Journal of The Korea Society of Computer and Information
Vol. 20 No. 11, pp. 183-190, November 2015
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.11.183

# A Study of Visual Programming Environment for NPE(Novice Programming Environment)

Ji-Wan Kim*, Hyun-Gon Seo **

## Abstract

This paper investigates the three main functions of a typical visual app programming environment for Novice Programming developers, and compares the features. The Scratch is a visual programming environment for education, anyone can create a story easy as possible variously interaction, games, animations and more. App inventor provides precise and professional application development capabilities as compared with scratch. App Inventor in runs independently of the computer platform, and has a feature that must be constantly connected to the server over the internet, while the Inventor app runs. M-Bizmaker is suitable for commercial application development, consists of m-BizBuilder, m-BizEngine, m-BizServer or the like, provides a cross-platform visual programming environment.

▸ Keyword : App, Visual Programming Environment, App inventor, Scratch, m-Bizmaker

## I. Introduction

이전의 프로그램 개발은 개발자 중심으로 이루어 졌지만 최근에는 소프트웨어를 사용하는 사용자들이 자신의 필요성에 따라 프로그램을 개발할 수 있는 시대로 전계되고 있다. 같은 맥락으로 가트너(Gartner)는 'IT 엑스포 2014'를 통해 '2015년 10대 전략기술' 중 하나로 'Computing Everywhere'를 제시했다[1]. 'Computing Everywhere'는 다양한 환경에서 모바일 이용자들의 욕구에 대응할 수 있는 기술이 더 강조되어 그로 인해 사용자 중심의 개발환경에 대한 관심이 증가할 것으로 예견했다.

사용자 중심의 프로그래밍환경에 대한 관심 증대는 개발환경의 변화로 이어져 코드기반 프로그래밍환경(Code Based Programming Environment)중심에서 비주얼프로그래밍환경(Visual Programming Environment)으로의 변화를 불러왔다. 비주얼프로그래밍환경은 사용자가 필요로 하는 프로그램을 쉽게 개발할 수 있도록 편리한 기능을 시각적으로 접근 할 수 있는 기능을 제공해준다[2]. 스마트폰 운영체제인 iOS와 안드로이드도 플랫폼의 시장 확대를 위해 프로그래밍 언어에 대한 전문지식이 없어도 누구나 스마트폰 앱을 쉽게 개발할 수 있는 비주얼 프로그래밍 환경을 제공하고 있다[3]. 같은 맥락으로 최근 구글은 자체 개발도구인 안드로이드스튜디오1.0을 공개했다[4]. 이는 지능형 코드 편집, 사용자 인터페이스(UI) 디자인 툴, 성능 분석 기능 등이 포함돼 클릭 몇 번으로 필요한 기능을 코드로 요청하고 프로젝트에 넣을 수 있어 사용자 중심의 비주얼프로그래밍 환경으로의 변화를 보여주고 있다.

최근 모바일기기의 대중화로 교육시스템에서도 웹을 모바일 웹앱으로 확장되는 변화가 일어나고 있고[5], 초등학생과 같이 코드 기반 프로그램을 개발하기 어려운 학습자들에게도 쉽게 프로그램을 개발하기 위한 시도들이 이루어졌다[6].

따라서 본 논문은 스마트 기기에 실행될 앱 개발을 원하는 NPE(Novice Programming Environments)를 위해 대표적인 비주얼프로그래밍환경인 스크래치, 앱인벤터, 엠비즈메이커의 각 환경별 특징을 알아보고, 그로부터 도출한 항목(Stand alone, Programming, Emulator, DB system, Target Platform 등)을 기준으로 NPE가 짧은 시간 동안 쉽고 편리하게 앱을 개발할 수 있는 환경을 비교·분석 한다.

본 논문의 구성은 다음과 같다. 2장에서는 비주얼 프로그래밍 환경에 대한 기존 연구를 분석하고, 3장 각 절에서 교육용 비주얼 프로그래밍 환경인 스크래치(Scratch)에 대해 알아보고, 앱개발 비주얼 프로그래밍 환경인 앱인벤터(App Inventor)와 엠비즈메이커(m-BizMaker)에 대해 알아본다. 4장을 통해

• First Author: Ji-Wan Kim, Corresponding Author: Hyun-Gon Seo
*Ji-Wan Kim(chamui81@gmail.com), School of Information Communication & Broadcastings, Halla University
**Hyun-Gon Seo(hgseo@halla.ac.kr), School of Information Communication & Broadcastings, Halla University
• Received: 2015. 10. 07, Revised: 2015. 10. 14, Accepted: 2015. 10. 18.
• This work was supported by the year 2014, Halla University.

앞서 언급된 세 가지 비주얼프로그래밍환경에 대해 비교, 분석하여, 5장에서 결론을 맺는다.

## II. Preliminaries

### 1. Related works

일반적으로 코드기반 프로그래밍은 먼저 개발에디터(Editor)에 플러그인(Plugin)과 SDK(Software Development Kit)를 설치하고, 시스템의 구성 요소 및 전체 개발 순서에 맞는 개발 언어와 API를 사용하여 프로그램 코드를 작성한다. 이와 같은 과정은 특히 조건문(if, if-else)과 반복(while, for) 등이 포함된 제어 구조를 이해한다는 점에서 전문개발자가 아닌 실제 사용자에게는 매우 어려운 일이다[7]. 따라서 프로그래밍을 막 배우기 시작한 미숙자에게는 위지윅(WYSIWYG : What You See Is What You Get) 기반 보다는 코드 작성이 필요 없는 비주얼 프로그래밍환경이 적합하다. 코드 작성 없이 그래픽을 이용해 프로그램을 구현하는 것은 구현된 프로그램의 실행흐름을 쉽게 보여준다[8]. 실행흐름을 시각적으로 보여주는 방법을 제공함으로써 사용자가 프로그램이 실행되는 과정을 쉽게 이해할 수 있으며 비주얼프로그래밍환경의 도입은 코딩 과정에서 생기는 세미콜론(semi-colon) 누락, 잘못된 코드 등으로부터 프로그래밍이 방해 받는 것을 없애고 더 기능적인 구현에 집중할 수 있도록 할 수 있다[9].

비주얼프로그래밍환경에 대한 연구는 통합개발환경(IDE : Integrated Development Environment)에 대한 연구에서부터 코드 작성이 필요 없는 비주얼프로그래밍환경에 대한 연구가 이루어지고 있다. 코드 작성이 필요 없는 비주얼프로그래밍환경(non-coding visual programming environment)은 GUI(Graphical user interface) 응용프로그램 개발용으로, 쉽고 빠르게 응용프로그램을 만들 수 있다. 기존 통합개발환경은 코딩, 디버그, 컴파일, 배포 등의 프로그램 개발에 관련된 작업을 하나의 프로그램 안에서 처리하는 대화형 인터페이스를 제공한다면, 코드 작성이 필요 없는 비주얼프로그래밍 환경은 코딩, 디버그, 컴파일, 배포 등의 과정이 자동화되거나 기존 통합개발환경에 비해 단순화 되어 있다.

대표적인 스마트폰 운영체제인 애플의 iOS와 구글의 안드로이드도 플랫폼의 시장 확대를 위해 프로그래밍 언어에 대한 적문지식이 없어도 누구나 스마트폰 앱을 쉽게 개발할 수 있는 비주얼 프로그래밍 환경을 제공하고 있다. 프로그래밍을 배우지 않은 일반 사용자들도 코드 작성이 필요 없는 비주얼프로그래밍환경의 도움을 받는다면 제작에 대한 흥미와 자신감을 얻을 수 있다. 비주얼프로그래밍환경은 일반 PC 환경 기반의 경우 엘리스(Alice)와 스크래치(Scratch), 모바일 운영체제의 경우 iOS에는 코로나가, 안드로이드에는 앱인벤터가 대표적이다[3,10]. 스크래치는 초등학생을 대상으로 한 프로그램 교육 활용 관련 연구가 다수고, 앱인벤터도 스크래치에 비해 고학년의 학생들을 대상으로 프로그래밍 교육을 진행하는 연구가 많다[3,5,10,12,13]. 하지만 프로그래밍 교육을 쉽게 접할 수 없는 일반 사용자가 개발할 수 있는 프로그래밍환경에 대한 연구가 부족하다. 그러므로 전문개발자를 포함해 일반 사용자까지 쉽고 편한 개발이 가능한 프로그래밍환경의 제공을 위한 연구가 필요하다.

## III. App Development Environment

본 절에서는 앱 개발을 위해 제안된 스크래치, 앱 인벤터 및 엠비즈메이커의 세부적인 기능과 특징 및 장단점을 보고한다. 스크래치는 앱 개발환경은 아닌지만 앱 인벤터를 개발한 MIT 미디어랩에서 개발된 것이기 때문에 본 논문의 비교대상에 삽입되었다.

### 3.1 스크래치

스크래치는 미국의 매사추세츠 공과대학(MIT) 미디어랩과 Lifelong kindergarten 그룹에서 2007년 1월에 개발한 교육용 프로그램 개발환경으로, 어린이를 비롯해 컴퓨터에 대한 지식이 전혀 없는 일반인까지도 쉽게 프로그램을 만들 수 있게 하려는 목적으로 개발되었다[11,12]. 이미지나 사운드를 비주얼 프로그래밍 블록(block)들을 마우스로 끌어다 연결해 가면서 프로그램을 만들기 때문에 프로그래밍을 처음 접하는 사람도 쉽고 다양하게 상호작용이 가능한 스토리, 게임, 애니메이션 등을 만들 수 있다. 스크래치는 비주얼프로그래밍환경과 커뮤니티 기반 웹 인터페이스로 구성되어 있다. 스크래치 2.0에서는 웹 인터페이스에서 프로젝트를 만들 수 있어, 스크래치 소프트웨어를 다운로드하거나 공유를 위해 프로젝트를 업로드 할 필요가 없다. 또한 웹캠을 사용하여 손이나 몸을 움직이는 것으로 프로젝트와 상호작용 할 수 있고, 웹 인터페이스상의 공유와 링크가 가능하다.

그림 1은 스크래치 비주얼프로그래밍환경 화면 구성을 나타낸다.


Fig. 1. Visual Programming Environment of Scratch

그림 2는 커뮤니티 기반 웹 인터페이스를 보여준다. 웹 인터

페이스에서는 만들기(Create) 기능으로 스크래치 비주얼프로그래밍환경에서와 같은 프로그래밍이 가능하고, 탐험하기(Explore)로 프로그램을 공유할 수 있으며, 토론하기(Discuss)를 통해 사용자들의 질의응답을 확인하고, 프로그래밍을 하면서 생긴 문제를 해결 할 수 있다. 도움말(Help)에서는 스크래치를 이용하는데 필요한 가이드 외 교육관련 자료, 통계, 스크래치에 연결 가능한 하드웨어 등을 볼 수 있다.


Fig. 2. Scratch community based Web interface

스크래치 프로젝트는 스프라이트(sprite)라는 객체로 만들어지며 스프라이트는 기본적으로 스크래치 내에 저장되어있는 이미지를 불러오거나 웹 인터페이스(http://scratch.mit.edu)에서 다운로드 받을 수도 있고 페인트에디터(paint editor)를 이용해 직접 그릴 수도 있다. 스프라이트는 스크립트(script), 코스튬(costume), 소리(sounds) 3가지 요소로 이루어진다. 스크립트를 통해 스프라이트의 동작을 구성하고, 코스튬은 스프라이트의 기본모양부터 변화된 모양까지 스프라이트에 코스튬을 적용하는 방법으로 전반적인 이미지를 다양하게 표현하며, 소리는 스프라이트와 관련된 음향이나 소리를 제어한다.

스크래치는 스크립트(script) 블록을 쌓는 것으로 프로그래밍하며, 이는 별도의 컴파일러 없이 비주얼 프로그래밍 환경이나 웹 인터페이스에서 프로그래밍 되고 실행된다. 개발에 사용되는 블록들은 8가지 기능(동작, 제어, 형태, 관찰, 소리, 연산, 펜, 변수)별로 구분되는데, 각 블록 그룹들은 색으로 구분된다. 또한 블록에 쓰여 있는 단어로도 직관적으로 프로그램의 구조와 알고리즘의 파악이 가능하다. 블록 각각의 기능은 표 1과 같다.

Table 1. Function of Block

| 블록 | 기능 | 블록 | 기능 |
|---|---|---|---|
| 동작 | sprite의 움직임 제어 | 제어 | 반복 및 조건에 따른 제어 |
| 형태 | sprite/stage의 모양변화 | 관찰 | 키보드 및 마우스 상태 값 읽음 |
| 소리 | 각종 소리 제어 | 연산 | 산술,논리연산 및 난수 발생 등 |
| 펜 | 펜의 굵기나 색 선택 | 변수 | 변수나 리스트(1차원 배열) 선언 |

또한 웹 인터페이스를 통해 각자 개발한 프로그램을 스크래치 온라인 커뮤니티(http://scratch.mit.edu)에 공유하며, 다른 사람의 프로젝트도 내려 받아 분석, 재구성이 가능하다. 그 밖에 레고(Lego), 아두이노(Arduino) 보드와 결합된 프로젝트 개발이 가능하다[10].

### 3.2 앱인벤터

앱인벤터는 안드로이드 스마트폰 앱을 개발할 수 있는 이벤트기반 비주얼프로그래밍환경이다. 앱인벤터는 구글과 MIT가 함께 지원하여 2010년 11월부터 무료로 서비스 되었다. 이후 2012년부터 MIT로 이전되어 현재 앱인벤터2로 서비스되고 있다. 앱인벤터의 기본 플랫폼은 스크래치에 기반하며, 블록을 사용하여 논리를 구성한다는 면에서 스크래치와 비슷하다[10]. 하지만 사용자 인터페이스의 레이아웃과 디자인을 먼저 구상하고 블록의 종류가 다양하여 스크래치에 비해 정교한 프로그램을 만들 수 있다. 또한 각종 센서 및 GPS 등 스마트 디바이스의 다양한 기능을 제어할 수 있다는 점에서 앱 개발 교육용 환경에 그치지 않고, 실생활에 적용 가능한 앱을 만들 수 있다. 따라서 앱인벤터를 통해 사용자는 초기 프로토타입(prototype)의 앱을 좀 더 전문적이고 빠르게 만들 수 있다[13].

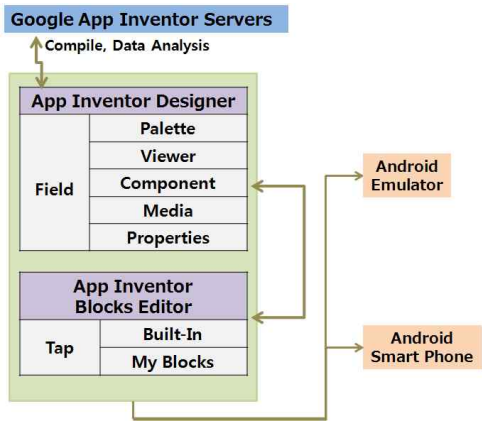


Fig. 3. Structure of App Inventor programming Environment

그림 3은 앱인벤터 프로그래밍 환경 구조를 설명한다. 앱인벤터는 플랫폼(Mac OS X, GNU/Linux, Windows OS)과 무관하게 실행될 수 있고, 앱을 제작하기 위한 시스템 환경은 자바와 크롬 브라우저에 최적화되어 있다. 앱인벤터가 실행되는 동안 지속적으로 인터넷을 통해 서버(Google App Inventor Servers)에 접속되어 있어야 한다. 앱인벤터2는 전적으로 네트워크상에서 프로그래밍 환경이 구성된다. 앱인벤터 서버는 구글에서 제공하는 것으로 프로젝트 관리와 사용자가 생성한 데이터를 받아 분석하고 결과를 만들어내는 역할을 수행한다. 앱인벤터는 프로그래밍 환경의 동작이 원격 서버를 통해서 동작하므로 앱 개발을 위해 화면에 표시할 데이터들을 서버에 업로드한 이후에 사용한다. 앱인벤터는 PC에 안드로이드 패키지(*.apk) 파일을 내려 받은 후 사용자에게 직접 공유하여야 하며, 개발한 앱에 대한 프로젝트 정보 전체가 서버에 저장된다.

앱인벤터는 Designer와 Blocks Editor로 나누어지며 필요

에 따라 Android Emulator를 동반한다. Designer는 버튼, 이미지, 소리, 동영상 등 앱에서 사용할 요소들을 배치하고 설계하는 도구로 별도의 프로그램이 아닌 앱인벤터 웹페이지에서 실행된다. 그림 4는 앱인벤터 디자이너의 화면구성을 보여준다. 그에 따른 디자이너 각 영역별 기능은 표 2와 같다.

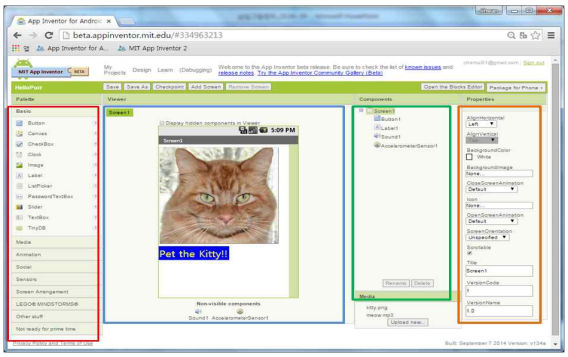

Fig. 4. Configuration of App Inventor Designer

Table 2. Configuration function of App Inventor Designer

| 영역 | 기능 | 비고 |
|---|---|---|
| Palette | 선택 가능한 컴포넌트 표시 | 도구모음 |
| Viewer | 앱을 구성하는 컴포넌트 배치 | 개발공간 |
| Component | 사용된 컴포넌트를 트리형식으로 표시<br>화면에 배치되는 구성요소. | 개발에 사용된 도구모음 |
| Media | 미디어 컴포넌트를 배치 | |
| Properties | 사용하는 컴포넌트 속성 설정 | 도구설정 |

Palette 영역에는 모든 그래픽 기능 컴포넌트가 그룹으로 묶여있다(Basic, Media, Animation, Social, Sensors, Screen Arrangement, LEGO MINDSTORMS, Other stuff 등). Viewer 영역은 스마트폰의 외관과 닮아 시각적으로 정렬돼 있는 컴포넌트를 통해 실제 스마트폰에 보일 결과 화면을 대략적으로 알 수 있게 해준다. Component 영역에서 Palette 영역으로부터 Viewer에 끌어다 놓은 모든 컴포넌트는 계층 트리구조로 나타낸다. 각각의 컴포넌트는 다른 컴포넌트의 종속 요소로 지정돼 동일한 속성이나 종속성을 갖도록 그룹을 형성한다. Properties 영역은 선택된 컴포넌트 오브젝트의 속성을 보여준다.

Blocks Editor는 그래픽 코드 블록들을 사용하여 앱 요소들의 동작을 작성하는 자바 웹스타트 애플리케이션이다. 코드 블록들을 쌓아 동작을 구성하는 과정은 일반적으로 안드로이드 개발에서 자바 프로그래밍의 영역에 해당한다. 사용자는 Blocks Editor를 통해 발생 가능한 이벤트에 원하는 동작을 연결하는 방식으로 각각의 컴포넌트를 동작시키고, 특정 작업을 할당해 앱 전체 기능을 형성한다[15]. 그림 5는 버튼을 눌렀을 때 이벤트 처리하는 블록의 집합(좌)과 그에 따라 나타나는 에뮬레이터 테스트 화면(우)을 보여준다. 프로그래밍 언어의 명령어 집합을 구현하는 방법이 블록으로 표현되어 나타나므로, 임의의 위치에 블록들을 맞출 수 있는 것이 아니며 모든 조각은 제자리가 있다.

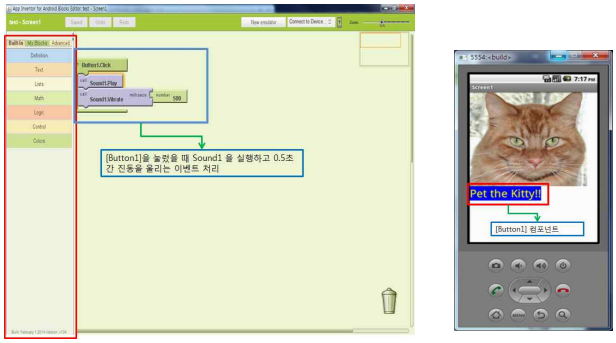

Fig. 5. Event processing of Blocks Editor and Emulator Test result

Blocks Editor의 각각의 탭은 표 3과 같다.

Table 3. Function of Blocks Edit

| 탭 | 기능 | 비고 |
|---|---|---|
| Built-In | 색, 연산자, 텍스트 등 내장된 기능을 가진 블록집합 | 기본 내장 블록 |
| My Blocks | Designer에서 추가한 컴포넌트를 보관<br>앱에서 사용하는 컴포넌트 오브젝트의 속성을 조절하는 특정 컴포넌트 블록 | Designer에서도 속성 변환 가능 |

개발한 앱은 직접 스마트폰을 연결하거나 에뮬레이터를 통해 사용해보는 것으로 앱의 UI 및 기능을 확인한다. 에뮬레이터는 가상의 스마트폰을 컴퓨터에서 실행하여 제작한 앱을 사용해보는 테스터 역할을 한다. 테스트가 끝난 앱은 패키징 과정을 거쳐 케이블로 연결된 스마트폰으로 옮기거나, 바코드 스캔, 컴퓨터로 다운로드 받아 옮기는 등의 방법으로 배포가 가능하다. 앱인벤터는 레고 마인드스톰(LEGO MINDSTORMS)을 제어하는 형태의 프로젝트 개발도 가능하다[16,17]. 또한 레고 마인드스톰에서 만든 NXT로봇을 제어할 수 있는 기능을 제공한다. NXT의 센서(NxtColorsensor, NxtDirectCommands, NxtDrive, NxtLightSensor, NxtSoundSensor, NxtTouchSensor, NxtUltrasonicSensor)들을 제어할 수 있는 각종 기능을 제공하며, 프로그래밍과 논리적 연산을 지도하는 교육적 용도로 사용된다.

### 3.3 엠비즈메이커

엠비즈메이커는 비즈니스 스마트앱 개발의 효율성을 높일 수 있도록 편의성과 사용자 중심의 UI를 제공한다. 또한 데이터베이스 처리 부분이 타 개발환경에 비해 특화되어있어 업무용 스마트앱 개발에 적합하므로, 사용자 본인이 수행중인 업무에 대해 DB 처리 등에 대한 기능을 추가하여 앱을 개발하는 것에 초점이 있다[18]. 또한 엠비즈메이커는 크로스 플랫폼 개발

도구이므로 엠비즈메이커로 제작된 앱은 m-BizEngine이 설치된 iOS, Android 등 다수의 모바일환경에서 실행가능하다[19].

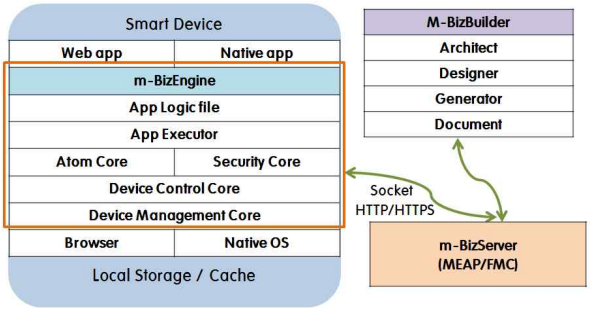


Fig. 6. Structure of m-BizMaker

그림 6은 엠비즈메이커의 구조를 설명하는 그림이다. 엠비즈메이커는 빌더(m-BizBuilder), 엔진(m-BizEngine), 서버(m-BizServer)으로 구성된다. m-BizBuilder의 역할은 Architect, Designer, Generator, Document의 영역으로 나뉘며, 영역에 따라 사용자 인터페이스(UI)의 구현, AP서버 및 DB서버를 구현 및 제반 기술 문서를 생성한다. m-BizBuilder에서 앱을 구현하는데 필요한 구성요소를 아톰(Atom)이라 표현하고, 디자인 요소와 같은 역할을 하는 동시에 프로그램 처리기능을 미리 내장하고 있어 프로그래밍을 쉽게 만든다. 이와 같은 m-BizBuilder의 구성은 그림 7을 통해 볼 수 있다.


Fig. 7. Configuration of m-BizBuilder

m-BizBuilder를 구성하는 도구의 기능은 다음의 표 4와 같이 Architect, Designer, AP서버 및 DB서버 기능을 구현하는 Generator 그리고 기술문석 작성을 위한 Document가 있는데 Designer는 UI를 구현하고, AP 서버의 비즈니스 로직 구현, Generator로 DB서버의 데이터 구조설계, DB 테이블 생성 및 데이터 입출력 개발한다. 또한 Document가 만든 제반기술 문서는 비즈니스 앱 관련 개발의 효율성을 높여준다.

m-BizEngine은 사용자들이 기존 사용하던 앱이나 콘텐츠들을 그대로 활용할 수 있도록 Android, iOS, Windows 등 여러 모바일 운영체제 위에서 실행되는 분산처리 구조의 클라이언트 측 플랫폼 형태로 설계되었다. m-BizBuilder에서 코딩작업 없이 쉽고 빠르게 앱을 구현하고 다양한 운영체제 환경, 디바이스 규격에 상관없이 배포하기 위해 패킹하고 앱과 메뉴를 하나로 묶어 표준 설치파일을 생성한다. m-BizEngine은 내부구조와 처리방식은 각 운영체제별로 다르지만, 사용자가 보는 UI방식과 처리결과는 동일하게 작동되도록 구현되어 OS별로 배포되는 것으로 엠비즈메이커로 개발한 앱이 가지는 호환성과 이기종간 통합성을 높인다.

Table 4. m-BizBuilder Tool's function

| Tool | 기능 | 비고 |
|---|---|---|
| Architect | 프로세스 분석 및 시스템 설계 | |
| Designer | 프로그램 기능 및 폼 디자인 | GUI |
| Generator | AP서버 및 DB서버 기능 구현 | 자동생성 |
| Document | 시스템 개발 기술문서 작성 | 자동생성 |

m-BizEngine은 사용자들이 기존 사용하던 앱이나 콘텐츠들을 그대로 활용할 수 있도록 Android, iOS, Windows 등 여러 모바일 운영체제 위에서 실행되는 분산처리 구조의 클라이언트 측 플랫폼 형태로 설계되었다. m-BizBuilder에서 코딩작업 없이 쉽고 빠르게 앱을 구현하고 다양한 운영체제 환경, 디바이스 규격에 상관없이 배포하기 위해 패킹하고 앱과 메뉴를 하나로 묶어 표준 설치파일을 생성한다. m-BizEngine은 내부구조와 처리방식은 각 운영체제별로 다르지만, 사용자가 보는 UI방식과 처리결과는 동일하게 작동되도록 구현되어 OS별로 배포되는 것으로 엠비즈메이커로 개발한 앱이 가지는 호환성과 이기종간 통합성을 높인다.

m-BizServer는 동일한 앱 모듈을 유선망과 연결된 Windows기반 PC에서부터, 무선망으로 접근하는 스마트폰, 태블릿PC 등 서로 다른 운영체제와 규격의 모바일기기까지 호환할 수 있는 기능을 제공한다. m-BizServer의 구성은 유무선 통합 모바일 앱 서버인 Application Server, 기존 레거시시스템과 연계하는 Gateway Server, 사용자 단말기와 시스템 보안 담당하는 MDM Server, 푸시메시지 처리해주는 Message Server, 앱을 관리해주는 AppStore Server 그리고 로그 관련 서비스를 제공하는 Log Server가 있다. 그림 8는 m-BizServer의 구성을 보여준다.

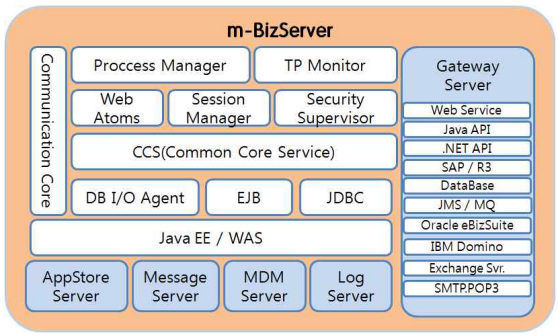


Fig. 8. Configuration of m-BizServer

Application Server는 앱의 실행에 필요한 CCS(Common Core Service)를 일괄제공하고, 스마트폰 및 태블릿 PC등 서

로 다른 OS와 다양한 규격 Device로 접속하는 조직 내·외부 불특정 다수의 동시사용자들의 데이터입출력 처리와 ERP, CRM, 전자결재, 그룹웨어 등의 레거시시스템 기능 연계처리 등의 서비스를 제공하는 모바일 앱 서버 플랫폼이다. 서버장치, 네트워크, 디바이스, 어플리케이션까지 다계층을 포괄한 End-to-End 보안 및 앱을 Web App(HTML5)형태로 변환하여 배포하는 기능을 갖는다.

Application Server에서는 트랜잭션 관리 미들웨어로 TP Monitor(Transaction Processing Monitor)를 사용하여 프로토콜에서 동작하는 세션과 시스템 및 데이터베이스 사이의 최소 처리 단위인 트랜잭션을 감시하여 일관성 있게 보관 및 유지하는 것으로 개방형 분산처리 시스템의 문제점을 극복하고 안정성과 무결성을 보장한다. TP-Monitor는 미들웨어에 비즈니스 로직을 구현하고 클라이언트 프로그램은 사용자에게 제공하는 모듈만 구현하고, 데이터베이스 서버 프로그램은 데이터 관리에 대한 기능만 구현하면 되도록 기능을 분리해 앱 개발을 쉽게 한다. 또한 DBMS(database management system)의 트랜잭션을 통합 관리하여 이기종 데이터베이스 간의 자원 관리가 가능하며, 적은 서버 자원으로 많은 클라이언트를 관리하여 DBMS의 오버헤드를 줄이고 응답시간을 향상시킨다.

MDM Server는 사용자 단말기의 등록과 접속 통제, 네트워크 및 주변장치 사용제어, 앱과 데이터 접근제어 등과 같은 시스템 보안 정책을 수립하고 통제하는 역할을 한다. 다수의 사용자가 접근 및 사용 가능하도록 개방해주는 B2C(business to customer) 환경을 제공하며, 반대로 비인가 단말기 접근 차단, 사용자 등급에 따른 접근, 사용을 제어할 수 있다.

Message Server는 메신저서비스, 시스템관리자 전송 공지사항, MDM관리자의 단말기 원격 제어 등의 메시지 푸시 서비스(Message Push Service)를 제공한다. 필요에 따라 서버 측의 요구에 의해 지정 단말기로 메시지를 전환, 처리할 필요가 있는 경우 사용된다. AppStore Server는 다양한 앱과 컨텐츠를 관리하고 서비스하는 서버이며, Log Server는 모든 로그정보를 수집·기록하고 관리, 분석, 추적하는 기능을 제공한다.

### 3.4 스크래치, 앱인벤터 및 엠비즈메이커의 기능비교

앞 절에서 3가지 비주얼프로그래밍환경에 대해 알아보았다. 그를 바탕으로 각 비주얼환경별 특징을 정리하면 다음과 같다.

스크래치는 독자적으로 프로세스가 구동되어 스크래치 그 자체로 개발이 가능한 비주얼프로그래밍환경이다. 프로그래밍 환경 내에 stage라는 테스트 도구가 탑재되어 개발 프로젝트의 실행이 즉시 확인 가능하며, 별도의 데이터베이스 시스템을 갖고 있지는 않다. 프로그래밍 환경에 인터프리터(interpretor)가 내장되어 있어 별도의 컴파일 과정이 필요하지 않고, 프로젝트 파일(*.sb)은 스크래치 환경에서만 실행가능하다. 따라서 스크래치에서 개발된 프로젝트는 해당 파일이 만들어진 스크래치 비주얼프로그래밍환경이나, 웹 인터페이스에서만 실행된다.

앱인벤터는 실행되는 동안 지속적으로 네트워크를 통해 구글 앱인벤터 서버에 접속해야한다. 접속하여 프로젝트 관리, 데이터 분석 및 결과 생성 등의 개발이 진행된다. 앱인벤터는 Designer, Blocks Editor로 명령어 집합을 구성하여 사용자 의도에 가깝게 프로그래밍하며 프로젝트를 테스트 및 결과 확인을 위해 타겟 디바이스에 직접 연결 또는 에뮬레이터를 활용할 수 있다. 데이터베이스 시스템은 타겟 디바이스에 탑재되는 TinyDB와 앱인벤터 서버에 탑재되어 있는 TinyWebDB로 구성되며, 그 용량이 극히 한정적이어서 상용으로 사용하는 것 보다 테스트용으로 적합하다. 앱인벤터로 개발한 프로젝트는 앱인벤터 서버를 통해 컴파일 되며, 그 형태는 안드로이드용 실행파일(*.apk)이다. 따라서 앱인벤터로 개발된 실행파일의 타겟 플랫폼은 안드로이드이고, 앱인벤터 또한 오픈 프로젝트이므로 개발자 커뮤니티가 개발자 포럼을 주축으로 매우 활발하다.

엠비즈메이커는 데이터베이스 서버에 접속하는 경우를 제외하고 독자적으로 개발이 가능하며, 엠비즈빌더를 이용해 개발한다. m-BizBuilder는 앱인벤터의 디자이너와 같이 사용자 인터페이스(UI)만을 구현하지만 실제적으로는 UI에 따른 명령어 집합과 데이터베이스 등이 구성되고, 이는 preview를 통해 확인할 수 있다. 컴파일은 서버를 통해 이루어지며, m-BizEngine이 설치된 시스템에 최적화 되어 생성된 실행파일은 앱 기반의 *.qpm과 웹 기반의 *.qwp 형태가 있다. 엠비즈메이커는 m-BizEngine이 설치된 iOS와 안드로이드 환경 등에서 실행파일이 실행되는 크로스플랫폼 비주얼프로그래밍환경이다.

표 5는 스크래치, 앱인벤터, 엠비즈메이커를 항목별로 비교한 것이다. 각 프로그래밍환경이 web과의 연결 없이 독자적 프로세스가 구동되어 개발 가능한지의 여부는 앱인벤터를 제외한 두 개의 프로그래밍 환경 모두 가능하다. 앱인벤터는 실행되는 동안 지속적으로 서버에 접속되어 있어야하므로 독자적 프로세스 구동은 불가능하다. 환경별 사용자 제어 프로그래밍 작업창을 보면 앱인벤터의 경우 Designer, Blocks Editor, 엠비즈메이커의 경우 m-BizBuilder로 구성되어 있으며, 스크래치는 별도의 구성없이 스크래치 자체가 사용자 제어 프로그래밍 작업창이 된다. 앱인벤터의 프로그래밍 작업창을 Designer와 Blocks Editor로 나눈 것은 m-BizBuilder에서 UI를 구성하는 것으로 사용자의 제어 범위가 제한되는 것과는 달리 Blocks Editor라는 명령어 집합 구현 도구를 별도로 두어 프로그램의 확장성을 높이는 장점이 있다.

개발 프로젝트 테스트 도구는 스크래치, 앱인벤터, 엠비즈메이커 모두 각각 stage, emulator, preview를 갖고 있다. 각 환경은 스크래치를 제외하고 모두 데이터베이스 시스템을 가지고 있으며, 앱인벤터의 경우 스마트폰 내부에 저장되는 Tiny DB와 서버에 저장되는 TinyWebDB로 구분된다. 앱인벤터 데이터베이스 시스템의 경우 DB에 저장할 수 있는 자료의 용량이 매우 제한적이다.

Table 5. Function comparison

| function / Item | Scratch | App Inventor | m-BizMaker |
|---|---|---|---|
| Stand alone | Stand alone | with Appinventor server | Stand alone |
| Programming | Scratch | Designer/Blocks Editor | M-BizBuilder |
| Emulator | Stage | Emulator | Preview |
| DB system | None | Tiny DB TinyWebDB | DB server |
| Compile | interpreter | Appinventor server | server/m-BizEngine |
| Target Platform | Web(Scratch) | Android | Cross-platform |
| Target machine Pre-installed software | None | None | m-BizEngine |
| Developer Community | Discuss Scratch | Developer forum | No |

개발 프로젝트가 컴파일 되는 위치는 스크래치의 경우 인터프리터가 내장되어 있어 별도의 컴파일 과정을 거치지 않아도 스크래치 환경에서 실행되며, 앱인벤터와 엠비즈메이커는 서버에서 컴파일 되어 실행파일이 만들어진다. 특히 앱인벤터의 경우 서버가 프로젝트와 데이터를 받아 분석하고 컴파일과정을 거쳐 실행파일을 만들어 낸다. 그로부터 얻을 수 있는 실행파일 형태는 각각 표 5와 같으며 스크래치 실행파일은 스크래치 환경에서만 실행될 수 있고, 앱인벤터는 안드로이드 플랫폼에 맞게 패키징(packaging)되어 안드로이드 플랫폼을 가진 디바이스와 프로그래밍 환경으로 공유 및 실행 가능하다. 엠비즈메이커는 크로스 플랫폼 환경이므로 만들어진 실행파일은 m-BizEngine이 설치된 iOS와 Android 플랫폼 등에서 실행된다. 따라서 엠비즈 메이커의 경우 타겟 디바이스에 실행파일이 설치되려면 먼저 m-BizEngine이 설치되어야 한다. 프로그래밍 환경에 대한 비용은 기업용 엠비즈메이커를 제외하고 모두 무료이며, 스크래치와 앱인벤터의 경우 오픈소스의 특성에 따라 개발자 커뮤니티가 활성화 되어있다.

## IV. Conclusions

본 논문에서는 스마트 기기에서 앱 개발을 원하는 NPE들에게 쉽고 편리하게 앱을 개발할 수 있는 대표적 3가지 비주얼프로그래밍환경에 대해 알아보고 비교·분석하였다. 스크래치는 프로그래밍에 대한 경험을 쌓기 위한 목적으로 개발된 교육용 비주얼프로그래밍환경이며, 쉽고 다양하게 상호작용이 가능한 스토리, 게임, 애니메이션 등을 만들 수 있다. 스크래치 프로젝트는 스프라이트(sprite)라는 객체로 만들어지며 스크립트를 통해 스프라이트의 동작을 구성하는 것으로 프로그래밍 한다. 개발된 프로젝트는 스크래치 비주얼프로그래밍환경이나, 웹 인터페이스에서만 실행된다는 제한점을 가진다.

앱인벤터는 스크래치를 기반으로 한 이벤트기반 비주얼프로그래밍환경이다. 각종 센서 및 GPS 등 스마트 디바이스의 다양한 기능을 제어할 수 있다는 점에서 스크래치에 비해 정교한 초기 프로토타입의 앱을 좀 더 전문적이고 빠르게 만들 수 있다. 앱인벤터는 Designer와 Blocks Editor로 나누어지며 플랫폼과 무관하게 실행될 수 있다. 앱인벤터가 실행되는 동안 지속적으로 인터넷을 통해 구글 인벤터서버에 접속되어 있어야하고, 프로그램의 용량이 제한받는다는 한계점을 가진다.

엠비즈메이커는 데이터베이스 처리 부분이 특화되어있어 업무용 앱 개발에 적합하고, 사용자 중심의 UI를 제공한다. 앱인벤터의 디자이너와 같이 사용자 인터페이스(UI)만을 구현하는 것으로 프로그램을 개발할 수 있다. 빌더(m-BizBuilder), 엔진(m-BizEngine), 서버(m-BizServer) 등으로 구성되며, m-BizEngine이 설치된 iOS와 안드로이드 환경 등에서 실행파일이 실행되는 크로스플랫폼 비주얼프로그래밍환경이다.

세 비주얼프로그래밍환경 모두 범위와 영역의 차이는 있지만 기능이 모듈 단위로 제공되어있어 쉽고 빠르게 프로그래밍 할 수 있다. 또한 교육용 프로그래밍 툴인 스크래치와 스크래치를 기반으로 만들어진 앱인벤터는 프로그래밍 교육을 위한 활용 및 포럼을 통해 프로젝트의 유기적 개선이 가능하다.

앱인벤터는 디바이스의 다양한 기능을 제어해서 프로그래밍을 정교하게 할 수 있다는 점에서 본 논문에서 살펴본 비주얼프로그래밍환경 중 NPE들에게 가장 적합한 환경임을 알 수 있다. 하지만 완벽한 객체지향형 프로그램을 만드는 것의 어려움, 프로그램의 용량에 대한 제한 등 앱인벤터가 갖는 한계점이 있으므로 앞으로 추가적인 앱인벤터에 대한 연구가 필요하다.

## REFERENCES

[1] Computing Everywhere http://news.inews24.com

[2] M. Takatsuka & M. Gahegan, “GeoVISTA Studio a codeless visual programming environment for geoscientific data analysis and visualization.”, Computers & Geosciences, Vol. 28, No.10, pp. 1131-1144, December 2002.

[3] Byungho Kim, “Computer Programming Education using App Inventor for Android”, Journal of the Korea Institute of Information and Communication

Engineering, Vol. 17, No. 2, pp.467-472, February 2013.

[4]"Google, Android Studio 1.0 Open", http://www.zdnet.co.kr/news/news_view.asp?artice_id=20141209083838&type=det&re=

[5] Chung Seong Jang, H. K. Rim and H. H. Choi, "Development of a Web-based interactive Education", Journal of The Korea Society of computer and Information, Vol. 19, No. 12, pp. 117~185, December 2014.

[6] Young Hoo Sung, "A Design of Smart-based collaborative learning model for programming education of elementary school students", Journal of The Korea Society of computer and Information, Vol. 20, No. 4, pp. 147~159, April 2015.

[7] S. Papadakis, M. Kalogiannakis, V. Orfanakis & N. Zaranis, "Novice Programming Environments. Scratch & App Inventor: a first comparison", IEEE '14: Proceedings of the 2014 Workshop on Interaction Design in Educational Environments, 2014.

[8] Ju Yeon Park, Seong Mo Park, "Android App Development System Using Modular Method", Journal of Korea Multimedia Society. Vol. 17, No. 5, pp.601~612, May 2014.

[9] A. Soares, "Reflections on Teaching App Inventor for Non-Beginner Programmers: Issues, Challenges and Opportunities", Information Systems Education Journal, Vol.12, No.4, 2014.

[10] D. Wolber, "App inventor and real-world motivation." Proceedings of the 42nd ACM technical symposium on Computer science education. ACM, pp,601-606, 2011.

[11] http://scratch.mit.edu/

[12] M. Resnick, et al. "Scratch: programming for all." Communications of the ACM Vol.52. No.11, pp. 60-67, 2009.

[13] S.C. Pokress & J.J.D. Veiga. "MIT App Inventor: Enabling personal mobile computing." arXiv preprint arXiv:1310.2830 , 2013.

[14] K. OWEN, "Android in Education Thesis Related Work (Exploration). 2011.

[15] F. Turbak, M. Sherman and F. Martin, D. Wolber & S.C. Pokress, "Events-First Programming In App Inventor", JCSC, Vol.29, No.6, 2014.

[16] S. Uludag, M. Karakus, and S.W. Turner. "Implementing IT0/CS0 with scratch, app inventor for android, and lego mindstorms." Proceedings of the 2011 conference on Information technology education. ACM, pp. 183-190, 2011.

[17] http://appinventor.mit.edu/

[18] App Technology and vision in Age of Mobile. http://www.mbizmaker.com/ups/mbizmaker_b2c/images/mobileapp.pdf. 2013

[19] Hyun Seung Son, Woo Yeal Kim and Young Chul Kim, "Development of Tool based on Model Transaction for Heterogenous Smart phone App", Korea Computer Congress 2013, pp. 536-538, June 2013.

## Authors

JiWan Kim received the B.S.degree in Computer Science and Engineering from Yeungnam University, Korea, in 2005. She received M.S. degree from Halla University, korea, in 2015.

She is interested in App development environment and mobile computing

Hyun Gon Seo received the B.S., M.S. degrees from KyungSung University, Korea in 1992, 1994, respectively. He received Ph.D. degree in Computer Science and Engineering from Yeungnam University, Korea, in 2004.

He was a BK21 guest professor at Daegu University, Korea from 2001 to 2003. Since 2004, He has been a guest professor at Yeungnam University. He is currently a Professor in the School of Information Communication and Broadcasting, Halla University. He is interested in IoT(internet of things), mobile computing, network protocols.