

# 효과적인 웹 콘텐츠 개발과 관리를 통합한 시스템의 설계 및 구현

유대승, 강만모, 심민석, 이명재  
울산대학교 컴퓨터·정보통신공학부

## Design and Implementation of An Integrated System for Effective Development and Management of Web Contents

Daesung Yoo, Manmo Kang, Minsuck Sim, Myungjae Lee  
School of Computer Engineering and Information Technology, Univ. of Ulsan.

### 요 약

본 논문에서는 일반문서 작성 에디터와 동일한 방식으로 웹 문서 작성이 가능한 일반문서와 웹 문서 동시작성 에디터의 설계 및 구현을 통하여 효율적인 웹 문서 작성에 대한 연구를 수행한다. 문서의 파싱한 결과를 데이터베이스에 저장함으로써 웹 문서를 유지하지 않고도 데이터베이스에 저장된 정보를 웹 어플리케이션이 필요에 따라 동적으로 문서를 생성함으로써 웹 문서 생성과 유지 및 관리를 통합하는 방법을 제시한다. 마지막으로 ASP 컴포넌트의 개념과 컴포넌트 사용의 필요성 및 이점에 대하여 재고하고 본 연구의 수행과정에서 개발한 ASP 컴포넌트(dll)를 통한 효율적인 웹 어플리케이션 및 컴포넌트 개발 방법을 소개한다.

## 1. 서 론

인터넷의 발전과 함께 기존 문서를 웹 문서로 변환하는 작업이 빈번히 이루어지고 있으며 또한 새롭게 작성되는 문서 중 웹 문서로 작성되어야 하는 필요성이 증대되고 있다. 최근에는 웹 문서 작성을 위해 HTML을 직접 작성하지 않고 일반문서를 작성하는 것과 같은 방법으로 쉽게 웹 문서를 작성할 수 있는 저작도구들이 많이 있다. 그러나 기존의 저작도구들은 순수한 웹 문서 작성을 위해서 개발된 것이 아니다.

따라서 다음과 같은 문제가 있다.

- 일반 문서 작성용 에디터와는 다른 사용자 인터페이스를 제공
- 다양한 글꼴, 크기, 색깔 사용의 제한
- HTML을 이용한 웹 문서 작성에 비해 추가적인 태그삽입
- 문서 작성시에 보는 것과 실제 브라우저 상에서 보는 것이 상이
- 웹 문서가 아닌 일반문서로의 저장 불가
- 웹 문서 작성 후 문서의 유지 및 관리의 기능이 없음

본 논문은 순수 웹 문서 작성을 위한 기존의 범용 HTML문서 저작도구들의 단점을 보완한다. 일반문서와 웹 문서의 동시작성을 제공하여 문서 작성과 유지 및 관리의 기능을 통합함으로써 저작 및 유지관리의 효율적인 방법을 제안한다. 그리고 현대의 다양한 웹 콘텐츠의 출현에 따른 특수목적 HTML 저작도구의 필요성을 재고하는 것을 목적으로 한다.

또한 본 논문에서는 ASP 컴포넌트를 정의하고 이를 사용해야 하는 필요성 및 이점을 정리한다. 본 연구에서 효율적인 컴포넌트 작성 방법을 제안하기 위하여 ASP 컴포넌트 작성의 기본 틀을 제공하는 프레임 컴포넌트(frame.dll)를 개발하였다. 본 시스템의 뷰어 모듈을 구성하

\*본 논문에서 사용되는 용어 중 웹 어플리케이션은 ASP 어플리케이션용, 컴포넌트는 ASP 컴포넌트를 지칭하며 ASP란 웹맥상 대부분 생략한다.

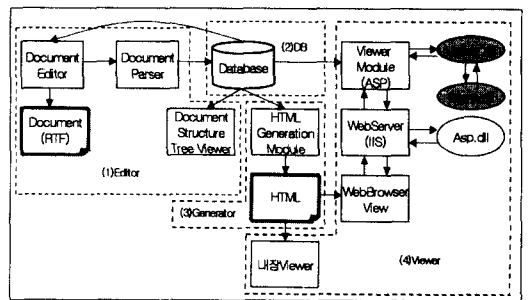
는 뷰어 컴포넌트(viewer.dll)는 본 연구에서 개발된 프레임 컴포넌트를 이용하여 개발되었다. 이를 통하여 프레임 컴포넌트가 컴포넌트 작성에 있어 효율적인 방법을 제공한다는 것을 검증하였다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구를 위해 개발된 시스템의 전반적인 구조와 각 모듈의 역할에 대해 설명한다. 3장에서는 ASP 컴포넌트 개발을 통한 웹 어플리케이션 개발의 필요성 및 장점을 살펴본다. 그리고 본 시스템에서 개발한 뷰어 컴포넌트와 프레임 컴포넌트의 구조 및 기능을 설명한다. 끝으로 4장에서 결론 및 향후 연구방향에 대해 살펴본다.

## 2. 구조 및 모듈의 역할

본 연구에서 개발된 시스템은 일반문서와 웹 문서의 동시 작성을 가능하게 하고 데이터베이스를 통한 효율적인 유지 및 관리 방법을 제안하기 위해 개발되었다.

본 시스템의 구조는 [그림 1]과 같으며 이장에서는 구조 및 모듈의 역할에 대하여 살펴본다.

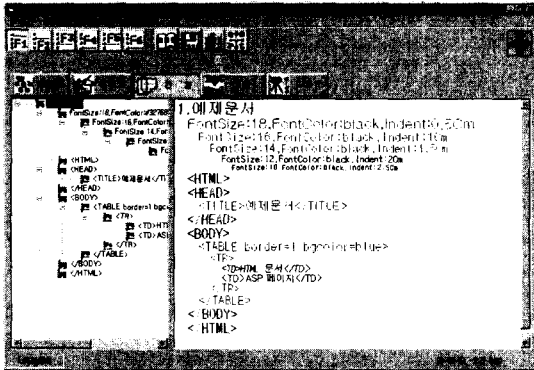


[그림 1] 시스템 구조도

## 2.1 시스템 개요

개발된 시스템은 에디터, 데이터베이스, 제너레이터, 뷰어로 나누어진다. 에디터에서 편집한 내용을 일반문서(rtf 또는 txt 포맷)로 저장하거나 문서 파서를 통해서 파싱된 결과를 데이터베이스에 저장한다. 데이터베이스에 저장된 결과를 기반으로 [그림 2]와 같이 트리 구조로 보여준다. 데이터베이스에 저장된 데이터는 제너레이터와 뷰어에서 사용한다. 제너레이터는 데이터베이스에 저장된 데이터를 이용하여 HTML문서를 생성하는 기능을 수행한다. 뷰어는 HTML문서를 기반으로 하는 뷰어와 데이터베이스의 데이터를 기반으로 웹 어플리케이션을 통한 뷰어로 구분된다. HTML 문서 이용 뷰는 내장 웹브라우저컨트롤을 이용하거나 시스템 기본 웹브라우저를 이용할 수 있다. 웹 어플리케이션 뷰는 웹 브라우저의 웹 서버에 대한 문서 요청에 의해 뷰어 모듈이 데이터베이스의 데이터를 이용해서 동적으로 문서를 생성하여 웹 브라우저에 보내지게 되고 이것을 웹 브라우저가 보여주게 된다.

[그림 2]는 본 연구에서 개발된 시스템의 실행 화면이다.



[그림 2] 메인 실행화면

## 2.2 에디터

에디터는 RTF문서 기반 편집기로 일반문서 편집방법의 편집기능을 제공하고 편집된 내용을 일반문서로 저장하거나 문서를 파싱한 결과를 데이터베이스에 저장한다. 데이터베이스에 저장된 데이터는 문서의 구조 파악을 쉽게 할 수 있도록 트리 구조 형태로 보여지게 된다. 그리고 데이터베이스의 데이터는 HTML문서 생성을 위한 제너레이터의 소스데이터로 사용되거나 문서 요청시 동적으로 문서를 생성하는 뷰어 모듈의 소스데이터로도 사용된다.

에디터로 작성된 문서는 일반문서, HTML문서, 데이터베이스 형태로 저장 가능한다. 한번의 에디팅으로 필요에 따라서 일반문서와 HTML문서의 동시 생성이 가능하다. 그리고 데이터베이스에 저장된 데이터를 이용해서 동적으로 웹 문서를 생성을 할 수 있다. 그러므로 여러 웹 문서를 유지해야 하는 시스템에서는 여러 웹 문서를 유지할 필요 없이 데이터베이스에 통합함으로써 효과적으로 문서를 유지 및 관리 할 수 있다.

## 2.3 데이터베이스

에디터에서 파싱한 결과는 6레벨의 계층구조로 나타나며 이를 저장하기 위한 6개의 테이블과 각 데이터의 글꼴정보(이름, 크기, 색깔, 들여쓰기)를 저장하기 위한 테이블로 구성된다. 데이터베이스에 저장된 데이터는 제너레이터와 뷰

어의 소스 데이터로 사용된다.

## 2.4 제너레이터

제너레이터는 에디팅한 문서의 파싱에 의해 저장된 데이터를 기반으로 HTML을 생성하는 기능을 수행한다. HTML태그를 생성하는 7가지의 기초 모듈을 이용하며 이 기초 모듈들은 웹 어플리케이션 뷰어에서 사용되는 프레임 컴포넌트를 제작하기 위한 기초모듈로도 사용된다.

기초모듈들은 3장의 컴포넌트 개발을 통한 웹 어플리케이션 개발에서 살펴보도록 하고 다음코드는 기초모듈들을 사용한 제너레이터 구성의 일부분이다.

```

HTMLbegin '<HTML>'
HEADbegin '<HEAD>'
STYLEbegin '<STYLE>'
  StyleText = vbTab + "BODY (background:#" +
    + Cstr(FoncColor(6)) + ")" + vbCrLf
  For i = 0 To 5
    StyleText = StyleText + vbTab + "H" + CStr(i + 1) +
    + vbCrLf
  Next
  WriteDoc StyleText
STYLEend '</STYLE>'
HEADend '</HEAD>'
BODYbegin '<BODY>'
  retVal = OutPutDocument(Code)
BODYend '</BODY>'
HTMLEnd '</HTML>'
    
```

## 2.5 뷰어

뷰어는 제너레이터에 의해서 생성된 HTML문서를 보여주는 뷰어와 데이터베이스에 저장된 데이터를 동적으로 생성해서 보여주는 뷰어로 구성된다. 전자는 내장 웹브라우저 컨트롤을 이용하거나 시스템의 기본 웹브라우저를 이용하는 방법이 있다. 후자는 웹서버가 동적 HTML 생성 모듈을 통해서 생성된 문서를 웹브라우저에 전송하는 방식이며 이 모듈은 마이크로소프트 트랜잭션 서버의 오브젝트 컨텍스트의 리스폰스 객체를 가져와서 직접 출력하는 프레임 컴포넌트를 이용한다. 이 컴포넌트는 제너레이터 모듈을 ActiveX DLL 컴포넌트로 변환한 것이다.

아래의 코드는 웹 어플리케이션 뷰어 모듈 전체 코드이다.

```

'뷰어 컴포넌트 생성
set Doc = server.CreateObject("viewer.ASP")
Code = Request.QueryString("Code")
'출력코드와 DSN을 인자로 viewer.dll의 GenDocument 호출
call Doc.GenDocument(cstr(Code),"Web")
    
```

위의 코드와 같이 컴포넌트 객체를 생성한 후 생성할 문서의 코드와 DSN(Data Source Name)을 인자로 하여 동적으로 HTML문서를 생성하는 함수(뷰어 컴포넌트의 인터페이스)를 호출한다. 위와 같이 실제 뷰어 모듈은 컴포넌트 내에 숨길 수 있고 수행속도 향상을 기대할 수 있다.

뷰어 컴포넌트는 내부적으로 프레임 컴포넌트를 참조해서 작성되었다. 이것에 대한 자세한 사항은 3.3절의 뷰어 컴포넌트와 프레임 컴포넌트의 구조 및 기능에서 자세히 설명한다.

## 3. 컴포넌트 개발을 통한 웹 어플리케이션 개발

이번 장에서는 ASP 컴포넌트에 대하여 간단하게 정리하

고 컴포넌트 사용상의 이점과 사용해야 하는 이유를 설명한다. 그리고 본 시스템에서 개발한 뷰어 컴포넌트와 프레임 컴포넌트의 구조 및 기능을 살펴본다.

### 3.1 ASP 컴포넌트

ASP 컴포넌트는 COM 기반의 컴포넌트로 특정 기능을 캡슐화하여 ASP 페이지에서 직접 또는 다른 ASP 컴포넌트를 통해 간접적으로 사용된다.[2] ASP 컴포넌트는 기본적으로 인프로세스(ActiveX DLL)로 액세스 하도록 하며 아웃프로세스 컴포넌트를 사용하려면 IIS의 설정을 변경해야 한다. 또한 ASP 컴포넌트는 웹 서버와 같이 여러 사용자에 의해 동시에 이용되므로 다중 스레드 모델을 사용해야 한다.[1][4]

### 3.2 ASP 컴포넌트 사용의 이점 및 사용 이유

컴포넌트 사용의 이점 및 사용 이유를 정리하면 다음과 같다.[1][2]

- 1) 재사용성을 증가시킨다.
- 2) ASP 페이지는 소스코드 자체가 배포되는 반면 컴포넌트는 컴파일되어 배포되기 때문에 비즈니스 목적에 맞다.
- 3) 시스템의 모듈별 개발 및 분산이 용이하다.(DCOM)[3]
- 4) win32 API를 직접 호출하지 못하는 것과 같은 스크립트 언어의 한계를 극복하고 기능을 확장한다.
- 5) 스크립트 방식의 실행에 비해 수행속도를 증가시킨다.(컴포넌트는 컴파일된 머신코드)

### 3.3 뷰어 컴포넌트와 프레임 컴포넌트의 구조 및 기능

이번 장에서는 본 시스템에서 개발한 뷰어 컴포넌트와 프레임 컴포넌트의 구조 및 기능을 소개한다.

뷰어 컴포넌트는 웹 어플리케이션에서 생성할 문서의 코드와 DSN(Data Source Name)을 전달받아 데이터베이스에 저장된 데이터를 기초로 HTML 문서를 생성하는 컴포넌트이다. 다음은 뷰어 컴포넌트의 일부분이다.

```
Dim DBCon As New ADODB.Connection
Dim Doc As New oose.HTML 'HTML생성 컴포넌트 선언
DBCon.Open DSN
With Doc
    .HTMLbegin '<HTML>'
    .BODYbegin '<BODY>'
        retVal = OutPutDocument(Code)
    .BODYend '</BODY>'
    .HTMLend '</HTML>'
End With
```

위의 코드는 2.4절의 제너레이터와 유사하다. 뷰어 컴포넌트는 제너레이터를 모델로 생성되었다. ASP 컴포넌트 작성시 MTS의 오브젝트 컨텍스트를 가져오고 이것으로 리퀘스트 객체나 리스폰스 객체등의 참조를 얻어서 여기에 매번 읽거나 써야 하는 방법은 코드의 가독성을 떨어뜨리고 개발을 불편하고 어렵게 만든다. 본 연구에서는 이와 같은 문제점을 해결하기 위하여 프레임 컴포넌트를 개발하였고 이것을 사용하면 읽기 쉽고 편리하게 ASP 컴포넌트를 개발할 수 방법을 제공한다.

다음은 기존방식과 제안방식을 비교한 예이다.

<pre>Dim Cont As New ObjectContext Dim Res As Object Set Cont = GetObjectContext() Set Res = Cont("Response") With Res     .Write "&lt;Table border=1 " _         + " bgcolor=Blue&gt;"     .Write "&lt;Tr&gt;"     .Write "&lt;Td width=400&gt;"     .Write "Simple Document"     .Write "&lt;/Td&gt;"     .Write "&lt;/Tr&gt;"     .Write "&lt;/Table&gt;" End With</pre>	<pre>Dim Doc As New oose.HTML With Doc     .TABLEbegin "border=1", _         " bgcolor=#ffd79d"     .TRbegin     .TDbegin "width=400"         .WriteDoc "Simple Document"     .TDend     .TRend     .TABLEend End With</pre>
---	--

[기존방식]

[제안방식]

다음은 프레임 컴포넌트의 일부분이다.

```
Public Doc As String
Public ObjContext As ObjectContext
Public ObjResponse As Object
Private Sub Class_Initialize()
    Set ObjContext = GetObjectContext()
    Set ObjResponse = ObjContext("Response")
End Sub
Public Sub WriteDoc(SourceDoc As String)
    ObjResponse.write SourceDoc
End Sub
Public Sub IHTMLbegin(ParamArray VarArg())
    If UBound(VarArg) >= 0 Then
        Doc = "<HTML "
        For i = 0 To UBound(VarArg)
            Select Case getPName(VarArg(i))
                Case "LANG"
                    Doc = Doc + " " + VarArg(i)
            End Select
        Next i
        WriteDoc Doc
    End Sub
```

## 4. 결론 및 향후 연구과제

본 논문에서는 현대의 다양한 웹 콘텐츠의 출현으로 다양한 요구 사항을 충족시키기 위해서는 특수목적 저작도구의 필요성을 도출하고 웹 문서의 생성과 유지 및 관리를 통합하는 시스템을 제안했다. ASP 컴포넌트의 개념과 컴포넌트 사용의 필요성 및 이점에 대하여 살펴보고 본 연구의 수행과정에서 개발한 ASP 컴포넌트를 이용한 효율적인 ASP 컴포넌트 개발방법을 제안했다.

향후 연구과제로는 웹 서비스를 운영하기 위한 웹 문서와 웹 어플리케이션과 같은 다양한 콘텐츠들의 생성 및 관리를 하나로 통합할 수 있는 방법에 대한 연구를 수행해야 한다.

### [참고문헌]

- [1] Sing Li, Panos Economopoulos, Professional COM Applications with ATL, Wrox Press, 1998
- [2] Shellery Powers, Developing ASP Components, O'Reilly, 1999
- [3] Guy Eddon, Henry Eddon, Inside Distributed Com, Microsoft Press, 1998
- [4] Ted Pattison, Program Distributed Application with COM and Microsoft Visual Basic 6.0, Microsoft Press, 1998
- [5] Clemens Szyperski, Component Software Beyond Object-Oriented Programming, ACM Press, 1998