com/alertbox>, 1997
- [2] Bieber, M., Vitali, F., Ashman, H., Balasbramanian, V., and Oinas-Kukkonen, H., "Fourth generation hypermedia: some missing links for the World Wide Web", *Int. J. Human-computer Studies*, Vol. 47, pp31-65, 1997.
- [3] Polle T. Zellweger, Bay-Wei Chang, Jock D. Mackinlay, "Fluid Links for Inforamtion and Incremental Link Transitions", *Proceedings of Hypertext'98*, Pittsburgh, PA, pp 50-57, June 20-24, 1998
- [4] Bush, V., "As We May Think." *The Atlantic Monthly*, pp101-108, 176(July), 1945, Reprinted and discussed in interactions, pp35-67, 3(2), Mar 1996
- [5] Murtaugh, Michel., "NiF Elastic Catalog", <http://tablesaw.media.mit.edu/ecat/>
- [6] Bederson, Benjamin B., James D. Hollan, Jason Stewart, David Rogers, Allison Druin, and David Vick. "A zooming web browser", *Proc. SPIE Multimedia Computing and Networking'96*, Volume 2667, pp260-271, 1996.
- [7] Young-a Jung, "Implementation of a Zoom Browser using the Focus+Context Technique", pp9-12, Feb, 1999.
- [8] Dominic Stanyer and Rob Procter, "Improving Web usability with the link lens", *Computer Networks*, 31, pp1533-1544, 1999
- [9] Sun mycrosystems, "Java Language overview - white page", <http://java.sun.com/docs/overviews/java/java-overview-1.html>, 1999