소형 화면 단말기를 위한 웹 문서 변환 기법

신 희 숙[†] · 마 평 수[††] · 조 수 선[†††] · 이 동 우[†]

요 약

본 논문에서는 기존의 일반 PC 화면에 적합하도록 작성된 웹 문서를 무선 환경의 핸드헬드 계열의 소형 단말기 화면에서도 효율적으로 표현되어지도록 변환하는 기법을 제시한다. 이는 선행 연구에서 나타나는 단순한 텍스트 위주의 추출 및 요약 형식의 변환과는 달리, 시각적인 분리에 근거한 내용 블록 단위를 설정하고 이를 기본으로 변환을 수행함으로써 보다 정확한 변환 결과를 얻으며, 내용 블록 단위들의 재배치와 새로운 인덱스 형식의 재표현을 통하여 편리한 인터페이스로 좌우스크롤 없는 웹 문서를 제공한다. 이를 위하여 본 논문에서는 Layout-Forming Tag Analysis Algorithm과 Component Grouping Algorithm을 사용하여 시각적 표현을 주도하는 태그 정보에 대한 구조적인 분석 및 내용 블록 단위의 추출을 시도하고, 분리된 블록들의 분류와 재구성 및 인덱스 생성 과정을 통하여 소형 단말에 적합한 웹 문서를 생성한다. 웹 문서 변환 시스템은 프락시 서버에서 동작하도록 설계되었고, 프로토타입의 구현을 통하여 제시하는 변환 기법을 평가하였다. 실제 웹 문서에 대한 검증 과정을 거쳤고, 복잡한 구조의 웹 문서에 대해 적합한 변환 결과를 보였다.

Web Document Transcoding Technique for Small Display Devices

Hee Sook Shin[†] · Pyeong Soo Mah[††] · Soo Sun Cho[†††] · Dong Woo Lee[†]

ABSTRACT

We propose a web document transcoding technique that translates existing web pages designed for desktop computers into an appropriate form for hand-held devices connected to the wireless internet. By defining a content block based on a visual separation and using it as a minimum unit for analyzing and converting processes, we can get web pages converted more exactly. We also apply the reallocation of the content block and the generation of new index in order to provide convenient interface without left-right scrolling in small screen devices. These methods, compared with existing ways such as text level summary or partial extraction method, can provide efficient navigation and a full recognition of web documents. To gain those transcoding benefits, we propose the Layout-Forming Tag Analysis Algorithm that analyzes structural tags, which motivate visual separation, and the Component Grouping Algorithm that extracts the content block. We also classify and rearrange the content block and generate the new index to produce an appropriate form of web pages for small display devices. We have designed and implemented our transcoding system in a proxy server and evaluated the methods and the algorithms through an analysis of transcoded results. Our transcoding system showed a good result on most of popular web pages that have complicated structures.

키워드 : HTML, 문서 변환(Document Transcoding), 소형 단말기(Small Display Devices)

1. 서 론

언제 어디서든지 웹을 사용하고자 하는 사람들의 욕구는 무선 인터넷 환경을 창출하였고, 1990년대 중반 이후 소형 단말기 시장의 성장은 무선 인터넷의 접속을 증가시켰다[1, 3].

이에 무선 이동 환경을 위한 각종 프로토콜 및 미들웨어 플랫폼 분야의 기술 연구와 함께 웹 정보의 표현 분야에서도 새로운 마크업 언어의 등장을 시작으로 무선 환경에 좀더 효율적으로 대응하기 시작하였다. WML, HDML, cHTML 등과 특정 브라우저를 위한 독자적인 마크업 언어도 나타났고, 2000년대 이후에 들어서는 xHTML, XML 등의 확장된 표현 기술과 함께 시멘틱 웹이라는 개념이 새로 등장하면서 무선 웹의 다음 세대를 준비해 가고 있는 모습도 보이고 있다[22].

하지만 현재의 웹은 그 자체가 유선 환경의 인터넷을 위해 설계되었고, 기존의 유선 웹 상에 존재하는 수많은 정보 또한 데스크탑 환경의 사용자를 위해 제작되어진 것이다. 따라서 이를 소형 화면의 단말을 통하여 브라우징 할 경우, 단말에서 제대로 표현하지 못하는 문제가 발생하게 된다.

이러한 문제를 해결하고자 많은 연구가 선행되었으나, 간단한 텍스트 형식의 추출 및 요약으로의 변환 또는 수작업을 동반한 변환 방식을 사용하는 경우가 대부분이다[7, 9, 10, 12]. 따라서 본 논문에서는 내용 블록 단위를 기본으로 하여 원본 웹 문서의 정보를 보다 정확하게 전달하고 소형 화면에 적합하게 표현되어지도록 변환하는 기법을 제안한다. 이는 웹 문서의 시각적 표현을 주도하는 태그 정보에 대한 분석과 내용 단위로 분리된 조각들의 적절한 재구성 및 새로운 인덱스의 생성을 통하여 전체 웹 페이지에 대해 좌우 스크롤 없는 편리한 인터페이스로 브라우징할 수 있는 기능을 제공한다.

본 논문의 구성은 다음과 같다.

1장의 서론에 이어서 2장에서는 웹 문서의 변환과 관련

† 정 회 원 : 한국전자통신연구원 컴퓨터소프트웨어기술연구소 연구원
†† 정 회 원 : 한국전자통신연구원 컴퓨터소프트웨어기술연구소 책임연구원
††† 정 회 원 : 한국전자통신연구원 컴퓨터소프트웨어기술연구소 선임연구원
논문접수 : 2002년 8월 23일, 심사완료 : 2002년 10월 18일

된 기존의 연구와 개발 현황에 대해 살펴보고, 3장에서는 제시하고자 하는 웹 문서 변환 기법 및 시스템에 대해 구체적으로 설명한다. 3.1에서는 본 논문이 제시하는 변환 기법의 특징에 대해 소개하고, 3.2에서는 웹 문서의 분석 및 변환을 위한 주요 알고리즘에 대해 설명한다. 3.3에서는 전체 시스템 구조의 설계와 구현을 다루며, 3.4에서는 새로운 변환 기법의 테스트 결과를 분석 및 평가한다. 마지막으로 4장에서 향후 연구 과제에 대한 소개와 전체 논문 내용의 요약으로 결론을 맺는다.

2. 관련 연구

웹 컨텐츠의 변환은 단말의 성능, 특히 해상도에 따라 적용하는 변환 기법을 달리한다. 따라서 먼저 이동 단말기의 종류에 대해 살펴보고, 각각의 단말 종류를 대상으로 하는 변환 기법 연구 결과와 컨텐츠 변환 상용 제품들에 대해 살펴보고자 한다.

무선 환경에서 웹 접속을 시도하는 단말의 유형은 크게 세 가지로 구분되어진다[2].

첫째는, 노트북 계열로 기존의 데스크탑 PC 환경과 유사한 단말들이다. 최소 800×600이상의 해상도를 가지므로 웹 컨텐츠의 변환이 필요하지 않고, 통신망 자원만 확보된다면 충분히 유선 환경과 동일한 웹 서비스를 받을 수 있다.

둘째로는 핸드헬드 계열로 Palm-Size PC, Hand-Held PC 등 일반적으로 PDA(Personal Digital Assistant : 개인정보 단말기)로 통칭되는 단말들이다. 이들은 보통 3-5인치 크기의 화면으로 320×240정도의 해상도를 지원하나, 최근에는 화면 크기와 해상도가 높아지는 경향을 보이면서 640×480의 해상도를 지원하기도 한다(삼성NEXiO의 경우 800×480으로 현재 최대).

셋째로 셀룰러 폰 계열의 단말들을 들 수 있다. 이는 이동 통신 업체들이 무선 전화용 단말의 기능을 확장하여 무선 웹 서비스를 제공하면서 등장한 기기로, 90×60정도의 해상도를 지원하며 20줄 정도의 텍스트 위주로 된 제한된 정보의 표현만이 가능하기 때문에 대부분이 WML, HDML, cHTML 등의 마크업 언어를 통하여 정보를 표현한다.

이상에서 살펴본 각 단말들은 서로 다른 해상도를 가지고, 사용하는 브라우저도 달라서 이들이 인식하는 마크업 언어까지 달라지는 모습을 보인다. 특히 셀룰러 폰 계열의 단말의 경우는 작은 해상도로 인하여 HTML로 제작된 기존의 웹 문서를 지원하지 못하고 이동 단말을 위한 별도의 컨텐츠를 제작하여 사용하는 경향을 보인다. 이것은 WML, HDML 등이 HTML의 부분을 표현하는 마크업 언어이므로 HTML로 제작된 웹 문서의 모든 정보를 변환할 수 없기 때문이다. 따라서 자동 변환을 수행한다 하더라도 컨텐츠 요약 및 부분 발췌 정도의 수준으로 이루어지고, 기존 컨텐츠에서 많은 부분의 변화를 요구하게 된다. 이런 변환은 컨텐츠 제작자의 의도를 반영하지 못하여 잘못된 내용으로 변환하는 오류를 범할 가능성도 높아진다. 따라서 셀룰러 폰 계열의 단말을 위한 웹 문서는 특정 마크업 언어로 별도 제작하거나 또는 서비스 제공자나 사용자의 간단한 수작업으로 원하는 부분의 HTML 웹 문서를 특정 형식으로 변환하는 툴을 이용하는 방식이 보다 유용하다. 이런 방식을 지원하는 시스템으로는 IBM의 WebSphere Transcoding Publisher[19], OpenTV의 SpyGlass Prism[20], Argo의 WAP Tool[21] 등이 있다.

앞서 두 번째 유형으로 분류된 핸드헬드 계열의 단말들은 대부분이 HTTP/HTML을 지원하나 불편한 인터페이스와 소형 화면으로 인해 브라우징에 어려움을 준다. 이들 단말은 소형, 집적화 기술의 발달로 해상도를 높여가고 있으나, 이동단말기의 가장 큰 특징인 '이동성'을 고려한다면 그 크기에는 절대적인 제한이 존재하게 된다. 또한 데스크탑 PC의 디스플레이 성능도 함께 성장하므로 유선을 대상으로 하는 웹 문서가 무선으로 옮겨지기에는 여전히 상대적인 격차가 발생하게 된다. 이러한 단말을 위한 적절한 웹 문서 변환이 요구되고 있기에, 본 연구에서는 HTML 브라우저를 탑재한 핸드헬드 계열의 이동 단말들을 위한 웹 문서 변환 기법을 다루고자 한다.

이와 관련된 기존 연구들을 웹 문서가 변환되는 단계 또는 시점을 분류 기준으로 하여 크게 세 가지로 나누어 보면 다음과 같다. 웹 서비스 구조를 프락시가 존재하는 3-Layer로 구성했을 때 서버-프락시-클라이언트를 변환 수행 단계 또는 시점으로 하여 분류한 것이다.

첫째, 서버 측에서 웹 문서의 변환이 이루어지는 경우 : 여러 단말을 지원하기 위해 별도로 제작된 문서를 웹 서버가 가지고 있거나, 또는 별도의 표현 기법에 대한 정의를 단말별로 이미 가지고 있는 경우이다. 웹 문서의 실시간 자동 변환 시스템을 사용하여 소형 단말을 지원할 수도 있지만, 대부분의 웹 서버 운영자는 수작업을 동반한 변환 매카니즘[9, 19] 을 이용하고, 따라서 가장 정확한 변환이 이루어지는 장점을 가진다. 하지만 서버측에서의 변환은 제한된 웹 정보에 한해서 이루어지고, 기존의 유선상에 존재하는 웹 정보의 방대한 양에 비하여 변환 서비스가 제공되는 문서의 범위가 너무 제한적인 단점이 있다. 그러나 변환의 정확성을 보다 중요하게 여기는 상용 제품들의 경우는 간단한 수작업을 동반하는 서버측 변환을 많이 시도하고 있다[19-21].

둘째, 클라이언트 측에서 웹 문서의 변환이 이루어지는 경우 : 웹 문서를 전송받은 클라이언트 측에서 적절한 변환을 수행하는 것으로 사용자의 요구사항을 반영한 변환과 개인 특성화가 쉽게 구현될 수 있는 장점이 있다. 하지만 유선과 동일한 형태의 정보를 클라이언트가 받은 후에 변환 과정을 단말에서 수행함으로써 네트웍 자원의 비효율적인 사용과 클라이언트 단말의 높은 컴퓨팅 파워를 요구하게 된다. 대표적인 예로는 단말에서 특정 부위의 줌인 줌아웃 인터페이스를 제공하는 SmartView[13], Pad++[11] 등이 있다. 앞으로 무선 망과 소형 단말의 성능이 크게 발전한다면 클라이언트 측의 변환 기법도 많이 활용되어질 것으로 보인다.

세째, 프락시 측에서 웹 문서의 변환이 이루어지는 경우 : 대부분의 자동 변환 시스템은 프락시로 동작하면서 다양한 단말들을 지원하기 위한 변환을 시도한다. 따라서 기존의 많은 논문[4, 12, 14, 18]에서도 프락시를 활용한 변환을 다루고 있는데, 이는 프락시의 기본 기능이 무선 환경의 단점을 보완하는 역할로써 적합하기 때문이다. 이 기법은 반드시 프락시 서버를 거쳐야 한다는 조건을 가지나 이미 유선 네트웍 환경에서 대부분의 인트라넷 등이 컨텐츠의 재활용, 보안, 네트웍 망의 효율적인 관리 등을 목표로 프락시 서버를 사용하고, 여기에 무선 환경에서는 클라이언트 단말이 반드시 거쳐야 하는 엑세스 포인트 또한 존재하므로 적절한 네트웍 망의 구성과 프락시의 배치로 해결할 수 있는 단점으로 본다[15, 16]. 관련 연구로는 팜파일럿 단말의 브라우저를 위한 변환 프락시를 선보이는 Top Gun Wingman[12]과 상용 제품으로 자동 변환을 수행하는 SpyGlass Prism[20] 등이 있다. Top Gun Wingman의 경우 브라우저에서의 변환도 부분적으로 수행되기도 하나 대부분의 변환은 프락시에 의해서 주도되고, Prism의 경우는 변환 툴도 함께 제공하고 있다. 대표적인 프락시 기반 자동 변환 시스템으로 Digester[4, 5]를 들 수 있는데, 이는 핸드헬드, 셀룰러 폰 계열의 단말을 모두 지원하며, 사람에 의한 직접적인 변환 수행을 통해 얻은 다양한 휴리스틱 변환 기법과 이들의 적절한 적용 규칙을 제시하고 있다.

이상으로 기존의 관련 연구들을 크게 세 가지로 분류하여 살펴보았는데, 물론 모든 기법들이 장단점을 가지므로 특정 환경, 특정 서비스에 대해서 좀더 적합한 변환 기법을 선별할 수 있을 것이다. 본 논문에서는 보다 많은 웹 정보를 다양한 클라이언트 브라우저에게 적합한 표현 형식으로 자동으로 변환하여 제공하고, 자원의 효율적인 사용을 위하여 프락시 기반의 변환 기법을 사용하고자 한다. 따라서 프락시의 기본 기능에 웹 문서 변환을 위한 모듈들을 추가하는 방법을 사용하여 프락시 기반의 웹 문서 변환 시스템을 설계하였다. 물론 경우에 따라서는, 본 논문에서 제시하는 변환 기법을 클라이언트 또는 서버에서 동작하는 변환 모듈로 응용할 수도 있을 것이다.

그 외에 컨텐츠 변환 기법의 다른 연구 분야로는 웹 문서를 분석하고 특정 부분을 추출해 내는 기법에 대한 연구[7-10]가 있다. 태그 규칙을 이용하여 특정 정보 부분을 추출하는 휴리스틱 알고리즘에 대한 연구[10], 시각적 근거를 이용하여 웹을 구조적으로 분석하는 기법을 다루는 연구[7], 리스트 형식의 텍스트 부분을 추출하여 VoiceXML로 변환하는 기법에 관한 연구[17]등이 있다.

3. 소형 화면의 단말기를 위한 웹 문서 변환 기법 및 시스템

3.1 본 논문이 제시하는 변환 기법의 특징

앞서 살펴본 다양한 선행 연구들과 달리 본 논문에서 제시하는 소형 화면의 단말기를 위한 웹 문서의 변환 기법은 다음과 같은 특징을 가진다.

첫째, 웹 문서에서 특정 부분을 추출 및 요약하는 것이 아니라 전체 웹 문서를 내용 블록 단위로 재배치 및 재표현함으로써 명확한 정보 전달을 이루고자 한다. 본 논문이 대상으로 하는 핸드헬드 계열의 이동 단말들은 320×240 또는 640×480의 해상도와 64MB이상의 RAM, 130MHz이상의 CPU 성능을 가지고 HTML 브라우저가 동작하는 환경이 갖추어져 있다[2, 3]. 이러한 단말의 향상된 성능과 함께 복잡한 구조에 많은 정보를 한꺼번에 표현하는 현재의 웹 문서의 특징을 반영하고자 한다. 여기에 멀티미디어 등과 같은 고급기능을 요구하는 PDA 사용자들의 성향을 고려한다면, 기존의 텍스트 요약 또는 부분 정보 추출 등의 방법으로는 이미 화려한 웹 문서에 익숙해진 사용자들의 욕구를 충족시킬 수 없을 것이다. 따라서 객체 및 태그 집합의 재배치로 전체 문서를 재구성하는 개념을 가지는 Re-composition 변환 방식을 제시하고자 한다. 선행 연구[4]에서는 변환의 정도에 따라서 이미지 포맷의 변환과 같이 표현 형식을 바꿔주는 Transformation과 삭제 또는 생략으로 처리하는 Elision 방식으로 분류하고 있다. 하지만 본 논문에서는 기존의 웹 문서가 가진 정보의 표현력을 그대로 전달하기 위하여 Re-composition으로 분류할 수 있는 변환 기법을 사용하고, 여기에 이미지, 플래쉬 등의 많은 컴퓨팅 자원을 요구하는 특정 객체에 한하여 Elision 또는 Transformation 방법을 부분적으로 사용하고자 한다. 이를 위해 시각적 표현을 주도하는 태그 정보와 문서 구조에 대한 semi-semantic 정보를 활용하여 웹 문서를 분석하고 내용 블록 단위를 분리 및 재배치하는 과정을 수행하게 된다. 이러한 과정은 기존의 구문 자동 변환 기법이 가지는 불명확한 변환을 보완하고 특정 내용 단위를 기본으로 하여 원본 웹 문서의 정보를 최대한 반영하게 된다[6].

둘째, 새로운 인덱스의 생성과 좌우 스크롤이 없는 구성으로 편리한 인터페이스를 제공하고자 한다. 많은 정보를 작은 화면 안에 집약적으로 표현하기 위하여 인덱스 역할을 하는 특정 내용 블록 단위를 선별하여 인덱스 선택 리스트로 재표현하고, 나머지 블록들은 사용자 단말의 화면 크기를 고려한 재배치로 좌우스크롤 없이 표현한다. Digester[4, 5]의 경우는 인덱스 페이지의 생성과 계층 구조를 가지는 다수의 문서 조각으로 나누는 변환을 수행하는데, 이러한 기존의 페이지 조각 나눔이나 카테고리 분류 방식은 정리된 표현이라는 의미에서 정보 인식을 도울 수 있으나, 분류의 단계가 많아지면 검색이 어려워지고 다수의 링크 연결은 이전 페이지로 되돌아가기에 불편함을 주기도 한다. 따라서 본 논문에서는 추출된 특정 블록 전체를 인덱스로 변환하는 방법과 내용 블록들을 좌우 스크롤 없이 재배치하는 방법에 대해 제안하고 있다.

셋째, 시각적 표현을 주도하는 태그 집합 및 속성값 등을

활용한 웹 문서 분석 방식을 사용하여 간단한 알고리즘을 통한 변환을 시도한다. 대부분의 웹 문서는 명확한 내용 전달을 위하여 의미상의 차이를 가지는 컨텐츠를 시각적으로 분리되도록 표현하는 특징을 가진다. 따라서 이러한 특성을 기반으로 시각적 분리를 나타내는 태그들의 분석으로부터 페이지 레이아웃의 semi-semantic 정보를 추출하고 이를 변환에 활용한다. [7]의 논문에서도 시각적 분리 표현 근거를 기반으로 한 연구를 소개하고 있으나, 내용 블록 단위의 추출을 목표로 하는 본 논문과는 달리 시각적 유사성을 이용한 패턴 추출 기법을 제안하는 차이점이 있다. 또한 Digester에서 제시한 다양한 휴리스틱 변환 기법들의 조합 적용 과정보다 간단한 알고리즘을 적용함으로써 실시간 자동 변환 시스템의 중요 변수인 변환 시간에 대한 성능을 높이고자 한다.

이와 같은 특징을 가지는 웹 문서 변환 기법은, 구조적 태그 및 속성의 분석을 통하여 전체 웹 문서를 내용 단위 조각으로 설정하는 Layout-Forming Tag Analysis Algorithm과 정의된 내용 단위 조각을 클라이언트 성능 정보와 내용 단위 조각의 속성 값에 따라서 유사한 조각들로 그룹화하여 내용 단위 조각 블록을 생성하는 Component Grouping Algorithm의 수행을 통하여 이루어진다. 따라서 본 논문에서는 이 두 가지 알고리즘과 함께 Component Block의 분류 과정과 인덱스 생성 기법 및 Component Block의 재구성 기법을 제안한다.

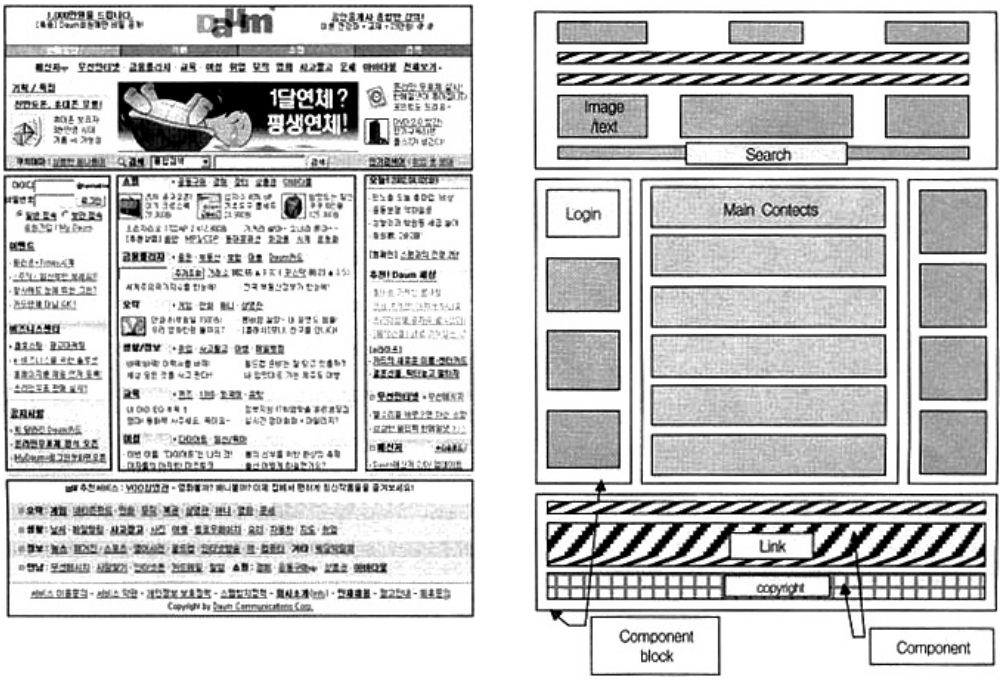
3.2 웹 문서 분석 및 변환을 위한 주요 알고리즘

본 연구에서는 웹 문서의 구조와 태그 사용에 대한 패턴, 웹 문서의 설계상의 공통점, 변환에 필요한 휴리스틱 기법 등을 찾고자 연구 대상 샘플 사이트를 선정하였다. 이는 한국 인터넷 정보 센터(KRNIC)[25]에서 제시한 웹사이트 분석 및 평가 전문 사이트들-랭키닷컴(www.rankey.com), 인터넷 메트릭스(www.internetmetrix.com), 코리안클릭(www.koreanclick.co.kr)-에서 제공하는 국내 유선 웹 페이지 사용(접속) 순위 통계치를 근거로 하였다.

각 조사기간에서 제시한 순위별 20개의 사이트를 선별하여 특징 조사를 수행하였고, 이를 내용별로 분류하였을 때, 커뮤니티 포탈이 6개, 종합검색 엔진이 5개로 모두 상위 11위권 이내에서 나타났고, 기타로 종합일간지, 스포츠 신문, TV 방송, ISP, 게임포탈, 기업 홈페이지 등이 있었다. 또한 상위 5개의 사이트가 전체 유선 웹 페이지 접속율의 95% 이상을 차지하는 특색도 보였다.

이들 웹 사이트들이 제공하는 정보와 표현 방법은 다양하나 전반적으로 한 화면 안에서 많은 정보를 한꺼번에 표현하고 링크와 이미지 객체를 주로 사용하는 특징을 보인다. 또한 대부분의 웹 사이트가 <TABLE> 요소를 사용하여 전체적인 웹 문서의 구조를 형성하고, 색상값, 이미지, 공백 표현 등을 사용하여 컨텐츠 내용의 논리적인 구분을 시각적으로 표현하고 있다. 즉, 전체 사이트를 시각적인 구분에 따라 나눈다면, 그것이 내용에 근거한 상이한 조각으로 분리가 되는 것이다. 이는 웹 문서를 디자인할 때 명확한 내용 전달을 위해서 의미상의 차이를 가지는 내용에 대해서 레이아웃 및 구조적 태그를 사용하여 시각적인 분리가 이루어지도록 구성하기 때문이다.

(그림 1)의 예제 사이트를 통하여 확인해 보면, 왼쪽 그림에서와 같이 시각적인 분리에 따라 전체 페이지를 조각 블록으로 구분하면 서로 상이한 내용을 가지는 블록으로



(그림 1) 시각적인 분리로 상이한 내용 블록을 표현하는 웹 문서의 예제도

구분되어 짐을 알 수 있다. 그리고 오른쪽의 그림에서와 같이 각 조각은 좀더 세분화된 단위로 나누어질 수 있고, 본 논문에서는 이와 같은 최소 내용 단위 조각을 Component로, 유사성을 가지는 최소 내용 단위 조각들을 그룹화한 것을 Component Block으로 정의한다.

웹 문서 특징 조사에서 나타난 바와 같이 구조적 태그, 특히 <TABLE>을 분석함으로써 전체 문서의 구조를 파악할 수도 있으나, 일부 무분별한 <TABLE>의 사용과 HTML 자체가 가지는 구조와 의미의 불명확한 구분 문제를 고려하여, <TABLE> 뿐 아니라 태그의 속성값, 포함하는 데이터의 특성, 객체의 위치정보 등도 함께 이용하여 웹 문서를 분석하고자 한다. 즉 태그 자체의 구문 분석과 함께 semi-semantic 정보를 활용함으로써 보다 명확한 웹 문서 분석을 수행하고자 한다. 여기서 <FRAME>을 이용한 분리 표현도 가능하나 웹 문서 특징 조사에서 <FRAME>을 사용한 사이트는 10% 미만으로, 특히 접속 순위 상위 사이트에 대해서는 거의 나타나지 않으므로 페이지 링크 연결로 <FRAME>을 처리하는 방법을 이용한다.

본 논문에서 제시하는 웹 문서의 변환 기법은 다음과 같은 주요 과정을 통하여 수행된다. 먼저 Layout-Forming Tag Analysis Algorithm을 이용하여 원본 웹 문서에서 Component 단위를 정의하고 분석에 필요한 semi-semantic의 구조적 정보를 얻는다. 이후 Component Grouping Algorithm을 통하여 정의된 Component를 단말의 화면 크기에 적합한 width를 가지도록 Component Block으로 묶고, 이때 각 블록은 표현 및 배치상의 유사성을 가지게 된다. 추출된 Component Block은 포함하는 컨텐츠의 특성에 따라 INDEX형 또는 BODY형으로 분류되고, 각각은 인덱스 형식으로 재표현 되어지거나 또는 순차적으로 재배치되어진다. 다음의 (그림 2)는 간단한 HTML 예제를 통한 변환 전 후의 모습을 나타낸다. 이 때 사용된 HTML 예제 코드는 (그림 4)에서 제시한다.

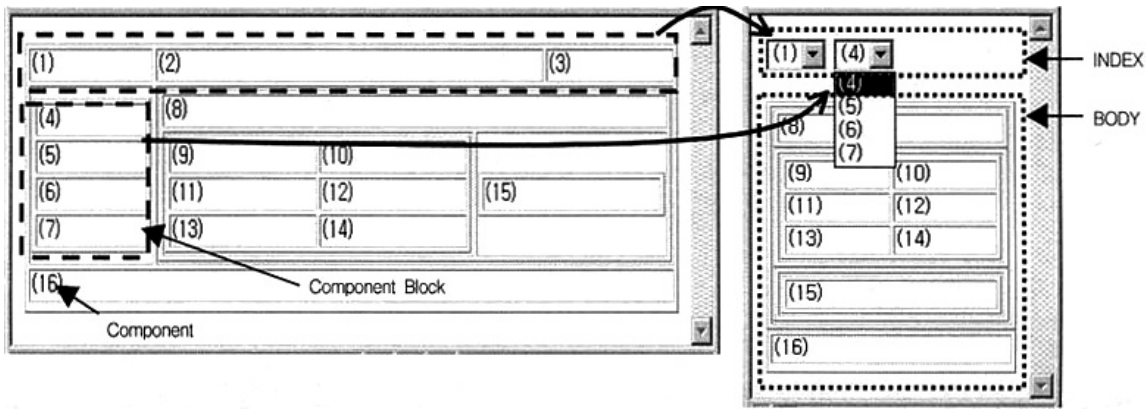
(그림 2)의 Component (1), (2), (3)과 Component (4), (5), (6), (7)은 각각 Component Block으로 묶여지고 INDEX형으로 분류되어 상단의 선택 리스트 형식으로 재표현 되어진다. 나머지 4개의 Component Block 즉, (8), (9, 10, 11, 12, 13, 14), (15), (16)은 BODY형으로 웹 페이지의 중앙에 차례대로 재배치된다.

3.2.1 Layout-Forming Tag Analysis Algorithm

이 알고리즘은 분석의 기본 단위가 되는 Component를 설정하고 이후 단계에서 필요로 하는 구조적 정보를 Component 단위로 추출하는 과정을 수행한다. 여기서는 <TABLE>, <TR>, <TD>, <IMG> 등의 태그를 주로 분석하고 특정 <TD> 요소를 Component로 설정한다. Component는 <TABLE> 이외의 요소를 내부에 포함하는가에 따라서 general Component와 inclusive Component로 구분하여 분석 과정을 수행하게 된다. 분석 과정을 거치면서 <TD> 요소는 <표 1>과 같은 semi-semantic 정보를 가지게 되고, 이 정보는 이후 Component Block의 추출 및 분류 단계에서 이용된다. 여기서 semi-semantic 정보는 구조적 표현을 주도하는 태그들의 구문 분석을 통하여 태그 정보가 내포하는 페이지 구조의 정보를 부분적으로 추출하여 얻는다. 본 논문에서는 <TABLE>의 width 속성값, 중첩 횟수, 전체 레이아웃에서의 상대적인 위치값, 이미지 속성값 등을 semi-semantic 정보로 이용한다.

〈표 1〉 〈TD〉 요소에 추가되는 semi-semantic 정보

Width	픽셀 단위로 재계산된 width 값.
CompNum	Component로 설정되는 경우 Component의 ID를 표현하는 값. • general Component : (sequence number, 0, 0). • inclusive Component (<TABLE> 이외의 요소를 데이터로 가지는 경우) : (0, 첫째자식의 CompNum의 첫 번째 숫자, 마지막 자식의 CompNum의 첫 번째 숫자).
ColNum	전체 레이아웃에서 몇 번째 Column에 위치하는가를 나타내는 숫자.
RowNum	전체 레이아웃에서 몇 번째 Row에 위치하는가를 나타내는 숫자.
TableDepth	중첩된 조상 <TABLE>의 개수. <TABLE>의 중첩 횟수를 나타냄.



(그림 2) 웹 문서 변환 기법의 주요 예제도(좌 : 변환 전, 우 : 변환 후)

```
input tag node tree ;
REPEAT {
  extract next tag node with preorder traversal ;
  IF ( tag == <TABLE> ) {
    IF ( TableDepth > threshold & Width <= MAX_WIDTH ) {
      define Width of all elements inside < TABLE > ;
    }ELSE {
      increase TableDepth ;
      define Width of < TABLE > element ;
    }
  }ELSE IF ( tag == < TR >) {
    IF (tag is not in the first row of nested table) increase
      RowNum ;
    IF ( tag's parent is root table ) ColNum = 0 ;
      define Width of < TR > element ;
  } ELSE IF ( tag == < TD >) {
    IF ( tag is not in the first row of nested table ) increase
      ColNum ;
    IF ( tag has < TABLE > as child node) define CompNum with
      ( 0, child < TABLE >.first < TD >.CompNum,
       child< TA BLE >.last < TD >.CompNum ) ;
    ELSE define CompNum with ( sequence, 0, 0 ) ;
    define Width of < TD > element ;
  } ELSE IF ( tag == < IMG >) {
    convert image format ;
    set Width of image ;
    IF (< MAP > is used for the image ) modify COORDS attribute
      value of < AREA > ;
  } ELSE IF ( other tags ) trivial functions for other tags ;
} UNTIL ( end-of-tag of HTML )
```

(그림 3) Layout-Forming Tag Analysis Algorithm

원본 HTML이 파싱되어 태그를 노드로 가지는 트리 데이터 구조로 표현되었을 때, 각 노드를 Preorder Traversal로 방문하면서 (그림 3)의 알고리즘을 수행한다. MAX_WIDTH를 사용자 단말 화면의 최대폭으로 설정하고 중첩적으로 사용되는 <TABLE>에 대해서는 분석 과정에 제한을 두어, 중첩 횟수가 경계값을 초과할 경우에는 <TABLE>과 그 아래의 모든 자손 노드를 부모 <TD>의 일반 데이터 값으로 간주한다. 중첩 횟수의 경계값에 따라 분석의 단계 및 반복 횟수가 조정되어 보다 작은 단위 조각에 대한 분석을 수행할 수 있고, 경계값의 비교와 함께 <TABLE>의 width 속성값의 비교를 통하여 MAX_WIDTH를 초과하는 경우는 분석 단계를 계속 수행하도록 한다. 그러나 일반적으로 MAX_WIDTH가 400인 경우, <TABLE>의 중첩이 3회를 초과하면 <TABLE>의 width는 MAX_WIDTH 미만의 값이 되어 전체 페이지 레이아웃 구성에는 영향을 주지 못하고 Component 또는 Component Block 내에서의 세부 조각 나눔에 사용되어 그 내부의 <TABLE>에 대한 분석이 무의미함을 보였다.

Width 설정 과정에서는 width를 속성으로 가지는 모든 태그에 대해 %로 설정된 값을 픽셀로 환산하고, MAX_WIDTH를 초과하거나 값이 설정되어 있지 않을 경우는 <TR> width의 최대값, <TD> width의 총합, <IMG> width의 최대값 등을 이용하거나 또는 MAX_WIDTH를 새로운 속성값으로 설정한다.

다음 (그림 4)의 예제를 통하여 위의 분석 알고리즘으로부터 얻은 구조적 semi-semantic 정보를 확인해 보면 <표 2>와 같은 결과가 나타난다. (그림 5)는 (그림 4)의 HTML 예제 코드를 랜더링했을 때 나타나는 시각적 모습으로 <TABLE>, <TR>, <TD>, Component를 강조 표현한 것이다.

〈표 2〉 Component에 추가되는 구조적 semi-semantic 정보

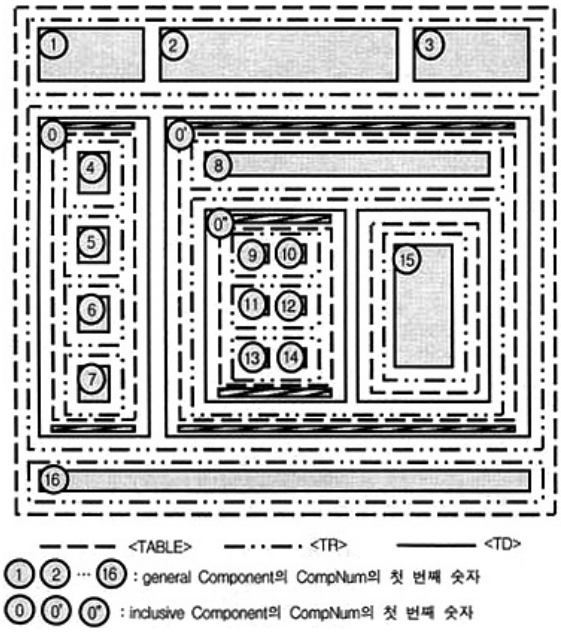
(A) : CompNum의 첫 번째 숫자
(B) : CompNum, RowNum, ColNum, TableDepth, Width의값 ((MAX_WIDTH < 500)로 가정)

(A)	(B)	(A)	(B)
①	(1, 0, 0), 1, 1, 1, 150	ⓞ"	(0, 9, 14), 3-5, 2-3, 2, 400
②	(2, 0, 0), 1, 2, 1, 500	⑨	(9, 0, 0), 3, 2, 3, 200
③	(3, 0, 0), 1, 3, 1, 150	⑩	(10, 0, 0), 3, 3, 3, 200
ⓞ	(0, 4, 7), 2-5, 1-1, 1, 150	⑪	(11, 0, 0), 4, 2, 3, 200
④	(4, 0, 0), 2, 1, 2, 150	⑫	(12, 0, 0), 4, 3, 3, 200
⑤	(5, 0, 0), 3, 1, 2, 150	⑬	(13, 0, 0), 5, 2, 3, 200
⑥	(6, 0, 0), 4, 1, 2, 150	⑭	(14, 0, 0), 5, 3, 3, 200
⑦	(7, 0, 0), 5, 1, 2, 150	⑮	(15, 0, 0), 3, 4, 3, 250
ⓞ'	(0, 8, 15), 2-5, 2-4, 1, 650 -> MAX_WIDTH	⑯	(16, 0, 0), 6, 1, 1, 800 -> MAX_WIDTH
⑧	(8, 0, 0), 2, 2, 2, 650->MAX_WIDTH		

```
< TABLE border = "1" >
    < TR >
      < TD width = "150" > (1) < /TD >
      < TD width = "500" > (2) < /TD >
      < TD width = "150" > (3) < /TD >
    < /TR >
    < TR >
      < TD >
        < !-- Other Tags (0) -- >
        < TABLE border = "1" >
          < TR >
            < TD width = "150" > (4) < /TD >
          < /TR >
          < TR >
            < TD width = "150" > (5) < /TD >
          < /TR >
          < TR >
            < TD width = "150" > (6) < /TD >
          < /TR >
          < TR >
            < TD width = "150" > (7) < /TD >
          < /TR >
        < /TABLE >
        < !-- Other Tags (0) -- >
      < /TD >
      < TD colspan = "2" >
        < !-- Other Tags (0') -- >
        < TABLE border = "1" >
          < TR >
            < TD width = "650" colspan = "2" > (8) < /TD >
          < /TR >
          < TR >
            < TD >
              < !-- Other Tags (0") -- >
              < TABLE border = "1" >
                < TR >
                  < TD width = "200" > (9) < /TD >
                  < TD width = "200" > (10) < /TD >
                < /TR >
                < TR >
                  < TD width = "200" > (11) < /TD >
                  < TD width = "200" > (12) < /TD >
```

```
                    </TR>
                    <TR>
                        <TD width = "200"> (13) </TD>
                        <TD width = "200"> (14) </TD>
                    </TR>
                </TABLE>
                <!-- Other Tags (0") -->
            </TD>
            <TD>
                <!-- nothing -->
                <TABLE border = "1">
                    <TR>
                        <TD width = "250"> (15) </TD>
                    </TR>
                </TABLE>
                <!-- nothing -->
            </TD>
        </TR>
    </TABLE>
    <!-- Other Tags (0') -->
</TD>
</TR>
<TR>
    <TD width = "800" colspan = "4"> (16) </TD>
</TR>
</TABLE>
```

(그림 4) 요약된 HTML 예제 코드



(그림 5) (그림 4)의 HTML에서 정의되는 Component와 구조적 태그의 시각적 구성

3.2.2 Component Grouping Algorithm

Component Grouping Algorithm에서는 Layout-Forming Tag Analysis Algorithm에서 정의된 Component 단위를 기준으로 단일 또는 다수의 Component에 대해 그 내부에 포함되는 모든 태그 집합을 별도 <TABLE>의 단일 <TD>로 묶어서 상위 조상 <TABLE>과 동등한 위치에 삽입하여 Component Block을 생성한다.

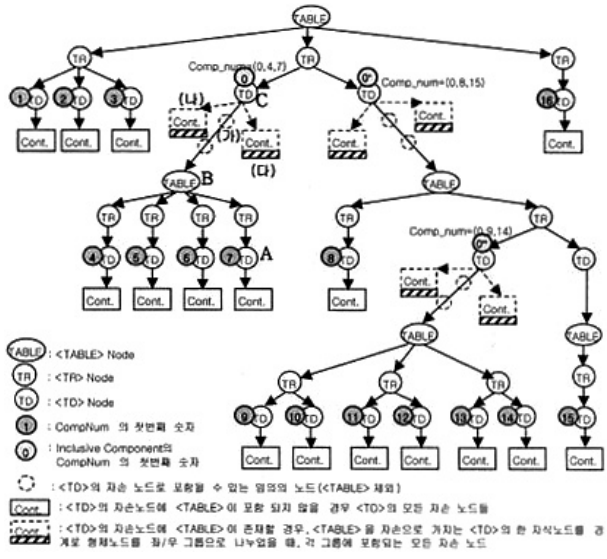
이 알고리즘은 앞서 정의된 Component를 검색하여 Component의 초기 Width 값이 MAX_WIDTH 이하인 경우에만 다수의 Component를 하나로 묶는 그룹화 과정을 수행한다. Component의 Width가 MAX_WIDTH를 초과하는 것은 내부 구조 분리 없이 데이터만을 가지는 경우이므로 width 속성값의 강제 조정으로 이미지, 텍스트 라인 등을 변경한다.

(그림 6)은 이 알고리즘을 나타내고, (그림 7)은 (그림 4)의 구조적 태그 집합을 트리 모형으로 나타낸 것으로 태그 간의 계층 관계를 쉽게 파악할 수 있다.

```
input Component node tree ;
REPEAT {
    extract next Component node with preorder traversal ;
    IF (Component has sibling nodes) {
        make Component group with the sibling nodes ;
    }
    // ex. nested table block (가) in Fig.7
    make table block with Component group or Component ;

    IF (table block has last < TD > & parent < TABLE > of
        current Component is nested table & first ancestor < TD > of the Component is inclusive Component) {
        // ex. other elements (나), (다) in Fig.7
        make table block for each two groups of child nodes ;
        // ex. C in Fig. 7
        make table block with first ancestor < TD > ;
    }
    rearrange Component Block ;
} UNTIL ( end-of-Component node )
```

(그림 6) Component Grouping Algorithm



(그림 7) (그림 4)의 HTML 예제의 구조적 태그 집합의 트리 모형

먼저 Component 노드의 형제 노드가 있는지 확인하고, 있다면 MAX_WIDTH가 넘지 않는 범위 내에서 유사한 형제 노드들을 묶는 그룹화 과정을 수행한다. 이때 유사성의

여부는 <표 1>의 구조적 정보와 <표 3>의 패턴 비교 변수 값, 그리고 Component가 포함하는 데이터 유형의 비교로 결정한다. (그림 7)의 ①, ②, ③의 Component는 (①), (②),(③) 또는 (①③), (②)의 그룹으로 묶을 수 있다. 테이블 블록화 단계에서는 각 그룹에 속하는 모든 태그 집합을 '<TABLE><TR>Component①,③</TR></TABLE>'과 같은 형식으로 Component 그룹을 단일 <TABLE> 요소로 표현하고, 이를 테이블 블록으로 정의한다. 테이블 블록의 재배치 단계에서는 상위 과정에서 새로이 생성된 테이블 블록을 Component 노드의 부모의 부모 노드인 <TABLE> 노드의 이전 형제 노드로 삽입한다.

<TD>는 <TABLE> 이외의 많은 다른 객체를 포함할 수 있으므로 이들을 적절한 위치에 배치하는 과정이 필요하다. 따라서 (그림 7)의 Component ⑦의 예제와 같이, A가 B의 마지막 <TD>이고 B가 중첩된 <TABLE>이면, B를 자손으로 가지는 상위 조상 <TD> C가 inclusive Component일 경우에 즉, <TD>가 <TABLE> 이외의 다른 객체를 내부에 포함하고 있을 때, <TD>의 자식 노드를 2개의 테이블 블록으로 정리하는 과정을 수행한다. (그림 7)의 경우 C의 자식 노드 중에서 B를 포함하는 자식노드 (가)를 기준으로 좌측과 우측의 모든 형제 노드를 각각 테이블 블록 (나)와 (다)로 묶고, 이후 C를 포함하는 테이블 블록을 만든다.

이와 같은 테이블 블록화 과정을 통하여 Component들은 하나의 표현 단위로 추출되고 이 단위가 Component Block이 된다.

3.2.3 Component Block의 분류 및 인덱스 생성

Component Block은 자신이 포함하는 컨텐츠의 패턴에 따라서 INDEX형과 BODY형으로 분류된다. INDEX형은 새로운 HTML의 상단에 선택 리스트 형식의 인터페이스로 재표현 되어지고, BODY형은 <BODY> 안에 그대로 재배치되어진다. 이러한 분류 과정은 아래의 <표 3>과 같은 패턴 비교 변수를 사용하여 임의의 경계값과의 비교를 통하여 수행된다. (그림 8)은 인덱스 생성 결과 예제 코드를 보인다.

〈표 3〉 패턴 비교 변수

변 수	기대되는 패턴
TextLength	유사 반복, 제한된 짧은 길이를 가짐.
ImageWidth	유사 반복, 제한된 width 값을 가짐.
LinkNumber	거의 모든 컨텐츠가 하이퍼 링크 정보를 가짐. 연결된 문서의 위치, 파일명에 유사성이 있음.
RowNum	작은 수로 제한됨. 웹 문서에서 상단에 배치된 블록으로 제한됨.
ColNum	최대 또는 최소값으로 제한됨. 좌측 또는 우측에 배치된 블록으로 제한됨.

```
<!-- index start -->
<script language = "JavaScript">
<!--
function change ( form ) {
    var list = form.selectedIndex ;
    if ( form.options [ list ].Value != "0" ) self.location.href = form.options [ list ].value ;
    form.selectedIndex = 0 ;
}
//-->
</script>
<hr>
<form name = "cts_form" method = "get">
<select name = "cts_select1" onchange = "change ( document.cts_form.cts_select1 )">
    <option value = "0" selected> 인덱스 1 </option>
    <option value = "http://go.daum.net/bin/go.cgi?relative = 1&amp ; url =
                /Mail-bin/login_f.cgi%2Ferror%3Dlogin"> 한메일넷 </option>
    <option value = "http://cafe.daum.net"> 카페 </option>
    <option value = "http://shop.daum.net"> 쇼핑 </option>
    <option value = "http://search.daum.net/?_top_4color&amp ; search"> 검색
                </option>
</select>

<select name = "cts_select2" onchange = "change ( document.cts_form.cts_select2 )">
    <option value = "0" selected> 인덱스 2 </option>
    <option value = "http://messenger.daum.net/?_top_head&amp ; msg"> 메신저
                </option>
    <option value = "http://mobile.daum.net/?_top_head&amp ; mobile"> 무선인터넷
                </option>
    <option value = "http://finance.daum.net/?_top_head&amp ; fin"> 금융플라자
                </option>
    <option value = "http://edu.daum.net/?_top_head&amp ; edu"> 교육 </option>
    <option value = "http://miznet.daum.net/?_top_head&amp ; women"> 여성
                </option>
    <option value = "http://job.daum.net/index.html?_top_head&amp ; 1"> 취업
                </option>
    <option value = "http://music.daum.net/?_top_head&amp ; 2"> 뮤직 </option>
    <option value = "http://movie.daum.net/?_top_head&amp ; 3"> 영화 </option>
    <option value = "http://4989.daum.net/?_top_head&amp ; 4"> 사고팔고 </option>
    <option value = "http://fortune.daum.net/?_top_head&amp ; 5"> 운세 </option>
    <option value = "http://avatarmall.daum.net/?_top_head&amp ; 6"> 아바타몰
                </option>
    <option value = "http://www.i-soccer.co.kr/?_top_head&amp ; 7"> 월드컵
                </option>
    <option value = "http://daum.net/doc/sitemap.html?_top_head&amp ; sitemap">
                전체보기 </option>
</select>
</form>
<hr>
<!-- index end -->
```

(그림 8) 인덱스 생성 결과 예제 코드

INDEX형으로 결정된 블록은 컨텐츠의 데이터 타입에 따라서 이미지와 텍스트로 구분 짓고 각각을 INDEX_I형과 INDEX_T형으로 세분한다.

INDEX_I형의 경우는 스크립트 파일을 생성하여 이미지를 인덱스로 표현하고, INDEX_T형의 경우는 <SELECT>를 생성하여 각 인덱스 값이 <OPTION>의 value 속성값으로 매핑되도록 구현한다. 이 때 사용되는 인덱스 데이터는 부분 텍스트로 잘려져서 전체 인덱스의 width가 제한된 길이를 넘지않도록 한다. 기존의 이미지나 텍스트의 하이퍼링크 정보는 스크립트 파일과 추가적인 태그의 데이터 및 속성값으로 유지되게 한다.

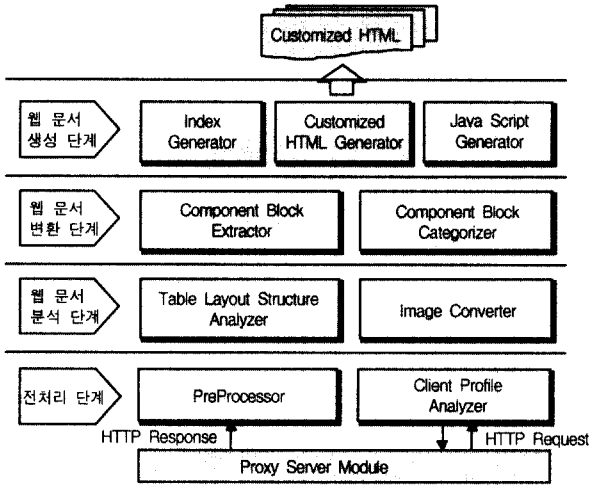
이와 같은 Component Block의 INDEX형 또는 BODY형으로의 분류 및 인덱스 생성 과정은 웹 문서의 브라우징에서 보다 편리한 인터페이스를 제공하고 제한된 화면 내에서 많은 정보를 일괄되게 전달하는 기능을 수행한다. 하지만 일부 웹 문서의 경우, 특히 검색 및 포탈 사이트의 경우는, 한 페이지내의 거의 모든 데이터 부분이 인덱스로 분리

되기도 한다. 따라서 <표 3>의 패턴 비교 변수의 경계값을 보다 제한적으로 사용하거나 인덱스 블록의 개수를 한정하여 패턴 비교 결과 값의 상대적인 우선 순위에 따라서 INDEX 형으로 분류한다.

이러한 새로운 인덱스의 생성을 통하여 기존의 페이지 조각 나눔과 계층 구조의 목차 페이지 생성을 통한 인터페이스보다 편리하고 쉽게 페이지 이동 및 검색이 가능해지고, 원 문서가 가지는 하이퍼링크의 의미를 보다 정확하게 표현할 수 있게 된다.

3.3 전체 시스템의 구조 설계 및 프로토타입의 구현

이상에서 살펴본 웹 문서의 변환 기법을 수행하기 위해 (그림 9)와 같은 구조의 시스템을 설계한다. 이 시스템은 동작 과정에 따라서 전처리 단계, 웹 문서 분석단계, 웹 문서 변환단계, 웹 문서 생성단계로 구분된다.



(그림 9) 소형 화면 단말기를 위한 웹 문서 변환 시스템의 모듈 구성 개념도

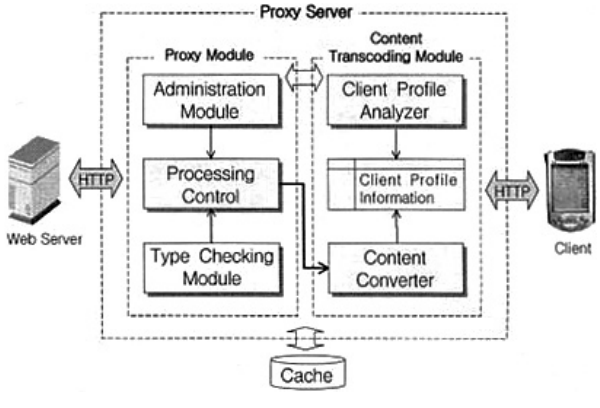
첫 번째의 전처리 단계에서는 외부 모듈과의 입출력 관리 및 원본 HTML 문서의 정제 과정을 수행한다. Client Profile Analyzer 모듈은 HTTP 헤더에서 사용자 프로파일 정보를 추출하고, 캐쉬 저장 여부에 따라 필요한 웹 문서를 웹 서버에게 요청한다. 응답 받은 원본 HTML 문서는 PreProcessor 모듈에서 파싱하여 well-formed HTML 문서로 수정하고 이후의 분석 단계를 위한 HTML Document Object Model(DOM) 트리의 데이터 구조를 생성한다. PreProcessor 모듈은 W3C HTML Tidy Project의 공개 소스를 활용하고 한글문제, 불분명한 태그 사용에 대한 민감한 오류 문제 등을 보완하여 구현하였다.

두 번째로 웹 문서 분석단계에서는 전처리 단계로부터 정제된 HTML 태그 집합을 DOM 트리 형식으로 입력받고, 3.2.1의 Layout-Forming Tag Analysis Algorithm을 통하여 Component 단위를 설정하고 페이지 레이아웃에 대한 semantic 정보 등을 구하는 구조적인 분석을 수행한다. 이때 클라이언트 성능 정보에 따라 이미지 변환 모듈도 함께 동작한다. 이미지 변환 부분은 본 논문이 다루는 범위 밖의 연구 분야이므로 공개된 소스로부터 이미지 포맷의 변환, 색상값 및 크기의 축소 부분을 발췌하여 본 시스템과 연동하도록 설계되었다.

세 번째로 웹 문서 변환단계의 Component Block Extractor 모듈에서는 정의된 Component들을 사용자 단말의 화면 크기와 웹 문서의 구조 정보를 이용하여 MAX_WIDTH를 초과하지 않는 범위 내에서 유사한 조각들로 그룹화하여 Component Block을 생성한다. 3.2.2의 Component Grouping Algorithm이 상위 모듈에서 동작하게 되고, 이후 Component Block에 대한 INDEX형 또는 BODY형의 분류 과정이 Component Block Categorizer 모듈에서 수행된다.

마지막으로 웹 문서 생성단계에서는 인덱스로 분류된 Component Block으로부터 이미지 또는 텍스트 인덱스 정보를 추출하고, 이를 표현하기 위한 스크립트 파일 및 추가 태그 집합을 생성한다. Customized HTML Generator 모듈에서 전 단계를 통하여 생성된 객체 요소들을 문서 양식에 따라 적절히 재구성하여 웹 문서를 생성하게 된다.

본 논문에서 제시하는 변환기는 앞서 언급한 바와 같이 그 동작 환경을 프락시 서버에 두고 있다. 다음의 (그림 10)은 프락시 서버에 추가되는 변환 모듈과, 이 변환 모듈이 프락시 서버와 함께 동작하는 과정을 보인다. 서버와 클라이언트 간의 HTTP 프로토콜에 따른 HTML 문서의 전송은 프락시의 기본 기능을 활용하고, 여기에 변환을 위한 중간 모듈로써, Content Converter 모듈을 추가하며, 필요한 경우 별도의 캐쉬와 사용자 프로파일 정보 관리를 위한 모듈을 추가한다. 프락시 서버는 W3C의 Jigsaw를 구현에 활용하여, 본 논문에서 제시하는 웹 문서 분석 및 변환 알고리즘의 구현 모듈과 연동하여 동작하게 하고, 주요 변환 모듈은 HTML4.0, HTTP1.1을 따르도록 웹 문서 변환 시스템의 프로토타입을 구현하였다.



(그림 10) 웹 문서 변환기와 프락시 서버 모듈간의 연동 구성도

3.4 알고리즘의 테스트 및 성능 평가

시스템의 구현 및 테스트는 알고리즘의 실현과 평가에 초점을 두었고, 웹사이트 분석 및 평가 전문 사이트들에서 제시한 순위별 50개의 사이트를 선정하여 테스트하였다. 다음 <표 4>는 웹 문서 변환 알고리즘의 테스트 결과에 대한 요약을 나타낸다.

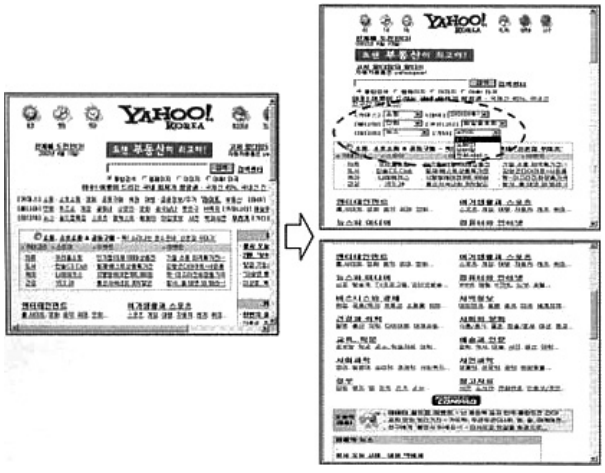
〈표 4〉 알고리즘의 테스트 결과 요약

비교 변수	백 분 율
좌우스크롤의 발생율	0% (해당 페이지 수/전체 테스트페이지 수)
문서 내용의 유지율	91% (해당 Component 수/전체 Component 수)
문서 구조상의 오류 발생율	14% (해당 테이블 블록 수/전체 테이블 블록 수)
에러 발생율	9% (해당 페이지 수/전체 페이지 수)

문서 내용의 유지율은 원본 페이지와 변환된 페이지가 가지는 컨텐츠의 차이에 대한 비율을 나타내는 값으로 Component 단위에서의 컨텐츠 유무를 비교 기준으로 하였다. 문서 구조상의 오류 발생율은 새로운 구조적 표현 즉 <TABLE>의 시각적인 표현이 적절하지 못한 경우를 나타내는 것으로 내부 <TR>과 <TD>의 속성값의 일관성 여부를 비교하였다. 에러 발생율은 스크립트 에러 및 객체 랜더링에서 발생하는 에러의 횟수로 계산한 값이다.

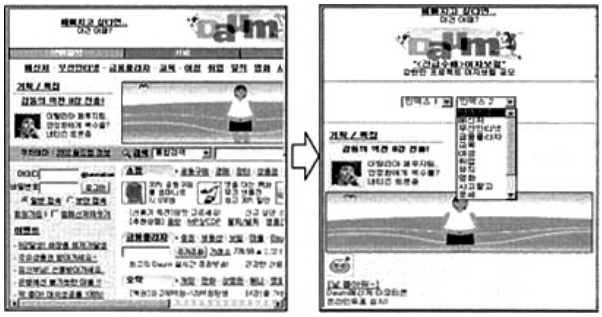
결과적으로, 구조적 구성을 가지는 웹 문서에 대해 최적으로 동작하여 좌우 스크롤없이 내용 단위별 재표현 및 재배치가 이루어졌으며, 특히 많은 정보를 표현하면서 복잡하게 구성된 페이지에 대해 적절한 변환 결과를 보였다.

문서 구조상의 오류는 새로운 Component Block 단위가 명확하게 구분되지 않고 부적절한 태그가 사용된 경우에 새로운 테이블 블록을 생성하는 과정에서 나타났다. 또한 colspan, rowspan 속성값의 처리과정과 Width 설정 과정에서 나타나는 오류로 분석된다. 그리고 부적절한 <FORM>이 사용된 웹 페이지에 대해서 스크립트 에러가 발생하는 경우도 있었다. 이는 PreProcessor 모듈의 동작에서 규칙에 어긋난 태그 사용을 완벽하게 수정할 수 없는 한계 때문에 나타나는 결과로 볼 수 있다.



(그림 11) Yahoo Korea 사이트를 변환한 결과 (MAX_WIDTH = 400)

(그림 11)~(그림 13)은 웹사이트 분석 및 평가 전문 기관에서 제시한 순위표에서 개별 순위 점수를 부여하였을 때, 합계 점수가 가장 높은 순서대로 상위 5개의 사이트에 대한 대표 변환 결과 페이지를 보인다.



(그림 12) Daum 사이트를 변환한 결과(MAX_WIDTH = 400)

(그림 13) DreamWiz, Hanmir, Naver 사이트(좌-우)의 변환 결과 (MAX_WIDTH = 500)

4. 결론 및 향후 연구

본 논문에서는 HTML 브라우저를 탑재한 소형 화면의 단말을 가진 사용자가 무선 인터넷에 접속하여 HTTP를 통한 웹 서비스를 이용하고자 할 경우, 기존의 일반 데스크탑 PC의 디스플레이 성능에 적합하도록 작성된 HTML 문서를 핸드헬드 계열의 소형 화면에서도 효율적으로 표현되어질 수 있도록 변환해 주기 위한 기법을 제시하였고, 이를 프락시 서버를 기반으로 하는 변환 시스템으로 설계하였다.

웹 문서의 분석 및 변환을 위한 주요 알고리즘으로 Layout-Forming Tag Analysis Algorithm과 Component Grouping Algorithm을 소개하였고, Component Block의 분류 및 인덱스 생성과 블록 단위의 재배치 방법에 대해 제안하였다.

본 논문이 제시하는 변환 기법을 통하여 기존의 웹 문서

는 이동 단말의 화면 크기에 적합한 모습으로 변환되어 좌우 스크롤없이 브라우징할 수 있고, 문서 구조에 대한 semi-semantic 정보와 Component 단위의 재배치를 통하여 변환된 결과 페이지에서도 원 문서가 가지는 정보를 명확하게 전달하는 효과를 얻을 수 있다. 이는 이동 환경의 고성능 소형 단말의 특성과 복잡한 구조에서 많은 정보를 한꺼번에 표현하는 현재의 웹 문서의 특징을 고려하여 사용자에게 편리하고 효과적으로 웹 서비스를 사용할 수 있도록 한다.

프락시 서버로 동작하는 변환 시스템의 설계와 프로토타입의 구현을 통하여 제시한 알고리즘의 성능을 평가하였고, 테스트 결과 페이지를 함께 보였다.

향후 과제로 이동 단말을 사용하여 웹에 접속하는 사용자에게 보다 편리한 인터페이스를 제공하기 위해 텍스트 위주의 웹 문서 일부를 청각적인 표현이 가능한 형식으로 변환하는 기법에 대한 연구를 계속하고자 한다.

즉 BODY 형으로 분류된 Component Block 중에서 텍스트 위주로 구성된 블록에 대해 VoiceXML 문서로 변환하는 등의 기법으로 HTML과 다수의 청각적 표현 문서를 함께 클라이언트에게 제공할 수 있을 것이다. 따라서 내용 블록 단위의 재배치와 인덱스의 생성으로 좌우 스크롤없는 브라우징을 제공하면서 텍스트 위주의 BODY형 Component Block에 대해서는 시/청각을 통한 웹 정보의 표현을 제공함을 목표로 관련 연구를 계속 수행하고자 한다.

참 고 문 헌

[1] E. A. Brewer, R. H. Katz, Y. Chawathe, et al. "A Network Architecture for Heterogeneous Mobile Computing," IEEE Personal Communications, Vol.5, No.5, pp.8-24, October, 1998.

[2] 박천교, 이윤철, "이동컴퓨팅 단말 동향", 한국전자통신연구원 주간기술동향, 제1027호, 2001.

[3] 배찬권, "정보통신산업동향 정보통신기기편 제7절 PDA", 정보통신정책연구원. 2001.

[4] T. Bickmore, A. Girgensohn and J. W. Sullivan, "Web Page Filtering and Re-Authoring for Mobile Users," The Computer Journal, Vol.42, No.6, pp.534-546, 1999.

[5] T. Bickmore and W. Schilit, "Digestor : Device-Independent Access to the World Wide Web," Computer Networks and ISDN Systems, Vol.29, No.8, pp.1075-1082, 1997.

[6] Y. H. Whang, C. H. Jung, J. H. Kim and S. K. Chung, "Web-Alchemist : A Web Trtanscoding System for Mobile Web Access in Handheld Devices," SPIE's International Symposium on The Convergence of Information Technologies and Communications (ITCOM 2001), Aug., 2001.

[7] Y. D. Yang and H. J. Zhang, "HTML Page Analysis Based on Visual Clues," IEEE International Conference on Document Analysis and Recognition (ICDAR 2001), pp.859-864, September, 2001.

[8] J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha and A. Crespo, "Extracting Semistructured Information from the Web," ACM PODS/SIGMOD'97, May, 1997.

[9] M. Hori, G. Kondoh, K. Ono, S. Hirose and S. Singhal, "Annotation-Based Web Content Transcoding," 9th World Wide Web Conference, 2000.

[10] D. W. Embley, Y. Jiang and Y. K. Ng, "Record-Boundary Discovery in Web Documents," ACM SIGMOD International Conference on Management of Data (SIGMOD'99), pp.467-478, May, 1999.

[11] B. Bederson and J. Hollan, "Pad++ : A Zooming Graphical Interface for Exploring Alternate Interface Physics," ACM User Interface Software and Technology, pp.17-26, 1994.

[12] E. Brewer, A. Fox, I. Goldberg, D. Lee and A. Polito, "Experience with Top Gun Wingman : A Proxy-Based Graphical Web Browser for the 3Com PalmPilot," IFIP Middleware'98, pp.407-424, 1998.

[13] N. Milic-Frayling and R. Sommerer, "SmartView : Flexible Viewing of Web Page Contents," World Wide Web Conference 2002, 2002.

[14] H. Bharadvaj, A. Joshi and S. Auephanwiriyakul, "An Active Transcoding Proxy to Support Mobile Web Access," IEEE Symposium on Reliable Distributed Systems, 1998.

[15] B. Zenel, "A General Purpose Proxy Filitering Mechanism Applied to the Mobile Environment," Wireless Networks Journal, Vol.5, pp.391-409, 1999.

[16] A. Joshi, "On Proxy Agents, Mobility, and Web Access," Mobile Networks and Applications Journal, Vol.5, pp.233-241, 2000.

[17] 최훈일, 장영건, "HTMLtoVoiceXML 변환기의 설계 및 구현", 정보과학회논문지 : 컴퓨팅의 실제, 제7권 제6호, pp.559-568, 2001.

[18] H. Takagi, C. Asakawa, "Transcoding Proxy for Nonvisual Web Access," ACM ASSETS'00, pp.172-171, November, 2000.

[19] IBM, WebSphere Transcoding Publisher, http://www-3.ibm.com/software/webservers/transcoding/index.html.

[20] OpenTV, SpyGlass Prism, http://www.opentv.com/support/ed_services/spyglass_prism.html.

[21] Argo, WAP Tool, http://www.argogroup.com/waptool/.

[22] World Wide Web Consortium, http://www.w3c.org/.

[23] W3C, HTML Tidy, http://www.w3c.org/People/Raggett/tidy/.

[24] W3C, Jigsaw, http://www.w3c.org/Jigsaw/.

[25] 한국인터넷정보센터(KRNIC), http://www.nic.or.kr/.

신 희 숙

e-mail : hsshin8@etri.re.kr

1998년 경북대학교 컴퓨터공학과 졸업 (학사)

2000년 포항공대 대학원 컴퓨터공학과 졸업(석사)

2000년~2001년 (주)데이콤 종합연구소 연구원

2001년~현재 한국전자통신연구원 컴퓨터소프트웨어기술연구소 연구원

관심분야 : 웹 문서 변환, 마크업 언어, 무선 인터넷 응용 기술 등

마 평 수

e-mail : pmah@etri.re.kr

1985년 서울대학교 식물병리학과 졸업 (학사)

1992년 City University of New York, USA 전산학과(석사)

1995년 Wright State University, USA 전산학과(박사)

1985년~1989년 시스템공학연구소 연구원

1989년~1990년 (주)태양금속 정보산업연구실 대리

1996년~현재 한국전자통신연구원 컴퓨터소프트웨어기술연구소 책임연구원

관심분야 : 멀티미디어 저장서버, 스트리밍 기술, 멀티미디어 검색 및 재생 기술, 웹 컨텐츠 기술 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 컴퓨터소프트웨어기술연구소 선임연구원

관심분야 : 웹 클라이언트 기술, 웹 이미지 분석, 웹 마이닝 등

이 동 우

e-mail : hermes@etri.re.kr

1995년 경북대학교 전자공학과 졸업(학사)

1997년 경북대학교 전자공학과 졸업(석사)

1997년~2001년 현대전자 주임연구원

2001년~현재 한국전자통신연구원 컴퓨터소프트웨어기술연구소 연구원

관심분야 : 지능 정보 단말, 멀티 모달 브라우저 등