소프트웨어 소스 코드의 저작권 관리를 위한 디지털 라이센스의 비교와 분류 그리고 크립텍스 모델

Discrimination and Comparison of Digital License for Copyright Management of Software Source Code and CRYPTEX Model

차병래, 정영기
호남대학교 컴퓨터공학과

ByungRae Cha(chabr@honam.ac.kr), YoungKee Jung(ykjung@honam.ac.kr)

요약

소프트웨어 산업은 21세기 정보화 사회의 발전에 있어서도 중요한 역할을 하고 있다. 소프트웨어 소스 코드의 소유권 분쟁이 발생 시 소유권을 증명하기 위해서는 원본의 소프트웨어 소스코드를 판별해야만 하는 문제점을 갖고 있다. 본 논문에서는 소프트웨어 소스코드의 원본 판별을 지원하기 위한 소프트웨어 소스코드의 디지털 라이센스는 소스코드의 예약어를 파싱하여 계층구조를 갖는 XML 파일로 표현하며, 복잡한 소스코드 대신에 소프트웨어 소스코드의 노드 패턴과 아키텍처 패턴인 트리 구조 형태로 표현할 수 있다. 그리고 디지털 라이센스에 의한 소스코드의 분류 가능성에 대한 시뮬레이션과 크립텍스 모델을 제안한다.

■ 중심어 : | 디지털 라이센스 | 소프트웨어 소스 코드 | 저작권 관리 | 크립텍스 |

Abstract

The software industry is so important to the 21C information society. Not only the digital content control but the technology of software source code for the intellectual property is so much mean to international competition. On occurring disputation property of software source code, we have to prove the fact, there is a problem to discriminate the original software source code.

In this paper, we make a study of the digital licence prototype for discriminate the original source code. Reserved words of software source code by parsing express to XML file that have hierarchical structure. Then, we can express node pattern and architecture pattern of software source code by tree structure form instead of complex software source code. And we make a simulation of discrimination possibility of digital license and propose CRYPTEX model.

■ keyword : | Digital License | Software Source Code | Copyright Managements | CRYPTEX |

I. 서 론

어느덧 IT 산업은 우리나라의 주력산업으로 자리를 잡았으며, 생산되는 모든 제품들이 소프트웨어 없이는 제 구실을 못할 뿐만 아니라 소프트웨어가 그 제품의 부가가치를 결정하는 지식기반의 경제 환경에서 우리는 살고 있다. 향후에는 다른 산업의 경쟁력 향상뿐만 아니라 차세대 성장산업의 기반이 되는 소프트웨어 산

* 본 연구는 한국과학재단 특정기초연구 (R01-2007-000- 20330-0)의 지원에 의한 것입니다.
접수번호 : #070911-001
접수일자 : 2007년 09월 011일
심사완료일 : 2007년 12월 14일
교신저자 : 차병래, e-mail : chabr@honam.ac.kr

업이 세계 경제의 주력산업으로 성장할 것은 분명하다.

소프트웨어 산업이 안정적인 성장기반을 확보하여 미래의 전략산업으로 육성하기 위해서는 소프트웨어 개발 인력의 역량 확보, 소프트웨어 산업의 국가적인 지원과 인프라 구축, 소프트웨어의 불법복제 사용 금지 등이 필요하다. 더불어 소프트웨어를 관리 및 보호를 위한 프레임워크, 그리고 이를 지원할 수 있는 제반 기술 개발이 필수적이다. 좀 더 세부적으로, 콘텐츠 산업의 활성화를 통하여 기간산업으로 육성하기 위해 필요한 국가차원의 콘텐츠 유통인프라 구축이 매우 필요하다. 특히, DRM 기술은 디지털 콘텐츠의 저작권 보호를 위한 기술로서 최근 IT환경의 발전에 따라 그 중요성을 더해가고 있으며 향후 시공간을 초월하는 정보화 사회가 발전할수록 콘텐츠 저작권의 중요성은 더욱 커질 것으로 예상된다.

하드웨어 및 소프트웨어 솔루션의 발전 속도가 사용자들이 만족하는 수준에 도달하고 있으며 사용자의 계속되는 콘텐츠의 수요에 따라 점차 콘텐츠 중심의 산업구조로 발전할 것으로 예상한다. 이러한 시점에서 DRM의 주요한 관리 대상인 콘텐츠의 적용 범위를 아스키코드나 유니코드로 이루어진 소프트웨어의 소스코드까지 영역을 확장하고자 한다. 소프트웨어의 소스코드는 아주 단순한 아스키코드 또는 유니코드로 구성된 파일로 되어 있다. 이러한 단순하고 지식집약적인 소스코드에 대한 불법 유통을 방지하기 위한 기술과 국가차원의 소프트웨어 관리 정책의 수립이 절실하게 필요하다. 소프트웨어 소스코드의 유출의 문제점을 분석하고 이를 보완하기 위한 기술을 언급한다. 또한 안전한 소프트웨어 소스코드의 유통을 위해 불법 복제 추적이 가능한 시스템을 설계 및 제안한다. 그리고 이를 뒷받침하는 기술로는 암호화, 압축, 분류 및 비교, 이동 에이전트 기술 등이 필요하다.

디지털 콘텐츠의 저작권 보호를 위한 연구에 비해서 소프트웨어의 소스코드에 대한 저작권을 관리하기 위한 기술은 아직도 초기 연구단계에 있다. 소프트웨어 소스코드의 소유권 분쟁이 발생 시 소유권을 증명하기 위해서는 원본의 소프트웨어 소스코드를 공개해야만 하는 문제점을 갖고 있다. 본 연구에서는 소프트웨어의 소유권 주장에 하드카피이외에 더 많은 정보를 부여하며, 컴퓨팅 환경에서 자동으로 처리를 위한 프로토타입을 연구하였다. 연구 내용으로는 소프트웨어 소스코드의 원본 판별을 지원하기 위한 소프트웨어 소스코드의 디지털 라이센스는 소스코드의 예약어를 파싱하여 계층구조를 갖는 XML 파일로 표현하며, 복잡한 소스코드 대신에 소프트웨어 소스코드의 아키텍처를 트리 구조 형태로 표현할 수 있다. 그리고 디지털 라이센스에 포함된 인덱스 패턴 정보, 아키텍처 패턴 정보 그리고 노드 패턴 정보를 이용한 소스코드의 분류 가능성에 대한 연구와 크립텍스 모델 제안을 하였다.

본 논문에서 2장은 소프트웨어 소스코드의 DRM 기술, 디지털 라이센스의 비즈니스 모델, 그리고 소스코드의 디지털라이센스에 대한 관련 연구를 기술하였다. 3장에서는 소프트웨어 소스코드와 디지털 라이센스의 비교, 그리고 디지털라이센스를 이용한 분류 3단계 과정을 기술하였다. 4장에서는 디지털라이센스의 능동적인 대처를 위한 크립텍스(CRYPTEX) 모델을 제안하고, 마지막으로 5장에서는 결론 및 향후 연구에 대해 기술하였다.

II. 관련 연구

2.1 소프트웨어의 소스코드를 위한 DRM

디지털 콘텐츠란 디지털로 되어 있는 문자, 소리, 화상, 영상 등의 형태로 이루어진 정보 내용물이며, 본 연구에서는 디지털 콘텐츠의 영역을 소프트웨어의 소스코드까지 확장한다. 소프트웨어의 소스코드는 논리적 사고과정을 컴퓨터가 처리할 수 있는 프로그래밍 언어를 통해 기술해 놓은 아스키코드나 유니코드로 만들어진 파일의 디지털 콘텐츠이다.

소프트웨어 소스코드의 패턴 매칭에 대한 연구는 Rieger[1]과 Yang[2]에 의해서 연구되어졌다. Rieger는 소프트웨어의 소스코드가 중복되거나 복제된 부분을 시각화하여 탐지하는 연구를 수행하였고, Yang은 두 개의 다른 프로그램을 구문론적으로 동일한 코드를 찾는 연구를 수행하였다. 프로그램 표절 검출 S/W로는

국외는 SIM[3], Dup[4], Plague, YAP[5], YAP3, MOSS[6] 등이 있다. 국내는 KAIST의 clonechecker와 부산대의 LOFC가 있으며[7], 프로그램 심의위원회의 *exEyesLight*[8]가 있다.

DRM(Digital Rights Management)은 디지털 콘텐츠의 보호를 위한 암호화 및 사용자 인증키 관리, 디지털 콘텐츠 유통 환경을 구성하는 주체들 간의 지적재산권 및 거래 규칙, 디지털 콘텐츠의 이용 및 분배, 사용 및 접근제어 등을 가능하도록 하는 디지털 저작권 관리에 대한 전반적인 기술이다. 즉, 콘텐츠의 암호화와 권한제어를 통해 콘텐츠에 권한을 가지고 있는 사용자들만이 그 권한에 따라 콘텐츠를 열어 볼 수 있도록 함으로서 디지털 콘텐츠의 생성, 유통, 사용 등에 관련된 전 분야의 서비스를 제공하는 것이다[9-11].

최근에는 파숫닷컴[12]에서 소스코드의 보안을 위해 DRM ONE 패키지를 개발하여 시판하고 있다. 이 패키지의 DRM ONE for CODE라는 소스코드 보안 솔루션은 소프트웨어 소스 자체에 대한 보안 취약점을 악용한 해킹 및 유출사고를 예방하기 위해 프로그램 소스코드를 중점적으로 보호하는 솔루션이다. 프로그램 소스코드의 생성 및 관리 과정에서 활용되는 다양한 전문 어플리케이션 및 주변 도구들을 지원하여 최첨단 기술인 소스코드를 보호하는 기능을 제공한다. DRM ONE for CODE의 특징으로는 다양한 S/W 개발 솔루션 및 협업 시스템 지원, 소스코드 생성 및 저장 시의 PC에서 자동 암호화 및 암호화 해제 가능, 다양한 오프라인 사용 정책에 따른 유연한 오프라인 사용 지원, 모든 온/오프라인 사용에 대한 부서/사용자/일시/문서 별 조회 및 통계 기능 등으로 사용내역 관리 기능 등의 특징을 갖고 있다.

대부분의 DRM 기술은 바이너리 파일에 대한 DRM 기술이다. 원래의 디지털 저작물인 바이너리 코드를 패키징을 수행하여 디지털 저작권 관리가 이루어진다. 그러나 디지털 저작물의 원천 요소인 소프트웨어 소스 코드에 대해서는 암호화이외에는 별다른 방법이 존재하지 않으며, 아직은 연구 초기 단계이다. 암호화 방법 역시 소프트웨어 소스 코드 개발자와 소프트웨어 소스 코드 구입자간의 1차적인 거래에만 저작권 관리를 지원할 뿐 그 이후에 발생하는 거래에 대한 저작권을 보호해 주지 못하고 있다.

소프트웨어 소스 코드에 대한 DRM 기술이 연구 초기 단계인 이유는 소스 코드 자체가 아스키코드나 유니코드로 구성되어 있기 때문이다. 이런 이유로 인하여, 암호화를 통한 1차적 거래에만이 1:1 관계의 디지털 저작권 관리가 이루어지고 있다.

본 연구는 소프트웨어의 소스코드에 대해서 DRM을 지원하기 위한 기초 연구이다. DRM의 저작권 보호 기술보다는 저작권 관리 기술을 제안하며, DRM의 영역을 소프트웨어의 소스코드까지 확장한다. 소프트웨어 소스코드에 대한 DRM 기술은 부재하며, 저작권 보호 기술로는 1:1 관계의 거래에서만 임의의 효력을 갖은 암호화 기술뿐이며, 초기연구단계에 머물러 있다. 소프트웨어 소스코드는 제품을 생산하는 공장의 설계도면을 뛰어넘어서 제품생산라인과 같은 아주 중요한 디지털 콘텐츠 자원이다. 그러나 정작 이 자원들을 지원하는 DRM 기술은 너무나도 부족한 상태이다.

2.2 디지털 라이센스의 비즈니스 모델

(1) 소프트웨어 불법 유출에 의한 분쟁

소프트웨어 개발자 A, B, C와 법원 D의 소프트웨어 불법 유출에 의한 문제 사항을 그림 1과 같이 제기한다. 소스코드의 원본의 소유자 A와 불법 유출자 B에 의해서 분쟁은 발생한다([그림 1]의 ①). 법원 D는 이 문제를 해결하기 위해서 개발자 A와 불법 유출자 B의 소프트웨어의 소스 코드를 제출할 것을 요구한다([그림 1]의 ②). 그러나 소프트웨어의 소스코드는 전문가가 감별해야 하기 때문에 법원 D는 소프트웨어 전문 개발자 C에게 소프트웨어 소스코드의 원본과 불법 유출본과의 차이점을 의뢰하게 된다([그림 1]의 ③). 소프트웨어 전문 개발자 C는 원본과 불법 유출 본을 비교하기 위해서는 소프트웨어의 소스코드를 이해해야 한다. C에 의해서 원본과 불법 유출 본을 감별하는 것은 전적으로 C의 주관성 및 소스코드의 이해 능력에 전적으로 의지하게 되어 객관성이 부족하게 된다([그림 1]의 ④). 여기서 제 2의 소스코드 불법유출 문제가 발생할 수도 있게 된다([그림 1]의 ⑤). 의도하지 않게도 C에 의해서 소스

코드의 지적재산권이 침해되는 문제가 발생하게 된다. 즉 소소코드의 원본과 불법 유출본을 감별하기 위해서 C가 소스코드를 이해하다보니 자동적으로 C에게 제 2의 소스코드가 불법 유출되는 문제가 발생하며, 제 2의 분쟁 소지를 갖게 된다.

[그림 1]의 저작권 침해 문제의 발생을 소프트웨어 소스코드의 디지털 라이센스에 의해서 제 2의 분쟁을 사전에 제거할 수 있다. 소프트웨어 개발자 A, B, C와 법원 D의 소프트웨어 불법 유출에 의한 문제 사항을 소프트웨어의 소스코드 대신에 디지털 라이센스로 대처하므로 제 2의 분쟁을 사전에 예방할 수 있다. 또한 C에 의한 주관적 감별을 디지털 라이센스의 패턴 분류 프로그램으로 대처함으로서 어느 정도 객관성을 갖출 수 있다.

DRM 기술은 여러 목적으로 제작된 디지털 콘텐츠가 제작·유통·사용되는 과정에서 해당 콘텐츠 저작권이 보호될 수 있도록 관리해 주는 시스템이다. DRM을 이루는 기술은 크게 두 가지로, 저작권 보호 기술과 저작권 관리 기술로 구분지울 수 있다. 대부분의 상업화된 DRM 기술은 저작권 보호기술이 주를 이루고 있다. 프로그램 소스코드에 대한 DRM 기술은 부재하며, 저작권 보호기술로는 1:1 관계의 거래에서만 임의의 효력을 갖은 암호화 기술뿐이며, 초기연구단계에 머물러 있다[12]. 소프트웨어 소스코드는 제품을 생산하는 공장의 설계도면을 뛰어넘어서 제품생산라인과 같은 아주 중요한 디지털 콘텐츠 자원이다. 그러나 정작 이 중요한 디지털 콘탠츠 자원들을 지원하는 DRM 기술은 너무나도 부족한 상태이다.

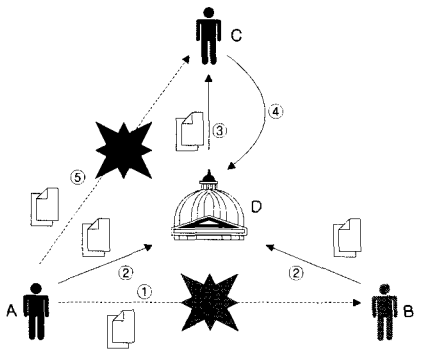

그림 1. 소프트웨어 분쟁에 의한 지적재산권 침해 문제의 다이어그램

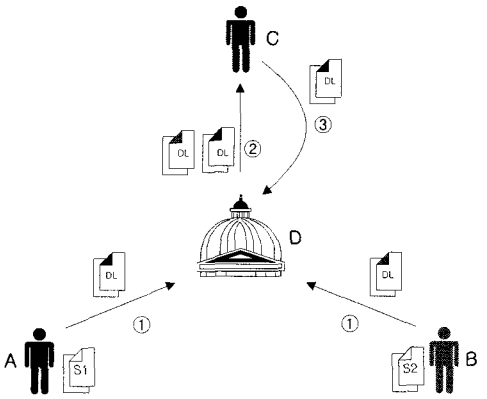

그림 2. 디지털 라이센스에 의한 지적 재산권 침해 방지

(2) 디지털 라이센스의 비즈니스 모델

[그림 1]의 저작권 침해 발생 문제는 [그림 2]와 같이 소프트웨어 소스코드의 디지털 라이센스를 적용함으로써 해결할 수 있다. A와 B의 분쟁 발생 시 소스코드를 법원에 제출하지 않고 A와 B가 갖고 있는 소스코드의 디지털 라이센스를 생성하여 법원에 제출하면 된다([그림 2]의 ①). 법원은 패턴 매칭 프로그램에 의해서 어느 정도 일치하는지를 객관적으로 보고서를 얻을 수 있으며, 추가적으로 C에게 자문을 구함으로써 분쟁 해결에 많은 도움이 될 것이다([그림 2]의 ②). 디지털 라이센스를 이용하면 또한 제 2 분쟁의 불씨를 제거하게 된다. C는 단지 소스코드가 아닌 디지털 라이센스를 검증하여 유출된 소프트웨어 소스코드가 원본에 어느 정도 일치하는 지에 대해 자문하면 된다([그림 2]의 ③).

디지털 라이센스에 의한 비즈니스 모델은 개발자 A의 지적재산권을 보호하고, 소스코드를 유출한 B를 좀 더 쉽게 판별하게 해주며, 또한 소스코드의 판별에 의한 제 2의 소스코드 유출을 사전에 막아서 제 2의 분쟁을 발생시키지 않는다. 더불어 C의 주관성에 의지하지 않고 객관적인 패턴 매칭 프로그램으로 확연한 보고서를 받을 수 있으며, C에 의해서 오랜 분석에 의한 자문에 크게 영향을 받지 않고, 짧은 시간의 패턴 매칭 프로그램 분석으로 분쟁 해결의 시간도 지체하지 않을 것이다[13-15].

2.3 소프트웨어의 디지털 라이센스

일반적인 DRM 기술은 디지털 저작물인 바이너리 코드를 패키징을 하여 디지털 저작권 관리가 이루어지며, 디지털 저작물의 원천 요소인 소프트웨어 소스 코드에 대해서는 암호화와 임의적인 접근제어 기술이외에는 별다른 방법이 존재하지 않는 연구 초기 단계이다. 소프트웨어 소스 코드에 대한 DRM 기술이 연구 초기 단계인 이유는 소스 코드 자체가 아스키코드나 유니코드로 구성되어 있기 때문이다. 이런 이유로 인하여, 암호화를 통한 1차적 거래에만이 1:1 관계의 디지털 저작권 관리가 이루어지고 있다. 암호화 방법 역시 프로그램 소스 코드 개발자와 프로그램 소스 코드 구입자간의 1차적인 거래에만 저작권 관리를 지원할 뿐 그 이후에 발생하는 거래에 대한 저작권을 보호해 주지 못하고 있다.

일반적으로 소프트웨어 등록 협회에 소프트웨어를 등록하면 하드 카피인 등록증 하나가 배달된다. 최근에는 좀 더 개선되어 웹을 통해서 소프트웨어 등록증을 출력할 수 있다. 그렇지만, 정작 분쟁이 생기면 이를 해결하기 위한 절차가 복잡해진다. 단지, 소프트웨어를 보관하고 있다가 분쟁이 발생하면 복사본을 제공받으며, 먼저 등록한 사람에게 법정에서 우선권을 주장할 수 있는 근거를 제공할 뿐 등록한 소프트웨어에 대한 좀 더 많은 정보를 제공하지는 못하고 있다. 하드 카피의 단순한 등록증에서 탈피하여 하드 카피의 내용에 부가적으로 소프트웨어의 소스코드에 대한 많은 정보를 제공할 수 있는 XML 형태의 디지털 라이센스를 설계 및 구축하였다.

소프트웨어를 이해하기 위해서는 코드와 아키텍처를 보다 면밀하게 구분할 필요가 있다. 코드 또는 소프트웨어는 컴퓨터에 의해서 수행되어질 프로그램을 구성하는 벽돌과 시멘트에 해당한다. 한편 아키텍처는 코드를 벽돌 삼아 쌓아올린 구조물을 의미하는 것이다. 코드는 행위에 제약을 가할 수도 있고 구조에 영향을 미치기도 한다. 그렇지만 모든 것이 오로지 코드에 의해서만 결정되는 것은 아니다. 본 논문에서는 소프트웨어 소스코드의 디지털 라이센스를 생성하는 과정을 [그림 3]에 나타내었다[16]. 생성된 디지털 라이센스를 이용하여 소스코드의 분류 가능성에 대해 연구하였다.

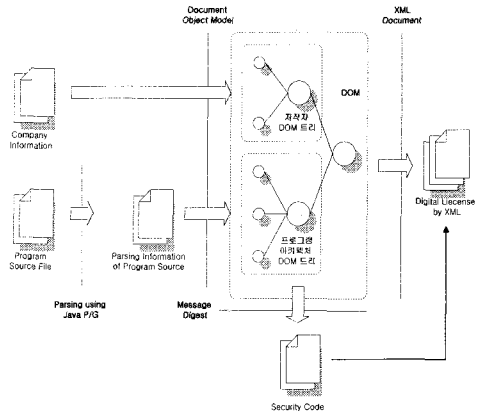

그림 3. 소프트웨어 소스코드의 디지털 라이센스 생성 과정

(1) 디지털 라이센스를 위한 DOM 트리의 구성

DOM 모델은 문서 객체들의 표준 집합을 통하여 응용 프로그램들에 대한 플랫폼 중립적이며 언어 중립적인 인터페이스를 정의한다. 본 연구에서는 소프트웨어 소스 코드의 디지털 라이센스 생성에 DOM 모델을 적용하며, 소프트웨어 소스코드의 저작자에 대한 정보를 갖는 저작자 DOM 트리와 소프트웨어 소스코드의 아키텍처를 나타내는 소프트웨어 아키텍처 DOM 트리로 [그림 4]와 같이 생성된다[16].

소프트웨어 소스 코드에 대한 암호화 기법은 단지 패키징 기법 중의 하나일 뿐이다. 본 연구에서는 디지털 라이센스를 이용한 분류 가능성을 제공하기 위해서 소프트웨어 소스 코드를 파싱하여 32개의 예약어, 라이브러리 그리고 함수 등을 DOM 트리로 [그림 4]와 같이 디지털 라이센스를 생성하였다. C언어 프로그램 소스 코드의 예를 들어, 모든 C 언어의 프로그램은 main() 함수와 서브 함수로 구성된다. 그러므로, main() 함수 또는 서브 함수가 디지털 라이센스의 프로그램 아키텍처 부분의 DOM 트리의 루트 노드가 되며, 또한 서브 함수는 main() 함수의 자 노드로 구성된다. 각 노드에는 사용된 라이브러리와 변수들 자료 형, 연산자들의 패턴 정보들을 갖는다.

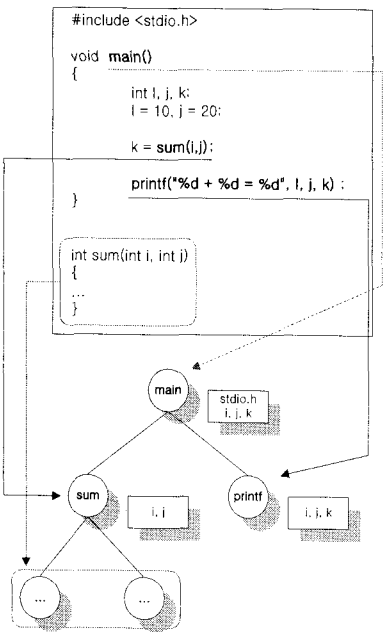

그림 4. 소프트웨어 소스 코드의 소프트웨어 아키텍처 DOM 트리의 변환 과정

(2) 노드 패턴, 연산자 그리고 자료 형 패턴 생성

소프트웨어 소스코드의 디지털 라이센스를 생성 시에는 최상의 루트 노드는 디지털 라이센스이다. 루트 노드의 밑에는 여러 노드가 존재할 수 있다. 노드는 소스코드를 파싱하여 생성된 예약어 토큰과 소프트웨어 구조에 의해서 계층 구조의 아키텍처로 [그림 4]와 같은 형태로 생성된다. 그 외에도 중괄호("{", "}")로 나타내어지는 블록을 이용한 자 노드 생성, 제어문인 조건문, 반복문 그리고 분기문에 의한 자 노드 생성 그리고 함수의 호출에 의한 자 노드를 생성한다. 그리고 함수의 코드 중에 연산자가 존재하지 않는 경우에는 노드를 생성하지 않는다. 모든 노드들에는 노드만이 갖는 패턴을 생성하게 된다. 노드에 나타내는 패턴은 입력된 자료 형과 수, 연산자 그리고 출력된 자료 형과 개수로 노드의 패턴을 구성하게 된다. 노드의 패턴 데이터를 이용하게 되면, 프로그램의 아키텍처가 동일하더라도 노드의 패턴 데이터로 프로그램의 기능이 다름을 구분할 수 있게 된다. 즉, 노드의 입력 패턴 수와 데이터 형태, 연산자 패턴, 출력 패턴 수와 데이터 형태로 표현되는 노드를 표현함으로 소프트웨어의 아키텍처 패턴 정보만을 비교하는 단점을 극복할 수 있게 되며, 소프트웨어의 소스코드의 분류 가능 능력을 향상시키게 된다.

III. 소프트웨어 소스코드 vs. 디지털 라이센스의 비교와 디지털 라이센스의 분류 3단계

소프트웨어 소스코드의 저작권 관리를 위한 보호 기술은 소극적인 보호 기술과 적극적인 보호 기술로 구분지울 수 있다. 소극적인 저작권 보호기술은 소스코드의 디지털라이센스 표현 기술, 소스코드의 디지털라이센스의 색인 및 검색 기술, 그리고 디지털라이센스의 분류 기술 등이다[13-21]. 그리고 적극적인 저작권 보호기술은 소스코드의 수동적인 파일형태에서 탈피하여, 능동적이며, 객체와 네트워크 기반 기술의 크립텍스 모델을 제안한다.

소프트웨어 소스코드의 저작권을 표현하는 디지털 라이센스는 상호간에 동작할 수 있는 공통적인 문서 구조인 DOM 모델을 적용하며, 바이너리 파일이 아닌 XML 파일로 기술 및 생성하며, XML 문서를 DOM 기술로 처리할 수 있다는 장점을 갖는다[17]. 소프트웨어 소스코드를 파싱하여 의미를 부여할 수 있는 최소 단위가 토큰이다. 이 토큰은 각각 독립적인 유닛을 나타낸다. 소프트웨어 소스코드를 구성하는 토큰을 이용하여 디지털 라이센스의 인덱스 패턴정보, 노드 패턴 정보 그리고 아키텍처를 구성하게 된다. 간단하게, 소스코드를 분류하기 위해서는 예약어와 함수 명으로 된 디지털 라이센스의 인덱스 패턴 정보에 의해서 소스코드를 분류하여 그룹화 할 수 있다. 인덱스 패턴 정보가 동일한 경우에는 어느 정도 유사한 프로그램으로 간주할 수 있다. 그래서, 노드로 구성된 프로그램 아키텍처에 의해서 프로그램을 분류할 수 있으며, [그림 4]와 같이 DOM을 이용한 추상 코드 아키텍처(ACA : Abstract Code Architecture)를 생성한다. 아키텍처는 코드 블록으로 된 노드 요소로 이루어진 가상의 구조물을 의미한다. 아키텍처 패턴 정보에 의해서도 분류정보를 제공할 수 있다. 인덱스 패턴과 프로그램 아키텍처가 동일한

경우에는 거의 동일한 프로그램이라고 할 수 있다. 그러나 프로그램도 생명주기를 갖기 때문에 과거의 프로그램의 패턴을 상속될 수 있다. 이런 경우를 객체의 버전업되었다고 하는데, 노드의 패턴 정보에 의해서 버전업 정보를 제공할 수 있다.

3.1 소스코드 vs. 디지털 라이센스의 비교

(1) 소스코드와 디지털 라이센스의 파일 크기 비교

통신 및 컴퓨팅 환경에서의 파일 크기에 대한 오버헤드 부분의 분석을 위해서 소프트웨어 소스코드의 크기와 디지털 라이센스의 크기에 대한 비교는 통신 및 저장 공간의 오버헤드 측면에서 필수적이다. 시뮬레이션 환경은 Pentium 4 3GHz, 512MByte의 메모리, 그리고 J2SDK 1.4.2_10 버전을 이용하였으며, 수행 결과 그래프를 Matlab을 이용하여 나타냈다. 소스 코드의 크기가 커질수록 디지털 라이센스의 크기를 나타내는 데이터 값의 진동의 폭은 더 커지면서 발산하는 경향을 [그림 5]에서 보였다.

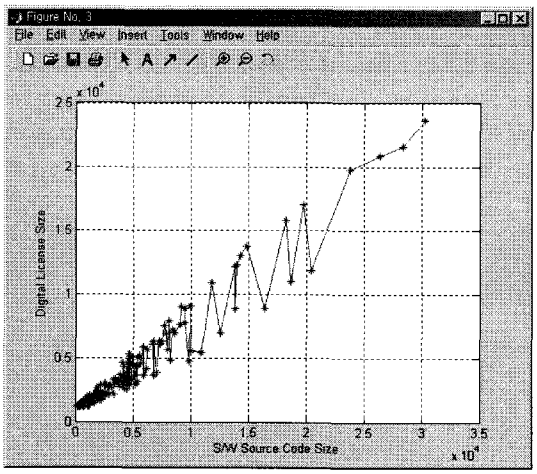

그림 5. 소스 코드와 디지털 라이센스의 파일 크기에 따른 비교 그래프

[그림 5]의 파일의 크기가 커질수록 주기적 진동에서 발산하는 경향을 갖는 이유는 개발자에 의해서 부가되는 코딩 스타일과 주석에 의해서 잡음이 부가된 것으로 추측된다. 같은 크기의 소스 코드도 특징 선택 항목을 무엇으로 결정하느냐와 개발자의 코딩 스타일 등의 소스 코드 개발 환경에 의해서 디지털 라이센스와 소스코드간의 진동에 영향을 미쳤다. 디지털 라이센스의 기초정보(저작자에 대한 정보)의 payload 부분은 변동이 없으나, 소프트웨어의 아키텍처를 구성하는 부분에서 많은 변동이 발생하였다. 디지털라이센스를 생성할 패턴의 원천이 되는 특징 선택을 어떻게 결정하는가에 의해서 소프트웨어의 아키텍처 부분의 진동에 의한 발산을 줄일 수 있을 것이다.

(2) 소스코드와 디지털 라이센스의 생성시간 비교

소프트웨어의 소스코드를 이용하여 디지털 라이센스를 생성하는데 시간과 코드의 크기를 극좌표를 이용하여 [그림 6]에 표현하였다.

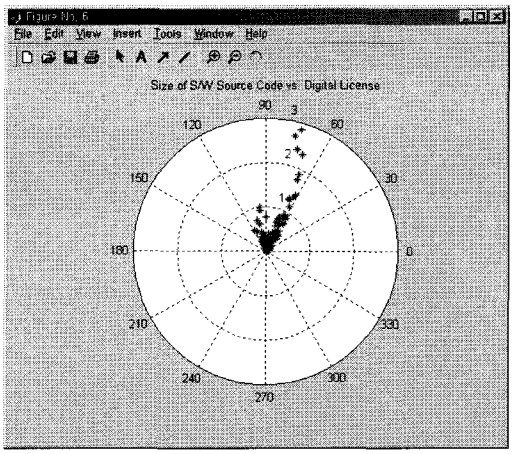

그림 6. 시간 축에 의한 소프트웨어 소스코드와 디지털 라이센스의 비교 및 분포

[그림 6]의 반지름은 디지털 라이센스를 생성하는 데 소요된 시간을 나타낸다. 소스코드의 크기가 클수록 디지털 라이센스를 생성하는 데 많은 시간이 소비되었다. 소요시간이 1초미만이 경우와 극좌표의 90도와 120도 사이에 존재하는 자료는 소스코드의 크기보다는 디지털 라이센스의 크기가 큰 경우이다. 극좌표의 120도에 근접할 수 록 디지털 라이센스가 소스크기보다 상당히 크다는 것을 나타내며, 90도에 근접할 수 록 소스코드의 크기와 디지털 라이센스의 크기가 같은 경우이다.

또한 90도 보다 작은 경우에는 소스코드보다는 디지털 라이센스가 작은 경우이다. 소스코드의 크기가 커질수록 디지털 라이센스가 60도에 근접하는 것을 관측할 수 있다.

3.2 디지털 라이센스의 분류 3단계 과정

디지털 라이센스를 구성하는 인덱스 패턴, 소프트웨어 아키텍처, 노드의 패턴 정보 그리고 메시지 축약과 암호화 등의 다양한 이러한 패턴 정보는 소프트웨어의 소스코드를 분류하거나, 클러스터링 그리고 검색할 수 있는 정보를 제공한다. 더 나아가서는 소프트웨어 버전 관리를 위한 정보도 제공할 수 있다. 소프트웨어 소스코드의 분류 및 검색 단계는 [그림 7]과 같이 3단계로 구성된다. 검색 및 분류를 위한 단계를 진행 할수록 좀 더 세밀한 분류가 이루어진다.

[단계 1] 인덱스 패턴 정보에 의한 분류
[단계 2] 소프트웨어 아키텍처 패턴에 의한 분류
[단계 3] 노드의 패턴 정보에 의한 분류

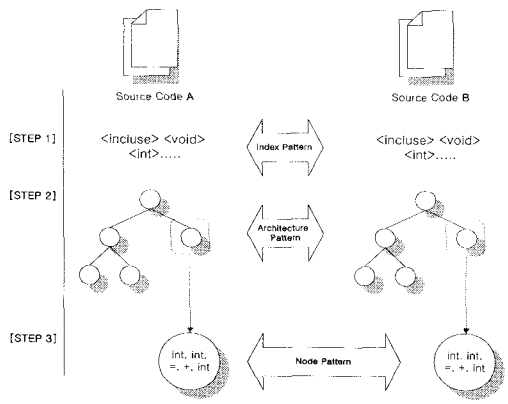

그림 7. 디지털 라이센스에 의한 분류 3단계 과정

단계 1의 수준은 인덱스 패턴에 의한 클러스터링 분류 정보를 제공한다. 이를 위해 인덱스 패턴은 소프트웨어 소스코드에 사용된 예약어의 종류와 순차 정보로 구성된다. 인덱스 패턴 정보에 의해서 동일한 예약어를 갖는 소프트웨어 소스코드를 분류가능하다. 그러나 인덱스 패턴 정보도 임의의 정도에서는 한계를 드러낼 것이다. 이에 부가적으로 단계 2의 수준은 거시적인 수준의 계층구조의 아키텍처를 검색한다. DOM을 이용한 추상 코드 아키텍처(ACA) 패턴 정보를 이용하면 같은 예약어 그룹으로 구성된 소프트웨어 소스코드도 계층구조의 소프트웨어 아키텍처에 의해서 그래픽 형태의 거시적인 분류 정보를 제공한다. 한 단계 더 나아가서, 단계 3의 수준은 미시적인 수준의 단말 노드의 패턴 매핑 및 검색한다. 같은 계층 구조의 소프트웨어의 아키텍처를 갖더라도 아키텍처를 이루는 노드의 패턴 정보에 의해서 입력 데이터의 수와 타입, 연산자의 순차나열 그리고 출력 데이터의 수와 타입에 의해서 더 세밀하게 분류 패턴 정보를 제공할 수 있다.

3.3 인덱스 패턴에 의한 분류

(1) 인덱스 패턴에 의한 시각적 분류 정보

디지털 라이센스에 의해서 사용된 예약어의 분포를 알 수 있다. 다음의 [그림 8]은 시뮬레이션에 사용된 170여개의 소스코드의 디지털 라이센스에 사용된 예약어의 분포와 이를 이용한 패턴을 시각적으로 보여준다.

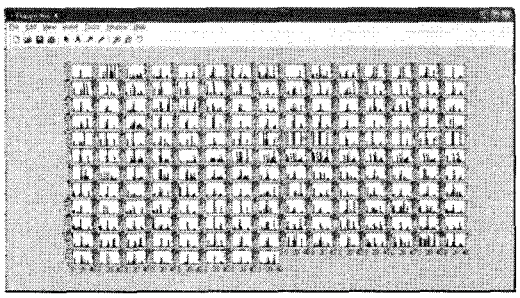

그림 8. 소프트웨어 소스코드의 디지털 라이센스를 이용한 예약어의 분포 및 패턴

(2) 인덱스 패턴에 의한 계층도

디지털 라이센스는 소스코드를 분석하면서 사용된 예약어를 순서에 의해서 중복되지 않게 인덱스 패턴을 생성한다. 이러한 정보에 의해서 하나의 예약어에 의한 계층도를 [그림 9]와 같이 추출할 수 있다. [그림 10]은 CASE를 첫 번째로 포함하는 그룹에 대한 인덱스 패턴의 수의 증가에 따른 분류 율을 나타내었다.

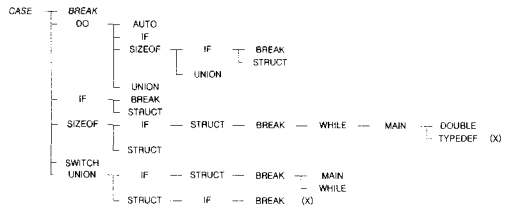

그림 9. 디지털 라이센스의 예약어를 이용한 색인 정보에 의한 추출된 CASE 계층도

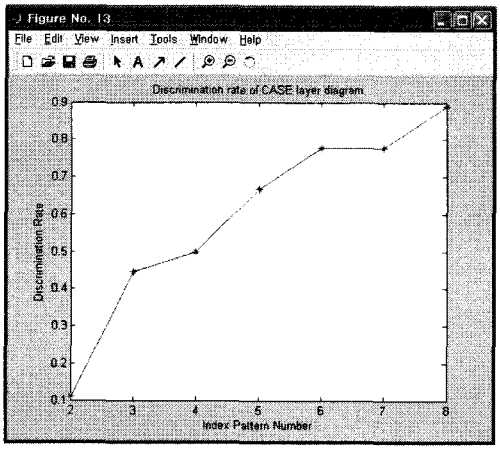

그림 10. CASE 그룹의 인덱스 패턴 수의 증가에 따른 디지털 라이센스의 분류율

(3) 인덱스 패턴 수에 의한 분류 성능

C 언어의 소스코드 샘플 170여개를 이용한 시뮬레이션을 수행하였다. 샘플 170여개는 모두 동일하지 않는 프로그램으로 구성되었으며, 샘플들을 100% 분류하지는 못했다. [그림 11]에서는 인덱스 패턴 수를 증가하면 미분류된 패턴의 수응 감소하여 분류 율은 향상되었음을 [그림 11]과 [그림 12]에 나타내었다.

3.4 디지털 라이센스의 아키텍처와 노드 정보에 의한 분류

먼저, 임의의 소프트웨어 A와 B라는 두 개의 C 언어로 작성된 소스 코드를 이용하여 A와 B의 디지털 라이센스를 발행하였다. [그림 13]은 A와 B의 소스 코드를 이용하여 발행한 디지털 라이센스를 자체 개발한 XML TreeView[17]에 의해서 나타낸 것이다. [그림 13]의 A와 B의 두 소스 코드는 거의 같은 트리 구조와 아키텍처 패턴을 가졌으나, 단말 노드 부분에서 두 소스 코드의 디지털 라이센스의 아키텍처가 다르게 표현되었으며, 더불어 노드의 패턴 정보도 다르게 표현되었다. [그림 13]의 왼쪽의 윈도우는 소프트웨어 소스코드의 아키텍처를 나타내며, 오른쪽 윈도우는 왼쪽 윈도우의 임의의 노드를 선택하였을 때 해당 노드의 패턴 정보를 보여주는 윈도우이다. 분쟁 해결을 위해서는 소소 코드의 패턴 매칭에 의한 방법도 있지만, 트리 구조로 표현된 소스코드의 아키텍처를 보면 전반적인 소프트웨어 소스코드의 조감도를 비교할 수 있는 장점을 갖으며, 분쟁 해결에 도움이 될 것이다. 인덱스 패턴은 프로그램을 구분 및 분류하는데도 이용되지만, 프로그램의 버전 관리에도 필요한 정보를 제공하는데 사용될 수 있다. 임의의 프로그램이 개발되어 유지보수 되면 버전 업 과정이 이루어진다. 버전 업 과정을 위해서는 최소한 인덱스 패턴은 동일하더라도 프로그램 아키텍처나 노드의 패턴이 동일해서는 안된다. 프로그램의 버전 업은 과거의 프로그램 버전과는 많은 부분은 동일하더라도 일부분에서는 노드 패턴의 변경, 노드 위치의 변동, 새로운 노드의 추가와 전 버전의 노드 제거 등의 과정이 나타나게 될 것이다. 이러한 정보에 의해서 프로그램의 버전 업 관리가 이루어지게 된다.

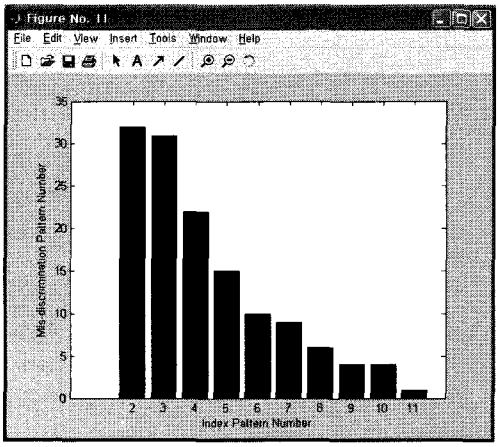

그림 11. 인덱스 패턴 수에 의한 미분류된 디지털 라이센스의 수

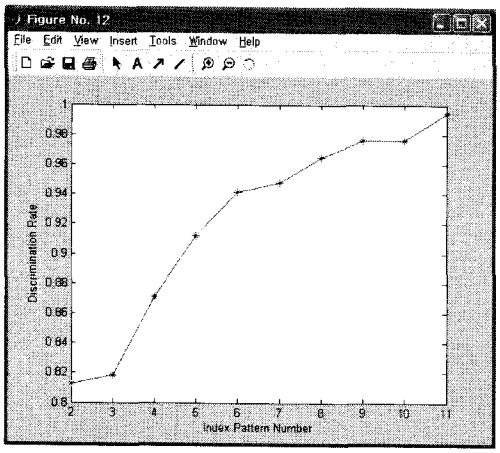

그림 12. 샘플데이터의 인덱스 패턴 수 증가에 따른 디지털 라이센스의 분류 율

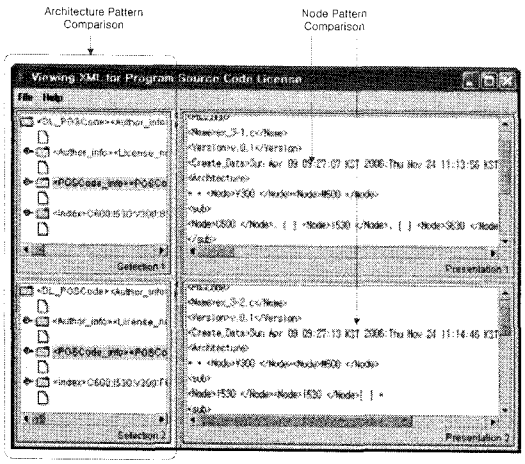

그림 13. 두 개의 디지털 라이센스에 의한 소스 코드의 아키텍처와 노드의 패턴 정보 비교

Ⅳ. 크립텍스 모델 제안

Web 2.0 또는 시맨틱 웹(Semantic Web)은 인터넷의 개체들을 단지 수동적인 객체에서 탈피하여 분산된 컴퓨팅 시스템에 의한 자동화 및 능동적인 객체로 전환하는 시대이며, 정보화시대의 직접적으로 중요한 인프라의 조립라인에 해당하는 소프트웨어의 소스코드에 대한 관리 및 보안측면은 아직도 초보단계에 머물러있다. 소프트웨어의 소스코드를 보호하기 위한 제반적인 지원 기술과 프레임워크는 너무나도 빈약한 상태이다.

크립텍스(CRYPTEX)는 '다빈치 코드'라는 소설에 잠깐 언급되는 비밀문서를 보관하는 하나의 보안 장치 이름이며, 여기서 아이디어를 얻어서 동일한 명칭으로 쓴다. 크립텍스는 문자조합에 의한 암호화된 키를 갖으며, 암호를 풀지 못하여 임의적으로 문서를 여는 경우에는 크립텍스 안의 초산이 파피루스로 만들어진 비밀문서를 녹여버리는 보안 장치이다. 이러한 현실세계의 보안 장치를 사이버 상의 보안 장치로 크립텍스라는 보안 모듈을 명명하고자 한다.

본 연구에서는 소극적인 보호기술인 디지털 라이센스 기술에서 발전시켜 적극적인 보호 기술인 크립텍스 모델을 제안한다. 제안하는 크립텍스 모델은 인증 및 인증되지 않은 주체로부터 객체인 소프트웨어의 소스 코드를 안전하게 보호 및 접근제어를 지원하기 위한 제반 기술들의 집합을 통칭하는 모델이다. 크립텍스를 이용하여 단지 수동적인 문서 상태의 소프트웨어 소스코드에 대해서 능동적이며, 접근제어 및 보안이 가능하며, 이동 및 위임기능을 부여할 수 있는 크립텍스 모델을 [그림 14]와 같이 제안한다.

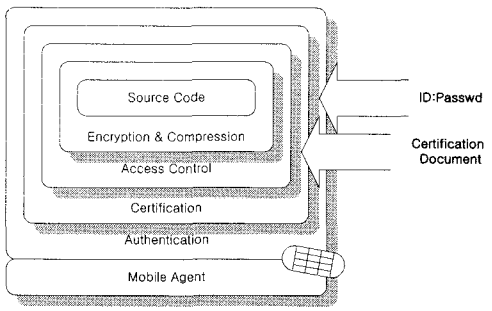

그림 14. 크립텍스 모델

크립텍스 = 인증서 + 접근제어 + 이동 에이전트 + 보안 및 압축 기능

4.1 크립텍스(CRYPTEX)의 기능

크립텍스의 핵심 기능은 소프트웨어 소스코드의 관리와 이를 접근제어하기 위한 알고리즘으로 구현된 이

동에이전트(Mobile Agent) 소프트웨어이다. 소프트웨어의 소스코드는 코드 자체가 아스키코드 또는 유니코드로 이루어졌으며, 프로그래밍 언어로 소프트웨어가 수행할 작업들이 기술되어 있는 수동적인 문서 상태이다. 능동적인 일을 수행할 수 있는 프로세스 상태가 아니기 때문에 이를 보완하기 위한 기능을 갖는 크립텍스에 포함시켜서 이동, 관리 및 접근제어를 수행하게 된다.

(1) 소스 코드의 이동 및 관리 기능

크립텍스 자체는 의미가 없으며, 소스코드를 관리하기 위해서 능동적인 관리시스템(캡슐 기능)의 역할 기능 수행과 네트워크를 통한 이동 기능을 갖는다. 관리 기능으로는 소스코드의 캡슐 기능, 인증서 관리 기능, 압축 기능, 디지털라이센스 기능, 인증 기능, 접근제어의 매칭 및 권한부여 기능, 그리고 네트워크를 통한 이동 기능을 갖는다. 이동 소프트웨어 에이전트(Mobile Software Agent)란, 사용자 대신 자치적(autonomous fashion)으로 네트워크에서 자유롭게 떠돌 수 있는 프로그래밍 개체(entity)를 일컫는다. 이동 에이전트는 자치성을 가지고 대신 동작하며, 어느 만큼의 순향성(proactivity)과 반응성(reactivity)을 가지고 있으며, 학습능력, 협력능력, 이동능력 등의 특성을 갖는다.

(2) 소스 코드의 보안 기능

인증 단계의 완료후의 사용자와 소스코드에 대한 접근제어, 접근 레벨에 의한 소스코드의 접근 제어 기능을 수행하게 된다. 접근제어 기능으로는 접근 정책과 레벨, 위임, 인증서의 시한 기능, 암호 기능 등을 수행한다.

4.2 주체/객체의 인증서

주체/객체의 인증서는 소스코드에 대한 디지털 라이센스를 포함한다. 주체는 개발자 및 사용자를 나타내며, 객체는 크립텍스에 포함된 소스코드이다. 디지털 라이센스는 소스 코드에 대한 노드와 트리로 구성된 아키텍처 정보와 개발자의 정보로 구성되어 있다. 디지털라이센스는 직접 소스코드를 열어보지 않더라도 소스코드에 동등한 정보를 제공하는 기능을 제공한다. 디지털라이센스에 의해서 소스코드를 분류 및 검색, 매칭이 가능하고, 효과적인 정보제공으로 지능적인 응용프로그램도 개발 및 지원이 가능하다. 주체의 인증서에 포함된 디지털라이센스와 객체의 인증서에 포함된 디지털라이센스의 내용에 의한 매칭이 이루어져야 주체/객체의 인증서의 결합이 완료된다. 또한 주체/객체의 인증서는 인증서의 유효시한이 지정되어 있다.

4.3 크립텍스의 물리적 접근제어 모델

과거의 소프트웨어 소스코드를 대변하는 디지털 라이센스의 수동적 인증에서 진보하여, 소프트웨어 소스코드의 능동적인 접근제어가 필요하다. 소프트웨어 소스코드는 아스키코드(ASCII code)나 유니코드(Unicode)에 있어서, 능동적인 접근제어가 불가능하다. 주체의 ID:Passwd와 주체/객체의 인증서에 의해서 객체인 크립텍스는 접근제어를 수행하게 된다. 먼저, 주체의 ID:Passwd는 주체의 신원을 식별하고, 책임추궁성이 시작되는 과정이다. 주체와 객체의 인증서에는 소프트웨어 소스코드에 대한 디지털 라이센스와 크립텍스의 접근제어 정보로 구성된다.

(1) 인증 단계

인증 단계는 소프트웨어 소스코드의 접근에 따른 책임추궁을 위한 신원 확인단계이다. 신원 확인은 인증기관에서 발급된 ID와 Passwd에 의해서 주체의 신원과 접근제어를 위한 레벨이 결정된다.

(2) 접근제어 단계

인증 단계의 완료에 의해서 주체의 신원이 확인되면, 이어서 발급된 주체와 객체의 인증서에 의해서 접근제어 단계가 수행된다. 접근제어의 정책 및 레벨은 크립텍스에 포함된 소스코드에 접근하기 위해서는 CA에 초기 등록자에 의해서 정책 수립, 레벨 부여, 위임 부여 그리고 판매 전략과의 매핑이 설정된다. 크립텍스의 접근제어는 크게 주체와 객체로 구분된다. 주체와 객체의 두 인증서에 의해서 레벨에 따른 접근 제어가 수행된다. 접근제어는 주체는 개발자, 사용자, 판매자, 구매자

등으로 구분되며, 객체는 판매의 전략과 접근제어에 의한 여러 가지 제약이 부여된다. 주체와 객체의 레벨이 같은 경우에는 해당 레벨에 맞는 접근제어가 이루어지면 된다. 그러나 주체와 객체의 레벨이 다르면 접근제어의 충돌이 발생하게 되며, 이를 처리하기 위한 예외처리가 준비되어야 한다.

- 객체의 레벨이 높은 경우 -> 주체의 레벨에 따른 접근제어 수행 [그림 15].
- 주체의 레벨이 높은 경우 -> 객체의 레벨에 따른 접근제어 수행 -> 인증기관으로부터 주체의 레벨에 해당하는 객체인 크립텍스를 새로 생성하여 다운로드하여 접근제어를 수행

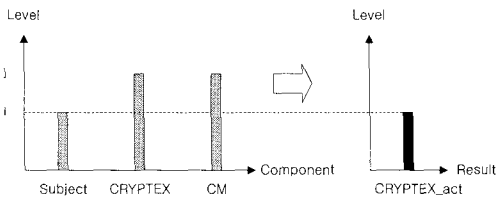

그림 15. 접근제어의 충돌 예제의 해결

V. 결론 및 향후 연구

디지털 저작물의 원천 요소인 소프트웨어 소스 코드에 대해서는 암호화이외에는 별다른 방법이 존재하지 않으며, 아직은 연구 초기 단계이다. 암호화 방법 역시 소프트웨어 소스 코드 개발자와 소프트웨어 소스 코드 구입자간의 1차적인 거래에만 저작권 관리를 지원할 뿐 그 이후에 발생하는 거래에 대한 저작권을 보호주지 못하고 있다. 소프트웨어 소스코드의 소유권 분쟁이 발생 시 소유권을 증명하기 위해서는 원본의 소프트웨어 소스코드를 공개해야만 하는 문제점을 갖고 있다. 본 논문에서는 소프트웨어의 소유권 주장에 하드카피 이외에 더 많은 정보를 부여하며, 컴퓨팅 환경에서 자동으로 처리를 위한 프로토타입을 연구하였다. 소프트웨어 소스코드의 원본 공개로 인한 저작권 침해와 기술 유출을 막기 위한 소프트웨어 소스코드의 디지털 라이센스의 분류 가능성에 대해서 연구하였다.

연구 내용으로는 소프트웨어 소스코드와 디지털 라이센스의 크기 비교와 생성시간에 대해 비교하였으며, 디지털 라이센스의 인덱스 패턴 정보에 의한 예약어와 함수명에 의한 그룹핑과 계층도 추출, 그리고 인덱스 패턴의 수에 따른 분류 율을 시뮬레이션 하였다. 또한 예제를 통해 아키텍처 패턴 정보와 노드 패턴 정보에 의해서 소스 코드를 구분할 수 있음을 보였다. 특별히, 소스코드의 아키텍처와 노드 정보를 이용하여 소프트웨어 소스코드의 거시적/미시적 조감도를 비교할 수 있으므로 분쟁 해결에 도움이 될 것이다. 마지막으로 소스코드를 보호하기 위한 크립텍스 모델을 제안하였다.

향후 연구로는 디지털 라이센스는 수동적인 저작권 관리를 위한 도구라면, 크립텍스는 소프트웨어 소스코드를 보호하기 위한 능동적인 도구가 될 것이다. 소스코드를 보호하기 위한 크립텍스를 구현하기 위한 연구가 필요하다.

참 고 문 헌

[1] R. Matthias and D. Stephane, "Visual Detection of Duplicated Code," Proceedings ECOOP Workshop on Experiences in Object- Oriented Re-Engineering, 1988.

[2] Y. Wu, "Identifying Syntactic Differences Between Two Programs," Software-Practice and Experience, Vol.21, No.7, pp.739-755. 1991(7).

[3] http://www.few.vu.nl/~dick/sim.html

[4] http://glimpse.arizona.edu/javadup.html

[5] http://www.ccsr.cam.ac.uk/~mw263/YAP.html

[6] http://www.cs.berkeley.edu/~aikem/moss.html

[7] 조동욱, "디지털 재산권 보호를 위한 S/W 프로그램 표절 감정 기술과 툴의 분석", 한국콘텐츠학회 2003 춘계종합학술대회 논문집, p.177-184, 2003.

[8] http://www.pdmc.or.kr/

[9] Intel, "Content Protection in the Digital Home,"

Vol.6, Issue 4, Intel Technology Journal, 2002(11).

[10] R. Michael, "Utilizing Content Protection Technologies," Intel Developer Forum, 2002(9).

[11] M. E. Ahmet, "Multimedia Protection in Digital Networks," CNIS 2003, 2003.

[12] http://www.fasoo.com

[13] 차병래, 특허 출원 : 10-2006-46156, *소프트웨어 소스 코드의 저작권 관리방법*, 2006(5).

[14] 차병래, "소프트웨어 소스코드의 저작권 관리를 위한 디지털 라이센스의 색인 및 검색에 대한 연구" 한국콘텐츠학회 논문지, pp.21-31. 2007(1).

[15] 차병래, 특허 등록 : 10-0740222, *소프트웨어 소스 코드의 저작권 관리방법*, 2007(7).

[16] 차병래, 김형종, 이동섭, "프로그램 소스 코드의 원본을 공개하지 않는 소유권 증명을 위한 디지털 라이센스 설계", 한국콘텐츠학회 논문지, pp.29-37, 2006(4).

[17] 차병래, 소프트웨어 등록 : 2006-01-185-000879, *프로그램 소스코드의 디지털 라이센스 뷰어*, 2006(2).

[18] 차병래, 소프트웨어 등록 : 2006-01-169-001840, *프로그램 소스코드 디지털 라이센스의 색인 생성기*, 2006(4).

[19] 차병래, 소프트웨어 등록 : 2006-01-169-001841, *프로그램 소스코드 디지털 라이센스의 검색기*, 2006(4).

[20] B. R. Cha, "Business Model and Comparasion of S/W Source Code vs. Digital License for IPRs," KES-AMSTA 2007, LNAI 4496, pp.825-832, 2007(5).

[21] B. R. Cha, K. J. Kim, and D. S. Lee, "Study of Digital License Search for Intellectual Property Rights of S/W Source Code," ICCSA 2007, LNCS 4707, Part III, pp.201-212, 2007(8).

저 자 소 개

차 병 래(ByungRae Cha) 정회원

- 1997년 2월 : 호남대학교 컴퓨터공학과(공학석사)
- 2004년 2월 : 목포대학교 컴퓨터공학과(공학박사)
- 2005년 3월 ~ 현재 : 호남대학교 컴퓨터공학과 전임 교수

<관심분야> : 컴퓨터 시스템 보안, 디지털 컨텐츠 보호

정 영 기(YoungKee Jung) 정회원

- 1986년 2월 : 서울대학교 전기공학과(공학사)
- 1994년 8월 : 한국과학기술원 전기전자공학과(공학석사)
- 2003년 8월 : 광주과학기술원 정보통신공학과(공학박사)
- 1999년 9월 ~ 현재 : 호남대학교 컴퓨터공학과 부교수

<관심분야> : 디지털컨텐츠, 3D영상처리