확장 가능한 아바타 생성 시스템에 관한 연구

조이기[O] 장미화 김원중
순천대학교 컴퓨터과학과
kwj@sunchon.ac.kr

A Study on Scalable Avata Construction System

Lee-Gi Cho[O] Mi-Hwa Jang Won-Jung Kim
Dept. of Computer Science, Sunchon National Univ.

요 약

아바타(Avata)는 가상현실게임, 채팅, 사이버 쇼핑몰, 가상교육 등에서 사용자의 역할을 대신하는 애니메이션 웹 캐릭터의 의미로 사용되고 있다. 과거 네티즌들은 사이버공간의 익명성에 매료되었지만 이제는 자신을 표현하려는 욕구를 강하게 느끼게 되어 이 두 가지를 모두 충족시켜 주는 아바타가 청소년층을 중심으로 큰 인기를 얻고 있다. 그러나 현재 웹에서 제공되고 있는 아바타 생성 시스템들은 이미 완성된 아바타만을 제공하거나, 몇 가지의 부분별 캐릭터를 제공하여 사용자들이 조합하여 사용할 수 있도록 하고 있기 때문에 자신만의 독특한 개성을 표현하고자 하는 네티즌들을 만족시키지 못하고 있다.

따라서 본 논문에서는 XML/SVG를 사용하여 확대/축소가 가능하고, 그래픽 사용자 인터페이스를 통하여 쉽게 자신만의 독특한 아바타를 생성하기 위한 시스템을 제안하였다.

I. 서론

아바타(Avata)는 고대 인도에서 땅으로 내려온 신의 화신(분신)을 나타내는 말이었으나 인터넷 시대가 열리면서 3차원 가상현실게임, 웹에서의 채팅 등에서 사용자의 역할을 대신하는 애니메이션 웹 캐릭터의 의미로 사용되고 있다. 즉, 아바타는 그래픽 위주의 가상사회에서 자신을 대표하는 가상육체라고 할 수 있다.

본 논문에서는 아바타와 웹 캐릭터를 같은 의미의 용어로 사용한다. 현재 아바타가 이용되는 분야는 채팅이나 온라인게임 외에도 사이버 쇼핑몰, 가상교육, 가상오피스 등으로 확대되었다. 아바타는 현실세계와 가상공간을 이어주며, 익명과 실명의 중간정도에 존재한다. 과거 네티즌들은 사이버공간의 익명성에 매료되었지만 이제는 자신을 표현하려는 욕구를 강하게 느끼게 되어 이 두 가지를 모두 충족시켜 주는 아바타가 청소년층을 중심으로 큰 인기를 얻고 있다. 그러나 현재 웹에서 제공되고 있는 아바타 생성 시스템들은 이미 완성된 아바타만을 제공하거나, 몇 가지의 부분별 캐릭터를 제공하여 사용자들이 조합하여 사용할 수 있도록 하고 있기 때문에 자신만의 독특한 개성을 표현하고자 하는 네티즌들을 만족시키지 못하고 있다. 또한, 최근에 확대/축소가 가능한 아바타 생성 시스템을 지원하는 사이트들이 등장하고 있지만 생성된 아바타의 확대/축소시에 이미지 정보의 손실로 많은 한계를 보이고 있다.

이것은 웹에서 먼저 표준으로 자리잡고 있는 그래픽 포맷인 GIF, JPEG 등이 비트맵(Bitmap)방식이어서, 현재 웹 상에서 이루어지는 모든 이미지 처리는 비트맵 방식일 수밖에 없기 때문이다. 그러나 이러한 비트맵 방식은 픽셀에 대한 압축기법으로 한 번 이미지의 크기나 형태를 결정하면 변형이 어렵다는 단점이 있다. 특히 웹 상에서 특정한 변형을 일으킨다거나 사용자가 요구하는 이미지로의 변환이 불가능하기 때문에 아바타 생성 시스템은 얼굴형에서부터 눈썹, 코 등과 같이 얼굴을 구성하고 있는 모든 부분 요소들을 많은 경우의 수를 고려하여 맵(Map)화하여 가지고 있어야 한다[2,4,5]. 이렇게 기성품(Ready-made)화된 부분요소들을 조합하여 웹 캐릭터를 생성하는 시스템은 미리 정의된 이미지를 사용자가 선택만 하게 함으로써 사용자는 자신만의 특성을 살릴 수 있는 웹 캐릭터를 만들기가 불가능하다. 또한 이렇게 선택되어 탄생하는 웹 캐릭터는 다른 사용자의 웹 캐릭터와 동일할 수도 있다는 문제점이 발생한다. 이는 개인화 서비스를 제공하기 위한 웹 캐릭터 구성에 크나큰 제약 사항이 아닐 수 없다.

본 논문에서는 비트맵(Bitmap) 형식의 이미지를 사용할 때의 단점을 극복하기 위한 방안으로 벡터(Vector) 형식의 이미지 중 차세대 마크업 언어로 부각되고 있는 XML의 그래픽 표준인 SVG로 웹 캐릭터 생성 시스템을 구현하였다. 또한 각 사용자의 요구에 맞는 변형이 쉽게 이루어 질 수 있도록 사용자에게 친숙한 인터페이스를 제공하였고, 모든 데이터를 데이터베이스에 저장함으로써 언제든지 사용자가 쉽게 자신의 캐릭터를 재구성하여 사용할 수 있도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서 이미지 표현 방법 등 관련 연구에 대해 소개하고, 3장에서는 아바타 생성 시스템의 구조와 각 모듈의 기능에 대해 설명하였다. 4장에서는 시스템의 구현에 대해 기술하고, 논문의 결론과 향후 연구 과제를 고찰하였다.

본 연구는 광주 · 전남테크노파크 연구개발사업에 의해 수행되었음

II. 관련 연구

2.1 이미지 형식

웹 캐릭터를 표현할 수 있는 이미지 형식은 크게 비트맵(Bitmap) 형식과 벡터(Vector) 형식으로 나눌 수 있다. 현재 웹 상에서는 비트맵 형식의 이미지 형식이 표준으로 자리를 잡고 있기 때문에 웹 캐릭터를 구성하는 이미지는 비트맵 형식을 사용하고 있다. 그러나 비트맵 형식은 다양한 웹 캐릭터를 표현하는데 많은 제약점을 가지고 있기 때문에 향후 벡터 방식을 이용하는 아바타 제공 사이트들이 등장할 것으로 예측된다.

벡터 방식은 선과 면으로 구성되기 때문에 서로 다른 개체들이 섞이거나 혼합되지 않고 확대나 축소시 이미지의 변화가 전혀 없다는 것이고, 비트맵 방식은 픽셀을 이용함으로 사진을 합성하거나 특수효과를 내는데 적합하지만 확대나 축소시에 계단현상 등의 이미지 손상이 일어난다. 또한 같은 이미지일 때, 비트맵을 픽셀 하나하나의 값을 저장해야 하는데 비해 벡터는 수학적 데이터만 저장하면 되기 때문에 저장용량이 현저히 적어진다. 웹 캐릭터를 만들 때, 사용자들이 만들고자 하는 아바타의 모든 부분 요소들을 제공한다는 것은 불가능하다. 이 문제를 해결하기 위해서는 각 부분 요소들의 기본 형태만을 제공하고, 사용자들은 그 기본 형태들을 확대/축소 등 여러 가지 방법으로 변형하여 본인이 원하는 아바타를 만들 수 있도록 하는 것이다. 그러나 [그림 1]에서 볼 수 있는 것처럼 비트맵 방식은 이미지를 변형할 때, 이미지가 손상된다. 따라서 사용자의 욕구를 만족시키기 위한 아바타 생성 시스템의 이미지 표현 방식은 [그림 2]와 같은 벡터 형식이 되어야 한다[3].

[그림 1]비트맵 이미지

[그림 2]벡터 이미지

2.2 그래픽 표준의 종류

그래픽 표준은 툴(Tool)에 따라 다양한 형태로 존재한다. 하지만 웹 상에서 사용하고 있는 비트맵 형식의 대표적인 그래픽 표준은 JPEG(*.JPG)와 GIF89a(*.GIF) 형식이고 벡터 형식의 표준은 Flash(*.SWF), Illerstrator(*.AI), Scalable Vector Graphic(*.SVG)이다. 본 논문에서 아바타 생성 시스템에 사용한 SVG는 2000년 8월에 World Wide Web Consortium(W3C)에 의해 Candidate Recommendation으로 발표되었으며, SVG를 활용하는 기술의 개발이 본격화 될 전망이다.

SVG는 XML을 기반으로 하는 그래픽 표준으로 벡터 형식을 취하고 있으며 Text 형식으로 그래픽 요소를 정의하기 때문에 있어서 수정 및 변환이 용이하다. 또한 Script 지원으로 사용자의 요구를 충족할 수 있는 편집이 가능하고, 애니메이션이나 멀티미디어 형식의 지원이 가능하여 차세대 웹 그래픽 표준으로 각광받고 있다. 웹에서 사용되는 대표적인 그래픽 표준들을 비교하면 [표 1]과 같다.

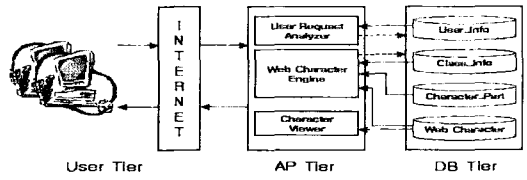
[표 1] SVG와 다른 그래픽 표준 비교 분석표

	SVG	Flash	AI	JPEG	GIF
이미지 형식	Vector	Vector	Vector	Bitmap	Bitmap
애니메이션 유무	O	O	×	×	O
스크립트 유무	O	O	×	×	×
크기변화 유무	O	O	O	×	×
HTML 표준	O	O	×	O	O
XML Based	O	×	×	×	×

III. 웹 캐릭터 생성 시스템

3.1 시스템 구조

본 연구에서 구현한 아바타 생성 시스템은 [그림 3]과 같이 User Tier, Application Tier의 User Request Analyzer, Web Character Engine, Character Viewer와 Database Tier의 User Info Store, Class Info Store, Character Part Store, Web Character Store로 구성되어 있다.



[그림 3] 시스템 구조도

User Tier의 사용자가 브라우저를 통해 사용자 정보를 입력하면 AP Tier의 User Request Analyzer를 통해 새로운 아바타의 생성인지 아니면 기존에 보유하고 있는 캐릭터의 수정인지를 결정한다. 이렇게 결정된 값에 의해 DataBase Tier의 Character Part Store 혹은 Web Character Store에서 데이터를 추출한다.

즉, 새롭게 캐릭터를 생성하는 경우에는 Character Part Store의 각 부분 요소에 대한 해당 기본 데이터를 추출해 온다. 추출한 데이터를 이용하여 Web Character Engine에서 사용자가 캐릭터를 생성하게 되고, 그 결과를 SVG 형식으로 변환 처리한 뒤, User Info Store의 사용자 정보를 이용하여 Web Character Store에 캐릭터 데이터를 저장한다. 이때 사용자에 의해 선택된 색상은 Class Info Store에 저장되어 있는 클래스 데이터에 적용되어 새로운 스타일 시트를 생성한다. 사용자가 기존 캐릭터를 수정하는 경우에는 Web Character Store에서 미리 저장된 데이터를 추출하여 사용자의 수정이 완료된 후 Web Character Store에 저장된다.

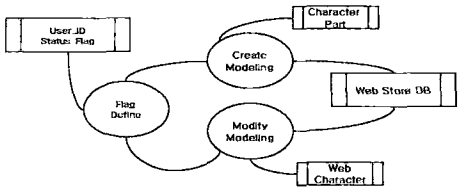
3.2 Application Tier

Application Tier에는 User Request Analyzer, Web Character Engine, Character Viewer가 있다.

① User Request Analyzer

Login한 사용자의 User_ID는 사용자가 종료하기 전까지 Session에 저장되어 Web Character를 구현하고자 할 때 [그림 4]과 같이 프로세스 흐름도를 도식화할 수 있다.

[그림 4]의 User Request Analyzer는 사용자가 입력한 ID와 User Info Store에서 추출한 Status_Flag를 정의하여 새로운 아바타를 만들 것인지 아니면 보유하고 있는 아바타를 수정할 것인지를 결정하여 그에 대한 아바타를 새롭게 정의한다.



[그림 4] User Request Analyzer Diagram

User Request Analyzer는 User_ID와 Status_Flag 값을 사용자와 시스템으로부터 입력받는다. 입력받은 값 중 Status Flag 값이 어떻게 정의되어 있는지 확인하고 확인된 값에 따라 아바타를 생성할 것인지 수정할 것인지 결정한다. 아바타를 생성할 때는 Character Part Store, 수정할 때는 Web Character Store에서 원하는 데이터를 추출, 변환하여야 한다. 이렇게 변환된 데이터는 사용자 정보와 함께 Web Character Store에 저장된다.

② Web Character Engine

User Info Store에서 가져온 사용자 정보와 Character Part Store에서 추출된 캐릭터 구성 요소의 기본형 데이터에 각 변형된 포인터 값을 변경 및 재조합한다. 수정된 각 부분 데이터를 새로운 포인터 값으로 생성한 뒤, SVG 형식의 데이터를 생성하고 Web Character Store에 저장한다. 또한 사용자가 기존에 생성하였던 캐릭터를 수정하는 경우에도 데이터를 Web Character Store에서 가져온다는 것을 제외하고는 비슷한 절차에 따라 수행된다.

③ Web Character Viewer

Web Character Store에서 생성한 데이터를 XML/SVG 형식의 웹 문서로 변환하는 역할을 한다. 이렇게 변환되어진 웹 문서는 사용자가 사용하는 브라우저에서 XML/SVG 형식의 웹 캐릭터를 디스플레이 한다.

Web Character Store에 저장되어 있는 데이터를 XML/SVG 형식으로 변환하는 시스템을 구현하는데 MS SQL 2000에서 제공하는 Template 방식을 사용하였다. MS SQL에서는 템플릿 파일을 XML형식으로 변환하기 위하여 URL Query나 XPath Query, Template Query 등을 지원하고 있으나, 본 연구에서 Template 방식을 채택한 이유는 다음과 같은 장점이 있기 때문이다.

(1) 템플릿 실행결과를 유효한 XML 문서로 만들 수 있다.
(2) SQL문이나 XPath Query에 전달할 수 있는 매개변수를 정의할 수 있다.
(3) Namespace를 선언할 수 있다.
(4) XSL 스타일 시트를 결과 문서에 적용하도록 할 수 있다.
(5) 보안을 향상시킨다.

3.3 DataBase Tier

아바타 시스템을 사용하는 사용자 정보를 제공하는 User Info Store와 각 색상에 대한 정보를 스타일 시트를 적용하여 클래스로 정의한 Class Info Store, 아바타의 각 부분요소에 대한 일련의 데이터를 제공하는 Character Part Store 및 사용자의 요청에 의해 수정된 포인터 값이 저장된 Web Character Store가 존재한다. 특히 Character Part Store와 Web Character Store는 SVG형식을 유지하면서 데이터를 저장하기 때문에 웹 상에서 아바타를 구성할 때에는 데이터를 가공하지 않고 제공되는 데이터 그대로를 사용한다.

IV. 구 현

웹 캐릭터 시스템을 구성하는데 있어 데이터베이스는 XML 포맷을 쉽게 저장할 수 있는 MS SQL Server 2000을 선택하였다. 그에 따라 MS SQL Server 2000에 가장 접근이 용이한 ASP(Active Server Page 3.0)와 Windows 2000 Server를 이용하여 구현하였다. 또한 XML 데이터를 처리하는데 있어서 MS의 MSXML 3.0 Parser를 사용하였고, 사용자가 수정하는 컨트롤러는 ActiveX를 사용하였다. 브라우저는 IE 5.0 이상의 버전에 SVG Viewer Plug-In이 설치되어 있어야 한다.

본 논문에서는 원시 시스템(Prototype System)으로 캐릭터의 얼굴 부분을 대상으로 구현 개발하였고, 부분적으로는 얼굴, 코, 눈, 머리, 입, 귀, 눈썹의 7가지로 나누어 캐릭터를 생성하였다.

4.1 Character Template

[표 2]는 MS SQL 문장으로 얼굴 캐릭터의 부분요소 데이터를 추출하는 부분으로 각각의 기능은 다음과 같다.

[표 2] Character Template의 Query

```
<!--
<FACE>---------------------------------- ①
  <sql:query>---------------------------- ②
        SELECT   D, CLASS ---------------- ③
        FROM     PATH ------------------ ④
        WHERE    USER_ID = @USER_ID ----⑤
       AND      PART    = @PART_1
       FOR      XML     AUTO
  </sql:query>
  </FACE>
 ------------------------------------------>
```

```
<!--
  <NOSE>---------------------------- ⑥
  <sql:query>
        SELECT   D, CLASS
        FROM     PATH
        WHERE    USER_ID = @USER_ID
        AND      PART    = @PART_2
       FOR      XML     AUTO
  </sql:query>
  </NOSE>
 ------------------------------------>
<!--
 중략------------------------------------- ⑦
 ------------------------------------>
</ROOT>
```

① FACE Part 부분을 생성하는 템플릿이다.
② Namespace가 sql로 선언이 되어 있듯이 사용되는 자식 엘리먼트의 명칭이 "sql:"로 시작한다. sql:query 라고 선언되어 있는 이 부분에 SQL 문장을 삽입할 수 있다.
③, ④ Path라는 엘리먼트를 생성하고 좌표값과 색상값인 D와 CLASS라는 속성을 생성하는 구문이다. 쿼리의 FROM절에 있는 테이블 명이 엘리먼트가 되고 추출하는 필드 명이 속성이 된다. 모든 부분에 거의 동일한 형식으로 제시된다.
⑤ Path 테이블에서 사용자가 Login한 ID값과 일치하는지 확인한다.
⑥ NOSE Part 부분을 생성하는 템플릿이다.
⑦ EYE Part, HAIR Part, MOUSE Part, EAR Part, EYEBROWS Part 부분을 생성하는 템플릿이다.

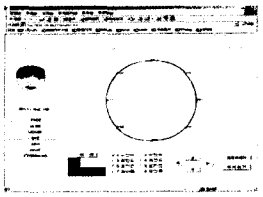
이렇게 정의한 템플릿은 Web Character Store에서 완성된 캐릭터를 호출한다.

4.2 사용자 인터페이스

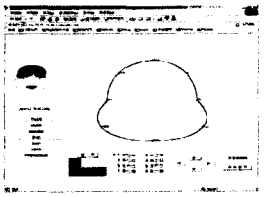
구현 시스템은 로그인과 7가지의 얼굴 부분 요소들을 정의할 수 있는 사용자 인터페이스로 구성되어 있다.

[그림 5]의 화면은 3개의 프레임으로 구성되어 있다. Character Part Store 또는 Web Character Store에서 가져온 데이터들을 결합하여 표현하는 좌측 상단 화면과 캐릭터의 7개 Part인 FACE, NOSE, MOUSE, EYE, EAR, HAIR, EYEBROWS를 선택할 수 있는 좌측 하단 메뉴 화면, Control이 적용되어 가변 포인트 값을 수정하는 메인 화면으로 구분된다. 좌측 상단에 완성된 화면은 Character Part Store에 있는 부분요소 데이터나 Web Character Store에 저장된 데이터를 템플릿을 통해 가져온 후 XML 형식으로 된 캐릭터를 디스플레이한다.

[그림 5]의 화면은 사용자가 부분 요소 중 얼굴과 관련된 데이터를 수정하는 화면을 나타내고 있다.



[그림 5] FACE 변경 전



[그림 6] FACE 변경 후

사용자에게 제공되는 얼굴 모형에는 8개의 가변 포인트가 있다. 이 포인트는 사용자가 얼굴 모형을 변형하는 데 있어서 수정할 수 있는 가장 최적화된 위치의 포인트이다. 얼굴의 가변 포인트는 상, 하, 좌, 우의 네 개와 그 사이의 포인트 값 네 개로 이루어져 총 8개의 가변 포인트가 있다. 사용자는 이렇게 제공되는 8개의 가변 포인트를 방향키를 이용하여 원하는 형태의 얼굴윤곽과 크기를 결정할 수 있다. 그리고 색상 선택을 통하여 얼굴의 색상을 결정하게 된다. [그림 5]의 기본으로 제공되는 화면에서 FACE의 4번 포인트와 6번 포인트를 변경한 화면이 [그림 6]이다. 포인트 값을 변경한 후 좌측 상단의 완성 캐릭터의 "캐릭터 새로고침" 버튼을 클릭하면 변경된 캐릭터가 디스플레이 되고, 하단 우측의 적용하기 버튼을 클릭하면 데이터베이스에 저장된다.

V. 결 론

본 논문에서는 차세대 마크업 언어로 각광받고 있는 XML(Extensible Markup Language)의 그래픽 표준인 SVG(Scalable Vector Graphic)을 이용하여 사용자가 원하는 형태로 이미지의 수정 및 부분 요소의 재사용성을 높인 아바타 생성 시스템을 설계 및 구현하였다. 연구 결과를 이용하면 에이전트 상의 웹 캐릭터 혹은 웹 상에서 일어날 수 있는 어떠한 컨텐츠에도 사용할 수 있는 기술을 구현할 수 있을 것이다.

또한 본 논문에서 구현한 캐릭터 빌더는 여러 맵을 보면서 선택해야 하는 번거로운 인터페이스를 하나의 부분 요소를 직접 수정하게 함으로서 간략화 하였고 원하는 형태의 독자적인 캐릭터를 만들 수 있게 하였다.

향후 연구과제로 본 논문에서 구현한 웹 캐릭터는 사용자가 원하는 부분을 직접 선택하여 수정하지 못하고 정해져 있는 포인트 값을 방향키를 이용하여 간접적으로 수정해야 하는 점이 개선되어야 할 것이다.

참고문헌

[1] Accessibility Features of SVG W3C Note 7 August 2000
http://www.w3.org/TR/2000/NOTE-SVG-access-20000807/

[2] Adobe User to User Forum
http://www.adobe.com/support/forums/main.html

[3] http://kmh.yeungnam-c.ac.kr/Mm2/flash/flash5/etc/bitvector.htm

[4] F. Yoshikawa, K. Wada, K. Toraichi, K. Mori, H. Kiduka, K. Katagishi and A. Okamoto, "A Note on a High Resolutional Communication System Using Fluency Theory", 1997 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, pp.916-919, August 1997

[5] John D. Hobby, "Space-Efficient Outlines from Image Data via Vertex Minimization and Grid Constrains", Graphical Models and Image Processing, Vol.59, No.2, pp.73-88, 1997