ISSN 2586-1816
융합정보논문지 제7권 제4호 pp. 69-74, 2017 DOI : https://doi.org/10.22156/CS4SMB.2017.7.4.069

ICT 환경에서 프로그램보호를 위한 소스코드 분석 사례 연구

이성훈[1], 이동우[2*]
[1]백석대학교 정보통신학부, [2]우송대학교 컴퓨터정보학과

A Study on Analysis of Source Code for Program Protection in ICT Environment

Seong-Hoon Lee[1], Dong-Woo Lee[2*]
[1]Division of Information Communication, Baekseok University
[2]Department of Computer Information, Woosong University

요 약 현재 우리 사회를 대변하고 있는 단어는 정보통신기술(ICT)일 것이다. 정보통신 기술의 발전으로 우리나라의 소프트웨어 산업은 양적으로, 질적으로 발전하고 있다. 이러한 소프트웨어 산업의 고도성장에 따른 문제점 중 하나는 소프트웨어에 대한 지적재산권 문제이며 이로 인해 다양한 유형의 분쟁들이 발생하고 있다는 점이다. 소프트웨어에 대한 대부분의 분쟁 중 응용 프로그램이 차지하는 비중이 상당히 많은 비율을 차지하고 있다. 이는 다양한 산업분야에서 필요로 하는 프로그램들이 수없이 개발되고 있기 때문에 자연스러운 현상이라 할 수 있다. 따라서 현재와 같은 고도화된 ICT 환경에서는 지적재산권에 대한 중요성이 점점 더 확장될 것이다. 본 연구에서는 프로그램에 대한 사례를 중심으로 하여 프로그램의 유사성 및 복제도 정도를 도출할 수 있는 방안을 소스 프로그램의 수준에서 제시하였다.

키워드 : 정보통신기술, 소프트웨어, 프로그램, 보호, 응용 프로그램, 저작권

Abstract ICT(Information Communication Technology) is a key word in our society on today. Various support programs by the government have given many quantitative and qualitative changes to the software industries. Software is instructions(Computer Program) and data structure. Software can be divided into Application program and System program. Application programs have been developed to perform special functions or provide entertainment functions. Because of this rapid growth of software industries, one of the problems is issue on copyright of program. In this paper, we described an analysis method for program similarity based on source code in program.

Key Words : ICT, Software, Program, Protection, Application program, Copyright

1. 서론

현재 우리 사회의 커다란 특징 중 하나는 정부의 각종 지원 프로그램에 의해 소프트웨어 산업이 양적, 질적으로 많은 성장과 변화를 가져왔다는 점이다. 소프트웨어란 일반적으로 프로그램과 같은 의미로 혼용하여 사용하고 있다. 하지만 정확한 의미는 컴퓨터를 작동시키고, 컴퓨터가 어떠한 일을 처리할 수 있도록 순서와 방법을 지시하는 명령어의 집합인 프로그램과 프로그램의 수행에 필요한 절차, 규칙, 관련 문서 등의 총칭을 의미한다[1-4]. 이러한 소프트웨어는 크게 시스템 소프트웨어와 응용 소프트웨어로 분류할 수 있다. 시스템 소프트웨어

Received 2017-07-27 Revised 2017-08-09 Accepted 2017-08-20 Published 2017-08-31
*Corresponding author : Dong-Woo Lee (dwlee@wsu.ac.kr)

는 컴퓨터를 작동시키기 위한 기본적인 프로그램으로 운영체제, 장치 드라이버 등이 이에 해당한다. 반면에 응용 소프트웨어는 사용자들이 필요에 따라 특정 업무를 처리하기 위해 개발되어진 프로그램으로 다양한 장르의 프로그램들이 존재한다. 이러한 소프트웨어 산업의 고도성장에 따른 문제점 중 하나는 소프트웨어에 대한 지적재산권 문제이며 이로 인해 다양한 유형의 분쟁들이 발생하고 있다는 점이다[5-13]. 전반적으로 정보통신기술의 발달과 더불어 불법복제 기술 또한 다양해지면서 소프트웨어에 대한 지적재산권 침해문제들이 발생하고 있는 것이다.

소프트웨어에 대한 대부분의 분쟁 중 응용 프로그램이 차지하는 비중이 상당히 많은 비율을 차지하고 있다. 이는 다양한 산업 분야에서 필요로 하는 프로그램들이 수없이 개발되고 있기 때문에 자연스러운 현상이라 할 수 있다. 따라서 현재와 같은 고도화된 ICT 환경에서는 지적재산권에 대한 중요성이 점점 더 확장될 것이며, 증가하는 분쟁에 대해 신속하고 정확한 해결이 요구되며, 더 나아가서는 분쟁의 해결을 위해 제도개선 및 체계화가 중요한 문제로 대두되고 있다. 본 연구에서는 프로그램에 대한 사례를 중심으로 하여 프로그램의 유사성 및 복제도 정도를 도출할 수 있는 방안을 소스 프로그램의 수준에서 제시하고자 하였다. 2장에서는 프로그램 분석을 위한 제반 사항들에 대해 기술하였다. 3장에서는 두 기관에서 개발된 소스 프로그램에 대한 분석 결과를 기술하였으며 마지막으로 4장에서 결론을 기술하였다.

2. 프로그램 분석 내용

하나의 프로그램이 다른 프로그램을 복제한 정도를 판단하는 방법에는 기준으로 삼는 프로그램에 따라 두 가지가 존재한다. 각 방법에 대한 구체적인 내용은 아래 Fig. 1을 이용하여 알아보자. Fig. 1의 예는 B 프로그램이 A 프로그램의 70%를 복제를 한 상태이고, 독자적으로 추가한 부분이 30%로 구성된 경우이다.

먼저 A 프로그램 기준 방식으로 A 프로그램의 얼마나 많은 부분이 복제되었는지를 기준으로 한 방식이다. A의 70%가 복제 당하였으므로 B가 A를 복제한 정도는 70%이다. 이 방식은 B가 아무리 많은 부분을 독자적으로 개발하였더라도 복제정도에 영향을 미치지 못한다.

다음 두 번째 방식인 B 프로그램 기준 방식은 B의 얼마나 많은 부분이 복제한 것인지를 기준으로 한다. B의 70%가 A로부터 복제한 부분이므로 복제정도는 70%이다. 이 방식에서는 B 프로그램에서 독자적으로 개발한 부분이 많으면 많을수록 복제정도는 작아진다.

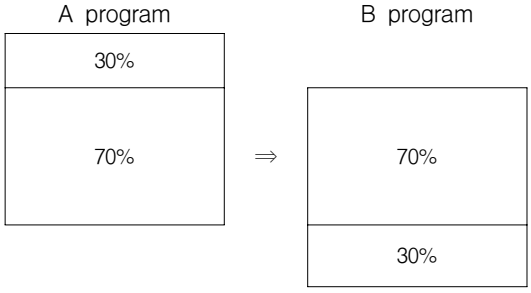

Fig. 1. Copying sample

본 프로그램 분석을 수행하는 데 있어 채택한 기본적인 분석 방식은 A 프로그램 기준 방식이다. 이 방식은 복제 정도(또는 비율)의 계산 시에 A 소프트웨어 전체에서 얼마나 많은 부분을 B가 복제 또는 도용하였는지를 기준으로 삼는다. 즉, A의 소프트웨어에 관한 지적재산권이 얼마나 많이 침해당하였는지를 파악하는 데 주안점을 두고 있다. 이와 상반된 방식으로는 B 프로그램 기준 방식은 이것은 B의 얼마나 많은 부분이 복제 또는 도용한 것인지를 기준으로 삼는다. 즉 B의 소프트웨어 중 얼마나 많은 부분이 A의 소프트웨어의 지적재산권을 침해하였는지를 파악하는 것이 목적인 경우이다.

본 프로그램 분석에서는 명시한 4개의 소프트웨어 모듈 이외에도 제출된 A 소스 프로그램과 B 소스 프로그램 전부를 상호 비교분석하였다. 먼저 프로그램의 유사성 및 도용여부를 판단하기 위해서는 먼저 소스 코드 수준에서의 비교 및 분석이 가장 확실한 방법이면서 가장 중요한 부분이다. 소스 코드의 내용이 동일하다면 이는 객관적으로 명확하게 도용되었다라고 단정할 수 있기 때문이다. 소스코드 수준에서 도용여부가 명확하게 결정될 수 있다면 다른 사소한 분석 항목에 대한 비교, 분석이 필요하지 않을 것이다. 따라서 소스 코드의 비교, 분석이 프로그램의 유사성 및 도용 여부를 결정하는데 있어 가장 중요한 분석 항목이라 할 수 있다.

```
File A_company.cpp: 3893 tokens, File B_company.cpp: 4278 tokens
Total: 8171 tokens

A_company.cpp: line 1055-1082 |B_company.cpp: line 266-293 [95]
 rotate_point (pt, _cen1, ang, _res1) | vRotate (pt, _cen1, ang, _res1);
 (*_res2) = (*_res1); | (*_res2) = (*_res1);
 return 1; | return 1;
 } | }

 COORD2 cen1 = _cen1; | rpoint cen1 = _cen1;
 COORD2 cen2 = _cen2; | rpoint cen2 = _cen2;
 |
 if (_r1 > _r2) | if (_r1 > _r2)
 { | {
 double tmp = _r1; | double tmp = _r1;
 _r1 = _r2; | _r1 = _r2;
 _r2 = tmp; | _r2 = tmp;
 |
 cen1 = _cen2; | cen1 = _cen2;
 cen2 = _cen1; | cen2 = _cen1;
 } | }
 |
 double dist = sqrt (dist2); | double dist = sqrt (dist2);s,s
 double mind = fabs (dist - _r2); | double mind = fabs (dist - _r2);
 double maxd = dist + _r2; | double maxd = dist + _r2;
 if (mind > _r1 || maxd < _r1) return | if (mind > _r1 || maxd < _r1) return
 |
 double cos_a = (_r1 * _r1 - _r2 * _r | double cos_a = (_r1 * _r1 - _r2 * _r
 double alpha = fabs (acos (cos_a)); | double alpha = fabs (acos (cos_a));
 |
 double ang = angle (cen1, cen2); | double ang = vAngle (cen1, cen2);

A_company.cpp: line 258-265 |B_company.cpp: line 238-245 [33]
 double numerator_12 = cy_43 * cx_31 - | double numerator_12 = cy_43 * cx_31 -
 double numerator_34 = cy_21 * cx_31 - | double numerator_34 = cy_21 * cx_31 -

 *_factor_34 = numerator_34 / denomina | *_factor_34 = numerator_34 / denomina
 *_factor_12 = numerator_12 / denomina | *_factor_12 = numerator_12 / denomina

 COORD2 v_12, v_p; | rpoint v_12, v_p;

A_company.cpp: line 691-697 |B_company.cpp: line 533-541 [33]
 if (dGap < 0.000001) return NULL; | if (dGap < 0.000001) return NULL;
 |
 int iPointNo = (int) ((_ed_ang - _st |
 COORD2* pPtArray = (COORD2*) malloc ( | int iPointNo = (int) ((_ed_ang - _st
 if (pPtArray == NULL) return NULL; | rpoint* pPtArray = (rpoint*) malloc (
 | if (pPtArray == NULL)
 COORD2 pt1, pt2; | return NULL;
 | rpoint pt;
```

Fig. 2. Output sample

3. 프로그램 분석 결과

먼저 프로그램 소스 전체에 대한 내용으로, 이 분석 항목은 A기관과 B기관에서 제출한 프로그램 소스를 모두 비교하는 것을 의미한다. 왜냐하면, B가 A를 복제한다고 가정할 때 파일의 이름들은 얼마든지 변경할 수 있으므로, 이름이 같은 파일끼리만 비교하는 것으로는 부족하고, 정확한 비교를 위해서는 모든 파일 간 상호 비교를 수행하여야 한다. 예를 들어 A가 M개의 소스 파일들로 구성되고, B는 N개의 소스 파일들로 구성된다고 하자. 그러면 수행하여야 할 파일 단위의 비교 개수는 A의 각 파일당 B의 모든 파일을 비교하여야 하므로 M x N이 된다. 우리가 쉽게 예측할 수 있듯이 이러한 비교 작업은 n^2 수준의 비교 회수를 요구하므로 파일 개수에 따라 비

교 요구 횟수가 엄청나게 늘어나고 또한 상당히 많은 시간이 소요될 수 있다.

현재까지 자주 사용된 소스 파일 비교 도구로는 Windiff라는 파일 간 또는 디렉터리 간 텍스트 비교 도구가 있다. 이 도구는 디렉터리 비교의 경우 양 디렉터리에서 이름이 같은 파일끼리만 비교한다. 따라서 Windiff는 프로그램 소스 전체 비교에 활용하기에는 적합하지 않다.

또 다른 소스 비교 도구로는 *SIM*(Similarity의 준말)이라는 소프트웨어가 있으며, 본 연구에서는 SIM을 사용하여 소스 비교를 수행하였다. SIM은 토큰(Token)을 기본 단위로 하여 연속되는 여러 개의 토큰으로 구성되는 런(Run)을 비교 단위로 하여 두 파일을 비교한다[14]. Fig. 2는 SIM의 출력 샘플이다.

Fig. 2의 출력 예는 A기관의 A_company.cpp와 B기관의 B_company.cpp를 비교한 결과이다. 이 결과가 제공하고 있는 정보는 3개 부분에서 표시한 것과 같이 유사한 코드가 검출되었고, 각각의 라인 번호는 출력된 것과 같으며, 각 부분의 토큰의 개수는 [ ]안의 수와 같다는 것이다. 이 예에서 유사하다고 출력된 부분을 자세히 보면 알 수 있듯이, SIM은 완전히 기계적인 바이트 단위의 텍스트 비교를 수행하는 것이 아니고 토큰을 단위로 하기 때문에 약간의 줄 바꿈, 변수명 변경 등에도 불구하고 유사한 부분을 검출한다. 이러한 검출 결과는 단지 복제도 산출의 참고자료로 제공될 뿐이며 SIM은 복제 여부 자체에 대해서는 아무런 결정도 내리지 않는다.

Table 1은 위와 같은 파일 대 파일 비교를 수행한 대상 파일의 분류, 각각의 개수와 그에 따른 비교 회수 등을 요약하였다. Table 1의 내역을 보면 프로그램 소스 파일은 .cpp 파일과 .h의 두 종류가 있고, 각 종류에서 상당한 부분이 공개 소프트웨어로 구성되어 있어 비교 대상에서 제외하였다. 왜냐하면 A의 공개 소프트웨어 이용 부분은 B가 사용하였더라도 복제라고 할 수 없다[15].

Table 1. Comparison target program source organization

File class		File number		comparison number
		A	B	
C File (*.cpp)	comparison target	535	369	535 x 369 = 197,415
	open source File(exclusion)	115	95	
C header File (*.h)	comparison target	514	338	514 x 338 = 173,732
	open source File(exclusion)	79	119	

Table 1에서의 비교 대상에서의 제외 결정에 가장 핵심적인 것은 어떻게 공개 소프트웨어로부터 가져온 소스 파일인가를 판명하는 것이다. 본 프로그램 분석에서는 공개 소프트웨어인 것이 확실하게 확인된 경우만 비교 대상에서 제외하도록 하였다. 공개 소프트웨어 여부의 확인은 주로 인터넷 공개 여부와 소스 파일 내에 공개 소프트웨어 표시 여부 등으로 판별할 수 있다. 예를 들면 Fig. 3, Fig. 4와 같은 저작권 표시(Copyright notice)의 존재는 공개 소프트웨어임을 확인해 주는 부분이다.

```
 * $Id: avc_bin.c,v 1.6 1999/08/23 18:17:16 daniel Exp $
 * Name: avc_bin.c
 * Project: Arc/Info vector coverage (AVC) BIN->E00 conversion library
 * Language: ANSI C
 * Purpose: Binary Files access functions.
 * Author: Daniel Morissette, danmo@videotron.ca
 **********************************************
 * Copyright (c) 1999, Daniel Morissette *
 * All rights reserved. This software may be copied or reproduced, in
 * all or in part, without the prior written consent of its author,
```

Fig. 3. Copyright sample 1

```
* MRCEXT: Micro Focus Extension DLL for MFC 2.1+
* Copyright (C)1994-5 Micro Focus Inc, 2465 East Bayshore Rd, Palo Alto, CA 94303.
* This program is free software; you can redistribute it and/or modify
* it under the terms of the GNU General Public License as published by
* the Free Software Foundation. In addition, you may also charge for any
* application using MRCEXT, and are under no obligation to supply source
```

Fig. 4. Copyright sample 2

Table 1의 비교대상 파일들에 대하여 SIM을 수행한 결과가 Fig. 5에 요약되어 있다. 전체적으로 소스프로그램의 유사성 내지 복제 정도는 소스 코드의 라인 등을 비교, 분석했을 때 약 1.2%의 유사성을 보이는 것으로 조사되었다. 따라서 이를 복제되었다라고 보기에는 어려운 결과라 할 수 있다.

File class	File list	File number (A comparison target)	line number (A comparison target)	copy rate(line number criteria)
.cpp File of A copied by B	- DGNReader.cpp, 28 other Files	29 (535)	4,025 (343,192)	(4.025+543)/ (343,192+46,414) = 1.2%
.h File of A copied by B	- Splash.h, 6 other Files	7 (543)	543 (46,414)	

Fig. 5. Result of training

지리정보시스템에서 중요한 요소 중 하나는 공간 연산과 관련된 부분이다. 이에 대한 두 관계사간의 공간 연산에 대한 구체적인 내용은 아래 Fig. 6, Fig. 7과 같다. 이를 바탕으로 볼 때 공간 연산에 대한 프로그램의 복제가 이루어졌다고 볼 수 없다.

```
- in objectCalcFunction.cpp

if (iFlag == 1) tt.do_or();
else if (iFlag == 2) tt.do_and();
else if (iFlag == 3) tt.do_sub();

- in boolonly.cpp

BOOL GPolygonOp :: do_and ( )
{
statusbar = (StattusBar *) new DosStatusBar( );
_messagehandler = (GDSMessage *) new
DOSMessage( );
m_booleng -> Do_Operation(BOOL_AND);
```

Fig. 6. A program spatial operation

```
- in boolonly.cpp

{       switch (operation)
        {
CASE (OR);
                CASE (AND);
                CASE (XOR);
                CASE (A_SUB_A);
                CASE (A_SUB_A);
                ..... }
```

Fig. 7. B program spatial operation

4. 결론

본 연구에서는 프로그램의 보호라는 관점에서 소스코드를 기준으로 하여 상호 유사성 혹은 복제 정도를 알아보는 내용이 핵심이다. 이에 두 개의 프로그램(A, B 프로그램)을 대상으로 하여 소스 코드의 유사성을 알아보기 위해 SIM이라는 도구를 사용하였다. A와 B 모두 공개 소프트웨어의 소스를 많이 사용하여 복제 정도 판단에서 제외된 소스 부분이 많았다. 제출된 모든 소스 파일의 공개 소프트웨어 여부를 프로그램 분석자가 확인하는 데는 한계가 있으므로, 일단 복제한 것으로 판단하여 소스 복제도 1.2%에 계산된 부분들도 공개 소프트웨어의 일부분일 가능성도 완전히 배제할 수는 없다. 이러한 상황과 소스코드 수준에서의 상당히 낮게 계산된 복제도 및 지리정보 시스템의 특성인 공간 연산에 대한 비교 결과인 공간 연산에 대한 상이점 등을 고려할 때 두 프로그램 간 B가 A를 복제하지 않았다고 할 수 있다.

향후 연구 방향으로는 프로그램 소스코드 수준에서 유사정도가 임계치 이상인 경우에 대하여 좀 더 면밀한 방법들에 대한 연구가 필요할 것이다.

REFERENCES

[1] I. Sommerville. (2001). *Software Engineering.* Boston : Addison-Wesley Publishing.

[2] J. Whitten & L. Bentley. (1994). *System Analysis & Gesign Methods.* New York : McGraw-Hill College.

[3] R. S. Pressman. (1992). *Software Engineering - A Practitioner's Approach. 3th edition.* New York : McGraw-Hill.

[4] T. Gildersleeve. (1985). *Successful Data processing Systems Analysis.* New York : Prentice Hall.

[5] E. Sebastine. (2008). *The Economic Properties of Software.* London : Jena Economic Research Papers.

DOI : 10.2139/ssrn.1430885
[6] F. Culwin & J. Naylor. (1995). *Pragmatic Anti-pragiarism in Action.* Dublin : Proceeding of 3th al Ireland conference on the teaching of computing.
[7] Turnitin. (2017). *What is Plagiarism.* Turnitin. http://www.plagiarism.org/
[8] Glatt Plagiarism Services. *Self-Detection Test Instructions.* Glatt Plagiarism Services. http://www.plagiarism.com/
[9] D. Abraham. (2015). Subject Matter, Scope, and User Rights in Copyright Law. *Studies in Law, Politics and Society,* 59-74.
DOI : 10.1108/s1059-433720150000067003
[10] A. H. Herbert. (1918). International Copyright Relations of the United States. *The Yale Law Journal, 27(3),* 348-354.
DOI : 10.2307/787438
[11] J. K. Harris. (1994). *Plagiarism in Computer Science Cources.* New York : Proceeding 1994 Ethics in Computer Age.
DOI : 10.1145/199544.199601
[12] S. Stokes. (2001). *Art and Copyright.* London : Hait Publishing.
[13] Copyright/Other Information Comments. (2017). *Chapter 1: An Overview of Copyright.* Legal Protection of Digital Informaion. http://digital-law-online.info/lpdi1.0/treatise9.html
[14] K. Zlato. (2006). *Determinants of Worldwide Software Piracy.* New Zealand.: Information Science Institude.
[15] K. Dan. (1999). *Why Open Source is the Optimum Economic Paradigm for Software.* Freecode. http://freecode.com/articles/why-open-source-is-the-optimum-economic-paradigm-for-software

저 자 소 개

이 성 훈(Seong-Hoon Lee) [정회원]

- 1995년 2월 : 고려대학교 일반대학원 컴퓨터학과 이학석사
- 1998년 2월 : 고려대학교 일반대학원 컴퓨터학과 이학박사
- 1998년 3월 ~ 현재 : 백석대학교 정보통신학부 교수

<관심 분야> : 분산시스템, 웹서비스, 그리드 시스템, 컨버전스, 융합산업

이 동 우(Dong-Woo Lee) [정회원]

- 1984년 8월 : 고려대학교 일반대학원 컴퓨터공학 공학석사
- 2005년 2월 : 고려대학교 일반대학원 전산과학과 이학박사
- 1995년 3월 ~ 현재 : 우송대학교 컴퓨터정보학과 교수

<관심 분야> : 웹기반분산시스템, 능동시스템, 데이터베이스, 컨버전스