Journal of The Korean Association
of Information Education Vol. 21, No. 5,
October 2017, pp. 557-565

http://dx.doi.org/10.14352/jkaie.2017.21.5.557

시각화방법을 이용한 객체지향프로그래밍 교육에 관한 연구

신우창
서경대학교 컴퓨터과학과

요 약

4차 산업혁명시대를 맞이하여 학생들에 대한 프로그래밍 교육이 더욱 중요시되고 있다. 그러나, 학생들이 프로그래밍 능력을 습득하는 데에는 많은 시간과 연습이 필요하다. 특히, 산업현장에서 널리 사용되고 있는 JAVA, C++와 같은 객체지향언어를 학습하는데 있어서 학생들은 더욱 많은 어려움을 느낀다. 본 논문에서는 객체지향프로그래밍 개념을 교육하고, 기능을 이해하며, 소스코드 분석 및 이해향상에 도움을 줄 수 있는 객체 상호작용 시각화 방법을 제안한다. 제안된 시각화 방법은 기존 소스코드를 자동적으로 변경하여 프로그램 실행과 동시에 객체들이 동작하는 모습을 시각적으로 보인다.

키워드 : 프로그래밍 교육, 시각화, 객체 상호작용, 지원 도구, 코드 생성

A Study on Object-Oriented Programming Education using Visualization Method

Woochang Shin
Dept. of Computer Science, Seokyeong University

ABSTRACT

In the era of the Fourth Industrial Revolution, programming education is becoming more important. However, it takes a lot of time and practice for students to acquire programming skills. In particular, students find it more difficult to learn object-oriented languages such as JAVA and C++, which are widely used in the industrial field. In this paper, we propose a visualization method of object interaction that can help to educate the concept of object-oriented programming, understand functions, and improve source code analysis and understanding. The proposed visualization method automatically changes the existing source code and visualizes the operation of the objects simultaneously with the execution of the program.

Keywords : Programming Education, Visualization, Object Interaction, Support Tools, Code Generation

본 연구는 2017학년도 서경대학교 교내연구비 지원에 의하여 이루어졌음.
논문투고 : 2017-09-05
논문심사 : 2017-09-14
심사완료 : 2017-09-30

1. 서론

인공지능, 사물인터넷, 클라우드 컴퓨팅, 빅데이터, 모바일 등의 주요 기술로 대표되는 4차 산업혁명시대를 맞이하여 학생들에 대한 프로그래밍 교육이 더욱 중요시되고 있다. 싱가포르 정보통신개발청은 2014년 4월부터 아이들에게 코딩을 가르쳐 논리적 사고력을 기르고, 향후 아이들이 소프트웨어 전문가로 성장할 수 있는 기반을 마련하기 위한 'CODE@SG' 프로젝트를 시행하고 있으며, 영국, 일본, 이스라엘 미국 등에서도 초·중등 및 고등교육 단계에서 코딩교육을 시행하고 있다. 우리나라도 세계적인 흐름에 맞추어 2018년부터 전국 초등 및 중등 교육과정에 의무교육으로 시행될 예정이다[7][8][10][16].

학생들이 프로그래밍 능력을 습득하는 데에는 많은 시간과 연습이 필요하다. 일반적으로 대학전공자의 경우 1학년에 C언어를 학습하며, 이를 바탕으로 객체지향 언어인 C++나 JAVA언어를 학습한다. C언어와 같은 절차 지향적 언어는 프로그램의 실행 흐름이 직관적이며, 프로그램의 주된 두 가지 구성요소인 함수와 변수를 이해한다면 프로그램을 작성하는데 큰 어려움이 없다. 이에 비해, JAVA나 C++언어와 같은 객체지향적 언어는 프로그램의 실행흐름이 직관적이지 않으며, 프로그램을 구성하는 객체와 캡슐화(capsulation), 상속성(inheritance), 다형성(polymorphism)과 같은 고유 개념들을 학습하고, 객체들 간의 상호작용으로 프로그램이 수행된다는 점을 이해하여야 된다는 점에서 학생들은 더욱 많은 어려움을 느낀다.

본 논문에서는 객체지향프로그래밍 개념을 교육하고, 기능을 이해하며, 소스코드 분석 및 이해 향상에 도움을 줄 수 있는 객체 상호작용 시각화 방법을 제안한다. 제안된 시각화 방법은 기존 소스코드를 자동적으로 변경하여 프로그램 실행과 동시에 객체들이 동작하는 모습을 시각적으로 보인다.

2장에서는 관련연구를 살펴보고, 3장에서는 시각화를 위한 시스템의 모델에 대하여 기술한다. 4장에서는 시각화 시스템의 구현에 대하여 설명하고, 5장에서는 학생들을 대상으로 시각화를 통한 교육효과를 검증한다. 6장에서 평가로 끝을 맺는다.

2. 관련 연구

객체지향 프로그래밍 교육에 관련하여 많은 연구들이 진행되어왔다.

먼저, 기존의 교육용 프로그래밍 언어를 활용하거나, 초보자를 위한 객체지향 프로그래밍 언어를 개발하는 연구가 있다. [2]는 교육용 시각 프로그래밍 언어인 Alice를 이용하여 학생들에게 효율적으로 객체지향 개념들을 교육시킬 수 있는 시각 프로그래밍 교육방법론과 교과과정을 제시하였다. 또한 몇몇 연구에서는 한글 객체지향 프로그래밍 언어를 개발하고 이를 프로그래밍 교육에 적용함으로써 초보자의 영어에 대한 부담을 들어주는 접근 방법이 연구되었다[5][8][9]. 그러나 산업현장에서 사용되지 않는 교육용 객체지향 언어를 별도의 시간을 들여 학습한다는 것은 현실적으로 대학교육 현장에서 어려움이 있다.

두 번째는 다양한 시각화기법을 도입하여 객체지향 프로그래밍 교육에 활용하는 연구이다. 프로그래밍 교육에서 어려운 분야중 하나인 알고리즘 교육에 있어서 시각화기법을 도입하여 학습의 효율성을 높이는 연구가 진행되었으며[4][6][10][14], 퍼즐화된 이미지를 활용하여 학생들의 학습동기를 높여주는 연구[17]와 정적/동적 프로그램 시각화 도구가 학생들의 프로그램 동작의 이해도를 높이고, 프로그래밍 스킬을 증진하는데 얼마만큼의 도움이 되는지를 측정하는 연구가 진행되었다[1][11][13].

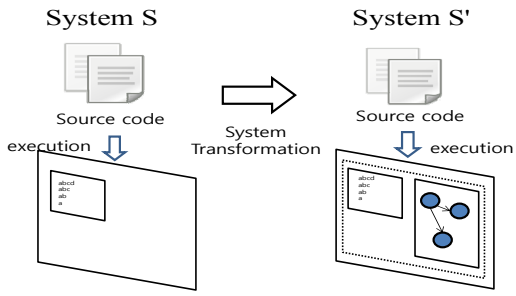
세 번째는 시각적인 프로그래밍 환경을 제공하여 초보자들도 개념적인 객체의 관계를 시각적으로 볼 수 있으며, 이를 통하여 객체지향 프로그램을 자동으로 생성하는 연구가 있다[3][15].

기존에 많은 연구들이 존재하지만, 본 연구는 객체지향 프로그램의 내부 동작 모습을 실시간으로 보여주기 위한 정형화된 시각화 모델을 제시하고, 이를 구현하며, 시험을 통하여 프로그램 이해에 관한 교육효과를 검증하는데 차별성이 있다.

3. 시각화를 위한 시스템 모델

본 연구에서 제시하는 객체 상호작용 시각화방법은 (Fig. 1)에 보이는 바와 같이 소프트웨어 시스템 s의 소

스코드를 자동 변환하여 시각화 기능이 추가된 소프트웨어 시스템 s′를 생성한다.



(Fig. 1) System Transformation for Visualization

시각화를 위한 시스템 변환과정은 기존의 기능 동작은 그대로 유지하면서 객체들의 상호작용을 실시간으로 화면에 보여주기 위하여, 시각화 모듈이 추가되며 기존 소스코드의 일부가 수정된다.

시각화를 위한 시스템 변환과정은 프로그래밍 언어의 문법(syntax)와 의미(semantic)에 따라 달라진다. 본 연구에서는 JAVA 언어를 대상으로 하여 시스템 모델을 설정하고, 시각화 시스템을 구축한다. 시스템 변환과정을 정형화하기 위하여 객체지향 시스템 모델 SYS를 다음과 같이 정의한다.

[정의 1] 객체지향 시스템모델 SYS는 다음의 요소들로 구성된 구조체(structure)이다.

SYS = (CLA, ≪)

- CLA : a set of classes
- ≪ : a partial order to represent inheritance relation between classes
  ≪ = { <c1, c2> | c1∈CLA, c2∈CLA, c1 inherits c2 }

[정의 2] 객체지향 시스템에 포함된 클래스 C는 다음의 요소들로 구성된 튜플(tuple)이다.

C = <cname, ATTR, FUNC, CONS>

- cname : class name
  - cname ∈ ID
- ATTR : a set of tuple <name, sort> that represent attributes
  - name ∈ ID
  - sort ∈ SORTExpr
- FUNC : a set of tuple <name, sig, body> that represent member functions
  name ∈ ID
  - sig = <s1x...xsn,s>, s1,..,sn,s∈SORTExpr
  - body = <st1,...,stn>, st1,..,stn∈ STA
- CONS : a set of tuple <sig, body> that represent object constructor
- ID : a set of all identifiers
- SORTExpr : a set of all sort expressions
  - {boolean, byte, short, int, long, float, double, char} ∪ CLA ∪ ENUM ⊂ SORTExpr
  - if s∈SORTExpr then array(s)∈SORTExpr
  - ENUM is a set of all enumeration type
- STA : a set of all program statements

앞에서 정의한 시스템 모델을 이용하여 시각화를 위한 시스템 변환과정은 다음 함수 TSYS로 정의할 수 있다.

[정의 3] 시스템 s를 시각화 시스템 s′로 변환하는 시스템 시각화 변환 함수 TSYS는 다음과 같이 정의한다.

TSYS(s) : SYS → SYS

- TSYS(s) = (CLA′, ≪′) where s=(CLA, ≪)
- CLA′ = { TCLA(c) | c∈CLA } ∪ { v }
  - v : visualization class
- ≪′ = { <TCLA(a), TCLA(b)> | <a, b>∈≪ }
- TCLA : 클래스 시각화 변환 함수

[정의 4] 클래스 c를 시각화 클래스 c′로 변환하는 클래스 시각화 변환 함수 TCLA는 다음과 같이 정의한다.

TCLA(c) : CLA → CLA

- TCLA(c) = <cname, ATTR, FUNC′, CONS′> where c=<cname, ATTR, FUNC, CONS>
- FUNC′ = { func′ | func∈FUNC, func′= TFUNC(func) }
- CONS′ = { cons′ | cons∈CONS, cons′= TCONS(cons) }
  - if CONS = {} then, CONS′ = { <<>, <cons_entry_st>> }

- cons_entry_st : 생성자 함수 호출이 일어났음을 시각화 모듈에게 알려주는 statement

시스템 시각화 변환과정에서 시각화 기능이 추가되기 위해서는 최종적으로 기능을 구현하는 멤버 함수의 코드가 수정되어야 한다. 멤버함수와 생성자함수에 시각화 기능을 추가하기 위한 변환 함수는 다음과 같다.

[정의 5] 멤버함수 f를 시각화 멤버함수 f′로 변환하는 멤버함수 시각화 변환 함수 TFUNC는 다음과 같이 정의한다.

TFUNC(f) : FUNC → FUNC

- TFUNC(f) = <name, sig, statements′> where f=<name, sig, statements>
- name : function name
- sig : method signature
- statements : a sequence of statements,
  - <st1,...,stn>
- statements′ : a sequence of statements
  - if sti (1<i<n) 가 return 문장이면, sti 앞에 exit_st 삽입. <entry_st, st1,...,sti-1, exit_st, sti,..., stn>
  - stn이 return 문장이 아니면 마지막에 exit_st 삽입. <entry_st, st1,...,stn, exit_st>
  - entry_st : 멤버함수 호출이 일어났음을 시각화 모듈에게 알려주는 statement
  - exit_st : 멤버함수가 반환됨을 시각화 모듈에게 알려주는 statement

[정의 6] 생성자함수 cons 를 시각화 생성자함수 cons′로 변환하는 생성자함수 시각화 변환 함수 TCONS는 다음과 같이 정의한다.

TCONS(cons) : FUNC → FUNC

- TCONS(cons) = <sig, <cons_entry_st, st1,...,stn>> where cons=<sig, <st1,...,stn>>
  - name : function name
  - sig : method signature
  - st1,...,stn : programming statements
  - cons_entry_st : 생성자 함수 호출이 일어났음을 시각화 모듈에게 알려주는 statement

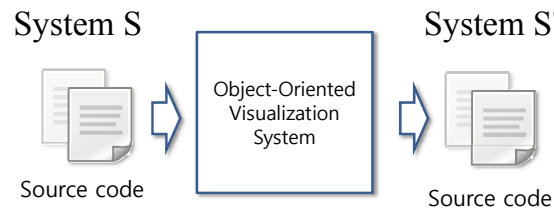
앞의 정의에서 기존 소스코드에서 시각화를 위하여 모든 멤버함수의 첫 명령문으로 entry_st를 추가하고, 마지막 명령문으로 exit_st를 추가한다. 생성자함수에 대해서는 첫 명령문으로 cons_entry_st을 추가한다. entry_st, exit_st, cons_entry_st 명령문은 [정의 3]에서 기술된 visualization 객체 v의 멤버함수를 호출하는 것으로, 시각화를 위한 시각과 정보를 런타임(runtime)에 v에 전달한다.

4. 시각화 시스템 구현

본 연구에서 구현된 시각화 시스템의 개발환경은 다음과 같다.

- Java Development Kit 1.8
- Eclipse Neon 2
- Jung Graph library 2.1.1
- JavaParser 3.3.3
  (https://github.com/javaparser)
- 샘플코드: bwscalc java code
  (https://github.com/brian-w-smith/bwscalc)

객체상호작용 시각화 시스템은 (Fig. 2)에 보이는 바와 같이 자바 소스 파일들을 입력으로 받아, 시각화를 지원하는 소스 파일들을 생성한다.



(Fig. 2) Input & Output of Visualization System

샘플코드는 Github에 등록된 계산기 프로그램을 사용하였다.

시각화 시스템은 앞장에서 정의한 시스템 시각화 변환 함수 TSYS를 구현한 것이다. TSYS의 입력인 시스템 s와 변환되는 시스템 s′를 [정의 3]에 따라 기술하

면 다음과 같다.

TSYS(s) = s′
s = (CLA, ≪)
CLA = {*Main, Parser, SuccessParser, Tokenizable, BasicTokenizable, Calculator, IntToken, MinusToken, OpToken, PlusToken, Token, AdditionExpression, ...*}
≪ = {<*IntToken, Token*>, <*MinusToken, OpToken*>, <*OpToken, Token*>, <*PlusToken, OpToken*>, ...}
s′ = (CLA′, ≪′)
CLA′ = {*ObjectVisualization, Main′, Parser′, SuccessParser′, Tokenizable′, BasicTokenizable′, Calculator′, IntToken′, MinusToken′, OpToken′, PlusToken′, Token′, AdditionExpression′*, ...}
≪′ = {<*IntToken′, Token′*>, <*MinusToken′, OpToken′*>, <*OpToken′, Token′*>, <*PlusToken′, OpToken′*>, ...}

CLA에 기술된 클래스들은 bwscalc 계산기 프로그램에 있는 클래스들인 반면, CLA′에 기술된 클래스들은 기존 클래스들이 클래스 시각화 변환 함수 TCLA를 통하여 변환된 클래스들이다. ObjectVisualization 클래스는 [정의 3]에서 기술된 visualization class v의 구현체이다.

클래스 SuccessParser를 시각화 클래스 SuccessParser′로 변환하는 클래스 시각화 변환 함수 TCLA를 [정의 4]에 따라 기술하면 다음과 같다.

TCLA(*SuccessParser*) = *SuccessParser′*
SuccessParser = <"SuccessParser", ATTR, FUNC, CONS>
ATTR = { }
FUNC = { *parser, parserExpression, parseBinaryExpression, parseTerm, parseValueExpression,* ... }
CONS = { *SuccessParser.SuccessParser* }
SuccessParser′ = <"*SuccessParser*", ATTR, FUNC′, CONS′>
FUNC′ = { *parser′, parserExpression′, parseBinaryExpression′, parseTerm′, parseValueExpression′*, ... }
CONS′ = { }

클래스 SuccessParser의 멤버함수들 중에서 파싱(parsing)을 수행하는 parse() 함수를 시각화 변환 함수 TFUNC를 통하여 변환하면 다음과 같다.

TFUNC(*parse*) = *parse′*
parse = <"parse", sig, statements >
sig = <*Expression, List*<*Token*>>
statements = <st1,...,stn>
- s1= "ListIterator<Token> tokenIter = tokens.listIterator();"
- s2 = "Expression expr = parseExpression(tokenIter);"
- stn = "return expr";
parse′ = <"parse", sig, statements′>
statements′ = <entry_st, st1,...,stn-1, exit_st, stn>
- stn is a “return“ statement
- entry_st = “ObjectVisualization.enterMethod(this, "parse");”
- exit_st = “ObjectVisualization.exitMethod();”

변환되기 전의 parse함수와 시각화 변환된 후의 parse′함수는 다음과 같다.

```
public Expression parse(List<Token> tokens) {
  ListIterator<Token> tokenIter = tokens.listIterator();
  Expression expr = parseExpression(tokenIter);
  if (tokenIter.hasNext()) {
    throw new ParseException("Extra text after expression: "
+ tokenIter.next().getValue());
  }
  return expr;
}
```

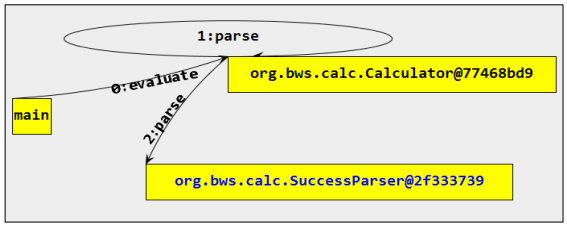
(a) Before transformation - SuccessParser.parse()

```
public Expression parse(List<Token> tokens) {
   // auto-generated for visualization
   ObjectVisualization.enterMethod(this, "parse");

   ListIterator<Token> tokenIter = tokens.listIterator();
   Expression expr = parseExpression(tokenIter);
   if (tokenIter.hasNext()) {
      throw new ParseException("Extra text after expression:
" + tokenIter.next().getValue());
   }
   // auto-generated for visualization
   ObjectVisualization.exitMethod();

   return expr;
}
```

(b) After tranformation - SuccessParser.parse()

(Fig. 3) parse() code before and after Visualization Transformation

(Fig. 3)의 시각화 변환된 parse() 함수의 코드에 보이듯이 함수가 시작되기 전에 시각화 클래스인 ObjectVisualization에 현재 호출되는 함수의 정보를 넘겨주며, 함수가 반환되기 전에 함수 실행 종료를 ObjectVisualization에게 알려준다. ObjectVisualization은 함수 시작과 종료에 대한 정보를 받아서, 이를 시각화 윈도우 화면에 실시간으로 그림으로 표현한다.



(Fig. 4) Visualization of SuccessParser.parse() call

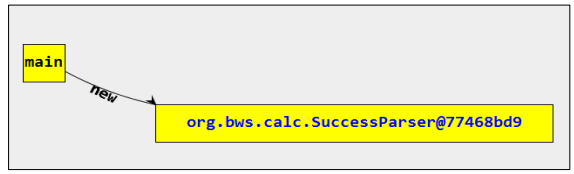
(Fig. 4)는 main() 함수가 Calculator 객체의 evaluate() 함수를 호출하고("0:evaluate"), 이어서 Calculator객체가 자신의 parse() 함수를 호출하며("1:parse"), Calculator.parse() 함수가 SuccessParser 객체의 parse()함수를 호출하는 ("2:parse") 것을 프로그램 실행과 함께 실시간으로 보여주는 시각화 화면이다.

클래스 SuccessParser는 생성자함수가 없다. 즉, SuccessParser의 CONS는 공집합이며 [정의 4]에 의하여 CONS′= { <<>, <cons_entry_st>> } 이다. 이것은 cons_entry_st 명령문만 포함하는 기본(default) 생성자 함수가 새로 추가됨을 의미한다. (Fig. 5)와 같이 SuccessParser는 시각화 변환된 후 기본 생성자함수가 새로 생성된다.

```
public class SuccessParser implements Parser {
   // auto-generated for visualization
   public SuccessParser() {
      ObjectVisualization.newObj(this);
   }
   ...
```

(Fig. 5) Newly created constructor of SuccessParser after visualization transformation

SuccessParser 객체가 main()함수에 의하여 생성될 때 시각화 윈도우 화면에 보이는 것은 다음 (Fig. 6)과 같다.

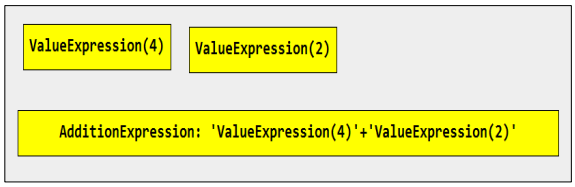


(Fig. 6) Visualization screen when SuccessParser object is created

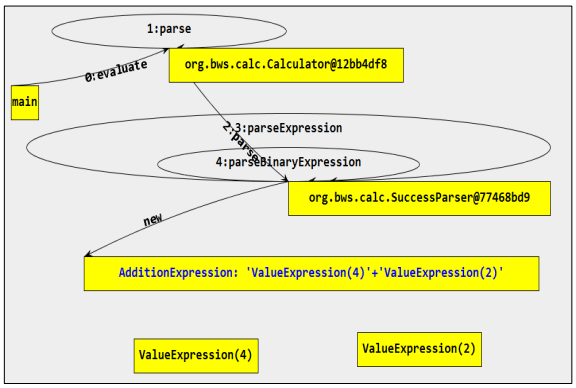
객체가 시각화 윈도우 화면에 표현될 때 그 이름은 기본적으로 "클래스명@참조주소" 형태이다. 이 경우 객체를 상호 구분할 수는 있으나, 그 의미를 파악하는 데에는 어려움이 있다.

시각화 윈도우 화면에서 객체의 이름은 Object.toString() 함수를 이용하여 찾는다. 즉, toString() 함수를 오버라이딩(Overriding)함으로써 객체의 이름을 이해하기 쉬운 문자열로 작성할 수 있다. (Fig. 7)에 보이는 바와 같이 ValueExpression 객체 두 개가 있을 경우, 이들 객체의 이름을 ValueExpression@ 77468bb0, ValueExpression@77468bc0으로 표현하는 것보다는 해당 객체의 값으로 표현하는 것이 그 의미를 파악하는데 도움이 된다. 또한 두 개의 수식

(Expression)을 더하는 표현식인 AdditionExpression의 경우에도 AdditionExpression@@77468be0 형태로 표현하는 것보다는 AdditionExpression.toString()을 오버라이딩하여 덧셈에 사용되는 두 수식의 종류와 값이 화면에 표현되도록 하는 것이 더욱 이해하는데 도움이 된다.



(Fig. 7) Object name change using toString() overriding



(Fig. 8) Visualization screen when running the calculator program

(Fig. 8)은 계산기 프로그램(bwscalc)에서 수식 "4+2-2"을 계산하는 동안, 객체들 간에 발생하는 상호작용을 시각화 윈도우 화면으로 실행과 동시에 보여주는 화면의 일부이다. 애니메이션으로 보여주는데 있어서 속도는 main() 함수에 들어가는 entry_st 명령문의 인자(parameter)로 지정할 수 있다.

5. 검증

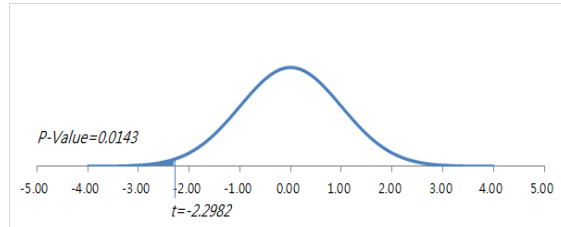
제안된 시각화 시스템의 교육효과를 검증하기 위하여, 컴퓨터과학과 전공학생 32명을 대상으로 객체지향 소스코드의 이해도에 관한 시험을 실시하였다. 시험대상인 32명을 16명씩 두 그룹, 일반그룹과 시각화그룹으로 나누었다. 일반그룹은 시각화 시스템을 사용하지 않고 이해도 시험(0~5점)을 실시하였으며, 시각화 그룹은 시각화 시스템을 사용하여 시험을 실시하였다. 검정을 위한 귀무가설과 대립가설은 다음과 같다.

- H_0: $\mu \leq \mu_0$
  H_a: $\mu > \mu_0$
  - μ : 시각화그룹의 시험성적
  - μ_0 : 일반그룹의 시험성적

두 그룹의 시험결과에 대하여 독립표본 t-test 분석을 시행한 결과는 다음 <Table 1>과 (Fig. 9)와 같이 나타났다.

<Table 1> One tailed t-test results

Group	N	Mean	Std.Dev.
Normal Group	16	3	0.8165
Visualization Group	16	3.625	0.7188
df	30		
t	-2.2982		
P(T<=t)	0.0143		



(Fig. 9) The Obtained t-value and P-value

일반그룹의 평균 점수는 3점이며, 시각화그룹의 평균 점수는 3.625이다. 두 그룹의 점수 값을 분석한 결과 자유도는 30이며, 검정통계량 t 값은 -2.2982이다. (Fig. 9)에 보이는 바와 같이 t=-2.2982의 왼쪽 꼬리 면적은 0.0143이며, 이는 유의수준 alpha=0.05 보다 더 작다. 따라서 귀무가설 H0는 기각된다. 즉, 유의수준 0.05에서 시각화 시스템을 사용하는 것이 프로그램을 이해하는데 도움이 된다는 결론을 뒷받침한다.

6. 결론

본 논문에서는 객체지향프로그래밍 개념을 교육하고, 기능을 이해하며, 소스코드 분석 및 이해향상에 도움을 줄 수 있는 객체 상호작용 시각화 방법을 제안하였다. 제안된 시스템은 별다른 노력 없이, 개발된 프로그램의 실행과 동시에 프로그램 내부 객체들이 동작하는 모습을 시각적으로 보여준다. 제안된 시스템의 교육효과를 검증하기 위하여 학생들을 대상으로 실험을 진행하였으며, t검증 결과 시각화 시스템이 학습효과에 유의미한 효과가 있음을 보였다.

향후 과제로는 현재 단독시스템으로 구현된 시각화 변환 시스템을 eclipse의 plug-in 모듈 형태로 변경하여 변환 과정 없이 실시간으로 시각화 기능을 사용하게 만드는 것이다.

참고문헌

[1] Ben-Bassat Levy, R., Ben-Ari, M., & Uronen, P. A., 2003, "The Jeliot 2000 program animation system.", Computers & Education, 40(1), 1-15.

[2] Deok-Gil Jung, Min-Po Jung, Hyuk-Gyu Cho, Young-Uhg Lho(2014). A Development of the Evaluation Metrics and Analysis of the Object-Oriented Visual Programming Education Using Alice Programming. *Journal of the Korea Institute of Information and Communication Engineering 18*(3), 742-748.

[3] Geunho Jeong, Hyun-Joo Moon, Cheon-Yeol Rhew, Chae-Woo Yoo, Hoo-Bong Song(1994). Design and Implementation of Class Visual Programming System for C++. Proceedings of the Korean Information Science Society 21(2A), 727-730.

[4] Hak-Chul Lee, Hee-Chul Kim, Sang-Ho Lee(1995). Implementation of a Program Visualization System for Algorithm Education. Proceedings of the Korean Information Science Society 22(2A), 761-764.

[5] JinHee Im(1995). Ssias is not C. Seong-An-Dang.

[6] JuHyuck Kim, Ki-Hwan Chon, Kyun-Rak Chong(1998). Design and Implementation of an Algorithm Instruction System using Visualization Techniques. *Journal of the Korea Information Science Society: Computing Practices 4*(3), 391-398.

[7] JungSook Sung, HyeonCheol Kim(2015). Analysis on the International Comparison of Computer Education in Schools. *The Journal of Korean Association of Computer Education, 18*(1), 45-54.

[8] JunSeok Cheon, Gyun Woo(2016). Saesark: A Korean Object-Oriented Programming Language for Beginners. *Journal of the Korea Contents Association 16*(3), 288-295.

[9] Kanemune Susumu, Kuno Yasushi(2009). Programming with Doolittle. Human Science.

[10] Kyung-hoon Kim(2017). 2015 Revised Curriculum The right direction of software education and its case study. Seoul Education, Vol 226, Spring. Seoul Education Research & Information Institute.

[11] Mehmet Tekdal, 2013, "The Effect of an Example-Based Dynamic Program Visualization Environment on Students' Programming Skills", Educational Technology & Sociery, 16(3), 400-410.

[12] Michael D. Byrne, Richard Catrambone, John T. Stasko, 1999, "Evaluating animations as student aids in learning computer algorithms", Computers & Education Volume 33(4), Pages 253-278.

[13] Moreno, A., & Joy, M. S., 2007, "Jeliot 3 in a demanding educational setting.", Electronic Notes in Theoretical Computer Science, 178, 51-59.

[14] Osman, Waleed Ibrahim, and Mudawi M. Elmusharaf., 2014, "Effectiveness of combining algorithm and program animation: A case study with data structure course." Issues in Informing Science and Information Technology Volume 11.

[15] Sangwook Kim, Kyungmin Koo, Mansoo Kim, Jieun Park, Jungmin Seo, Hoyeon Seo, Choonhee Lee(1993). A Visualization System for Object-Oriented Programming. *Journal of the Korea Information Science Society 20*(12), 1773-1792.

[16] Suhwan Kim, JeongByeong Chae(2014). Trend Analysis of Educational Programming Language and Teaching-Learning Examples. KERIS Issue Report RM 2014-25. Korea Education and Research Information Service.

[17] Yun-Jung Lee, In-Joon Jung, Gyun Woo(2013). Implementation and Experimentation of StyleJigsaw for Programming Beginners. *Journal of the Korea Contents Association 13*(2), 19-31.

저자소개

신 우 창

2003 서울대학교 대학원 컴퓨터공학과 (공학박사)

2003~2008 서경대학교 인터넷정보학과 교수

2008~현재 서경대학교 컴퓨터과학과 교수

관심분야: 소프트웨어 영재교육, 객체지향 프로그래밍, 소프트웨어 개발방법론

e-mail: wcshin@skuniv.ac.kr