

---

## API의 저작권 보호에 관한 고찰\*

### —자바 API의 저작물성을 중심으로—

---

권사현\*\* · 박성필\*\*\* · 김용길\*\*\*\*

#### I. 서 론

#### II. 기술적 배경

1. 자바 언어, 자바 플랫폼, 자바 API의 개념
2. 자바 문법

#### III. 오라클 대 구글(Oracle v.

- Google) 사건 전개과정과 쟁점
1. 사건의 전개과정
  2. 사건의 법적 쟁점

#### IV. 자바 API의 저작물성

1. 아이디어-표현 이분법
2. '창조적 개성'의 인정여부
3. 저작권법 제101조의2 적용여부

#### V. API를 저작권법으로 보호하지 않을 경우의 대안으로서 부정경쟁방지법

1. API에 대한 법적 보호의 필요성
2. 부정경쟁방지법의 적용 가능성

#### VI. 결 론

---

\* 이 논문은 2017년 권사현의 한국과학기술원 공학석사학위 논문 “자바 API의 저작물성”의 일부를 수정 보완하였음.

\*\* 법무법인(유한) 대륙아주, 변호사, 공학석사(제1저자).

\*\*\* 한국과학기술원 지식재산대학원 책임교수, 미국변호사, 법학박사(교신저자).

\*\*\*\* 원광대학교 법학전문대학원 교수, 법학박사(제2저자).

## 초 록

오늘날 컴퓨터 및 소프트웨어의 급속한 발달과 사회와 문화의 다변화로 인하여 새로운 형태의 창작물이 발현되고 이들의 경제적 가치가 높아지게 되었다. 아이디어의 영역에 있던 창작물에 대한 보호 필요성이 대두되고 있는데, 대표적인 사례로서 ‘방송 포맷’, ‘게임의 진행방법’, ‘컴퓨터프로그램의 응용 프로그램 인터페이스’ 등이 논의되고 있다. 컴퓨터프로그램의 API의 저작권과 관련하여 현재 미국에서 자바 API의 저작권을 주장하는 오라클과 그 이용자인 구글 사이에 소송이 진행되고 있다. 자바(JAVA)의 영향력이 광범위하기 때문에 개발자 커뮤니티에서는 오라클과 구글 사이의 저작권 소송의 결과가 미칠 파장에 대한 우려가 많은 것으로 보인다. 특히 저작권 침해의 위험이 “언제 어디서나 기존의 것을 갖다 쓸 수 있는” API를 이용한 개발 환경의 특성을 망칠 것이라고 걱정하는 목소리가 높다. 미국 항소법원이 프로그램 개발환경의 현실을 충분히 고려하지 않고, 저작물성에 대한 경직된 이론을 고수하여 API의 저작물성을 인정한 것은 동의하기 어렵다. 선(Sun)과의 협상이 결렬된 이후 구글이 안드로이드 개발을 단념했다면 지금 우리가 보고 있는 많은 모바일 혁신이 진행되지 않거나 더디게 진행되었을지도 모른다. 우리 법의 해석상 API는 창작성이 없거나 저작권법 제101조의2에 따라 저작권 보호범위에서 제외되므로 저작권법의 보호를 받지 못한다고 생각된다. 이 경우 보호의 필요성이 있는 API는 부정경쟁방지법상의 보호를 모색하여야 할 것이다.

## 주제어

자바, API, 저작물성, 합체의 원칙, 필수장면의 원칙, 사실상의 표준, 인터페이스, 선언코드, 구현코드

## I. 서론

오늘날 컴퓨터 및 소프트웨어의 급속한 발달과 사회와 문화의 다변화로 인하여 새로운 형태의 창작물이 발현되고 이들의 경제적 가치가 높아지게 되었다. 즉 ‘아이디어-표현 이분법(idea-expression dichotomy)’에 따라 구분되는 아이디어의 영역에 있던 창작물에 대한 보호 필요성이 대두되고 있다. 대표적인 사례로서 ‘방송 포맷’, ‘게임의 진행방법’, ‘컴퓨터프로그램의 응용 프로그램 인터페이스(Application Programming Interface: 이하 ‘API’ 라고 한다)’ 등이 논의되고 있다.<sup>1)</sup> 컴퓨터프로그램의 API의 저작권과 관련하여 현재 미국에서 자바(JAVA) API의 저작권을 주장하는 오라클과 그 이용자인 구글 사이에 이른바 ‘스마트폰 전쟁’이라고 불리는 세기의 소송이 진행되고 있다.

이 소송의 쟁점은 자바 API의 선언코드(declaration line)<sup>2)</sup>와 자바 API의 구조·순서·조직(structure, sequence and organization: SSO)이 저작물성이 있는지 여부 및 저작물성이 있다면 구글이 오라클의 승낙 없이 37개 자바 API 패키지(Package)를 안드로이드 플랫폼에 복제하여 사용한 것이 공정이용에 해당하는지 여부이다. 이러한 자바 API 저작권 소송(이하 ‘이 사건’이라 한다)에서 제1심인 캘리포니아 연방지방법원(United States District Court, N.D. California, 이하 ‘지방법원’이라 한다)은 자바 API의 저작물성을 부정하여 오라클의 청구를 기각하였으나,<sup>3)</sup> 항소심인 연방항소법원(United States Court of Appeals, Federal Circuit: CAFC, 이하 ‘항소법원’이라 한다)은 자바 API의 저작물성을 인정하여 사건을 다시 파기 환송하였다.<sup>4)</sup> 이에 따라 파기 환송된 사건을 다시 심리한 지방법원은 구글의 공정이용 항변을 받아들여 다시 오라클의 청구를 기

1) 이상정, “미래 저작권 정책 방향 수립을 위한 연구”, 한국저작권위원회, 2016, 72면.

2) ‘선언코드’라는 용어는 본 논문에서 메소드와 클래스의 헤더(headers)를 의미한다. ‘헤더’가 기술적으로는 좀 더 정확한 용어이지만, 본 논문에서는 ‘선언코드’와 ‘헤더’라는 용어를 구분 없이 사용한다.

3) Oracle America, Inc. v. Google Inc. 872 F.Supp.2d 974 (N.D. Cal., 2012).

4) Oracle America, Inc. v. Google Inc. 750 F.3d 1339 (Fed. Cir., 2014).

각함으로써 자바 API 저작권을 둘러싼 법적 논쟁이 한층 가열되고 있다.<sup>5)</sup>

그런데 국내에서는 아직 API의 저작물성에 대한 본격적인 논의가 진행되고 있지 않고, ‘사용자 인터페이스(User Interface: UI)의 저작물성에 관한 로터스 판결<sup>6)</sup>을 소개하고 있는 것에 그치고 있는 것으로 보인다.<sup>7)</sup> 본 논문에서는 우선 자바와 자바 API에 대한 기술적 배경을 설명하고(II), 이 사건의 전개과정과 쟁점을 살펴본 후(III), 자바 API의 저작물성에 대하여 미국법과 우리 법의 관점에서 검토하고(IV), 우리법상 API의 적절한 보호방안은 무엇인지(V) 살펴보고자 한다.

## II. 기술적 배경

### 1. 자바 언어, 자바 플랫폼, 자바 API의 개념

#### (1) 자바 언어와 자바 플랫폼

자바 언어는 C와 C++와 같이, 사람이 읽을 수 있는 언어이다. 사람이 읽을 수 있는 코드인 ‘소스 코드(Source Code)’는 컴퓨터에 의해 읽히지 않는다. 오직 사람이 읽을 수 없는 ‘오브젝트 코드(Object Code)’만이 컴퓨터에 의해 사용될 수 있다. 대부분의 오브젝트 코드는 0과 1로만 구성된 이진 언어이다. 따라서 소스 코드로 작성된 컴퓨터 프로그램은 실행 전에 오브젝트 코드로 실행(컴파일)되어야 한다. 그런데 자바 언어는 다른 언어와 달리 소스 코드가 컴파일되면 ‘바이트 코드(Byte Code)’라는 중간 단계의 코드로 변환되

5) Daria Vasilescu-Palermo, “APIs and Copyright Protection: The Potential Impact on Software Compatibility in The Programming Industry”, *J. Marshall Rev. Intell. Prop. L.*, Vol.16(2016), p.153; Tyler J. Demasky, “Oracle V. Google: Setting a Standard pr Handicapping an Industry?”, *N.C. J.L. & Tech.* Vol.18 On,1(2016); Deba Alam, “Oracle America, Inc. V. Google, Inc.: The Battle Over APIs Continues”, *DePaul J. Art Tech. & Intell. Prop. L.*, Vol.26(2015), p.39.

6) Lotus Development Corp. v. Borland International, Inc., 49 F.3d 807 (1st Cir. 1995).

7) 이해완, 『저작권법』, 제3판, 박영사, 2015, 190면.

고, 이 바이트 코드는 다시 자바 ‘가상 머신’에 의해 이진 기계 코드(오브젝트 코드로) 변환되어 컴퓨터에서 실행된다. 이러한 바이트 코드와 ‘가상 머신’이 자바의 이식성을 확보해 주었다. 자바 언어로 작성된 프로그램이 실행되기 위한 소프트웨어로서 ‘자바 플랫폼’은 자바 가상머신(Java Virtual Machine: JVM)과 자바 API로 구성된다. 이하에서 언급되는 ‘자바’는 특별한 경우가 아니면 ‘자바 플랫폼’을 의미한다.

## (2) 라이브러리와 API

프로그래밍을 하다 보면 반복되는 부분을 함수로 만들 때가 있다. 그런데 이 함수가 프로그래밍에서 반복적으로 사용된다면, 즉 종전에 프로그래밍을 할 때 만든 덧셈 함수가 그 후 프로그래밍하는 데 다시 필요하다면 덧셈 함수를 또다시 만들 필요는 없을 것이다. 똑같은 함수를 다시 만들 필요 없이 그 함수를 라이브러리로 사용함으로써 더욱 효율적인 프로그래밍이 가능하게 될 것이다. 결국 라이브러리는 함수들의 집합이라고 할 수 있는데, 미리 만든 함수를 모아 놓은 것이다. 도서관에서 필요한 책을 빌리듯이 라이브러리에서 필요한 함수를 찾아 쓸 수 있기 때문에 ‘라이브러리’라고 하는 것이다. 그리고 이러한 라이브러리에 접근하기 위한 규칙들을 정의한 것을 API라고 한다.<sup>8)</sup> API는 프로그래머가 라이브러리에 있는 여러 함수를 이용하여 프로그램을 작성할 때 해당 함수의 내부 구조를 알 필요 없이, 단순히 API에 정의된 대로 입력 값을 넣고 결과 값을 사용할 수 있게 해 주는 것이다.<sup>9)</sup>

자바 언어를 만든 선 마이크로시스템즈(Sun Microsystems, Inc., 이하 ‘Sun’ 또는 ‘선’이라 한다)는 즉시 사용이 가능한 자바 프로그램들(라이브러리)을 작성하고, 이들 프로그램들을 ‘패키지’라고 불리는 것으로 그룹화시켜 조직화하였다. Sun은 자바 언어에서 특정 동작(기능)을 위한 함수를 소위 ‘메소드

8) 예를 들어 식당에 가서 메뉴판을 보고 직원에게 주문하는 상황에 비유하자면 주방을 라이브러리, 메뉴판을 API라고 할 수 있다.

9) 대표적으로 윈도우(Windows)와 같은 운영체제가 제공하는 운영체제 API는 응용 프로그램이 운영체제의 다양한 기능을 사용할 수 있게 해 준다.

(Method)’로 정의하고, 특정한 메소드들과 변수들로 구성된 것을 ‘클래스(Class)’라고 정의하였다. Sun은 클래스들을 다시 ‘패키지’로 그룹화시켰다.<sup>10)</sup> 이러한 ‘패키지 - 클래스 - 메소드’ 구조화는 개발자들에게 응용 프로그램 인터페이스(API)를 제공한다. 자바가 1996년에 처음 도입되었을 때, 자바 API는 미리 작성된 8개의 패키지를 포함하고 있었다. Sun에 따르면 이들 패키지 중 적어도 3개 패키지(java.lang, java.io, java.util)는 자바를 사용할 수 있는 근본적인 ‘코어’ 패키지로서, 자바 언어 자체를 사용하려는 사람은 누구나 이들 3가지 핵심 패키지를 사용하여야 한다. 자바 API는 최초의 8개 패키지에서 2008년에 이르러서는 6천 개 이상의 메소드를 가지고 600개 이상의 클래스로 구성된, 166개의 패키지로 성장했다.<sup>11)</sup>

## 2. 자바 문법

메소드는 한번 선언되면 `java.package.Class.method()`; 형식의 명령어로 프로그램의 어디인가에서 호출된다. 메소드가 프로그램의 어디인가에서 호출될 때, 일반적으로 ‘변수’가 입력 값으로써 메소드에 전달된다. 메소드의 출력 값은 ‘반환(return) 값’이라고 한다. 일례로 두 개의 숫자를 입력 값으로 해서 둘 중 큰 수를 반환(리턴)하는 메소드를 들 수 있다.

메소드는 다시 메소드 헤더와 메소드 바디로 구성된다. 메소드 헤더<sup>12)</sup>는 메소드의 이름, 메소드에서 사용되는 숫자, 명령, 그리고 데이터 형(type) 등을 포함한다. 메소드 바디(구현코드)가 서로 다르게 작성되더라도 같은 기능을 가지는 메소드를 작성하기 위해서는 반드시 같은 방식으로 메소드 헤더

10) 즉, 각 패키지는 클래스로 세분화되고, 이들 클래스는 다시 메소드로 세분화된다. 예를 들어, “`Int a = java.lang.Math.max(2, 3)`”에서 `java.lang`(패키지)은 `Math`(클래스)를 포함하고, `Math`는 `max`(메소드)를 포함하며, `max`는 2개의 입력 값 중에서 더 큰 값을 반환한다. 여기서 3이라는 결과 값은 `java.lang.Math.max`라는 명령어와 정확한 형식의 입력 값으로 작성되어야 호출될 수 있다.

11) Oracle America, Inc. v. Google Inc. 872 F.Supp.2d 974, 978 (N.D. Cal., 2012).

12) 메소드 헤더는 ‘선언(declaration) 코드’라고도 한다.

를 작성해야 한다. 메소드 바디<sup>13)</sup>는 메소드를 구현하는 코드 블록(block)이다. 입력 값을 취하고, 그것들을 계산하고, 계산 값을 반환하는 실제 업무를 하는 것은 메소드 바디이다.<sup>14)</sup>

메소드 다음으로 높은 수준의 구문(syntax)은 클래스이다. 클래스는 자바 언어의 기본적인 구성 요소로서, 자바 언어로 작성된 프로그램은 하나 이상의 클래스로 이루어진다. 클래스에는 하나 이상의 메소드가 있을 수 있고, 패키지에는 하나 이상의 클래스가 있을 수 있다. 자바 프로그램의 모든 코드는 클래스에 위치해야 한다. 클래스 헤더(선언코드)는 클래스 이름과 클래스를 정의하는 다른 정보들을 포함하고 있고, 클래스 바디는 메소드와 변수를 포함한다.

그리고 편의를 위해 폴더에 파일을 보관하는 것과 마찬가지로, 클래스는 다시 ‘패키지’로 그룹화된다. 이상의 메소드, 클래스, 패키지를 보여 주는 간단한 소스 코드 예는 아래와 같다.<sup>15)</sup>

|                                                   |                      |
|---------------------------------------------------|----------------------|
| <b>package</b> java.lang;                         | // 패키지 java.lang를 선언 |
| <b>public class</b> Math {                        | // 클래스 Math를 선언      |
| <b>public static int</b> max (int x, int y)     { | // 메소드 max를 선언       |
| if (x > y) return x;                              | // 구현, x를 반환하거나      |
| else return y;                                    | // 구현, y를 반환         |
| }                                                 | // 메소드를 닫음           |
| }                                                 | // 클래스를 닫음           |

위 메소드를 다른 프로그램(또는 클래스)에서 호출하기 위해서는 해당 프로그램의 소스 코드에 다음과 같이 입력하면 된다.

```
int a = java.lang.Math.max (2, 3);
```

13) 메소드 바디는 ‘구현(implementation) 코드’라고도 한다.

14) 구글은 안드로이드 API의 메소드 바디를 직접 작성하였고, 이는 구글이 복제한 37개 자바 API 패키지 코드의 97%를 차지한다.

15) Oracle America, Inc. v. Google Inc. 872 F.Supp.2d 974, 979-981 (N.D. Cal., 2012).

이러한 코드를 발견하면 컴퓨터는 java.lang 패키지에 있는 Math 클래스의 max 메소드를 찾아서, 입력 값으로 2와 3을 입력하고, 반환되는 3을 변수 a의 출력 값으로 입력한다.

‘public static’으로 시작하는 선언코드는 전적으로 문법에 따라 작성된 것이다. 특정 기능을 수행하기 위해, 자바 언어는 메소드 선언에서 특정한 형태를 가질 것을 요구하기 때문이다. 그것을 표현하는 방법에서 선택의 여지가 없다. 이처럼 동일한 기능을 달성하기 위해서는, 선언코드는 반드시 동일해야 하지만, 메소드의 이름과 그 메소드들이 그룹화되는 방식은 반드시 동일할 필요가 없다.

### III. 오라클 대 구글(Oracle v. Google) 사건 전개과정과 쟁점

#### 1. 사건의 전개과정<sup>16)</sup>

구글은 2005년 8월 스마트폰 플랫폼을 개발하기 위한 계획의 일환으로 안드로이드사를 인수하고, 자신들의 플랫폼(안드로이드 플랫폼)에 자바 언어를 사용하기로 결정했다. 이후 구글은 2005년 후반 모바일 장치를 위하여 전체 자바 플랫폼을 사용하고, 자바를 안드로이드 플랫폼의 오픈소스 구성품으로 하는 공동 개발 파트너십의 가능성에 대하여 Sun과 수개월에 걸쳐 협상하였지만 합의에 실패하였다.<sup>17)</sup>

16) Oracle America, Inc. v. Google Inc. 872 F.Supp.2d 974, 977-979 (N.D. Cal., 2012)와 Oracle America, Inc. v. Google Inc. 750 F.3d 1339, 1347-1348 (Fed. Cir., 2014) 부분을 요약.

17) 당사자들 사이의 의견의 충돌 부분은 안드로이드 프로그램이 자바 가상 머신 및 다른 자바 프로그램과 호환되게 하는 것을 구글이 거절한 것에서 비롯되었다. Sun(오라클)은 그러한 입장이 자바의 “Write Once, Run Anywhere” 철학에 반하기 때문에, 구글에 대하여 자바 API 패키지를 사용할 수 있는 라이선스를 부여하지 않았다. 이후 실제로 구글은 안드로이드 플랫폼에 37개 자바 API는 그대로 복제하면서 다른 자바 API는 그대로 복제하지 않아 안드로이드용 앱(프로그램)이 일반적으로 자바와 호환되지 않게 되었다.

구글은 Sun과의 합의가 결렬되자 독자적으로 가상머신을 설계하고(소위 ‘달빅’ 가상머신), 자바 언어를 이용해 자바 API의 기능을 독자적으로 구현하기로 결정했다. 특히 구글은 이 사건에서 문제 된 37개의 자바 API 패키지의 모든 기능을 구현(implement)할 수 있도록 구현코드를 독자적으로 작성하였다.<sup>18)</sup> 다만, 구글은 37개 자바 API 패키지에서 7,000줄에 이르는 선언코드는 그대로 복제하였다. 이후 구글은 2년 반에 걸친 독자적인 안드로이드 API 구현코드 작성을 통해 2007년 안드로이드 플랫폼을 출시하였다. 안드로이드 플랫폼의 API는 이 사건에서 다툼의 대상이 된 37개 패키지를 포함하여 168개의 패키지를 가지고 있다. 자바와 안드로이드의 37개 패키지 코드를 상호 비교하면, 단지 3%의 코드만이 동일하였는데 그 동일한 코드들은 메소드와 클래스의 이름, 매개 변수 및 기능을 특정하는 선언코드이었다. 그 후 오라클은 2010년에 Sun을 인수·합병하였고, 이에 따라 자바에 관한 권리를 취득하게 되었다. Sun이 오라클로 변경된 이후에 오라클은 구글에 대해 안드로이드 플랫폼이 오라클의 자바 API 저작권을 침해했다고 주장하며 지방법원에 소송을 제기하였다.

## 2. 사건의 법적 쟁점

구글이 7,000줄에 이르는 자바 API 선언코드와 37개 자바 API 패키지 전체 구조·순서·조직(SSO)을 복제하였다는 사실과, 구글이 새로 작성한 6,000개 이상의 메소드 구현코드(메소드 바디)에는 저작권법상의 문제가 없다는 데에 당사자 사이에 다툼이 없었다. 따라서 이 사건의 쟁점은 구글이 자바 API의 166개의 패키지 중 37개 패키지의 7,000줄에 이르는 선언코드와 위 37개 패키지의 SSO를 복제한 것이 오라클의 저작권을 침해하였는지 여부이다. 즉, 문제가 된 것들은 ① 자바 API 저작권의 문언적 요소(7,000줄의 선언코드)와 ② 자바 API 저작권의 비문언적 요소(37개 자바 API 패키지의 SSO) 2

18) 오라클과 구글 양측은 모두 37개 자바 API 패키지의 97%를 차지하는 안드로이드의 구현코드(즉, 메소드 바디)들이 자바의 구현코드와는 다르다는 것을 인정했다.

가지이다.

먼저 ① 자바 API 저작권의 문언적 요소에 관하여 살펴보면, 구글은 자바 API의 선언코드를 그대로 복제하였다. 따라서 오라클은 선언코드에 대해서는 문언적 복제 문제를 제기하고 있는 것이다. 다음으로 ② 자바 API 저작권의 비문언적 요소에 관하여 살펴보면, 오라클은 구글이 자바 API의 SSO를 유사하게 복제하였다고 주장하며 SSO의 전체에 대한 비문언적 복제를 주장했다.

## IV. 자바 API의 저작물성

### 1. 아이디어-표현 이분법

저작물이란 인간의 사상 또는 감정을 ‘표현’한 창작물을 말한다(저작권법 제2조 제1호). 따라서 아이디어는 저작물로 보호되지 않고 표현만이 저작물로 보호된다. 이를 아이디어-표현 이분법이라고 한다. 우리 저작권법은 미국과 달리 아이디어-표현 이분법에 대하여 명문규정을 두고 있지는 않지만, 대법원 1993.6.8. 선고 93다3073, 3080 판결은 저작권법이 보호하고 있는 것은 사상, 감정을 말, 문자, 음, 색 등에 의하여 구체적으로 외부에 표현한 창작적인 표현형식이지, 표현되어 있는 내용, 즉 아이디어나 이론 등의 사상 및 감정 그 자체가 아니라고 판시하여 미국의 입법 및 판례와 같은 입장을 취하고 있다.

미국은 연방대법원의 Baker v. Selden 사건 판결<sup>19)</sup>에서 비롯된 아이디어-표현 이분법이 판례로 확립되었고,<sup>20)</sup> 1976년에 개정된 미국 저작권법 제102

19) Baker v. Selden, 101 U.S. 99(1879).

20) 이 사건의 쟁점은 Selden의 책에 설명된 무기방식을 이용하여 무기장부를 제작, 판매한 것이 저작권의 침해로 되는가, 즉 Selden의 책에 기술된 무기방식 자체가 어떠한 배타적인 권리를 갖게 되는가의 문제였는데, 이에 관해 미국연방대법원은 “저작권이 보호하는 책에서의 표현(이 사건의 경우 무기방식의 설명 자체)과 책이 설명하고 있는 기술 자체는 명백히 구분되어야 한다. 무기방식과 같은 아이디어는 저작권의 보호영역이 아니라

조 (b)항은 이를 명문화하고 있다. 즉, 저작자의 원저작물에 대한 저작권 보호는 그 형태 여하를 불문하고 당해 저작물에 기술, 설명, 예시 또는 화체된 관념, 절차, 과정, 체제, 조작방법, 개념, 원칙 또는 발명에 대하여는 적용되지 아니한다고 규정하고 있다.<sup>21)</sup>

그리고 아이디어-표현 이분법에서, 저작권법의 보호가 컴퓨터 프로그램의 문자적 요소뿐만 아니라 비문자적 요소에도 미칠 수 있음은 확립된 견해이다.<sup>22)</sup> 여기서 컴퓨터 프로그램의 문자 요소는 소스 코드와 오브젝트 코드이고,<sup>23)</sup> 비문자적 요소는 프로그램의 사용자 인터페이스뿐만 아니라 프로그램의 SSO를 포함한다.<sup>24)</sup> 프로그램의 비문자적 요소가 보호되는지 여부는, 각 사건의 특정한 사실관계에 따라, 문제된 구성 요소가 ‘표현’이냐, 아니면 ‘아이디어’이냐에 따라 결정된다.

---

특허권의 보호영역으로 본다”는 이유로 저작권 침해를 인정하지 않았다.

21) 17 USC § 102 (b) In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.

22) Whelan Assocs., Inc. v. Jaslow Dental Laboratory, Inc. (3rd Cir. 1986). 이 사건에서 제3연방항소법원은 아이디어-표현 이분법을 적용하여 컴퓨터 프로그램의 저작권 보호를 문자적 요소를 넘어 비문자적 요소인 SSO까지 확장하였다. Kravetz, Paul I., “Idea/Expression Dichotomy and Method of Operation: Determining Copyright Protection for Computer Programs”, *DePaul Bus. LJ*, Vol.8(1995), p.87. 한편, 제2연방항소법원은 알타이(Altai) 사건에서 저작권 침해사건의 실질적 유사성(substantial similarity) 판단기준으로 ‘추상화-여과-비교(abstraction-filtration-comparison)’의 3단계 기준을 정립하여 많은 법원의 저작권 침해 판단에 영향을 미쳤다. *Computer Associates International v. Altai, Inc.*, 982 F.2d at 702 (2d Cir. 1992). 이 3단계 기준은 컴퓨터 프로그램의 비문자적 요소가 저작권 보호를 받는 표현인지 판단하는 데에도 활용된다. Kravetz, op.cit., p.93.

23) 미국 법원은 지속적으로 소스 코드와 오브젝트 코드가 저작권법을 통해 보호받는다고 판단하여 왔다. 가령 애플(Apple) 대 프랭클린컴퓨터(Franklin Computer) 사건에서 제3연방항소법원에 따르면, “컴퓨터 프로그램은 오브젝트 코드이든 소스 코드이든 ‘어문저작물(literary work)’이며, 오브젝트 코드나 소스 코드로부터의 허락받지 않은 복제로부터 보호된다(Thus a computer program, whether in object code or source code, is a “literary work” and is protected from unauthorized copying, whether from its object or source code version).” *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240 (3d Cir. 1983).

24) *Johnson Controls, Inc. v. Phoenix Control Sys., Inc.*, 886 F.2d 1173, 1175 (9th Cir. 1989).

## 2. ‘창조적 개성’의 인정여부

### (1) ‘선택의 폭 이론’ 및 ‘창조적 개성 심사 엄격화의 원칙’

저작권법으로 보호되는 저작물이기 위해서는 표현에 창작성이 있어야 한다(저작권법 제2조 제1호). 우리 저작권법이 법문에서 저작물을 ‘창작물’이라고 표현한 것은 이를 분명히 한 것이다. 그리고 이 창작성의 의미에 대하여 대법원 2005.1.27. 선고 2002도965 판결은 “창작성이란 완전한 의미의 독창성을 말하는 것이 아니며 단지 어떠한 작품이 남의 것을 단순히 모방한 것이 아니고 작자 자신의 독자적인 사상 또는 감정의 표현을 담고 있음을 의미하므로 누가 하더라도 같거나 비슷할 수밖에 없는 표현, 즉 저작물 작성자의 창조적 개성이 드러나지 않는 표현을 담고 있는 것은 창작성이 있는 저작물로 볼 수 없다(소위 ‘선택의 폭 이론’)”고 판시하였다.<sup>25)</sup>

그리고 대법원은 위 2002도965 판결을 통해 “기능적 저작물은 그 표현하고자 하는 기능 또는 실용적인 사상이 속하는 분야에서의 일반적인 표현방법, 규격 또는 그 용도나 기능 자체, 저작물 이용자의 이해의 편의성 등에 의하여 그 표현이 제한되는 경우가 많으므로 작성자의 창조적 개성이 드러나지 않을 가능성이 크며, (중략) 기술 구성의 차이에 따라 달라진 표현에 대하여 동일한 기능을 달리 표현하였다는 사정만으로는 그 창작성을 인정할 수는 없고 창조적 개성이 드러나는지 여부를 별도로 판단하여야 한다”고 하여 ‘기능적 저작물의 창조적 개성 심사 엄격화의 원칙’을 천명하였다.<sup>26)</sup>

25) ‘선택의 폭 이론’은 일본의 中山信弘 교수가 저작물의 창작성을 새롭게 개념정의한 것이다. 이에 따르면 사상과 감정을 표현하는 데 선택의 폭이 충분히 있으면 저작자의 창작성이 인정된다. 中山信弘, “創作性についての基本的考え方 (シンポジウム 創作性)”, 『著作権研究』, Vol.28(2001), pp.2-11; 위 대법원 2005.1.27. 선고 2002도965판결은 “저작권법은 기술을 보호하는 것이 아니라 표현을 보호하는 것이므로 기술사상의 변경이 없다 하더라도 표현에 있어서 선택 가능성이 있고, 그 표현에 저작자의 개성이 나타나 있다면 저작권법상 무의미한 개변이라고 할 수 없다”고 하여 선택의 폭 이론을 수용하고 있다.

26) 이해완, 앞의 책, 31쪽, 위 법리를 적용하여 기능적 저작물의 창작성을 부정한 판결로는 대법원 2011.5.13. 선고 2009도6073 판결, 서울서부지방법원 2013.8.16. 선고 2012가합5803 판결 등이 있다.

## (2) 자바 API의 선언코드

### 1) 선언코드의 ‘선택의 폭’

이 사건에서 예시로 든 “`int a = java.lang.Math.max(2, 3);`”라는 코드는 2, 3 두 수 중 더 큰 값을 변수 `a`에 입력하는 코드이므로, 클래스 이름으로 `Mathematics` 또는 그 줄임말인 `Math`를, 메소드 이름으로 `maximum` 또는 그 줄임말인 `max`를 사용하는 것이 매우 자연스럽고 편리하다. 따라서 `java.lang.Math.max( );` 라는 명령어 구조의 API를 작성하면서 사용할 수 있는 표현의 수는 매우 제한적이고,<sup>27)</sup> 이는 ‘누가 하더라도 같거나 비슷할 수밖에 없는 표현’이라고 볼 여지가 많다. 이 사건에서 지방법원은, 구글이 37개 자바 API의 메소드, 클래스 그리고 패키지 이름을 복제한 것은 자바 API 저작권을 침해한 것이라는 오라클의 주장을 자바 API가 저작물성이 없다는 이유로 기각하였다. 지방법원은 합체의 원칙, 필수장면의 원칙 등을 저작물성 판단에 고려함으로써 자바 API와 같은 컴퓨터프로그램의 저작물성 인정 범위 자체를 줄이는 논리구조를 취하였다.<sup>28)</sup>

반면 항소법원의 입장은 “저작물 작성 시 아이디어를 표현하는 방법에 있어 저작자에게 선택의 여지가 있고, 그러한 선택이 남의 것을 모방한 것이 아니라면 그것이 아무리 기능적인 저작물이라고 하더라도 저작물성이 있다”

27) `max` 대신 `maximum`, `larger` 정도가 사용될 수 있을 것이다. 즉, 선택의 폭이 매우 좁다.

28) ‘합체의 원칙(merger doctrine)’이란 어떤 아이디어 자체가 표현과 구별되지 않는 경우, 즉 어떤 아이디어를 표현하는 방법이 극히 적어서 아이디어와 표현이 합체된(merged) 경우 그러한 표현은 저작물로 인정하지 않는다는 원칙이다. “공기처럼 무료인(as free as air)” 아이디어에 대한 독점을 인정할 수 없다는 경제학적인 고려에서 발전한 원칙이다. Abrahamson, Scott, “Seen One, Seen Them All—Making Sense of the Copyright Merger Doctrine”, *UCLA L. Rev.*, Vol.45(1997), pp.1125-1127. 한편 ‘필수장면의 원칙(scenes a faire doctrine)’은 보편적인 설정이나 테마로부터 필수적으로 흘러나오는 표준 문구가 저작권 보호를 받지 못하게 하는 원칙이다. 이는 문자적으로 “scenes ... that must be done”이라 해석되며, 경찰관련 소설에서 경찰이 범인의 발자국을 쫓는 상황, 마피아 영화에서 입에 시가를 물고 거친 말을 내뱉는 이탈리아 출신의 폭력배 등이 전형적인 필수장면의 예이다. Ocasio, Sandro, “Pruning Paracopyright Protections: Why Courts Should Apply the Merger and Scenes a Faire Doctrines at the Copyrightability Stage of the Copyright Infringement Analysis”, *Seton Hall Cir. Rev.*, Vol.3(2006), pp.304-309. 이를 국내에서는 ‘표준적삽화의 원칙’이라고도 한다.

는 것으로 요약할 수 있다. 그리고 이러한 기준에 따라 저작물성이 있다면, 합체의 원칙, 필수장면의 원칙 등은 저작물성 판단에 장애가 되지 않고, 단지 저작물 침해 여부의 판단에만 고려될 수 있다는 것이다. 저작권 보호를 ‘저작물성 인정과 그에 대한 침해’의 2단계 구조로 보았을 때, 항소법원은 제1단계인 저작물성에 대해서는 기능적 저작물인지 여부를 불문하고 매우 폭넓게 인정하되, 제2단계에서 그 보호 범위를 축소하는 논리구조를 취한 것이다.

## 2) 합체의 원칙 및 필수장면의 원칙의 법적 성격

이처럼 이 사건에서는 합체의 원칙과 필수장면 원칙이 ‘저작물 성립요건’의 문제인지(제1설), 아니면 ‘저작권 제한’의 문제인지(제2설)에 대해 지방법원과 항소법원의 견해가 나뉘었다. 국내에서는 이를 명시적으로 언급하고 있는 문헌은 찾을 수 없으나, 저작권법 해설서의 체계에 비추어 보면 이에 대한 견해가 통일되어 있지 않는 듯하다.<sup>29)</sup> 한편, 법원 역시 이에 대한 명시적인 입장을 표명하고 있지 않다. 합체의 원칙을 수용하고 있다고 평가되는 서울중앙지방법원 2007.1.17. 선고 2005가합65093 판결은 “아이디어를 표현하는 데 실질적으로 한 가지 방법만 있거나, 하나 이상의 방법이 가능하다고 하더라도 기술적인 또는 개념적인 제약 때문에 표현방법에 한계가 있는 경우에는 그러한 표현은 저작권법의 보호대상이 되지 아니하거나 그 제한된 표현을 그대로 모방한 경우에만 실질적으로 유사하다”고 판시하여 이에 대한 명확한 판단을 유보하였다.

## 3) 논의의 실익

그런데 이러한 관점의 차이는 저작권 침해소송의 증명책임 문제에서 차이점을 발생시킨다. 합체의 원칙을 주장하는 것은, 제1설에 따르면 상대방의 저작권 침해 주장에 대한 ‘부인’에 해당하지만, 제2설에 따르면 ‘항변’이 된

29) 합체의 원칙을 ‘저작물성의 성립요건’에서 다루고 있는 견해로는 이해완, 앞의 책, 52면; 이규호, 『저작권법』, 제5판, 진원사, 2015, 13면; 오승중, 『저작권법』, 제3판, 박영사, 2013, 84면이 있고, ‘저작권 보호의 제한과 예외’로 다루고 있는 견해로는 임원선, 『실무자를 위한 저작권법』, 제4판, 한국저작권위원회, 2015, 226면이 있다.

다. 따라서 제1설에 따르면 저작권 침해를 주장하는 원고가 자신의 저작물이 합체의 원칙의 적용을 받지 않는다는 사실을 주장·입증해야 하고, 제2설에 따르면 피고가 원고의 저작물이 합체의 원칙에 따라 보호받지 못한다는 사실을 주장·입증해야 한다.

또한 이러한 관점의 차이는 합체의 원칙 등이 등록관청의 저작물 등록심사권<sup>30)</sup>의 범위에 속하는지 여부에서도 차이점을 발생시킨다. 등록관청이 저작권 등록신청을 받아 이를 심사하는 경우, 제1설에 따르면 등록관청은 ‘저작물’에 해당되는지 여부를 심사하기 위해 합체의 원칙 등을 고려하여야 하지만, 제2설에 따르면 합체의 원칙 등을 고려하여 심사할 필요는 없다.

#### 4) 엇갈린 판단

이 사건에서 지방법원은 제1설에 따라 자바 API 선언코드에 합체의 원칙, 필수장면의 원칙을 저작물성 판단에 적용하였다. 즉, 자바 문법에 따르면 주어진 기능을 수행하기 위해서는 자바 API 선언코드를 반드시 하나의 방법으로 작성하여야 하는데, 아이디어를 표현하는 방법이 하나뿐일 경우 합체의 원칙에 따라 저작물성이 없으므로, 자바 API 선언코드는 저작물성이 없다고 판단하였다.<sup>31)</sup>

위와 같은 지방법원의 판단에 대해 항소법원은 지방법원이 합체의 원칙과

30) 대법원 1996.8.23. 선고 94누5632 판결은 “저작권법의 규정내용과 저작권등록제도 자체의 성질 및 취지에 비추어 보면, 현행 저작권법이나 같은 법 시행령이 등록관청의 심사 권한이나 심사절차에 관하여 특별한 규정을 두고 있지 않다고 하더라도 등록관청으로서는 당연히 신청된 물품이 우선 저작권법상 등록대상인 ‘저작물’에 해당될 수 있는지 여부 등의 형식적 요건에 관하여 심사할 권한이 있다고 보아야한다”고 판시하고 있다.

31) 지방법원에서 구글은 자신이 오라클의 37개 자바 API의 메소드, 클래스, 패키지 이름을 복제했다는 사실을 다투는 대신, 이들의 저작물성이 없다는 것과, 설령 저작물성이 있더라도 자신의 복제행위가 저작권법상 공정이용(fair use)에 해당한다고 주장하였다. Daria Vasilescu-Palermo, *supra* note 5, 166면 참조. 다만 구글이 공정이용 주장에 힘을 기울였음에도 불구하고 지방법원의 배심원들은 이 쟁점에 대한 결론을 내리지 못했고, 심지어 이에 관한 사실관계 판단도 제대로 이루어지지 못하였다. Tyler J. Demasky, *supra* note 5. 항소법원은 지방법원이 구글의 공정이용 여부에 대한 배심원 판단을 하지 못하는데 오류가 있음을 지적하였다. Oracle America, Inc. v. Google Inc. 750 F.3d 1339 (Fed. Cir., 2014). 1377면.

필수장면의 원칙을 자바 API 선언코드의 저작물성 판단에 적용한 것은 잘못이라고 판단했다. 우선, 합체의 원칙에 관하여는 Sun(오라클)이 자바 API 선언코드를 작성하는데 유일한 또는 제한된 방법만이 있는 것이 아니었기 때문에, 합체의 원칙은 자바 API 선언코드의 저작물성 인정에 장애가 되지 않는다고 판단하였다. `java.lang.Math.max` 예로 돌아가 보면, 선(오라클)은 `java.lang.Math.max` 메소드를 `java.lang.Math.maximum` 또는 `java.lang.Arith.larger`로 이름 지을 수도 있었으므로, 합체의 원칙이 적용되지 않는다고 판단했다. 나아가 항소법원은 지방법원이 합체의 원칙을 적용함에 있어서 ‘구글이’ 선언코드를 복제할 때 가능한 선택권(옵션)에 초점을 맞추는 오류를 범했음을 지적했다. 저작물성에 대한 평가는, 침해자의 침해 시가 아니라, 원저자의 창작 시에 이루어져야 한다는 원칙이 확립되어 있기 때문이다.<sup>32)</sup> 따라서 자바 API 선언코드가 저작물성이 있느냐에 대한 질문의 초점은 ‘Sun’이 자바 API 패키지 작성 시에 선택 가능했던 선택지가 하나밖에 없거나 제한적이었느냐 여부라는 것이다. 즉, Sun(오라클)은 “`java.lang.Math.max`” 메소드를 작성했고, 자바 프로그래머는 이 메소드를 사용하기 위해서 그 이름으로 호출해야 하지만, 그러한 사실이 구글이 “`java.lang.Math.max`”와 다른 이름으로 선언코드를 작성하는 것을 방해하지 않았다는 것이다. 결국 오라클이 자바 API의 명령어 구조에서 선택한 표현들은 아이디어와 합체되지 않는다는 것이다.<sup>33)</sup> 다음으로 필수장면의 원칙에 관하여는, Sun이 자바 API 선언코드를 작성할 당시에 외부적인 요인들<sup>34)</sup>로 인하여 선언코드를

32) Apple Computer, Inc. v. Formula Int'l, Inc., 725 F.2d 521, 524 (9th Cir. 1984).

33) 다만, 항소법원도 자바 API의 핵심 패키지 3개에 대해서는 합체의 원칙에 따라 구글의 침해 행위를 인정하기 어려울 수 있다는 점을 인정했다. 이들 3개 핵심 패키지는 선(오라클)이 이들 패키지를 작성할 당시에 자바 언어로 이들 패키지에 있는 메소드와 클래스를 표현하는 데 제한된 방법들만 있었을 가능성이 있기 때문이다. 그리고 그러한 경우에는 이들 3가지 패키지의 표현은 아이디어와 합체되었고, 구글이 허락없이 복제하더라도 저작권 침해행위가 되지 않았을 것이다.

34) i) 프로그램이 수행되는 컴퓨터의 기계적인 사양(mechanical specifications of the computer on which a particular program), ii) 호환성(compatibility requirements of other programs with which a program is designed to operate in conjunction), iii) 컴퓨터 제작업자들의 디자인 표준(computer manufacturers' design standards), iv) 산업계가

작성하는 데 제한된 방법만이 있는 것이 아니었기 때문에, 필수장면의 원칙 역시 자바 API 선언코드의 저작물성 인정에 장애가 되지 않는다고 판단하였다. 구글은 ① 프로그래머들이 자바 API에 익숙하여 그 사용에 편리함을 느끼고 있고, ② 자바API는 대체불가능할 정도로 일반적이므로 표현이 아닌 아이디어로 다루어져야 한다고 주장했지만—합체의 원칙과 마찬가지로—필수장면의 원칙의 초점은 복제자가 아니라 창조자에 있으므로, ‘Sun’이 자바 API를 작성할 당시의 외부적인 요인에 비추어 선택의 폭이 있었는지에 초점을 맞추어야 하고, ‘구글’이 자바 API 선언코드를 복제할 당시로 맞춰서는 안 된다고 판시하면서 구글의 주장을 배척하였다.

## 5) 소 결

합체의 원칙, 필수장면의 원칙 등을 저작물의 성립요건 문제로 다루고, 저작권 보호를 주장하는 측에서 그 창작성을 증명하여야 하는 것이 타당하다. 그리고 저작권 등록관청 역시 등록 신청된 물품이 합체의 원칙 등을 고려하여 저작물성을 판단하여야 한다(제1설의 입장).

그 이론적인 근거는 다음과 같다. 첫째, 인간의 사상 또는 감정의 표현에 ‘창작성’이 있다고 인정될 때에만 저작권의 보호를 긍정할 수 있는데, 만일 특정한 사상 내지 아이디어를 표현하는 방법 자체가 하나밖에 없거나 효율성 등의 측면에서 극히 제한되어 있다면 그러한 표현에는 저작자 나름대로의 ‘창조적 개성’이 발휘된 것으로 볼 수 없을 것이라는 점에서 창작성 자체가 부정되어야 할 것이다. 둘째, 정책적인 관점에서도 아이디어를 표현하는 방법이 제한되어 있는 경우, 그러한 표현에 저자 사후 70년간이라는 장기간 독점적인 권리를 주장하는 자가 있다면(저작권법 제39조) 그 권리를 주장하는 자가 그 표현에 저작물로서 보호받을 만한 창조적인 개성이 있다는 것을 증명하도록 하는 것이 타당하다. 이러한 표현에 대한 보호는 실질적으로 아이

---

요구하는 제반 요구사항들(demands of the industry being serviced), v) 컴퓨터 산업에서 일반적으로 승인된 프로그램 제작 관행(widely accepted programming practices within the computer industry). Computer Associates International v. Altai, Inc., 982 F.2d at 702, (2d Cir. 1992).

디어에 대한 보호가 되므로, 특허의 경우 출원자가 그 신규성과 진보성을 증명하여 등록을 통해 권리를 부여받는 것과 유사하게, 저작물성을 주장하는 자가 그 표현에 합체의 원칙, 필수장면의 원칙이 적용되지 않는다는 것을 주장·증명할 필요가 있기 때문이다. 결국 자바 API의 선언코드는 합체의 원칙, 필수장면의 원칙에 따라 저작물성 자체가 부정되어야 한다.<sup>35)</sup>

### (3) 자바 API의 SSO

#### 1) 사실상의 표준

자바 언어의 문법은 메소드들을 특정 클래스로 그룹화할 것을 요구하지 않으므로 구글은 max 메소드를 Math 클래스 이외의 다른 클래스의 소속으로 배치할 수도 있었다. 그럼에도, 구글이 안드로이드 플랫폼을 만들면서 자바와 동일한 ‘패키지 - 클래스 - 메소드’의 구조를 취한 것은 자바 API의 SSO에 관한 오라클의 저작권을 침해한 것은 아닌지가 문제된다.

실제 자바 문법은 특정한 그루핑(조직화) 패턴을 요구하지 않는다. 구글이 자바 API와 동일한 기능을 자유롭게 복제할 수 있다고 하더라도, 구글이 이들 메소드들을 동일하게 그루핑하여야 한다는 자바 문법은 없다. 예를 들어 max 메소드가 반드시 java.lang.Math 클래스에 속해 있어야 할 자바 문법상 이유는 없고, java.lang.Number 클래스에 속하게 할 수도 있다.

하지만 구글이 max 메소드를 java.lang.Number 클래스가 아닌 java.lang.Math 클래스에 속하도록 안드로이드 API를 구성한 이유는 그것이 자바 언어를 사용하는 개발자들에게 익숙할 뿐만 아니라, 기존의 자바 API로 작성된 프로그램 코드를 새로 수정할 필요가 없다는 현실적인 이유 때문이었다. 즉, 자바 언어를 사용하는 개발자들에게 오라클의 자바 API(특히 그중에서도 37개 자바 API)는 ‘사실상의 표준’에 해당하였기 때문에, 구글이 자바 개

35) 한편, 저작권법은 이름, 제목 또는 짧은 문구나 표현을 보호하지 않는다[Sega Enters., Ltd. v. Accolade, Inc., 977 F.2d 1510, 1524 n. 7 (9th Cir.1992), 국내 판결로는 만화제목 ‘도복이’는 사상 또는 감정의 표현이라고 보기 어려워 저작물로서 보호받을 수 없다는 대법원 1977.7.12. 선고 77다90 판결 참조], 자바 API의 선언코드는 이와 같은 법리에 의해서도 저작물성이 부정되어야 한다.

발자들을 안드로이드 플랫폼에 유입시키기 위해서 자바 API의 SSO를 복제한 것이다.

## 2) 사실상의 표준 이론 및 로터스 사건 판결

사실상의 표준(de facto standard)이란 특정한 표현이 이미 사용자나 업계에 널리 알려지거나 관행으로 굳어져 있는 것을 말한다. 컴퓨터프로그램의 사용자 인터페이스(UI)가 대표적인 사례이다. ‘파일(File) - 저장(Save)’, ‘편집(Edit) - 복사(Copy)’ 등의 명령어 계층을 다르게 구성하거나 다른 용어를 사용할 수도 있지만 그렇게 하면 기존의 명령어 계층과 용어에 익숙한 사용자로부터 외면당하기 쉽다. 이런 경우에 사용자 인터페이스를 저작권법으로 보호한다면, 후발사업자는 사실상의 표준을 활용할 수가 없으므로 경쟁에 있어 매우 불리한 처지에 처하게 된다. 사실상의 표준 이론은 저작권 보호로 인하여 경쟁이 제한되는 결과가 초래되지 않도록 이러한 경우에는 이에 대한 보호가 제한된다는 것이다.<sup>36)</sup> 사실상의 표준 이론은 유명한 로터스 사건에서 적용되었다.<sup>37)</sup>

이와 같이 사실상의 표준 이론에 따라 저작권법에 의한 보호가 거부될 수도 있음은 일반적으로 인정되고 있으나,<sup>38)</sup> 그것이 저작물 ‘성립요건’인지, 아니면 ‘저작권 제한’의 요소인지에 대해서는 합체의 원칙 등과 마찬가지로 별다른 논의가 없었다. 그리고 이 사건에서 지방법원과 항소법원이 이에 대해

36) 임원선, 앞의 책, 211면.

37) Lotus Development Corp. v. Borland International, Inc., 49 F.3d 807 (1st Cir. 1995).

로터스 사건의 쟁점은 ‘컴퓨터프로그램의 메뉴 명령어 계층 구조(Lotus 1-2-3 메뉴구조)가 저작권으로 보호받을 수 있는지’ 여부였다. 로터스 사건 항소법원은 사실상의 표준화에 의한 호환성의 요구를 적극적으로 참작하여 저작물성을 부정하였다(“로터스 메뉴가 조작방법에 해당한다는 것은 프로그램의 호환성을 고려할 때 더욱 명백하다. 로터스 측의 주장에 따르면, 사용자는 여러 가지 프로그램을 사용할 때 각 프로그램마다 동일한 기능에 대하여 다르게 정하여진 사용방법을 배워 익혀야 한다. … 이것은 매우 불합리하다.”).

38) 오승중, 앞의 책, 89면, 각주 102번(“특히 저작권법 사건에 대하여 가장 큰 권위를 가지고 있는 미국 제2, 9 항소법원이 ‘사실상의 표준화’에 따른 합체를 인정하여 왔고, 더군다나 위와 같은 연방대법원의 Lotus 판결이 내려져 현재까지 유지되고 있으므로 법원들도 이에 따를 것으로 보인다.”)

서도 결론을 달리하였다.

### 3) 엇갈린 판단

지방법원은 로터스 사건 판결, Sega판결<sup>39)</sup> 그리고 Sony 판결<sup>40)</sup>이 이 사건과 궤를 같이한다고 보았다.<sup>41)</sup> 따라서 자바 API의 SSO는 창의적이고 독창적이기는 하지만, 이는 코드의 상호운용성을 보장해 주는 ‘시스템’ 또는 ‘조작 방법’이므로 미국 저작권법 제102조 (b)항에 따라 저작물로 볼 수 없다고 판단하였다. 반면 항소법원은 상호운용성은 공정이용과 관련이 있을 뿐, 저작물성과는 관련이 없다고 판단하였다. 그러면서 ① 지방법원이 인용한 Sega 판결과 Sony 판결은 모두 공정이용에 관한 것이지 저작물성에 관한 것이 아니라고 보았다.<sup>42)43)</sup> 또한 저작물성의 초점은 저작권 보호를 주장하는 당사

39) Sega Enterprises Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992). Sega 사건에서는 피고 Accolade가 원고 Sega의 게임 콘솔인 Genesis에서 작동하는 비디오 게임을 만들기 위해 Sega의 프로그램을 역분석한 것이 문제가 되었다. 원고는 피고가 역분석을 위해 원고 저작물을 복제한 것이 저작권 침해라고 주장했으나, 제9 연방순회항소법원은 호환성이 있는 프로그램을 만들기 위해 인터페이스 정보를 추출하는 것과 같은 정당한 목적을 위하여 프로그램코드를 역분석하는 것은 저작권 침해가 아니라 공정이용이라고 판단하였다.

40) Sony Computer Entm't, Inc. v. Connectix Corp., 203 F.3d 596 (9th Cir.2000). Sony 사는 소니 플레이스테이션 콘솔(소형 컴퓨터)과 플레이스테이션 게임을 개발하여 판매하는 회사인데, 플레이스테이션 콘솔 ROM 칩 상에는 펌웨어라고 불리는 소니 BIOS (Basic Input-Output System)가 기록되어 있다. 한편, Connectix사는 플레이스테이션 콘솔이 비싸 소비자들이 부담을 느낀다는 점에 착안하여 가상 게임 스테이션(Virtual Game Station: VGS)라는 소프트웨어를 만들기로 하고 소니 BIOS를 역분석하였다. 그런데 역분석 과정에서 Connectix사의 개발자들은 자신들의 컴퓨터 RAM에 소니 BIOS를 복제하였는데, 이러한 행위에 대해 Sony사가 저작권 침해소송을 제기하였다. 미국 연방항소법원은 Connectix사가 자신이 구입한 플레이스테이션 콘솔에서 추출한 소니 BIOS를 역분석한 것은 ‘공정이용’에 해당하는 것으로서 저작권침해를 구성하지 않는다고 판시하였다. 그 이유로서 Connectix사가 새롭게 개발한 VGS상에 저작권 보호를 받는 소니 BIOS의 코드를 전혀 채용하지 않았고, 다만 게임 소프트웨어와 VSG 사이의 호환을 위하여 소니 BIOS에서 추출한 인터페이스 부분을 사용하였지만, 이는 저작권법상 보호받지 못하는 기능에 해당한다고 보았다.

41) 지방법원은, Sega 판결과 Sony 판결을 “상호운용성을 달성하기 위해 복제할 필요가 있는 인터페이스는 미국 저작권법 제102조 (b)항에 따라 ‘저작물성’이 없다”는 것을 판시한 판결들이라고 해석하였다.

42) 실제, Sega 판결의 유일한 쟁점은 Accolade의 중간 복제가 공정이용인지 여부였다. 법

자의 프로그래밍 표현에 있어서 선택의 폭에 있지, 원고의 프로그램과 상호 운용성을 달성하려고 하는 피고의 선택에 있지 않다고 지적하였다. 따라서 원고가 일단 저작물성 있는 저작물을 창조하였다면, 상호운용성을 달성하고자 하는 피고의 열망은 저작물성과는 무관한 상업적이고 경쟁적인 목표에 불과하다는 것이다.<sup>44)</sup> 아울러 ② 항소법원은 자바 API 패키지가 소프트웨어 산업에서 유효한 표준이 되었기 때문에 자유롭게 복제할 수 있다는 구글의 주장은 명시적으로 배척하였다.<sup>45)</sup> 즉, 구글은 자신들의 독자적인 API를 개발하고, 프로그래머들에게 이것을 채택해 달라고 ‘로비’할 수도 있었지만, 그렇게 하지 않고 오라클의 선언 코드와 SSO를 복제하기로 결정했다. 그리고 이러한 구글의 의도는 저작물성과는 관련이 없다는 것이다.

#### 4) 소 결

앞서 본 합체의 원칙, 필수장면의 원칙이 ‘아이디어-표현 이분법’의 영역, 즉

---

원은 Sega의 소프트웨어 코드가 보호받을 수 있는지 여부에 대해서 판단하지 않았다.

43) 항소법원은 지방법원이 로터스 판결을 인용한 것도 잘못이라고 보았다. 즉, i) 로터스 사건의 피고는 로터스 1-2-3의 소스 코드를 복제하지 않았지만, 구글은 7,000줄에 이르는 오라클의 선언코드를 그대로 복제했다는 점, ii) 로터스 판결에서 문제 된 명령어들(Copy, Print 등)은 창작성이 없지만, 자바 API의 선언코드와 SSO는 모두 독창적이고 창의성이 있다는 점, iii) 로터스 판결에서 문제된 명령어 계층은 컴퓨터 시스템을 ‘동작시키는 데 필수적’이지만, 자바 API의 SSO는 구글이 안드로이드 플랫폼을 개발하는 데에 반드시 복제할 필요가 없었다는 점 등에 비추어 보면, 이 사건과 로터스 판결은 사실관계가 다르므로, 이 사건에 로터스 판결을 인용하는 것은 잘못된 것이라고 보았다.

44) 그러한 측면에서 보면, 안드로이드 플랫폼이 자바 플랫폼과 어떤 측면에서 상호운용성이 있는지 여부는 자바 API가 저작물성이 있는지 여부와는 무관하게 된다. 구글이 달성하려고 노력한 ‘상호운용성’은, 소프트웨어 개발자들이 이미 자바 API 사용법에 대한 교육을 받고 사용한 경험이 있다는 사실에 대한 활용성이었으며, 구글의 관심사는 자바 개발자들을 이용해 안드로이드 플랫폼의 개발 프로세스를 가속화하는 것이었다. 항소법원은 이러한 경쟁적인 목표가 공정이용의 문제와 관련이 있을 수 있지만, 자바 API의 선언 코드와 SSO의 저작물성과는 관련이 없는 것으로 결론지었다.

45) 한편, 본건 항소법원 판결 이후 이른바 소프트웨어 산업에서 상호운용성을 높이고 반공유지(anticommons) 문제를 해결하기 위해 표준화(standardization)가 중요하지만, 저작권 보호를 강조할수록 표준화가 어려워진다는 견해도 제시되고 있다. 가령 Asay, Clark D., “Software's Copyright Anticommons”, *Emory Law Journal*, Vol.66 No.2(2016), pp. 265-331 참조. 이 논문은 상호연결성이 강화된 세계에서 반공유지 문제의 해결을 위해 공정이용 기준의 재해석을 제안하고 있다.

저작물성 인정 여부의 문제였다면, 어떤 표현이 사실상의 표준인지 여부는 저작물성과는 관련이 없는 것으로 보아야 한다. 사실상의 표준인지 여부는 저작물의 창작 당시가 아니라 나중에 가서 표준화라는 상태가 발생하였을 때 결정되는데다가, 저작물의 창작 당시에는 그 창작자의 입장에서는 표현 선택의 폭이 넓었음으로 저작자의 창조적 개성이 발휘된 것으로 보아야 하기 때문이다. 따라서 어떠한 API가 개발자 커뮤니티에서 사실상의 표준에 해당하고, 그렇기 때문에 API 저작자의 허락 없이 이용하는 것이 가능하다는 주장은 공정이용을 주장하는 피고의 항변 사항에 해당한다고 보는 것이 타당하다.

이러한 관점에서 보면, 로터스 판결 역시 로터스 1-2-3 명령어 체계가 미국 저작권법 제102조 (b)항에 따라 저작물성 자체가 부정된다고 판시하기보다는, 그러한 표준적인 UI(User Interface)를 사용하는 것은 공정이용에 해당한다고 판시하는 것이 보다 적절하였다고 생각한다. 로터스 판결과 이 사건은 저작물성 판단 대상이 사용자 인터페이스(User Interface: UI)에서 응용 프로그래밍 인터페이스(API)로 변경되었을 뿐이므로 동일한 법리가 적용되어야 하기 때문이다.

### 3. 저작권법 제101조의2 적용여부

#### (1) 자바 API와 자바 언어의 구분

##### 1) 프로그램 언어와 프로그램의 구별

미국에서는 자바 API의 저작물성에 대한 논쟁이 미국 저작권법 제102조 (b)항의 해석과 관련된 것인 데 반해, 우리 저작권법에는 미국 저작권법 제102조 (b)항과 같은 규정이 없다. 반면 우리 저작권법은 컴퓨터프로그램 저작물에 대해 미국 저작권법에는 없는 규정을 두고 있는데, 그것이 바로 저작권법 제101조의2이다. 즉, 우리 저작권법은 ‘컴퓨터프로그램 저작물’을 정의함으로써 컴퓨터프로그램이 저작물성이 있는 저작권법상 보호 대상임을 분명히 하면서도(저작권법 제2조 제16호), 프로그램 언어, 규약, 해법은 명시적으로 저작권법의 보호 대상에서 제외하였다(저작권법 제101조의2). 여기서 API는 저작권으로 보호받지 못하는 ‘프로그램 언어’인가, 아니면 보호받는

‘컴퓨터프로그램 저작물’인가가 문제 된다. 우리 저작권법하에는 프로그램 언어란 프로그램을 표현하는 수단으로서 문자·기호 및 그 체계를 말하고 (저작권법 제101조의2 제1호), 구체적으로는 C, BASIC, FORTRAN, COBOL 등이 여기에 해당한다고 설명된다.<sup>46)</sup> 이러한 설명에 따르면, 자바 언어 자체는 현행법상 저작물로 보호받지 못하지만, 자바 언어로 작성된 자바 API는 컴퓨터프로그램 저작물로서 보호되고, 자바 API를 이용해 작성된 프로그램 역시 컴퓨터프로그램 저작물로서 보호된다는 결론에 이른다.

## 2) API의 저작물성 제한필요성

그런데 개발자들이 “자바를 이용하여 안드로이드 앱을 개발한다”고 할 때 자바는 자바 언어 자체뿐만 아니라 당연히 자바 API까지도 포함한다. 안드로이드 앱 개발자들이 자바 API를 이용하지 않고, 처음부터 순수히 자바 언어만을 가지고 프로그래밍을 한다는 것은, 이 사건에서와 같이 구글과 같은 대형 IT 회사가 아니고서는 현실적으로 불가능하다. 이와 같은 프로그램 개발 현실에 비추어 보면, 특정한 프로그램 언어와 그 언어로 작성된 API를 명확히 구분하고 API에 저작물성을 인정하는 것은 지나치게 컴퓨터프로그램 저작권의 보호범위를 넓혀 프로그램 개발 업무에 큰 장애를 초래할 수 있다.<sup>47)</sup>

그리고 구글은 2년 반의 시간과 대규모 인력, 자본을 들여 자바 언어로 직접 안드로이드 API 구현코드들을 작성했지만, 그런 구글조차도 37개의 자바 패키지의 선언코드와 SSO를 그대로 복제할 수밖에 없었다. 물론 구현코드를 독자적으로 작성했듯이 구글은 이러한 상황을 무시하고 선언코드와 SSO 역시 독자적으로 작성할 수도 있었을 것이다. 하지만 그렇게 독자적으로 작성된 안드로이드 API로는 이미 자바 API에 익숙한 개발자들을 안드로이드 플랫폼으로 끌어들이기 어려웠을 것이고, 개발자들도 불필요하게 시간과 노력을 들여 새로운 API를 익힐 수밖에 없었을 것이다. 자바 API 선언코드와 안드로이드 API 선언코드가 다르다면, 개발자들은 이미 자바 API로 만든 프로

46) 이해완, 앞의 책, 187면.

47) 위의 책, 27면.

그램을 안드로이드 플랫폼에서 동작하게 하기 위해, 예를 들어 “int a = java.lang.Math.max(2, 3);”로 작성된 코드를 “int a = java.language.Mathematics.maximum(2, 3);”이라고 불필요하게 변경해야 한다. 구글은 안드로이드 API일부를 자바 핵심 API 선언코드와 동일하게 함으로써 개발자들의 작업상의 불편과 수고를 덜어주고 더 많은 개발자들이 안드로이드 앱(프로그램)을 개발토록 한 것이다. 그리고 API의 저작물성을 인정하여 다른 플랫폼에서 이를 활용할 수 없도록 한다면, 개발자들은 동일한 기능에 대하여 각 플랫폼마다 다르게 정하여진 사용방법을 익혀야 하는데, 이는 기존의 코드의 재활용성을 크게 저해할 뿐만 아니라 개발자들에게 불필요한 시간과 노력을 들이게 할 뿐이므로 매우 불합리하다.

따라서 자바 언어를 사용하려는 개발자 누구나 사용하는 적어도 3개의 핵심 패키지는 자바 언어 그 자체로 보아 저작물성을 부정하여야 한다.

## (2) 자바 API와 프로그램 규약

### 1) API가 프로그램 규약인지 여부

가사 API가 프로그램 언어 자체가 아니라는 엄격한 입장을 취한다고 하더라도, API는 우리 저작권법 제101조의2 제2호가 규정한 ‘프로그램 규약’에 해당하므로 저작권법의 보호를 받지 못하는 것은 아닌지 문제된다. 저작권법으로 보호받지 못하는 프로그램 ‘규약’이라 함은 특정한 프로그램에 있어서 프로그램 언어의 용법에 관한 특별한 약속을 말한다(저작권법 제101조의2 제2호). 즉, 저작권법에서 프로그램 언어는 언어라고 하는 일반적인 체계를 말하고, 프로그램 규약은 어떤 특정한 프로그램을 위한 특별한 약속을 말하는 것이다. 규약은 구체적으로는 인터페이스(interface)와 프로토콜(protocol)로 나타난다(프로토콜은 광의의 인터페이스에 포함된다). 인터페이스는 본래 둘 이상의 기계장치를 연결시켜 작동하게 하는 경우에 상호 간에 정확한 정보가 전달될 필요가 있으므로 이를 위해 정보의 신호, 배열, 타이밍 등을 정확히 연계시켜 주는 것이다. 구체적으로 컴퓨터에 있어서 인터페이스는 하드웨어와 하드웨어(하드웨어 인터페이스), 하드웨어와 소프트웨어, 소프트웨어

와 소프트웨어(API), 그리고 소프트웨어와 사람 사이(UI)를 연결시켜 준다.

우리 저작권법이 인터페이스를 보호대상에서 제외한 것(저작권법 제101조의2)에 대하여는 두 가지 해석론이 존재한다. ① 하나는 저작권법이 인터페이스를 보호대상에서 제외한 것은 아이디어로서의 인터페이스를 배제하는 취지이고, 그것이 구체적인 프로그램으로 ‘표현’되어 있다면, 보호대상에 포함되는 것으로 해석해야 한다는 견해와, ② 그렇게 볼 경우 형식적으로는 표현을 보호하는 것이지만 실질적으로 아이디어를 보호하는 것이 되므로 프로그램 규약에 대하여는 그것이 어떠한 형태로 되어 있더라도 보호하지 않는다는 뜻으로 해석하여야 한다는 견해이다.<sup>48)</sup>

## 2) 입법취지: 프로그램 규약은 아이디어이다

그런데 입법연혁적으로 저작권법 제101조의2는 2009.4.22. 구 컴퓨터프로그램 보호법을 저작권법에 흡수시키면서 구 컴퓨터프로그램 보호법 제3조를 일부 자구만 수정해서 그대로 저작권법에 옮겨온 것이다. 그리고 구 컴퓨터프로그램 보호법 제3조의 취지에 대해 제정 컴퓨터프로그램 보호법 입법 당시 소관 상임위원회인 경제과학위원회의 심사보고서는 다음과 같이 설명하고 있다.<sup>49)</sup>

법안 제2조에서 프로그램의 보호범위를 프로그램의 표현 그 자체로 한정하고, 법안 제4조에서 프로그램을 작성하는 과정 또는 표현하는 수단으로 사용되는 프로그램 언어, 규약 및 해법 등에 대하여는 적용을 배제하고 있음. 이는 프로그램의 보호범위를 아이디어나 노하우(Know-how)에까지 확대시키지 않음으로써 이미 개발된 기술의 적절한 활용을 허용하고 프로그램의 자유로운 개발을 보장하려는 것으로서 우리나라의 기술수준에 비추어 볼 때 적합한 용어정의라고 사료됨.

위 심사보고서에 따르면 우리나라 입법자는 ‘규약’을 아이디어나 노하우(Know-how)로 보는 입법적 결단을 내렸다고 볼 수 있다. 저작권법 제101조의2는 미국에는 없는 규정으로서 우리 입법자들이 컴퓨터프로그램의 특수

48) 이해완, 앞의 책, 187면; 김규성, 『컴퓨터프로그램 보호법』, 세창출판사, 2007, 6면.

49) 김철, “컴퓨터프로그램보호법안심사보고서”, 국회 경제과학위원회, 1986, 6면.

성을 충분히 고려하고, 산업재산권적인 성격이 있는 컴퓨터프로그램의 자유로운 개발을 보장하기 위해 프로그램 언어, 규약, 해법에 대해서는 명확히 법적 보호를 제한한 것으로 보아야 한다. 따라서 위 조항을 단지 ‘아이디어-표현의 이분법’을 성문화한 것으로 볼 것이 아니라, 더 나아가 인터페이스가 프로그램의 형태로 ‘표현’되어 있더라도 법적으로 보호하지 않겠다는 적극적인 의미를 지닌 것으로 보아야 한다.

### (3) 법원의 저작권법 제101조의2 활용도 제고

이처럼 우리 저작권법 제101조의2는 컴퓨터프로그램을 표현하는 수단으로서 필요한 프로그램 언어, 규약, 해법에 대해 저작권법의 적용배제를 규정함으로써 저작권법에 의한 저작권의 물적 보호범위를 제시하고 있다. 프로그램 저작권의 보호범위를 이러한 프로그램 요소들에게까지 미치게 하면, 후속개발 및 소프트웨어 기술의 발전에 지장을 줄 가능성이 생겨나는 것을 방지하고자 하는 것이 그 입법취지라 할 것이다. 그런데 실제로 컴퓨터프로그램과 그 규약 또는 해법을 엄격히 구별하는 것은 매우 곤란하다고 할 수 있다. 따라서 자바 API는 자바 플랫폼과 자바 응용프로그램 사이를 연결시켜 주는 규약(Interface)으로서 저작권물성이 없다고 보아야 한다. 향후 국내에서 오라클이 안드로이드 스마트폰 제조사인 삼성전자나 엘지전자를 상대로 저작권침해 소송을 제기하는 경우, 국내 법원은 저작권법 제101조의2에 따라 오라클의 청구를 기각하는 방안도 고려하여야 할 것이다.

## V. API를 저작권법으로 보호하지 않을 경우의 대안으로서 부정경쟁방지법

### 1. API에 대한 법적 보호의 필요성

컴퓨터프로그램과 같은 디지털 저작물은 완전하고 무한한 복제가 용이하

게 이루어질 수 있을 뿐만 아니라 인터넷을 통해 많은 사람에게 쉽게 유통될 수 있다는 점에서, 무단복제를 방지할 수 있는 제도적 장치가 없다면 아무도 컴퓨터프로그램을 개발하려 하지 않을 것이고, 컴퓨터프로그램 관련 산업은 존립할 수조차 없게 될 것이다. API 역시 개발자의 시간과 노력, 자본이 투입된 지적 창조물로서 경우에 따라 적절한 법적 보호가 필요하다.

## 2. 부정경쟁방지법의 적용 가능성

그런데 위에서 살펴본 바와 같이, API가 저작물성이 없거나(선언코드) 그 이용이 공정이용에 해당하므로(SSO) 저작권법에 의한 보호가 주어지지 않는다고 보아야 한다면, API에 대하여 어떠한 법적 보호가 주어질 수 있는지가 문제된다.

현행법상으로는 부정경쟁방지 및 영업비밀보호에 관한 법률(이하 ‘부정경쟁방지법’이라 한다) 제2조 제1호 (차)목[이하 ‘(차)목’이라고 한다]에 의한 보호가 가장 적절한 방안이 될 것으로 판단된다.<sup>50)</sup> 즉, (차)목은 “그 밖에 타인의 상당한 투자나 노력으로 만들어진 성과 등을 공정한 상거래 관행이나 경쟁 질서에 반하는 방법으로 자신의 영업을 위하여 무단으로 사용함으로써 타인의 경제적 이익을 침해하는 행위”를 부정경쟁행위로 규정하고 있고, 이에 대하여 권리자는 침해자에 대하여 금지청구와 손해배상책임을 물을 수 있는데(부정경쟁방지법 제4조, 제5조), API를 제작하는 데 상당한 투자나 노력이 투입

50) 부정경쟁방지법 (차)목 침해에 대해서는 형사처벌 규정이 없다. 이러한 측면에서 부정경쟁방지법 (차)목에 의한 보호가 저작권법이나 특허법에 의한 보호보다 그 보호의 강도가 낮음은 부인할 수 없다. 한편, API를 특허권으로 보호하는 것도 고려할 수 있으나, 이에 대해서는 컴퓨터프로그램의 특허적격성에 대한 기존의 논의가 그대로 적용될 수 있을 것이다(대표적으로, 권태복, “컴퓨터프로그램 보호방식에 관한 비교 연구”, 한국저작권위원회, 2012. 참조). 이에 대해서 간단한 의견을 밝히자면, 특허 보호가 없더라도 기술혁신이 일어난 대표적인 분야가 소프트웨어 분야이고, 특허가 소프트웨어 기술의 향상에 기여했다는 실증적 증거는 없는 반면, 특허 괴물에 의한 중소기업 또는 개인개발자들의 피해와 선행기술 데이터베이스 부족으로 인한 부실특허의 폐해가 드러나고 있으므로 컴퓨터프로그램에 대한 특허권 보호는 타당하지 않다고 본다.

되었다면, 이를 무단으로 이용하여 제작자의 경제적 이익을 침해하는 행위에 대하여 이 법에 의한 보호가 가능할 것이다.<sup>51)</sup>

이와 관련하여 최근 우리 고등법원은 ‘팜히어로사가 사건(서울고등법원 2017.1.12. 선고 2015나206371 판결, 현재 상고심 계속 중)<sup>52)</sup>에서 (차)목 위반 요건에 대하여 아래와 같이 의미있는 판시를 하였다.

“지식재산권법에 의한 보호의 대상이 되지 않는 타인의 성과인 정보(아이디어)를 이용하는 것은 본래 자유롭게 허용되는 것으로 모방과 이용이 가능하다고 할 것이지만, 그와 같은 타인의 성과 모방이나 이용행위에 공정한 거래질서 및 자유로운 경쟁질서에 비추어 정당화할 수 없는 ‘특별한 사정’이 있는 경우로서 그 지적성과물을 보호해주지 않으면 그 지적성과물을 창출하거나 고객흡입력 있는 정보를 획득한 타인에 대한 인센티브가 부족하게 될 것이 명백한 경우 등에는 그와 같은 모방이나 이용행위가 부정경쟁행위로서 허용될 수 없다고 할 것이다. 따라서 타인의 성과 모방이나 이용행위의 경과, 이용자의 목적 또는 의도, 이용의 방법이나 정도, 이용까지의 시간적 간격, 타인의 성과물의 취득 경위, 이용행위의 결과(선행자의 사업이 과멸적인 영향을 받는 경우 등) 등을 종합적으로 고려하여 거래 관행상 현저히 불공정하다고 볼 수 있는 경우로서, 절취 등 부정한 수단에 의하여 타인의 성과나 아이디어를 취득하거나 선행자와의 계약상 의무나 신의칙에 현저히 반하는 양태의 모방, 건전한 경쟁을 목적으로 하는 성과물의 이용이 아니라 의도적으로 경쟁자의 영업을 방해하거나 경쟁지역에

51) 이상정, “미래 저작권 정책 방향 수립을 위한 연구”, 한국저작권위원회, 2016, 73면

52) [사안의 개요] 외국 게임사인 원고는 2010년경 출시한 ‘매치-3-게임(게임 속의 특정한 타 일들이 3개 이상의 직선으로 연결되면 함께 사라지면서 점수를 획득하도록 고안된 게임)’을 바탕으로 2013년에 ‘팜히어로사가’ 게임(이하 ‘이 사건 원고 게임’이라 한다)을 개발하여 페이스북 플랫폼을 통하여 전 세계에 출시한 이후, 2013.12. 경에는 모바일 플랫폼으로, 2014.6. 경에는 카카오톡 플랫폼으로 이 사건 원고 게임을 각 출시하였다. 국내 게임사인 피고는 2014.2. 경 카카오톡 플랫폼으로 매치-3-게임인 ‘포레스트 매니아’ 게임(이하 ‘이 사건 피고 게임’이라 한다)을 개발 및 출시한 이래, 인터넷을 통하여 이 사건 피고 게임을 제공해왔다. 원고는 이 사건 피고 게임이 이 사건 원고의 게임을 모방한 것이므로 피고가 위 게임을 출시하여 제공하는 행위는 저작권 침해 및 부정경쟁행위에 해당한다고 주장하면서, 피고를 상대로 서비스 중지 및 손해배상을 청구하였다.

[쟁점] 팜히어로사가 사건은 i) 동일하거나 유사한 게임규칙, 전개방식 등을 사용하여 게임을 출시하는 경우, 이러한 게임규칙 등을 저작권 보호 대상으로 볼 수 있는지와 ii) 유사한 게임을 출시하여 제공하는 행위가 타인의 성과물무단사용행위(차목)로 인한 부정경쟁행위 또는 불법행위를 구성하는지 여부가 쟁점이었다.

서 염가로 판매하거나 오로지 손해를 줄 목적으로 성과물을 이용하는 경우, 타인의 성과를 토대로 하여 모방자 자신의 창작적 요소를 가미하는 이른바 예측적 모방이 아닌 타인의 성과를 대부분 그대로 가져오면서 모방자의 창작적 요소가 거의 가미되지 않은 직접적 모방에 해당하는 경우 등에는 예외적으로 타인의 성과 모방이나 이용행위에 공정한 거래질서 및 자유로운 경쟁질서에 비추어 정당화될 수 없는 ‘특별한 사정’이 있는 것으로 보아 민법상 불법행위 또는 부정경쟁방지법 제2조 제1호 (차)목에서 규정하는 부정경쟁행위에 해당한다.”

팜히어로사가 사건은 (차)목의 적용에 있어서 지식재산권법에 의하여 보호되지 않는 성과에 대한 모방이나 이용행위는 원칙적으로 허용되나 공정한 거래질서 및 자유로운 경쟁질서에 비추어 정당화될 수 없는 ‘특별한 사정’이 있는 경우에만 규제 대상이 된다는 것을 구체적으로 적시하였다는 점에서 의의가 큰 판결이다.<sup>53)</sup> 이는 결국 원칙적으로 경쟁을 허용하되, 각 사건의 개별·구체적인 사정에 비추어 정당화될 수 없는 ‘특별한 사정’이 있는 경우에는 규제를 하겠다는 것이므로, 저작권법에 의한 API의 경직된 보호의 대안으로 충분히 검토할 여지가 있다고 보인다. 물론 (차)목의 입법에 참고가 되었던 기존의 판례들, 즉 개별 지식재산권법에 의한 보호요건을 갖지 못한 경우에도 불법행위책임이 성립될 수 있고, 그 구제수단으로 침해행위의 금지 또는 예방을 구할 수 있다는 판례들<sup>54)</sup>을 고려하면, (차)목에 해당하지 않아도 일반불법행위 책임으로 보호받을 수도 있을 것이다.

## VI. 결 론

오라클 대 구글 사건의 쟁점은 구글이 자바 API의 166개의 패키지 중 37개 패키지의 7,000줄에 이르는 선언코드와 위 37개 패키지의 SSO를 복제한

53) 법무법인(유한) 화우, “게임물의 저작권 침해 및 부정경쟁행위 해당 여부에 관한 판결”, 법무법인(유한) 화우, 2017, 3면.

54) 대법원 2012.3.29. 선고 2010다20044 판결; 대법원 2010.8.25. 자 2008마1541 결정.

것이 오라클의 저작권을 침해하였는지 여부이다. ① 자바 API 저작권의 문언적 요소(7,000줄의 선언코드)에 관하여 살펴보면, 합체의 원칙과 필수장면의 원칙은 저작물 성립요건으로 보아야 하고, 자바 API 선언코드는 이들 원칙에 따라 저작물성이 없고, ② 자바 API 저작권의 비문언적 요소(37개 자바 API 패키지의 SSO)에 관하여 살펴보면, 자바 API가 사실상의 표준인지 여부는 저작물성과는 관련이 없지만 공정이용 항변의 논거로 볼 수 있고, 자바 API의 SSO는 로터스 판결과는 결론에서는 동일하게 저작권으로 보호받을 수는 없다고 보아야 한다. 그리고 국내에서 자바 API의 저작권법에 의한 보호 가능성이 문제된다면, 위와 같은 내용에 더해 우리 저작권법 제101조의2에 근거하여 저작권법에 의한 보호를 거부할 수도 있을 것이다. 다만, 최근 우리 하급심 판결 취지에 따르면 부정경쟁방지법에 의한 보호 가능성은 열려 있다고 하겠다.

자바 API의 저작권을 놓고 오라클과 구글 사이에 7년 이상 소송이 계속되는 동안, 자바 개발자 커뮤니티는 저작권을 주장하는 오라클의 조치에 실망하며 점점 동력을 잃어가고 있다. 그리고 항소심 판결에 따라 저작권 침해에 의한 사용금지 및 손해배상의 위험을 안고 있던 구글은 결국 2017년 5월에 개최된 개발자 대회(Google I/O 2017)에서 코틀린(Kotlin)을 안드로이드 플랫폼의 공식 지원 언어로 추가하면서<sup>55)</sup> 향후 안드로이드 플랫폼에서 자바를 코틀린으로 대체한다는 계획을 발표하기에 이르렀다.

만일, 구글이 Sun과의 협상이 결렬된 이후 자바 API 저작권 침해에 대한 우려로 안드로이드 플랫폼 개발을 단념했다면 현재 어떻게 되었을까. 단언하기는 어렵지만 아마도 스마트폰 시장은 아이폰이 장악하거나, 심비안과 같은 기존의 모바일 OS가 그대로 사용되는 등 시장의 경쟁이 사라져 지금 우리가 보고 있는 많은 모바일 혁신이 진행되지 않거나, 더디게 진행되었을 지도 모른다. 소프트웨어 기술 혁신을 위한 저작권법의 역할을 다시 생각해 볼 시점이다.

---

55) 이전에는 자바만이 지원되었다.

## 참고문헌

### 〈단행본(국내 및 동양)〉

- 김규성, 『컴퓨터프로그램 보호법』, 세창출판사, 2007.
- 오승중, 『저작권법』, 제3판, 박영사, 2013.
- 이규호, 『저작권법』, 제5판, 진원사, 2015.
- 이기수, 『지적재산권법』, 제2판, 한빛지적소유권센터, 1996.
- 이해완, 『저작권법』, 제3판, 박영사, 2015.
- 임원선, 『실무자를 위한 저작권법』, 제4판, 한국저작권위원회, 2015.
- 정지훈, 『거의 모든 IT의 역사』, 메디치미디어, 2010.
- 특허청, 『특허법과 저작권법의 조화를 통한 창조적 소프트웨어 기업 보호방안 연구』, 진한엠앤비, 2015.

### 〈학술지(국내 및 동양)〉

- 강이삭, 박성필 · 김용길, “상표권 침해정보제공에 따른 정보통신서비스 제공자의 규제에 관한 문제점”, 『지식재산연구』, 제12권 제1호(2017).
- 김용길, “지식재산권분쟁의 재판의 해결제도에 관한 연구—조정과 중재를 중심으로”, 『중재연구』, 제19권 제1호(2009).
- 김철, “컴퓨터프로그램보호법안심사보고서”, 국회 경제과학위원회, 1986.
- 박성필 · 김용길, “디지털시대의 법률환경 변화와 미디어산업의 대안적 비즈니스모델의 모색”, 『지식재산연구』, 제1권 제2호(2006).
- 박성필 · 김용길, “오픈 이노베이션 전략의 법적 함의: SCO 대 IBM 소송을 중심으로”, 『지식재산연구』, 제8권 제4호(2013).
- 中山信弘, “創作性についての基本的考え方 (シンポジウム 創作性)”, 『著作権研究』, Vol.28(2001).

### 〈학술지(서양)〉

- Abrahamson, Scott, “Seen One, Seen Them All—Making Sense of the Copyright Merger Doctrine”, *UCLA L. Rev.*, Vol.45(1997).
- Asay, Clark D., “Software's Copyright Anticommons”, *Emory Law Journal*, Vol.66 No.2(2016).
- Daria Vasilescu-Palermo, “APIs and Copyright Protection: The Potential Impact on Software Compatability in The Programming Industry”, *J. Marshall Rev. Intell.*

*Prop. L.*, Vol.16(2016).

Deba Alam, "Oracle America, Inc. V. Google, Inc.: The Battle Over APIs Continues", *DePaul J. Art Tech. & Intell. Prop. L.*, Vol.26(2015).

Kravetz, Paul I., "Idea/Expression Dichotomy and Method of Operation: Determining Copyright Protection for Computer Programs", *DePaul Bus. LJ*, Vol.8(1995).

Ocasio, Sandro, "Pruning Paracopyright Protections: Why Courts Should Apply the Merger and Scenes a Faire Doctrines at the Copyrightability Stage of the Copyright Infringement Analysis", *Seton Hall Cir. Rev.*, Vol.3(2006).

Tyler J. Demasky, "Oracle V. Google: Setting a Standard pr Handicapping an Industry?", *N.C. J.L. & Tech.*, Vol.18 On,1(2016).

#### 〈기타 자료〉

권태복, "컴퓨터프로그램 보호방식에 관한 비교 연구", 한국저작권위원회, 2012.

김병일, "프로그램저작권 제한규정 개선을 위한 연구", 한국저작권위원회, 2009.

법무법인(유한) 화우, "게임물의 저작권 침해 및 부정경쟁행위 해당 여부에 관한 판결", 법무법인(유한) 화우, 2017.

이상정, "미래 저작권 정책 방향 수립을 위한 연구", 한국저작권위원회, 2016.

## A Study on the Copyright Protection of the API —Focusing on the Copyrightability of the JAVA API—

Kwon Sahyun\*, Park Sungpil\*\*, Kim Yongkil\*\*\*

Due to the widespread impact of Java on our reality, the developer community seems to be concerned about the wave of copyright consequences between Oracle and Google. In particular, there is a high concern that the risk of copyright infringement will ruin the characteristics of the development environment, which Java is capable of “bringing the old, anytime, anywhere”. In conclusion, it is regrettable that the CAFC adhered to the rigid theories of copyrightability and accepted the copyrightability of the API without fully considering the reality of the program development environment. If Google has given up on Android development since the breakdown with Sun, many of the mobile innovations we see may have been slow. In our interpretation of the copyright law, the API is not protected by the copyright law because it is not ‘creativity’ or is excluded from the scope of copyright protection pursuant to Article 101-2. In this case, APIs that need protection should

---

\* Attorney of DR & AJU INTERNATIONAL LAW GROUP LLC, Graduate School of Future Strategy, studying to attain Master’s degree of Intellectual Property(Master of Engineering/MBA), Member of Korean Bar(First Author).

\*\* Head Professor, Graduate School of Intellectual Property Program, Korea Advanced Science & Technology (“KAIST”), Member of Michigan Bar, Doctor of Juridical Science(Correspondence Author).

\*\*\* Professor in the Graduated School of Law, Wonkwang University, Doctor in Law(Co-Author).

seek protection under the Patent Act or the Unfair Competition Prevention Act.

Keyword

JAVA, API, Copyrightability, Merger Doctrine, De Facto Standard, Interface, Declaration, Implementation