Journal of The Korea Society of Computer and Information
Vol. 18, No. 4, April 2013
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2013.18.4.141

.net Framework 환경에서 하이브리드 웹 기반의 매니지드 구조화 시스템 프로그래밍 개발방식-스마트 웹 중심으로

장 승 영*

Developed Type in the Managed Structured System Programming of Hybrid Web in .NET Framework Environment-Centric of Smart Web

Seung-Young Jang*

요 약

현재 웹 구현체계는 절차지향 방식과 객체지향 방식으로 나누어 개발 해 왔다. 이러한 두 방식 모두 단순한 외형적 디자인 구성과 하이브리드 웹에 대한 보조 쓰임새 역할 구성방식으로 구현되어 개발에 대한 시간과 인력의 소비로 야기 되는 비용 문제까지 나타나고 있다. 문제해결을 위한 방안으로 .NET Framework을 기반 하에서 적용할 수 있는 매니지드 구조화 시스템 프로그래밍 방식을 제안 했다. CS의 개념을 적용한 .NET기반의 구조화 시스템 설계방식은 UI 구성측면에서의 인터페이스 구성에 최적화된 디자인 구성을 우선시 적용하여 영역별 Class 분담형식을 취해 상속관계를 유지할 수 있는 환경을 제공하며, UX와 UI의 연계성을 유지하여 UX 객체의 인터페이스 구성을 객체와 연계를 통해 Class 설계과정에서 객체형식으로 적용하게 된다. 이러한 개발방식을 통해 UI·UX 측면의 표현방식을 중심으로 구현 되고 개발에 따른 효율성 해결에도 도움을 갖게 되었다.

▸ Keywords : .NET, 스마트 웹, 웹 개발방식, UI, 클래스, 웹디자인

Abstract

The current web implementation largely falls into process-oriented and object-oriented methods. Both ways implement the programs to be used for simple design of external outfit and auxiliary function of hybrid webs and it might lead to cost problems arising from waste of time and man-power to develop with it. Therefore, managed structural system programming which can be applied under .NET framework is suggested in a bid to solve these problems. The structural

•제1저자 : 장승영
•투고일 : 2013. 1. 14, 심사일 : 2013. 3. 6, 게재확정일 : 2013. 3. 26.
* 남부대학교 디지털경영정보학과(Dept. of Digital Management Information, Nambu University)

system design method under .NET environment by applying CS concept prefers to adopting the optimized design configuration suited to proper interfaces of UI configuration, which offers the environment to maintain the inheritance relationship by taking form of the balanced class allotment by domain, and continue to keep the connectivity between UX and UI which leads to application of interface of UX objects as form of objects through connection with objects in class design process. This development method enables programs to be implemented focused on representation methods of UI and UX, and helps relieve much of the burden on development costs.

▸ Keywords : .NET, Smart Web, Web Development Method, UI, Class, Web Design

I. 서 론

1990년대 윈도우가 등장함에 따라 도스에서 윈도우 인터페이스로 변모하게 되었다. 인터페이스 변화 속에 1997년에서야 웹이라는 환경에 주목받기 시작했다.

세계 최초로 웹의 구조는 1991년 팀 버너스리(Tim Berners Lee)가 세상에 공개한 이후 절차지향 프로그래밍(Procedural Programming) 개발방식으로 구현 되어져서 웹에 발전을 이룰 수 있었다. 또한 통신 기술의 발전에 힘입어 인터넷의 대중화된 포털시대가 도래 하자 사용자의 다양한 정보에 요구를 표하게 되었다. 다양한 정보를 보장하기 위한 목적에서 웹에 구성방식에도 많은 변화를 보여 왔다. 변화된 구성방식은 객체의 구성요소들을 통해 객체지향 프로그래밍(Object-Oriented Programming)방식으로 적용하기 시작하게 되었으며, 이로 인해 웹2.0 시대를 여는 주요인으로 미루어 짐작할 수 있다.

웹2.0은 "사용자를 위한 웹"이라는 주제 속에서 사용자가 참여 속에 소통의 포털 사이트만 살아남게 되는 결과를 낳게 되었고 오프라인에서 온라인으로 시점변환이 나타나게 되는 현상으로 이어졌다. 온라인의 시점 전환은 인터넷 홍보 전략 측면에서도 적은 비용을 들여 많은 홍보와 사용자의 참여를 유도 해나가는 사용자 중심의 블로그를 창출하는 성과를 얻게 되었다. 웹2.0의 성과와 발전 과정으로 이어지면서 지금의 차세대 스마트 웹의 기본 개념과 웹의 표준화를 이루게 되는 결과도 갖추게 되었다.

웹 표준은 하이브리드 웹의 등장과 성장 가능성을 크게 열어 놓았다. 하이브리드 웹은 스마트 웹이 나아갈 방향을 제시하는 역할뿐만 아니라 무엇보다 웹3.0 시대의 시멘틱 웹으로 진입하기 위한 전환점으로 인식하게 되었다[1].

새로운 웹 방식의 하이브리드 웹 등장은 접근과 사용을 고려한 최적화 된 웹 표현방식의 기술로 분류되고 있다. 새로운 웹 방식인 하이브리드 웹은 모바일 웹과 모바일 앱(어플)의 "기능 융합"이라고 정의 된다. 기능적 융합은 플랫폼의 UI를 중심으로 구성하고 있다는 점에서 다양한 접근방법과 최적의 환경을 용의하게 구축할 수 있는 효율성을 보유하게 된다.

하이브리드 웹은 배포, 디바이스 접근제어, 로컬 스토리지 접근, Push Notification 등의 장점을 지니고 있다. 효율성 측면에서도 웹기반의 다양한 플랫폼을 지원가능 하게 되었다. 웹기반 플랫품은 기존 웹에서 사용되는 HTML, CSS, Java Script 등을 사용할 수 있으며, 웹페이지 소스를 재가공하여 활용할 수 있는 호환이 매우 뛰어나다는 점에서 주목 받는 기술이다[2]. 이처럼 구성방식의 기술을 통해 PC상에서 구현되어진 네이티브 웹과 같은 효과를 스마트 웹에서도 구현할 수 있게 되었다는 점에서 기업 스스로 안고 있는 여러 문제를 완화할 수 있는 방안으로 인식되어지기 시작했다. 하지만 하이브리드 웹의 기술적 제공에도 불구하고 소비자 중심의 다양한 네이티브 웹을 표현 하는데 많은 난항을 겪고 있다. 그 이유는 구현방법상의 문제와 구현 체계의 범위에 있어 어려움을 겪고 있기 때문이다[3].

구현방법 체계문제는 원하는 서비스를 소비자에게 제공하기까지에 대한 시간, 인력의 소비(消費)에 문제가 가장 큰 요인으로 지적받고 있다. 또한 아직까지 웹 표준에 적합한 개발툴이 보급되지 않는 이유가 소비자 중심의 다양한 플랫폼을 제공하지 못한다고 본다.

소비의 문제는 체계적 설계방식에 대한 문제에서 비롯된다. 설계방식의 문제가 소비자의 다양한 플랫폼을 제공하기 위한 한계를 부각 시키는 걸림돌로 작용되어지고 있다. 이러한 설계구현방식의 문제해결을 극복하기 위한 목적에서 매니지드 구조화 시스템 프로그래밍(Structured System Programming) 방식을 제안 하는데 있다.

본 연구에서 제안하고자 하는 매니지드 구조화 시스템 프로그래밍 방식은 웹을 구현하고 표현하는데 있어 구성 단계에서부터 서비스 수립단계까지 방법을 제시하는데 연구목적을 두었다. 웹에 구성에 있어 편의성 측면을 주된 필수 항목으로 채택하여 UI(User Interface)측면과 UX(User Experience)측면으로 사용자 관점에서 접근하기 시작했다. 매니지드 구조화 시스템 프로그래밍의 구성측면은 인터페이스의 구조적 방식에서 그 해답을 찾게 되며, 사용자 중심의 서비스 환경에 의해 다르게 표현되어져야 하는 근거 때문에 구분 짓게 된다. 매니지드 구조화 시스템 프로그래밍의 개발 방법에 제시는 결과적으로 콘텐츠의 다양성 확보를 위한 표현접근 기술방식에 따라 달리 묘사된다. 사용 언어 표현에서도 정적 언어인 하이브리드 웹과 동적 언어인 C# 코드의 혼합방식을 근거를 통해 구조화 시스템 프로그래밍의 적용범위에 기준을 두고 정립된 기준언어를 적용하게 된다. 정립된 기준은 코딩의 재활용여부와 자원의 분배에 대한 기술적 폭의 탄력성에 따라 효율을 높이는 척도로 좌우 되는 설계방식을 취할 수 있다. 무엇보다 가장 큰 성과는 하이브리드 웹의 장점으로 강조 되고 있는 단순한 접근성 측면을 다양성 측면 단계까지 끌어낼 수 있는 방안을 모색할 수 있다는 점을 들 수 있다. 보다 쉽게 개발하고 보다 다양한 플랫폼을 사용하므로 편의성 측면을 개선시킬 수 있는 개발방법을 제시하는데 목표를 둔다.

본 연구는 하이브리드 웹 기반 매니지드 구조화 시스템 프로그래밍의 구현에 관한 하나의 접근방식을 나타내는데 있다. Ⅱ장에서는 이론적 배경으로 웹 개발방식에 대한 논의와 스마트 폰의 운영 체제에 대한 주요 특징들을 고찰한다. Ⅲ장에서는 구조화 시스템 프로그래밍 방식 적용을 위한 스마트 웹 모형 설계를 각 영역별로 분류하여 제시하고 그에 따른 영역별 구현과 결과들을 제시한다. Ⅳ장에서는 결론 및 추후 보완적인 측면들을 논의한다.

Ⅱ. 관련연구 분석

웹 자체 개발 방법에 있어 지금껏 절차지향 프로그래밍(PP) 방식과 객체지향 프로그래밍(OOP) 방식을 수용하여 두 방식을 통해 웹 개발에 적용하고 있다. [표 1]와 같이 절차지향 프로그래밍과 객체지향 프로그래밍 방식은 현재 적용된 방식이며, 구조화 시스템 프로그램(SSP) 방식은 본 연구에서 고안한 방식이다.

기존 웹 개발방식으로 많이 적용되어지고 있는 절차지향 프로그래밍 방식은 명령형 프로그래밍과 동의어의 응용된 프로시저 호출개념을 의미한다. 프로시저 호출 가능여부에 의해 개발에 따른 유지보수가 쉽고 단순한 순차적 프로그래밍이나 비구조적 프로그래밍보다 여러 상황에서 장점 지니고 있다. 그러나 기능 중심의 개발 추세로 확장성이 매우 어렵다는 단점도 있다. 이러한 결함에 의해 최근 자바 기술을 적용한 CS 개념의 객체지향 프로그램 방식이 웹에서도 도입되어지는 동기가 되었다[4].

표 1. 프로그램 개발 방식 비교
Table 1. A Program Development Approach Compares

프로그램 방식	절차 지향	객체 지향
구성	모듈성	클래스, 객체, 메서드, 메시지
특징	유효범위	다중상속
장점	간단한 프로그램 사용	높은 응집력과 낮은 결합력을 지향
HTML5 역할구분	도움역할	도움역할
개발방식 적용범위	구조적 방법론 (기능)	구조적방법론 (기능) 객체지향방법론 (객체) CBD방법론 (컴포넌트)
개발방식 수용범위	구조적방 법론	선택수용가능 (1선택방식수용)

객체지향 프로그래밍 방식은 컴퓨터 프로그램이 명령어의 목록으로 보는 시각에서 벗어나 여러 개의 독립된 단위의 객체들을 하나의 모임으로 이해하고 이를 각각의 메시지로 상호소통 속에서 데이터를 처리하는 객체 방식을 뜻한다. 객체지향 프로그래밍은 프로그램이 유연하고 변경이 용이한 탓에 대규모 소프트웨어 개발에 많이 사용된다. 특히, 직관적인 코드 분석을 가능하게 하는 장점 때문에 프로그래밍을 습득하기 수월하며, 개발·보수를 간소하게 할 수 있다는 용이성을 개념화하고 있다. 객체지향 프로그래밍의 개발 방식의 수용에 있어 1가지 방식만을 채택한다는 점 때문에 과도한 객체화 프로그램 성향으로 두드러지게 나타난다. 객체화의 과도한 요인은 실제 원하는 의사를 충분히 반영하질 못한다는 점에서 큰 문제로 지목받고 있다. 절차·객체지향 프로그래밍 두 방식에 대한 난점과 HTML5의 보조 쓰임새 역할에 대한 의미에서 벗어나고 시스템화 된 웹 개발 방식을 지향하려는 목적에 상충되는 이유에서 구조화 시스템 프로그래밍 방식을 고안하게 되었다.

구조화 시스템 프로그래밍의 가장 큰 의미는 인터페이스 상속에 의한 인위적 캡슐화 형식은 구현방식이며, 다양한 개발방식을 포괄하여 개발하기 위한 목적에서 비롯되어졌다. [그림 1]와 같이 쓰임새 맞는 역할 구성을 객체로 정의하여 완성하고 인터페이스의 다양성을 확보하기 위한 수위에서 모색된 방식이다.

그림 1. 매니지드 구조화 시스템 프로그래밍 방식 구조
Fig. 1. A Managed Structured System Programmatic Structure

구조화 시스템 프로그래밍 방식은 시스템 규모가 커지고 복잡해질수록 더욱 큰 실효를 얻게 되며, 개발에 투여 되는 시간과 인력에 대한 소비를 최대한 줄일 수 있는 효과를 낼 수 있는 장점을 지닌다. 반면, 설계 과정의 운영체제의 분석이 정확하지 않으면 매우 어려운 여건으로 접하게 되는 단점을 갖게 됩니다. 그래서 개발방식의 적용 여부에 앞서 스마트 폰의 운영 체제에 대한 여부파악이 선행 되어야할 부분이며, 이에 따른 스마트 폰의 종류에 따라 간략히 살펴봐야 한다.

최근 출시되는 스마트 폰의 종류가 늘어나고 있는 추세다. 이중 Apple에서 출시된 iPone3·4·5 및 iPad1·2가 전 세계적으로 폭발적인 인기를 얻게 되었다. 지속적인 성장률 속에 HTC, 삼성전자, LG전자 등 Apple을 대항하기 위한 수많은 안드로이드 폰이 출시되고 시판되어지고 있다. Microsoft사에서도 뒤늦은 Windows Phone8기반의 스마트 폰을 10월경 출시를 내놓기도 했다. 고성능 Mobile 출현은 "IT의 르네상스 시대"로 들어서게 되는 계기를 마련했다.

이를 반영하듯 이동통신 사업자들도 패쇄적인 서비스 형태에서 개방적인 서비스 형태로 모바일 시장의 패러다임의 진화를 갖게 되었다. 각 이동통신 사업자들은 최근 3G·4G 환경의 서비스를 제공하기 시작했다. 서비스 환경변화는 스마트 폰으로 어떻게 활용될 수 있는지에 대한 가치와 기술 진화에 크게 운용되기 시작했다.

표 2. 스마트폰 운영체제별 렌더링 엔진 비교
Table 2. A Smartphone Operating System Specific Rendering Engine Compares

운영체제	OS 개방성	브라우저
IOS (애플)	패쇄적	Safari
Android (구글)	개방적	Chrome
WinPhone 8 (MS)	개방적	Internet Explorer

[표 2]와 같이 현재 국내 시장에서 널리 이용되어지고 있는 모바일 운영체제는 Android와 IOS로 압축해볼 수 있다.

Android 운영시스템은 Google을 중심으로 개발 된 오픈소스 기반의 모바일 운영체제를 말한다. Android는 개방성 때문에 보안에 취약하다는 단점이 있지만, 삼성, LG, HTC 등 브랜드에서 탑재되고 사용되어지고 있다. Android에 내장된 Web 브라우저의 명칭은 Chrome Lite라고 불리고 있다. 렌더링 엔진은 Safari 처럼 WebKit를 채택하여 HTML5를 포함한 고성능 Web 브라우저다. Android 2.2 버전부터는 Flash Player 10.1이 내장되어 PC와 똑같은 기능으로 재생 가능하다. 파일업로드는 불가능하며, 파일 다운로드는 가능하다. ZIP 압축파일 등은 내장 메모리에 저장 되며, 그 외 브라우저 창 제어가 불가능하다[5].

IOS는 Apple이라는 단일회사가 iPhone 등의 소형기기용으로 개발한 운영시스템이다. 패쇄적 플랫폼으로 Apple 제품에만 탑재가 이루어지고 있다. iPhone은 기본적으로 모비일 Safari라는 웹 브라우저가 내장되어 있다. Safari는 오픈소스 렌더링 엔진으로 WebKit를 채택하여 HTML5등의 최신 기술을 도입하고 Web 브라우저에 적용되고 있다. 모바일 Safari는 플러그인을 설치할 수 없기 때문에 Flash Player 등으로 제공되는 Flash 콘텐츠를 제공 받을 수 없다. 파일업로드, 다운로드기능이 불가능 하다[5]. 이들 운영체제별로 간략히 살펴본 결과 공통사항으로 WebKit을 채택한 HTML5 도입여부를 확인할 수 있다. 각기 다른 유형의 스마트 폰에서 HTML5 지원은 웹에 대한 스크립트의 역할을 대처할 수 있는 기반으로 조성되었다는 점이다. 결국, Web의 UI 제작과 표현이 보다 다양하게 제공할 수 있는 환경변화가 각 기업 간에 무언의 약속으로 이루어졌음을 엿볼 수 있다.

III. 구조화 시스템 프로그래밍 스마트 웹 모형 설계

1. 시스템 설계

구현 된 시스템 환경은 〔표 3〕와 같다.

표 3. 스마트 웹 구현환경
Table 3. A Smart Web Implementation Environment

구분	사양
OS	Windows Server 2008 R2 Standard
웹서버	IIS 7.0
Database	MS SQL 2008, oracle 9i
Database 연동	OLEDB
서비스플랫폼	.net Framework 4.0
구현언어	C#, HTML5
개발툴	Visual Studio 2010
테스트환경	아이폰4, Internet Explorer v9

스마트 웹에 환경은 Windows Server 2008 R2 Standard의 IIS7을 적용하게 되었다. IIS7은 ASP.NET 런타임의 확장성 모듈과 핵심 서버를 통합하여 ASP.NET를 쉽게 구현할 수 있는 환경을 제공한다. ASP.NET 런 타임버전 4.0을 적용하여 .NET Framework가 제공하는 기능의 범위의 폭이 넓어지고 다양한 컨트롤 패널을 사용할 수 있게 된다. 또한 IIS 서버측면에서도 확장성에 도움을 주고 있다. 버전에 대한 확장성은 기존에 형성된 모든 패턴의 콘텐츠에 보급되어 성능의 확장을 적용할 수 있으며, 모든 타입에 통합관리로부터 얻어지는 직·간접적인 이점을 갖게 된다. IIS7의 채택은 모든 운영권을 하나의 서버 방식으로 동일 시 되고 새로운 통합방식 모드로 지원 가능한 이점에 의해 적용하게 되었다.

〔그림 2〕와 같이 데이터베이스는 MS SQL 2008의 사용과 oracle 9i을 OLE DB(Object Linking and Embedding, Database)형식의 객체연결 구동방식으로 적용 하였다. 객체연결 삽입 데이터베이스 연동방식은 웹을 구현하기 위한 오브젝트 데이터베이스와 SQL문의 다양한 폭을 비 관계형 데이터베이스 형식으로 지원하고 있기 때문이다. 무엇보다 오라클 SQL 서버에 복잡한 데이터베이스를 단순한 텍스트 파일과 스프레드시트로 구성할 수 있으며, 가능한 모든 인터페이스를 추가하지 않고 자체 COM 모드의 오브젝트를 활용할 수 있다. COM 오브젝트는 닷넷이 보유하고 있는 기능 중 하나이며, 닷넷의 인터페이스에 매핑이 손쉽게 제공 된다.

그림 2. 서비스 시스템 흐름도
Fig. 2. A Service System Flowchart

데이터베이스의 환경인 MS SQL 2008 사용은 .NET 프레임워크의 핵심적인 정책 기반으로 컨트롤 되고 있다는 이점을 갖추고 있다. SQL 컨트롤은 ADO.NET Entity Framework 지원으로 테이블과 열에 직접적인 프로그래밍 없이 여건에 맞게 배정을 할 수 있는 논리적 데이터 엔터티를 운용하고 있다. 또한 .NET Framework는 새로운 LINQ (Language Integrated Query) 개념의 SQL유사 쿼리 구문을 기본으로 지원하고 있다. 쿼리 구문을 활용하여 크기와 복잡 여부에 상관없이 보고서 및 분석 자료를 관할 할 수 있는 확장성 인프라를 제공하게 된다. 개발 툴은 Visual Studio 2010을 사용하게 되었다. Visual Stu dio는 Microsoft사가 다양한 언어로 프로그래밍 할 수 있는 통합 개발 환경의 툴을 보급하고 있다. 매니지드 구조화 시스템 프로그래밍 개발방식에 있어 주역 언어가 C# 언어이며, C# 언어를 Visual Studio의 언어로 제공하고 있다. 통합 개발 도구인 Visual Studio는 다양한 플랫폼을 쉽게 개발할 수 있는 환경인 DMM(Document Map Margin) 기능을 제공하고 있다. DMM 기능은 독립된 메서드 및 Class의 호출하고 호출 관계를 쉽게 이해할 수 있도록 지원하는 트리 형태로 지원된다. 트리 형태의 지원은 구조의 스택 정보를 순차적으로 접근할 수 있으며, 복잡한 인터페이스 프로그래밍 시에 호출연관 관계를 구조적으로 표현해 주어 선언부와 구현부를 쉽게 검색할 수 있게 해준다.

Visual Studio는 구조화 시스템 설계 방식에 유의한 개발 환경을 제공해 주는 통합적 개발 툴로 적격한 툴 여부에 대한 인지하게 되며, 매니지드 구조화 시스템 프로그래밍 환경을 기반여건을 조성하게 된다.

2. 객체 인터페이스 클래스 설계

Class 설계 방식은 구조화 시스템 설계 구성방식을 구현

하기 위한 주역 역할을 수행한다. Class 설계 방식의 도입 목적은 소스에 재활용에 있으며, 설계 시 참고할 부분을 몇 가지로 나누어 볼 수 있다.

첫째, 소스에 대한 정보를 상호호출 가능한 연동성.

둘째, 소스의 수정정보가 반영될 수 있는 관리성.

셋째, 시스템의 성능을 향상 시킬 수 있는 확장성.

기본 Class 설계는 쓰임새 역할 부분으로 크게 나누어 Director 구성을 통해 객체 Class 구성과 사용자 인터페이스 Class 구성으로 구분하게 된다. 각각의 단위별 Class는 연동성을 바탕에 두어 제작되었다. Class 설계 구성은 각 Class 간에 대한 편의와 효율 부분에서 실효를 얻기 위해 적용하였다. 객체 Class의 사용은 하나의 Class에 코드와 데이터가 모두 집약된 기술 형식의 장르에 속하며, 객체의 캡슐 개념으로 정의해 볼 수 있다. 객체의 형태는 실제 메모리를 할당받는 물리적인 요소들을 품고 있으면 생성자와 소멸자를 호출하게 된다.

Class 구성은 구조적 함수를 가지고 있다. 구조적 함수는 정보에 대한 호출을 책임지며, Class 작업 관리와 확장에 매개체 역할을 관리한다. 결과적으로 Class 설계 적용 방식은 Web에 대한 확장과 개발 편의를 효율적으로 얻기 위한 방법으로 설계 되어졌다. Class 구성은 크게 각 역할에 용도에 의해 구분지어 구성 하였으며, 객체의 호출 방식에 따라 각 영역으로 분류되어 적용하게 되었다.

첫째. 메인화면의 구성영역.

둘째. 단순한 정보를 텍스트기반으로 제공하는 HTML 영역.

셋째. 게시판, DB 정보를 컨트롤하는 영역.

기능별·역할별로 3계층으로 나누어 객체 간에 계층별 트리형태 구조로 설계방식을 적용하였다. 계층별 트리구조는 기본 객체의 Class와 구성객체의 Class간에 객체에 대한 인터페이스의 오류를 줄여 각 객체에 대한 분할 없이 사용할 수 있게 되어 의존 종속관계를 갖게 된다.

〔그림 3〕와 같이 객체의 의존관계는 객체의 편의와 객체간의 인터페이스의 관리가 용의할 수 있도록 설계를 그려나갈 수 있게 되었다. 무엇보다 엔트리 포인트를 사용하여 접근에 대한 제약을 두지 않게 되었으며, 객체 간에 인터페이스를 개별로 구현해야 하는 불편함을 최소한 줄일 수 있는 이점을 얻게 되었다. 자료형태의 Class는 값을 가져오는 형태로 구현해주는 방식에 의해 적용해 구성 객체가 동등한 상속 구조로 표현 되어질 수 있도록 하였다. 객체의 상속은 소스 코드의 편의를 높이는 효과를 얻게 되었다. 객체 예외처리 방식은 객체의 소멸처리가 자동으로 처리될 수 있는 닷넷 환경 기능이 내장에 있으며, 객체의 공유가 이루어지지 않을 때에 발생되는 모든 문제에 대한 고려 대상에서 제외시켰다.

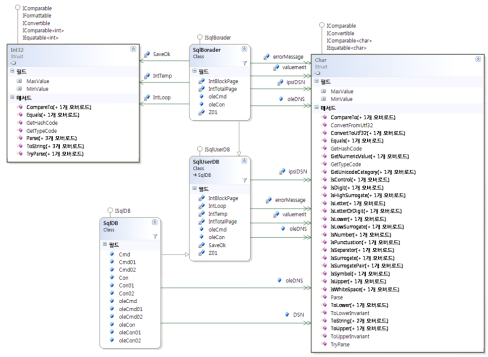


그림 3. 클래스 다이어그램
Fig. 3. A Class Diagram

사용자 인터페이스 Class 구성은 .NET Framework에서 기본으로 제공되는 사용자 정의 컨트롤을 이용하여 구성되어진다. 인터페이스 컨트롤들은 프로퍼티, 이벤트, Method를 통해서 웹 애플리케이션의 개발 모델이 매우 단순화된 구조로 구성되어져 있다. 단순구조는 개발자가 직접 필요한 컨트롤을 만들어 사용할 수 있으며, 확장과 융통성 측면에 매우 능률적 설계를 구현할 수 있게 한다. 특히, 비주얼 디자인 영역에서 매우 유용하게 적용되고 자원 활용기능을 제한 없이 운용해 웹 페이지의 재사용이 가능하게 되었다. 객체 인터페이스 Class 구성은 스마트 웹을 구성하고 있는 폼을 중심으로 화면부분과 로직부분으로 나누게 된다. 스마트 웹은 정적으로 생성 되는 HTML5 전달방식과 C#코드로 작성된 동적 전달방식이 서로 실행되어 하나의 표준화된 문서가 완성되게 된다. 동적 전달방식은 *.ascx 확장자를 가진 별개의 파일로 저장 되다. 사용자정의 컨트롤의 *.ascx 파일은 임의의 HTML 코드를 작성할 수 있는 템플릿 기반의 컨트롤로 구성되어져 있다. 템플릿 기반의 컨트롤은 Class를 상속하여 별도의 객체 Class로 구현할 수 있다. 그리고 HTML 태그와 Behind 코드가 서로 분리하여 사용된 컨트롤을 효율적으로 유지·보수할 수 있다.

〔그림 4〕와 같이 사용자 인터페이스 Class 구성은 크게 각 역할에 따른 쓰임새 의해 구성되며, 디자인 영역에 대한 재사용방식에 따라 의도적인 분류에 의해 적용하게 되었다.

첫째. 메인디자인 영역.

둘째. 서브디자인 HTML 영역.

셋째. 게시판, DB 정보를 컨트롤하는 디자인 영역.

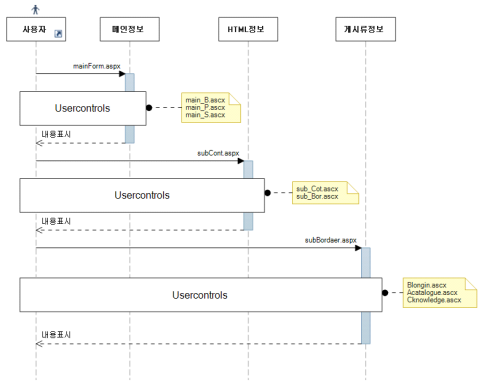


그림 4. 유저컨트롤 다이어그램
Fig. 4. A User Control Diagram

영역별 객체 Class는 기능에 의한 역할과 동등하게 구성되어져 해당 객체 값을 가지고 오게 된다. 이러한 방식은 다양한 형태의 컨트롤을 자유롭게 구현할 수 있는 구조설계로 적용하였다. 범용적 컨트롤 개발이 가능하게 되면서 페이징 컨트롤이 필요한 부분을 서비스할 수 있도록 충분히 제공하게 되었다. 상호간·객체간의 코드에 대한 재사용으로 효율성이 높아 다양한 작업을 수행할 수 있는 이점이 갖추게 되었다.

3. 사용자 인터페이스 클래스 설계

구조화 시스템 프로그래밍 방식에 있어서 스마트 웹에 적용되는 사용자 인터페이스 클래스 설계 구성은 핵심적인 요건에 속한다. 스마트 웹의 인터페이스 설계 디자인을 지나치게 단순하게 또는 복잡한 짜임새일 경우 사용자에게 혼란을 주는 문제가 따르게 된다[6].

착오에 의한 잘못된 구성방식은 디자인의 요소 및 차별성에 따른 역효과를 가져 올 수 있고 웹에 대한 본래의 뜻에 부합되는 시점 효과가 의미전달 과정에서 쇠퇴하는 현상으로 나타나게 된다. 전달 수행 측면에서도 전달하고자 하는 목적에 벗어나는 행위로 인해 오히려 브랜드 이미지에 악영향을 미치게 된다. 스마트 폰의 인터페이스 설계는 각각의 스마트 폰의 운영체제 환경에 대한 특수성과 구현하고자 하는 표현방식에 맞는 설계방식을 참고하게 된다[5].

첫째, 휴대성과 이동성.
둘째, 제한된 키 값의 입력 방식.
셋째, 해상도에 따른 색상 표현.
넷째, 사용에 대한 인지성.

또한 스마트 폰의 강점인 편리한 이동 구조의 원리 때문에 전체 화면이 그리 크지 못하다. 이러한 화면의 크기는 많은 정보를 한 화면에 보여주거나 제공하는데 많은 문제가 따른다. 이동에 의한 정보수행 작업은 사용자 입장에서 빨리 이루어질 수 있는 판단을 인지하여 화면 구성 설계단계에 순차적으로 이행해 내려가야 한다. 스마트 웹에 화면 구성은 스타일 가이드라인 조항을 정리하여 웹에 대한 설계방향을 정립하게 된다.

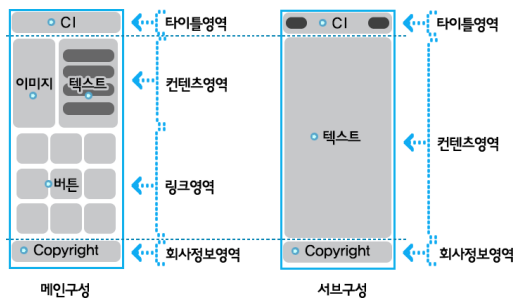
첫째, PC상의 웹과 동일한 UI의 기능제공.
둘째, 스마트 웹과 어플리케이션의 동일한 연동 기능제공.
셋째, 온라인서비스와 오프라인서비스의 동일한 브랜드이미지 제공.

화면구성에 있어 상단 프레임에는 사용자에게 주요 공지항목을 쉽게 각인 시키고 접근성을 고취시키기 위한 방도로 이미지화 된 알림형식의 메뉴구성을 취하게 되었다. 메뉴구성은 사용자의 간단한 버튼 형식으로 표현하여 일목요연한 시각적인 요건을 반영 하였다. 시각적 사용성에 대한 고려항목을 확정해 사용자 인터페이스에 기준을 두며, 디자인 흐름의 구성을 열거하여 사용자 인터페이스와 동일성 있게 내비게이션 흐름을 유도하는 구성으로 적용하게 되었다.

첫째, 사용편의성.
둘째, 명확한 시각적인 구조.
셋째, 가독성.

레이어 구성방식도 사용자의 목적에 따른 단계별 시각적 분류체계를 반영하여 적용하게 되었다.

첫째, 메인화면 구조를 고려하여 디자인 요소와 맞게 구획 설정.
둘째, 일관된 형식을 구성하여 다음 구성에 대한 예상할 수 있는 구조 설정.
셋째, 분류 단계에서 공통된 형식을 분류하여 정보를 제공하는 순차 설정.



기본적으로 콘텐츠 영역별 흐름의 방향은 세로를 준하는 기본형식으로 배치하였다. [그림 5]와 같이 영역별 구간은 상·중·하 3영역 활용별 시각적 분리를 구획하여 구조요소를

적용하게 되었다.

상단, 타이틀 영역.

중단, 콘텐츠 영역.

하단, 기관정보영역.

영역별 구간 제시는 사용자들의 심리를 만족시킬 있는 관심을 유도할 수 있도록 중요한 골자로 부각 시켜 이를 반영하고자 하였다. 사용자 시선이 먼저 가는 상단에는 타이틀 이미지를 배치하여 원하는 Site에 접근 여부를 재확인 할 수 있도록 판별방식을 적용 하였다. 그 외 세부항목을 순차적으로 배치하여 자연스럽게 상단에서 하단으로 시점의 동선을 계획에 의해 제시하게 되었다. 시각적 흐름은 영역 속성에 대한 통일과 조합을 적합하게 적용하여 콘텐츠 위치를 예상할 수 있는 공간배치에 유효성을 극대화 시키려는 의도로 구성하게 되었다. Site의 콘셉트를 시각적으로 표현 할 때 우선적으로 참고할 요건은 색상 선택을 들 수 있다. 색상을 통해서 Site의 표현하고자 하는 이미지가 잘 반영되기 때문에 각별히 주의해서 선정해야할 사항으로 본다. [그림 6]와 같이 색상은 주어진 테마를 토대로 시선의 표현과 일치될 수 있는 이미지로 구별하게 되었다.

첫째, 주조색상.(Primary Color)

둘째, 보조색상.(Secondary Color)

셋째, 배경색상.(Background Color)

그림 6. Main Color
Fig. 6. A Main Colo

등차 구분으로 화려함 보다는 일관성 있는 색상처리로 인해 인지률을 높이는 역할을 수행할 수 있다. 주조색상과 보조색상은 무채색 계열과 산뜻한 색상으로 시선 효과를 겨냥하려고 하였다.

[그림 7]와 같이 배경색상은 무채색 계열을 사용하여 단순하고 깔끔한 상(像)을 주려고 하였다.

폰트칼라도 무채색 계열을 사용하여 색상과 동일한 깔끔한 이미지를 반영하여 일관성 있는 칼라로 구성 하였다.

색상은 세 가지 색상을 한정하여 접근성을 고려한 단일한 컬러로 구성 하였으며, 전체 디자인 흐름에 맞는 정결한 컬러를 활용하여 가변 폭을 고려한 디자인으로 구성하게 되었다.

[그림 8]와 같이 레이아웃 구성 또한 3단계 영역요소를 구획하여 시각적 동선에 일치되도록 구성하였다.

무엇보다 예측 가능한 형식체계와 터치스크린 방식의 장점을 최대한 반영될 수 있도록 배치구도를 적용하였다. 이미지는 콘텐츠와 이미지가 잘 융합 되도록 구상하여 전체적인 Site의 이미지를 잘 표현해 낼 수 있도록 디자인에 초점을 맞추어 설계 하였다.

그림8. Depth별 디자인 화면
Fig. 8. A Depth-specific Design Screen

이미지 파일 형식도 PNG(Portable Network Graphics) 타입을 사용해 이미지 원본과 동일한 해상도를 Site에 반영할 수 있도록 하였다. 스마트 웹에서 디자인 구성 중 아이콘은 전반적인 Site 성격을 좌우하는 묘사방식에 속한다. Smart Web에서의 아이콘의 쓰임새는 많은 가치를 가지고 있지만, 그중에서 큰 부분을 차지하는 사항이 메뉴구성 리듬에 있다. Smart Web을 구현할 때 아이콘의 기능은 텍스트의 배제에서 활용범위가 크며, 전체 Site에 대한 이미지를 표현하기 위해 사용되는 그래픽 심벌을 의미한다[6].

아이콘 제작은 표현기법 측면에서 논리성과 정확성이 부여되며, 사용자에게 인지 능력을 뚜렷이 반영되어야 한다. 아이콘을 제작 시 참고할 사항은 기능별로 나누어 제작이 이루어

저야 한다[7].

첫째, 문자보다 빠른 인지 가능한 신속성.

둘째, 의미전달이 가능한 의미성.

셋째, 시각적 기억이 반영 될 수 있는 기억성.

넷째, 누구나 보면 공감할 수 있는 대중성.

다섯째, 시각적 즐거움을 줄 수 있는 흥미성.

[그림 9]와 같이 아이콘 처리방식은 마우스 커서가 손가락으로 대치되는 멀티 터치스크린 방식을 사용하기 때문에 기본크기를 가로 70px, 세로 70px으로 규정하여 적용하였다. 메인은 화면구성의 인식도를 참고하여 기본크기를 적용하였고 서브는 표현상의 의미 전달을 목적으로 하기 위해 가로 77px, 세로 96px를 구성하였다. 일관된 수치는 레이아웃 구성을 명확한 해석을 높이는 원칙에 입각하여 메인과 서브의 크기에 구별을 두고 시각적 방향에 설득을 높이려는 의도에서 표현하게 되었다. 입체감과 테두리를 주어 아이콘에 링크의 개념을 무의식적으로 인지할 수 있도록 간접표현 방식을 적용하고 통일된 디자인적 요소의 흐름을 벗어나지 않는 기능으로 중심을 두게 되었다.

4. 구현 결과

기존 개발 방식은 [그림 10]과 같이 각 페이지를 개별방식으로 구성하게 된다. 개별구성 방식은 웹의 용량뿐만 아니라 개발에 대한 효율성·인력 부분에 소모를 가중시켜왔다.

반면, 구조화시스템 프로그래밍 방식의 형태는 구조 역할에 따라 혹은 쓰임새에 따라 3계층으로 분류하여 구분하고 계획하여 설계된다.

첫째, 메인 영역.

둘째, HTML 영역.

셋째, 게시영역.

[그림10]과 같이 모든 설계표현 방향도 기존 웹 방식을 배제한 Class 설계에 의해 상속관계를 유지하는 구조형태로 진행되어진다. 구조화 설계는 인터페이스 간의 호출에 의한 상속관계 범위를 유지하면서 개발과 관리에 손쉬운 확보를 주안으로 삼게 되었다. 메인화면은 각기 기능의 쓰임새에 따라 크게 콘텐츠부분과 링크부분으로 나누어 화면의 프레임 구성을 띄게 된다. 메인화면의 쓰임새에 의해 상·중·하 3영역 활용별 시각적 분류를 구획하여 구조요건으로 적용하게 되었다. 메인화면에 콘텐츠 영역의 공지항목은 쉽게 각인 시키고 접근성을 고취시키기 위해 알림형식을 적용하게 되었다. 또한 버튼식 메뉴구성을 표현하여 일목요연한 시각적인 요건을 반영 하였다. 특히, 버튼에 입체감과 테두리를 주어 링크의 개념을 표현하고 통일된 하나의 디자인적 요소의 흐름을 벗어나지 않는

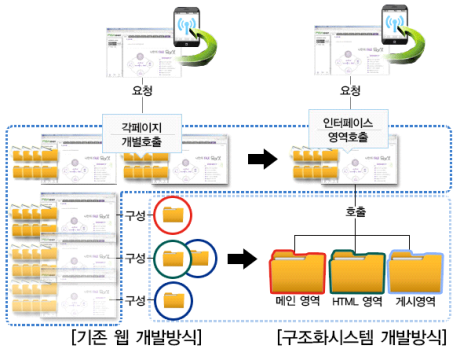


그림10. 개발방식 비교

Fig. 10. A Development Practices Compares

기능에 대한 범주를 활용할 수 있게 하였다.

메인화면 Class 구조는 영역속성에 따라 사용자 인터페이스의 디자인 호출과 Class를 호출의 객체 호출방식으로 구획하여 진행 하였다. 화면의 폼 모드는 사용자 인터페이스를 이용한 역할 쓰임새로 구별하여 호출방식으로 전개 하였다. 사용자 인터페이스의 사용에 대한 구조적 흐름은 닷넷이 소유한 자동화된 기능에 의해 생성방식으로 생성된다. 자동화 적용 구조방식은 화면구성에 먼저 선행되어져야 하며, 화면구성 영역에 부합되는 영역 위에 드래그 방식으로 자동호출 방식을 적용하게 된다. 객체 Class는 자료형태의 Class로 먼저 선언부에 선언 해 놓아야 한다. 선언된 Class는 각 속성에 맞는 호출 값을 제시하게 된다. 호출방식은 *.cs 확장자를 가진 별개의 파일을 호출하게 되며, 메인 화면의 콘텐츠 영역의 공지항목과 같은 DB의 항목 등을 *.cs 파일에 public 형식에 의해 선언하는 형식을 표(表)하게 된다. 선언한 각 멤버에 의해 필요에 따라 콘텐츠 영역에 호출하여 적합하게 사용하게 되는 구조를 이루고 있다. 선언한 멤버 구성은 Method 에 의한 코드 값을 구획화 하여 선언하게 된다.

구획화 선언으로 메인 화면의 DB 정보 호출은 DB호출 담당하는 객체 Class가 링크정보 호출은 링크호출을 담당하는 객체 Class로 각 호출기능에 맞는 해당 정보를 제공하게 된다.

[그림 11]은 호출 기능방식의 절차에 의해 사용자 인터페이스 Class 객체를 선언하고 이벤트 Handler 메서드를 정의하여 DataBind 통해 렌더링 되어 진다. 메인화면의 수행 절차를 걸쳐 짜임새 있는 구현이 이루어진다. HTML 영역은 텍스트기반으로 정보를 제공하는 영역으로 구성 된다. HTML 영역은 각 해당 콘텐츠 자료를 DB화 처리가 우선시 되게 된다.

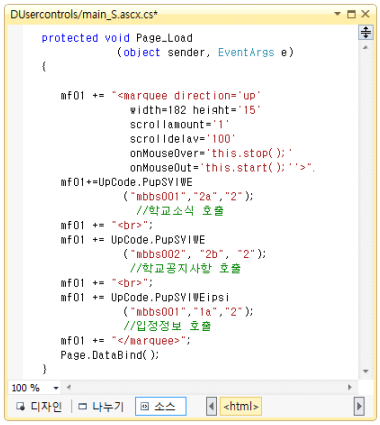
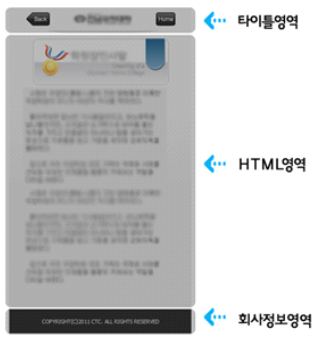


그림11. 메인화면 정보 클래스 호출 코드 (main_S.ascx.cs).
Fig. 11. A Main Screen Information Class Code Calls (main_S.ascx.cs)

기존 Web에 대한 구성은 흔히 각 HTML을 프레임 네임 속성에 의해 호출하는 방식으로 구축되어져 왔다. 다시 말해 개별구성방식을 취하고 있다. HTML의 자료 호출방식은 모든 콘텐츠를 개별적으로 HTML 문서화 시켜 단위별 문서 호출방식으로 이루는 구성방식을 적용하였다. 이러한 방대한 문서들을 스마트 웹에서 수용하기에는 그리 좋지 못한 플랜으로 본다. 다량의 HTML 문서 취합은 용량의 보관과 절차적 수정에 대한 Load 상황의 문제로 즉각적인 정보제공 대응에 비효율성을 갖게 된다. 비효율성에 벗어나기 위한 이유에서 HTML 정보를 템플릿 할 수 있는 환경으로 유지하려는 의도에서 DB 구성으로 적용하게 되었다. HTML 영역은 메인화면 영역과 동일한 화면구성 방법으로 구현된다.



영역은 각기 기능의 쓰임새에 따라 〔그림 12〕와 같이 크게 콘텐츠부분과 링크부분으로 나누어 화면 분할되어 구조를 띄게 된다. 상·하단인 타이틀 영역과 회사 정보영역은 Site에 접근 여부를 재확인 할 수 있도록 식별이 가능하도록 시각적 구성요소로 적용 하였다.

HTML에 영역에 대해서는 메인화면 구조 방식과 동일한 적용방식으로 진행되어진다. 하지만 HTML 영역에서는 메인화면구성에 대한 호출선언방식이 다르게 표현된다. HTML 영역의 화면 구성에 대한 폼은 Class를 담당한 사용자 인터페이스 리딩 과정은 동일 시 된다. HTML 영역의 호출방식은 동적 호출방식을 적용하게 된다. 동적인 호출의 주목적은 해당 값에 대한 용의성을 고려한 방안으로 적용 되었다. 동적 사용방식은 컨트롤 Class의 호출방식이 고정방식이 아닌 값에 의한 가변방식으로 적용 된다는 점이 메인영역과 다르게 표현된다. 가변방식은 Panel 메서드를 생성하여 해당 값에 대한 적용에 의해 사용하게 된다.

HTML 영역에 알맞게 배치시킨 화면위에 Panel 컨트롤을 생성한 후 패턴의 값을 적용하게 된다. Panel 패턴 값을 if문을 사용하여 적용되는 값에 의해 객체 담당 Class를 호출하고 컨트롤 되어질 수 있는 환경으로 구현하게 된다.

게시류 영역은 주로 DB에 대한 접근방식을 고려하여 적용 된다.

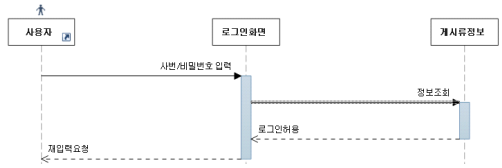


그림13. 로그인 시퀀스 다이어그램
Fig. 13. A Login Sequence Diagram

〔그림 13〕와 같이 게시류 영역에서 적용 초점은 로그인 방식이며, 로그인 방식의 권한 여부에 따라 정보의 제공이 이루어질 수 있도록 분할하여 사용하게 되었다. 사용자 권한 방식의 절차를 통해 각 영역에 의하여 구성하게 된다. DB 호출에 있어 HTML 영역 방식과 동일한 동적방식을 취한다. 하지만 게시류 영역 구성은 이보다 더 복잡한 구조를 띄며, DB 호출에 있어 데이터베이스의 커넥션 정보를 유동변동 방식을 사용해 구성하고 있다. DB의 호출을 위해서는 web.config 파일에 의해 connectionStrings에 add함수를 적용해 각 DB의 정보 값을 기재한다. DB 정보 값을 MS가 제공하고 있는 표준화된 DB 연결 기술인 OLE DB를 사용하여 자동 호출하게 된다. 이러한 DB 정보의 적용은 수정에 대한 편의성과 DB 정보 보안에 대한 숨김 장치로 인식해도 좋을 것 같다. web.config에 쓰임새 역할에 따라 달리 적용하거나 동일한

Class상에서 값의 전달 방식을 달리 적용하여 구현될 수 있는 환경을 조성할 수 있게 된다.

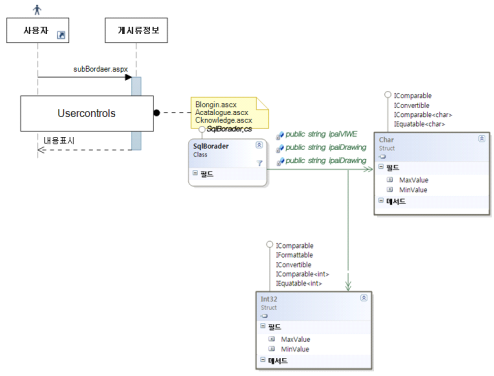


그림14. 게시 영역 클래스 호출 흐름도
Fig. 14. A Published Class Diagram Call Flow

〔그림 14〕와 같이 게시 영역에 있어 글을 보기 위한 모든 인터페이스 구조는 로그인 접근방식의 권한에 의해 제공되는 형식을 취하고 있다. Class 접근방식은 HTML 영역과 동일한 방법 적용되며, 로그인 접근에 의해 권한부여를 통하여 제공되는 호출방식은 세션 값에 의해 적용된다. 전달되어진 값의 체크에 의하여 세션의 논리적 연결을 통해 값이 유지되며, 값에 의한 하나의 담당 Class를 호출하여 처리방식으로 표현된다.

로그인 화면 게시물리스트 화면

〔그림 15〕와 같이 담당 호출방식은 권한에 따라 로그인 화면과 View 화면이 구분 되어져 호출되는 화면으로 정리되고 구현된다.

Ⅳ. 결론

본 연구에서 하이브리드 웹에 대한 개선점을 찾기 위한 목적으로 매니지드 구조화 시스템 개발방식을 제안하게 되었다. 매니지드 구조화 시스템 개발방식의 제안은 개발 방법론을 중심으로 제시되었으며, 구조적 설계방식의 개선을 통해 다양한 플랫폼 제공하게 되었다. 스마트 웹 설계방식은 인터페이스 설계방식과 구조화 시스템 설계방식으로 설계되어 객체에 대한 활용을 높이는 인식성 있는 설계방식을 적용하게 되었다. 구조화된 설계 방식은 단순한 시스템 구성 방식이 아닌 콘텐츠 중심의 분야별 과제로 분석되어지며, 이를 토대로 범용적 구조를 갖게 되었다. 구조화 시스템 설계형식을 구성하기까지는 다음과 같은 분석 단계를 통해 얻게 되었다.

첫째. 각기 다른 운영체제를 고려한 모바일 웹 로드맵에 따른 역할 분석.
둘째. Site 기본 중심 설계의 검색엔진에 대한 최적화 분석.
셋째. 실시간 양방향성에 대한 인터페이스 측면 분석.
넷째. 방문목적의 인지 높은 디자인요소 분석.
다섯째. 화면영역에 대한 기본적 페이지 분석.
여섯째. 정확한 내비게이션의 흐름 분석.
일곱째. 콘텐츠 영역에 대한 영역별 분석.

상호간 분석에 준한 구조화 시스템 설계방식으로 방향의 초점을 맞추어 스마트 웹에 적용될 수 있도록 구성 되어졌다.

구조화 시스템 설계 적용은 사용자 중심의 접근이 용의하도록 반영할 수 있게 되었었다. 그 결과 사용자 중심의 UI의 표현 또한 다양한 각도에서 적용할 수 있는 성과를 얻게 되었다. 또한 Web 서비스 되는 파일용량에서도 적은 용량의 서비스가 제공 될 수 있게 되었다. 물론 스마트 웹에 대한 적용은 단순한 편의적 개발과정에서 비롯된 것만은 아니다. 웹에 대한 분석과정이 콘텐츠의 묘사에만 국한 되어져 개발되어지기보다는 최소한 광범위한 기술적 범용단계까지 설계방식이 이루어져야 한다고 본다. 이러한 Web 표현 기술방식만으로는 모든 단말기의 기능 측면에서 완벽하게 구현하고 표현하는 되는 많은 문제점 야기하고 있다. 현재 상용화 되고 있는 3D 그래픽 표준 기술이나 2D가속을 지원하는 환경에 맞는 구현 또한 어려움을 안고 있다. 음성, 영상, 멀티미디어 지원, 카메라 장치 제어 기능을 요구하는 앱에서 까지도 구현하기는 어려움이 있는 것이 현 상황이다. 이는 차세대 웹 표준이 개발 진행 중이며, 각가지의 단말기의 제어에 관한 표준 웹 기술에 대한 정의가 진행 중이기 때문이기도 하다. 따라서 웹과 웹기반 앱 개발 기술이 부족해서라기보다는 관련 웹 표준과 이를 지원하는 브라우저와 단말기 성능상의 한계 때문에 시간이 더 필요로 한 상황이다. 또한 웹과 웹기반 앱 개발 기술의 확장에 관심과 표현방식의 연구가 계속 진행 되어져야 할 과제로 남아 있다.

참고문헌

[1] SHee-Wan Kim, So-Young Kang, Jae-Hwa Kang, Dong-Soo Kim, "A Design on the Audit Framework of the User Interface for the Web Accessibility", Korean Society Of Computer And Information, Vol.15, No. 4, pp.107-118, 2010.

[2] S.Y. Lee, H.W. Jung, "Trends of Standardization on Future Mobile Web Platform", Electronics and Telecommunication, Vol.25, No. 3, pp.11-17, 2010.

[3] Won Seok Lee, "Studies of the services model for mobile business", TTA Jour nal, No.128, pp. 50-54, 2010.

[4] Kyung-Soo Joo, Jung-Woong Woo, "A Development of the Unified Object-Oriented Analysis and Design Methodology for Security-Critical Web Applications Based on Object-Relational Database - Forcusing on Oracle11g -", Korean Society Of Computer And Information, Vol.17 No. 12, pp. 169-177, 2012.

[5] Seok-yeol Kang, "Workload-Aware Load Balancing for Cluster Web Servers", Korea IT Business Promotion Association, 10-PO-12, 2010.

[6] Kim Myeongsan, Suh Seunghui, "Study on the relation between Web Identity and Brand Personality of Promotion Site", A Journal of Brand Design Association of Korea, Vol.4 No. 1, pp.183-200, 2006.

[7] Shin Jaeun, "A study of the Effective User Interface Design on the Mobile Device -With focus on the Cellular Phone-", Communication Design Association of Korea, Vol.14 No. 1, pp.45-56, 2003.

저 자 소 개

장 승 영

2009 : 남부대학교
디지털경영정보학과 공학석사.

2013 : 남부대학교
디지털경영정보학과 공학박사

현 재 : 전남과학대학교
정보전산원 대리 재직 중

관심분야 : Web 개발·기획·분석·
정책수립

Email : sychaing@hotmail.com