creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

사회학석사학위논문

개방적 정보기술을 통한 권력 확장에 관한 연구

: 구글의 안드로이드 스마트폰 사례를 중심으로

2014년 8월

서울대학교 대학원

사 회 학 과

김 수 종

개방적 정보기술을 통한 권력 확장에 관한 연구

: 구글의 안드로이드 스마트폰 사례를 중심으로

지도교수 서 이 종

이 논문을 사회학석사 학위논문으로 제출함.

2014년 7월

서울대학교 대학원

사 회 학 과

김 수 종

김수종의 사회학석사 학위논문을 인준함.

2014년 7월

위 원 장 _____ (인)

부 위 원 장 _____ (인)

위 원 _____ (인)

요약(국문초록)

본 논문은 개방적 정보기술이 특정 행위자의 권력을 확장시키게 되는 기제를 확인하는 것을 그 목표로 한다. 다양한 행위자의 생산 참여, 기술의 자유로운 사용과 변형을 통한 기술의 개방, 그리고 권력 확장을 가능하게 하는 기술의 지배는 표면적으로는 양립하기 어렵다. 본 논문에서는 모바일 운영체제인 안드로이드 운영체제 및 관련 기술적 요소를 연구 대상으로 하여, 이러한 개방 구조와 지배 구조가 결합된 방식을 행위자-네트워크 이론과 기술체계이론을 통해 분석하였다.

우선 안드로이드 운영체제를 중심으로 구성된 인간, 비인간 행위자의 네트워크인 '안드로이드 체계'를 크게 세 영역으로 나누고, 각 영역에서의 지배 구조를 확인하였다. 첫째, 안드로이드 운영체제 영역에서는 생산에 대한 개방적 참여가 이루어지나, 구글은 기술의 통합과정과 공식화 과정을 장악함으로써 생산 과정과 결과물을 지배하고 있다. 둘째, 모바일 어플리케이션 영역에서 기술은 많은 행위자가 안드로이드 어플리케이션을 생산할 수 있도록 설계되어 있으며 이에 따라 전 세계에서 지역적, 문화적 특성에 따라 다양한 어플리케이션이 생산, 사용되고 있으나, 이러한 모바일 어플리케이션들은 대부분 구글의 정보하부구조에 의존하고 있다. 마지막으로 스마트폰 기기의 영역에서, 오픈소스 라이선스는 자유롭게 기술을 변형하고 사유화하길 원하는 기기 제조사들이 안드로이드 운영체제를 선택하도록 하는 요인이 되었으며 실제 많은 제조사들에 의해 다양한 안드로이드 스마트폰 기기가 생산되고 있으나, 구글은 이러한 기기들이 만족시켜야 하는 정합성의 기준을 통제함으로써 제조사들과 이들의 생산품을 지배하고 있다.

안드로이드 체계에서 이러한 지배 구조들은 서로 결합하여 작용한다. 구글은 구글플레이스토어라는 사유화된 기술을 의무통과점으로 설정하여 기기 제조사, 어플리케이션 개발자를 자신의 기술체계에 편입시킨다. 이후 체계 내에서 각 행위자의 지위는 변동하며, 이 과정에서 정합성, 안정성은 권력을 확장하는 수단으로 사용된다. 이와 같이 기술의 개방성에도 불구하고 기술의 생산 구조와 생산되는 기술의 구조를 통해 지배와 권력 확장이 이

루어질 수 있다. 그리고 중국과 한국의 사례 비교를 통해 동일한 기술체계일지라도 지배 구조의 유지 여부에 따라 전혀 다른 양상을 보일 수 있음을 확인할 수 있다.

이와 같이 하나의 기술체계에서도 개방과 지배가 공존할 수 있다. 하지만 공존의 양상은 기술적 종속에서부터 문화적 분화에 이르기까지 많은 차이를 보일 수 있다. 사물의 인터넷의 도래 등 기술적 변화로 인해 개방성이 강화되고 있는 시기에 사적 권력의 지나친 확장과 이로 인한 기술적 종속을 피하기 위해서는 기술체계의 지배 구조에 대한 사회, 국가적 차원의 견제와 개입이 필요하다.

주요어 :

행위자-네트워크 이론, 기술체계, 개방적 기술, 권력, 구글, 안드로이드

학 번 : 2012 - 20081

제목 차례

I. 서론 ····· 1
1. 문제제기 ····· 1
2. 선행연구검토 ····· 4
(1) 정보기술과 코드(code) ····· 4
(2) 구글에 대한 비판 ····· 5
3. 이론적 자원 ····· 8
(1) 행위자-연결망 이론 ····· 8
(2) 기술체계이론 ····· 9
(3) 행위자-연결망 이론과 기술체계 이론에서의 권력 ····· 12
4. 연구 설계 ····· 14
(1) 연구 대상 ····· 14
1) 정보기술의 특성 ····· 15
2) 스마트폰 기술의 구조 ····· 16
3) 안드로이드 운영체제의 특성 ····· 18
4) 구글과 안드로이드 운영체제 ····· 20
(2) 연구 방법 ····· 21
(3) 연구 자료 ····· 22
(4) 논문의 구성 ····· 23
II. 안드로이드 운영체제 영역에서의 개방성과 지배 ····· 25
1. 안드로이드 운영체제 영역의 개방성 ····· 25
(1) 안드로이드 운영체제 생산에 대한 개방적 참여와 그 방식 ····· 25
(2) 기술의 공식화와 참여의 심화 ····· 29
2. 안정성의 지배: 구글에 의한 운영체제의 규정 ····· 31
(1) 코드통합과정에서 구글의 지위 ····· 31
1) 편중된 기여 ····· 31
2) 편향적 검증 ····· 34
3) 역할 장악 ····· 35
(2) 공식화 과정에서 구글의 역할 ····· 36
(3) 구글의 권한 배분과 기술체계의 배타성 ····· 39
III. 모바일 어플리케이션 영역의 개방성과 지배 ····· 44
1. 모바일 어플리케이션 영역의 개방성 ····· 44
(1) 개발자의 참여 확장을 위한 기술적 설계 ····· 44
1) 일반적 언어를 확보하기 위한 기술적 동원 ····· 44

2) 일반적 도구를 확보하기 위한 기술적 삽입 ····· 49
(2) 모바일 어플리케이션의 맥락화 ····· 53
2. 구글의 정보하부구조에 대한 기술적 의존 ····· 60
(1) 기술적 공동화 ····· 60
(2) 구글에 대한 기술적 의존 ····· 63
1) 사적 정보하부구조의 유형과 비중 ····· 63
2) 구글의 정보하부구조의 유형과 의존 요인 ····· 67
IV. 스마트폰 기기 영역의 개방성과 지배 ····· 71
1. 스마트폰 기기 영역의 개방성 ····· 71
(1) 안드로이드 운영체제의 라이선스 구조 ····· 71
(2) 기술의 불투명성과 체계의 확장 ····· 73
(3) 기술적 요소의 다양화 ····· 77
2. 정합성의 지배: 구글에 의한 기기의 규정 ····· 83
(1) 안드로이드 운영체제의 정합성 ····· 83
(2) 정합성의 정의와 외부 행위자의 통제 ····· 85
V. 기술체계의 결합과 구글의 권력 확장 ····· 90
1. 의무통과점의 형성과 하위체계의 결합 ····· 90
(1) 기술적 의존 구조의 형성 ····· 90
(2) 기술체계의 결합과 구글의 목적 ····· 93
2. 행위자의 지위변동 ····· 100
(1) 공적 기술의 쇠퇴와 기술적 사영화 ····· 100
(2) 행위자의 지위 변동 ····· 104
3. 기술의 지배를 통한 권력의 확장 ····· 108
(1) 기술체계 내 지위에 따른 상호작용 방식 ····· 108
1) 모바일 어플리케이션에 대한 방관 ····· 108
2) 제조사에 대한 통제의 강화 ····· 109
(2) 기술을 통한 권력의 확장 ····· 110
1) 권력으로서의 정합성: 기술적 기준의 정치화 ····· 110
2) 권력으로서의 안정성: 기술체계의 경계 작업 ····· 120
(3) 제도와 개방적 기술 체계: 한국과 중국의 사례 ····· 122
1) 중국의 사례: 의존구조의 배재와 기술의 지역적 변용 ····· 122
2) 한국의 사례: 경제적, 정책적 의존과 구글의 권력 강화 ····· 125
VI. 결론 ····· 129
VII. 참고문헌 ····· 131
VIII. 부록 ····· 136

표 차례

<표 1> 안드로이드 체계의 주요 구성요소 ····· 14
<표 2> 연구 자료 ····· 22
<표 3> 개방핸드셋연합 가입기업 ····· 26
<표 4> 안드로이드 운영체제 코드 기여자의 소속 ····· 32
<표 5> 코드 기여 집단별 코드 등록 건수 및 통과·거부율 ····· 35
<표 6> 안드로이드 운영체제 검증자 배정 ····· 35
<표 7> 안드로이드 초기 탑재 기기 ····· 39
<표 8> 안드로이드 운영체제 버전별 플래그십 기기 ····· 41
<표 9> 데이터 수집 일자와 수집 국가 ····· 54
<표 10> 사용국가 수에 따른 어플리케이션 수 및 목록 총 등장횟수 ····· 54
<표 11> 기술권역의 구분 ····· 58
<표 12> 상위 10개 개발도구 API의 점유율 ····· 63
<표 13> 상위 5개 광고 API의 점유율 ····· 64
<표 14> 구글에 의해 만들어진 코드의 기능과 확인 방법 ····· 65
<표 15> 구글코드를 포함하고 있는 어플리케이션의 비율 ····· 64
<표 16> 화면 크기와 해상도의 분포 ····· 79
<표 17> 안드로이드 운영체제 4.3 정합성정의문서의 세부항목 ····· 88
<표 18> 구글의 광고매출 비중. 단위는 백만 달러. ····· 98
<표 19> 안드로이드 운영체제 버전별 정합성기준의 유형과 수 ····· 111
<표 20> 안드로이드 버전별 정합성기준의 항목 비교 ····· 113
<표 21> 추가된 정합성기준 항목들의 내용 ····· 118
<표 22> 제도적 규제의 결과 ····· 127
<표 23> 개방 구조와 지배 구조의 강도에 따른 기술체계의 유형 ····· 129
<표 24> 평균 유사성 행렬 ····· 136
<표 25> 기술거리 행렬 ····· 137
<표 26> 기술사용 거리의 계산 사례 (기준국가: 미국) ····· 138
<표 27> 어플리케이션의 구글코드 사용 여부 ····· 139
<표 28> 안드로이드 운영체제의 변화 ····· 154

그림 차례

<그림 1> 기술적 행위의 혼종성 ····· 15
<그림 2> 스마트폰에서 기술적 요소간의 상호작용 ····· 17
<그림 3> 안드로이드 운영체제에 기부된 코드 목록 ····· 28
<그림 4> 안드로이드 운영체제 코드의 코드 관리 절차 ····· 37
<그림 5> 안드로이드 운영체제의 구조 ····· 45
<그림 6> 안드로이드 체계에 의한 자바 체계의 훼손 ····· 48
<그림 7> 이클립스에 ADT가 설치된 모습 ····· 51
<그림 8> 모바일 어플리케이션 개발 경력에 따른 모바일 운영체제 선호도 ····· 52
<그림 9> 한국을 기준으로 한 기술거리의 분포 ····· 55
<그림 10> 중국을 기준으로 한 기술거리의 분포 ····· 56
<그림 11> 일본을 기준으로 한 기술거리의 분포 ····· 56
<그림 12> 인도네시아를 기준으로 한 기술거리의 분포 ····· 56
<그림 13> 기술권역의 분포 ····· 58
<그림 14> 개발자와 사용자 간 소통 채널의 확보 ····· 60
<그림 15> 구글 지도를 사용하는 교통, 여행 관련 어플리케이션의 예 ····· 66
<그림 16> AOSP 홈페이지에 포함되어 있는 구글 관련 정보들 ····· 69
<그림 17> 각 운영체제 별 스마트폰 화면 크기의 상대적인 비교 ····· 78
<그림 18> 안드로이드 운영체제에서 화면 크기와 해상도의 구분 ····· 80
<그림 19> 시간에 따른 안드로이드 버전별 기기 수 분포 ····· 81
<그림 20> 구글 API와 다른 코드 간의 상호작용 ····· 91
<그림 21> 스마트폰 기기와 어플리케이션 간 결합을 위한 기술적 우회 ····· 96
<그림 22> 의무통과점으로서의 구글플레이스토어 ····· 97
<그림 23> 구글의 목표 ····· 97
<그림 24> 안드로이드 기본 기능과 구글 어플리케이션의 기능 비교 ····· 102
<그림 25> 스마트폰 기기의 형태 비교. ····· 105
<그림 26> 안드로이드 체계의 관계망 ····· 107
<그림 27> 정합성기준의 필수, 권장 항목의 증가 ····· 112
<그림 28> 안드로이드 어플리케이션 시장의 종류 ····· 124
<그림 29> 국가별 모바일 운영체제 점유율 비교. ····· 126
<그림 30> 안드로이드 코드 반영 절차 ····· 156

I. 서론

1. 문제제기

20세기 말 인간은 개인 컴퓨터(personal computer)와 인터넷(Internet)의 보급으로 누구나 정보를 처리, 저장, 교환할 수 있는 정보 사회(information society)의 도래를 목격하였다. 그리고 최근에는 스마트폰(smartphone) 사용자 수가 세계적인 증가 추세를 보이고 있다. 스마트폰 기술은 개인컴퓨터, 휴대폰 등의 다른 정보기술과 비교하여 몇 가지 특징을 보이고 있다. 스마트폰은 개인컴퓨터와 달리 휴대가 용이하며 이동 중에도 인터넷에 접속할 수 있다. 또한 스마트폰 사용자는 선호에 따라 기기의 기능을 변경, 추가할 수 있다.

스마트폰 기술의 확산을 통해 정보사회는 더욱 심화되고 있다. 사람들은 길을 가다 스마트폰으로 사진을 찍고 이를 바로 인터넷으로 공유하며, 약속 장소로 걸어가면서 스마트폰으로 최단경로 및 소요 시간을 계산해내고, 출퇴근길에 인터넷에 접속해 해외의 소식을 실시간으로 검색한다. 또한 스마트폰 사용자는 필요한 여러 가지 기능을 언제 어디서나 직접 추가, 변경할 수 있게 되었다. 이와 같이 사용자의 정보 교환 활동이 컴퓨터가 존재하는 고정된 위치에서만 일어났던 이전과 달리, 스마트폰의 확산에 따라 정보기술의 사용은 이동성을 갖게 되었다. 또한 스마트폰을 통해 필요한 정보기술을 사용자가 직접 선택, 사용할 수 있게 되었다. 이와 같이 스마트폰의 확산은 정보기술 사용의 시공간을 확장하고, 사용되는 정보기술을 개인별로 개별화시켰다.

개방성은 이러한 스마트폰 기술의 확산에 기여할 수 있다. 특히 전 세계 스마트폰 운영체제의 대부분을 차지하고 있는 안드로이드 운영체제는 개방성을 표방하고 있다. 즉 누구나 이 운영체제의 개발에 참여할 수 있고, 인터넷 상에서 안드로이드 운영체제를 다운로드 받거나 수정, 변형, 재배포할 수 있다. 또한 안드로이드 운영체제에서 동작하는 모바일 어플리케이션에 대한 지식과 개발 도구를 쉽게 구할 수 있기 때문에 누구나 모바일 어플리

케이션을 만들 수 있다. 이로 인해 많은 수의 사람들이 안드로이드 스마트폰과 관련된 기술을 개발하고 이를 사용하게 되었다.

정보기술의 개방성과 확산으로 인해 정보기술은 사용되는 사회, 정치, 문화적 맥락에 따라 다양한 방식으로 변화될 수 있다. 특히 소프트웨어와 같은 정보기술은 누구나 특별한 도구 없이도 만들고 수정할 수 있기 때문에 다양한 행위자들에 의해 쉽게 확산되고 다양화된다. 더군다나 정보기술이 일반에 공개되어 있고, 새로운 기술을 만들기 위해 필요한 정보와 도구를 쉽게 얻을 수 있을 경우 확산과 다양화의 압력은 크다.

정보기술의 분화, 다양화는 정보기술을 새로운 생산자, 사용자와 관련 기술의 관계망 속에 위치시킨다. 이에 따라 하나의 행위자가 관련 행위자들을 통제하기는 어려워지고 정보기술과 관련된 행위자들의 지위가 변화해간다. 하지만 실제로는 안드로이드 스마트폰 기술의 확산에 따라 정보통신기업인 구글(Google) 사의 권력이 강화되고 있다. 특히 구글은 기기 제조사, 소프트웨어 개발자, 사용자, 이동통신망 사업자 등에 대해 강력한 영향력을 미치고 있다. 더욱이 이러한 영향력은 행위자의 수가 증가하고 이에 따른 분화의 압력이 거세질수록 강해지고 있다. 개방과 지배의 공존이라는 모순적 현상은 기술체계에 대한 더욱 구체적이고 실증적인 분석을 요구하고 있다.

이에 따라 본 연구에서는 “어떻게 개방적 정보기술을 통해 특정 행위자의 권력이 확장될 수 있는가?”라는 질문에 답하고자 한다. 이 질문에 답하기 위해 구체적으로 다음과 같은 연구 질문을 설정하였다.

첫째, 안드로이드 운영체제 생산은 어떻게 이루어지는가? 특정 행위자는 이러한 생산 체제를 어떻게 지배할 수 있는가?

둘째, 모바일 어플리케이션의 생산은 어떻게 이루어지는가? 특정 행위자는 이러한 생산 체제를 어떻게 지배할 수 있는가?

셋째, 스마트폰 기기의 생산은 어떻게 이루어지는가? 특정 행위자는 이러한 생산 체제를 어떻게 지배할 수 있는가?

넷째, 안드로이드 운영체제, 어플리케이션, 기기의 결합을 통해 전체 지배 구조는 어떠한 특성을 가지게 되는가? 각 행위자의 지위는 어떻게 변화

하는가? 우월한 지위를 점유한 행위자는 어떠한 방식으로 다른 행위자에게 영향을 미치는가?

2. 선행연구검토

(1) 정보기술과 코드(code)

정보기술의 발전과 확산으로 인한 사회 변화는 지금까지 사회과학에서 주요 연구주제 중 하나였다. 하지만 이러한 연구들은 주로 인간 행위자 간의 정보 이동에 초점을 맞추고 있다. 정보기술을 통해 이동하는 정보에는 게시판 이용자 간의 논쟁, 게임에서 사용자가 조정하는 캐릭터의 움직임, 메신저 프로그램에서 지인 간의 사소한 대화 등 담길 수 있다. 하지만 데이터의 저장과 이동만을 관찰해서는 정보 사회의 특성을 명확하게 확인할 수 없다. 왜냐하면 이러한 관점이 '코드(code)'[1]라는 정보기술의 구조적 요소를 간과하고 있기 때문이다. 코드는 정보의 흐름을 규정하지만, 코드 그 자체도 전기적 신호의 덩어리로 쉽게 변형이나 이동이 가능하다.

정보기술의 사용자와 관련된 정보는 모두 코드에 의해 전자적 신호 혹은 비트(bit)로 변환되어 처리되기 때문에, 이러한 코드를 넘어선 사용자는 존재하지 않는다. 예를 들어 스마트폰 사용자가 화면에 제시된 버튼을 누르지 않고 스마트폰을 사용할 수는 없으며, 제시된 버튼 외에 다른 선택은 주어지지 않는다. 정보기술의 사용자는 코드에 의해서 정해진 범위 내에서만 행위할 수 있으며, 코드에 의해 규정되는 영역 이외의 것은 기술적으로 가능하지 않다. 이와 같이 사용자가 기술에 의해 규정된 일정한 틀 안에서만 행동하게 된다는 점에서 정보기술은 '몰입적'이다. 인터넷에 대한 상징적 비유의 하나인 '사이버 공간(cyber space)'도 마치 공간적 구조에 의해

1) 정보기술은 정보를 수집, 저장, 처리, 분배하는 것에 관련된 기술이며, 이를 구현하기 위해서 하드웨어와 소프트웨어가 동원된다. 여기서 하드웨어(hardware)란 메모리 칩, 연산을 담당하는 부품과 같이 기술의 물리적인 요소들을 가리키는 반면 소프트웨어(software)는 이러한 물리적 요소들을 제어, 통제하여 동작하게 하는 논리적이며 비물질적인 부분을 가리킨다. 소프트웨어는 코드로 이루어져 있다. 코드는 컴퓨터를 동작시키기 위해 쓰이는 프로그래밍 언어(programming language)의 덩어리로서 기술체계에서 데이터의 흐름을 규정한다. 코드는 인간에 의해 작성되고 이는 컴퓨터가 처리할 수 있는 이진법 형태의 언어인 기계어(machine language)로 번역되며, 기계는 번역된 언어에 따라서 동작한다. 컴퓨터 공학에서는, 개발자가 코드를 작성하는 작업을 프로그래밍이라고 하고, 프로그래밍으로 만들어진 코드를 이진법의 기계어로 번역하는 것을 컴파일(compile)이라고 한다.

서 사람들의 행동 범위가 제약되듯 코드로 이루어진 기술적 한계에 의해서 인터넷 사용자의 행위가 제약되는 특성을 나타내고 있다는 점에서 정보기술의 이러한 몰입성과 연관되어 있다. 하지만 지금까지 사이버 공간에 대한 연구는 이러한 공간의 구조적 특성 보다는 주어진 공간 내부에서 정보의 교환을 통해 일어나는 사람들 간의 상호작용 방식에 편중되어 있다.

레식(Lessig, 2006)은 정보기술의 사용자가 코드에 의해 제약, 통제된다는 점에 착안하여, 코드를 통해 사이버 공간을 법률적으로 규제함으로써 지적 재산권, 언론의 자유, 표현의 자유 등을 보호할 수 있다고 주장한다. '코드는 법이다(code is law)'라는 그의 주장은 지금껏 잘 고려되지 않았던 정보기술의 몰입성과 제약성을 명확히 드러내고 있을 뿐만 아니라, 기술로 창조된 새로운 영역에서 인간의 자유와 권리를 보장하기 위한 규범적 수단으로 코드를 고려했다는 점에서 혁신적이라고 할 수 있다. 하지만 그의 주장은 두 가지 측면에서 비판될 수 있다. 우선 그의 주장은 정보기술의 투명성을 전제로 하고 있다. 코드에 대한, 코드를 통한 규제는 정보기술의 코드가 공개되어 있거나 코드의 동작 방식에 대한 구체적인 지식을 얻을 수 있을 경우에만 가능하며, 기업들이 사적으로 보유하고 있는 기술은 많은 경우 코드나 내부 동작을 확인하기 어렵다. 둘째 레식의 주장은 코드가 사용자에 영향을 준다는 사이버 공간의 단선적 개념화에 근거해 있다. 이 때문에 그의 규제 방안은 사이버 공간에서의 인간을 단지 사용자로만 간주한다. 하지만 코드는 사이버 공간에서 개인이나 집단에 의해 다시 생산되기도 하기 때문에, 사용과 생산을 오고가는 복합적이며 역동적인 과정 대한 고려가 필요하다.[2)]

(2) 구글에 대한 비판

정보통신업체인 구글이 제공하는 다양한 기술들의 사회적 영향력이 증대하고 있다. 예를 들어 구글의 검색 시스템은 우리가 정보를 찾고 이를 사

2) 예를 들어 사용자들은 온라인 게임의 규칙, 게시판의 정책, 커뮤니티의 개인정보 공개 수위 등을 결정할 수 있으며, 이러한 결정에 따라 사이버 공간의 구조, 즉 코드가 변화하거나 새로이 생성된다.

용하는 방식에 근본적인 변화를 가져왔다. 구글의 서비스가 대부분 무료임에도 불구하고 구글이 사용자의 개인정보를 수집하고 이를 광고주에게 제공함으로써 수익을 얻는다는 점, 즉 우리가 구글의 생산자이자 생산품이라는 점에서 구글의 정보기술에 대한 더욱 면밀한 분석이 필요하다(Olsthoorn, 2012). 더욱이 최근에는 세상의 모든 정보를 체계화하겠다는 구글의 목표, 그리고 '사악해지지 말자(Don't be evil)'는 구글의 모토(motto)를 더 이상 신뢰할 수 없다는 우려도 증가하고 있다.

구글에 대한 비판 중 가장 주목할 만한 것이 구글화 혹은 구글라이제이션(Googlization)에 관한 시바 바이디야나단의 비판이다(Vaidhyanathan, 2012). 구글은 현대의 거대한 도서관이자 파일저장고이며, 정부 기록의 창고이자 각종 상품 자료의 보관소로서 인류가 드나들 수 있는 거대한 통로를 제공하였다. 미국의 미디어 학자이자 법학자인 바이디야나단은 이러한 방식으로 모든 것이 구글화되고 있다고 주장한다. 구글화는 크게 세 가지 측면에서 인류의 관심과 행동에 영향을 미친다. 우선 구글은 개인의 정보, 습관, 의견 판단에 영향을 미친다. 또한 글로벌 감시체제와 '인프라 제국주의'를 통해 세계 전체에 영향을 미친다. 뿐만 아니라 구글은 책, 데이터베이스, 웹에 축적된 엄청난 지식의 사용에 영향을 미친다. 바이디야나단은 무비판적인 신뢰를 누그러뜨리고 구글을 면밀히 검토해야 하며, 이를 위해 사용자가 구글의 고객이 아닌 상품이라는 점을 깨달아야 한다고 주장한다. 구글은 사용자의 욕망과 선호를 광고주에게 파는 것이다.

바이디야나단은 구글의 서비스와 이러한 서비스를 구성하는 구글의 기술 전반을 면밀하게 관찰하여 많은 정보를 제공하고 있다. 더욱이 진화된 정보기술을 통해 특정 기업이 전 세계 사람들의 일상에 광범위한 영향을 미치고 이를 통해 이익을 얻는 현상에 대한 비판은 매우 시의적절해 보인다. 바이디야나단의 비판은 실재적 위험에 대한 분석에 기반한 것이 아니며, 구글의 기술이 가진 '잠재적' 위험에 대한 우려에 그치고 있다. 또한 구글 시스템을 구성하는 요소 간 상호작용을 심층적으로 기술하거나 진화과정에서 구글과 다른 행위자 간의 상호작용을 조망하는 등의 체계적인 접근이 이루어지지 않은 채 논란거리의 나열에 그치고 있다는 점에서 바이디야나

단의 분석을 사회과학적 접근으로 보기도 어렵다. 뿐만 아니라 구글의 기술적 발전, 확장에 대한 일부 우려는, 정보기술이 발전하고 인간의 의존이 커짐에 따라 기술이 우리를 지배하게 될 것이라는 기술적 결정론적 관점에서 비롯하고 있다. 아울러 기술 결정론적 우려에 구글에 대한 음모론적 불신이 더해진다. 거대 기업의 의도가 사악할 수도 있기 때문에 이러한 지배가 우리에게 잠재적으로 부정적인 결과를 가져올 수 있다는 것이다.

본 연구는 구글라이제이션을 비판적으로 수용, 확장한다. 구글라이제이션은 민간 기업에 의한 정보기술과 네트워크 사적 확장을 잘 개념화하고 있다. 하지만 이를 기업의 팽창으로 보는 것은 현상을 지나치게 단순화하는 것이다. 오히려 중요한 것은 구글이 새로운 개발자, 제조사, 새로운 기술과 협상하고 이들을 포섭함으로써 자신의 네트워크를 확장하고 있다는 것이다. 따라서 구글이 시스템을 건설하기 위해 인간, 비인간 적 요소를 어떻게 동원하였고 이러한 동원에 의해 만들어진 네트워크가 어떠한 특성을 가지고 있는지 살펴야 한다. 이를 통해 '전지전능'한 구글의 대한 결정론적 두려움에서 벗어나, 구글이 여러 행위자를 자신의 의도에 맞게 정렬하는 방식과 정렬된 네트워크를 통해 행위와 권력이 어떠한 방식으로 전파, 굴절되는지를 확인한다.

3. 이론적 자원

(1) 행위자-연결망 이론

행위자-연결망 이론(Actor-Network Theory)에 의하면 현상(state of affair)에 변화를 주는 것은 무엇이든 행위성을 가지는 것이므로, 인간뿐만 아니라 비인간(non-human)도 행위자가 될 수 있다. 행위자 연결망 이론에서는 비인간 행위자(non-human actor)나 행위소(actant)와 같은 용어를 사용하며, 이는 각각 비인간의 행위성을 지칭하거나 비인간, 인간의 행위성을 포괄적으로 고려하기 위해 사용된다.

미셸 칼롱(Callon, 1986)은 과학기술을 설명함에 있어 사회과학이 겪는 주요 난점들을 확장된 불가지론, 일반화된 대칭성, 자유로운 연합의 방법론적 원칙을 준수함으로써 극복할 수 있다고 주장하였다. 그에 따르면 행위자가 네트워크를 건설하는 번역(translation)의 과정은 크게 네 가지 단계를 거치며 진행된다. 문제제기(problematization) 단계에서 한 행위자는 일군의 다른 행위자들과 그들이 '원하는 것'을 정의한다. 그리고 의무통과점(obligatory passage point)를 제시하여 다른 행위자들의 목적 달성에 있어 자신을 필수 불가결한 것으로 만든다. 이렇게 모인 행위자들은 언제든지 각자의 정체성, 목적, 이해관계 등을 다른 방식으로 정의할 수 있기 때문에, 관심끌기(interessement) 단계에서 한 행위자는 다른 행위자의 정체성을 강제하고 안정화시키려한다. 이 과정에서 한 행위자는 다른 행위자들에게 다른 정체성을 부여하려 하는 다른 모든 실체들과의 연결을 자르거나 약화시킨다. 이 후 등록하기(enrollment) 단계에서는 각 행위자의 역할이 정의되고, 역할은 이를 수락하는 행위자의 속성이 된다. 그리고 마지막 동원하기(mobilization) 단계에서 모든 행위자들 또는 그들의 매개물, 등가물, 대변인 들이 다시 한 행위자에 의해 대변되며 강제적인 관계의 연결망이 형성된다. 모든 행위자는 이와 같이 다른 행위자에게 정체성, 의도, 역할을 부여하여 자신을 중심으로 한 연결망을 건설하려고 한다.

행위자의 의도나 목적이 기술에 내재되어 연결망이 확장되기도 한다. 기

술은 사람과 사물 간의 관계뿐 만 아니라 인간관계를 용이하게 하거나 제약한다. 설계자는 잠재적인 사용자를 재현하여 제품으로 물질화(materialize)시키며, 이에 따라 기술은 사용자의 예상된 정체성, 목적을 담은 스크립트(script)를 포함하게 된다. 스크립트는 사용자와 기술에 특정한 능력, 행위, 책임을 위임하여 책임의 지형(geographies of responsibilities)를 창조, 변형, 강화한다(Akrich, 1992). 하지만 모든 사용자가 기술의 스크립트를 수동적으로 따르는 것은 아니며, 사용자는 기입(subscription), 탈기입(de-inscription), 반계획(antiprogram)을 통해 기술과의 관계를 능동적으로 설정한다. 여기서 반계획이란 설계자의 계획과 상충되는 사용자의 행위계획(또는 사용자 계획과 상충되는 설계자 계획)을 말한다. 기입과 그 반대인 탈기입은 행위자에게 규정된 것 혹은 금지된 것에 대한 행위자의 동의나 거부, 그리고 기술에 의해 주어진 규정을 재협상하는 범위를 언급하기 위해 사용된다. (Akrich & Latour, 1992)

행위자-연결망 이론적 관점은 이종적(heterogeneous)인 동맹(alliance), 집합(assemblage), 조합(association)들에 대한 분석을 통해 사회학의 연구범위를 확장시키는 동시에, 기존 접근법에 대한 변화를 요구한다. 브루노 라투어(Bruno Latour, 2007)는 사회학자들이 사회적인 것(the social)을 설명하기 위해 다시 사회적인 것을 동원하는 모순에 빠졌다고 비판하고 이를 극복하기 위해서 집단의 형성, 행위성, 사물의 행위성, 사실 등에 등이 구체적으로 고려되어야 한다고 주장한다.

(2) 기술체계이론

토마스 휴즈(Thomas Hughes)은 19세기 말에서 20세기 초에 걸친 미국 전력 시스템의 역사에 대한 분석을 통해 기술체계(technological system) 이론을 발전시켰다(Hughes, 1983). 기술체계는 인공물, 조직, 과학적 요소들(책이나 논문, 대학 강의나 연구 계획), 법적인 장치, 자연자원, 과 같이 수많은 인간, 비인간으로 구성된다. 기술체계의 요소들은 서로 상호작용하고 하나의 구성 요소의 변화가 다른 요소들에 영향을 미치기 때문에, 각각

의 특성은 체계 전체 속에서만 제대로 파악될 수 있다. 기술체계 요소 중 특히 체계건설자(system builder)는 전기설비와 같은 인공물에서 지주회사 등과 같은 조직에 이르기 까지 기술체계 내의 다양한 요소들을 발명, 개발하여 기술체계를 만든다.

기술체계의 요소들은 서로 상호작용을 주고받는 반면, 기술체계와 환경(enrivironment) 간에는 일방적인 영향만이 있을 뿐이다. 따라서 기술체계의 환경에는 기술체계에 영향을 주지만 체계로부터 영향을 받지는 않는 것과 기술체계로부터 영향을 받지만 기술체계에 영향을 주지는 않는 것의 두 종류가 존재한다. 예를 들어 화석 연료 공급은 전기 시스템에 영향을 주는 환경적 요소인 반면, 전기기구 생산업자가 소유한 하청업체의 경우 기술체계로부터 영향을 받기만 하는 환경적 요소가 된다. 기술체계는 시간이 지남에 따라 예측이 어려운 환경적 요인들은 기술체계에 통합하려는 경향을 보인다. 사회적인 것은 기술체계와 환경 모두의 구성 요소로 작용할 수 있기 때문에, 기술의 '사회적 맥락' 또는 '사회적 배경'과 같은 개념은 피해야 한다(Hughes, 1987:52).

기술체계는 상품과 서비스의 생산성을 높이기 위한 목적으로 유용하거나 바람직하다고 생각되는 방식으로 물질세계를 재조직화한다. 기술체계는 적절한 모든 수단을 동원하여 문제를 해결하거나 목표를 달성하며, 기술체계 내의 모든 요소들이 이러한 목표에 직간접적으로 기여한다. 하나의 체계는 체계와 그 하위체계(subsystem)의 반복되는 유형 속에 위치하므로, 기술체계를 분석함에 있어 그 분석의 수준을 정하는 것이 중요한다. 휴즈는 공학자들이 분석하는 기술체계는 기술적(technical)인 요소 외의 것들은 정치적인 것으로 치부하는 경향이 있는 반면, 일부 사회과학자들은 분석의 수준을 너무 높게 잡아 정작 기술적인 것의 내용은 거의 문제 삼지 않는다고 비판한다(위의 책:55).

기술체계의 진화, 확장 과정은 발명(invention), 개발(development), 혁신(innovation), 기술이전(technology transfer), 성장 및 경쟁과 공고화(growth, competition, and consolidation) 단계로 나타낼 수 있다. 지주회사, 발전소, 백열전구 등은 누군가에 의해 처음 발명되어 성공적인 개발단

계를 거침으로써 기술체계로 발전하며, 혁신 단계에서 생산, 판매, 서비스 부분 등의 다양한 구성요소와 결합되어 더욱 복잡한 체계로 확장된다. 기술체계는 다른 시공간으로 이전되어 적응의 과정을 거치며, 기술체계는 성장하며 경쟁하거나 합병, 표준화를 통해 다른 체계와 상호연결 되거나 통합되기도 한다. 기술체계 진화의 각 단계에서 중요 결정을 내리는 체계건설자의 유형에도 차이가 있을 수 있다. 발명과 개발의 단계에서는 발명가-기업가(inventor-entrepreneur), 혁신, 경쟁, 성장 단계에서는 관리자-기업가(manager-entrepreneur), 공고화 단계에서는 자본가-기업가(financier-entrepreneur)가 결정적인 문제들을 풀어낸다(위의 책:57).

기술체계이론은 행위자-연결망 이론과 유사한 시기에 발전되었으며, 이 두 접근법 모두 사회와 기술의 이분법을 비판하며 인간과 비인간의 상호작용을 강조하였다는 점에서 공통점을 가지고 있다. 기술체계 이론의 초기에 모멘텀(momentum)과 같은 개념을 통해 기술의 자율성을 강조되긴 하였지만, 이 후에는 기술체계를 기술과 사회의 긴밀한 상호작용으로 "빈틈없는 그물망(seamless web)"으로 재확인하였다는 점에서 두 접근법은 더욱 공통점을 찾아가고 있다. (Hughes, 1986; Latour, 2007) 기술체계이론은 이종적인 연결망을 형성함에 있어 체계건설자와 같은 주도적인 행위자의 역할에 더 많이 주목하는데, 이는 행위자-연결망 이론에서 혼종적 연결망을 건설하는 이종적 공학자(heterogenous engineer)의 개념과도 연관성이 있다(Law, 1987).

행위자-연결망 이론은 비인간 및 행위의 분석을 위한 철학적 기반을 마련해 주고 있다고 한다면 기술체계이론은 기술진화 단계 구분, 체계와 환경의 구분 등 기술과 관련된 영역에서 유용하게 사용될 수 있는 개념적 도구들을 제시하고 있다(Hanseth, Margunn & Berg, 2004). 이때 행위자의 개념은 개체 자체의 행위성을 강조하는 반면 체계의 '요소'는 체계를 구성하며 체계 내부의 다른 개체에 영향을 미치는 측면을 강조한다.

집단 혹은 체계 수준. 각 집단은 고정된 것이 아니며 항상 새로이 정의되고 변화한다. 체계를 구성하는 요소나 범위 또한 고정되어 있지 않다. 연구자가 집단 혹은 체계 자체를 어떻게 정의하느냐가 집단 혹은 체계의 존

재를 규정한다.

(3) 행위자-연결망 이론과 기술체계 이론에서의 권력

행위자-연결망 이론에 의하면 '권력'은 그 자체로 설명이 필요한 대상이다. 권력은 지위의 변동과 같은 현상으로 인한 '효과'이며, 사회과학적 연구 절차에서는 현상의 원인으로 가정되는 것이 아닌 가장 마지막에 도출되는 결과물이어야 한다. 권력과 지배는 생산되고, 구성되고 조합되어야 한다(위의 책:64). 반면 기존 사회학은 권력 자체를 설명하려고 하지 않고 이를 일종의 '미지의 담지체(mysterious container)'로 가정해 둔 채 여러 가지 현상과 사례들을 권력의 개념으로 설명하려 한다(위의 책:83). 그리고 이러한 '사회적인 것의 사회학자들(sociologists of the social)'에게는 권력의 개념을 동원하는 것 자체가 자동적인 설명을 제공하는 것으로 여겨진다. 그렇기 때문에 권력은 '분석가뿐만 아니라 행위자 또한 잠들게 한다'(위의 책:85). 이러한 경향은 단지 권력의 개념에만 적용되는 것이 아니라 기존의 사회학자들이 '사회적인 것(the social)'이라고 규정한 모든 것(사회, 권력, 구조, 맥락 등)들 모두에 해당된다.

권력의 형성과 유지를 밝힘에 있어서 사물, 기술과 같은 비인간은 중요한 역할을 한다. 행위자-연결망 이론에 있어 현상의 변화를 가져오는 모든 것은 행위성(agency)를 가지고 있는 것이며, 따라서 비인간도 행위성을 가질 수 있다. 비인간 행위자(non-human actor)들은 단지 인간 행위자의 보조적인 역할을 하는 것이 아니며, 지배와 권력에 있어 중심적인 역할을 차지한다(위의 책:83). 어떠한 비인간 행위자도 없는, 오직 면대면 의사소통과 사교기술에 기반한 인간 행위자 간의 권력 관계는, 현대 사회에서 관찰하기도 어려울 뿐더러 발견된다고 해도 매우 짧은 시간 동안만 지속되는 일시적인 상호작용에 그칠 뿐이다. 지속되는 행위는 인간-인간, 비인간-비인간 간 상호작용만으로는 형성되기 어려우며, 인간 행위자와 비인간 행위자 간의 상호작용의 연쇄를 통해 일어난다(위의 책:75).

기술체계이론에서도 권력과 지배의 개념을 확인할 수 있다. 기술체계 이

론에 따른 체계건설자는 체계를 만드는 과정에서 다양성, 다원주의, 혼란을 제어하고 단일성, 중앙집중, 일관성을 강제하며, 이 과정은 종종 다른 체계의 파괴를 수반한다(Hughes, 1987). 즉 체계건설자는 체계에 속한 인간 행위자를 통제하고 사물과 기술이 만들어지고 조합되는 방식을 조율함으로써 체계의 건설과 유지를 도모하는 것이다. 기술체계를 정의하고 체계 내의 요소들을 연결하는 역할로 인해 체계건설자는 기술체계 내에서의 우월한 지위를 확보하며, 우월적 지위는 다른 행위자에 대한 제어, 통제와 같은 권력 행사를 통해 드러난다.

위의 두 이론적 설명을 종합해 보면, 기술체계 내에서의 권력 관계는 기술체계 내 인간, 비인간 행위자의 상호작용 방식을 확인하고, 이를 통해 이들의 역할과 상대적인 지위를 분석함으로써 설명될 수 있다.[3] 그리고 권력의 확장은 중심 행위자 혹은 체게건설자가 기술체계에 더 많은 행위자를 포섭하고 이들보다 우월한 지위를 획득하여, 생산, 유통, 사용 과정에서 행위자들을 자신의 목적에 부합하는 방식으로 통제할 수 있게 되는 것이라고 할 수 있다.

3) 권력 관계에서 비인간 행위자의 권력 관계 개입은 일반적이지만, 본 논문에서는 비인간 행위자의 역할이 고려되지 않았던 기존 사회학적 권력 개념과의 연속성을 위해 필요에 따라 '기술권력'이라는 용어를 사용하여 비인간 행위자의 역할과 기술체계의 특성을 부각시키고자 한다.

4. 연구 설계

(1) 연구 대상

본 논문의 연구 대상은 스마트폰 운영체제인 안드로이드 운영체제 및 연관 행위자들로 이루어진 기술체계이다. 본 논문에서는 이 기술체계를 '안드로이드 체계'로 지칭한다.[4] 안드로이드 체계는 대략적으로 <표 1> 같은 요소들을 포함할 수 있다.[5] 각 요소들은 각각 하나의 하위체계로서 그 내부에 또 다른 기술적 요소들을 포함하며, 요소의 일부를 다른 하위체계와 공유할 수 있다.

구글
안드로이드 운영체제
안드로이드 운영체제 개발 참여자 집단
안드로이드 운영체제 개발 도구
안드로이드 운영체제 코드 관리 시스템
안드로이드 운영체제 코드 기여자
안드로이드 운영체제 코드 검증 조직
안드로이드 운영체제를 변형하는 사용자
안드로이드 운영체제 정합성 검증 도구
안드로이드 어플리케이션
안드로이드 어플리케이션 시장 운영자
안드로이드 어플리케이션 시장 기술
안드로이드 어플리케이션 개발자
안드로이드 어플리케이션 개발 도구
안드로이드 어플리케이션에 사용되는 외부코드 및 그 개발자
안드로이드 탑재 기기 제조사
안드로이드 탑재 기기
구글 및 안드로이드와 관련한 조약, 협정으로 맺어진 기업, 단체의 조합
안드로이드 기기 및 어플리케이션 사용자

<표 1 > 안드로이드 체계의 주요 구성요소

4) 본 논문에서는 기술체계의 한 요소로서 운영체제를 지칭하기 위해서 '안드로이드' 혹은

안드로이드 체제를 분석하기 위해서는, 우선 안드로이드 운영체제 기술의 특성을 살펴볼 필요가 있다.

1) 정보기술의 특성

```
package com.example.android;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class HelloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        TextView tv = new TextView(this);
        tv.setText("Hello world");
        setContentView(tv);
    }
}
```

<그림 1> 기술적 행위의 혼종성: “Hello world”를 스마트폰의 화면에 표시하는 어플리케이션의 코드 중 일부

정보기술의 개발자가 반드시 정보기술을 구성하는 모든 코드를 직접 작성하는 것은 아니다. 스마트폰, 개인용 컴퓨터 등과 같은 정보기술에 필요한 여러 기능을 모두 직접 만드는 것은 개발자에게 엄청난 시간과 노력을 요구한다. 단순한 기술을 개발하는 경우에도 개발자가 처음부터 끝까지 모든 코드를 직접 작성하는 경우는 드물다. <그림 1>은 스마트폰 화면에 “Hello world”라는 문구를 표시하는 매우 기초적인 작업을 수행하는 프로그램 코드의 일부이다. 이 중 ‘import’는 미리 만들어진 코드를 포함시키기 위한 명령어이며, 이 중 ‘setContentView’, ‘onCreate’ 등의 이름을 가진 코드가 사용되고 있다.[6] 이와 같이 간단한 작업을 수행하는 코드에도 미리

‘안드로이드 운영체제’라는 용어를 사용하며, 전체 기술체계를 지칭할 때는 ‘안드로이드 체계’이라는 용어를 사용할 것이다.

5) 기술체계와 외부의 환경의 경계는 불확실할 뿐만 아니라 지속적으로 변하기 때문에, 시간이 지남에 따라 기술체계에 새로운 요소가 포함되거나 한 시점에서 체계의 요소였던 것이 다른 시점에서는 환경적 요인, 혹은 체계와 무관한 요인 간주될 수 있으며, 이에 따라 요소의 목록 또한 변화할 수 있다. 목록의 유동성은 방법론적인 불안정성을 나타내는 것이 아니며, 오히려 기술체계의 동적 특성 또는 행위자들의 번역에 따라 확장, 변화, 소멸하는 연결망의 특성을 반영하는 것이다.

만들어진 코드가 사용되며, 이러한 코드를 사용하지 않을 경우 개발자가 작성해야 할 코드의 양이 대폭 증가하거나 기술의 제작이 현실적으로 불가능할 수도 있다.

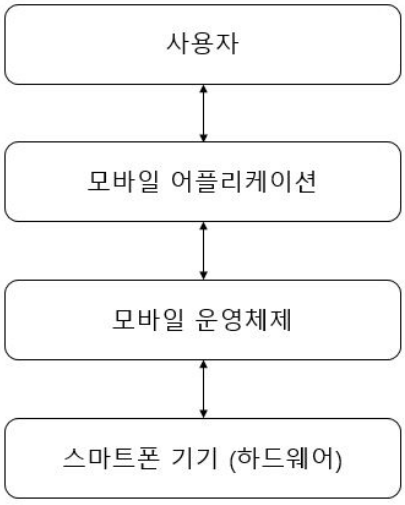
이러한 '미리 만들어진 코드'는 다른 개발자나 기업들이 제작, 공개, 배포한다. 이미 만들어 놓아 그대로 사용할 수 있는 코드를 활용하여 개발자는 코드 작성에 소요되는 시간을 절약하고 코드의 기능을 다양하게 확장할 수 있다. 뿐만 아니라 미리 만들어진 코드는 외부의 전문가들이 특정 기능에만 집중하여 제작한 것이기 때문에 개인이 자체 제작한 코드에 비해 일반적으로 좋은 성능을 보인다. 이러한 코드들을 컴퓨터 공학에서는 라이브러리(library)라고 부르며, 라이브러리의 '출입구'라고 할 수 있는 API(Application Programing Interface)를 코드에 포함시킴으로써 쉽게 그 기능을 이용할 수 있다.

많은 소프트웨어가 미리 만들어진 코드를 포함하고 있기 때문에 소프트웨어가 사용자에게 제공하는 기능, 정보가 단일한 개체에 의해 의도된 것이라고 할 수는 없다. 소프트웨어의 개발자는 여러 개발자들이 만든 코드, 그리고 그것이 제공하는 정보, 기능을 적절하게 '정렬'하여 원하는 '조합'을 만들어내며, 소프트웨어는 이러한 '혼종적인 조합'에 의해 구동된다.

2) **스마트폰 기술의 구조**

스마트폰 기술은 <그림 2>과 같은 구조로 이루어져 있다. 사용자는 스마트폰에서 특정한 '모바일 어플리케이션(mobile application)'을 선택하여 실행시킨다. 모바일 어플리케이션은 스마트폰에서 동작하는 작은 소프트웨어로서, 통화, 게임, 동영상 시청, 인터넷 서핑 등의 세부적인 작업을 수행한다. 스마트폰을 사용한 모든 행위는 각기 다른 기능을 가진 모바일 어플리케이션을 통해 이루어진다. 사용자는 모바일 어플리케이션 마켓(mobile application market)에 접속하여 자신의 스마트폰에 원하는 모바일 어플리케이션을 선택적으로 설치할 수 있다. 모바일 어플리케이션이 스마트폰에

6) 예제 출처: http://devsw.tistory.com/44



<그림 2> 스마트폰에서 기술적 요소간의 상호작용

설치되었다면, 이 후 사용자는 언제든지 어플리케이션을 실행할 수 있게 된다.

사용자가 모바일 어플리케이션을 이용하며 입력한 정보는 '모바일 운영체제(mobile operating system)'로 전달된다. 모바일 운영체제는 모바일 어플리케이션이 보낸 정보에 따라 화면표시장치, 통신부품, 음향부품 등 스마트폰의 하드웨어를 동작시킨다.[7] 이와 같이 모바일 운영체제는 소프트웨어의 일종이지만 다른 소프트웨어가 하드웨어를 동작할 수 있도록 해주며, 여러 소프트웨어가 동시에 구동될 때 하드웨어의 자원을 관리, 분배한다. 모바일 환경에서는 특히 배터리 용량, 정보 저장용량, 정보 처리 용량에 제약이 심한 스마트폰에서 여러 개의 어플리케이션이 지속적으로 구동되기

7) 사용자가 이용하는 모바일 어플리케이션이 다양하고 스마트폰 기기마다 포함된 하드웨어 부품에 차이가 있는 상황에서, 모바일 운영체제는 모바일 어플리케이션이 스마트폰 기기의 개별적 특성에 영향을 받지 않고 동작할 수 있도록 해야 한다. 이를 위해 모바일 운영체제는 스마트폰 기기의 각 요소들을 추상화(abstraction)한다. 예를 들어 한 스마트폰에는 A 사에서 제조한 스피커가 있고, 다른 스마트폰에는 B사에서 제조한 스피커가 있더라도, 어플리케이션은 각각의 특성을 고려할 필요 없이 운영체제에 소리를 재생해달라는 명령만 내리면 되는 것이다. 이 후 운영체제는 이 명령과 정보를 각 스피커에 맞는 형태로 바꾸어 스마트폰에서 실제로 소리가 나올 수 있도록 한다.

때문에 자원과 동작의 관리가 중요한 문제로 대두되며, 이에 따라 모바일 운영체제 기능에 대한 중요성도 커졌다.

본 논문에서 '제조사'라고 함은 주로 안드로이드 스마트폰의 하드웨어와 관련된 사업을 하는 기업을 의미한다. 제조사는 여러 가지 부품을 조합하여 스마트폰 기기를 만들며, 이 기기에 맞게 안드로이드 운영체제를 변형, 수정하고 기기에 이 운영체제를 탑재하는 역할을 한다. 그리고 제조사는 주로 안드로이드 운영체제를 탑재한 스마트폰 기기를 판매하여 수익을 얻는다.

반면 '개발자'는 안드로이드 스마트폰에서 동작하는 모바일 어플리케이션을 만드는 사람들을 의미한다. 모바일 어플리케이션은 개인용 컴퓨터에서 동작하는 소프트웨어들과 비교하여 상대적으로 크기가 작고 동작이 단순하며, 인터넷을 통해 모바일 어플리케이션을 개발할 수 있는 도구와 정보들을 손쉽게 구할 수 있기 때문에, 모바일 어플리케이션의 개발은 기업뿐만 아니라 개인에 의해서도 이루어질 수 있다. 개발자가 만든 모바일 어플리케이션은 어플리케이션 마켓이나 인터넷을 통해 사용자의 스마트폰으로 이동하며, 사용자는 모바일 어플리케이션을 자신의 스마트폰에 설치한 뒤 이용한다. 모바일 어플리케이션을 만들기 위해서 개발자는 기본적인 컴퓨터 공학적 지식과 함께 안드로이드 운영체제의 구조에 대한 지식이 필요하다.

3) 안드로이드 운영체제의 특성

전 세계 스마트폰 사용자는 10억 명을 돌파한 것으로 추산되고 있다(Reisinger, 2012.10.17.). 스마트폰 사용 시간 또한 늘어나고 있어서, 한국에서는 사용자의 70% 이상이 하루 평균 2시간 이상을 스마트폰으로 보내며, 미국 사용자들도 스마트폰에 하루 평균 2시간 반 이상을 사용한다(한국인터넷진흥원, 2012; Khalaf, 2013.4.3.). 특히 한국에서는 30대 이하의 젊은 세대를 중심으로 스마트폰이 개인용 컴퓨터와 텔레비전 등의 타 매체를 대체하고 있다(정보통신정책연구원, 2012).

전 세계 대부분의 스마트폰에는 구글이 주도하여 개발하고 있는 안드로

이드 운영체제가 설치되어 있다. 뿐만 아니라 구글은 전 세계 140개 국가에 안드로이드 모바일 어플리케이션을 유통하고 있으며, 그 수는 80만 개가 넘는다. (Google, 2013, Appbrain, 2013.10.4.)[8]

안드로이드 체계는 특히 개방성을 지향한다는 점에서 특징적이다. 안드로이드 체계의 개방성은 세 가지 측면에서 관찰될 수 있다. 첫째, 일반인들도 안드로이드 운영체제의 생산 과정에 참여할 수 있다. 안드로이드 운영체제의 개발은 안드로이드 오픈소스 프로젝트(Android Open Source Project, AOSP)[9]라고 불리는 단체에서 주도하고 있다. 이 단체는 안드로이드 운영체제의 개발 과정을 인터넷 상에 공개해놓고 있으며, 누구나 공개된 정보를 참조하여 새로운 기능을 추가하거나 잘못된 점에 대한 수정을 요청할 수 있다.

둘째, 모바일 어플리케이션 개발자는 안드로이드 운영체제에서 동작하는 어플리케이션을 만들고 이를 별도의 비용 없이도 출시, 배포할 수 있다. 안드로이드 어플리케이션을 만들 때 필요한 도구들은 인터넷 상에서 다운받아 무료로 사용할 수 있다. 또한 개발자들은 기존에 사용하는 프로그램에 'tool'또는 'kit'의 명칭으로 불리는 프로그램을 추가하는 것만으로 안드로이드 어플리케이션을 개발할 수 있으므로, 오랜 시간을 들여 새로운 환경에 적응할 필요가 없다.

셋째, 제조사는 비교적 쉽게 스마트폰에 안드로이드 운영체제를 탑재할 수 있다. 제조사의 용이성은 크게 두 가지에서 설명될 수 있다. 우선 제조사는 스마트폰에 안드로이드 운영체제를 탑재하기 위해 이 운영체제를 구매하거나 별도의 라이선스 비용을 지불할 필요가 없다. 또한 제조사는 안드로이드 운영체제를 자사에서 만들려는 기기의 특성에 따라 자유롭게 변형, 수정할 수 있으며, 이렇게 변형된 운영체제에 대한 저작권 또한 제조사에 귀속된다.

---

8) 총 스마트폰 사용 시간 중 어플리케이션 사용 시간의 비율은 80%에 달한다(Khalaf, 2013.4.3.).

9) 안드로이드는 운영체제의 이름이자 이 운영체제를 개발하는 조직의 이름이기도 하다. 본 논문에서는 동일한 명칭으로 인한 혼동을 최소화하기 위해서, 운영체제를 지칭할 때는 '안드로이드 운영체제', 이 운영체제를 개발하는 조직을 지칭할 때는 'AOSP(Android Open Source Project, 안드로이드 오픈소스 프로젝트)' 라는 명칭을 사용하기로 한다.

이러한 개방성은 다양한 행위자의 참여로 이어진다. 우선 많은 수의 모바일 어플리케이션 개발자가 안드로이드 어플리케이션을 개발하려고 한다. 또한 다양한 제조사들이 안드로이드 운영체제의 자신의 기기에 탑재하려고 한다. 뿐만 아니라 다양한 사람들이 운영체제의 개발 과정을 살펴볼 수 있고, 이에 대해 여러 가지 의견을 개진할 수 있다. 그리고 다양한 행위자의 참여는 이들에 의해 생산되는 기술의 다양성으로도 이어진다. 실제로 안드로이드 어플리케이션 마켓에는 세계에서 가장 많은 수의 어플리케이션이 유통되고 있다. 또한 안드로이드 운영체제를 탑재하고 있는 스마트폰의 수는 전체 스마트폰의 80%에 달한다.

4) 구글과 안드로이드 운영체제

구글은 웹 검색과 클라우드 컴퓨팅(cloud computing)을 주 사업 영역으로 하는 미국의 정보통신기업으로 래리 페이지(Larry Page)와 세르게이 브린(Sergey Brin)이 1998년 설립하였다. 창업 초기 구글은 웹 검색엔진 개발에 주력하였으나, 이후 동영상 스트리밍 서비스인 유튜브(Youtube), 지도 서비스인 구글 맵스(Google Maps), 이메일 서비스인 지메일(Gmail), 웹 브라우저이자 컴퓨터 운영체제인 크롬(Chrome), 소셜네트워크서비스인 구글 플러스(Google+) 등으로 사업을 다각화하였다. 구글은 이러한 기술과 서비스를 제공하는 과정에서 사용자에게 광고를 노출함으로써 수익의 대부분을 올리고 있다. 구글의 검색 서비스는 세계에서 가장 많은 사람들이 사용하고 있으며, 2014년에는 구글이 세계에서 가장 높은 브랜드 가치를 가진 기업으로 선정되기도 하였다(Gordon, 2014.5.21.).

구글의 모바일 운영체제 사업은 2000년 대 중반부터 본격화되었다. 구글은 2005년 모바일 운영체제인 '안드로이드'를 개발하고 있던 동명의 회사 안드로이드(Android Inc.)를 인수하고 핵심 직원들을 구글로 영입하였다. 이 후 2008년 구글은 안드로이드 운영체제의 발표와 동시에 정보통신업계의 기업들로 구성된 개방형핸드셋연합(OHA, Open Handset Alliance)의 출범을 선언하고, 안드로이드 운영체제의 개발이 이 연합에 의해서 이루어

질것이라고 밝혔다. 아울러 안드로이드 운영체제의 개발, 사용과 안드로이드 운영체제에 기반한 어플리케이션의 개발은 안드로이드 오픈소스 프로젝트(AOSP, Android Open-Source Project)라는 이름으로 진행되고 있다. 이 프로젝트에는 누구나 참여할 수 있으며, 이를 통해 안드로이드 운영체제의 개발, 수정 과정에 의견을 내거나 안드로이드 운영체제에 기반한 스마트폰 기기나 어플리케이션을 만드는 데 필요한 도구와 정보를 얻을 수 있다.

(2) 연구 방법

기술에 대한 사회학적 분석은 수많은 비인간 행위자를 포함하게 된다. 하지만 인간-비인간 간의 결합과 상호작용은 일상적인 경우 쉽게 모습을 드러내지 않는다. 이는 비인간 행위자가 스스로를 설명할 수 없을 뿐만 아니라, 많은 경우 기술 내부의 다양한 상호작용들이 하나의 단일한 대상으로 추상화, 혹은 블랙박스화(black-boxing)되어있기 때문이다. 예를 들어 '자동차를 운전'하는 행위도 운전자, 자동차를 구성하는 기술적 요소, 그리고 이러한 기술적 요소를 설계, 조직한 행위자, 또한 자동차가 특정한 방식으로 움직이도록 하는 경찰관, 교통체계와 법률 간 복잡한 상호작용의 결과물이다. 분석의 어려움을 극복하기 위한 한 가지 방법은, 기술이 새로이 생성, 변화, 파괴되는 순간에 주목하는 것이다. 이러한 상황 속에서 관찰자는 비인간과 인간 행위자들 간의 상호작용을 더욱 명확하게 관찰할 수 있다(Latour, 2005).

하지만 기술체계의 관점에서는 기술에 대한 사회학적 연구를 용이하게 하는 지점을 추가로 발견할 수 있다. 우선 기술적 대안을 분석할 수 있다. 현재 존재하는 연합은 체계건설자와 주요 행위자들의 선택에 의한 결과물이다. 반면 특정한 상황, 조건 하에서 기각된 기술적 대안들은 목표에 부합하지 않는 것, 목표와 반대되는 결과나 바람직하지 않은 결과를 가져오리라 예상되는 것, 다른 대안에 비해 상대적으로 적은 효용성을 가진 것들로서 체계건설자의 의도와 기술체계의 구조를 분석하는데 유용하게 활용될 수 있다. 둘째, 기술체계와 다른 체계 간의 갈등 및 경쟁 과정을 분석에 활

용할 수 있다. 이는 평상시에 잘 드러나지 않는 기술체계의 내부적 상호작용 방식, 기술체계의 경계나 목적이 다른 기술체계와 경쟁, 갈등을 통해 표면화되기 때문이다. 경쟁 기술체계는 다른 기술체계의 내부적 상호작용 방식과 상호작용의 의도를 폭로하고, 기술체계에서 핵심적인 기술적 요소가 무엇인지를 확인시켜 주며, 기술체계 간의 경계가 어디인지를 보여준다. 마지막으로, 거대 기술체계의 경우 기술의 생산이나 사용에 참여하는 행위자의 수가 많기 때문에, 기술이 생산되거나 사용되는 과정 혹은 생산 혹은 사용되는 기술체들의 경향성이 분석에 활용될 수 있다. 이러한 경향성은 암묵적, 이면적 상호작용이나 오랜 시간 혹은 넓은 지역에 걸쳐 일어나는 거시적 상호작용 양상을 관찰하는데 유용하다. 따라서 본 연구는 안드로이드 체계의 분석을 위해, 정보기술의 상호작용을 더욱 자세히 담고 있는 코드의 수준에서 위에서 제시된 측면들을 확인, 분석한다.

(3) 연구 자료

본 연구에서 주요하게 사용하는 연구 자료는 다음과 같다.

자료 분류		주요 출처
안드로이드 운영체제 소스코드[10]		source.android.com
안드로이드 운영체제 구조		source.android.com
안드로이드 운영체제 코드 심사 과정		android-review.googlesource.com
안드로이드 운영체제 코드 생산 절차		source.android.com/source
안드로이드 운영체제 버전별 분포		developer.android.com
안드로이드 어플리케이션 개발 도구		developer.android.com
안드로이드 어플리케이션 개발 설명문서		developer.android.com
안드로이드 정합성 관련 문서 및 프로그램	정합성정의문서	source.android.com/compatibility/
	구글의 정합성 설명	source.android.com/compatibility/
개방핸드셋연합		openhandsetalliance.com
구글이 제공 서비스의 약관		play.google.com/intl/ko_kr/about/play-terms.html
구글 홈페이지		google.com
안드로이드 기기 관련 정보		android.com phonearena.com gsmarena.com
안드로이드 어플리케이션 관련 정보		play.google.com

	appannie.com appbrain.com
스마트폰 시장 분석 자료	nielsen.com gartner.com flurry.com appbrain.com appannie.com
언론 보도 자료	국외 언론사 일반 언론(New York Times, Wall Street Journal 등) 기술 전문 언론 (Ars Technica, ZDNet, CNET 등) 국내 언론사 일간지 기술관련 언론(전자신문, 디지털타임즈 등)

<표 2> 연구 자료

(4) 논문의 구성

본 논문의 2장, 3장, 4장에서는 각각 운영체제 개발 참여자, 개발자, 제조사를 포함하는 안드로이드 체계의 세 가지 하위체계를 분석하여 각 하위체계에서 기술적 개방과 분화, 확장이 이루어지는 방식을 확인하고, 각 하위체계의 개방성 이면에 존재하는 구글에 의한 지배 구조를 확인한다. 5장에서는 이러한 하위체계들이 결합하여 전체 안드로이드 체계를 구성하는 방식과, 하위체계 간의 결합 방식과 상호작용에 의한 각 행위자의 지위 변화, 그리고 이 결과 다른 요소들의 행위에 대한 통제, 억제가 강화되는 방식을 살펴볼 것이다. 또한 기술체계 내에서 형성된 이러한 권력이 국가의 제도적 권력과 상호작용하는 방식으로 통해, 체계를 통제하고 제어하는 체계건

10) 하나의 소프트웨어를 구성하는 코드 덩어리를 '소스코드(source code)'라고 하며, 윈도우즈(Windows), 리눅스(Linux) 등 다른 운영체제와 마찬가지로 안드로이드 운영체제도 소스코드의 덩어리라고 할 수 있다. 안드로이드 운영체제의 구조와 기능은 소스 코드가 어떻게 작성되었는지에 따라 결정되는데, 안드로이드 운영체제의 온라인 저장소를 살펴보면 소스코드가 담긴 모든 파일을 직접 확인할 수 있다.

설자의 권력이 외부의 다른 영역으로 확장될 수 있음을 확인한다. 결론인 6장에서는 본 연구의 결과를 간략히 정리하고, 사적 지배구조의 확산과 기술적 종속을 벗어나기 위한 방안에 대해 언급한다.

II. 안드로이드 운영체제 영역에서의 개방성과 지배

1. 안드로이드 운영체제 영역의 개방성

안드로이드 운영체제의 생산 구조는 개방성을 표방하고 있다. 안드로이드 운영체제의 생산 과정에는 여러 행위자가 제한 없이 참여할 수 있으며, 공식화의 작업을 통해 이러한 행위자들에 의해 기부된 여러 가지 코드가 하나의 안드로이드 운영체제로 확정된다.

(1) 안드로이드 운영체제 생산에 대한 개방적 참여와 그 방식

구글은 2007년 11월 새로운 모바일 운영체제인 안드로이드를 공개함과 동시에, 전 세계 34개 회사의 연합인 개방핸드셋연합(Open Handset Alliance, OHA)를 출범시켰다. 개방핸드셋연합은 전 세계 휴대전화 사용자 수가 30억에 육박함에도 불구하고 지금껏 변화하는 사용자 요구에 대응하기 위한 소프트웨어 개발자, 이동통신사, 기기 제조사 들 간의 협업이 부족했다고 비판하며, 사용자에게 "더 좋은 경험을 주는 기기"를 "저가에 생산, 공급"하기 위하여 "운영체제, 미들웨어, 사용자 친화적인 인터페이스와 어플리케이션이 완전히 통합된 소프트웨어"인 안드로이드 운영체제를 만들겠다고 밝혔다(OHA, 2007.11.5.). 이후 개방핸드셋연합에 가입한 기업의 수는 84개로 증가하였으며, 여기에는 모바일 기기 제조, 반도체 설계 및 제조, 소프트웨어 개발, 컨설팅 등 다양한 분야의 기업이 포함되어 있다(<표 3> 참조).

안드로이드 운영체제는 안드로이드 오픈소스 프로젝트(Android Open Source Project, AOSP)에 의해서 개발된다. OHA에 속한 기업들은 AOSP에 코드를 기부하고 오류를 보고, 수정하며, 코드 검증 절차에 관여하는 등 안드로이드 운영체제 생산 과정 전반에 참여하고 있다. 하지만 안드로이드 운영체제의 개발은 이러한 기업들 외의 다른 행위자에게도 개방되어 있다. 안드로이드 운영체제의 개발에 관심이 있는 개인이나 기업은 안드로이드

사업 분야	기업명
이동통신	창립 기업: KDDI, NTT DoCoMo, Sprint Nextel, T-Mobile, China Mobile, Telecom Italia , Telefonica 이후 가입: Bougues Telecom, China Telecommunication, China United Network Communication, LG Uplus, Softbank Mobile, Telus, Vodafone
소프트웨어	창립 기업: Ascender, eBay, Esmertec, Google, Living Image, Nuance Communications, PacketVideo, Skypop, SONiVOX 이후 가입: Andango Ingeneria, Access, Cooliris, Motoya, NXP Software, Omron Software, SVOX, VisualOn.
반도체	창립 기업: Audience, Broadcom,, Intel, Marvell, NMS Communication, Nvidia,, Qualcomm, SiRF, Synaptic, Texas Instrument, 이후 가입: AKM Semiconductor, ARM, Atheros Communication, CSR Plc., Cypress Semiconductor, Freescale Semiconductor, Gemalto, Imagination Technologies, Media Tek, MIPS Technologies, Nvidia, Renesas Electronics, ST-Ericsson, Via Telecom.
모바일 기기 제조	창립 기업: HTC, LG, Motorola, Samsung 이후 가입: Asus, Huawei, Sony Ericsson, Toshiba, Garmin, Acer, ZTE CCI, FIH, Alacatel Mobile Phones, Lenovo, Oppo, Pantech, Fujitsh, Kyocera, Nec, Sharp, Dell
상업화	창립 기업: Aplix, Noser Engineering, TAT - The Astonishing Tribe, Wind River 이후 가입: Accenture, Borqs, Intrinsyc, Sasken Communication Technologies, SQL Star International, Teleca AB, Wipro Technologies

<표 3> 개방핸드셋연합 가입기업

(개방핸드셋연합 홈페이지 자료 재구성. http://www.openhandsetalliance.com/. 2014년 6월 16일 방문)

개발 과정에 참여하거나 안드로이드 운영체제에 관해 토론할 수 있다. 이 중 개발 과정에 대한 참여는 안드로이드 운영체제 기술에 직접적인 방법이며, 개인이나 기업은 안드로이드 운영체제의 오류를 보고하거나 운영체제에 자신이 만든 코드를 포함시켜달라고 요청하는 방식으로 안드로이드 운

영체제의 개발, 수정, 변경에 참여할 수 있다.

> 코드가 왕입니다. 우리는 당신이 등록하는 어떠한 변경사항이라도 리뷰하길 바라니, 소스코드를 확인하시고, 오류나 기능을 선택하여 코딩해주세요.[11]
> [후략]
> (출처: AOSP 홈페이지, http://source.android.com/source/contributing.html)

우선 개인 및 기업은 안드로이드 운영체제에 자신이 직접 만든 코드를 반영해달라고 요청할 수 있다. 안드로이드 운영체제 기능의 수정, 개선, 추가는 각 기능에 대해당하는 하위 프로젝트(subproject)에서 진행되는데, 안드로이드 운영체제의 생산에 참여하고자 하는 사람은 자신이 만든 코드를 인터넷을 통해 AOSP에 등록할 수 있다. <그림 3>는 안드로이드 운영체제에 기부된 코드의 목록의 예를 나타내고 있다. 이 목록에는 등록된 각 코드가 의도하는 바를 나타내는 제목('subject')과 코드 작성자('owner'), 안드로이드 운영체제에서 변경, 개선, 수정하고자 하는 기능('project'), 관련된 정보가 갱신된 일자('updated') 등이 표시되어 있다. 또한 각 코드 항목에 대한 내용검토('CR', code review) 및 통합검토('V', verification)의 진행 여부와 결과도 확인할 수 있다.

AOSP는 등록된 코드에 대해 <그림 30>과 같은 절차를 진행하여 실제 안드로이드 운영체제에 반영할지 여부를 결정한다. 이 절차의 주요한 작업들은 하위 프로젝트 내에서 각 역할을 맡은 사람들에 의해 진행된다. 안드로이드 운영체제를 수정, 보완하려는 사람은 '깃(Git)'과 '레포(Repo)'와 같은 소프트웨어 버전 관리 시스템으로 부터 안드로이드 운영체제의 최신 코드를 전송받아 원하는 부분을 수정, 추가하고, 이를 적절한 하위프로젝트에 등록함으로써 기여자(contributor)가 된다. 이러한 등록은 소프트웨어 리뷰 시스템인 '게릿(Gerrit)'을 통해 이루어지며, 이 시스템은 프로젝트 관리자, 검증자, 승인자와 같은 사람들에게 새로운 코드가 등록되었음을 알린다. 승인자(approver)는 기여자가 등록한 코드에 대한 내용검토(code-review)를

11) 원문은 다음과 같다. : Code is King. We'd love to review any changes you submit, so please check out the source, pick a bug or feature, and get coding.

<그림 3> 안드로이드 운영체제에 기부된 코드 목록
(출처: https://android-review.googlesource.com, 2014년 6월 16일)

실시하여 해당 코드가 하위프로젝트에 잘 부합하는지를 판단하고 제안된 코드를 안드로이드 운영체제에 포함할지 여부를 결정하며 기여자에게 코멘트를 남기기도 한다.[12] 이후 검토자(verifier)는 기여자의 코드를 실제 운영체제에 통합(merge) 시켜 정상적으로 동작하는 지를 테스트한다. 테스트를 성공적으로 통과한 코드는 게릿 시스템에 의해 안드로이드 운영체제에 자동적으로 통합되며, 이 과정에서도 시스템이 기존 코드와의 충돌 여부를 검사한다. 이와 같이 기여자의 코드는 승인자의 코드 확인, 검증자의 코드 확인과 동작 검증, 리뷰 시스템의 코드 통합 검증을 모두 통과해야 실제 운영체제에 반영될 수 있으며, 이러한 과정을 통과하지 못한 기여자는 지적된 부분을 수정하여 다시 코드를 등록하거나, 코드 등록을 포기하여야 한다.

개인이나 기업은 자신이 발견한 안드로이드 운영체제의 오류를 보고하는 방식으로 생산 과정에 참여할 수도 있다. 안드로이드 기기에서 오류를 발견한 사용자, 안드로이드 어플리케이션을 만들기 위해 사용되는 '개발자 도

12) 하위프로젝트에 대한 '부합'여부를 판단하는 기준은 명시적으로 제시되어 있지 않다.

구'에서 오류를 발견한 개발자, 안드로이드 운영체제 코드에서 문제점을 직접 발견한 사람은 문제점을 AOSP에 보고할 수 있다. 하지만 보고된 모든 오류들이 수정되는 것은 아니다. 보고된 오류는 우선 AOSP 관리자(AOSP maintainer)에 의해 검토되며, 검토 결과에 따라 신규(new), 논의 중(open), 거절(no-action), 해결됨(resolved)의 4 가지 범주로 분류된다. 이중 신규 보고는 아직 관리자에 의해 검토되지 않았거나 오류 보고에 조치를 취할만한 충분한 정보가 기재되어 있지 않은 경우이다. 그리고 논의 중인 보고는 조치가 필요한 것으로 판단되었으나 아직 해결되지 않은 경우이며, 거절된 보고는 어떠한 조취도 필요하지 않은 것으로 판단되는 경우이다.[13] 이 중 '거절'로 분류된 보고는 무시되고 '해결됨'으로 분류된 오류가 차후에 공개되는 안드로이드 운영체제에 반영된다.[14]

(2) 기술의 공식화와 참여의 심화

기술의 모든 세부적 사항을 미리 지정된 개인이나 기업이 전담하여 생산하는 경우와 비교했을 때, 개방적 참여 방식은 더 많은 수의 행위자를 생산에 관여시키게 된다. 그리고 기술체계의 확장에 따라서 개방적 참여는 특정 행위자가 미칠 수 있는 영향력의 범위를 확장시키는 정치적 장치의 역할을 하여 행위자의 참여를 촉진시킨다.

안드로이드 운영체제는 개방성과 함께 상업성을 표방하고 있다. 이를 위해서 안드로이드 운영체제는 단순한 코드의 조합을 넘어서 하나의 '상품'이 되어야 한다. 공식화는 기술적 오류와 같은 문제점들을 제거하고 상품으로서 가치를 높일 수 있는 기능을 추가하여, 안드로이드 운영체제를 새로운 단일 상품, 새로운 '버전(version)'으로 정의하는 과정이다. <표 28>에서 확인할 수 있듯 안드로이드 운영체제의 첫 출시부터 수십 차례에 걸친 공식

13) 오류 보고의 거절 여부를 판단하는 요인은 크게 9가지이며 다음과 같다.: 스팸(Spam), 중복(Duplicate), 재발생 불가(Unreproducible), 이미 해결(Obsolete), 오류 아님(WorkingAsIntended), 계획 없음(Declined), 정보 부족(NotEnoughInformation), 사용자 오류(UserError), 외부 오류(WrongForum), 질문(Question)

14) 오류 보고에 대한 처리 과정은 다음의 웹페이지에 기술되어 있다.: http://source.android.com/source/life-of-a-bug.html

화가 이루어졌다. 개인 및 기업은 공식화의 결과물인 공식 안드로이드 운영체제에 기반하여 스마트폰 기기를 개발하거나 개인화된 운영체제를 만든다. 안드로이드 운영체제가 발달할수록, 개인과 기업은 안드로이드 운영체제 개발을 주관하는 AOSP에 의해 공식화된 운영체제를 선호하게 된다. 만약 공식화된 안드로이드 운영체제를 사용하지 않으려면, 직접 안드로이드 운영체제를 이루는 각 코드들을 수집하고 이들 간의 상호호환성을 검증하는 복잡하고 번거로운 작업을 감수해야 때문이다.

기여된 코드도 개방적 참여 과정과 공식화 과정을 거쳐 공식 안드로이드 운영체제의 일부가 된다. 개인과 기업이 자신의 목적을 달성하기 위해 제안한 기술이 공식화된다면, 안드로이드 운영체제를 통해 행위자의 영향력이 확대될 수 있다. 반면 다른 행위자의 기술이 채택되어 안드로이드 운영체제가 특정 행위자의 이익에 반하는 방향으로 변형된다면, 이 행위자는 의도한 목적을 달성할 수 없거나 경쟁자의 성공으로 인해 큰 피해를 입을 수도 있다. 스마트폰 기기 제조사뿐만 아니라 반도체 회사, 이동통신 회사, 소프트웨어 회사, 컨설팅 회사 등 다양한 업종의 기업들이 안드로이드 운영체제의 생산 과정에 참여하고, 이들의 수가 점차 증가하고 있는 이유를 여기에서 찾을 수 있다.

2. 안정성의 지배: 구글에 의한 운영체제의 규정

구글은 개방성을 통해 되도록 많은 행위자를 안드로이드 운영체제에 연계하여 '생태계'를 조성하려하고 있다. 하지만 이 과정에서 구글은 안드로이드 운영체제를 자신의 상품이나 지적재산으로 규정하고, 소유권, 저작권과 같은 수단을 동원하여 행위자들에게 영향을 미칠 수 없게 되었다.

하지만 이러한 '전통적 수단'의 부재에도 불구하고 구글은 안드로이드 운영체제 생산 과정에서 가장 강한 영향력을 유지하고 있다. 구글의 이러한 영향력은 개방적 생산 구조 속에서 기술의 안정성을 유지하는 과정에서 발현된다. 안정성이 유지되기 위해서는 기술체계의 다양성, 이질성이 통제되어야 하며, 이를 위해서는 인간, 비인간 행위자의 행위 또한 통제, 강제 되어야 하며, 구글은 이러한 안정성을 유지 과정에서 주요한 역할을 차지하고 있기 때문이다. 구글은 안정성에 대한 지배를 통해 안드로이드 운영체제 생산에 연관된 행위자들을 서열화 하고, 경쟁사와 배타적인 기술체계를 유지할 수 있다.

(1) 코드통합과정에서 구글의 지위

누구나 안드로이드 운영체제에 자신의 코드를 기여할 수 있으나, 여러 가지 코드들을 실제 안드로이드 운영체제에 반영하는 통합과정은 구글의 지배하에 있다. 통합과정에 대한 구글의 지배는 크게 세 측면으로 나누어 살펴볼 수 있다.

1) 편중된 기여

<표 4>와 같이 안드로이드 운영체제에 기부된 코드들의 수집 주체는 크게 구글, 구글을 제외한 개방핸드셋연합 가입 기업, 개방핸드셋연합에 소속되지 단체 및 개인, 안드로이드 오픈소스 프로젝트 소속 개발자로 나눌 수 있다. 이 중 개방핸드셋연합에 소속되지 않은 단체 및 개인에는 개인 개발

기여자 분류	기여자의 이메일 주소에 포함된 도메인 명
구글	google.com
AOSP	android.com
개방핸드셋연합 소속사 (구글 제외, 22개)	intel.com, arm.com, sonymobile.com, freescale.com, lge.com, nvidia.com, motorola.com, marvell.com, nxp.com, mediatek.com, broadcom.com, zte.com.cn, huawei.com, ti.com[15], samsung.com, sharp.co.jp, sta.samsung.com, asus.com, ct.jp.nec.com, htc.com, nttdonomo.co.jp, stericsson.com, imgtec.com[16]
OHA 외부 단체 및 개인 (32개)	(온라인 개발자 커뮤니티) acm.org, android-x86.org, codeaurora.org, linaro.org (개인 도메인 및 웹사이트 개인 계정) antonc.com, codefi.re, einfach.org, greyltc.com, gmail.com, googlemail.com, gtempaccount.com martin.st, c-lis.co.jp (정보통신 기기 관련 개발자 및 업체) akting.eu, byd.com, chemlab.org, elyzion.net, glassechidna.com.au, justremotephone,com, igoweb.org, levelupstudio.com, mozilla.com, multicorewareinc.com, sleepydragon.org, tieto.com, qisda.com, orange.com, rojecti.fi radiusnetworks.com, realsil.com.cn, (미국 정부 기관) tycho.nsa.gov

<표 4> 안드로이드 운영체제 코드 기여자의 소속

자, 일반 기업, 온라인 커뮤니티 등이 포함되어 있다.[17]

15) 하드웨어 업체인 텍사스 인스트루먼트 사(Texas Instrument)의 인터넷 도메인

16) 이메지네이션 테크놀로지(Imagination Technology) 사의 인터넷 도메인. 이 회사는 개방핸드셋 연합에 소속되지는 않았으나, 안드로이드 운영체제에 기반한 웨어러블 기기 개발과 관련하여 구글과 협력 협약을 체결하여서 이 그룹으로 분류되었다(Imagination Technologies, 2014.4.1.).

17) 기여자의 코드 등록 건수(코드 검증 시스템에 기여자가 등록한 사안의 수) 는 각 행위자의 비중을 상대적으로 비교할 수 있는 척도가 될 수 있다. 이에 대한 분석을 위해 안드로이드 코드 리뷰 홈페이지(https://android-review.googlesource.com)에 등록된 코드 중 2014년 6월 21일부터 2014년 6월 30일까지 10일 간 마지막 업데이트가 이루어진 코드 항목을 수집하였다. (즉 기여자, 승인자, 검토자가 이 기간에 마지막 활동을 한 것이

구글은 이 모든 기여자 중 가장 많은 건수의 코드를 등록하였다. 본 조사의 대상인 전체 949 건의 코드 중 41.4%에 해당하는 390건의 코드가 구글에 의해서 등록된 것이며, 이는 개방핸드셋연합 소속사들이 기여 비중인 30.1%와 외부 단체 및 개인의 기여 비중인 21.7%와 비교했을 때도 월등히 높은 수치이다. 더욱이 개방핸드셋연합 소속인 22개의 회사가 기여한 코드 등록 건수나 32개 소속의 외부 단체와 개인이 기여한 코드 등록 건수보다 단일 기업이 구글이 등록한 코드 등록 건수가 많다는 점에서, 코드 기여에 있어서 구글의 비중을 확인할 수 있다.[18] 더군다나 개방핸드셋연합 소속사에 대한 구글의 영향력을 고려하여 구글을 포함한 개방핸드셋연합 소속사 전체의 코드 기여 비중을 산출하면, 이는 총 등록 코드의 71.2%에 달하게 된다.

이러한 사실은 기술 생산에 대한 개방적 참여가 반드시 참여자의 다양성 또는 참여자의 균등한 참여를 통한 생산으로 귀결되지 않는다는 것을 확인시킨다. 이 생산 방식은 단지 참여의 '가능성'만을 열어두며 어떠한 행위자

다. 모든 데이터는 2014년 7월 15일 마지막으로 확인되었다.) 수집된 등록 코드 항목은 총 949건이다. 안드로이드 코드 검토 홈페이지에는 코드의 기부자, 등록된 각 코드에 할당된 승인자 및 검토자의 이름과 이메일 주소가 표시되어 있으나, 이들이 속한 단체명은 명시적으로 표시되어 있지 않다. 하지만 이메일 주소는 코드 검토 홈페이지에서 각 행위자의 소속을 확인하는 중요한 단서로 사용될 수 있으며, 그 이유는 두 가지이다. 우선 google.com, lge.co,kr, intel.com 등 기업의 인터넷 도메인(domain)으로 끝나는 이메일은 구글, LG전자, 인텔과 같은 기업에 소속된 개발자에게 할당된 것일 가능성이 매우 높다. 둘째, 기술 개발 과정에서 많은 사람 간의 협업이 요구되는 경우, 개발자들은 서로의 이메일 주소로부터 그 사람의 소속 단체를 확인하고, 이로부터 전문 기술 분야(그래픽 장치, 중앙연산장치, 이동통신 등) 등 의사소통에 필요한 추가적인 정보를 확인한다. 반면 개발자가 자기가 소속되지 않은 기업의 메일 계정을 사용하거나, 기업에 소속된 개발자가 지메일(gmail.com)과 같은 개인 이메일 계정을 사용하는 경우는 코드 검토 홈페이지에서 개발자 간의 상호작용 방식을 보았을 때 매우 드물다.

18) 지식과 기술의 편중은 안드로이드 운영체제의 개발 초기부터 지속된 것이다. 앤디 루빈은 2003년 안드로이드(Android Inc.)라는 회사를 공동 창업하고 동명의 운영체제를 개발하기 시작했다. 이 회사는 원래 사용자의 위치와 선호를 분석하는 디지털 카메라 운영체제를 개발하려 하였으나, 2004년 핵심적인 코드들은 그대로 유지한 채 스마트폰 운영체제를 개발하는 것으로 목표를 바꾸었다(PC World, 2013.4.16.). 구글은 2005년 이 회사를 인수하고 앤디 루빈을 안드로이드를 총괄하는 수석부사장으로 임명하는 등 핵심 직원들을 영입하였다. 안드로이드 사 인수에 따라 개방핸드셋연합이 출범 전까지 안드로이드 운영체제는 구글의 소유였으며 이를 개발해왔던 핵심 인사들도 구글에 소속되어 있었다. 즉 개방핸드셋연합에 의한 안드로이드 운영체제의 개발은 어떠한 지식과 자원도 존재하지 않는 상태에서 시작된 것이 아니었으며, 애초에 개방핸드셋연합 내부에서도 안드로이드 운영체제와 관련된 지식과 기술은 구글에 편중되어 있었다.

들이 어떠한 비중으로 참여할 수 있는지에 대해서는 규정하지 않는다. 그리고 합의된 기준이 부재한 개방성과 기술적 지식 및 자원의 비대칭 하에서는, 특정한 목적을 가진 행위자가 압도적인 양의 기술을 기여하고, 이를 통해 특정 행위자가 기술에 대한 타 행위자의 영향력을 상대적으로 축소시키고 자신의 목적에 부합하는 기술을 생산하는 폐쇄적 체계를 건설하는 것이 가능해 진다.

2) **편향적 검증**

코드의 기여는 안드로이드 운영체제 생산 과정의 단지 일부분에 불과하다. 기여자에 의해 등록된 코드가 안드로이드 운영체제에 반영되기 위해서는 이 코드가 승인자, 검증자에 의한 검증 과정을 통과하여야 한다. 구글이 등록한 코드들은 검증 과정에서 가장 높은 통과율을 보이고 있다. <표 5>은 각 기여자 집단별 통과, 거부, 미결 비율을 보여주고 있는데, 구글이 등록한 전체 코드의 74.9%가 검증을 통과하여 운영체제에 반영된 것을 확인할 수 있다. 이러한 높은 통과율은 개방핸드셋연합 소속사의 통과율 10.6%과 외부 단체 및 개인의 통과율 14.1%과는 대조적이며 전체 코드의 통과율인 42.1% 보다도 월등히 높은 수치이다.

구글이 등록한 코드 중 3.3% 만이 검증이 종료되지 않은 상태로 남아있는 것에 반해 개방핸드셋연합 소속사가 등록한 코드의 84.9%와 외부 단체 및 개인이 등록한 코드의 75.7%가 미결상태로 남아있다. 이는 구글 외의 다른 기여자에 대한 검증이 편향적으로 이루어지고 있음을 나타낸다. 대부분 검증 자체가 이루어지지 않는다는 점에서, 이러한 편향성은 검증자, 승인자 개개인의 편향적 기술 판단에 의한 것이기 보다는, 다른 기여자들의 코드 검증을 의도적으로 지연, 무시하는 '편향적 제도'에 의한 것이라고 볼 수 있다. 이러한 제도는 암묵적이며 비능동적인 차별과 배제 수단으로서, 이를 통해 구글은 기술 생산에 대한 지배를 강화시키면서도 여러 행위자의 코드를 명시적으로 거부할 때 발생할 수 있는 직접적인 비난도 피할 수 있게 된다.

	등록 건수 (총 등록 건수 대비 비율)	**통과 건수 (통과비율)**	**탈락 건수 (탈락비율)**	**미결 건수 (미결비율)**
구글	390 (41.1%)	**292 (74.9%)**	39 (10.0%)	59 (15.1%)
안드로이드 소속 개발자	61 (6.4%)	**48 (78.7%)**	11 (18.0%)	2 (3.3%)
OHA 소속사 (구글 제외)	292 (30.1%)	31 (10.6%)	13 (4.5%)	**248 (84.9%)**
OHA 외부 개인 및 단체	206 (21.7%)	29 (14.1%)	21 (10.2%)	**156 (75.7%)**
기여자 전체	949 (100%)	400 (42.1%)	75 (7.9%)	474 (49.9%)

<표 5> 코드 기여 집단별 코드 등록 건수 및 통과·거부율

3) 역할 장악

안드로이드 운영체제에 기여된 코드를 검토하는 승인자, 검증자는 대부분 구글의 개발자이다. 본 분석에서 수집된 949건의 코드에는 총 1316명의 검증자, 승인자가 배정되었다.[19] <표 6>은 등록된 코드에 배정된 승인자, 검증자들의 소속을 나타내고 있는데, 구글 소속의 승인자, 검증자가 전체의 67.4%에 달하는 것을 확인할 수 있다. 뿐만 아니라 전체 등록 코드의 67.5%에는 적어도 한 명 이상의 구글 직원이 배정된 것으로 분석되었다. 따라서 검증 과정에서 3분의 2 이상의 코드가 구글의 방향성과 합치되는지 여부를 검사받게 되는 것이다.

소속 분류	배정 횟수	배정 비율
구글	887	67.4%
안드로이드 오픈소스 프로젝트	214	16.3%
개방핸드셋연합 소속사 (구글제외)	98	7.4%
외부 단체 및 개인	117	8.9%
총계	1316	100%

<표 6> 안드로이드 운영체제 검증자 배정

19) 한 코드에는 여러 명의 승인자, 검증자가 배정될 수 있으며, 배정되는 사람의 숫자는 코드에 따라 다르다. 또한 한 코드에 같은 곳에 소속된 승인자, 검증자가 복수로 배정될 수 있다.

또한 안드로이드 운영체제의 각 기능에 관련된 하위프로젝트를 총괄하는 사람도 구글의 직원이다.[20] 이러한 사람들은 '프로젝트 리드(project lead)'라고 불리며, 프로젝트 리드는 프로젝트의 모든 기술적인 측면을 주관할 뿐만 아니라 안드로이드 운영체제의 출시 일정에 맞추어 프로젝트의 개발과 검증을 계획, 진행하며 프로젝트에 소속된 사람들 간의 관계를 조율하는 역할을 한다. 특히 검증자와 승인자를 배정하는 것 또한 프로젝트 리드의 중요 업무의 하나이다. 실력이 확인된 검증자를 승인자로 '승진'시키는 것도 프로젝트 리드의 역할이다.

따라서 등록된 코드의 검증은 기본적으로 구글에 의해 임명, 배정된 승인자, 검증자에 의해서 검토될 뿐만 아니라 안드로이드 운영체제의 기술적인 측면을 모두 구글이 총괄하기 때문에, 여러 개발자에 만들어진 코드는 결국 구글의 판단을 거친 후에만 운영체제에 실제 통합되는 것이라고 할 수 있다.

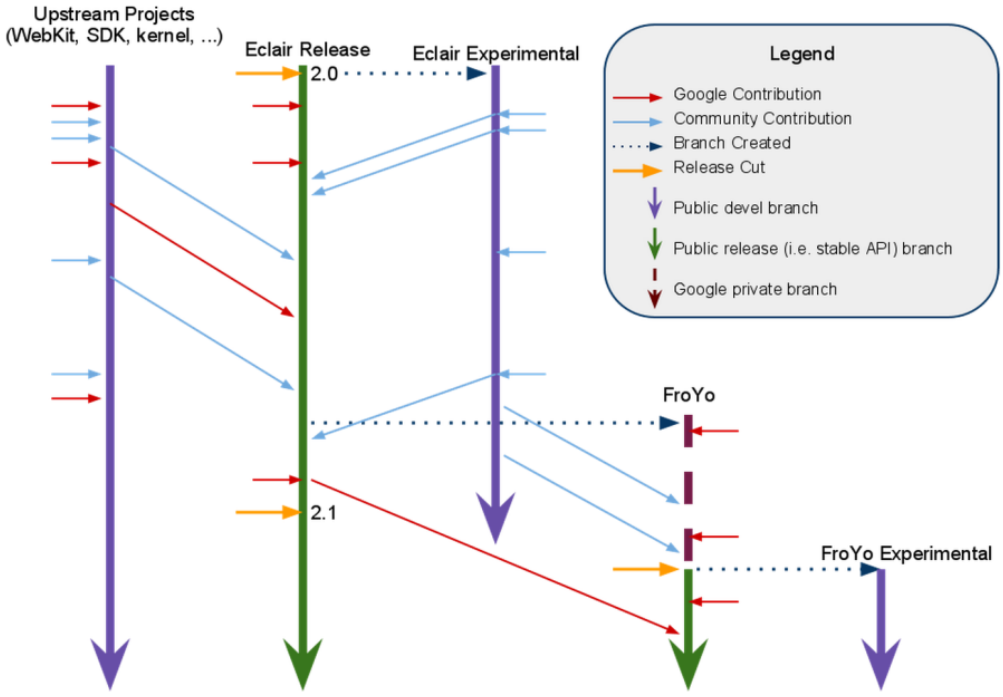
(2) 공식화 과정에서 구글의 역할

구글은 안드로이드 오픈소스 프로젝트의 구성원 중 특정한 코드 덩어리를 '안드로이드'라는 이름으로 규정, 공식화할 수 있는 유일한 행위자이다. <그림 4>에는 안드로이드 운영체제의 코드가 여러 행위자에 의해서 변경, 보완되는 과정이 설명되어 있다. AOSP는 동시에 여러 버전의 코드를 유지하며, 각 코드 버전은 수정, 보완과정을 거치며 변화하고 한 코드 버전에서 다른 코드 버전이 파생되기도 한다. 그리고 이러한 변화가 지속적으로 일어나기 때문에 적절한 시점의 코드에 새로운 버전 번호 혹은 버전 명을 붙여 코드를 구분한다.[21]

---

20) https://source.android.com/source/roles.html 참조

21) 그림에서 아래쪽을 향하고 있는 굵은 화살표들은 각 코드 버전 혹은 브랜치(branch)의 변화를 나타내며, 비스듬한 방향의 화살표들은 구글(빨간색)이나 AOSP의 개발자들(하늘색)에 의해 특정 버전(화살표가 출발하는 선)에 기여된 코드가 공식 코드(화살표가 도착하는 선)에 최종적으로 포함되었음을 나타낸다. 그리고 수평방향의 노란색 화살표는 특정 시점을 기준으로 코드에 정식적인 버전 명이 부여되는 릴리즈 컷(release cut)이 일어났음을 의미한다. 이 그림에는, 안드로이드 운영체제의 코드가 '2.0'라는 버전 명으로 정식 공개되고, 이것이 이 후 구글과 커뮤니티 개발자들에 의해 추가적으로 수정되어



<그림 4> 안드로이드 운영체제 코드의 코드 관리 절차
(출처: http://source.android.com/source/code-lines.html, 2014년 6월 1일)

이러한 코드 변화 과정에서 두 가지 특징을 확인할 수 있다. 첫째, 오직 구글만이 공식 안드로이드 코드를 수정할 수 있다. 구글 이외의 개발자는 '상향 프로젝트'(그림의 'Upstream Projects') 또는 '실험 버전'(예를 들어 그림에서의 'Eclair Experimental'이나 'FroYo Experimental')에만 코드를 직접 기여할 수 있다. 우선 개발자는 안드로이드 운영체제 외의 다른 소프트웨어의 개발을 목적으로 하는 오픈소스 프로젝트[22]에 코드를 기여할 수 있으며, 이러한 프로젝트들의 코드 중 일부가 구글에 의해 선택적으로 공식 안드로이드 운영체제에 반영된다. 또한 공식 안드로이드 운영체제 버전

'2.1' 버전이 정식 공개되는 과정이 설명되어 있다. 또한 2.0에서 2.1의 변화 과정 도중 프로요(Froyo)라는 이름으로 코드가 파생(branch created, 수평방향 파란색 점선 화살표)되었으며, 이 코드는 오직 구글에 의해서만 수정, 보완되다가(Google private branch) 정식 공개되는 과정을 확인할 수 있다.

22) 이러한 상향 프로젝트의 예로서 웹킷(WebKit)을 들 수 있다. 웹킷은 웹 브라우저를 만드는데 사용되는 코드로서, 오픈소스 프로젝트인 '웹킷 오픈소스 프로젝트(WebKit Open Source Project)'에 의해 개발되고 있다. 구글은 웹킷의 코드를 이용하여 안드로이드 운영체제의 웹 브라우저를 개발하고 있다.

공개와 동시에 이 공식 버전과 동일한 코드를 가진 '실험 버전'이 생성되는데, 실험 버전 은 공식 버전과는 개별적으로 관리되며 이 버전의 코드 중 일부가 구글에 의해 선택적으로 공식 코드에 반영된다. 이와 같이 구글 이외의 행위자가 기여한 코드는 구글에 의해 공식 코드에 선택적으로 수용되며,[23] 공식 안드로이드 운영체제를 직접 수정할 수 있는 것은 오직 구글뿐이다[24].

둘째, 오직 구글만이 특정한 코드를 '안드로이드'로 이름 붙일 수 있다. 구글은 비공개 수정과정을 거쳐 특정한 코드를 특정한 버전 명의 안드로이드 운영체제로 규정하고 이를 공식화하여 일반에 공개한다. 이러한 비공개 수정 절차에 대해 구글은, 자신이 안드로이드 운영체제의 '전략적 방향(strategic direction)'을 책임지고 있기 때문에 이 기간 중 소수의 플래그십(flagship) 기기[25]에 집중하여 '안드로이드 스마트폰의 이상적 발전 방향'[26]에 따라 기능을 최적화하고 안드로이드 운영체제 관련 지적자산을 위한 보호장치를 확충하고 있다고 설명한다. 또한 이 과정에서 민감한 기능이 유출되는 것을 막고 공식적으로 안정된 단일 버전의 개발에만 집중하기 위해서 이러한 비공개 방식을 채택했다고 밝히고 있다.[27]

이러한 특성은, 개방성을 통해 기여된 여러 기술들 중 일부를 선택하고 필요한 기술을 추가하여 특정한 코드의 조합을 만들어내고, 이를 '안드로이드'로 명명하는 권한이 오직 구글에게 있다는 것을 의미한다. 이와 같이 안드로이드 오픈소스 프로젝트에서 구글은 자신의 목적 달성을 위해 필요한

---

23) 그림에서 보라색 수직 방향 화살표에서 출발하여 초록색에 도착하는 파란색 사선방향 화살표가, 구글 이외 개발자가 기여한 코드를 구글이 선택적으로 수용하고 있음을 의미한다.

24) 그림에서 녹색 선에 맞닿아 있는 수평방향 빨간 색 화살표이가 구글의 코드 수정을 의미한다.

25) 여기서 플래그십 기기는 구글과 스마트폰 제조사가 협력하여 개발한 최신 안드로이드 스마트폰을 의미한다. 플래그십 기기는 새로운 버전 명의 안드로이드 운영체제의 공식화와 유사한 시기에 출시되며, 구글과 제조사 내부적으로는 최신 운영체제가 실제 스마트폰 기기에 탑재되어 안정적으로 동작할 수 있는지를 점검하고, 외부적으로는 최신 성능과 기능을 홍보하기 위한 목적을 가지고 있다.

26) 원문은 다음과 같다. : We develop the next version of Android by working with a device partner on a flagship device whose specifications are chosen to push Android in *the direction we believe it should go*. (이탤릭체는 본문 인용부분을 위한 강조임)

27) http://source.android.com/source/code-lines.html#about-private-code-lines 참조

기술을 추가하는 개발자일 뿐만 아니라, 개방적 참여를 통해 기여되는 수많은 기술을 검열, 취사선택하여 기술 조합을 완성하는 문지기(gate keeper)이자 특정한 기술 조합을 안드로이드로 규정하는 명명자로서, 기술체계를 특정한 방향으로 변화시킬 수 있는 체계건설자의 중심적 지위를 획득할 수 있게 된다.

(3) 구글의 권한 배분과 기술체계의 배타성

개방핸드셋연합에 속한 기업은 안드로이드 운영체제의 코드에 영향을 미칠 수 있다. 개방핸드셋연합에 소속된 기업들은 AOSP에도 참여하여 안드로이드 운영체제의 기여, 심사, 통합 과정에 관여하기 때문이다. 하지만 이와 더불어 개방핸드셋연합 소속사는 연합에 소속되지 않은 일반 기업들이 갖지 못한 특권을 부여 받는다. 이러한 특권은 구글로부터 안드로이드 운영체제에 대한 지식과 안드로이드 운영체제에 대한 차별적인 접근권을 제공받는 것이다.

안드로이드 운영체제가 아직 확산되지 않았던 초기에는 구글의 우월적 지위도 아직 확립되지 않았다고 할 수 있다. 오히려 안드로이드 스마트폰을 개발하는 제조사들이 낮은 시장 점유율[28]을 만회하기 위해 새로운 제조사의 확보에 총력을 기울였던 구글보다 우월한 지위를 차지하고 있었을 가능성이 크다. <표 7>는 첫 안드로이드 기기가 출시된 2008년에서 2009년까지 출시된 모든 안드로이드 기기를 조사한 것이다.[29] 2009년 말까지 전 세계에서 출시된 18 종의 기기 중에서 개방핸드셋연합에 가입된 모바일 기기 제조사가 출시한 스마트폰은 15개에 이른다.

하지만 안드로이드 운영체제의 확산에 따라 기술의 생산 과정을 장악하고 있는 구글이 제조사에도 영향력을 미치게 되었다. 앞서 언급된 것처럼, 구글은 새로운 안드로이드 운영체제의 공식화를 비공개 상태에서 독자적으

28) 2008년 시장 점유율 조사에서 안드로이드 운영체제는 목록에 이름을 올리지도 못했다 (Gartner, 2009).
29) 본 자료는 미국의 휴대전화 정보제공 사이트인 폰아레나(www.phonearena.com)과 지에스엠아레나(www.gsmarena.com)에서 수집한 정보를 조합한 것이다.
30) 미국에서 운영되고 있는 터키계 휴대용 기기 제조회사

기기명	제조사	출시일
HTC Dream	**HTC**	2008년 10월
HTC Magic	**HTC**	2009년 04월
Samsung Galaxy	**삼성**	2009년 06월
General Mobile DSTL1 Imaginary	General Mobile[30]	2009년 07월
HTC Hero	**HTC**	2009년 07월
Sony Ericsson Xperia X3	**소니 에릭슨**	2009년 07월
Lenovo O1	레노보	2009년 08월
Archos 5 Internet Tablet	Archos[31]	2009년 09월
T-Mobile Pulse	**화웨이[32]**	2009년 10월
Motorola Cliq	**모토롤라**	2009년 10월
HTC Tatoo	**HTC**	2009년 10월
Samsung Moment	**삼성**	2009년 11월
LG GW620	**LG**	2009년 11월
Motorola Droid	**모토롤라**	2009년 11월
HTC Droid Eris	**HTC**	2009년 11월
Samsung Behold 2	**삼성**	2009년 11월
Acer Liquid	**에이서**	2009년 12월
Samsung Galaxy Spica	**삼성**	2009년 12월

<표 7> 안드로이드 초기 탑재 기기 (어두운 색은 OHA 가입 제조사가 출시한 기기를 의미)

로 준비하며, 이 기간 동안 새 운영체재를 특정 제조사의 기기에 이를 탑재하여 시험한다. 이러한 플래그십 기기는 코드 공개와 같은 시기에 출시되며, 지금까지 HTC[33], 엘지, 삼성, 모토롤라[34], 에이수스[35]에 의해 만들어졌다. 이 기업들은 모두 개방핸드셋연합 가입 기업일 뿐만 아니라, 에이수스를 제외한 나머지 기업은 연합 창립 당시부터 구글과 함께한 기업들이다. 플래그십 기기의 출시 시점은 안드로이드 운영체제의 공개 시점과 최대 3개월까지의 차이를 보이나, 2010년 이후의 플래그십 기기는 모두 최신 안드로이드 운영체제의 공개와 동시에 출시되었다(<표 8>). 이렇듯 개방핸드셋연합 가입 기업들이 운영체제의 공개와 동시에 스마트폰을 출시할 수

31) 프랑스의 휴대용 전자기기 제조회사
32) 중국의 전자제품 제조회사
33) HTC는 중국의 스마트폰 제조사이다. 스마트폰 이전에는 노트북 컴퓨터 및 휴대 기기를 생산하였다.
34) 모토롤라(Motorola)는 미국의 전자제품 제조업체이다. 모토롤라의 통신장비 제조부문은 2011년 모토롤라 모빌리티(Motorola Mobility)로 분사되었으며, 이 회사는 같은 해에 구글에 인수되었다. 이 후 구글은 2014년 모토롤라 모빌리티의 스마트폰 제조 부문을 중국의 전자제품 제조회사인 레노보(Lenovo)에 매각하였다.
35) 에이서스(Asus)는 중국의 전자제품 제조회사이다.

안드로이드 버전명 (버전번호)	공개시점	최초 탑재 모델명	제조사	기기출시 시점
Android 1.0	2008.9	T-mobile G1	HTC	2008.10
Cupcake (1.5)	2009.4	T-mobile mytouch 3G (영국 출시)	HTC	2009.7
Donut (1.6)	2009.9	Motorola Devour	모토롤라	2010.2[36]
Eclair (2.0)	2009.10	Motorola Droid	모토롤라	2009.10
Froyo (2.2)	2010.5	Nexus One (Update)	HTC	2010.5
Gingerbread (2.3)	2010.12	Nexus S	삼성	2010.12
Honeycomb (3.0)	2011.2	Motorola Xoom	Motorola	2011.2
Ice Cream Sandwich (4.0)	2011.11	Galaxy Nexus	삼성	2011.11
Jelly Bean (4.1)	2012.7	Nexus 7	에이수스	2012.7
Kit Kat (4.4)	2013.10	Nexus 5	LG	2013.10

<표 8> 안드로이드 운영체제 버전별 플래그십 기기

있는 것은, 구글이 운영체제의 코드에 대한 독점이고 폐쇄적인 권한을 유지하는 공식화 준비 기간 동안, 이러한 기업들에게만 기술에 대한 접근 권한과 안드로이드 운영체제 기술에 대한 지식을 제공하고 있기 때문이다. 이와 같이 구글은 코드에 대한 독점적인 권한을 차별적으로 분배함으로써, 개방핸드셋연합에서 다른 소속 기업들, 특히 스마트폰 기기 제조사에 비해 우월한 지위를 점유할 수 있다.

구글은 이러한 권한 배분을 통해 제조사에 대한 영향력을 유지하고, 이를 통해 제조사와 설정한 공통의 목표를 달성하려 한다. 이 목표는 정보통신 기술 업계에서 형성되어 있는 기존 관계를 해체시키고 새로운 기술체계를 확장하는 것이다. 안드로이드 운영체제를 어떠한 스마트폰 기기에서든지 사용할 수 있게 하자는 개방핸드셋연합의 '개방성' 또한, 기존의 운영체제-스마트폰 관계(심비안-노키아폰, iOS-아이폰, 블랙베리 운영체제-블랙베리)에 대한 탈피를 전제하고 있다. 제조사 또한 개방핸드셋연합과 안드로이드 운영체제를 통해 무료로 새로운 운영체제를 획득하고 기존의 기술체계들을 와해시키는 것이 필요했으며, 이는 2007년 개방핸드셋연합에 가입한 제조사 중 자체 개발된 모바일 운영체제를 보유한 기업이 없었다는

36) 안드로이드 도넛 버전의 경우 직전의 컵케익 버전과의 시간 및 코드 차이가 크지 않고, 다음 버전인 이클레어와의 공개 시점도 한 달 밖에 차이가 나지 않는다. 따라서 많은 기업들이 이전 버전인 도넛을 탑재하거나 좀 더 늦은 시점에 최신 버전인 이클레어를 탑재하는 방식을 택하였다.

점에서 드러난다[37].

또한 이러한 배타성은, '개방적 협업을 통해 새로운 운영체제를 개발한다'는 목표에도 불구하고 마이크로소프트, 노키아, 애플, RIM와 같이 상대적으로 오랜 기간 모바일 운영체제 기술을 축적해온 기업들이 이 연합에 포함되지 않았다는 사실에서도 확인된다. 이 회사들에게 가입을 요청하지 않았느냐는 질문에, 구글의 안드로이드 사업총괄자는 직접적인 답변을 피하며 '연합에 속한 각 기업은 무엇인가를 공헌해야 한다고' 말하고, 이 연합은 '참여하고 기여하는 사람들에게 열려있는' 것이라고 말하였다. (ZDNet, 2007.11.5.) 이는 곧 구글과 개방핸드셋연합이 기존의 회사들 및 그들이 개발한 운영체제들을 기술적 교류가 불가능한 경쟁, 대립 대상으로 바라보고 있음을 보여준다. 이와 같이 개방핸드셋연합의 '개방'은 어떠한 기업이든 참여하여 어떠한 행위든 할 수 있는 '자유'가 아니다. 연합에 참여하는 기업들은 특정한 목표에 기여할 수 있어야 한다. 그리고 이 목표가 현재의 '문제점'을 인식하고 이를 혁신하는 것에서부터 시작되므로, 축적된 기술을 통해 기존 시스템에서 주도적인 위치를 차지하고 있던 기업들은 이 연합에 소속될 수 없었다. 이 연합이 뜻하는 '개방'은 최대한 많은 수의 행위자들을 확보하는 것뿐만 아니라 설정된 목표에 부합하는 행위자들을 선택적으로 기술체계에 참여시킴으로써 기존 체계에 대항하는 연합을 구성하기 위한 배타성이다.

또한 구글이 안드로이드 운영체제 생산과정에서 확보한 독점적인 권한을 배분하여 개방핸드셋연합에서 우월적인 지위를 유지하고 있기 때문에, 개방핸드셋연합은 (연합의 명칭과는 대조적으로) 구글이라는 개별 기업의 입장이 가장 중요하게 통용되는 폐쇄성을 띄고 있다. 우선 개방핸드셋연합의 가입을 결정하는 것은 구글이다. 어떤 기업이든 연합에 가입할 수 있는 것은 아니며, 가입하고자 해도 구글에 의해 거절될 수도 있다. 예를 들어 한국의 스마트폰 제조사인 팬택(Pantech)은 안드로이드 스마트폰 개발을 결정했지만 구글이 팬택의 연합 가입을 거부했다. 또한 개방핸드셋연합은 운

37) 삼성은 2009년 10월 모바일 운영체제인 바다(bada)를 발표하였으나, 2013년 2월 개발 종료를 선언하였다.

영 절차나 규정 조항들이 외부에 밝혀져 있지 않은 채 운영되고 있으나[38], 연합 내부의 갈등이 생길 때 연합의 목표를 명확히 하고 이를 통해 다른 소속사를 견제, 감독하는 것은 주로 구글의 몫이었다. 이는 구글과 중국의 모바일 기기 제조사인 에이서스 간의 갈등 사례에서도 확인할 수 있다. (이 사례는 5장에서 자세하게 다루어진다.)

38) 이러한 폐쇄성 때문에 개방핸드셋연합이 단지 기업의 사교 모임 정도로 기능하고 있는 것으로 보기도 하지만, 이는 안드로이드 운영체제의 생산 과정과 구조를 정확하게 파악하지 못하여 비롯된 주장이다.

III. 모바일 어플리케이션 영역의 개방성과 지배

1. 모바일 어플리케이션 영역의 개방성

(1) 개발자의 참여 확장을 위한 기술적 설계

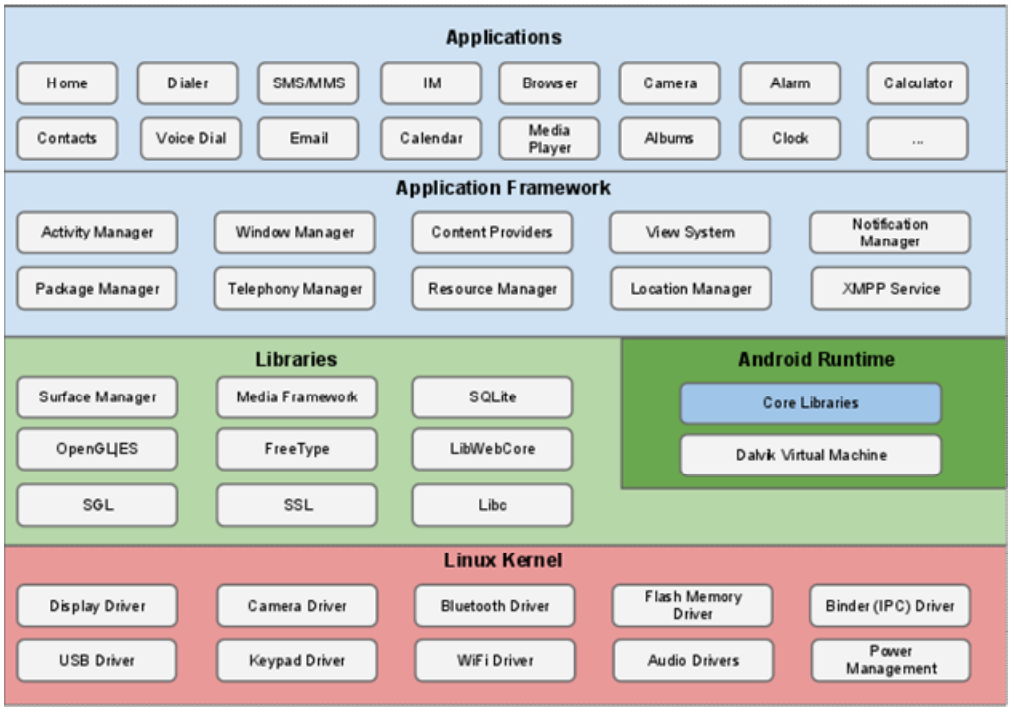
안드로이드 어플리케이션 개발자는 어플리케이션을 만드는 '생산자'이자 안드로이드 운영체제라는 기술적 기반과 어플리케이션 개발자 도구라는 생산 도구를 이용하는 '사용자'이기도 하다. 따라서 이와 같은 환경에서 '생산자 설정하기'는 '사용자 설정하기(configuring the user)'를 통해서도 이루어질 수 있다(Woolgar, 1991). 안드로이드 운영체제에서 동작하는 모바일 어플리케이션이 가능한 한 많이 만들어지도록 하기 위해 구글은 많은 수의 생산자를 필요로 했으며, 많은 수의 생산자를 확보하기 위해서는 이들이 사용하는 기술적 기반과 생산 도구가 최대한 많은 수의, 일반의, 보통의 사람들을 대상으로 설계되어야 했다. 이를 위해 구글은 기술적 기반과 생산 도구가 가장 포괄적이며 일반적인 정체성을 목표로 할 수 있도록 설계했으며, 이러한 설계에 의해 기술체계는 사용자이자 생산자인 많은 수의 개발자들의 관심을 끌게(interessment)된다.

1) 일반적 언어를 확보하기 위한 기술적 동원

소프트웨어를 만들기 위해서는 프로그래밍 언어(programming language)로 이루어진 코드를 작성하여야 한다. 프로그래밍 언어의 종류는 C 언어, C++ 언어, 자바(Java), 파이썬(Python) 등으로 다양하며, 각 프로그래밍 언어는 각기 다른 문법(syntax)를 갖는다.[39]

안드로이드 운영체제는 크게 하드웨어와 관련된 부분인 커널 영역(kernel space)과 사용자가 이용하는 어플리케이션을 동작시키기 위한 기능

39) 프로그래밍 언어의 문법이란 각 언어에서 문자를 배열, 조합하는 규칙을 말하며, 특정 언어를 사용하여 코드를 작성하기 위해서는 반드시 그 언어의 문법에 따라야 한다.



<그림 5> 안드로이드 운영체제의 구조
(출처: http://source.android.com/devices/tech/security)

의 집합인 사용자 영역(user space)으로 구분될 수 있다. 안드로이드 운영체제에서 커널(kernel)이라는 부분은 C 또는 C++이라는 프로그래밍 언어로 만들어진다. 반면 안드로이드의 사용자 영역은 모두 자바 언어로 만들어진다.

이를 더욱 구체적으로 살펴보면 다음과 같다. <그림 5>에서 확인할 수 있는 바와 같이 안드로이드 운영체제의 코드는 각기 다른 기능을 담당하는 여러 부분으로 구분되며, 이 중 화면, 카메라, 메모리, 키패드, 와이파이, 오디오 등을 구동시키는 부분인 리눅스 커널과 이러한 커널의 동작을 보조하는 기능을 하는 코드들의 모임인 라이브러리(library)는 C 언어로 작성된다. 반면 연락처, 전화걸기, 웹검색 등의 기능을 가진 모바일 어플리케이션과 여러 가지 어플리케이션들이 실제로 스마트폰에서 동작할 수 있도록 자원 관리, 위치 검색, 전화, 화면 표시 등의 기능을 제공하는 어플리케이션 프레임워크(Application Framework) 부분은 자바 언어로 이루어져 있다.

이와 같이 안드로이드 운영체제를 구성하는 프로그래밍 언어는 C언어와 자바 언어로 이분화되어 있다.

이러한 기술적 구조는 구글이 선택한 것이다. 구글과 안드로이드 사는 안드로이드 운영체제의 커널 부분을 C언어로 만들어진 자유소프트웨어인 리눅스 커널로 대체하여 개발에 드는 비용과 시간을 줄일 수 있었다. 이후 구글은 사용자 영역에 사용할 프로그래밍 언어로 자바, 자바스크립트(Javascript), 파이썬, 루아(Lua) 등 몇 가지를 검토한 끝에 최종적으로 자바를 선택했다(ZDNet, 2012.4.24.). 구글이 자바를 선택한 이유에는 이미 전 세계에 600만 명의 자바 개발자가 존재한다는 점, 자바가 모바일 기기 개발에 가장 적합하다는 점, 개발에 필요한 문서와 도구가 다수 존재한다는 점, 통신 사업자와 기업들이 자바를 선호한다는 점 등이 있다[40](Oracle, 2012:38-40).

이와 같이 사용자 영역에 자바 언어를 선택하고 모바일 어플리케이션도 자바 언어로 만들어지게 됨에 따라, 운영체제에서 자바 언어로 만들어진 코드를 구동시키기 위해 필수적인 가상머신이라는 기능이 필요해 졌다.[41] 이에 따라 구글은 자체적인 가상머신을 제작하고 이것에 달빅가상머신(Dalvik Virtual Machine)[42]이라는 이름을 붙였다.

이와 같이 구글은 C언어에 기반한 구조 위에 자바 언어에 기반한 구조를 결합시켜, 자바 언어를 사용하는 개발자, 자바를 위해 만들어진 개발 도구들, 자바로 만들어진 모바일 기기 등으로 이루어진 거대한 기술체계를 새로운 기술체계의 확장에 동원했다. 그리고 이러한 기술적 동원은 가장 널리 사용되는 자바 언어의 문법에 익숙한 대다수의 개발자들과, 이미 존

---

40) 오라클의 설명에 의하면 자바는 기업용 데스크탑의 97%, 미국 내 컴퓨터의 89%, 30억 개의 모바일 전화, 1억 2500만 개의 TV에서 구동되고 있다. 또한 전 세계적으로 900만 명의 자바 개발자가 존재한다. (Oracle, 2014.)

41) 자바는 개발자가 한번 개발한 코드를 다양한 환경에서 동작할 수 있도록 하기 위하여 자바가상머신(Java Virtual Machine)이라는 도구를 사용한다. 자바가상머신은 자바 코드의 실행이 가능한 환경을 만들어 다양한 컴퓨터 환경에서 자바 코드가 문제없이 동작할 수 있도록 한다.

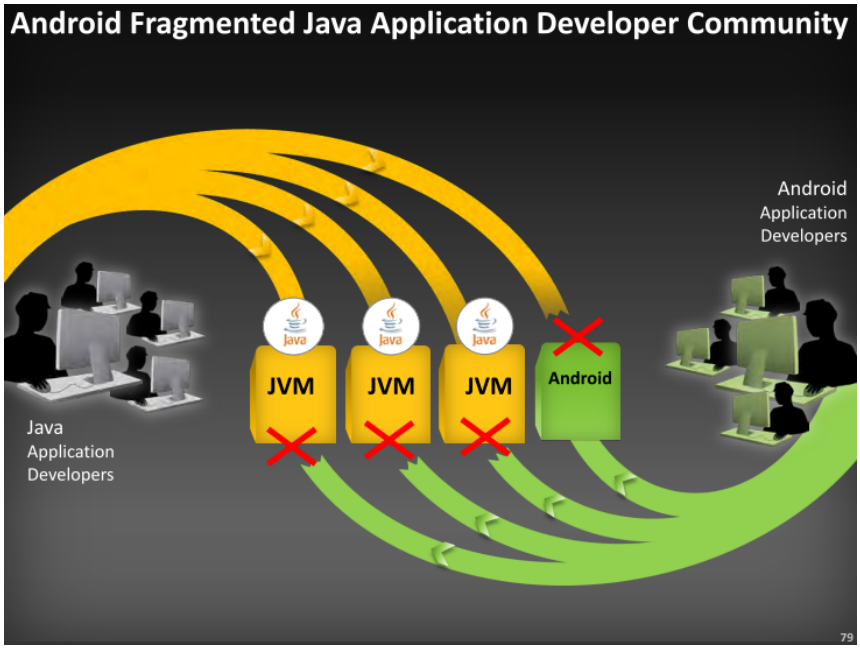
42) 달빅가상머신이란 이름 자체도 구글의 가상머신 개발 담당자인 댄 본스타인(Dan Bornstein)이 자신의 선조가 살았던 아이슬란드 도시 이름을 따서 만든 것이다.

재하는 도구, 지식을 이용해 이 언어를 쉽게 배울 수 있는 잠재적 개발자들에 의해 많은 수의 어플리케이션이 개발되도록 하기 위한 것이다.

여기서 달빅가상머신은 C언어에 기반한 기술의 안정성과 자바 언어에 기반한 기술의 확장성을 결합시키는 장치이자 체계의 확장을 위해 다른 체계의 기술적 요소를 동원하기 위한 장치이다. 또한 달빅가상머신은 이러한 동원으로 인해 안드로이드 체계에 다른 행위자의 영향력이 확장되는 것을 막기 위해 설계되었다. 가상머신 중 가장 널리 쓰이는 자바가상머신(Java Virtual Machine, JVM)은 오라클이라는 기업의 소유물이기 때문에, 달빅가상머신은 구글이 특허 및 라이센싱 문제를 피하기 위해서 이를 대체할 새로운 가상 머신인 것이다. 즉 구글은 달빅가상머신을 통해 다른 기술체계를 동원하되 이로 인한 영향력 확장을 회피하는 '우회적 확장'을 의도한 것이다.

달빅가상머신은 한 기술체계가 확장될 때 다른 시스템과 어떻게 상호작용하는지를 보여준다. 자신의 권력을 확장, 강화하고 다른 행위자의 권력은 차단, 약화시키려는 체계건설자의 의도는 달빅가상머신이라는 하나의 기술적 요소에 모두 내재되어 있다. 이와 같이 기술적 요소는 체계 내부의 필요성뿐만 아니라 다른 기술체계 속한 요소들의 장점, 다른 체계가 영향력의 범위와 같은 '기술적 지형'을 고려해서 만들어진다.

여기에서 구글이 차용한 자바의 '문법'은 물질적인 요소가 아니기에, 안드로이드 운영체제에 자바를 도입되어 자바의 기술체계가 물질적으로 축소, 붕괴되는 것은 아니다. 오히려 새로운 체계에서 자바 언어가 사용되는 것은 언어, 도구, 자원에 의해 유지, 확장되어온 자바 기술체계의 확장으로 이해할 수 있다. 하지만 이러한 확장이 기술체계의 중심 행위자에 의해 승인되지 못할 경우, 확장된 부분은 부정되고 오히려 확장된 요소를 공격하려는 시도가 일어난다. 예를 들어 일부 자바 기술에 대한 소유권을 가지고 있는 오라클 사는 구글이 자바 언어를 '도용'함으로 인해 자신들의 권리가 침해되었을 뿐만 아니라, '한번 만들어 모든 곳에서 적용할 수 있는' 자바의 개발환경이 손상되었다고 주장하였다(<그림 6> 참조). 또한 오라클은



<그림 6> 안드로이드 체계에 의한 자바 체계의 훼손
(Oracle, 2012)

구글이 오라클과 관련된 어떠한 것으로 부터도 영향 받지 않기 위해 클린룸(clean room) 상태의 개발을 시도했으나 실패했다고 주장했다(Oracle, 2012).

또한 달빅가상머신이 체계의 확장을 위한 구조임에도 구글이 이를 통해 이루어지는 모든 확장을 승인하는 것은 아니다. 모든 안드로이드 어플리케이션은 달빅가상머신 위에서 동작하며, 다른 운영체제일지라도 달빅가상머신을 탑재할 수만 있다면 안드로이드의 어플리케이션을 구동하는 것이 가능하다. 더군다나 달빅가상머신이 오픈소스의 범위에 해당하기 때문에 이러한 작업은 충분히 가능하다. 하지만 구글은 다른 운영체제에서 안드로이드 어플리케이션 시장에 접근하지 못하도록 하고 있다. 오라클과 구글 간의 분쟁, 안드로이드 어플리케이션 시장에 대한 접근 제한의 사례들은 목적에 부합하지 않는 체계의 확장이 중심 행위자에 의해 거부될 수 있음을 보여주고 있다.

2) 일반적 도구를 확보하기 위한 기술적 삽입

개발자들은 코드를 작성하고, 작성한 코드를 다른 코드들과 연결시키며, 모든 코드를 기계가 해석할 수 있는 형태의 언어로 번역해야 한다. 이러한 작업들을 수행하려면 각각의 기능을 제공하는 여러 개의 프로그램이 필요하지만, IDE(Integrated Development Environment, 통합개발환경)라는 소프트웨어는 개발자가 다양한 작업을 한 곳에서 처리할 수 있도록 한다. 이클립스(Eclipse)는 오라클 사의 넷빈즈(Netbeans)와 함께 자바 개발에 가장 많이 사용되는 IDE의 하나이다. 이클립스는 이클립스 재단(Eclipse Foundation)에 의해서 개발, 배포되며 이클립스공공라이선스(Eclipse Public License, EPL)를 따르는 오픈소스 프로그램이기 때문에 누구나 인터넷 상에서 다운로드 받아 사용할 수 있다. 특히 이클립스는 개발자가 다양한 프로그래밍 언어를 한 곳에서 작성할 수 있도록 하려는 목적을 가지고 만들어졌으며, 이클립스와 결합되어 이클립스의 기능을 강화, 확장하는 코드인 플러그인(plug-in)이 다양하게 존재한다. 개발자는 이클립스에 필요한 플러그인을 설치하여 추가적인 프로그래밍 언어나 기능을 사용할 수 있다. 또한 이클립스는 폭넓은 사용자층을 확보하고 있으며, 사용법과 기능에 관한 문서와 자료 등 이클립스의 사용을 돕는 자료들도 인터넷을 통해 쉽게 구할 수 있다. 이와 같이 이클립스 재단, 이클립스의 코드, 이클립스의 플러그인, 이클립스 사용자들은 기술체계를 형성한다.

구글은 안드로이드 어플리케이션의 개발에 필요한 도구들을 제공한다. 특히 구글이 2009년부터 제공하고 있는 SDK(Software Development Kit, 소프트웨어 개발 도구)인 ADT(Android Development Tools, 안드로이드 개발 도구)는 어플리케이션 개발에 필수적인 도구들을 포함하고 있다. ADT는 이클립스와 결합되는 플러그인이기 때문에, ADT를 다운로드받아 이클립스에 설치하면 이클립스에서 안드로이드 어플리케이션 개발을 위한 기능들(어플리케이션 디자인, 작성된 코드의 오류의 검증과 최적화, 어플리케이션과 운영 체제 연결 등)을 사용할 수 있다. ADT는 안드로이드 운영

체제를 구성하는 코드는 아니지만 안드로이드 운영체제가 어플리케이션에 제공할 수 있는 자원의 종류와 특성, 운영체제에 의한 규정과 기술적 제약을 나타내며 어플리케이션의 '개발 환경'을 제시한다. 이 때문에 이클립스에 ADT 플러그인을 설치하는 것은 일반 개발자가 어플리케이션을 만들기 위해서 해야 하는 첫 번째 작업이다.

<첫 번째 앱 만들기>
이 수업을 시작하기 전, 개발 환경이 준비되어 있는지 확인하세요. 당신은
1. 안드로이드 SDK를 다운로드 해야 합니다.
2. 이클립스를 위한 ADT 플러그인을 설치해야 합니다. (이클립스 IDE를 사용하려는 경우)
3. SDK 관리자를 이용하여 최신 SDK 도구들과 플랫폼을 다운로드 해야 합니다.

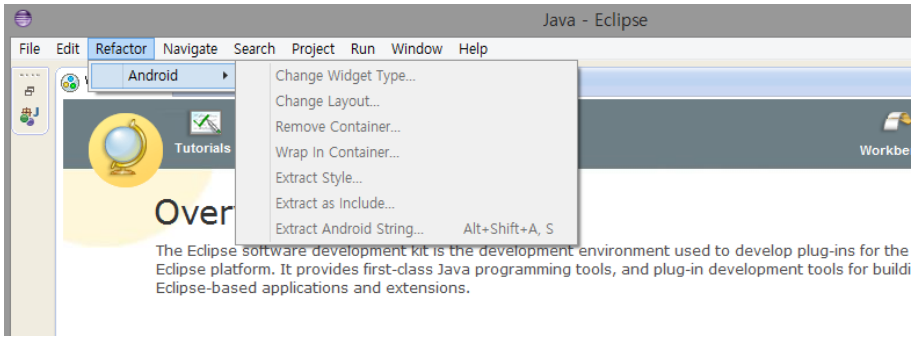
안드로이드 개발자 웹사이트 中[43]

구글은 이클립스에 ADT를 결합시킴으로써 기존의 이클립스 사용자가 안드로이드 시스템에 쉽게 편입될 수 있게 하였으며 새로이 개발을 시작하려는 사람도 이미 널리 퍼져있고 무료로 쉽게 사용할 수 있는 이클립스를 통해 안드로이드 시스템에 더욱 쉽게 연결될 수 있도록 하였다. 이를 통해 구글은 새로운 IDE를 개발하기 위해 필요한 비용과 시간을 절약할 수 있었을 뿐만 아니라 이클립스 코드, 플러그인, 자원 등의 이클립스 체계의 요소들을 안드로이드 시스템의 요소로 편입할 수 있게 되었다. ADT를 통해 '잘 알려지지 않은' 안드로이드의 기능 및 구조는 이클립스의 안정되고 '잘 알려진' 구조에 삽입된다.

ADT는 통합개발환경에 결합된 도구들을 제공하는 이클립스 플러그인입니다. 이것은 당신에게 안드로이드 어플리케이션의 빠른 개발을 돕는 많은 기능들을 제공합니다. [중략] ADT가 이클립스의 플러그인이기 때문에, 당신은 이클립스의 안정적인 기능성과 함께 ADT와 결합된 안드로이드만의 특

43) http://developer.android.com/training/basics/firstapp/index.html

징들을 접하게 됩니다. (Google, 2013.3.18.)

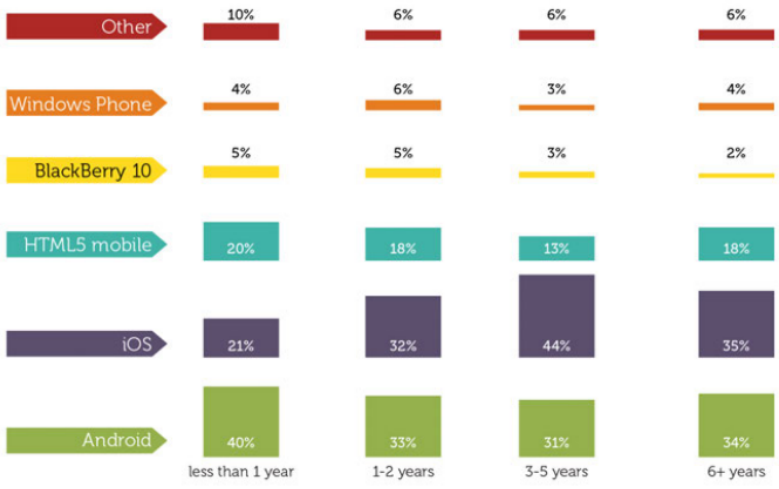


<그림 7> 이클립스에 ADT가 설치된 모습

블랙베리, 바다, 심비안과 같은 다른 모바일 운영체제들도 모두 이클립스에서 자신의 코드가 개발될 수 있도록 각각 플러그인을 제공하고 있으며, 윈도우즈 모바일 또한 개발자에게 자사 유료 IDE인 비쥬얼스튜디오(Visual Studio) 중 일부를 무료로 사용할 수 있도록 하고 있다.[44] 특히 블랙베리의 경우, 어플리케이션 다양성의 부족과 개발의 어려움을 사용자 감소의 원인으로 보고, 개발자와의 연계를 강화하기 위해서 다양한 언어를 사용하는 개발자들을 위한 코드를 대거 공개하였다. 그리고 여기에는 안드로이드 등의 다른 운영체제에서 작동하는 어플리케이션을 블랙베리에서도 동작할 수 있도록 하는 코드들이 포함되었다. 이는 기술 시스템의 유지와 확장을 위해 다른 시스템 요소를 유인하는 것이다.[45] 이처럼 많은 개발자와 어플리케이션을 확보하는 것은 모바일 운영체제의 생존에 있어 매우 중요하다. 개발자의 수가 많을수록 많은 어플리케이션이 운영체제에 결합될 수 있으며, 어플리케이션 수가 증가할수록 운영체제가 다양한 시간과 장소

44) 반면 iOS의 개발자는 애플사가 독자적으로 개발한 엑스코드(Xcode)라는 IDE를 사용해야 한다. 이 IDE는 오브젝트 C(Object C)라는 특수한 프로그래밍 언어를 사용할 뿐만 아니라 애플사가 개발한 운영체제에서만 동작한다.

45) 블랙베리 운영체제의 개발사인 RIM은 개발자가 비쥬얼스튜디오, 코르도바 등 기존의 IDE에서 C++, 자바스크립트, HTML5, CSS를 이용해 어플리케이션을 만들 수 있는 도구들을 공개하였다. 뿐만 아니라 Adobe Air와 안드로이드 환경에서 이미 개발된 어플리케이션을 블랙베리에서도 동작시킬 수 있게 하는 도구도 공개하였다. (TechRepublic, 2012.7.9.)



<그림 8> 모바일 어플리케이션 개발 경력에 따른 모바일 운영체제 선호도 (VisionMobile, 2013)

에서 다양한 용도로 사용되며 사용자와의 결합을 확장시킬 수 있다.

이와 같이 개발자들은 구글의 설계로 인해 널리 알려진 프로그래밍 언어와 널리 사용되는 생산 도구를 이용하여 모바일 어플리케이션을 개발할 수 있게 되었다. 이는 모바일 어플리케이션을 처음 개발하는 사람들이 안드로이드 운영체제를 가장 선호한다는 점에서도 드러난다. <그림 8>은 모바일 어플리케이션 개발 경력에 따른 운영체제 선호도를 나타내고 있다. 여기에서 2년 이내의 경력을 가진 경우 안드로이드 운영체제를 가장 선호하는 것을 확인할 수 있으며, 특히 1년 이내의 경력을 가진 개발자들이 안드로이드 운영체제를 선호하는 비율은 40%에 달하여 애플 사의 iOS에 비해서도 두배에 육박하고 있다. 이는 안드로이드 어플리케이션의 개발에 사용되는 언어와 도구의 습득과 접근이 다른 운영체제에 비해서 쉽다는 것을 의미한다.[46]

46) 이와 반대로 모바일 어플리케이션 개발 초심자가 iOS를 선택하는 비율이 낮은 이유는, 배우기가 어렵고 개발에 필요한 장비 구입과 수수료 지불에 비용이 지출되기 때문이다. (VisionMobile, 2013)

(2) 모바일 어플리케이션의 맥락화

모바일 어플리케이션 영역의 개방성은 기술체계가 많은 개발자를 수용할 수 있도록 설계되었다는 사실 뿐만 아니라, 이렇게 유입된 개발자들이 외부의 제약 없이 자신이 원하는 목적과 기능을 가진 모바일 어플리케이션을 자유롭게 생산, 배포할 수 있다는 사실과도 연관된다.[47] 그리고 이 때문에 다양한 어플리케이션이 전 세계에서 각각의 특수한 맥락에 맞게 생산, 사용될 수 있게 되었다.[48]

안드로이드 운영체제에서 각 모바일 어플리케이션이 사용되는 국가의 수를 확인함으로써 전 세계에서 각각의 맥락에 따라 다양한 어플리케이션이 생산, 사용되는 것을 확인할 수 있다. <표 10>은 전 세계 35개국을 대상으로 각 국에서 사용되는 안드로이드 어플리케이션 목록을 수집한 뒤, 이로부터 각 어플리케이션이 사용되는 국가 수를 조사한 것이다.[49] 여기에서 30개 국가 이상에서 사용되는 모바일 어플리케이션은 총 8개로, 조사된 전체 어플리케이션의 1% 밖에 되지 않는다. 반면 단일 국가에서만 사용되

---

47) 구글은 안드로이드 어플리케이션에 대한 심사 및 검열을 실시하지 않고 있다.

48) 세계 모바일 운영체제의 80%를 차지하고 있는 구글은 전 세계에 유통망을 확보하여 개인 및 작은 규모의 기업이 자신의 기술을 더욱 쉽게 확산시킬 수 있도록 하고 있다. 구글은 개발자가 어플리케이션 출시 국가를 선택할 때 고려해야할 요소를 아래과 같이 나열하고 있는데, 이는 어플리케이션이 적절한 내용, 설명, 요건을 갖추었다는 개발자 자체적인 판단만 있다면, 배포, 유통 과정 자체에 각 국가의 정책, 법률, 관습이 전혀 장애가 되지 않는다는 것을 의미한다. 개발자는 구글의 네트워크를 통해 개발자는 세계 140개국에 어플리케이션을 출시할 수 있으며, 이 과정은 어플리케이션을 출시하려는 국가의 위치와 개수에 상관없이 단지 몇 번의 클릭만으로 이루어진다.

- 앱이 지역화된 자원을 필요로 하는지 여부
- 개발자 콘솔에서 지역화된 앱에 대한 설명이 필요한지 여부
- 특정 국가들의 앱에 대한 법적 요구사항
- 시간대 지원, 지역별 가격 책정 등

안드로이드 개발자 홈페이지

(http://developer.android.com/distribute/googleplay/about/distribution.html)

49) 본 논문에서는 전 세계 35개국을 대상으로 2012년 1월부터 2012년 7월까지 3개월을 간격으로 총 7회에 걸쳐 각 국의 구글 플레이에서 인기 무료 어플리케이션으로 제시된 어플리케이션 50개의 목록을 수집하였다. 상위 50개의 어플리케이션은 총 어플리케이션 사용 시간의 60%를 차지한다(Nielsen, 2011). 그리고 전 세계의 구글 무료 어플리케이션의 수는 전체 어플리케이션의 70% 이상에 달한다. (Austin, 2013.3.11.)

는 어플리케이션은 전체의 81.0%에 달하며 10개 미만의 국가에서 사용되는 어플리케이션이 전체의 95.2%에 이른다는 사실로부터, 각 국가별로 특화, 맥락화된 모바일 어플리케이션이 생산, 사용되고 있다는 것을 확인할 수 있다.

사용 국가 수	어플리케이션 수	비율
30-35개 국가	8	1.0 %
20-29개 국가	7	0.9 %
10-19개 국가	22	2.9 %
2-9개 국가	109	14.2 %
단일 국가	621	81.0 %
총계	767	100 %

<표 10> 사용국가 수에 따른 어플리케이션 수 및 목록 총 등장횟수
(2013년 7월 20일 기준)

하지만 복수 국가에서 동일하게 사용되는 기술들도 지역, 문화의 영향을 받은 것이다. 이를 확인하기 위해 우선 각 국가의 안드로이드 어플리케이션을 비교하여 국가 간 어플리케이션의 '유사성'을 측정하고[50], 국가들을

데이터 수집 일자 (총 7회)	2012년 1월 20일, 4월 20일, 7월 20일, 10월 20일, 2013년 1월 20일, 4월 20일, 7월 20일.
자료 수집 국가 (총 35개국)	미국, 호주, 오스트리아, 캐나다, 영국, 독일, 프랑스, 스위스, 이탈리아, 벨기에, 네덜란드, 핀란드, 스웨덴, 스페인, 터키, 러시아, 폴란드, 한국, 중국, 일본, 말레이시아, 인도네시아, 필리핀, 베트남, 대만, 인도, 싱가포르, 태국, 홍콩, 멕시코, 브라질, 칠레, 아르헨티나, 남아프리카 공화국, 이집트

<표 9> 데이터 수집 일자와 수집 국가

또한 구글 플레이에는 각 어플리케이션의 기능, 목적 등이 명시되어 있으며, 별도의 기술적 작업을 통해 각국의 어플리케이션을 실제로 다운로드 받아 사용해 볼 수 있으므로 이를 통해 추가적인 분석을 진행하였다. 그리고 스마트폰 및 어플리케이션에 대한 언론 보도 자료, 어플리케이션 개발사의 홈페이지나 홍보물, 시장 조사 업체의 어플리케이션 시장 분석 자료 등도 살펴보았다.

50) 35개 국가에 대해 두 국가 간의 공통 어플리케이션 수의 평균이 일정 값 이상인 경우를 측정하면 각 성분이 0또는 1인 35x35 대칭 행렬을 얻게 되며, 이 행렬을 '유사성 행렬'으로 정의한다. 이 행렬의 각 성분은 성분의 열과 행에 해당되는 국가 간 기술사용의 유사성 여부를 나타낸다(<표 24> 참조).

구조적 등위성(structural equivalence)을 기준으로 분류할 수 있다[51]. 이러한 분류를 통해 만들어진 집단 또는 권역(zone)은 다른 국가에 대한 대응 패턴이 유사한 국가들로 구성되므로, 본 논문에서는 이것을 '기술권역'으로 지칭한다. 또한 두 국가가 서로 다른 기술권역으로 분류된 횟수를 '기술거리'로 정의한다.[52]

<그림 9> 한국을 기준으로 한 기술거리의 분포

51) 구조적 등위성을 이용한 분류 방법으로 클러스터 분석(cluster analysis) 방법이 사용되었다. 본 연구에서는 유사성 행렬에서 적률상관계수(pearson correlation)가 0.4 이상인 국가들을 하나의 집단으로 분류하였다.

52) 분류된 집단의 수와 집단에 소속된 국가들은 데이터 측정 시점에 따라 변화하며, 한 시점에서 같은 그룹으로 분류된 두 국가가 다른 시점에서는 다른 그룹으로 분류되는 일이 빈번히 발생한다. 이를 고려하기 위해 기준 국가 정하고, 한 시점에서 특정 국가가 기준 국가와 같은 그룹으로 분류되었을 경우 0의 값을, 다른 그룹으로 분류되었을 경우 1의 값을 부여함으로써 각 국가 기술사용 분류의 변화를 연속된 수치로 표현하였다. 여기서 각 국가 수치의 합은 총 7번의 측정 중 해당 국가가 기준 국가와 다른 그룹으로 분류된 횟수를 나타내는데, 이 값이 클수록 기준 국가와 다른 집단으로 분류될 확률이 높은 것인 반면 값이 작을수록 동일한 집단으로 분류될 가능성이 높은 것이다. 예를 들어 A국가의 패턴이 [0 1 1 0 0 0 0]이고 B 국가의 패턴이 [1 1 1 1 1 1 1]일 경우, 패턴의 합계가 2인 국가A는 기준 국가와 2번만 다른 그룹으로 분류된 반면, 패턴이 합계가 7인 국가B는 미국과 항상 다른 그룹으로 분류된 것이므로, 국가A는 국가B 보다 기준 국가와 더 유사한 위치에 있을 가능성이 높다고 할 수 있다.

<그림 10> 중국을 기준으로 한 기술거리의 분포

<그림 11> 일본을 기준으로 한 기술거리의 분포

<그림 12> 인도네시아를 기준으로 한 기술거리의 분포

위의 그림들은 기준국가를 각각 한국, 중국, 일본, 인도네시아로 하였을 때 기술거리의 분포를 나타내고 있다.[53] 이 동아시아 국가들은 지리적으로 인접해있지만 각각을 기준으로 한 세계적 거리 분포는 상이한 양상을 보인다. 우선 한국과 중국은 기술거리가 1로 가까울 뿐만 아니라 두 나라를 기준으로 한 거리 분포도 유사하다. 이는 한국 기준의 분포인 <그림 9>와 중국 기준의 분포인 <그림 10>의 비교를 통해 확인될 수 있다(기준 국가를 중심으로 기술 거리가 가까울수록 어둡게 표시되어 있으므로 그래프의 음영 분포를 비교할 수 있다). 또한 한국, 중국, 홍콩, 대만은 서로의 기술거리가 0으로 매우 가까우나 다른 국가들과는 모두 먼 거리를 보이고 있기 때문에, 이들 국가들이 세계적으로 뚜렷이 구별되며 상호간에 매우 유사한 기술을 갖고 있음을 알 수 있다. 반면 일본 기준 분포인 <그림 11>에서 일본은 34개국 중 동북아시아 인접국인 한국, 중국과 기술거리가 가장 멀고 태국, 말레이시아 등 동남아시아와의 거리도 멀다. 그러나 캐나다, 미국, 오스트레일리아, 영국 등 북미와 서・북유럽과 가까운 거리를 유지하고 있다. 이는 일본의 기술사용, 선택 방식이 아시아권 보다는 서구 선진국들과 더욱 유사하다는 것을 보여준다. 한편 <그림 12>의 인도네시아는 동남아시아 문화권인 말레이시아, 베트남, 태국, 필리핀과의 거리가 0으로 매우 가까우며 인도, 폴란드, 러시아, 터키와 남미(아르헨티나, 멕시코, 브라질, 칠레), 아프리카(이집트, 남아공)와의 거리도 0~1 수준으로 매우 가깝다. 하지만 인접국인 호주와의 거리는 6으로 가장 멀뿐만 아니라 서구 선진국과의 기술거리도 멀다.

분석에 있어 서로 간의 기술거리가 0인 국가들은 항상 동일한 국가로 취급되므로 편의상 이들 국가를 묶어서 '국가군'으로 취급할 수 있다. 이 중 3개 이상의 국가로 이루어진 국가군에 단일한 명칭을 붙이면, '국가군1'은 극동아시아 국가(한국, 홍콩, 대만)로 이루어지며, '국가군2'는 동남아시아

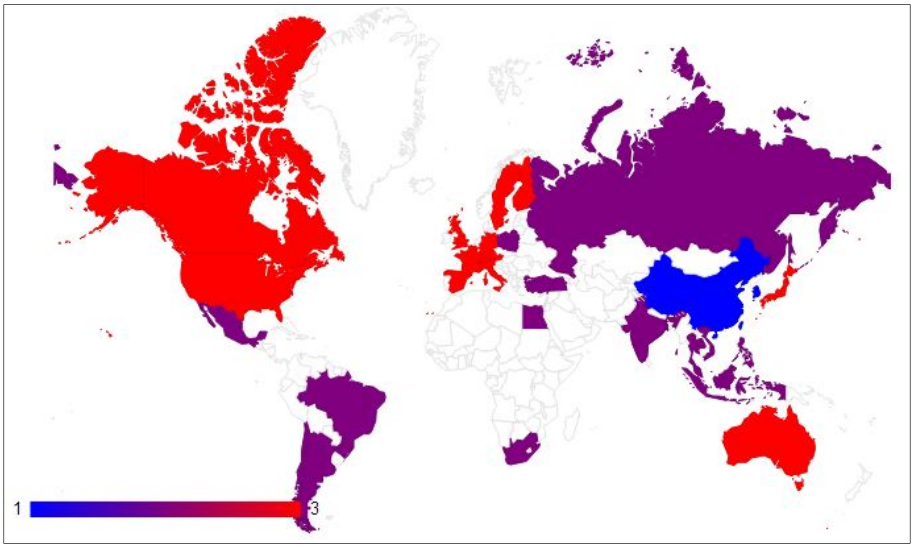
53) 단 기술사용 거리의 비교는 반드시 기준 국가를 중심으로 논의되어야 한다. 기준 국가가 아닌 두 국가의 거리 차가 크다면 두 국가의 기술사용의 차이도 크지만, 기술사용의 차이가 큰 두 국가가 항상 큰 거리의 차를 보인다고 할 수 없다. 또한 기준 국가가 아닌 두 국가의 거리 차가 작다고 해서 두 국가가 기술사용의 유사성을 보인다고 말할 수 없다. 이러한 이유는 기술사용 거리는 모든 국가 간의 차이를 기준 국가 중심으로 투영(projection)한 것이기 때문이다.

국가(말레이시아, 인도네시아, 태국, 필리핀, 베트남)과 인도, 이집트로 이루어진다. 국가군3은 남미에 위치한 국가(아르헨티나, 브라질, 멕시코, 칠레)와 동유럽 국가(러시아, 폴란드, 터키) 및 남아프리카 공화국으로 구성되며, '국가군4'는 서유럽 일부 국가(오스트리아, 독일, 네덜란드)와 핀란드, 캐나다, 일본으로 이루어진다. 그밖에 미국과 호주, 이탈리아와 스위스, 벨기에와 스페인, 영국과 스웨덴은 두 국가로 이루어진 국가군을 형성한다.

기술권역의 분류를 통해 <표 11>와 같은 결과를 얻을 수 있고, 각 권역의 지리적 위치를 도시한 것이 <그림 13>이다. (권역1은 파란색, 권역2는 보라색, 권역3은 빨간색으로 표시되어 있다.)

권역	구성
권역 1	중국, 국가군1
권역 2	국가군2, 국가군3, 싱가포르
권역 3	국가군4, 벨기에, 스페인, 이탈리아, 스위스, 영국, 스웨덴, 미국, 호주, 프랑스

<표 11> 기술권역의 구분



<그림 13> 기술권역의 분포

기술거리 및 권역에 대한 분석을 통해 모바일 어플리케이션이 지리적, 문화적 요인에 영향을 받고 있음을 확인할 수 있다. 우선 기술권역의 분포

는 국가의 지리적 위치와 연관성을 가지고 있다. 동일한 권역에 속하는 국가군1, 국가군2, 국가군3과 서유럽의 국가군(오스트리아-독일-네덜란드, 스위스-이탈리아)들은 지리적으로 인접한 국가들로 구성되어 있으며 동일 권역 내에 국가 간 기술거리가 가장 가깝다.

구체적으로 권역 1은 중국, 한국, 대만, 홍콩 등 일본을 제외한 극동아시아의 유교, 한자문화권에 해당한다. 그리고 권역 2에는 중남아메리카, 동남아시아와 남아시아 국가, 아프리카와 동유럽 국가들이 해당한다. 이 권역에 해당하는 국가는 주로 비영어권 국가로, 이슬람 및 카톨릭 국가를 포함하며, 다른 권역의 국가에 비해 경제적으로 덜 발전된 국가들이 권역2에 포함되어 있다. 마지막으로 권역3은 북미의 국가들, 서유럽과 북유럽 국가들, 그리고 호주와 일본을 포함한다. 문화적 측면에서는 주로 북미, 호주, 유럽의 기독교 국가들, 호주, 영국을 포함한 영어사용권 국가들이 포함되며, 경제적 측면에서 이 권역의 국가들은 일본과 서유럽, 북유럽의 선진국들이다.

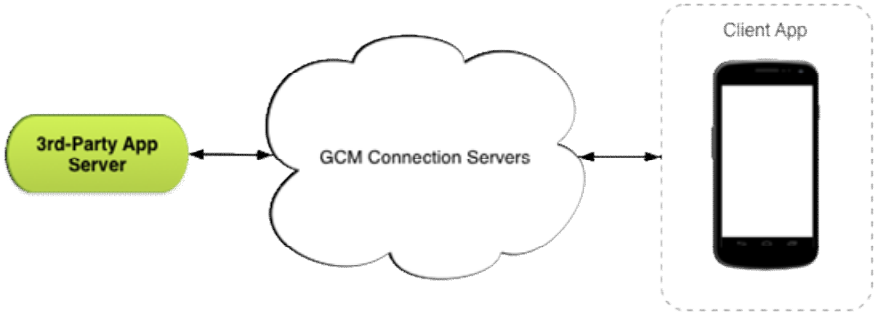
이러한 특성은 스마트폰의 일상화로 인해 나타나게 된 것이다. 모바일 어플리케이션은 스마트폰의 편재성(ubiquity)에 기반하고 있다.[54] 즉 스마트폰은 시간과 장소에 관계없이 접속성(connectivity)과 향상된 계산력(computing power)을 제공한다. 사용이 시공간적으로 한정되어 있는 개인용 컴퓨터와는 달리, 편재적인 접속성과 계산력을 통해 스마트폰 사용자는 일상적 시공간에서 정보기기를 이용할 수 있게 된 것이다. 이러한 일상화로 인해 스마트폰의 이용 방식 또한 사용자의 지리적, 사회적, 문화적 맥락과 더 많은 연관을 가지게 되었으며, 기술거리와 권역의 분포는 이러한 추세를 반영하고 있다.

54) 이동성은 물리적, 신체적 이동성(physical mobility)과 인터넷을 통한 가상적 이동성(virtual mobility)으로 나눌 수 있다(Kellerman, 2006). 여기서 말하는 편재성은 물리적 이동성의 차원으로서, 스마트폰을 통해 사용자는 물리적 이동성과 가상적 이동성을 동시에 누릴 수 있게 되었다.

2. 구글의 정보하부구조에 대한 기술적 의존

(1) 기술적 공동화

개발자가 모바일 어플리케이션을 운영, 개선하기 위해서는 사용자와의 상호작용이 지속되어야 하며, 이를 위해 사용자가 어플리케이션을 이용하지 않을 때에도 상호작용을 유지할 수 있는 채널이 확보되어야 한다. 특히 어플리케이션을 계속 사용하게 하려면 개선된 기능과 업데이트된 정보를 사용자에게 지속적으로 '방송'할 수 있는 채널을 확보해야 한다. 기존 웹 기반 서비스에서는 사용자가 직접 입력한 정보에 따라 우편, 전화, 문자, 이메일을 보내는 방식으로 이러한 채널을 확보할 수 있었다. 반면 안드로이드 어플리케이션은 대부분 구글의 유통망에 의해 확산되므로, 개발자가 사용자를 자체적으로 식별하기 어려워졌다. 만약 모바일 어플리케이션이 사용자에 대한 개별적인 가입 절차를 포함하고 있거나 사용자의 개인정보 유지하고 있다면 별도의 상호작용 채널을 만들 수 있겠지만, 제약된 환경에서 간단한 기능을 수행하려는 목적을 지닌 어플리케이션들은 대부분은 가입, 인증과 같은 별도의 식별 절차를 가지고 있지 않다. 또한 한 사람이 여러 가지 어플리케이션을 보유하게 됨에 따라 한 어플리케이션을 사용하는 시간도 길지 않다.[55] 따라서 개발자들은 사용자에 대한 정보를 별도의



<그림 14> 개발자와 사용자 간 소통 채널의 확보: 구글이 제공하는 소통 채널인 구글클라우드메시징의 소개
(출처: http://developer.android.com/google/gcm/gcm.html)

채널에 의지할 수밖에 없으며, 이러한 채널을 이용하기 위해 특정 코드를 모바일 어플리케이션에 삽입한다. <그림 14>은 이러한 채널 중 가장 대표적인 구글클라우드메시징(Google Cloud Messaging, GCM)을 설명하는 것으로써, 채널을 통한 개발자-사용자 간 상호작용 방식을 개략적으로 나타내고 있다.

또한 일부 모바일 어플리케이션들은 기능과 정보를 제공하기 위해 사용자의 지역적 특성을 이용한다. 이를 위해서는 모바일 어플리케이션이 사용자 위치 관련 정보를 처리하는 코드를 포함하고 있어야 한다. 예를 들어 개인의 위치한 지역의 정보를 얻으려면 위성을 이용해 측정한 '수치'를 실제 지형, 도로와 건물의 위치, 행정 구역, 식당과 영화관의 위치를 기준으로 비교, 분류하여야 한다. 하지만 충분한 자본, 인력, 시간의 여유가 있지 않은 이상, 비교, 분류의 기준이 되는 지리정보나 지역데이터를 직접 수집하기는 어렵다[56]. 더군다나 코드로 구현되더라도 지역에 대한 충분한 정보가 확보되지 않으면 제대로 된 서비스가 제공되기 어렵다. 따라서 대다수의 어플리케이션 개발자는 지역정보, 개인정보를 처리하기 위해 다른 개발자의 코드에 의존하게 되었다.

외부 행위자와 모바일 어플리케이션을 연결하는 이러한 구조는 개발자가 특정한 코드를 모바일 어플리케이션에 삽입함으로서 완성된다. '외부코드'라고 칭할 수 있는 이 코드는 모바일 어플리케이션의 개발자가 아닌 외부의 행위자에 의해서 만들어진 코드로, 개발자는 특수한 작업을 수행하기 위해서 이러한 미리 만들어진 외부코드를 모바일 어플리케이션에 삽입한다.[57]

2장에서 살펴보았듯 안드로이드 운영체제가 표방하는 개방성은 많은 사람들의 자발적 참여에 기반하는 것이며, 안드로이드 운영체제의 생산과 배

55) 안드로이드 운영체제와 iOS에서 사용되는 모바일 어플리케이션에 관한 한 조사에 따르면, 2013년 3분기 전 세계 사용자들은 평균 26.8 개의 어플리케이션에 30시간 15분을 사용하였다. 이에 따르면 사용자가 하나의 어플리케이션에 평균적으로 소비하는 시간은 하루 2.3분에 불과하다. (Nielsen, 2014.7.1)

56) 모바일 어플리케이션과 개발자의 증가에 따라 이러한 작업을 처리할 수 있는 시간과 자원을 가진 개발자의 비율은 작아질 수밖에 없다.

57) 컴퓨터 공학 분야에서는 이러한 코드를 라이브러리(library) 또는 API(Application Programing Interface)라고 칭한다.

포 과정에서 특정 행위자가 직접적인 이익을 얻고 있을 경우 다른 행위자의 자발적 참여를 유도할 명분 자체가 사라지게 된다.[58] 또한 안드로이드 운영체제의 생산 구조는 외부의 기업이나 단체가 자신의 사적 이익을 위해 만든 기술(예를 들어 특정 카드사의 결제 시스템이나 특정 사업자의 지역 정보 서비스)가 운영체제 에 직접 통합되는 것을 막는다. 즉 안드로이드 운영체제 생산의 개방성은 특정 행위자의 이익을 대변하지 않는 중립성과 연관된다. 안드로이드 운영체제는 기본적으로 기기 식별체계, 금융망, 광고망 등과 같은 정보하부구조를 제공하지 않는다. 또한 '지도' 프로그램과 같이 안드로이드 운영체제를 구성하는 기본 구조도 이해 관계자의 부재와 사유화된 구조의 등장에 따라 점차 도태되고 있다.

이와 같이 정보기술체계의 개방성은 공적 정보하부구조의 부재로 연결된다.[59] 특히 안드로이드 어플리케이션의 다양화로 인해 필요성이 증대되는 정보하부구조들이 안드로이드 운영체제에 기본적으로 내재되기는 어려우며, 이러한 정보하부구조가 존재하더라도 외부의 하부구조에 비해 미약한 정보와 기능을 가지게 된다. 즉 개방성으로 인한 기술적 공동화(空洞化)로 인해, 다양화, 맥락화된 안드로이드 어플리케이션은 기업의 사적 하부구조에 의존하게 된다.

---

58) 안드로이드 체계에 대한 대부분의 비판은, '미시적 관찰'에 근거하여 개방성을 표방하는 안드로이드 운영체제를 통해 사적 이익을 추구하는 구글의 이율배반성을 공격하는 것이다. 하지만 구글이 항상 개방성에 배반되는 방식으로 행위하는 것은 아니며, 하위 체계의 미시적 수준에서 기술적 공동화는 구글에 의해 의도된 것이라기보다 기술적 개방성에 의해 발생한 현상이라고 보는 것이 맞다. 하지만 참여와 공유의 개방성과 통제와 의존의 폐쇄성이 양립하고 있는 기술체계의 거시적 관점에, 이러한 미시 현상 또한 체계건설자의 목적 달성을 위해 설계된 메커니즘의 한 부분이라고 할 수 있다.

59) 여기서의 정보하부구조(information infrastructure)란 모바일 어플리케이션의 동작에 필요한 데이터가 유통되는 정보네트워크를 말한다. 모바일 어플리케이션이 동작하기 위한 하부구조로 동작하는 구글의 기준 체계들에서 스타와 룰레더가 제시한 하부구조의 특징들을 관찰할 수 있다(Star & Ruhleder, 1996; Bowker & Star, 2000). 이러한 기준들은 데이터의 조합 속에 존재하며(배태성), 매번 새로이 만들어질 필요가 없으며(투명성), 특정 이벤트나 장소에만 한정된 것이 아니다. 또한 이러한 기준들은 실행공동체(community of practice)와 관련이 있다. 실행공동체는 실행에 대한 전문성, 관심 등을 공유하고 더 나은 방법을 학습하기 위해 상호작용하는 집단인데, 이러한 기준들은 실행공동체 멤버십의 일부로서 학습되고, 공동체의 실행의 관습에 의해 형성되거나 관습에 영향을 준다. (Lave & Wenger, 1991; Wenger, 1998)

(2) 구글에 대한 기술적 의존

1) 사적 정보하부구조의 유형과 비중

모바일 어플리케이션에서 사용되는 정보하부구조의 유형을 살펴보면 구글의 영향력이 확연히 드러난다. 미국 시장에 등록된 어플리케이션 중 많이 사용되는 상위 10개의 외부코드(API) 중 4개가 구글에 의해서 만들어진 것이다(<표 10> 참조). 더군다나 이 순위에서 가장 많이 사용되는 것으로 조사된 '안드로이드 지원 라이브러리'는 AOSP에 의해 무료로 배포되는 기본적인 코드란 점을 고려해 볼 때, 사실상 개발자가 사용하는 외부코드의 대부분은 구글의 것이다. 이 중 구글클라우드메시징은 개발자와 사용자 간

API 이름	API가 포함된 어플리케이션 비율	개발자	라이선스	기능
안드로이드 지원 라이브러리 (Android Support Library)	41.68	AOSP	오픈소스	개발 도구 제공
구글 클라우드 메세징 (Google Cloude Messaging)	11.81	구글	상용	서버-기기 간 통신
구글 애널리틱스 (Google Analytics)	12.34	구글	상용	어플 사용 통계
플러리 애널리틱스 (Flurry Analytics)	7.28	플 러 리 (Flurry)	상용	어플 사용 통계
구글 인앱 구매 (Google In-app Billing)	4.03	구글	상용	어플 내 상품 구매
구글 쥐슨 (Google gson)	6.85	구글	상용	데이터 처리
Actionbar Shaerlock	7.25	개인	오픈소스	어플 디자인 개선
Apache HttpMime API	5.73	개인	오픈소스	웹 서버 통신
아마존 인앱 구매 (Amazon in-app purchasing)	1.47	아마존	상용	어플 내 상품 구매
FMOD Ex Programmers API	2.77	FMOD	상용	음향 기능 확장

<표 12> 상위 10개 개발도구 API의 점유율
(앱브레인(Appbrain) 조사자료. 2013년 3월 25일 기준)

의 소통 채널이며, 구글인앱구매는 사용자가 모바일 어플리케이션을 사용하며 상품을 구매할 수 있도록 하는 정보하부구조이다. 또한 개발자는 구글 애널리틱스라는 코드를 어플리케이션에 포함하여 구글로부터 사용자들의 어플리케이션 사용방식, 사용통계 등의 분석 자료를 받아볼 수 있다. (이와 같이 구글에 의해 만들어져 모바일 어플리케이션에서 사용되는 외부 코드를 '구글코드'로 칭하도록 한다.)

개발자는 광고 API를 어플리케이션에 포함하여 사용자에게 최적화된 광고를 표시하고 수익을 얻을 수 있다. 이러한 광고 API 부문에서도 구글코드가 가장 많이 활용되고 있는 것을 알 수 있다. 구글의 자회사인 애드몹(Admob)의 API는 미국 시장에서 출시된 전체 어플리케이션 중 37.7%에 포함되어 있으며 이는 다른 광고 API과 비교했을 때 압도적인 비율이다.[60] (<표 13>)

API 이름	API가 포함된 어플리케이션 비율 (%)
Admob	**37.66**
Millenial Media	3.75
InMobi	3.11
Chartboost	2.25
Tapjoy	1.53

<표 13> 상위 5개 광고 API의 점유율
(앱브레인(Appbrain) 조사자료. 2013년 3월 25일 기준)

전 세계 각 국의 인기 어플리케이션을 대상으로 한 분석에서도 구글에 대한 의존은 확연하게 드러난다.[61] 전체 어플리케이션의 57%는 4가지 중 적어도 한 가지 이상의 구글코드와 관련되어 있다. 이 4종류의 코드는 어플리케이션의 외부 관찰을 통해 그 포함여부를 확인할 수 있는 것이며 내부에는 외부에 드러나지 않은 구글코드가 더 포함되어 있을 수도 있다는

60) 구글은 2009년 11월 9일 모바일 광고회사인 애드몹을 7억 5천만 달러에 인수하였다. 애드몹은 2014년 전 세계에서 44억 달러의 수익을 거둬들일 것으로 예상되고 있다. (Shalvey, 2014.4.23.)

61) 2013년 10월 20일 전 세계 35개국, 각 국 당 50 개씩 선정된 인기 어플리케이션 총 748개 중 663개를 대상으로, 각 어플리케이션이 <표 14>에서 제시된 4종류의 구글의 코드를 사용하고 있는지 확인하였다. 대상 어플리케이션의 이름과 분석 결과는 부록의 <표 27>에 제시되어 있다.

코드 유형	코드를 포함하고 있는 어플리케이션 비율
구글 지도 (Google Map API)	16.9%
구글 클라우드 메세징 (Google Cloude Messaging)	41.0%
구글 인앱 구매 (Google In-app Billing)	17.9%
구글 서비스 관련	14.0%
위 4가지 중 하나 이상 포함	57.0%

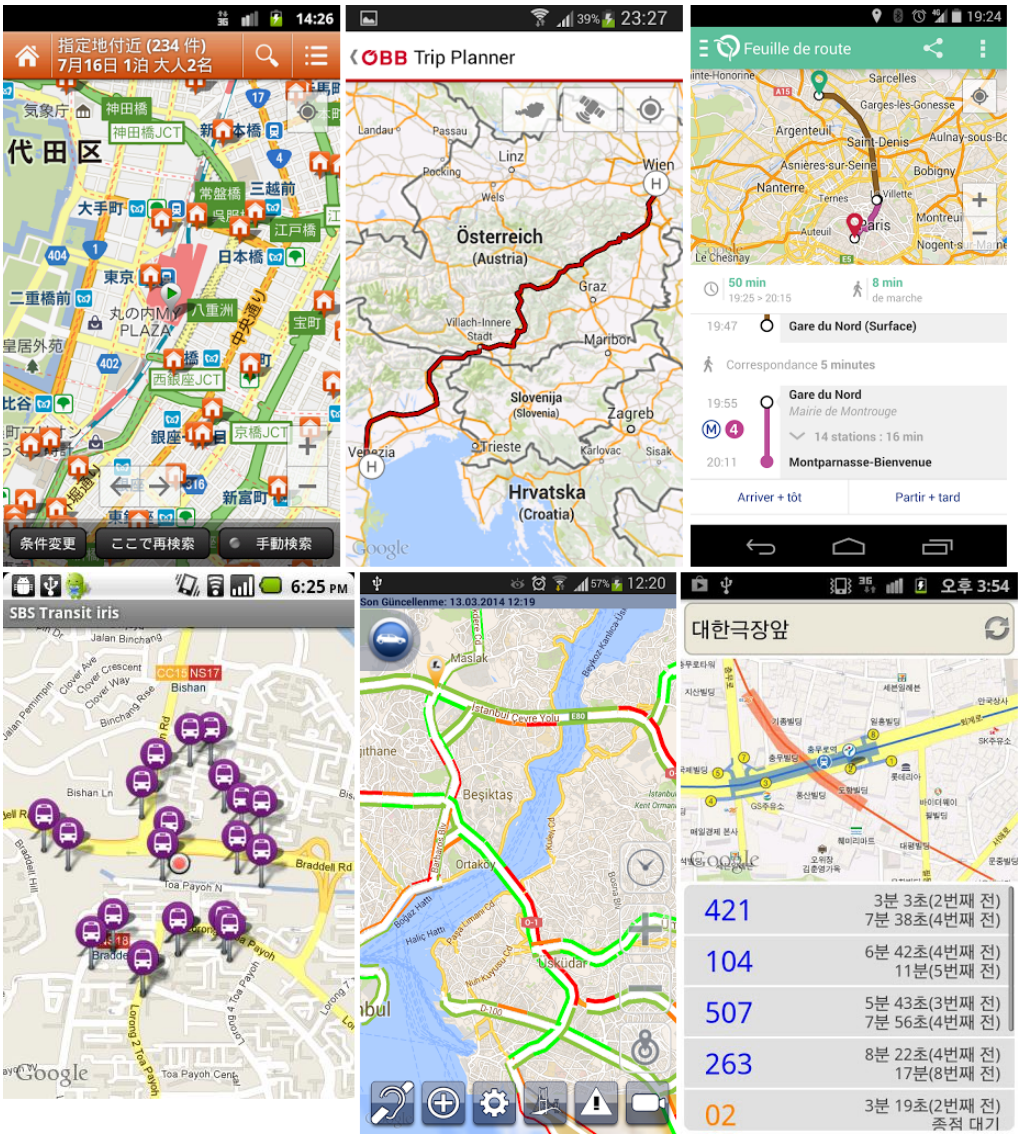
<표 15> 구글코드를 포함하고 있는 어플리케이션의 비율

점에서, 실제 비율은 이 보다 훨씬 클 가능성이 매우 높다.

특히 전체의 41.0%에 달하는 어플리케이션은 구글의 클라우드 메시지 기능을 사용하고 있는 것으로 확인되었다. 또한 정보를 지도상에 표시하게 하는 구글 지도의 코드 또한 21.8%의 어플리케이션에서 사용되고 있다. 하지만 지역정보와 직접적으로 관련된 어플리케이션 분류에서 이 비율은 더 높아진다. 예를 들어 34개의 교통, 여행 관련 어플리케이션 중 73.5%에 해당하는 25개의 어플리케이션이 구글지도를 사용하고 있다. <그림 15>에서와 같이 어플리케이션이 사용되는 지역, 어플리케이션에서 제공하는 정보가 다를 지라도 동일한 코드가 사용되고 있는 것이다. 또한 전 세계 금융

코드 유형	코드의 기능	확인 사항
구글 맵 (Google Map API)	지도상에 사용자 위치, 지역 정보, 경로나 거리 표시	- 해당 권한 포함 여부 :MAP_RECEIVE - 모바일 어플리케이션 내 구글지도 기능 포함 여부
구글 클라우드 메세징 (Google Cloude Messaging)	개발자가 모바일 어플리케이션이 설치 기기에 메시지 전송	해당 권한 포함 여부 : C2D_MESSAGE
구글 인앱 빌링 (Google In-app Billing)	모바일 어플리케이션 내에서 상품 구입, 결재	해당 권한 포함 여부 : BILLING
구글 서비스 관련	구글 계정 정보 사용 구글 서비스 사용	해당 권한 포함 여부 : READ_GSERVICE, CHECK_LICENSE

<표 14> 구글에 의해 만들어진 코드의 기능과 확인 방법



<그림 15> 구글 지도를 사용하는 교통, 여행 관련 어플리케이션의 예: じゃらん (일본), öBB Scotty (오스트리아), RATP : Subway Paris (프랑스), SBS Transit iris(싱가포르), İBB CepTrafik (터키), 지하철 종결자 : Smarter Subway(한국). (상단 왼편부터 오른쪽으로. 괄호 안은 사용 국가)

회사 어플리케이션 25개 중 18개가 구글지도를 사용하고 있다. 이러한 모바일 어플리케이션에서 구글지도는 주로 사용자 주변에 있는 금융사 지점을 지도에 표시하기 위한 용도로 사용되고 있다.

2) 구글의 정보하부구조의 유형과 의존 요인

구글이 제공하는 하부구조들을 좀 더 구체적으로 확인해 보면 다음과 같다. 첫째, 사용자의 위치를 인식하기 위해서는 위성 장비를 통해서 측정된 수치가 지도상에 투영되어 행정구역을 기준으로 분류되고 다른 지형지물, 다른 사용자의 위치와 비교되어야 특정 사용자의 '위치'가 인식될 수 있다. 따라서 이러한 지도 체계는 사용자의 위치를 인식하기 위한 정보하부구조이다.

둘째, 사용자는 여러 종류의 모바일 어플리케이션을 이용하는데, 특정 어플리케이션이 스마트폰에서 실행되고 있을 경우 개발자는 어플리케이션을 통해 해당 사용자와 정보를 주고받을 수 있지만, 사용자가 특정 어플리케이션을 사용하지 않고 있을 경우 개발자가 사용자에게 접근할 방법은 없다. 이러한 소통의 일방성을 극복하기 위해서는 어플리케이션의 개발자가 사용자를 파악하고 접근할 수 있도록 하는 체계가 필요하다. 이러한 체계는 어플리케이션이 설치된 스마트폰에 일련의 전자적 식별번호를 부여하고 이를 통해 개발자가 원하는 시점에 사용자와 소통할 수 있게 한다. 이러한 식별체계와 소통채널은 개발자에서 사용자로의 소통을 가능케 하는 정보하부구조이다.

셋째, 사용자가 모바일 어플리케이션을 사용하며 금전적 거래를 할 수 있도록 하려면 개발자는 사용자의 카드번호, 계좌번호, 전화번호 등을 카드사, 통신사, 은행 등에 알려주고 이들로부터 자금을 받아야 한다. 이를 위해서는 금융사, 통신사들과 어플리케이션 간의 연결망이 구축되어 있어야 하며, 이러한 연결망을 통해 사용자의 경제적 조건들이 식별되고 거래가 이루어지게 된다. 따라서 이러한 금융망은 사용자의 경제적 특성을 인식하고 금전적 거래를 가능하게 하는 정보하부구조이다.

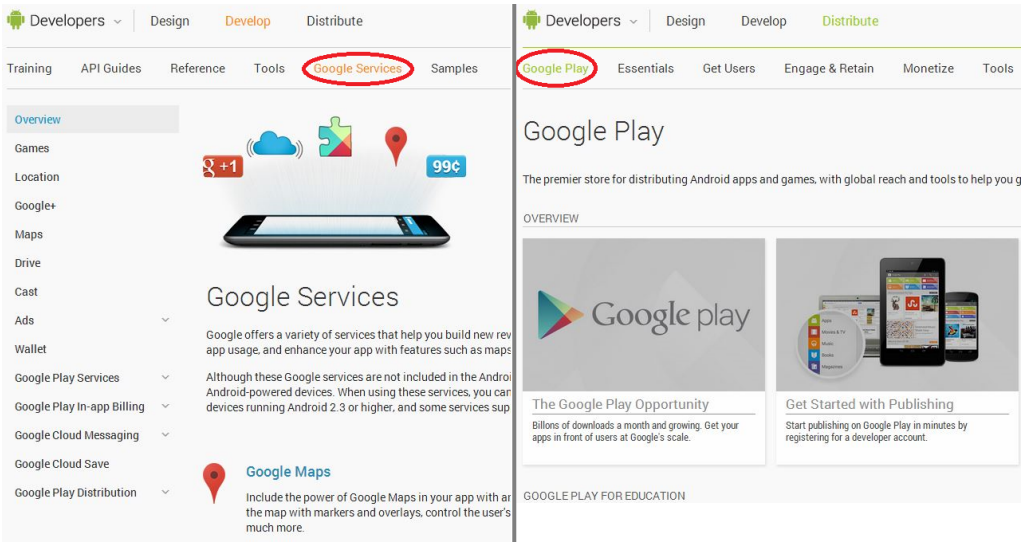
넷째, 많은 수의 모바일 어플리케이션은 광고를 통해 수익을 얻는다. 특히 어플리케이션의 대부분을 차지하고 있는 무료 어플리케이션의 경우, 어플리케이션 자체의 판매를 통해 수익을 얻을 수 없기 때문에 다른 기업의 광고를 개제하고 이로부터 광고료를 받는다. 하지만 개발자들이 다양한 광

고주들을 직접 접촉하기는 어려우며, 특히 개인 개발자들이 개발과 함께 광고 판매, 판촉 활동을 하는 것은 더욱 어렵다. 따라서 개발자들은 미리 구축된 광고망(Ad Network)가 필요하다. 광고망은 스마트폰에 광고하기 원하는 광고주들을 확보해 놓고 있으며, 개발자들은 자신의 어플리케이션을 광고망에 접속시켜 이들의 광고를 표시하고 수수료를 받을 수 있다. 또한 광고망은 위치 등의 정보를 이용하여 사용자와 연관성 있는 광고를 표시하도록 한다. 따라서 광고망 또한 사용자 특성을 파악하고 경제정보의 흐름을 가능하게 하는 정보하부구조 이다.

개발자들은 외부의 하부구조 중에서도 구글의 하부구조, 구글의 코드에 의존한다. 그 이유는 몇 가지로 나누어볼 수 있다. 우선 개발자들은 구글의 코드를 포함시킴으로써 구글의 발달된 기술을 어플리케이션에 결합시킬 수 있다. 지도, 사용자 분석, 광고 기술은 구글이 안드로이드 운영체제 출시 전부터 보유하고 있던 기술이다. 뿐만 아니라 구글은 지도, 광고, 정보 분석 분야에서 우월한 시장 위치를 점유하며 많은 기술력과 정보를 축적했다. 예를 들어, 구글 지도(Google Maps)는 전 세계를 대상으로 한 인터넷 지도 서비스이며, 2008년 가장 많은 네티즌이 방문한 인터넷 지도 서비스 중 하나이다(CNET, 2009). 또한 구글은 미국에서 가장 큰 검색 사업자로서 2008년도 미국 전체 검색의 약 70%를 차지하였는데 (CNET, 2009), 수익의 대부분을 검색 과정에서 개인정보를 측정하고 이를 이용한 맞춤형 광고를 판매해서 얻었다. 따라서 모바일 어플리케이션에서 구글 코드의 확산은 이러한 기술적 지배력이 새로운 영역으로 침투, 전파로 인한 것으로 이해할 수 있다.

또한 개발자들은 다른 개발자의 코드 보다 구글코드를 더 쉽게 사용할 수 있다. 구글은 안드로이드 운영체제의 생산을 주도하는 행위자이지만 안드로이드 운영체제가 구글의 소유인 것은 아니다. 하지만 구글의 사적인 코드는 중립적이고 개방적인 운영체제와 매우 긴밀한 연관성을 갖고 있다. 구글의 코드는 운영체제와 어플리케이션의 개발 과정에서 쉽게 이용할 수 있다. 예를 들어, 안드로이드 어플리케이션 개발에 필수적으로 사용되는 프로그램인 안드로이드 개발자 도구(Softward Development Kit)을 통해 구

글의 코드를 손쉽게 다운로드 받을 수 있다. 또한 개발자들이 구글의 코드를 사용하기 위해서 알아야할 내용 또한 안드로이드 운영체제 관련 웹사이트에서 함께 볼 수 있다. <그림 16>은 안드로이드 오픈소스 프로젝트의 홈페이지 곳곳에 구글 관련 정보가 존재한다는 것을 보여주고 있다. 안드로이드 운영체제에 관해 가장 많은 정보를 제공하고 있는 사이트에서 구글 플레이서비스(Google Play Service), 구글인앱구매, 구글클라우드메시징, 구글지도('Location'과 'Maps'), 구글플러스('Google+'), 구글캐스트('Cast') 구글드라이브('Drive'), 애드몬('Ads'), 구글지갑('Wallet') 등 구글의 다양한 기술에 대한 광고와 구체적인 사용법을 제공하고 있는 것이다. 많은 사람에 의해 자유롭게 개발, 사용되어 '공적'인 성격을 띄고 있는 안드로이드 운영체제와 구글의 '사적' 기술 간의 경계가 모호해 짐에 따라, 안드로이드 스마트폰 및 어플리케이션을 만들려고 하는 제조사, 개발자는 의도 하지 않았는데도 관련 정보와 지식을 습득하거나 개발 도구를 사용하는 과정에서부터 구글의 기술에 노출되는 것이다. 이와 같이 '기술적으로 개방적이고 경제적으로는 중립적'으로 여겨지는 안드로이드 '생태계'가 실재적으로는 모든 관련 행위자를 구글의 사유화된 기술로 유인하는 거대한 코드 구조로



<그림 16> AOSP 홈페이지에 포함되어 있는 구글 관련 정보들
(출처: http://developers.android.com)

작용하고 있다.

IV. 스마트폰 기기 영역의 개방성과 지배

1. 스마트폰 기기 영역의 개방성

(1) 안드로이드 운영체제의 라이선스 구조

기술에 대한 권리의 범위와 형태는 라이선스(license)를 통해 규정된다. 특히 정보기술, 그 중에서도 소프트웨어 분야에서는 코드의 사용범위를 규정해 놓은 라이선스를 규정하는 것이 중요하다. 그 이유는 소프트웨어는 하드웨어에 비해 상대적으로 복제, 변형이 쉽다. 예를 들어 하드웨어 복제하기 위해서는 원본과 동일한 재료, 디자인, 생산 기술 등이 필요하다. 반면 단지 자판 몇 개의 조작만으로도 원본과 동일한 소프트웨어를 만들 수 있다.

이러한 라이선스 중에서도 개방성을 보장할 수 있는 것들이 있다. 그 대표적인 예가 자유소프트웨어 라이선스(free software license)와 오픈소스 라이선스(open software license)이다. 우선 자유소프트웨어 라이선스가 적용되는 기술은, 해당 기술의 모든 내부 동작과 기능이 기술되어 있는 소스코드(source code)가 일반에 공개되어 있어 누구나 자유롭게 이를 이용, 수정, 재배포할 수 있다. 예를 들어 한 회사가 자사의 제품에 오픈소스 라이선스를 적용하고 있다면, 다른 기업이 이 소스코드를 그대로 복제하여 동일한 제품을 만들고 이를 판매할 수도 있다. 하지만 이 경우 복제한 회사는 이 사실을 일반에 알리고 자사 제품의 소스코드를 공개하여야 한다. 이와 같이 자유소프트웨어 라이선스가 적용되는 기술은 판매 수익을 얻기 어렵기 때문에, 많은 경우 다수의 개발자 또는 회사 간의 협업을 통해 개발된다(Vainio. N. and T. Vadén, 2007). 오픈소스 소프트웨어는 자유소프트웨어와 마찬가지로 소스코드가 공개되어 있으나, 자유소프트웨어와는 다르게 복제나 차용을 통해 새로 개발한 기술의 소스코드를 공개할 의무가 없다.

안드로이드 운영체제에서 리눅스 커널은 자유소프트웨어 라이선스인

GPL(General Public License)을 따른다. GPL이 적용되는 기술의 저작권은 모두 자유소프트웨어 재단(Free Software Foundation, FSF)이 가진다. 안드로이드 운영체제의 리눅스 커널은 'GPL version 2'를 따르며, 이 라이선스는 제3자에 의한 코드의 수정, 재배포를 허용하지만[62] 제3자에게 모든 소스코드를 공개해할 의무를 부여한다[63](GNU, 1991). 그리고 코드 공개 여부는 자유소프트웨어 재단에 의해 상시 감시된다. 이와 같이 GPL은 저작자의 독점적 권리를 부정하고 사회적인 공유와 협력을 강조하며, GPL을 따르는 기술은 생산자의 권력을 확장시키기보다 공유라는 목표를 추구하는 자유소프트웨어 재단의 권력을 확장시키게 된다.

이에 반해 안드로이드 운영체제에서 커널을 제외한 나머지 부분은 오픈소스 라이선스인 아파치라이선스(Apache Software License, ASL)를 따른다.[64] 아파치라이선스가 적용된 기술의 소스코드는 누구나 자유롭게 수정, 배포할 수 있다. 이와 같이 누구나 무료로 사용하고 변형할 수 있다는 점, 그리고 그것의 '재료'이자 '구조'라고 할 수 있는 코드를 제공한다는 점에서 안드로이드 운영체제는 일반적으로 거래되는 상품과 달리 '탈상품화' 되어 있는 것처럼 보인다.

> 안드로이드 플랫폼[운영체제]는 가장 진보적이고, 개발자 친화적인 오픈소스 라이선스 중 하나로 공개될 것이며, 오픈소스 라이선스는 이동통신 사업자와 기기 제조사가 제품을 디자인함에 있어 상당한 자유와 유연성을 줄 것이다. [중략] 제조사와 이동통신 사업자는 혁신적인 새 제품을 더욱 빠르고 저렴하게 출시하기 위해 안드로이드를 자유롭게 변경(customize)할 수 있다.
>
> 개방핸드셋연합 발표문 (OHA, 2007.11.5.)

---

62) 코드의 수정 가능(you can change the software or use pieces of it in new free programs; and that you know you can do these things.), 재배포의 자유(the freedom to distribute copies of free software)가 허용되고 있다. (GNU, 1991)

63) GPL은 자유소프트웨어 재단(Free Software Foundation)에 의해 처음 제안되었다. FSF는 필요에 따라 GPL을 개정할 뿐만 아니라 다른 기업들의 라이선스 준수 여부를 감시한다.

64) 안드로이드 운영체제는 '아파치 라이선스 2.0'을 따르고 있다.

하지만 아파치라이선스는 GPL과는 달리 제3자의 코드 공개를 의무로 하지 않는다. 구글은 아파치라이선스를 선택한 이유에 대해서 다음과 같이 밝히고 있다. 구글에 따르면 안드로이드 운영체제를 사용하여 개발된 모든 소프트웨어의 소스코드를 공개하도록 하는 것은 사용자에게 특정한 방식을 '강요'하는 것으로써, 이는 '자유와 선택'이라는 안드로이드의 원칙에 위배된다.[65]

> 왜 아파치 소프트웨어 라이선스인가?
> [중략] 안드로이드는 자유와 선택에 관한 것이다. 안드로이드의 목표는 모바일 세상에서 개방성을 확장시키는 것이고, 우리 소프트웨어의 모든 사용을 예측하거나 지배하는 것이 가능하다고 생각하지 않는다. 따라서 우리는 모두가 개방적이고 수정 가능한 기기를 만들 것을 권장하지만, 그렇게 하라고 강요하는 것이 우리의 역할이라고 생각하지 않는다. LGPL[66]의 사용은 종종 사람들에게 그러한 것들을 강요하게 될 것이다. [후략]
> (출처: 안드로이드 홈페이지. http://source.android.com/source/licenses.html, 2014년 6월 16일 방문)

(2) 기술의 불투명성과 체계의 확장

개별적인 스마트폰 기기의 특성에 맞추어 안드로이드 운영체제를 수정, 보완하는 것이 스마트폰 기기 제조사의 임무이기 때문에, 제조사는 안드로이드 운영체제의 라이선스 선택은 가장 큰 영향을 받는다. 구글은 오픈소스 라이선스를 선택한 이유가 주로 제조사에 대한 고려 때문임을 밝히고 있다[67]. 하지만 안드로이드 스마트폰을 만드는 제조사나 안드로이드에서 동작하는 어플리케이션을 만드는 개발자가 오픈소스 혹은 자유소프트웨어 라이선스를 선택하는 비율은 매우 적다. 우선 (구글이 직접 개발한 플래그

65) 이에 반해 자유소프트웨어 진영에서는 안드로이드 운영체제가 오픈소스 라이선스를 선택함으로써, 공유와 협력의 가치를 확장할 기회가 사라졌다고 비판한다(Stallman, 2011).
66) LGPL(Lesser General Public License)는 자유소프트웨어 라이선스의 일종이다. 이 라이선스에서는 GPL(General Public License)의 강력한 코드 공유 조항들이 다소 약화되었다.
67) http://source.android.com/source/licenses.html

십 스마트폰인 넥서스(Nexus) 시리즈를 제외하고) 안드로이드 스마트폰에 탑재된 운영체제의 코드가 공개된 사례는 없다. 또한 소스코드를 공개하고 있는 어플리케이션의 비중도 매우 작다.[68]

구글 또한 코드공개로 인해 개발자, 제조사가 갖게되는 문제점들을 강조하고 있다. 구글은 자유소프트웨어를 선택할 경우 기기개발자가 코드 공개, 제3사에 대한 권한 허용, 저작권 분쟁으로 인해 기기개발자에게 제약이나 부담을 안겨줄 수 있다고 밝히고 있다.[69] 코드 공개에 따라 개발에 동원된 지식이 쉽게 노출될 수 있으며, 이로 인해 성능이나 도용 여부 등에 대한 문제제기 가능성도 높아질 수 있다. 또한 코드를 수정하려는 사람들에게 적절한 수준의 권한을 부여하면서 제조사 자신의 권익도 지켜야 하는 새로운 문제가 발생한다. 즉 코드를 보거나 사용하게 됨에 따라 다른 행위자들을 제조사에 영향력을 미칠 수 있게 된다. 오픈소스 라이선스는 코드를 숨기고 제조사들을 이러한 '침범'으로 부터 보호한다.

이와 같이 구글이 '강요'를 피하기 위해 제시했다는 선택의 기회는 사실상 명목적인 것이다. 제조사, 개발자는 쉽게 새로운 기술을 만들고 이에 대한 독점적인 정보와 권한을 소유할 수 있다는 점에 주목하며, 구글 또한 이러한 점을 강조하고 있다. 기술에 대한 독점적 소유는 특히 기술을 통해 경제적 이익을 얻으려는 행위자들에게 매력적이기 때문에, 오픈소스 라이선스는 안드로이드 운영체제를 탑재하여 판매하려는 제조사의 수를 늘리고 안드로이드 스마트폰 기기의 '생태계'를 확대하는 데 중요하게 작용한다(Paul, 2007.11.7.). 하지만 이러한 '정해진 응답'에도 불구하고 선택지의 제공 자체가 무의미한 것은 아니다. 선택의 기회를 제공함으로써 기술에 대한 정치적, 사회적 책임은 개별적 행위자 선택의 문제로 전환되는 반면 구글은 개방성의 모토는 단지 기술 수준에 한정하여 이러한 책임으로부터 멀어질 수 있게 되었기 때문이다. 그리고 두 가지 다른 라이선스의 '결합'으

---

68) 위키피디아(wikipedia)에 현재 소스코드를 공개하고 있는 것으로 조사된 안드로이드 어플리케이션은 총 38개로, 수십만개에 달하는 안드로이드 어플리케이션 중 그 비율은 매우 미미하다.
http://en.wikipedia.org/wiki/List_of_free_and_open-source_Android_applications

69) http://source.android.com/source/licenses.html 참조.

로 인해, 자유소프트웨어 라이선스를 따르는 기술은 사회적 공유를 목표로 함에도 불구하고 오히려 오픈소스 라이선스에 기반한 기술들을 보완, 강화하여 기술의 사유화로 인한 이득을 더욱 증대시킨다.

이렇듯 안드로이드 운영체제의 오픈소스 라이선스는 운영체제를 이용하는 제조사, 개발자의 권리를 보장하는 장치로 동작하고 있다. 자유소프트웨어 라이선스 하에서 행위자는 기술에 대한 배타적인 권리를 행사할 수 없는 '투명한 행위자' 였다면, 오픈소스 라이선스 하의 행위자들은 기술에 대한 배타적인 권리와 지식을 행사하는 '불투명한 행위자'이다. 그리고 역설적이게도 체계의 확장은 개방적 구조의 폐쇄성과 불투명성에 기인한다.

오픈소스 라이선스는 안드로이드 체계를 확장시키는 장치이다. 오픈소스 라이선스는 안드로이드 운영체제를 확산시키려는 행위자와 모바일 운영체제를 사용하려는 행위자에게 모두 유리하게 긍정적으로 작용하였다. 우선 안드로이드 운영체제가 기존 기술체계들과 경쟁하기 위해서는 개방성을 통한 기술의 빠른 확산이 요구되었다. 2003년 이후 매년 2배에 가까운 판매량 성장을 기록한 블랙베리(Blackberry)[70]와 2007년에 출시되어 함께 폭발적인 판매량을 보여준 애플(Apple Inc.) 아이폰(iPhone)은 스마트폰 시장의 성장 가능성을 보여주었다. 구글은 미국 검색시장에서 가장 높은 점유율을 확보하고 있었으나[71], 이러한 새로운 가능성에 주목하여 신규 진출하는 스마트폰 분야에서도 높은 점유율을 확보하고자 했다. 그리고 스마트폰 분야가 급속하게 확장되고 있는 만큼 제품과 기술의 개발, 확산은 지체 없이 빠른 속도로 이루어져야 했다. 2005년 미국증권거래소에 제출한 기업실적 보고서에서 구글은 다음과 같이 밝혔다.

> 만약 PC 외 통신 기기들에서 동작하는 제품과 기술의 개발이 지체된다면, 우리는 온라인 서비스 시장에서 갈수록 중요해지는 부문에서 높은 점유율

70)블랙베리의 개발사인 RIM(Research In Motion) 사의 경영실적 발표에 의하면 전 세계 블랙베리 가입자 수는 2003년 3월 534,000 명, 2004년 2월 1,069,000 명, 2005년 2월 2,510,000 명, 2006년 3월 4,900,000 명으로 증가하였다

71) 시장조사업체인 컴스코어(ComScore, 2006.6.18)에 의하면, 구글은 온라인 검색에서 2005년 6월 36.9%, 2006년 6월 44.2.%의 점유율을 확보해 야후(Yahoo), 마이크로소프트 등을 제치고 1위를 차지하였다.

을 차지할 수 없을 것이다. (Google, 2005)

또한 오픈소스 라이선스는 안드로이드 운영체제를 사용하려는 행위자들에게도 유리하게 작용했다. 우선 스마트폰 기기 제조사가 안드로이드 운영체제를 기기에 설치하면, 운영체제의 구입, 사용을 위한 로열티를 절약할 수 있으며, 운영체제의 직접 개발로 인한 부담도 덜 수 있다. 특히 스마트폰의 발전 초기에 제조사는 모바일 운영체제라는 새로운 기술을 필요로 했지만 하지만 이들의 선택지는 한정되어 있었다. 이 시기 대표적 모바일 운영체제에는 iOS, 심비안(Symbian)[72], 윈도우모바일(Windows Mobile)[73], 블랙베리운영체제가 있었다(Gartner, 2009). 이 중 iOS와 블랙베리운영체제는 애플의 아이폰과 블랙베리[74]라는 휴대폰에만 탑재되는 운영체제였으며, 심비안이나 윈도우 모바일을 기기에 탑재할 경우 제조사는 각각 심비안(Symbian Ltd.)이나 마이크로소프트(Microsoft)에 로열티를 지불해야 했다.[75] 반면 제조사가 자신의 스마트폰에 알맞은 운영체제를 직접 개발할 경우 상당한 자금과 인력을 투자해야할 뿐만 아니라, 새로운 운영체제의 개발과 안정화로 인해 시간이 지체된다면 빠르게 성장하는 시장에 성공적으로 진입하기 어려웠다. 이러한 상황에서 안드로이드 운영체제의 오픈소스 라이선스는 제조사들이 이 운영체제를 선택하게 하는 주요 요인으로 작용할 수 있었다.

또한 오픈소스 라이선스를 통해 안드로이드 운영체제가 다양한 방식으로 변형될 수 있게 됨에 따라 스마트폰 사용자가 갖는 선택의 폭도 넓어질 수 있다. 이와 같이 오픈소스 라이선스는 행위자들이 기술에 쉽게 접근할 수

---

72) 1998년 에릭슨(Ericsson), 모토롤라(Motorola), 노키아(Nokia), 사이언(PSION) 사는 합작회사인 심비안(Symbian Ltd)를 설립하고 동명의 모바일 운영체제를 개발하였다. 2008년 노르웨이의 휴대폰 제조회사인 노키아(Nokia)가 심비안을 인수하고 심비안 재단(Symbian Foundation)을 설립하여 이 운영체제를 오픈소스화 하였다.

73) 마이크로소프트 사가 개발한 모바일 운영체제

74) 캐나다의 통신장비 제조회사인 리서치인모션(Research In Motion, RIM) 사가 개발한 휴대폰.

75) 2009년 심비안 재단(Symbian Foundation)의 출범과 함께 심비안은 오픈 소스인 이클립스 공공 라이선스(Eclipse Public License)로 공개된다. 그러나 심비안의 많은 부분에 더욱 제한된 라이선스인 심비안 공공 라이선스(Symbian Public License)가 부여되었으며 전체 소스 코드는 재단에 소속된 회원사에게만 공개되었다.

있도록 하며, 행위자들은 공식화 이후에도 기술을 다양한 목적과 상황에 맞게 적응(adaptation)시키며 안드로이드 운영체제의 확산과 분화를 촉진시켰다. 안드로이드는 많은 제조사와 개발자들에 의해 사용되게 되었다. 사람들은 다양한 모바일 기기와 어플리케이션에 끌려 안드로이드 운영체제가 탑재된 스마트폰을 사용하고, 사용자의 수가 늘어나면 안드로이드 운영체제에 기반한 모바일 기기와 어플리케이션을 개발하려는 기업과 개발자들이 늘어난다. 이와 같이 기기의 종류, 어플리케이션의 수와 사용자의 수는 서로의 양적 증가를 촉진시킨다.

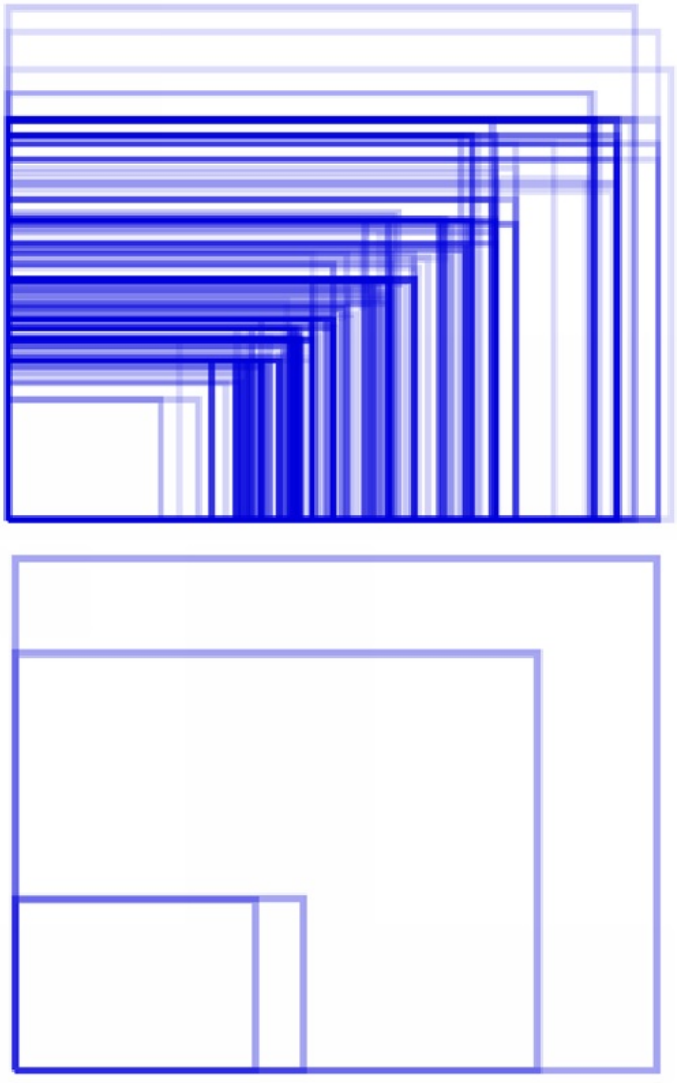
(3) 기술적 요소의 다양화

안드로이드 운영체제를 이용하는 제조사의 수가 늘어남에 따라, 사용되는 스마트폰의 종류도 다양해진다. 예를 들어 2013년 7월 오픈시그널(Opensignal)[76] 사는 전 세계에 분포된 안드로이드 스마트폰 중 자사 모바일 어플리케이션이 설치된 682,000대의 기기 정보를 수집하였다.[77] 이 조사의 결과, 확인된 스마트폰 기기의 종류는 총 11,868 가지에 달하며, 2012년에 약 3,992 종류의 기기가 확인된 것과 비교하여 증가된 수치이다. 또한 확인된 스마트폰 제조사도 119개에 달하며, 스마트폰들에 탑재된 안드로이드 운영체제의 버전은 총 8종류였다. 또한 스마트폰에 탑재된 화면의 크기 또한 다양한데, 이를 애플사의 iOS를 탑재하고 있는 스마트폰들의 스크린

76) 오픈시그널은 지역별 무선통신 품질 수치를 제공하는 것을 전문으로 하는 영국 회사이다.

77) 안드로이드 운영체제를 사용하기 위한 확인, 등록, 계약 등의 별도 절차가 존재하지 않기 때문에, 구글이나 AOSP, 개방핸드셋연합도 누가 어떠한 방식으로 이 운영체제를 사용하고 있는지에 대한 정확한 통계자료를 수집하기 어렵다. 더욱이 만약 어떠한 제조사가 안드로이드 운영체제의 사용 사실과 안드로이드 스마트폰의 제작, 출시 여부를 공개적으로 밝히지 않는다면, 이 제조사가 안드로이드 운영체제를 사용했는지 알 수 있는 방법은 없다. 또한 제조사가 안드로이드 운영체제를 자체적으로 변형시키는 과정에서 해당 운영체제를 '안드로이드 운영체제'로 식별할 수 있는 모든 방법이 제거되었다면, 이 제조사가 만든 스마트폰을 '안드로이드 스마트폰'으로 보기 어렵다. 안드로이드 운영체제의 개방성은 이 운영체제를 탑재하고 있는 기기의 수와 종류를 정확하게 확인하는 것을 어렵게 한다. 하지만 예시로 제시된 자료와 같이 표본 조사나 사례 수집 등 제한적 방식을 통해 조사된 자료만으로도 안드로이드 운영체제가 탑재된 스마트폰이 매우 다양하다 사실을 확인하는 것에는 무리가 없는 것으로 판단된다.

종류 수와 비교하면 그 다양성이 더욱 확연히 드러난다[78](<그림 17> 참



<그림 17> 각 운영체제 별 스마트폰 화면 크기의 상대적인 비교. 위의 그림이 안드로이드 스마트폰의 경우. 아래의 그림은 iOS 탑재 기기의 경우. (Opensignal, 2013)

조). 더욱이 이 조사가 일부 표본만을 대상으로 하고 있기 때문에, 전 세계

78) 오픈시그널 사는 341만 명의 사용자로부터 스마트폰 화면 크기에 대한 정보를 수집했다고 밝히고 있다. (Opensignal, 2013)

스마트폰 사용자가 10억 명 이상이라는 점을 고려한다면 실재적으로 더욱 다양한 종류의 스마트폰, 제조사, 화면 크기가 존재할 것이 분명하다.

AOSP 또한 안드로이드를 탑재한 스마트폰의 화면 크기 분포에 대한 통계를 제공하고 있다. <표 16>은 2014년 5월 말 전 세계에 존재하는 안드로이드 스마트폰의 화면 크기와 화면 해상도의 분포를 나타내고 있다.[79] 이 자료에 따르면 안드로이드 스마트폰의 화면크기는 2-3인치 정도부터 10인치 이상까지 다양하며, 화면 해상도 또한 100-130 dpi 정도부터 300 dpi 이상 까지 다양하다.

		화면 해상도						
		ldpi	mdpi	tvdpi	hdpi	xhdip	xxhdpi	합계
화면 크기	작음 (small)	7.2%	-	-	-	-	-	7.2%
	보통 (normal)	-	12.0%	-	34.2%	19.6%	14.6%	80.4%
	큼 (large)	0.6%	4.4%	1.6%	0.6%	0.6%	-	7.8%
	매우큼 (xlarge)	-	4.0%	-	0.3%	0.3%	-	4.6%
	합계	7.8%	20.4%	1.6%	35.1%	20.5%	14.6%	

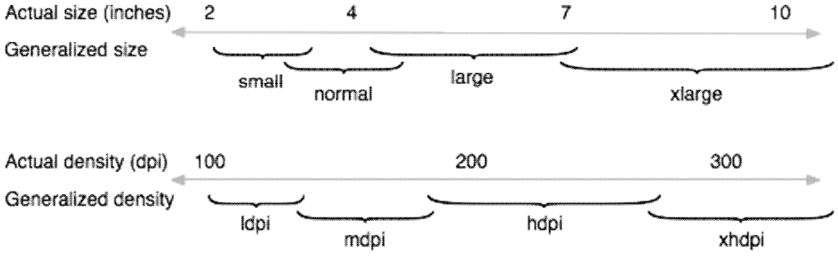
<표 16> 화면 크기와 해상도의 분포. 2014년 5월 29부터 7일간 조사 (출처: 안드로이드 홈페이지. https://developer.android.com/about/dashboards/index.html. 2014년 6월 19일 방문)

스마트폰 기기와 제조사의 다양화와 함께 스마트폰에 탑재되는 안드로이드 운영체제 종류도 다양해지고 있다. AOSP는 지속적으로 기존 안드로이드 운영체제의 성능을 개선하거나 기능을 추가하고 오류를 수정하며 이를 새로이 일반에 공개하기 때문에, 상이한 기술적 특성을 가진 안드로이드 운영체제는 각기 다른 '버전(version)'으로 구분된다. 각 제조사는 스마트폰 기기가 생산되는 시점과 기기가 갖추고 있는 사양에 따라서 서로 다

79) 화면 크기는 직사각형 화면의 대각선 길이(인치)를 나타내며, 화면 해상도는 스마트폰 화면이 1인치 당 표시하는 점의 수(dpi, dots per inch)로 측정한다. 화면의 크기와 해상도는 모바일 어플리케이션와 사용자의 상호작용에 중요한 영향을 미치지만, 개발자가 상이한 화면 크기와 해상도를 각 스마트폰 마다 개별적으로 상세하게 고려하기는 어렵다. 이 때문에 안드로이드 운영체제는 크기와 해상도를 몇 가지의 단계로 대략적으로 구분하며, 개발자는 구분된 크기와 해상도를 기준으로 모바일 어플리케이션을 제작한다.

른 버전의 안드로이드 운영체제를 설치할 수 있다. 또한 이미 출시된 제품들에 대해 제조사가 새로운 안드로이드 운영체제를 적절히 업데이트(update)하지 않는다면, 그 스마트폰은 최근 출시된 다른 스마트폰들에 비해 오래된 버전의 운영체제를 탑재하고 있게 된다. 이러한 요인들로 인해, 스마트폰 기기에 탑재되어 있는 안드로이드 운영체제 버전은, 안드로이드 스마트폰이 확산되고 공개된 버전이 늘어날수록 다양해지게 된다. AOSP은 전 세계에서 사용되고 있는 안드로이드 운영체제의 버전 정보를 수집하여 그 상대적인 비율을 공개하고 있으며, 이는 <그림 19>에서 확인할 수 있다. 2009년 12월부터 조사된 이 자료에 따르면 안드로이드 스마트폰에는 다양한 종류의 안드로이드 운영체제가 탑재되어 있다. 예를 들어 2010년 10월에는 4 종류의 운영체제가 사용되었던 반면, 2011년 10월에는 6 종류, 2012년 10월에는 8 종류, 2013년 10월에는 5 종류의 운영체제가 사용되고 있는 것을 확인할 수 있다.[80]

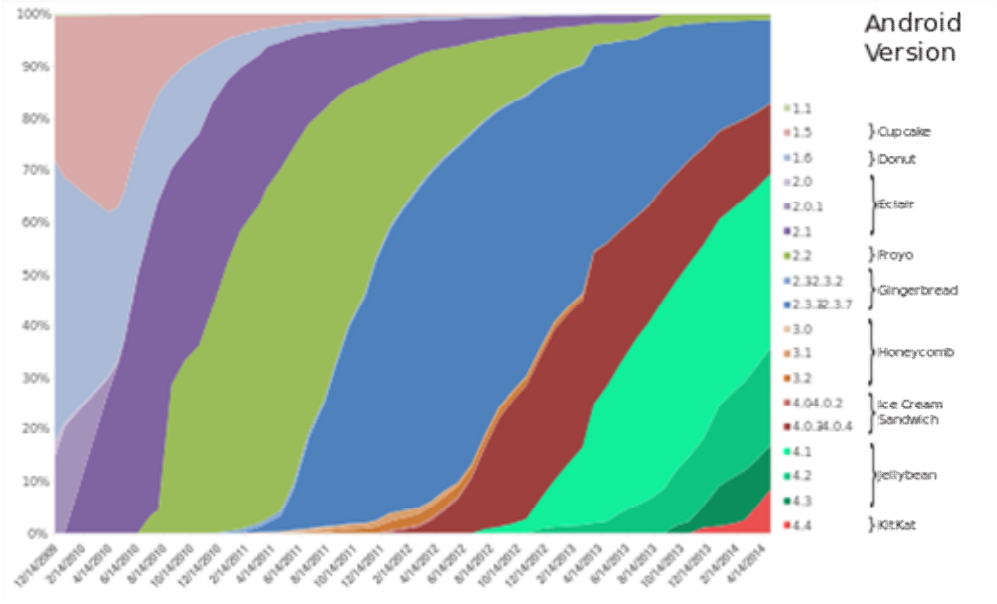
스마트폰 기기에 탑재된 운영체제가 사용자 개인에 의해 변형되기도 한다. 컴퓨터 및 스마트폰 기기에 대한 해박한 지식을 가지고 있는 일부 사



<그림 18> 안드로이드 운영체제에서 화면 크기와 해상도의 구분
(출처: 안드로이드 홈페이지.
https://developer.android.com/guide/practices/screens_support.html. 2014년 6월 19일 방문)

80) 운영체제의 버전은 버전 번호이나 코드명(codename)으로 구분할 수 있다. 하나의 코드명에는 여러 가지 버전번호를 가진 안드로이드 운영체제가 속하게 된다. 새로이 공개되는 안드로이드 운영체제는 반드시 새로운 버전번호를 부여받는다. 이러한 개선 작업이 어느 정도 지속된 후, 구글은 필요에 따라 운영체제에 새로운 코드명을 부여한다. 예를 들어 '젤리빈(Jellybean)'은 코드명 '아이스크림샌드위치(Ice Cream Sandwich)'의 다음 코드명으로써, 4.1에서 4.3.1 사이의 버전번호를 부여받은 운영체제는 이 코드명을 사용한다. 따라서 안드로이드 4.2.1 운영체제는 코드명 젤리빈을 사용한다.

용자들은, 제조사가 스마트폰 출고 시에 설치한 안드로이드 운영체제를 변경, 수정, 개선하여 새로운 운영체제를 만든다. 이러한 운영체제를 커스텀 펌웨어(custom firmware)라고 부르는데, 이러한 운영체제들은 기존 운영체



<그림 19> 시간에 따른 안드로이드 버전별 기기 수 분포
(출처: 구글 자료를 기반으로 재구성된 자료. 원자료: https://developer.android.com/about/dashboards/index.html. 그래프 출처: http://en.wikipedia.org/wiki/Android_version_history. 2014년 6월 19일 방문)

제 보다 개선된 성능과 디자인을 갖거나, 특수한 기능을 포함하고 있다. 세계적으로 수십 개의 안드로이드 커스텀펌웨어가 존재하며,[81] 일부 제조사는 스마트폰에 대한 커스텀펌웨어 개발을 장려하기도 한다. 예를 들어 2011년 스마트폰 제조사인 HTC는 제조사만이 설치, 변경할 수 있던 펌웨어를 개방(unlock)하여 사용자가 커스텀펌웨어를 자유롭게 설치할 수 있도록 했다. HTC는 이러한 사실을 발표하며 다음과 같이 설명했다.

81) 다음과 같은 커스텀 펌웨어의 예들이 있다: Androitonic, Android Open Kang Project, Avatar ROM, AOSB ROM, Baked ROM, CarbonRom, Chameleon OS, Cyanfox, CyanogenMod, Debian Kit, Eclipse, Evervolv, JellyBAM, LinuxOnAndroid, Liquid Smooth, MiniCM, MIUI, MoKee OpenSourse, Omni, Paranoid Android, PAC ROM, REplicant, RootBox, SlimRoms, SOKP ROM, :Ubuntu for Android, Xenon HD.
(출처: 위키피디아. http://en.wikipedia.org/wiki/List_of_custom_Android_firmwares)

[중략] 우리는 당신[고객]들의 목소리를 들어왔고, 2011년 모델부터 시작하여 우리의 부트로더(bootloader)를 개방할 것입니다. [중략] 우리는 우리의 팬들과 충성스런 고객들의 에너지와 열정을 보게 되어 매우 기쁘며, 당신이 무엇을 할 수 있는지 보게 되어 매우 흥분됩니다. HTC는 당신의 혁신을 열정적으로 기대하고 있습니다.

2. 정합성의 지배: 구글에 의한 기기의 규정

(1) 안드로이드 운영체제의 정합성

스마트폰 기기의 다양성은 여러 제조사가 각기 다른 기술적 환경에 안드로이드 운영체제를 사용하고 있다는 것을 반증한다. 이러한 다양성은 사용자의 선택 폭을 넓히며 더욱 다양한 상황에서 스마트폰이 사용될 수 있게 한다. 하지만 스마트폰 기기와 탑재된 운영체제의 다양성은 파편화(fragmentation)의 문제를 야기한다. 여기서 파편화란 스마트폰 기기와 안드로이드 운영체제의 다양성이 너무나 심화된 나머지 일부 스마트폰에서 구동되는 어플리케이션이 다른 스마트폰에서는 구동되지 않는 문제를 말한다. 파편화로 인해 개발자가 만든 어플리케이션이 개발자가 예상한 스마트폰 기기에서 동작할 것임을 보장할 수 없게 된다. 또한 스마트폰 사용자가 원하는 모바일 어플리케이션은 무엇이던지 이용할 수 있다고 보장할 수도 없게 된다. 이와 같이 모바일 어플리케이션과 스마트폰 기기, 사용자 간의 상호작용 중 일부가 불가능해질 수 있으므로, 파편화는 안드로이드 체계에 위험 요소로 작용한다.

따라서 개방성과 파편화로 인한 위험을 제거하기 위해서는, 하나의 모바일 어플리케이션이 모든 스마트폰 기기에서 동작할 수 있도록 해야 한다. 이와 같이 이상적인 기술적 특성을 '정합성(compatibility)'이라고 한다.[82]

---

82) 정보기술은 여러 가지 기술적 요소들의 상호작용을 통해 동작하게 된다. 예를 들어 사용자가 스마트폰을 통해 인터넷을 사용하는 과정에서는 모바일 어플리케이션, 스마트폰 운영체제, 스마트폰 기기, 무선 이동통신 기지국, 광케이블, 이동통신사의 데이터통신 서버 등 수많은 기술적 요소 간의 상호작용이 일어나게 된다. 그리고 이러한 상호작용이 가능하기 위해서는 기술적 요소들이 미리 정해진 방식에 따라서 정보와 기능을 주고받아야 한다. 한 기술적 요소가 다른 기술적 요소에게 정확한 정보를 전송하였다고 해도, 미리 정해진 형식을 따르지 않았다면 이 정보를 수신하는 측은 정보를 수신하지 못하거나 잘못된 정보를 수신하게 된다.

이와 같이 미리 정해진 상호작용 형식을 만족하여 기술적 요소 간의 정상적인 상호작용이 일어나는 것은 '정합성(compatibility)' 부른다. 정보기술에서의 정합성은 본래 두 개의 서로 다른 하위체계가 상호작용하기 위해서 따라야할 최소한의 규칙으로 규정된다. 예를 들어 두 컴퓨터가 상호작용하거나, 하나의 소프트웨어가 임의의 하드웨어에서 문제없이 동작하기 위해 만족시켜야할 상호작용 형식이 정합성으로 규정되며, 정합성으로 규정된 조건들을 만족시키는 기술은 서도 데이터나 기능을 주고받을 수 있는 것이다. 정합

AOSP는 제조사가 정합적인(compatible) 기기를 만들어야 하는 이유 3가지를 제시한다. 첫째 이유는, 사용자들이 개인화 가능한(customizable)한 기기를 원하기 때문이다. 모든 사용자들이 모바일 어플리케이션을 설치해 스마트폰의 기능을 자신에게 맞게 조절, 확장하고자 하기 때문에, 제조사들이 변경한 안드로이드 운영체제는 여러 어플리케이션을 동작시킬 수 있을 만큼 신뢰성이 있어야 한다. 둘째 이유는, 모바일 어플리케이션 개발자의 숫자가 제조사의 숫자보다 훨씬 많기 때문이다. 한 제조사가 사용자들이 필요로 할 수 있는 모든 어플리케이션을 직접 만드는 것은 불가능하기 때문에, 이러한 어플리케이션을 만드는 제 3의 개발자 집단(third-party developers)이 필요하다. 그리고 이러한 개발자들이 다양한 기기별 특성에 대한 복잡한 고려 없이도 쉽게 어플리케이션을 만들 수 있는 환경이 조성되어야 한다. 셋째 이유는 모두가 공통의 생태계(common ecosystem)을 필요로 하기 때문이다. 정합성을 갖춘 스마트폰이 많아질수록 개발자는 시간과 노력을 특정 기기의 오류를 수정하기 위해서가 아닌 더 많은 모바일 어플리케이션을 개발하는 데 사용할 수 있다는 것이다.[83]

AOSP는 안드로이드 체계의 정합성을 달성하기 위해 몇 가지 방안을 마련해 놓고 있다. 아래의 인용문에서도 알 수 있듯, 정합성을 만족시키기 위한 장치에는 3가지(정합성 프로그램, 안드로이드 개발자도구, 구글플레이)가 있다.

> 안드로이드의 목적은 개발자들이 혁신적인 앱을 만들 수 있는 개방적 플랫폼을 만드는 것입니다. 안드로이드 정합성 프로그램(Android Compatibility Program)은 안드로이드 플랫폼의 기술적 세부사항을 정의하고 다양한 기기에서 개발자들의 앱이 동작할 수 있음을 보장하는 도구들을 제조사에게 제공합니다. 안드로이드 개발자도구(Android SDK)는 개발자들이 앱이 요구하는 기기 사양을 명확히 선언하는데 사용하는 도구들을 제공합니다. 그리고

성은 특히 하나의 기능을 위해 많은 요소들이 중첩되어야 하는 정보기술의 영역에서 중요하다. 기업 간의 연합이나 초국가적 협의체에 의해 제정되는 전자기술, 정보통신 관련 기술표준은 이러한 정합성의 중요성을 보여주는 대표적인 예이다(Schmidt & Werle, 1998).

83) 안드로이드 홈페이지. http://source.android.com/compatibility/. 2014년 6월 20일 방문

구글플레이(Google Play)는 기기가 적절하게 구동시킬 수 있는 앱들만을 표시합니다.
(출처: 안드로이드 홈페이지. http://source.android.com/compatibility/. 2012년 6월 20일 방문)

개발자는 어플리케이션을 만들 때 반드시 안드로이드 개발자도구를 사용해야 하며, 이 도구에서는 개발자들이 필요한 기술적 사양들(영상 표시, 처리 성능, 입출력 기능 등)을을 어플리케이션 내부에 선언해 놓을 수 있도록 하고 있다. 모바일 어플리케이션 마켓인 구글플레이는 이러한 선언을 바탕으로 사용자의 스마트폰 기기에서 정상적으로 동작할 수 있는 어플리케이션만을 표시한다. 하지만 개발자도구와 구글플레이는 단지 모바일 어플리케이션이 필요로 하는 기술적 환경을 '선언'하고 이 선언에 기준하여 알맞은 어플리케이션을 '표시'하는 소극적인 역할에 그치며, 파편화로 인한 문제를 막고 정합성을 높이는 직접적인 역할을 하지는 못한다. 정합성을 달성하기 위한 적극적이고 직접적인 방법은 모든 스마트폰 기기의 기술들이 특정한 기준을 만족시켜 모바일 어플리케이션이 모든 스마트폰에서 동작할 수 있음을 '보장'하는 것이다. 이러한 정합성을 보장하기 위해 마련된 장치가 바로 안드로이드 정합성 프로그램이다.

(2) 정합성의 정의와 외부 행위자의 통제

안드로이드 정합성 프로그램은 제조사들이 정합성을 갖춘 기기를 만들게 하기 위한 장치이다. AOSP는 이 프로그램이 네 종류의 집단에 도움을 줄 수 있다고 밝히고 있다.[84] 우선 정합성 프로그램은 어플리케이션 개발자에게 일관된 기술적 환경을 제공한다. 둘째, 이 프로그램은 소비자들의 일관된 어플리케이션 경험을 가능하게 한다. 셋째, 정합성 프로그램은 제조사들이 정합성을 갖추면서도 차별화된 기기를 만들 수 있도록 한다. 마지막으로 이 프로그램은 정합성과 관련된 비용과 부담을 최소화한다. 이를 종합

84) http://source.android.com/compatibility/overview.html 참조.

하여 보면, 안드로이드 정합성 프로그램이 주는 가장 큰 효과는 '항상성(consistency)'이라고 할 수 있다. 즉 제조사가 정합성 프로그램을 만족시킴으로써, 개발자는 일관된 기술적 환경에서 개발을 하고, 소비자들은 일관된 어플리케이션 경험을 얻을 수 있으며, 제조사는 기기를 차별화하면서도 일관되게 어플리케이션을 동작시킬 수 있다. 이러한 항상성은 어느 기기에나 적용되는 동일한 기준이 있어야만 이루어질 수 있다. 안드로이드 정합성 프로그램은 크게 정합성정의문서(Compatibility Definition Document, CDD)의 준수와 정합성검사도구(Compatibility Test Suite, CTS)의 실행으로 이루어지며, 이것들은 모든 기기가 만족시켜야할 기술적인 기준을 제시하고 있다.

정합성정의문서에는 제조사가 만든 스마트폰이 만족시켜야할 소프트웨어적, 하드웨어적으로 만족시켜야할 기준이 항목별로 구체적으로 기술되어 있다. 구체적으로 살펴보면, 2009년 12월에 공개된 안드로이드 운영체제 버전 1.6의 2차 수정버전에 대해 첫 정합성정의문서가 공개된 이후, 2014년 6월 현재 안드로이드 버전 2.1, 2.2, 2.3, 4.0, 4.1, 4.2, 4.3, 4.4의 9개 안드로이드 버전에 대한 정합성정의문서가 공개되었다.[85] 예를 들어 이 중 안드로이드 운영체제 버전 4.3에 대한 정합성정의문서의 구체적인 항목은 <표 17>에 제시되어 있다. 이 표는 정합성정의문서의 대분류와 그 내부의 세부항목들을 나타내고 있으며, 각 항목은 또 다시 여러 개의 필수사항 및 권장사항을 포함한다. 이 중 '소프트웨어 정합성' 항목은 제조사가 변경, 수정, 개선한 안드로이드 운영체제가 만족시켜야할 기술적 조건들을 포함하고 있다. 또한 '어플리케이션 패키징 정합성'은 제조사의 운영체제가 모바일 어플리케이션의 설치, 실행에 있어 만족시켜야 할 기준들을 포함하며, '멀티미디어 정합성'은 제조사의 스마트폰에서 음향이나 영상을 저장, 재생할 때 만족시켜야 하는 요건들을 나타낸다. 그리고 '개발자 도구와 옵션 정합성'은 제조사들이 스마트폰에 모바일 어플리케이션 개발자를 위한 도구들을 포함시켜야 함을 규정하고 있으며, '하드웨어 정합성'은 제조사가 스

85) 안드로이드 첫 버전(1.0)부터 버전 1.6 출시 전까지는 제조사의 기기에 대한 기준이 존재하지 않는다. 따라서 안드로이드 운영체제 개발의 초기 시기에는 운영체제를 탑재하고 있는 어떠한 기기라도 구글의 코드를 탑재할 수 있었다.

소프트웨어 정합성	
하드 API 정합성	
소프트 API 정합성	허가
	빌드 파라미터
	의도 정합성
네이티브 API 정합성	
웹 정합성	웹뷰 정합성
	브라우저 정합성
API 동작 정합성	
API 명칭공간	
가상머신 정합성	
사용자 인터페이스 정합성	바탕화면
	위젯
	알림
	검색
	토스트
	테마
	라이브 월페이퍼
	최근 어플리케이션 표시
	입력 관리
	잠금 화면 미디어 원격 조정
	드림
	위치
	유니코드
기기 등록	
접근성	
텍스트 투 스피치	
어플리케이션 패키징 정합성	
멀티미디어 정합성	
미디어 코덱	
비디오 인코딩	
비디오 디코딩	
오디오 녹음	
오디오 지연	
네트워크 프로토콜	
개발자 도구와 옵션 정합성	
개발자 도구	
개발자 옵션	
하드웨어 정합성	
디스플레이와 그래픽	스크린 특성
	디스플레이 메트릭
	스크린 방향성
	그래픽 가속
	레거시 어플리케이션 정합 모드
	스크린 유형
	스크린 기술

	외부 디스플레이
입력 장치	키보드
	비터치식 방향키
	방향키
	터치스크린 입력
	허위 터치 입력
	마이크
센서	가속계
	자력계
	GPS
	자이로스코프
	기압계
	온도계
	광센서
	근접센서
데이터 연결	전화
	와이파이
	와이파이 다이렉트
	와이파이 TDLS
	블루투스
	근거리 통신(NFC)
	최소 네트워크 기능
카메라	후면 카메라
	전면 카메라
	카메라 API 동작
	카메라 방향
메모리와 저장	최소 메모리와 저장공간
	어플리케이션 공유 저장공간
USB	
성능정합성	
보안 모델 정합성	
허가	
사용자와 프로세스 격리	
파일시스템 허가	
대체 실행 환경	
복수 사용자 지원	
프리미엄 SMS 경고	
커널 보안 사양	
프라이버시	
디스크 전체 암호화	

<표 17> 안드로이드 운영체제 4.3
정합성정의문서의 세부항목

마트폰에 필수적으로 포함해야 하거나 선택적으로 포함할 수 있는 기술적 요소들(화면, 키보드, 센서, 통신 기능, 카메라, 메모리 등)와 각 요소의 특성, 성능을 규정하고 있다. ‘성능 정합성’은 스마트폰이 가진 어플리케이션 구동 성능의 하한선을 규정하고 있으며, ‘보안모델정합성’은 사용자의 데이

터를 보호하고 외부의 침입을 막기 위해 제조사가 스마트폰에 포함시켜야 할 기술들을 규정하고 있다.

구글은 정합성의 정의를 통해 제조사의 생산물을 통제하여 파편화로 인한 위험성을 제거하고 기술체계의 안정을 도모한다. 제조사의 변형으로 인해 운영체제의 분화가 심화되어 파편화가 일어나면 일부 운영체제에서 모바일 어플리케이션을 실행할 수 없게 된다. 그리고 분화된 운영체제 별로 사용자와 모바일 어플리케이션의 소규모 집단이 형성된다. 이와 같이 제조사, 개발자, 사용자가 처해 있는 기술적 환경이 상이해지면 하나의 행위자가 생산과정을 통제하거나 모든 곳에서 사용되는 핵심 기술을 만들기는 어려워지며, 이는 곧 기존의 지배 구조가 약화됨을 뜻한다. 따라서 구글은 정합성을 통해 기술의 변형을 규제하고, 필수적인 기술 사양을 규정한다. 즉 구글은 분화의 압력에 저항하기 위해 정합성의 기준들을 통해 '안드로이드'라는 단일한 기술적 정체성을 정의하고 있는 것이다. 정합성의 유지는 안드로이드 체계에서 가장 중요한 목표로서, 구글은 안드로이드 체계의 정합성 기준을 만족하는 제조사에게만 어플리케이션 시장의 접근과 상표권의 사용을 허락하고 있다.[86]

86) 정합성 프로그램 외에도 파편화를 방지하기 위한 수단이 존재한다. 구글은 개방핸드셋연합에 소속된 회사들에게 안드로이드의 반-파편화(anti-fragmentation) 조항에 동의하게 하였다.

V. 기술체계의 결합과 구글의 권력 확장

1. 의무통과점의 형성과 하위체계의 결합

(1) 기술적 의존 구조의 형성

이미 살펴보았듯, 안드로이드 운영체계의 개방성으로 인해 오히려 안드로이드 어플리케이션은 구글의 정보하부구조에 의존하게 되었다. 모바일 어플리케이션에 삽입된 구글코드는 구글의 정보하부구조를 사용할 수 있도록 해주는데, 이러한 구글코드는 구동을 위해서 또 다른 코드를 필요로 하도록 설계되어 있다. <그림 20>은 구글코드가 동작하는 방식을 나타내고 있다. 모바일 어플리케이션(그림에서 'Your App'으로 표시된 부분)에 삽입된 구글코드(그림에서 'Client Library'라고 표시된 부분)는 스마트폰에 설치된 구글플레이서비스(Google Play Service, 그림에서 'Google Play services APK'라고 표시된 부분)라는 또 다른 코드를 호출한다. 구글플레이서비스라는 코드는 어플리케이션 마켓의 역할을 하는 코드인 구글플레이스토어(그림에서 'Google Play Store'라고 표시된 부분)를 통해 설치되고 최신 버전으로 업데이트 된다. 구글플레이스토어를 통해 구글플레이서비스가 설치, 업데이트되지 않는다면 어플리케이션이 사용하고 있는 구글코드가 정상적으로 동작할 수 없다. 안드로이드 운영체제가 오픈소스 소프트웨어인 것과 달리 구글플레이서비스는 오직 구글에 의해서 만들어지고 구글의 승인 하에만 설치가 가능한 사유화된 코드이다. 따라서 안드로이드 운영체제를 탑재한 스마트폰에서 구글코드를 포함한 모바일 어플리케이션을 동작시킬 수 있도록 하려면, 제조사는 기기 출시 이전에 구글의 승인을 받고 구글플레이스토어를 설치하여야 한다.[87]

이러한 구조의 특징은 '외부화'와 '집중'이다. 우선, 구글코드는 독자적으

87) 안드로이드 개발자 홈페이지에는 다음과 같이 명시되어 있다. "구글플레이서비스는 구글플레이스토어를 통해서 전달되기 때문에, 서비스의 업데이트는 이동통신사나 기기 제조사의 운영체제 코드 변화에 영향을 받지 않습니다." (출처: 안드로이드 개발자 웹페이지. http://developer.android.com/google/play-services/index.html)

<그림 20> 구글 API와 다른 코드 간의 상호작용 (출처: http://developer.android.com/google/play-services/index.html, 2014년 6월 11일 방문)

로 동작할 수 없으며, 모바일 어플리케이션의 '외부'에 존재하는 구글플레이스토어라는 개별적 코드를 반드시 필요로 한다. 만약 어플리케이션에 삽입된 구글코드가 독자적으로 기능하게 설계되었다면, 모든 스마트폰에 구글플레이스토어가 설치될 필요가 없었을 것이다. 이와 더불어 다른 행위자는 구글플레이스토어의 설치를 결정할 권한이 없으며, 오직 구글만이 구글플레이스토어의 설치를 결정할 수 있다. 구글플레이서비스를 제조사 혹은 사용자가 임의로 설치, 업데이트 할 수 있었다면 스마트폰이 출시될 때부터 구글플레이스토어를 포함해야할 필요가 없었을 것이다. 따라서 이러한 구조는 구글플레이스토어의 설치 권한과 구글코드를 포함한 어플리케이션의 작동 가능 여부를 결정할 권한을 구글에 집중시킨다. 이와 같이 외부화와 집중화된 기술적 구조로 인해 구글코드를 포함한 어플리케이션들이 자체적으로 동작할 수 없으며 특정 행위자의 결정에 영향을 받기 때문에, 이러한 구조를 '기술적 의존구조'로 칭할 수 있다.

이러한 기술적 의존구조는 정보기술 분야에서 흔히 발견될 수 있다. 예를 들어 컴퓨터 운영체제인 윈도우(Windows)의 일부 기능이 제거된다면

윈도우용 프로그램이 동작하지 못할 수 있으며, 이 프로그램의 동작 여부는 마이크로소프트(Microsoft)사가 내린 결정에 영향을 받을 수 있다. 소셜 네트워크서비스의 하나인 페이스북(Facebook)의 일부 기능이 제거된다면 페이스북과 연동되는 기능이나 서비스이 동작하지 못할 수 있기 때문에, 이러한 기능과 서비스의 동작 여부는 페이스북의 기술적, 정책적 변화에 영향을 받는다. 이와 같이 기술적 의존구조는 외부의 다른 기술에 기반하여 동작하는 모든 정보기술에 적용 가능한 개념이다. 하지만 안드로이드 체계에서 발견되는 구글 중심의 폐쇄적이고 사유적인 의존 구조는 개방적이고 공유적인 기술적 환경 하에서 형성되었다는 점에서 특징적이다. 안드로이드는 배태된 의존구조가 없는 개방성을 추구하는 운영체제를 표방한다. 개방성은 의존, 지배의 구조를 배제함으로써, 한 기업의 선택이 다른 기업과 안드로이드 체계를 구성하는 기술적 요소 전체에 영향을 미치지 못하도록 한다. 즉 안드로이드 체계의 개방성을 위해서는 중심장애점(central point of failure)이 없어야 하며, 이는 아래의 인용문에서도 확인할 수 있다.

> 안드로이드는 서로 다른 폼팩터[88]를 가진 다양한 기기 안드로이드기기를 위해 만들어진 오픈소스 소프트웨어 스택[89]이다. 안드로이드의 주된 목적은 통신사, 제조사와 개발자가 그들의 창의적인 아이디어들을 실현시키도록 하고, 이들이 사용자의 모바일 경험을 증진시키는 성공적이고 실질적인 제품을 도입할 수 있게 하는 것에 있다. 또한 우리는 한 회사가 다른 기업의 창조를 억제하거나 통제할 수 할 수 있는 중심장애점[90]이 없도록 하고 싶다. 그 결과는 소스 코드가 개인화(customization)와 포팅(porting)에 개방되어 있는 완전한, 생산품질의 소비재이다.
>
> (출처: AOSP 홈페이지, https://source.android.com/source, 2014년 6월 21일)

88) 스마트폰의 폼팩터(form factor)란 스마트폰의 화면 크기, 연산 성능, 통신 속도 등 기기의 각 부품이 가진 물리적 치수의 조합을 의미한다.

89) 소프트웨어 스택(software stack)이란 하나의 목표나 결과를 달성하기 위해 함께 동작하는 소프트웨어의 묶음을 의미한다.

90) 여기서 중심장애점(central point of failure)는 단일장애점(single point of failure)와 유사한 의미를 갖는다. 만약 기술체계 전체에서 하나의 기술적 요소의 고장으로 전체 체계가 동작할 수 없게 될 때, 이러한 기술적 요소를 단일장애점이라고 칭한다.

이와 동일한 방향성을 '왜 안드로이드 소스코드를 개방하는가?'에 대한 아래의 답변에서도 확인할 수 있다.

> 구글은 직접 모바일 어플리케이션을 출시하였던 경험에 대응하여 안드로이드 프로젝트를 시작하게 되었습니다. 우리는 창의적인 아이디어를 현실로 만들 수 있는 개방적인 플랫폼을 통신사, 제조사, 개발자가 언제든지 사용할 수 있게 하고 싶습니다. 우리는 또한 중심장애점이 없게 하여, 다른 회사의 혁신을 제한하거나 통제하는 회사가 없도록 하고 싶습니다. AOSP의 가장 중요한 목표는 모두가 혜택을 받을 수 있도록 오픈소스 안드로이드 소프트웨어가 가능한 폭넓고 정합적으로 구현되는 것입니다.
> (출처: AOSP 홈페이지, https://source.android.com/faqs.html, 2014년 6월 21일)

하지만 개방성의 '취지'와는 달리 모바일 어플리케이션은 구글플레이스토어와 구글플레이서비스, 그리고 구글코드로 구성된 구조에 의존한다. 구글플레이서비스는 구글코드를 포함한 모바일 어플리케이션의 동작에 영향을 미칠 수 있으며, 이는 구글플레이서비스가 중심장애점으로 작용할 수 있음을 의미한다.

(2) 기술체계의 결합과 구글의 목적

'부활' 혹은 '은폐'된 중심장애점을 통해 구글이 표면적으로 내세우는 행위자의 자율성 보장과 기술체계의 붕괴 위험성이 아닌 다른 목적을 우선시하고 있다는 점을 알 수 있다. 이러한 목적을 확인하기 위해서는 구글플레이스토어라는 기술을 좀 더 자세히 살펴볼 필요가 있다. 구글플레이스토어는 ('스토어(store)'라는 단어가 나타내듯) 주로 모바일 어플리케이션의 시장 기능을 하고 있다. 개발자는 자신이 만든 모바일 어플리케이션을 구글플레이스토어에 등록하고 원하는 국가에 출시할 수 있으며, 사용자는 구글플레이스토어에서 사용하고 싶은 어플리케이션을 선택, 설치한다. 이와 같이 구글플레이스토어는 모바일 어플리케이션을 확산, 유통시키는 정보하부

구조이다. 반면 기술적 의존구조에서 구글플레이스토어는 모바일 어플리케이션의 기능이 온전히 동작하기 위해서 필수적인 코드이며, 이러한 측면에서 구글플레이스토어는 안드로이드 어플리케이션의 기능이 구동되도록 하는 정보하부구조이기도 하다. 이와 같이 구글플레이스토어는 안드로이드 어플리케이션의 기능적, 확산적 정보하부구조의 핵심 부분이다.

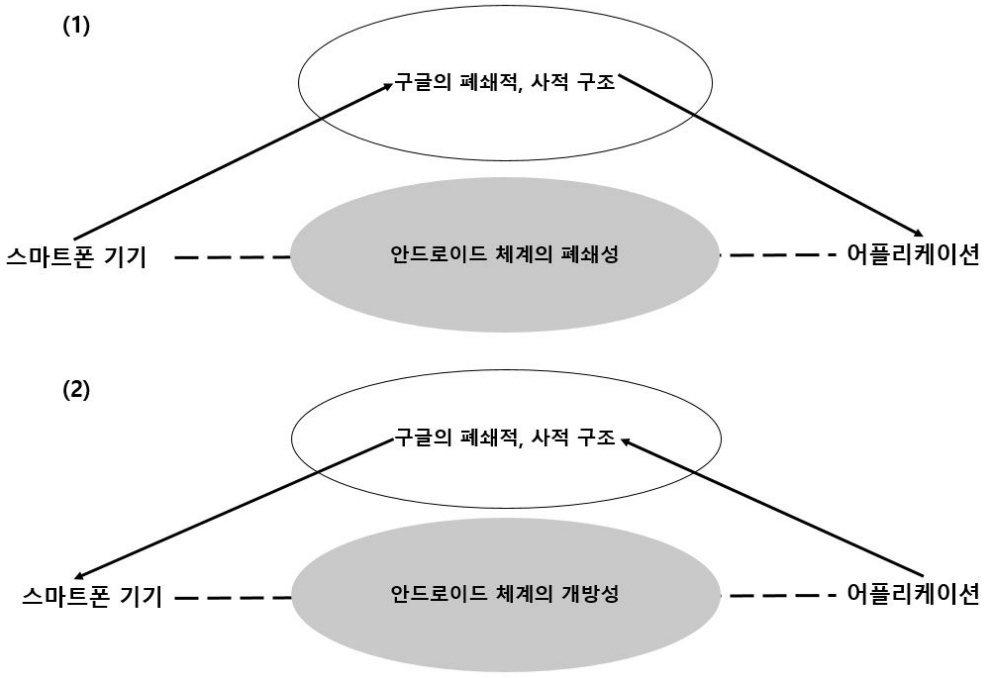
구글플레이스토어를 통한 두 하부구조의 중첩이 어떠한 효과를 가져왔을까? 만약에 기능적 정보하부구조와 확산적 정보하부구조가 통합되지 않았다면, 사용자가 구글의 마켓을 이용해 특정 어플리케이션을 설치하더라도 이를 실행하는 동안에는 구글의 마켓이 반드시 필요하지는 않을 것이다. 왜냐하면 어플리케이션에 구글코드가 포함되어 있더라도 이 구글코드가 구글플레이스토어를 호출하지 않을 것이기 때문이다. 하지만 두 하부구조의 중첩으로 인해 구글플레이스토어는 정보 상품의 시장과 더불어 어플리케이션의 기술적 기반이 되었으며, 이로 인해 구글플레이스토어는 선택, 설치라는 순간적 행위뿐만 아니라 어플리케이션의 사용이라는 반복적, 지속적인 행위에 영향을 미칠 수 있게 되었다.

기술적 구조를 통한 영향력의 확장은 구글플레이스토어라는 기술적 요소의 중요성을 더욱 강화시킨다. 구글플레이스토어가 있어야만 어플리케이션은 구글코드의 기능을 정상적으로 동작시킬 수 있으며, 이를 통해 어플리케이션 개발자가 수익, 명성 등의 목표를 달성할 수 있다. 또한 이는 제조사에도 영향을 미친다. 안드로이드 스마트폰 사용자가 구글의 어플리케이션 마켓에서 다양한 어플리케이션을 선택하고 설치한 어플리케이션이 구동될 수 있으려면 구글플레이스토어가 스마트폰 기기에 설치되어야 한다. 하지만 이미 언급되었듯, 구글플레이스토어를 설치하기 위해서는 구글의 승인이 필요하다. 제조사가 스마트폰 기기에 구글플레이스토어를 설치하기 위해서는 AOSP에 제시된 정합성프로그램을 이행해야 한다. 즉 제조사가 만드는 스마트폰 기기는 정합성정의문서에 언급되어 있는 모든 조건들을 만족시켜야 하며, 정합성검사도구를 통해 검증되어야 구글플레이스토어를 설치할 수 있다. 이와 같이 구글플레이스토어는 제조사와 스마트폰 기기가 모바일 어플리케이션과 개발자가 상호작용하기 위해 필수적인 기술적 구조

이며, 제조사가 이러한 필수적 구조에 포함되기 위해서는 구글이 제시하는 기준을 통과해야 한다.

여기서 스마트폰 기기와 모바일 어플리케이션이 결합하기 위해서는 구글의 폐쇄적이며 사적인 구조로의 '기술적 우회'가 필요하지만, 기술적 우회가 발생하는 요인은 다르다는 것을 알 수 있다. <그림 21>의 (2)번은 모바일 어플리케이션이 스마트폰에 탑재되기 위해서 필요한 기술적 우회를 도시하고 있다. 안드로이드 어플리케이션이 실행되기 위해서는, 어플리케이션이 반드시 안드로이드 운영체제를 탑재하고 있는 스마트폰 기기에 결합(설치)되어야 한다. 3장에서 설명되었듯이 안드로이드 체계의 개방성은 공적 정보하부구조의 부재를 발생시킬 뿐만 아니라, 운영체제와 스마트폰 기기의 파편화로 인해 모바일 어플리케이션의 성공적인 동작 가능성을 낮춘다. 이러한 장애물을 피하기 위해 어플리케이션은 구글의 구조에 의존할 수밖에 없게 된다.

반면 (1)번은 스마트폰 기기가 모바일 어플리케이션과 결합하기 위해서 필요한 기술적 우회를 나타내고 있다. 스마트폰 기기에는 다양한 모바일 어플리케이션이 설치될 수 있어야 기능의 추가, 확장이 가능하고, 이는 사용자가 스마트폰을 구매하는 요인이 될 수 있다. 하지만 안드로이드 체계의 폐쇄성으로 인해 스마트폰 기기가 다양한 모바일 어플리케이션을 확보하기 위해서는 구글의 구조가 필요하며, 이는 두 가지의 요인에 기인한다. 첫 번째 요인은 구글에 대한 기술적 의존 구존이다. 모바일 어플리케이션은 구글의 정보하부구조를 필요로 하며, 이러한 하부구조를 이용하기 위해서는 반드시 구글플레이스토어, 구글플레이서비스가 스마트폰에 설치되어야 한다. 두 번째는 안정성과 정합성에 대한 구글의 지배이다. 구글이 안드로이드 운영체제의 생산 과정에서 지배적 위치를 점유하고 있기 때문에, 제조사가 운영체제를 원하는 방식으로 변형하거나 운영체제에 관련한 지식을 경쟁사보다 먼저 획득하기 위해서는 구글이 제시한 기준들을 따라야 한다. 또한 특정 제조사가 구글플레이스토어를 탑재하기 위해서는 정합성기준을 따라야 하는데, 이는 구글이 제조사들에게 생산 과정 참여권한과 차별화된 지식과 정보의 제공 여부를 판단하는 기준이기도 하다. 세 번째는

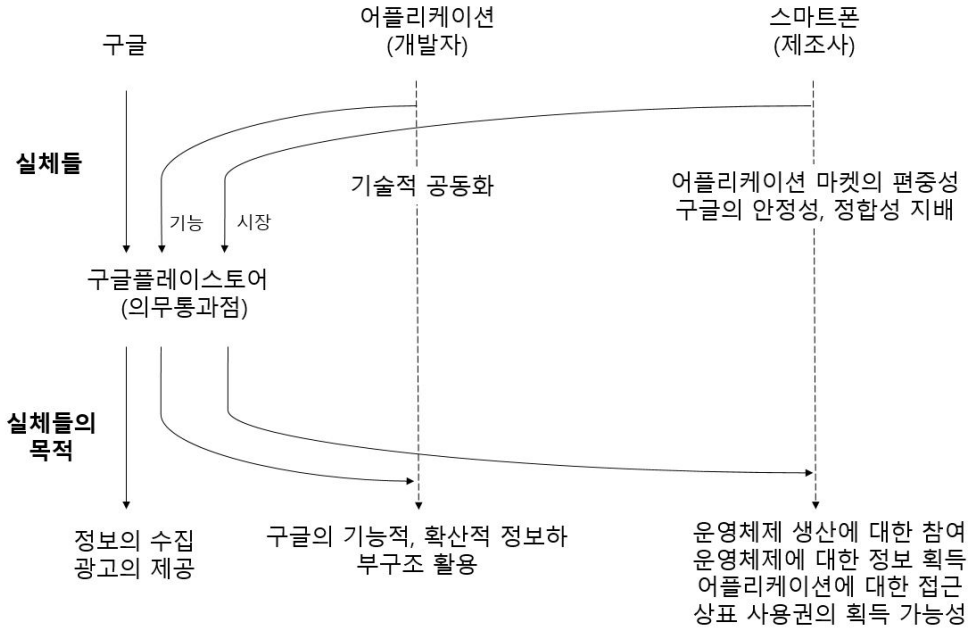


<그림 21> 스마트폰 기기와 어플리케이션 간 결합을 위한 기술적 우회

시장적 지배력이다. 구글의 승인을 얻은 제조사는 구글플레이스토어를 탑재할 수 있을 뿐만 아니라 '안드로이드'라는 상표도 이용할 수 있기 때문에 더욱 많은 사용자에게 제품을 판매할 수 있다.[91] 사용자는 구글플레이스토어가 제공하는 다양한 어플리케이션을 사용하기 위해 '안드로이드', '구글플레이'라는 로고가 들어간 스마트폰을 구매한다.

이와 같이 스마트폰 기기와 모바일 어플리케이션은 구글플레이스토어라는 동일한 기술적 구조를 통해 상대 기술과 결합할 수 있게 된다. 또한 두 기술체의 우회가 구글플레이스토어라는 동일한 기술을 통해 이루어지기 때문에, 구글플레이스토어는 안드로이드 체계에서 서로 다른 기술적 요소 또는 행위자를 연결하는 의무통과점(obligatory passage point)으로 작용한다.

91) 'Android', 'Google Play'의 명칭과 로고는 등록된 상표로 이것들을 사용하기 위해서는 반드시 구글의 승인을 얻어야 한다. 따라서 '안드로이드 탑재' 혹은 '안드로이드 폰' 등으로 설명되는 스마트폰들은 모두 구글의 사전 승낙을 얻은 제조사에서 만든 것이다. 상표권에 대한 승낙은 구글플레이스토어 및 구글이 만든 어플리케이션(Gmail, Youtube 등) 사용에 대한 승낙과도 밀접한 연관이 있다.



<그림 22> 의무통과점으로서의 구글플레이스토어

구글은 의무통과점을 설정하여 안드로이드 운영체제, 스마트폰 기기, 운영체제, 개발자, 제조사, 사용자를 자신을 중심으로 정렬하고 하나의 체계, 하나의 연합(assemblage)으로 만들 수 있었다(<그림 22> 참조). 구글이 이러한 구조를 통해 얻으려는 목적은 무엇인가? 구글의 표면적 목적은 최대한의 정보를 수집, 가공하는 것이다(<그림 23> 참조). 이러한 목표는 구글의 실제 수익 모델과도 연계되어 있다. 구글은 기술을 이용하는 사용자의 특성을 분석하여 개인의 목적, 취향 등을 알아내고, 이러한 정보를 체계화하여 맞춤형 광고를 제공한다. 광고를 통한 매출은 구글의 전체 매출 중 높은 비중을 차지하고 있다.[92] 스마트폰에서 최대한의 정보를 수집, 처리하고 이로부터 광고수익을 얻기 위해서 구글은 더 많은 구글코드가 동작하도록 해야 한다. 구글코드는 사용자의 정보를 수집하고 사용자에게 광고를 표시할 수도 있기 때문이다. 더 많은 구글코드가 동작하기 위해서는 구글코드가 많은 스마트폰에 설치되어야 할 뿐만 아니라, 전 세계의 다양한 스마트

92) 2010년부터 2013년도 까지 구글의 총매출에서 총광고매출의 비율을 계산하여 보면 다음과 같다.

<그림 23> 구글의 목표 (구글 홈페이지, https://www.google.co.kr/intl/ko/about/, 2014년 6월 11일 방문)

폰에서 코드의 동작을 보장할 수 있어야 한다. 이러한 측면에서 구글플레이서비스는 구글코드가 다양한 스마트폰에서 동작할 수 있는 항상적 기반을 제공하는 기능을 한다. 사용자가 누구이던지, 어떠한 종류의 스마트폰이든지, 구글에 의해 승인된 구글플레이서비스만 있으면 구글코드가 동작하기 위해 필요한 조건들이 충족되기 때문이다. 개방성을 통한 기술적 공동화가 구글코드를 물리적으로 확산시킨다면, 구글플레이서비스는 확산된 구글코드들이 구글의 영향력을 실질적으로 향상시킬 수 있도록 한다.

안드로이드 스마트폰 사용자의 수적 증가, 안드로이드 스마트폰을 탑재한 기기 수의 증가를 통한 체계의 확장은 구글의 실질적 이득으로 이어진다. 왜냐하면 더 많은 사람이 안드로이드 스마트폰을 사용할수록 구글이

시기	2010년도	2011년도	2012년도	2013 년도
총매출	29,321	37,905	46,039	55,519
총광고매출	22,889	36,532	43,686	50,547
광고매출 비중	**78.1%**	**96.4%**	**94.9%**	**91.0%**

<표 18> 구글의 광고매출 비중. 단위는 백만 달러.
(구글 공시자료를 재구성.
자료 출처: http://investor.google.com/, 2014년 6월 15일 방문)

획득하는 정보가 늘어나고, 이를 통해 구글은 더 많은 기기에 더욱 정교한 맞춤형 광고를 표시할 수 있기 때문이다. 구글의 목적이 일견 추상적이고 이상적으로 보이지만, 지금까지 이들이 웹페이지, 지역, 도서관, 정부, 개인 정보 등 가능한 모든 정보를 체계화하고 사용자에게 맞춤형 광고를 제공하며 수익의 창출해 왔다는 점을 고려하면, 이는 오히려 현실적이고 구체적인 목표라고 할 수 있다.

2. 행위자의 지위변동

(1) 공적 기술의 쇠퇴와 기술적 사영화

스마트폰 발전의 초기 단계, 모바일 운영체제는 개별 제조사의 선택에 의해 결정되었다. 시장조사 업체인 가트너(Gartner, 2009)에 의하면 2007년에는 모바일 운영체제 중 심비안이 63.5%로 가장 큰 시장 점유율을 보이고 있었다. 그 밖에도 12.0%를 차지한 윈도우 모바일, 각각 9.6% 씩을 차지한 RIM와 리눅스(Linux), 그 밖에도 3% 미만의 점유율을 차지한 맥 운영체제(Mac OS)[93]와 팜 운영체제(Palm OS) 등이 있었다. 이 중 (이후 iOS으로 재명명 되는) 맥 운영체제와 RIM은 애플의 아이폰과 블랙베리에만 탑재되는 운영체제였으며, 심비안이나 윈도우 모바일을 기기에 탑재할 경우 제조사는 각각 심비안(Symbian Ltd.)과 마이크로소프트(Microsoft)에 로열티를 지불해야 했다.[94] 안드로이드 운영체제는 이들에 비해 늦게 공개되었지만 모바일 기기 제조사와 이동통신망 사업자에게 몇 가지 이점들을 제공했다.. 제조사는 무료로 공개되는 안드로이드 운영체제를 기기에 탑재함으로써 로열티를 절약하고 통신사는 상품의 다양성을 증가시킬 수 있기 때문이다. 하지만 아직 기술과 시장이 발달하지 않은 상황에서 제조사는 상당한 비용과 위험을 감수해서라도 운영체제를 독자 개발하기로 결정할 수 있었으며, 굳이 운영체제가 아니더라도 키보드 모양, 제품 외형, 설치된 모바일 어플리케이션의 종류 등을 통해 자신의 상품을 차별화할 수 있었다. 이 때문에 초기 스마트폰 기술체계에서는 제조사가 가장 영향력 있는 위치에 있었다고 할 수 있다. 하지만 안드로이드 운영체제와 모바일 어플리케이션이 기술체계에서 차지하는 지위가 변동됨에 따라 이러한 행위자들

---

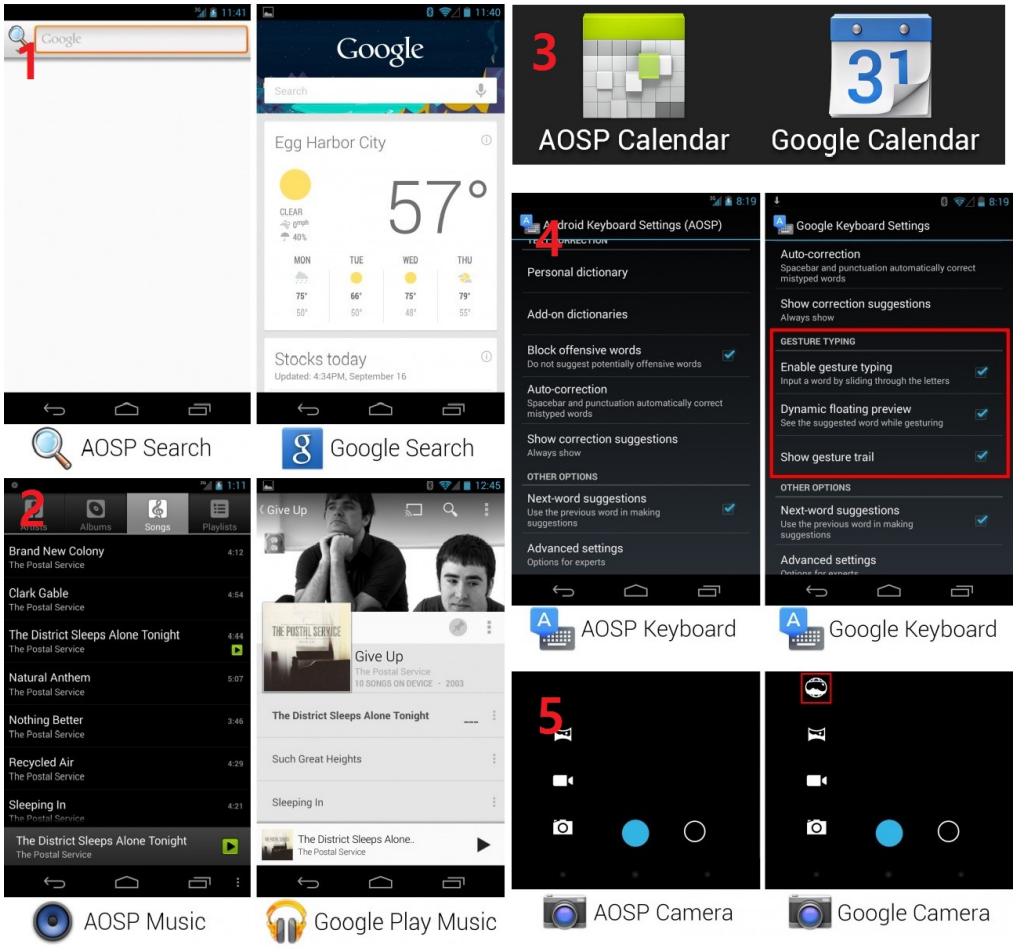
93) 애플사가 개발한 스마트폰인 아이폰(iPhone)은 'OS X' 혹은 'Mac OS X'라고 불리는 운영체제를 탑재하고 있었으나, 애플사는 2008년 5월 아이폰에 탑재되는 운영체제를 'iPhone OS' 다시 이름 붙였다. 그리고 2010년 6월에는 이 운영체제에 다시 'iOS'라는 새로운 이름을 붙였다.

94) 2009년 심비안 재단(Symbian Foundation)의 출범과 함께 심비안은 오픈 소스인 이클립스 공공 라이선스(Eclipse Public License)로 공개된다. 그러나 심비안의 많은 부분에 더욱 제한된 라이선스인 심비안 공공 라이선스(Symbian Public License)가 부여되었으며 전체 소스 코드는 재단에 소속된 회원사에게만 공개되었다.

의 지위도 변동되었다.

우선 안드로이드 운영체제에서 기본적으로 제공되는 기능들의 비중은 점차 축소되고 있다. 안드로이드 운영체제는 '공용' 어플리케이션을 기본적으로 포함하고 있지만 이러한 공용 어플리케이션 (혹은 번들(bundle) 어플리케이션)의 기능 개선은 상용 어플리케이션에 비해 상대적으로 정체되어 있다. <그림 24>는 AOSP 버전의 안드로이드 운영체제에서 기본으로 제공되는 기능들(검색, 음악 재생, 달력, 키보드 입력, 카메라 기능)과 그와 동일한 작업을 수행하는 구글의 어플리케이션을 비교한 것이다. 1번의 비교에서 구글의 어플리케이션은 안드로이드 운영체제의 기본 기능에서는 제공하지 않는 음성 검색과 위치정보기반 검색을 제공하고 있다. 이 밖에도 구글의 검색은 음악 검색, 텍스트 음성 변환, '구글나우(Google Now)'[95]와 같은 기능을 포함하고 있다. 2번에서 '구글플레이뮤직(Google Play Music)'이라는 어플리케이션은 안드로이드 운영체제의 기본 음악 재생 기능에 비해 정보를 더욱 다채로운 방식으로 표시한다는 것을 확인할 수 있다. 뿐만 아니라 구글플레이뮤직은 구글의 온라인 음악 저장소 및 구글의 음원판매처와 연동되기 때문에 사용자가 듣고 싶은 음악을 언제든지 온라인에서 다운로드 받아 재생할 수 있는 반면 안드로이드 운영체제의 기본 기능에서는 이러한 것들을 제공하지 않는다. 3번에서는 구글의 달력 어플리케이션과 안드로이드 운영체제의 기본 달력 기능을 비교하고 있다. 이 그림에서 제시된 아이콘 그림만으로는 큰 변화가 관찰되지 않지만, 사실 안드로이드 운영체제의 기본 달력 어플리케이션은 구글의 달력 기능으로 완전히 '대체'되었으며, 구글의 달력 기능은 기본 기능과 달리 코드가 공개되어 있지 않은 사유화된 기술이다. 4번은 안드로이드 운영체제의 기본 키보드 입력 기능과 구글의 키보드 입력 어플리케이션인 '구글키보드(Google Keyboard)'를 비교하고 있다. 구글키보드는 기본 기능에서 제공되지 않는 기능들을 제공하고 있는 것을 알 수 있다.[96] 5번은 안드로이드 운영체제의 기본 카메라

95) 구글 나우는 구글이 제공하는 서비스로 개인 비서와 같은 역할을 한다. 이 기능은 해야 할 일, 가야할 곳, 교통 상황 등 개인의 일정을 관리하고, 관심사, 뉴스, 정보를 지속적으로 확인하며, 대중교통, 환율, 주변 명소, 이벤트 등 현지 정보를 검색하는 등의 기능을 가지고 있다. (참조: 구글 나우 홈페이지. http://www.google.com/intl/ko/landing/now/)



<그림 24> 안드로이드 기본 기능과 구글 어플리케이션의 기능 비교. 안드로이드 버전 4.3 기준 (Amadeo, 2013.10.21.)

기능과 구글의 카메라 어플리케이션을 비교하고 있다. 그림에 표시된 것처럼, 구글 카메라는 기본 기능에서 제공하지 않는 포토스피어(Photosphere)[97], 소셜네트워크 공유 기능 등을 포함하고 있다. (Amadeo, 2013.10.21.)

이와 같은 안드로이드 운영체제 기본 기능의 '상대적 열화'는 특히 안드

96) 이 비교에서 구글키보드는 기본기능에서는 제공하지 않는 동작타이핑(gesture typing) 기능을 제공한다. 이 기능은 스와이프(swype)로도 불리며, 사용자는 키보드를 하나씩 누를 필요가 없이 키보드 사이를 미끄러지듯 지나가는 것 만으로 단어를 입력할 수 있다.

97) 포토스피어 기능은 사용자가 사진 촬영 시 스마트폰을 상하좌우의 방향으로 움직여서 파노라마 사진을 만들 수 있도록 해준다. (출처: 포토스피어 기능 소개 페이지. (https://www.google.co.kr/intl/ko/maps/about/contribute/photosphere/. 2014년 6월 21일 방문)

로이드 운영체제의 생산 구조를 구글이 지배하고 있다는 점에서 중요하다. 안드로이드 운영체제의 변화에 가장 큰 영향을 행사하는 구글이 운영체제의 세부 기능을 더욱 발전시키지 않고 '방치'하거나, 자사의 어플리케이션으로 대체하고 있기 때문이다. 특히 구글이 안드로이드 운영체제 기본 기능(사진, 달력, 메신저, 검색, 카메라 등)의 대부분에 대응되는 상용 어플리케이션들을 보유하고 있고, 이러한 어플리케이션들에는 안드로이드 운영체제의 기본 기능과는 대조적으로 최신 기능과 디자인을 적용하고 있다는 점에서 이러한 방치의 '의도성'을 확인할 수 있다. 즉 공적이고 개방적인 기술은 정체되어 있는 반면 사적이고 폐쇄적인 구글 어플리케이션은 개선, 발전되고 있으며, 이러한 공적 기술의 상대적 열화는 구글의 전략에 의한 것이다. 이 전략은 개방성, 공공성을 통해 기술을 확장시키면서, 동시에 공적인 기술을 버려진 기술(abandonware), 닫힌 기술, 죽은 기술로 만들어 사적인 기술을 최대한 확장하려는 목적을 가지고 있다. (Amadeo, 2013.10.21.)

안드로이드 운영체제의 기술적 방치로 인해 모바일 어플리케이션의 중요성이 부각된다. 안드로이드 운영체제에 포함된 기본 기능이 지속적으로 수정, 개선되지 않기 때문에 제조사나 사용자가 모바일 어플리케이션을 설치하여 스마트폰 기술을 보충, 확장해야하기 때문이다. 특히 제조사는 스마트폰 기기에 안드로이드 운영체제를 탑재하는 것만으로는 사용자에게 최신의 기능과 디자인을 제공하기 어려워진다. 비용을 들이지 않고 안드로이드 운영체제를 탑재했더라도 추가적으로 어플리케이션을 개발, 구입하거나 사용자가 어플리케이션을 자체적으로 설치할 수 있는 어플리케이션 마켓을 제공해야 하는 것이다.

이러한 현상은 기술적 사영화(technological privatization)라고 할 수 있다. 이는 기술체계에서 공적 기술 요소가 의도적으로 약화, 축소되고, 이로 인해 사적 기술 요소가 강화, 증가하는 과정이며 이는 개인화(customization) 또는 기술적 공동화와는 구별된다. 우선 기술적 사영화는 체계건설자의 전략적 선택으로 인해 다른 행위자의 역할, 기능에 변화가 일어나는 반면, 개인화는 사용자가 자신의 취향이나 선호에 따라서 기술을

변형, 조합하는 자발적인 과정이다. 또한 기술적 사영화에서 공적 기술 요소가 의도적으로 약화되는 것에 비해, 공동화는 기술체계에 요구되는 중립성으로 인한 공적 기술 요소의 부재를 말한다.

(2) 행위자의 지위 변동

스마트폰 사용에서 모바일 어플리케이션의 비중이 늘어남에 따라, 제조사가 기기를 통한 차별화의 요인은 줄어들게 된다. 어플리케이션의 확장과 다양화에 따라, 사용자는 자신이 선택, 이용하는 어플리케이션의 종류에 따라 스마트폰의 기능을 적응시키기 때문에, 개인화는 '과정의 문제'로 변화되었다. 이는 핸드폰의 기능이 기기의 디자인과 자체적 기능에 대한 '선택의 문제'였던 것과 비교할 수 있다. 이는 모바일 기술사용의 초점이 하드웨어 혹은 펌웨어에서 어플리케이션이라는 소프트웨어로 변화되었다는 것을 의미하며, 이에 따라 전자의 생산에 초점을 맞추고 있는 제조사의 영향력은 축소되었다. 안드로이드 운영체제의 버전에 따른 스마트폰 기기 형태의 특성에서도 이러한 변화를 확인할 수 있다. <그림 25>에서 첫째 줄은 안드로이드 운영체제의 도넛(Donut)버전을 탑재하고 있는 스마트폰, 두 번째 줄은 이클레어(Eclair)를 탑재한 스마트폰, 셋째 줄은 아이스크림샌드위치(Ice Cream Sandwich, ICS)를 탑재한 스마트폰, 넷째 줄은 젤리빈(Jellybean)을 탑재한 스마트폰이다.[98] 이들의 운영체제는 도넛과 이클레어는 2009년 하반기에 발표되었으며, ICS와 젤리빈은 각각 2011년 하반기, 2012년 하반기에 출시되었다. 사례로 주어진 스마트폰은 도넛과 이클레어, ICS와 젤리빈의 두 집단으로 분리하여 살펴보면, 시간이 지남에 따라 스마트폰 형태의 변화 경향을 확인할 수 있다. 구체적으로 살펴보면 스마트폰

98) 본 자료는 안드로이드 홈페이지에서 공개된 안드로이드 운영체제 버전 별 스마트폰 목록에서 각 제조사 당 하나씩의 스마트폰 기기를 임의로 추출한 것이다. 각 스마트폰의 제조사와 모델명은 다음과 같다. (가장 좌측 상단부터 오른쪽으로. 괄호 안은 모델명): 소니에릭슨(Xperia X8), 샤프 (LYNX SH-10B), HTC(매직), REGZA(IS04), 삼성(Gem), 모토롤라(SPICE), NEC(Smartia), LG(Axis), 에이서(beTouch E140), 에이수스(A50), 삼성(갤럭시 S3), 샤프(PANTONES5 SoftBank 107SH), 샤프(AQUOS ISW16SH), 팬택(베가 레이서2 IM-A830K), 에이수스(PadFone), HTC(One S), HP(Slate 7), 소니에릭슨(Xperia Z), 교세라(Hydro Edge), LG(Nexus 4)

<그림 25> 스마트폰 기기의 형태 비교.
(출처: www.android.com/devices, 2014년 2월 11일 방문)

하드웨어 디자인 요소였던 물리 키보드(physical keyboard)와 물리 버튼(physical button)은 사라지고, 터치 방식으로 동작하는 소프트 키보드(soft keyboard)와 소프트 버튼(soft button)으로 대체되었다. 도넛과 이클레어의 경우에는 스마트폰 외형과 버튼 모양과 개수에서도 어느 정도의 다양성이 보였지만, ICS와 젤리빈의 경우 스마트폰 외형이 거의 동형화되고 버튼의 개수는 3개, 버튼의 모양도 유사해진 것을 볼 수 있다.

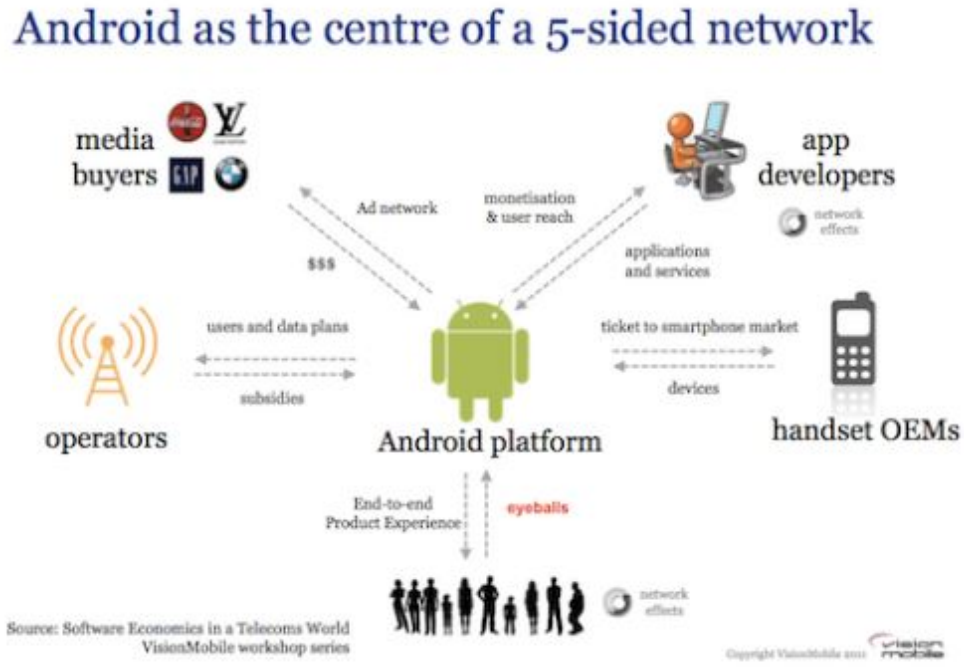
또한 어플리케이션의 확장과 다양화는 제조사에 대한 구글의 영향력을 강화시켰다. 모바일 어플리케이션의 수가 적아 구글의 어플리케이션 마켓의 비중이 낮을 때는 제조사가 구글의 승낙을 받고 마켓을 설치할 유인이 많지 않았던 반면, 어플리케이션의 수가 증가하고 이에 대한 수요가 늘어나면서 이러한 승낙은 필수적인 것이 되었기 때문이다.

이러한 요인에 따라 안드로이드 체계를 이루고 있는 행위자들의 관계망은 휴대폰을 중심으로 한 관계망과는 다른 양상을 보이고 있다. 기존 관계망에서 가장 큰 영향을 미치고 있었던 것은 이동통신사업자와 제조사였다. 통신사업자는 사용자가 사용할 수 있는 휴대폰의 종류를 결정하고, 제조 과정에도 영향을 주었다. 통신사업자와 제조사는 카르텔을 형성하였고 이들에 의해 개발된 휴대폰의 운영 체제는 폐쇄적이었으므로, 소프트웨어 개발자가 속할 곳은 없었다.

오픈소스로서 많은 행위자가 자유롭게 운영체제를 이용할 수 있기 때문에, 안드로이드는 하나의 '생태계'를 형성하였다. 안드로이드 생태계는 안드로이드와 관련한 수많은 행위자들의 관계망이라고 할 수 있다. 이러한 관계망 속에는 어플리케이션 개발자, 스마트폰 기기 제조사, 사용자, 망 운영자 등이 있으며, 구글과 안드로이드 운영체제는 <그림 26>와 같이 중심적인 위치를 차지하고 있는 것으로 확인되고 있다.

안드로이드 운영체제 개발 이전에도 구글이 컴퓨터 검색 시장에서 절대적인 위치를 차지하긴 했으나, 전체 규모나 수익의 측면에서 다른 기업을 압도할 수 있는 위치에 있지는 않았다. 구글이 안드로이드 개발사를 인수하던 시점이나, 안드로이드 운영체제가 공개되고 이 운영체제를 탑재한 첫 번째 스마트폰이 출시되는 시점까지도 구글의 시가 총액은 마이크로소프트의 50%에도 미치지 못했다. 하지만 2012년 구글의 시가 총액은 마이크로소프트의 90% 수준까지 육박했다. 따라서 구글이 풍부한 자본이나 기존의 시장 지배력만을 이용해 안드로이드 체계를 확장했다 보기는 어렵다.

또한 구글은 안드로이드 운영체제나 그 저작권을 판매하여 직접적인 수익을 올리지 못한다. 구글은 안드로이드의 개발, 제공을 전담하고 있지만, 정작 안드로이드의 이용, 수정, 재배포를 막지 못하고 운영체제나 그 라이선스 및 특허권을 판매하여 수익을 얻지도 못한다. 이는 구글이 정보기술 시장에서 지배적인 사업자가 아니었을 뿐만 아니라 안드로이드 운영체제를 이용한 직접적 수익 활동을 통해 구글의 권력이 증가된 것도 아니라는 것을 말해준다. 기술을 통한 권력의 확대, 재생산이 없었다면 구글이 기존의 관계를 변화시키고 '생태계' 내에서 권력 관계의 정점에 위치하기는 어려웠



<그림 26> 안드로이드 체계의 관계망 (Constantinou, 2011.2.11.)

을 것이다.

3. 기술의 지배를 통한 권력의 확장

(1) 기술체계 내 지위에 따른 상호작용 방식

기술체계가 항상 안정적으로 유지되는 것은 아니다. 특히 개방적인 안드로이드 체계에는 여러 개인, 기업이 제약 없이 참여할 수 있기 때문에 체계를 약화, 붕괴시키는 위험요소가 등장할 가능성은 더욱 높다. 체계의 붕괴는 사용자가 어플리케이션을 사용할 수 없거나 제한적으로 사용하게 되는 것을 의미하며, 이러한 붕괴를 막기 위해 체계건설자인 구글은 개발자와 제조사를 통제해야 한다.

1) 모바일 어플리케이션에 대한 방관

하지만 이러한 통제는 기술적 요소의 이탈을 불러올 수 있다. 행위자-연결망 이론에 따르면 이러한 이탈은 중심행위자의 관심끌기(interessment)가 실패한 것으로, 구글이 기업 또는 개인에게 '안드로이드 개발자'라는 정체성을 부여하지 못하고 연합의 안정성이 파괴된 경우이다(Callon, 1986). 이러한 이탈은 제조사 보다는 개발자에게 발생할 가능성이 크다. 안드로이드 어플리케이션 개발자는 개발, 유지하는 기술체계의 크기가 스마트폰 제조사나 구글에 비해서 상대적으로 작기 때문에, 상대적으로 쉽게 기술체계를 다른 운영체제에 맞게 변환시킬 수 있다. 개발자는 안드로이드에 한정되지 않으며, 애플사의 iOS, 심비안(symbian), 윈도우 모바일(Windows Mobile), RIM 등 다른 운영체제에 기반한 어플리케이션을 개발할 수 있다. 이와 같이 개발자가 유지하고 있는 하위기술체계의 크기가 작고 그 기술적 모멘텀(Technological momentum)이 상대적으로 약하기 때문에, 개발자는 적은 자원만을 가지고도 빠른 속도로 기술체계를 변화시킬 수 있다. 또한 다른 운영체제도 어플리케이션의 수적 증가에 초점을 두고 있기 때문에, 이러한 전환은 더욱 가속화된다.

개발자의 이탈로 인한 체계 붕괴의 위험성이 개발자의 행위에 의한 위험

성과 대치되는 상황에서 구글의 선택은 전자를 최소화하는 것이다. 즉 구글은 개발자의 이탈을 막고 어플리케이션의 수적 증대를 촉진하는 방식으로 체계를 정렬하고 있다. 개발자가 운영체제의 사용자 수나 어플리케이션 마켓의 특성, 개발의 편의성 등을 고려하여 운영체제를 선택한다. 이 때문에 구글은 다양한 안드로이드 어플리케이션을 확보하기 위해서 어플리케이션 마켓에 등록되는 어플리케이션에 대한 검열을 실시하지 않는다. 특히 어플리케이션의 수가 미약한 수준에 불과했던 초기에는 어플리케이션 개발을 장려하기 위한 경진대회를 세계적인 수준에서 개최하기도 하였다. 또한 구글은 다른 언어로 만들어졌거나 다른 운영체제를 대상으로 개발된 어플리케이션을 안드로이드의 환경에 맞게 변환시킬 수 있는 도구들을 제공한다. 반면 구글은 불법적, 악의적 어플리케이션에 의한 위험성은 별도로 통제하고 있지 않다.[99]

2) 제조사에 대한 통제의 강화

반면 제조사는 개발자와 비교하여 구글의 기술체계를 이탈하기 쉽지 않다. 우선 제조사는 스마트폰 기기를 만들기 위해 개발자 보다 상대적으로 큰 수준의 기술체계를 유지해야 한다. 개발 전의 기획, 설계 단계에서부터 구글과의 라이선스 협의 및 운영체제의 수정, 보완, 스마트폰의 배포 및 마케팅에 이르기 까지 하나의 스마트폰이 출시되기 까지 제조사는 하드웨어에서 소프트웨어에 이르는 다양한 기술적 요소들을 조율해야 한다. 이 때문에 제조사의 기술체계는 개발자의 그것에 비해 진화 및 변화의 속도가 낮다. 또한 제조사 간에는 출시 시기와 개발일정을 둘러싼 경쟁이 존재하기 때문에, 한 제조사가 안드로이드 스마트폰을 개발하기로 결정했다면 개발 과정에서 운영체제나 하드웨어 구성을 바꾸는 것은 개발자가 어플리케이션의 기능을 변경하는 것은 기업에 부담을 초래한다.

또한 제조사는 구글과 단순히 경제적인 요인으로만 연결되지 않는다. 제

99) 하지만 이러한 전략은 검증되지 않은 여러 어플리케이션이 마켓에 등록될 수 있게 하였으며, 일부는 안드로이드 어플리케이션의 품질이 매우 낮아 안드로이드 마켓은 마치 '기술적 빈민가' 같다고 비판하기도 한다. (Derene, 2011.12.13.)

조사와 구글은 다면적인 관계를 맺고 있다. 구글은 제조사에 직원을 파견하거나 운영체제 개발과정에 제조사를 참여시키는 등의 방법으로 운영체제의 개발에 필요한 자원을 지원하기도 한다. 또한 구글은 다양한 기업과 연계관계를 맺고 있기 때문에, 구글과의 마찰은 이러한 연계 기업과의 관계에도 악영향을 미칠 수 있다. 만약 제조사가 한 제품의 생산 과정에 있어서 구글과 어떠한 방식으로든 마찰을 일으켰다면, 이는 추후 다른 제품의 개발 과정에 영향을 미칠 수 있다. 그리고 구글은 검색, 동영상 서비스, 지도, 전자 거래, 광고 등 정보통신의 다양한 분야에서 서비스를 벌이고 있기 때문에, 제조사는 이러한 서비스들과의 관계도 고려해야 한다.

더군다나 앞장에서 살펴본바와 같이, 어플리케이션의 확장으로 스마트폰 영역에서 제조사의 지위는 약화되고 있다. 제조사는 운영체제와 어플리케이션, 어플리케이션 마켓 등 소프트웨어에 종속되고 있으며, 기존의 체계에 속하지 않고 시장에 진입하거나 제품을 개발하는 것은 극도로 어려워지고 있다. 모토롤라나 팬택과 같은 제조사들은 아예 안드로이드 운영체제 만을 사용하겠다고 대외적으로 밝히고 있다. 또한 삼성, LG, 에이서, 에이수스와 같은 기업들도 출시하는 대부분의 스마트폰에 안드로이드 운영체제를 탑재하고 있다.

제조사의 이탈 위험이 전무한 것은 아니다. 아직 우분투(Ubuntu)나 타이젠(Tizen)과 같은 대안들이 존재하나 안드로이드 스마트폰과 아이폰으로 양분된 시장에서 이러한 운영체제를 탑재했을 경우 많은 판매고를 올리기 어렵다. 뿐만 아니라 이들 운영체제에서 사용가능한 어플리케이션의 수는 안드로이드에 비해 매우 미미한 수준이다.

(2) 기술을 통한 권력의 확장

1) 권력으로서의 정합성: 기술적 기준의 정치화

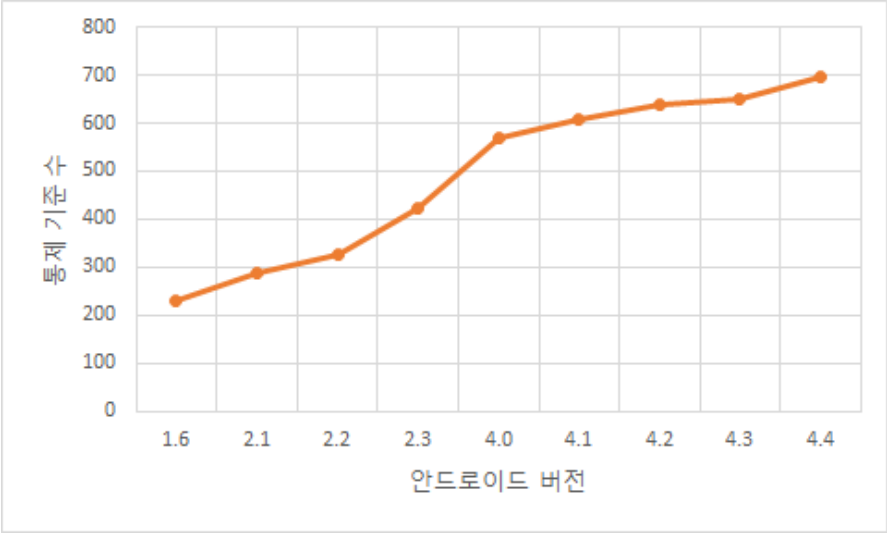
구글이 스마트폰 기기 영역 혹은 스마트폰 기기의 하위체계를 통제하는 주요 수단은 '기술적 정합성'이다. 앞서 기술되었듯 구글은 정합성 프로그

유형	세부 유형	안드로이드 버전 번호								
		1.6	2.1	2.2	2.3	4.0	4.1	4.2	4.3	4.4
필 수 항목	must	152	210	220	272	376	389	414	426	456
	required	11	4	6	11	33	39	40	40	43
	shall	0	0	0	0	0	0	0	0	0
	합계	**163**	**214**	**226**	**283**	**409**	**428**	**454**	**466**	**499**
권 장 항목	should	28	30	47	65	76	90	93	95	106
	recommended	0	0	0	0	2	2	2	2	3
	may	35	43	48	64	73	75	78	76	78
	optional	3	1	4	9	9	13	12	10	10
	합계	**66**	**74**	**99**	**138**	**160**	**180**	**185**	**183**	**197**
총합		229	288	325	421	569	608	639	649	696
1.6 버전 대비 증가율 (%)		-	**26**	**42**	**84**	**150**	**166**	**179**	**183**	**204**

<표 19> 안드로이드 운영체제 버전별 정합성기준의 유형과 수

램을 통해 제조사가 생산하는 스마트폰 기기의 구체적인 특성을 통제한다. 하지만 정합성의 기준은 고정되어 있지 않다. 안드로이드 운영체제가 5년여 동안 44번의 변경, 개선을 거치는 과정에서 정합성정의문서는 8차례에 거쳐 변경되었다. 안드로이드 버전별 정합성기준을 비교해보면, <표 19> 및 <그림 27>의 결과를 얻을 수 있다. 우선 <표 19>은 정합성기준을 만족시키기 위해서 제조사가 만족시켜야 하는 기준의 수를 나타내고 있다. 정합성기준은 그 중요도에 따라서 필수와 권장으로 나눌 수 있다. 이 중 필수 항목은 'must', 'required', 'shall' 또는 'must not', 'shall not' 이란 표현을 포함하며 절대적으로 필요하거나 혹은 절대적으로 금지하는 기술 사항을 나타낸다. 반면 권장항목은 해당 항목이 특정한 환경에서는 무시되거나('should', 'recommended') 받아들여질 수 있는('should not', 'not recommended') 정당한 이유가 존재할 수도 있음을 의미한다.[100] 구글은 정합성정의문서의 개정 시마다 단 한 번의 예외도 없이 제조사가 만족시켜야할 필수, 권장 항목의 수를 증가시키고 있다. 필수, 권장 항목의 증가에 따라 2009년 9월 안드로이드 운영체제 버전 1.6을 사용하는 제조사가 만족시켜야할 필수, 권장 항목이 229개였던 것에 반해, 2010년 12월 공개된 2.3

100) 안드로이드 정합성기준에서 권장, 필수 항목과 관련되어 사용되는 용어의 의미는 인터넷국제표준화기구(Internet Engineering Task Force, IETF)의 기준을 따른다. (IETF, 1997)



<그림 27> 정합성기준의 필수, 권장 항목의 증가

버전에 대해서는 421개로 83.8% 증가하였고, 2011년 10월 공개된 4.0 버전에 대한 정합성 필수, 권장 항목은 569개로 148.4%, 2013년 10월에 공개된 4.4 버전에 대해서는 696개로 무려 203.9% 증가하였다.

이를 더욱 구체적으로 살펴보기 위해, 안드로이드 운영체제별 정합성기준을 항목별로 분석해보았다. <표 20>은 4개의 정합성기준에 존재하는 항목들을 분석한 것으로서, 시간이 지남에 따라 새로운 항목이 추가되고 있는 반면, 기존의 기준은 사라지지 않고 남아있어 전체적으로 항목의 수가 증가하는 것을 볼 수 있다. (추가된 각 항목을 4.4 버전을 기준으로 설명하면 <표 21>과 같다.) 예를 들어 그래픽 가속 기능에 대한 기준은 안드로이드 1.6 버전에는 존재하지 않았으나, 2.3 이후의 버전에는 존재한다. 그리고 스크린 유형과 스크린 기술에 대한 기준은 버전 2.3 까지도 존재하지 않았으며, '와이파이 다이렉트'와 'TDLS'에 대한 기준은 4.0 까지도 존재하지 않았다. 제조사가 정합성을 갖춘 기기를 만들기 위해서는 이와 같이 추가된 항목을 포함하여 더 많은 기준을 충족시켜야 하는 것이다.

정합성기준		안드로이드 버전			
항목	세부항목	1.6 (2009년 9월)	2.3 (2010년 12월)	4.0 (2011년 10월)	4.4 (2013년 10월)
소프트웨어 정합성					
하드 API 정합성		O	O	O	O
소프트 API 정합성	허가	O	O	O	O
	빌드 파라미터	O	O	O	O
	인텐트 정합성	O	O	O	O
네이티브 API 정합성		O	O	O	O
웹 정합성	웹뷰 정합성	O	O	O	O
	브라우저 정합성	O	O	O	O
API 동작 정합성		O	O	O	O
API 명칭공간		O	O	O	O
가상머신 정합성		O	O	O	O
사용자 인터페이스 정합성	바탕화면				O
	위젯	O	O	O	O
	알림	O	O	O	O
	검색	O	O	O	O
	토스트	O	O	O	O
	테마		O	O	O
	라이브 월페이퍼		O	O	O
	최근 어플리케이션 표시			O	O
	입력 관리			O	O
	잠금 화면 미디어 원격 조정				O
	드림				O
	위치				O
	유니코드				O
기기 등록				O	O
접근성				O	O
텍스트 투 스피치				O	O
어플리케이션 패키징 정합성					
		O	O	O	O
멀티미디어 정합성					
미디어 코덱		O	O	O	O
비디오 인코딩		O	O	O	O
비디오 디코딩		O	O	O	O
오디오 녹음		O	O	O	O

오디오 지연			O	O	O
네트워크 프로토콜				O	O
개발자 도구와 옵션 정합성					
개발자 도구		O	O	O	O
개발자 옵션					O
하드웨어 정합성					
디스플레이와 그래픽	스크린 특성	O	O	O	O
	디스플레이 메트릭	O	O	O	O
	스크린 방향성	O	O	O	O
	그래픽 가속		O	O	O
	레거시 어플리케이션 정합 모드		O	O	O
	스크린 유형			O	O
	스크린 기술			O	O
	외부 디스플레이				O
입력 장치	키보드	O	O	O	O
	비터치식 방향키	O	O	O	O
	방향키	O	O	O	O
	터치스크린 입력	O	O	O	O
	허위 터치 입력			O	O
	마이크			O	O
센서	가속계	O	O	O	O
	자력계	O	O	O	O
	GPS	O	O	O	O
	자이로스코프		O	O	O
	기압계		O	O	O
	온도계		O	O	O
	광센서		O	O	O
	근접센서		O	O	O
데이터 연결	전화	O	O	O	O
	와이파이	O	O	O	O
	와이파이 다이렉트				O
	와이파이 TDLS				O
	블루투스		O	O	O
	근거리 통신 (NFC)		O	O	O
	최소 네트워크 기능		O	O	O
카메라	후면 카메라	O (전후 구분없음)	O	O	O
	전면 카메라		O	O	O
	카메라 API 동작	O	O	O	O

	카메라 방향		O	O	O
메모리와 저장	최소 메모리와 저장공간		O	O	O
	어플리케이션 공유 저장공간		O	O	O
USB		O	O	O	O
성능정합성					
		O	O	O	O
보안 모델 정합성					
허가		O	O	O	O
사용자와 프로세스 격리		O	O	O	O
파일시스템 허가		O	O	O	O
대체 실행 환경			O	O	O
복수 사용자 지원					O
프리미엄 SMS 경고					O
커널 보안 사양					O
프라이버시					O
디스크 전체 암호화					O

<표 20> 안드로이드 버전별 정합성기준의 항목 비교
(O: 정의됨, 회색으로 채워진 항목: 정의되지 않음)

추가된 정합성기준의 항목 중 일부는 사용자의 정보를 보호하고 스마트폰 사용으로 인한 피해를 방지하기 위한 목적을 가지고 있다. ‘프리미엄 SMS 경고’, ‘커널 보안 기능’, ‘프라이버시’, ‘디스크전체 암호화’ 항목이 이와 관련된 항목으로, 유료 문자메세지의 발송 전에 사용자에게 과금 가능성을 경고하고, 사용자의 개인정보를 보호하는 등 정보기술의 이용과정에서 발생할 수 있는 사용자의 피해를 줄이도록 한다. 이러한 기능은 안드로이드 운영체제의 초기에는 포함되지 않았으나, 사용자의 증가와 스마트폰 사용 방식의 다양화에 따라 그 필요성이 커지고 있는 것들이다.

한편 ‘사용자 인터페이스 정합성’ 분류에 포함된 항목들도 지속적으로 증가했으며, 추가된 항목의 수도 상대적으로 많다. 이 분류에서는 스마트폰의 기본화면(home screen) 설정, 스마트폰의 바탕화면 설정, 사용자가 최근 사용한 어플리케이션을 표시하는 기능, 터치스크린을 통한 입력 기능, 스마

<소프트웨어 정합성>

- 바탕화면: 다른 어플리케이션의 바탕화면 대체를 허용 하려면 반드시 특정 변수를 선언해야 한다.
- 테마(Theme): 제조사는 특정 테마를 절대 변경해서는 안 된다. 제조사는 기본테마를 변경해도 된다. 제조사는 시스템 상태 아이콘에 반드시 흰색을 사용해야 한다.
- 라이브 월페이퍼(Live Wallpaper): 라이브 월페이퍼를 안정적으로 구동시킬 수 있는 기기에는 반드시 라이브월페이퍼가 구현되어야 하며, 안정적으로 구동시킬 수 없는 기기에는 절대 라이브 월페이퍼가 구현되면 안 된다.
- 최근 어플리케이션 표시: 제조사는 이 기능을 변경하거나 제거해도 된다.
- 입력관리: 특정 변수를 선언한 기기에는 반드시 제3의 입력 방식을 추가, 설정할 수 있는 방법이 구현되어야 한다.
- 잠금 화면 미디어 원격 조정: 잠금화면을 지원하고 사용자가 바탕화면에 위젯을 추가할 수 있는 기기는, 잠금 화면에 원격 조정을 삽입할 수 있게 해야 한다. (원격 조정은 화면 잠금 상태에서도 음악 재생기와 같은 어플리케이션을 조정할 수 있게 하는 기능이다.)
- 드림(Dreams)[101]: 기기에는 드림이라는 기능과 드림의 설정 기능이 반드시 구현되어야 한다.
- 위치: '설정'의 '위치' 메뉴에 '위치 모드'가 반드시 표시되어야 한다.
- 유니코드(Unicode): 기기에는 유니코드에 정의된 이모지(Emoji) 문자를 입력할 수 있는 방법이 반드시 구현되어야 한다.
- 기기등록: 기기에 특정 클래스(devicePolicyManager)가 반드시 구현되어야 한다.
- 텍스트투스피치(Text-to-Speech, TTS): 기기는 TTS와 관련된 요건들을 반드시 만족해야 한다.

<어플리케이션 패키징 정합성>

- 오디오 지연: 이 항목에는 음향 출력 형식과 음향 출력의 시간지연 상한선 등이 규정되어 있다.
- 네트워크 프로토콜: 이 항목에는 기기가 포함에야 하는 미디어 네트워크 프로토콜에 대한 규정이 포함되어 있다. (미디어 네트워크 프로토콜은 미디어의 전송에 사용되는 통신 규약이다.)
- 네트워크 프로토콜: 이 항목에는 기기가 포함에야 하는 미디어 네트워크 프로토콜에 대한 규정이 포함되어 있다. (미디어 네트워크 프로토콜은 미디어의 전송에 사용되는 통신 규약이다.)

<개발자 도구와 옵션 정합성>

- 개발자 옵션: 이 항목에는 개발자가 어플리케이션을 개발할 때 사용할 수 있는 설정

에 대한 규정이 포함되어 있다.

<하드웨어 정합성>

- 그래픽 가속: 이 항목에는 그래픽 출력을 위한 코드의 포함 여부를 규정하고 있다.
- 레거시 어플리케이션: 이 항목에는 특정 스크린 사이즈를 제공하는 모드에 대해 규정되어 있다.
- 스크린 유형: 이 항목에는 기기의 화면 해상도에 대한 규정들이 포함되어 있다.
- 스크린 기술: 이 항목에는 기기의 화면에서 지원해야 하는 기능에 대한 규정이 포함되어 있다.
- 허위 터치 입력: 이 항목에는 터치 입력이 없는 기기에서의 포인터 입력에 대해 규정되어 있다.
- 마이크: 기기는 마이크를 포함할 수 있다. 이 항목은 마이크의 기능에 대한 세부 기준이 포함되어 있다.
- 자이로스코프: 기기는 자이로스코프를 반드시 포함해야 한다. 이 항목에는 자이로스코프의 세부 기능이 규정되어 있다.
- 기압계: 기기는 기압계를 포함할 수 있다. 이 항목에는 기압계가 포함되었을 경우 이에 대한 세부 기능이 규정되어 있다.
- 온도계: 기기는 온도계를 포함할 수 있다.
- 광센서: 기기는 광센서를 포함할 수 있다.
- 근접센서: 기기는 근접 센서를 포함할 수 있다. 이 항목에는 근접센서가 포함되었을 경우 이에 대한 세부 기능이 추가로 규정되어 있다.
- 와이파이 다이렉트: 와이파이 다이렉트의 포함 여부와 기능에 대한 규정. (와이파이 다이렉트는 단말기들이 무선랜에서 기지국 역할을 하는 AP(Access Point)를 통하지 않고 직접 통신할 수 있도록 해준다.)
- 와이파이 TDLS: 와이파이 TDLS의 포함 여부와 기능에 대한 규정. (TDLS(직접링크 설정기법)은 단말기들이 AP를 통하지 않고 직접 통신할 수 있도록 해주며, 이 때 통신의 품질을 보장한다.)
- 블루투스: 이 항목에는 기기의 블루투스 통신 제공 여부와 세부 기능에 대한 설정이 포함되어 있다. (블루투스는 스마트폰과 이어폰, 오디오 등 다른 기기를 무선으로 연결하는 통신 방식이다.)
- 근거리통신: 이 항목은 기기의 근거리통신 제공 여부와 세부 기능에 대한 설정에 관한 것이다. (근거리통신은 매우 가까운 거리에서 동작하는 무선 전송 방식이다.)
- 최소네트워크 기능: 기기는 반드시 하나 이상의 데이터 통신을 지원해야 한다.
- 전면카메라: 기기는 전면 카메라를 포함할 수 있다. 이 항목에는 전면카메라에 대한 세부 기능이 규정되어 있다.
- 카메라 방향: 이 항목에는 스마트폰의 방향에 따라 카메라를 회전하는 기능에 대해

규정되어 있다. • 최소메모리와 저장공간: 기기가 포함에 할 수 있는 메모리와 저장공간의 최소 용량 대해 규정되어있다. • 어플리케이션 공유 저장: 기기는 반드시 어플리케이션을 위한 공유 저장소를 제공해야 한다. 이 항목에는 이 저장소에 대한 세부 규정이 포함되어 있다. **<보안 모델 정합성>** • 대체실행 환경: 기기는 안드로이드 운영체제가 기본적으로 제공하는 어플리케이션 실행 방식 이외에 추가로 다른 방식을 제공할 수 있다. 이 항목에는 기타 실행 방식에 대한 세부 규정이 포함되어 있다. • 다중 사용자 지원: 안드로이드 운영체제에서는 여러 명의 사용자가 각기 다른 계정을 사용할 수 있다. 본 항목에는 이 기능에 대한 세부 규정이 포함되어 있다. • 프리미엄 SMS 경고: 안드로이드는 프리미엄 SMS에 대해 경고하는 기능을 지원한다. 기기는 미리 지정된 번호들에 대해서는 SMS를 보내기 전 사용자에게 반드시 경고해야 한다. • 커널 보안 기능: 안드로이드가 제공하는 보안 시스템의 사용에 대한 규정이 포함되어 있다. • 프라이버시: 화면의 내용이나 출력되는 음향을 기록하는 기능이 구현되었다면, 이 기능이 활성화었을 경우 기기는 반드시 사용자에게 이 사실을 알려야 한다. • 디스크 전체 암호화: 기기에 잠금화면이 구현되어 있으면, 기기는 반드시 디스크전체 암호화를 지원해야 한다.

<표 21> 추가된 정합성기준 항목들의 내용

트폰의 잠금 화면 또는 스마트폰을 충전하는 도중에 모바일 어플리케이션을 조정하는 기능 설정 등에 관한 항목이 새로이 신설되었다. 이러한 항목들에 의해 권고, 강제되는 설정 및 기능은 주로 스마트폰 사용자가 안드로이드 운영체제를 이용하면서 직접적으로 다루게 되는 부분들이다. 정합성정의문서에 사용자 인터페이스 관련 규정의 증가는, 안드로이드 운영체제의 발전이 이러한 부분에 집중하여 이루어지고 있으며, 구글이 제조사의 변형에도 불구하고 안드로이드 운영체제와 사용자 간의 상호작용을 동일하게 통제하려 한다는 것을 말해준다.

뿐만 아니라 정합성정의문서의 개정에 의해서 새로 추가된 것은 아니지만 제조사에게 더욱 상향된 기준을 요구하는 된 항목도 있다. 예를 들어

101) 드림(dreams)는 사용자가 충전 상태인 기기의 어플리케이션을 사용할 수 있도록 하는 인터렉티브 화면보호기이다.

2.3 버전의 안드로이드 운영체제를 탑재하려는 기기는 최소 128MB의 메모리 공간을 갖추어야 했는데 비해, 4.4 버전의 기기는 최소 340MB를 확보해야 한다. 기기가 갖추어야 할 최소 저장 공간의 경우도 2.3 버전의 경우 150MB 이었는데 비해, 4.4 버전에서는 1GB로 증가하였다.[102] 이는 새로운 안드로이드 운영체제에 대한 정합성을 획득하기 위해서는 제조사가 더 큰 메모리와 저장공간을 가진 기기를 만들어야 한다는 것을 의미한다.

일견 이러한 강제는 구글플레이스토어를 탑재하기 위해 충족해야 하는 기준이지, 안드로이드 운영체제를 탑재하기 위해 충족해야 하는 기준이 아니기 때문에 문제가 없다고 말할 수도 있다. 이러한 주장에 의하면 정합성 기준을 문제시 하는 것은 사기업의 전략에 대한 지나친 간섭이 될 수 있다. 하지만 앞서 설명되었듯 안드로이드 체계에서 공적 기술과 구글의 사유화된 기술 간의 경계는 불명확하며 안드로이드 운영체제, 어플리케이션, 기기의 생산에 구글이 절대적인 영향을 미치고 있기 때문에, 안드로이드 운영체제(또는 개방적 기술체계 전체)에서 구글플레이스토어(또는 구글의 기술적 의존구조)를 분리하고 각각에 '자유, 선택'과 '기술적 진보를 위한 통제'의 상반된 가치를 설정하는 것은 기만적이다. 오히려 구글플레이스토어라는 기술을 기점으로 상반된 가치와 상호작용 방식의 설정하는 것은 구글의 전략이다. 앞서 설명되었듯, 구글은 라이선스의 선택, 어플리케이션 개발 등에 있어 제조사와 개발자들의 '자유와 선택'을 존중한다고 밝히고 있다. 정보기술이 동작하기 위해서는 기술적 요소 간의 정합성이 확보되어야 한다는 점을 고려해도, 이러한 '자유와 선택'이 실질적으로 존중되려면 정합성은 최소 수준으로 유지되어야 하고, 권고와 강제 항목 또한 최소한으로 유지되거나 점차 축소되어야 한다. 하지만 구글은 체계의 개방성을 내세우면서도 행위자와 기술체에 대한 통제를 증가시키는 모순을 보여주고 있다.

이와 같이 정합성정의문서의 변화과정을 통해서 확인되는 것은, 정합성 기준이 기술 간 정상적인 상호작용을 위한 최소한의 규정에 그치지 않고

102) 하나의 음악 파일이 평균 3MB의 크기를 가진다고 가정하였을 때, 150MB의 저장공간에는 약 50곡의 음악을 저장할 수 있는 반면, 1GB의 저장공간에는 약 330곡의 음악을 저장할 수 있다.

다른 기술적 행위자의 생산 과정, 생산물을 체계건설자의 목표, 의도에 합치하는 방향으로 통제하려는 목적을 가진다는 점이다. 기준을 설정하여 정합성을 정의하는 것은 구글이며, 구글에 의해 정의된 정합성을 만족시키는 기술과 행위자만 구글의 연합에 편입될 수 있다. 정합성기준을 통해 구글은 안드로이드 '생태계'를 자신의 정치적, 경제적 의도에 따라 재조합, 동원하며, 이 과정에서 행위를 제약하고 생산물을 통제하며 권력을 확장한다. 또한 구글이 정합성의 기준을 변화시키고 더 다양하고 세부화된 기준들을 제시함에 따라 제조사가 연합에 포함되는 것은 더욱 어려워지며, 어플리케이션과 사용자의 증가와 함께 정합성기준의 임의적 변동은 구글에 대한 제조사의 종속을 더욱 심화시킨다. 안드로이드 한 고위 엔지니어는 구글이 정합성을 "제조사가 구글이 원하는 것을 만들도록 하는 클럽"으로 이용하고 있다고 말했다[103](Patel, 2011.5.12.).

2) 권력으로서의 안정성: 기술체계의 경계 작업

대만에 기반을 둔 모바일 기기 제조사인 에이서는 다수의 안드로이드 스마트폰을 출시해온 제조사로서 개방핸드셋연합의 일원으로 AOSP에서 안드로이드 운영체제의 개발과정에도 참여하고 있었다. 하지만 에이서(Acer)는 구글의 '협박'에 의해 한 스마트폰 출시를 포기하였다. 에이서가 출시하려던 스마트폰에는 중국 알리바바(Alibaba)사가 안드로이드 운영체제를 변형시켜 개발한 알리윤(Aliyun)이란 운영체제가 탑재될 예정이었다. 이에 대해 구글의 임원은 알리윤이 정합성을 만족시키지 못하였다("does not succeed in being compatible")고 비판하였다. 알리바바의 한 임원은, 구글이 에이서에게 안드로이드와 관련된 협력과 라이센싱의 중단을 경고했다고 밝혔다(Cheng, 2012.9.13.).

이 사례에서 주목할 점은 알리바바 사가 구글의 기준을 만족시킨 운영체제를 개발하여 안드로이드 체계에 편입하고자 것이 아니었다는 점이다. 만약 알리바바 사의 목적이 '편입'이었다면, 구글은 구글플레이스토어의 탑재

103) 원문은 다음과 같다. "obvious to the OEMs that we are using compatibility as a club to make them do what we want."

를 금지하는 것만으로도 충분했을 것이다. 알리윤은 안드로이드 운영체제의 코드를 변형하고 새로운 어플리케이션들을 만들어 전혀 새로운 기술체계를 구축하려고 하였다. 그리고 이것이 성공하면 안드로이드 체계는 새로운 경쟁 체계를 맞이하여 사용자, 개발자, 제조사, 어플리케이션, 스마트폰을 빼앗길 위기에 처하게 된다. 이러한 상황에서 구글은 단지 하나의 스마트폰을 금지하는 것을 넘어서, 에이서를 안드로이드 체계에서 추방시키겠다고 협박한 것이다.

구글의 임원이 '정합성'을 통해 설명했지만, 이 사례를 정합성의 차원에서는 설명하기는 어렵다. 구글이 문서를 통해 정합성을 통제하고 있지만, 정합성을 만족시키는 기기를 개발하고 이를 검증하여 구글플레이스토어의 사용 허락과 안드로이드 체계에 대한 편입을 요청하는 것은 제조사의 역할로 규정되어 있기 때문이다. 그리고 구글에게는 정합성기준을 만족시키지 못하더라도 타 제조사의 기기의 생산을 금지할 권한이 없다. 이 사례에서 구글이 에이서 사의 스마트폰 생산을 취소할 수 있었던 것은, 기술적 요소의 통합, 공식화 과정을 지배하여 '안드로이드 운영체제'가 무엇인지를 규정할 수 있는 권한이 구글에게만 있기 때문이다. 구글이 통제하고 있는 안정성은 다른 행위자의 행위를 통제하는 수단으로 사용되고 있다. 특히 이 사례에서 권력으로서의 안정성은 기술 내부에 대한 통제에서 더 나아가 다른 기술체계의 형성을 가로막는 목적도 갖고 있다.

그리고 기술체계 내부의 행위자에게 구글이 행사하는 권력은 더욱 강력해지고 있다. 구글과 제조사 사이의 체결된 '모바일 어플리케이션 유통 합의(Mobile Application Distribution Agreement, MADA)'를 통해서 이를 구체적으로 확인할 수 있다(Edelman, 2014). 구글과 제조사 간의 모바일 어플리케이션 유통 합의는 비공개로 이루어지며, 지금까지 구글과 HTC, 구글과 삼성전자 간에 체결된 합의서가 폭로된 바 있다. 이에 따르면 제조사가 안드로이드 스마트폰을 출시하기 위해서는, 구글이 만든 어플리케이션을 기본 탑재하고 특정 기능을 실행시킬 때 이러한 어플리케이션들이 자동적으로 실행되도록 설정해야할 뿐만 아니라, 이 어플리케이션을 스마트폰의 첫 화면에 배치해야 하고 스마트폰 기기의 판매 현황을 주기적으로

구글에 보고해야 한다. 구글이 제조사를 이용해 어플리케이션의 설치, 동작, 배치, 확산에 이르는 모든 과정을 통제할 수 있을 정도의 권력을 얻게 된 것은, 기술체계 내부에 지배 구조들을 면밀하게 설계하고 이 구조들을 연결하는 동시에 정합성, 안정성을 통해 기술적 정체성에 대한 장악력을 강화시켰기 때문에 가능하다.

(3) 제도와 개방적 기술 체계: 한국과 중국의 사례

1) 중국의 사례: 의존구조의 배재와 기술의 지역적 변용

구글의 권력이 비단 제조사, 사용자, 개발자에게만 적용되는 것은 아니다. 구글의 권력은 국가적 권력과 충돌하기도 한다. 특히 정치적인 이유로 정보통신망에 대한 강력한 검열을 실시하고 있는 국가의 경우, 자유로운 정보의 이동은 국가 권력에 대한 도전으로 받아들여진다. 중국의 경우가 대표적이라고 할 수 있다. 중국은 개인, 기업 등에 대한 통제와 치밀한 정보 검열 시스템으로 대표되는 폐쇄적인 정보통신 체계를 유지하고 있다.[104] 하지만 13억 중국의 인구의 시장적 가능성으로 인해 전 세계 정보통신기업들은 이러한 폐쇄성에도 불구하고 사업을 확장하려고 노력하고 있다.

구글은 안드로이드 출시 이전부터 중국 정부와 갈등을 빚어왔다. 2010년 3월 구글은 중국 정부의 검열에 저항하여 중국에서의 검색 사업을 철수하였다. 중국 본토에서 모든 설비가 철수함에 따라 중국에서 발생하는 구글 검색은 홍콩 등 인근 지역의 서버에서 처리하게 되었다.[105] 이에 따라 안드로이드 스마트폰 사용자도 해외에 있는 서버를 통해 구글플레이에 접속하게 되었다. 하지만 이러한 요인들로 인해 중국의 기기 제조사들은 구글

---

104) 중국의 검열, 감시 시스템은 '중국의 만리방화벽(防火長城, Great Firewall of China)'으로 불리며, 정식 명칭은 '황금방패 프로젝트(金盾工程)'로서 중국의 공안부에 의해 운영되고 있다.

105) 이에 대해 중국 정부도 이에 대한 보복으로 전 세계의 모든 구글 검색 사이트에 대한 접근을 원천 봉쇄하기도 하였다.

의 기술 탑재를 기피하게 되었으며, 사용자는 정보전송 거리의 증가로 인한 낮은 접속성으로 인해 기술사용에 어려움을 겪게 되었다(Leber, 2012.12.6.).

중국 내에서 안드로이드 운영체제가 탑재된 스마트폰은 전체 스마트폰의 72%에 이르고 있다(<그림 29> 참조). 하지만 구글의 구조가 안드로이드 체계에서 배제됨에 따라 중국에서의 안드로이드 체계는 독특한 형태로 변형되었다. 우선 오픈소스로서 안드로이드 운영체제의 개방성은 중국에서도 많은 스마트폰이 출시되는 요인이 되었다. 안드로이드 운영체제는 인터넷에 공개되어 있으며 중국에 있는 개인과 기업일지라도 AOSP에 접속하여 안드로이드 운영체제를 다운로드 받을 수 있다. 하지만 개방성을 통한 안드로이드 운영체제의 확산에도 불구하고, 구글코드, 구글플레이스토어와 같은 구글의 기술은 정치적인 이유로 기피되거나 정상적으로 동작하기 어렵게 되었고 이는 중국에서 구글의 기술적 지배구조가 확산되지 못한 이유이다.[106] 구글의 지배구조 형성이 실패함에 따라 중국의 제조사들은 구글로부터 구글플레이스토어의 사용 허가를 받기 위해 정합성을 충족시킬 필요가 없어졌으며, 개발자도 모바일 어플리케이션의 기능을 구현하기 위해 구글코드가 아닌 다른 수단을 동원하게 되었다.

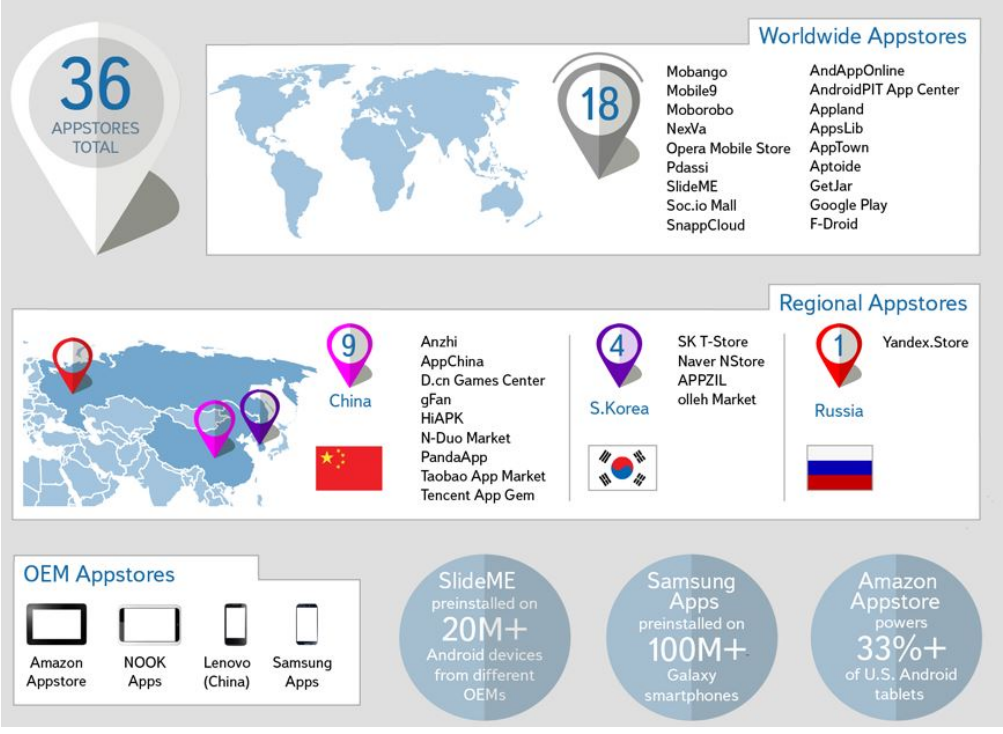
이에 따라 중국에서는 다양한 구글 비인증 기기가 시중에 유통되고 있다. 또한 안드로이드 어플리케이션 마켓 또한 파편화되었다. 중국 본토에만 9개의 어플리케이션 마켓이 운영되고 있으며, 이는 전 세계 36개의 어플리케이션 마켓의 4분의 1에 해당하는 수치이다(<그림 28> 참조[107]). 이와 더불어 중국 정부는 안드로이드 운영체제의 확산과 구글의 권력 확장을 경계하고 있다. 중국 정부의 공업신식화부는 “우리나라의 모바일OS 연구/개발르이 안드로이드에 너무 의존적이며 오픈소스 구조의 안드로이드로 인해 기술의 핵심과 지침을 구글에게 지배당하고 있다”고 밝히기도 하였다(최은

106) 구글플레이스토어는 중국 내에서 어플리케이션 시장의 4.2%만을 차지하고 있는 것으로 조사되었다(Umeng, 2013).

107) 2013년 전 세계에는 총 36개의 안드로이드 어플리케이션 시장이 개설되었으며, 시장을 만든 주체, 시장이 대상으로 하는 국가, 시장에 출시된 어플리케이션의 수 등에서 각기 차이를 보이고 있다. 36개의 어플리케이션 시장 중 18개의 시장은 전 세계 사용자를 대상으로 하며 중국, 한국, 러시아를 대상으로 하는 시장도 각각 9개, 4개, 1개 였다.

주, 2013.3.6.). 뿐만 아니라 중국 정부는 세일피시 운영체제(Sailfish OS)와 같은 새로운 운영체제 개발을 지원하고 있으며, 중국 정부가 자체적으로 차이나운영체제(COS)라는 공식 운영체제를 개발하여 발표하기도 하였다 (Le, 2014.1.30.).

이를 통해 개방적 정보기술 환경에서 기술적으로 유지되는 폐쇄적인 구조가 존재하지 않을 때 나타날 수 있는 현상을 확인할 수 있다. 정합성을 강제하고 이를 정치적으로 활용하여 사적 영역의 경계작업을 지속하는 세력이 없을 때, 개방적 기술은 개방, 확산을 지속하며, 결과적으로는 각기 다른 방식으로 변형되어 서로 개별적인 운영체제들만이 남게 된다. 이 경우 모바일 어플리케이션 또한 각각의 운영체제에 맞도록 개발되어야 하기 때문에, 동일한 안드로이드 코드의 변형일지라도 한 운영체제에서 동작하는 어플리케이션이 다른 운영체제에서는 동작하지 못할 가능성이 높다. 중



<그림 28> 안드로이드 어플리케이션 시장의 종류
(One Platform Foundation, 2013)

국의 사례에서 알 수 있듯이 폐쇄적인 구조화가 없는 상태에서 정보기술은 행위자와 각 상황의 특성을 반영한 분화를 겪게 되며, 동일한 코드라도 다양한 방식의 분화적 변용이 일어나게 된다. 그리고 중국의 경우, 이러한 변용을 통제하여 다시 기술적 통일성을 확보하려는 행위자는 구글이 아닌 중국정부이다.

2) 한국의 사례: 경제적, 정책적 의존과 구글의 권력 강화

한국은 전 세계에서 안드로이드 운영체제의 점유율이 가장 높은 국가 중의 하나이다. 한국에서 안드로이드 운영체제는 전체 스마트폰의 90.1%에 탑재되어 있으며, 이는 애플의 iOS 점유율 9.6%와 비교했을 때 매우 높은 수준이다[108](<그림 29>). 또한 한국에서 안드로이드 운영체제의 점유율은 조사 대상인 15개 국가 중에서도 가장 높다. 한국 스마트폰 운영체제 분야에서 구글의 절대적 우위는 인터넷 검색시장에서 구글의 낮은 점유율과는 대조적이다.

뿐만 아니라 한국의 정보통신기업들은 안드로이드 운영체제를 탑재한 스마트폰의 개발에 수익의 대부분을 의존하고 있다. 삼성전자와 LG전자, 팬택과 같은 대기업들은 모두 안드로이드 스마트폰을 주력 상품으로 내세우고 있다. 이 중 삼성전자는 모바일 운영체제를 자체적으로 개발하기도 하였으나, 개발자, 제조사의 확보에 실패한 뒤 안드로이드 스마트폰 개발에 집중하고 있으며, 세계적으로도 가장 많은 수의 안드로이드 스마트폰을 생산, 출하하고 있다. 삼성은 2012년 세계적으로 456만 여대의 스마트폰을 판매하여 전체 안드로이드 스마트폰 판매량의 29.7%을 차지하였다. LG전자 또한 2012년 세계 스마트폰 판매량의 3.8%를 차지하여 삼성전자, 애플에 이어 3위를 차지하고 있다(Gartner, 2013).

---

108) 스마트폰 운영체제 점유율 수치는 조사 기관에 따라서 차이를 보이나, 한국에서 안드로이드 운영체제가 절대적인 우위를 점하고 있는 것은 분명하다. 아틀라스 모바일 인덱스(ATLAS Mobile Index)라는 조사 기관에 의하면 2013년 12월 출하된 스마트폰의 84.1%가 안드로이드 운영체제를 탑재하고 있었으며, 이는 iOS의 7.0%와 비교하여 매우 높은 수치이다. 시장조사업체인 가트너(Garner)도 한국의 안드로이 운영체제 점유율을 93.4%로 조사하였으며, 이는 전 세계에서 가장 높은 수치이다(박상률, 2014.7.7.).

<그림 29> 국가별 모바일 운영체제 점유율 비교.
(2013년 2월 기준. 출처:
http://connect.icrossing.co.uk/2013-mobile-market-share-infographic_10062)

안드로이드 운영체제로의 '쏠림'은 한국 정부의 정책에서도 확인된다. 한국 정부는 최근 '창조경제', '정부3.0', '스마트 정부', '공공데이터 개방' 등의 정책적 기조를 유지하고 있으며, 이러한 정책의 주요 일환으로 공공기간과 민간의 모바일 어플리케이션의 개발을 장려하고 있다. 이러한 정책들을 통해 정부는 공공기관의 서비스를 온라인화 하거나 이들이 보유하고 있는 데이터를 체계화하여 일반에 공개하고, 개발자들에게 창업 인프라, 사업 자

금, 기술교육을 지원해 모바일 어플리케이션을 만들 수 있는 환경을 조성하고 있다. 한국 모바일 시장을 안드로이드 운영체제가 장악하고 있으므로, 이러한 정책은 결국 공공 서비스와 데이터를 제공하여 안드로이드 어플리케이션의 개발을 촉진하는 것이라고 할 수 있다.

하지만 안드로이드 운영체제에 대한 경제적, 정책적 의존은 결국 한국에서 구글의 권력을 증가시키는 원동력이 되고 있다. 이는 구글과 정부 간의 갈등 혹은 대립 양상을 살펴보는 것으로 확인할 수 있다. 최근 들어 한국 정부는 수 차례에 걸쳐 구글에 권고안 및 지침을 내린바 있다. 하지만 국내의 다른 기업들과는 달리, 구글은 이러한 지침을 이행하지 않거나 아예 무시하고 있다(<표 22>). 뿐만 아니라 구글이 독점규제법 위반 여부에 대한 조사에서 무혐의 처분을 받고 미래창조과학부의 소액결제서비스사업자 선정과정에서 구글에 대한 '봐주기'의혹이 불거지는 등 정부가 구글의 '눈치'를 보고 있는 것이 아니냐는 비판도 제기되고 있는 상황이다(이순혁, 2014.4.1.).

제도적 조치	관련 조직 및 법률	구글의 대응
스마트폰 선탑재 앱 가이드라인 (2014년)	미래창조과학부	미이행
인터넷 검색서비스 발전을 위한 권고안 (2013년)	미래창조과학부	미이행
개인정보 이용내역 통지 제도 (2012년)	방송통신위원회	미이행

<표 22> 제도적 규제의 결과

구글의 지배력 확장과 함께 개발자, 사업자와의 관계에서도 구글은 일방적인 권력을 행사하고 있다. 구글은 2012년 구글플레이 운영정책을 변경하여 구글플레이스토어를 통해 배포되는 모바일 어플리케이션 내부에서 사용되는 결제시스템에 자사의 결제 시스템인 구글월렛(Google Wallet)만을 사용하도록 강제했다. 구글의 결제시스템을 사용하려면 30%의 결제 수수료를 지불해야하며, 이는 다른 결제시스템의 수수료율의 3배에 이른다. 뿐만 아니라 2014년 구글은 구글플레이스토어 유료 어플리케이션 판매 수익금의 통신사, 구글 간 배분 비율을 변경하였다. 기존에는 유료 어플리케이션의 판매금의 30%에 해당하는 수수료를 구글과 통신사가 1대 9의 비율로 분배하였으나, 이 조치에 의해 수수료를 구글과 통신사가 절반씩 나누게

되어 구글의 몫이 5배 증가했다.

스마트폰 시장에 쉽고 빠르게 진출하여 애플사에 대응하려는 국내 기업들은 안드로이드 운영체제에 집중하게 되었고, 안드로이드 스마트폰에 편중된 기술 생산으로 인해 국내 스마트폰 시장은 안드로이드 운영체제에 잠식되었다. 동일한 기술의 확산은 기회와 동시에 위기를 가져다준다. 우선 이러한 '통일성'은 새로운 거대 산업의 창출이 가능한 기술적 조건이다. 대부분의 국민이 동일한 운영체제를 사용한다면 민간과 정부에서 기능과 서비스를 제공하기 위해 생산한 정보기술이 더욱 많은 사람들에게 확산되어 영향을 미칠 수 있기 때문이다. 하지만 한국은 개방성과 통일성이 주는 이점에만 집중하여 민간과 정부에서 구글의 사유화된 기술구조에 대한 비판적 검토가 이루어지지 않고 있는 모습을 보여주고 있다. 이로 인해 한국에서는 경제적, 정책적으로 구글에 대한 종속이 심화되고 있다. 이는 안드로이드 운영체제의 높은 점유율에도 불구하고 국가 수준의 견제와 통제를 통해 사유화된 구조가 제거되고, 기술이 국내 기업과 개발자에 의해 다양하게 변형, 파편화되고 있는 중국과는 대조되는 것이다.

VI. 결론

하나의 기술체계에는 개방 구조와 지배 구조가 공존할 수 있다. 개방 구조는 기술의 생산, 이용에 참여해 많은 행위자가 기술체계에 유입될 수 있도록 하는 반면, 특정 행위자는 지배 구조를 통해 생산, 사용되는 기술이 자신의 목적에 부합하도록 하며 이 과정에서 기술체계 내부의 다른 행위자들이 통제된다. 이러한 개방과 지배의 '강도' 따라 기술체계는 여러 유형을 가질 수 있다(<표 23> 참조). 만약 다양한 생산자와 사용자의 제약 없는 참여가 보장되지만 생산 과정, 생산물, 사용 행위를 제약하는 지배 구조가 부재하다면, 이러한 기술은 각 생산자와 소비자, 사용자의 맥락에 따라 분화될 것이다. 그리고 정보기술의 광범위한 분화는 기술 간 상호작용을 불가능하게 하여 파편화를 가져올 가능성이 크다. 또한 만약 특정 행위자가 생산 과정, 생산 결과, 기술의 확산 과정을 완전히 통제하나 다른 행위자는 기술의 생산, 변형과정에 참여하거나 기술을 무료로 쉽게 이용할 수 없다면, 이는 완제품의 상업적 판매와 같은 한정적 상호작용만 일어나는 폐쇄적 기술체계라고 할 수 있다. 반면 안드로이드 체계와 같은 경우는 많은 행위자의 기술사용, 생산 참여와 강력한 지배 구조가 공존한다. 이러한 기술체계에서 지배는, 안정성, 정합성 등의 수단을 통해 기술의 '정체성'을 정의할 자격을 가진 중심 행위자가 자신의 목적에 부합하는 인간, 비인간 행위자만을 포섭하고 나머지를 배제하는 정치적 경계작업을 수행함으로써 이루어진다.

우리가 이러한 정치적 기술체계에 더욱 관심을 가져야 하는 이유는, 지

		개방 구조	
		약함 (폐쇄적)	강함 (개방적)
지배 구조	약함 (분권적)	미체계화된 기술	**파편화된 기술체계**
	강함 (독재적)	**폐쇄적 기술체계**	**정치화된 기술체계**

<표 23> 개방 구조와 지배 구조의 강도에 따른 기술체계의 유형

배와 권력 확장이 개방적 구조의 이면에서 진행되기 때문이다. 개방과 폐쇄, 자율과 통제의 공존에서 중요한 점은, 개방적 구조가 표면적으로 드러나 있는 반면 지배를 가능하게 하는 표준과 정책의 생산, 정보 유통 방식은 외부에 잘 드러나지 않으며 설계 과정에서 이미 기술체 내부에 배태되어 있는 경우도 많다는 점이다. 따라서 기술의 표층뿐만 아니라 내재화, 블랙박스화(black-boxing)된 지배 구조를 확인하는 것이 이러한 공존의 '모순성'을 해결하는 방법이 될 것이다.

안드로이드 운영체제의 사례는 이러한 이면적 구조에 의해 제조사, 개발자가 '기술적 종속'의 위험에 처해 있음을 보여주고 있다. 기술적으로 종속된 행위자는 기술을 자율적으로 선택, 사용하지 못하며, 기술적 대안이 실종되고 중심 행위자에 의한 정치적 배제의 위험도 크기 때문에 특정 기술을 추종하게 된다. 특히 안드로이드 운영체제의 점유율이 90%를 넘어서고 있음에도 불구하고 산업체와 정부, 민간에 의한 대응이 이루어지지 않고 있는 우리나라의 경우는 구글에 대한 기술 종속의 위험성이 훨씬 크다고 할 수 있다. 이는 안드로이드 운영체제 기술에 대한 지속적인 감시와 분석을 통해 구체적인 안전장치들을 마련하고 있는 유럽이나, 국가적인 차원에서 대안적인 기술의 생산을 장려하여 구글의 권력 확장에 대항하고 있는 중국의 경우와는 대비되는 것이다. 기술의 생산과 사용을 단순히 개별 행위자의 선택 문제, 시장의 문제로 치부해서는 기술적 지배 구조는 이해될 수도 극복할 수도 없으며, 기술사회학적 분석에 기반한 사회적, 국가적 대응이 있어야만 권력 확장과 종속에 적절히 대비할 수 있을 것이다.

VII. 참고문헌

박상률. (2014.7.7.). “한국 안드로이드 점유율 93.4%‥전세계 최고”. 『한국경제』. http://www.wowtv.co.kr 에서 인출

이순혁. (2014.4.1.). “구글은 ‘글로벌 기업’이라서?…정부, 관대하거나 무력하거나”. 『한겨레』. http://www.hani.co.kr 에서 인출.

정보통신정책연구원. (2012). 『방송매체 이용행태 조사』. 정보통신정책연구원.

최은주, (2013.3.6.). "中 정부, 안드로이드 견제 움직임 :중국 내 구글 영향력 너무 강하다.". 『OSEN』. http://osen.mt.co.kr/ 에서 인출

한국인터넷진흥원. (2012). 『스마트폰이용실태조사』. 한국인터넷진흥원.

ABI research. (2012.10.16). Average Size of Mobile Games for iOS Increased by a Whopping 42% between March and September. Retrieved from http://www.abiresearch.com.

Akrich, M. (1992) “The De-scription of Technical Objects,” in W. Bijker & J. Law (eds), *Shaping Technology-Building Society : Studies in Sociotechnical Change*. MIT Press: 205-24.

Amadeo, R. (2013.10.21.). Google's Iron Grip on Android: Controlling Open Source by Any Means Necessary. *Ars Technica*. Retrieved from http://arstechnica.com.

AppBrain. (2013). Number of available Android applications. Retrieved from http://www.appbrain.com/stats/number-of-android-apps.

Austin, S. (2013.3.11.). The Surprising Numbers Behind Apps. *The Wall Street Journal*. Retrieved from http://blogs.wsj.com/digits.

Bijker, E. W. and John Law (eds) (1992). *Shaping Technology/Building Society: Studies in Sociotechnical Change*, MIT Press.

Bowker, G. and S. Star. (2000). *Sorting Things out*, MIT University Press.

Callon, M., (1986). “Some elements of a sociology of translation: domestication of the scallops and the fishermen of St Brieuc Bay“. Law, J. ed. (1986). *Power, Action and Belief: a new sociology of knowledge?*, London, Routledge. pp 196-223.

Cheng, R. (2012.9.13.). Alibaba: Google forced Acer to drop our new mobile OS. CNET. Retrieved from http://news.cnet.com.

ComScore, (2006.7.18.). Google's U.S. Search Market Share Continues to Climb in June; Yahoo! Also Posts Gains. Retrieved from http://ir.comscore.com.

Constantinou, A. (2011.2.11.). The Android Monopoly and how to harness it. *VisionMobile.* Retrieved from

Derene , G. (2011.12.13.). Android's Cheap, Low Quality Apps Make It Feel Like A Technological Ghetto. *Business Insider.* Retrieved from http://www.businessinsider.com.

Edelman, B. (2014). Secret Ties in Google's "Open" Android. Retrieved from http://www.benedelman.org/news/021314-1.html.

Gartner. (2009). *Market Share: Smartphones, Worldwide, 4Q08 and 2008*. Gartner. Retrieved from https://www.gartner.com.

Gartner. (2013). *Market Share Analysis: Mobile Phones, Worldwide, 2Q13*. Gartner. Retrieved from https://www.gartner.com.

GNU. (1991). GNU General Public License, version 2. Retrieved from http://www.gnu.org/licenses/gpl-2.0.html.

Google. (2012). *Google's Opening Slides,* Retrieved from http://www.groklaw.net/pdf3/OraGoogle-Trial-GoogleOpeningStills.pdf.

Google. (2013). *Supported locations for distribution to Google Play users*. Retrieved from https://support.google.com/googleplay/android-developer/table/3541286?hl=en.

Gordon, S. (2014.5.21.). Global brands: Google pips Apple in popularity. *Financial Times*. Retrieved from http://www.ft.com.

IETF. (1997). Key Words for Use in RFCs to Indicate Requirement Levels. Retrieved from http://www.ietf.org/rfc/rfc2119.txt.

Imagination Technologies. (2014.4.1.). *New MIPS-based Ingenic Newton platform targets wearables and other consumer IoT devices*. Retrieved from http://blog.imgtec.com

Kellerman, A. (2006). Personal Mobilities. Reoutledge.

Khalaf, S. (2013.4.3.). Flurry Five-Year Report: It's an App World. The Web Just Lives in It. *Flurry.* Retrieved from http://blog.flurry.com.

King, R. (2012.4.24.). "Trial: Android chief on why Java was picked for Android". *ZDNet.* Retrieved from http://www.zdnet.com.

Knorr-Cetina, K. (2002). Global Microstructures: The Virtual Societies of Financial Markets. *American Journal of Sociology*. 107:4, pp.905-50.

Latour, B. (1992). Where are the Missing Masses? The Sociology of a Few Mundane Artifacts, in Bijker, W. and J. Law (eds.). (1992). Shaping Technology/Building Society: Studies in Sociotechnical Change, MIT Press. pp. 225 - 258.

Latour, B. (2002). Morality and technology: the end of the means. Theory, Culture & Society, 19, 5-6, 247-260.

Latour, B. (2007). Reassembling the Social: An Introduction to Actor-Network Theory. Oxford University Press.

Lave, J. and E. Wenger. (1991). *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press.

Le, D. (2014.1.30.). China hopes to dispel 'copy others' reputation. *BBC.* Retrieved from http://www.bbc.com

Leber, J. (2012.12.6.). Android Takes Off in China, But Google Has Little to Show for It. *MIT Technology Review.* Retrieved from

http://www.technologyreview.com

Lessig, L. (2006). *Code 2.0*. Basic Books.

Nielsen. (2011). Nielsen Smartphone Analytics. *Nielsen*. Retrieved from http://www.nielsen.com.

Nielsen. (2014). Smartphones: So Many Apps, So Much Time. *Nielsen*. Retrieved from http://www.nielsen.com.

OHA. (2007.11.5.). Industry Leaders Announce Open Platform for Mobile Devices. Retrieved from http://www.openhandsetalliance.com/press_110507.html.

Olsthoorn, P. (2012). *The Price We Pay for Google*. Eburon Elements.

Opensignal. (2013). Android Fragmentation Visualized. Retrieved from http://opensignal.com/reports/fragmentation-2013/fragmentation-2013.pdf.

Oracle. (2012). Opening Statements. Retrieved from http://www.oracle.com.

Oracle. (2014). Learn About Java Technology. Retrieved from http://www.java.com/en/about/

Oudshoorn, N. & T. Pinch (2007). User-Technology Relationships: Some Recent Developments. In Hackett, E.J., Amsterdamska, O., Lynch M., & Wajcman, J. (eds). *The Handbook of Science and Technology Studies* (pp. 541-566). The MIT Press.

Patel, N. (2011.5.12.). How Google Controls Android: Digging Deep into the Skyhook Filing. *The Verge*. Retrieved from http://www.theverge.com

Paul, R. (2007.11.7.). Why Google chose the Apache Software License over GPLv2 for Android. *Ars Technica*. Retrieved from http://arstechnica.com.

Pinch, T. and W. Bijker. (1984) "The Social Construction of Facts and Artefacts: Or How the Sociology of Science and the Sociology of

Technology Might Benefit Each Other". *Social Studies of Science*, 14(3):399-441

Reisinger, D. (2012.10.17.). Worldwide smartphone user base hits 1 billion. *CNET*. Retrieved from http://news.cnet.com.

Schmidt, S. and Werle, R. (1998) *Coordinating Technology*. The MIT Press.

Shankland, (2009.12.21.). Google conquers 2008 search market in U.S., *CNET*. Retrieved from http://www.cnet.com.

Shalvey, K. (2014.4.23.). Facebook Earnings Nearly Triple; Does Facelift Loom? *Investor's Business Dailey*. Retrieved from http://www.investors.com.

Star, S. and K. Ruhleder. (1996). Steps Towards an Ecology of Infrastructure: Design and Access for Large Information Spaces. *Information Systems Research*. 7(1): 111-134.

Umeng, (2013). China Mobile Internet: Fragmentation and Shorter Life Cycle. Retrieved from http://www.slideshare.net/umengnews/umeng-2013-first-half-insight-report-of-china-mobile-market.

Vaidhyanathan, S. (2012). *The Googlization of Everything: (and Why We Should Worry)*, University of California Press.

VisionMobile. (2013). *Developer Economics Q3 2013*. Retrieved from http://www.visionmobile.com/product/developer-economics-q3-2013-state-of-the-developer-nation/.

Wenger, E. (1998). *Communities of Practice: Learning, Meaning, and Identity*, Cambridge University Press.

Woolgar, (1991). Configuring the User: the Case of Usability Trials. in J. Law (ed), *A Sociology of Monsters: Essays on Power, Technology and Domination*. Routledge.

VIII. 부록

	Argentina	Australia	Austria	Belgium	Brazil	Canada	Chile	China	Egypt	Finland	France	Germany	Hongkong	India	Indonesia	Italy	Japan	Malaysia	Mexico	Netherlands	Philippines	Poland	Russia	Singapore	South Africa	Korea	Spain	Sweden	Swerlan	Taiwan	Thailand	Turkey	United Kingdom	United States
Argentina	50.0	18.3	19.7	22.7	27.1	18.6	34.0	9.4	20.1	17.0	20.1	18.9	14.6	20.7	19.4	21.7	9.6	20.9	32.4	17.7	19.4	20.0	18.9	19.9	22.1	8.4	26.1	16.3	18.1	14.0	17.7	19.7	19.9	14.6
Australia	18.3	50.0	23.1	24.6	19.3	32.3	18.0	11.1	16.9	23.0	22.7	23.9	17.3	16.9	16.7	25.0	12.7	18.9	19.6	22.6	17.6	16.4	15.9	23.4	22.3	8.4	22.4	22.1	22.4	15.4	17.4	17.4	29.3	32.1
Austria	19.7	23.1	50.0	26.3	19.7	24.3	19.9	10.9	17.3	22.4	22.6	33.3	16.1	17.6	15.1	24.9	11.6	18.9	21.1	21.4	16.6	18.9	17.0	21.1	21.0	9.0	23.7	20.9	25.0	15.6	16.4	18.6	23.9	20.7
Belgium	22.7	24.6	26.3	50.0	23.3	26.3	23.1	11.4	19.1	25.3	26.4	25.1	16.0	18.7	15.9	25.9	11.9	19.9	24.3	25.4	18.1	20.0	19.1	22.4	24.1	7.1	26.6	22.4	25.3	15.3	17.9	21.3	24.3	20.6
Brazil	27.1	19.3	19.7	23.3	50.0	18.7	27.7	9.4	20.0	19.0	21.9	18.6	15.0	22.0	20.3	21.9	9.9	22.6	27.6	18.6	20.7	19.7	21.1	19.6	21.9	8.9	24.9	17.4	17.9	14.0	20.4	21.1	20.6	16.4
Canada	18.6	32.3	24.3	26.3	18.7	50.0	18.4	10.6	16.3	23.9	23.6	24.7	16.1	16.1	15.6	24.7	13.3	18.4	20.3	23.9	16.4	16.7	15.9	23.0	21.7	7.7	22.7	22.7	22.9	16.1	15.9	16.6	28.6	29.1
Chile	34.0	18.0	19.9	23.1	27.7	18.4	50.0	9.7	20.1	17.7	20.1	19.0	14.4	20.1	18.6	21.7	9.4	22.3	33.9	18.4	19.7	18.9	18.1	19.3	21.7	7.6	25.7	16.4	18.3	13.3	17.3	18.9	18.9	15.3
China	9.4	11.1	10.9	11.4	9.4	10.6	9.7	50.0	11.6	10.3	10.9	10.9	17.9	10.1	9.6	12.1	7.0	12.9	9.7	9.1	9.9	11.0	11.6	15.1	11.1	6.3	10.7	10.6	9.0	16.4	10.4	10.6	11.7	9.7
Egypt	20.1	16.9	17.3	19.1	20.0	16.3	20.1	11.6	50.0	15.7	16.0	16.0	15.1	22.1	17.9	16.9	10.0	20.1	18.7	15.9	17.3	16.3	15.3	20.6	20.0	6.7	17.3	15.9	14.4	14.1	15.1	18.7	20.0	14.6
Finland	17.0	23.0	22.4	25.3	19.0	23.9	17.7	10.3	15.7	50.0	20.0	21.9	14.3	16.4	14.0	21.7	10.7	16.4	19.3	20.9	13.7	18.0	17.9	19.7	20.7	7.1	21.4	22.3	20.4	13.6	14.9	17.0	22.3	20.9
France	20.1	22.7	22.6	26.4	21.9	23.6	20.1	10.9	16.0	20.0	50.0	22.7	15.6	16.1	15.1	24.7	12.4	18.1	21.1	19.7	16.1	19.3	18.3	19.3	20.6	8.4	24.0	17.9	21.1	14.6	15.6	18.4	22.4	20.3
Germany	18.9	23.9	33.3	25.1	18.6	24.7	19.0	10.9	16.0	21.9	22.7	50.0	15.9	17.0	14.4	26.4	11.3	17.9	20.3	21.4	15.7	17.6	17.0	20.9	21.6	8.0	24.1	19.4	24.9	16.6	15.0	17.6	24.4	21.0
Hongkong	14.6	17.3	16.1	16.0	15.0	16.1	14.4	17.9	15.1	14.3	15.6	15.9	50.0	15.3	15.1	18.0	10.9	20.4	16.0	14.6	15.0	14.1	13.3	23.7	16.4	8.6	16.4	14.9	14.1	25.6	18.0	15.7	17.1	15.0
India	20.7	16.9	17.6	18.7	22.0	16.1	20.1	10.1	22.1	16.4	16.1	17.0	15.3	50.0	21.1	18.0	8.4	22.7	21.3	13.4	22.1	19.1	17.7	21.7	24.4	8.0	18.1	14.7	13.6	14.4	16.9	19.9	22.3	13.9
Indonesia	19.4	16.7	15.1	15.9	20.3	15.6	18.6	9.6	17.9	14.0	15.1	14.4	15.1	21.1	50.0	17.1	9.9	28.7	20.0	12.7	26.1	16.9	14.9	21.9	19.3	8.7	16.6	12.3	13.4	16.1	21.4	16.7	17.9	15.9
Italy	21.7	25.0	24.9	25.9	21.9	24.7	21.7	12.1	16.9	21.7	24.7	26.4	18.0	18.0	17.1	50.0	12.9	21.0	24.3	23.0	18.6	19.3	19.0	22.4	23.3	9.6	27.0	19.6	22.9	17.1	17.9	20.1	25.1	21.4
Japan	9.6	12.7	11.6	11.9	9.9	13.3	9.4	7.0	10.0	10.7	12.4	11.3	10.9	8.4	9.9	12.9	50.0	10.6	10.3	10.6	8.4	9.7	9.7	12.3	10.9	7.0	12.3	10.4	10.3	12.0	11.4	9.6	11.9	13.0
Malaysia	20.9	18.9	18.9	19.9	22.6	18.4	22.3	12.9	20.1	16.4	18.1	17.9	20.4	22.7	28.7	21.0	10.6	50.0	22.9	15.4	27.7	17.4	17.0	30.3	22.4	10.0	19.1	15.7	16.9	19.9	23.7	21.4	22.1	16.6
Mexico	32.4	19.6	21.1	24.3	27.6	20.3	33.9	9.7	18.7	19.3	21.1	20.3	16.0	21.3	20.0	24.3	10.3	22.9	50.0	19.7	21.4	19.4	19.0	21.9	22.0	8.9	29.4	18.6	20.3	14.4	19.4	20.3	20.9	17.6
Netherlands	17.7	22.6	21.4	25.4	18.6	23.9	18.4	9.1	15.9	20.9	19.7	21.4	14.6	13.4	12.7	23.0	10.6	15.4	19.7	50.0	13.9	14.0	14.0	19.1	20.0	6.7	23.9	21.7	24.6	13.7	14.6	15.9	22.9	18.0
Philippines	19.4	17.6	16.6	18.1	20.7	16.4	19.7	9.9	17.3	13.7	16.1	15.7	15.0	22.1	26.1	18.6	8.4	27.7	21.4	13.9	50.0	15.3	15.3	23.4	22.3	7.7	18.7	13.4	14.3	14.4	19.1	17.6	19.7	15.1
Poland	20.0	16.4	18.9	20.0	19.7	16.7	18.9	11.0	16.3	18.0	19.3	17.6	14.1	19.1	16.9	19.3	9.7	17.4	19.4	14.0	15.3	50.0	21.3	18.1	19.9	7.1	19.1	14.9	14.6	13.4	14.9	17.1	17.7	14.3
Russia	18.9	15.9	17.0	19.1	21.1	15.9	18.1	11.6	15.3	17.9	18.3	17.0	13.3	17.7	14.9	19.0	9.7	17.0	19.0	14.0	15.3	21.3	50.0	16.9	19.7	7.1	18.6	14.3	14.4	14.1	14.6	19.4	17.0	14.4
Singapore	19.9	23.4	21.1	22.4	19.6	23.0	19.3	15.1	20.6	19.7	19.3	20.9	23.7	21.7	21.9	22.4	12.3	30.3	21.9	19.1	23.4	18.1	16.9	50.0	23.7	9.6	21.0	20.0	18.9	21.4	19.9	20.4	25.7	21.9
South Africa	22.1	22.3	21.0	24.1	21.9	21.7	21.7	11.1	20.0	20.7	20.6	21.6	16.4	24.4	19.3	23.3	10.9	22.4	22.0	20.0	22.3	19.9	19.7	23.7	50.0	7.4	21.6	18.3	19.3	16.1	17.9	21.0	25.7	18.9
Korea	8.4	8.4	9.0	7.1	8.9	7.7	7.6	6.3	6.7	7.1	8.4	8.0	8.6	8.0	8.7	9.6	7.0	10.0	8.9	6.7	7.7	7.1	7.1	9.6	7.4	50.0	8.6	6.7	6.9	8.6	10.3	7.1	7.3	8.1
Spain	26.1	22.4	23.7	26.6	24.9	22.7	25.7	10.7	17.3	21.4	24.0	24.1	16.4	18.1	16.6	27.0	12.3	19.1	29.4	23.9	18.7	19.1	18.6	21.0	21.6	8.6	50.0	21.1	22.4	15.9	18.9	19.3	22.7	19.6
Sweden	16.3	22.1	20.9	22.4	17.4	22.7	16.4	10.6	15.9	22.3	17.9	19.4	14.9	14.7	12.3	19.6	10.4	15.7	18.6	21.7	13.4	14.9	14.3	20.0	18.3	6.7	21.1	50.0	19.7	14.6	14.6	16.3	23.1	18.6
Swerlan	18.1	22.4	25.0	25.3	17.9	22.9	18.3	9.0	14.4	20.4	21.1	24.9	14.1	13.6	13.4	22.9	10.3	16.9	20.3	24.6	14.3	14.6	14.4	18.9	19.3	6.9	22.4	19.7	50.0	12.9	14.3	16.6	21.6	19.1
Taiwan	14.0	15.4	15.6	15.3	14.0	16.1	13.3	16.4	14.1	13.6	14.6	16.6	25.6	14.4	16.1	17.1	12.0	19.9	14.4	13.7	14.4	13.4	14.1	21.4	16.1	8.6	15.9	14.6	12.9	50.0	18.3	15.0	16.4	14.9
Thailand	17.7	17.4	16.4	17.9	20.4	15.9	17.3	10.4	15.1	14.9	15.6	15.0	18.0	16.9	21.4	17.9	11.4	23.7	19.4	14.6	19.1	14.9	14.6	19.9	17.9	10.3	18.9	14.6	14.3	18.3	50.0	18.4	17.3	14.3
Turkey	19.7	17.4	18.6	21.3	21.1	16.6	18.9	10.6	18.7	17.0	18.4	17.6	15.7	19.9	16.7	20.1	9.6	21.4	20.3	15.9	17.6	17.1	19.4	20.4	21.0	7.1	19.3	16.3	16.6	15.0	18.4	50.0	20.3	14.4
United Kingdom	19.9	29.3	23.9	24.3	20.6	28.6	18.9	11.7	20.0	22.3	22.4	24.4	17.1	22.3	17.9	25.1	11.9	22.1	20.9	22.9	19.7	17.7	17.0	25.7	25.7	7.3	22.7	23.1	21.6	16.4	17.3	20.3	50.0	25.6
United States	14.6	32.1	20.7	20.6	16.4	29.1	15.3	9.7	14.6	20.9	20.3	21.0	15.0	13.9	15.9	21.4	13.0	16.6	17.6	18.0	15.1	14.3	14.4	21.9	18.9	8.1	19.6	18.6	19.1	14.9	14.3	14.4	25.6	50.0

<표 24> 평균 유사성 행렬

	Argentina	Australia	Austria	Belgium	Brazil	Canada	Chile	China	Egypt	Finland	France	Germany	Hongkong	India	Indonesia	Italy	Japan	Korea	Malaysia	Mexico	Netherlands	Philippines	Poland	Russia	Singapore	South Africa	Spain	Sweden	Switzerland	Taiwan	Thailand	Turkey	United Kingdom	United States
Argentina	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
Australia	5	0	1	3	5	1	5	5	6	1	3	1	5	6	6	2	1	5	6	5	1	6	5	5	4	5	3	2	2	5	6	5	2	0
Austria	4	1	0	2	4	0	4	6	5	0	2	0	6	5	5	1	0	6	5	4	0	5	4	4	3	4	2	1	1	6	5	4	1	1
Belgium	2	3	2	0	2	2	2	6	3	2	2	2	6	3	3	1	2	6	3	2	2	3	2	2	3	2	0	3	1	6	3	2	3	3
Brazil	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
Canada	4	1	0	2	4	0	4	6	5	0	2	0	6	5	5	1	0	6	5	4	0	5	4	4	3	4	2	1	1	6	5	4	1	1
Chile	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
China	4	5	6	6	4	6	4	0	3	6	4	6	1	3	3	5	6	1	3	4	6	3	4	4	3	4	6	5	5	1	3	4	5	5
Egypt	1	6	5	3	1	5	1	3	0	5	3	5	4	0	0	4	5	4	0	1	5	0	1	1	2	1	3	4	4	4	0	1	4	6
Finland	4	1	0	2	4	0	4	6	5	0	2	0	6	5	5	1	0	6	5	4	0	5	4	4	3	4	2	1	1	6	5	4	1	1
France	2	3	2	2	2	2	2	4	3	2	0	2	5	3	3	1	2	5	3	2	2	3	2	2	1	2	2	1	1	5	3	2	1	3
Germany	4	1	0	2	4	0	4	6	5	0	2	0	6	5	5	1	0	6	5	4	0	5	4	4	3	4	2	1	1	6	5	4	1	1
Hongkong	5	5	6	6	5	6	5	1	4	6	5	6	0	4	4	5	6	0	4	5	6	4	5	5	4	5	6	6	5	0	4	5	6	5
India	1	6	5	3	1	5	1	3	0	5	3	5	4	0	0	4	5	4	0	1	5	0	1	1	2	1	3	4	4	4	0	1	4	6
Indonesia	1	6	5	3	1	5	1	3	0	5	3	5	4	0	0	4	5	4	0	1	5	0	1	1	2	1	3	4	4	4	0	1	4	6
Italy	3	2	1	1	3	1	3	5	4	1	1	1	5	4	4	0	1	5	4	3	1	4	3	3	2	3	1	2	0	5	4	3	2	2
Japan	4	1	0	2	4	0	4	6	5	0	2	0	6	5	5	1	0	6	5	4	0	5	4	4	3	4	2	1	1	6	5	4	1	1
Korea	5	5	6	6	5	6	5	1	4	6	5	6	0	4	4	5	6	0	4	5	6	4	5	5	4	5	6	6	5	0	4	5	6	5
Malaysia	1	6	5	3	1	5	1	3	0	5	3	5	4	0	0	4	5	4	0	1	5	0	1	1	2	1	3	4	4	4	0	1	4	6
Mexico	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
Netherlands	4	1	0	2	4	0	4	6	5	0	2	0	6	5	5	1	0	6	5	4	0	5	4	4	3	4	2	1	1	6	5	4	1	1
Philippines	1	6	5	3	1	5	1	3	0	5	3	5	4	0	0	4	5	4	0	1	5	0	1	1	2	1	3	4	4	4	0	1	4	6
Poland	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
Russia	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
Singapore	3	4	3	3	3	3	3	3	2	3	1	3	4	2	2	2	3	4	2	3	3	2	3	3	0	3	3	2	2	4	2	3	2	4
South Africa	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
Spain	2	3	2	0	2	2	2	6	3	2	2	2	6	3	3	1	2	6	3	2	2	3	2	2	3	2	0	3	1	6	3	2	3	3
Sweden	3	2	1	3	3	1	3	5	4	1	1	1	6	4	4	2	1	6	4	3	1	4	3	3	2	3	3	0	2	6	4	3	0	2
Switzerland	3	2	1	1	3	1	3	5	4	1	1	1	5	4	4	0	1	5	4	3	1	4	3	3	2	3	1	2	0	5	4	3	2	2
Taiwan	5	5	6	6	5	6	5	1	4	6	5	6	0	4	4	5	6	0	4	5	6	4	5	5	4	5	6	6	5	0	4	5	6	5
Thailand	1	6	5	3	1	5	1	3	0	5	3	5	4	0	0	4	5	4	0	1	5	0	1	1	2	1	3	4	4	4	0	1	4	6
Turkey	0	5	4	2	0	4	0	4	1	4	2	4	5	1	1	3	4	5	1	0	4	1	0	0	3	0	2	3	3	5	1	0	3	5
United Kingdom	3	2	1	3	3	1	3	5	4	1	1	1	6	4	4	2	1	6	4	3	1	4	3	3	2	3	3	0	2	6	4	3	0	2
United States	5	0	1	3	5	1	5	5	6	1	3	1	5	6	6	2	1	5	6	5	1	6	5	5	4	5	3	2	2	5	6	5	2	0

<표 25> 기술거리 행렬

국가	측정 시점 별 동일 그룹 소속 여부 (기준: USA)							거리
	2012.1	2012.4	2012.7	2012.10	2013.1	2013.4	2013.7	
USA	0	0	0	0	0	0	0	0
Australia	0	0	0	0	0	0	0	0
Austria	0	1	0	0	0	0	0	1
Canada	0	1	0	0	0	0	0	1
Finland	0	1	0	0	0	0	0	1
Germany	0	1	0	0	0	0	0	1
Japan	0	1	0	0	0	0	0	1
Netherland	0	1	0	0	0	0	0	1
Italia	0	1	0	0	0	0	1	2
Switzerland	0	1	0	0	0	0	1	2
Sweden	0	1	1	0	0	0	0	2
UK	0	1	1	0	0	0	0	2
Belgium	0	1	0	1	0	0	1	3
Spain	0	1	0	1	0	0	1	3
France	0	1	1	0	0	0	1	3
Singapore	0	1	1	0	1	0	1	4
Argentina	0	1	1	1	0	1	1	5
Brazil	0	1	1	1	0	1	1	5
Chile	0	1	1	1	0	1	1	5
Mexico	0	1	1	1	0	1	1	5
Poland	0	1	1	1	0	1	1	5
Russia	0	1	1	1	0	1	1	5
South Africa	0	1	1	1	0	1	1	5
Turkey	0	1	1	1	0	1	1	5
Hongkong	1	0	1	0	1	1	1	5
Korea	1	0	1	0	1	1	1	5
Taiwan	1	0	1	0	1	1	1	5
China	1	0	1	0	1	1	1	5
Egypt	0	1	1	1	1	1	1	6
India	0	1	1	1	1	1	1	6
Indonesia	0	1	1	1	1	1	1	6
Malaysia	0	1	1	1	1	1	1	6
Philippines	0	1	1	1	1	1	1	6
Thailand	0	1	1	1	1	1	1	6
Vietnam	0	1	1	1	1	1	1	6

<표 26> 기술사용 거리의 계산 사례 (기준국가: 미국)

어플리케이션 이름	구글 지도	구글 클라우드 메세징	구글 계정	구글 인앱 결제
mehr-tanken	1	0	0	0
Viber	0	1	1	1
FonectaCaller	0	1	0	1
ooVoo Video Call, Text & Voice	0	1	0	1
Agent:videocalls&messages	0	1	0	1
textPlusFreeText+Calls	0	1	0	1
Skype-freeIM&videocalls	0	1	0	0
Tinder	1	1	1	0
Badoo-MeetNewPeople	0	1	1	1
Ola	1	1	1	0
GOSMSPro	0	0	0	1
Triiing	0	1	0	0
WEB.DEMail	0	1	0	1
GMXMail	0	1	0	1
DaumMail-다음메일	0	1	0	0
Mail.Ru	0	1	0	0
GG	0	1	1	0
KakaoTalk:FreeCalls&Text	0	1	0	1
LINE:FreeCalls&Messages	0	1	0	1
mypeopleMessenger	0	1	0	0
MessengerTelcel	0	1	0	0
Mxit	0	1	0	0
ZelloWalkieTalkie	0	1	0	0
Snapchat	0	1	0	0
QQInternational-Chat&Call	0	0	1	0
WeChat	0	0	0	1
TangoMessenger	0	0	0	1
HotorNot	0	1	0	1
Vine	0	1	0	0
LINEBAND-family,friends	1	1	1	1
WhatsAppMessenger	1	1	1	1
Tuenti	0	1	1	0
KakaoGroup	0	1	0	0
Zalo - Nhắn gửi yêu thương	0	1	0	0
Tumblr	0	1	0	0
KakaoStory	0	1	0	0
네이버카페 - Naver Cafe	0	1	0	0
VK	1	1	1	1
Foursquare	1	1	1	0
Mikz	1	1	1	0

Twitter	1	1	1	0
Facebook	1	1	1	0
FacebookMessenger	0	1	1	0
ДругВокруг-друзьярядом	0	1	0	1
Одноклассники	0	1	0	0
LinkedIn	0	1	0	0
LOVOO-LiveDating&Friends	0	1	1	1
LINEマンガ	0	1	0	1
KikMessenger	0	0	0	1
FiresNearMeNSW	1	0	0	0
生活行VoiceGO!(中文搜尋平台)	1	1	1	1
SwissPhoneBook	0	1	0	0
Pinterest	1	1	1	0
Circle-TheLocalNetwork	0	1	0	0
HotPepperGourmet	1	1	1	0
食在方便-台灣在地美食＆餐廳APP	1	0	1	1
PagesJaunes - recherche locale	0	1	0	0
Argos	1	1	1	0
VakantieVeilingen.nl	1	1	0	0
OpenRiceHongKong	1	1	0	0
AppievanAlbertHeijn	1	0	0	0
Thuisbezorgd.nl	0	1	0	0
Wrapp	0	1	0	0
淘宝	0	1	0	0
Hemnet	1	0	1	0
NFLMobile	1	1	0	0
AccuWeather	1	1	1	0
Buienalarm	1	1	0	0
Weatherzone	1	0	1	0
ForecaWeather	1	0	1	0
TheWeatherChannel	1	0	0	0
MyObservatory(我的天文台)	1	0	0	0
GOWeatherForecast&Widgets	0	1	1	1
La chaine météo	0	1	1	0
Buienradar.nlPhone	0	1	0	1
Météo-France	0	1	0	0
ilMeteoWeather	0	1	0	0
MeteoSwiss	0	1	0	0
Weeronline	0	1	0	0
RunKeeper-GPSTrackRunWalk	1	1	1	1
DBNavigator	1	1	1	0
SNCB/NMBS	1	1	1	0
Voyages-SNCF	1	1	1	0
RATP:SubwayParis	1	1	1	0
MyTransportSingapore	1	1	1	0

SBBMobile	1	1	1	0
NSReisplannerXtra	1	1	1	0
SeoulBus(서울버스)	1	1	0	1
じゃらん	1	1	0	0
SBSTransitiris	1	1	0	0
KMB&LW	1	1	0	0
MTRMobile	1	1	0	0
2GIS:maps&businesslistings	1	1	0	0
jakdojade.pl	1	0	1	1
Maps	1	0	1	1
BusTrackerTaipei	1	0	1	0
ÖBB Scotty	1	0	1	0
qando	1	0	1	0
雙鐵時刻表	1	0	1	0
DeLijn	1	0	1	0
Offi-PublicTransportBuddy	1	0	0	0
ÖBB CepTrafik	1	0	0	0
STHLMTraveling(SL)	1	0	0	0
지하철종결자:SmarterSubway	1	0	0	0
Sygic:GPSNavigation&Maps	0	1	0	2
NAVITIME	0	1	0	1
NationalRailEnquiries	0	1	0	1
EniroSE	0	1	0	0
9292	0	1	0	0
STIB	0	1	0	0
WazeSocialGPSMaps&Traffic	0	1	0	0
mobileMPK	0	0	1	0
乗換案内	0	0	0	1
Meta, dé Magister leerlingapp!	0	0	0	1
CalorieCounter-MyFitnessPal	0	1	0	0
OLXFreeClassifieds	1	1	0	0
dealfish.co.th	0	1	0	0
ОбъявленияAVITO.ru	0	0	0	1
MyVodafone	1	1	1	0
AreaClienti3	1	1	1	0
MySingTel	1	1	1	0
119	1	0	1	0
MyWind(AppufficialeWind)	1	0	0	0
MiOrange	1	0	0	0
MySwisscom	0	1	0	0
Hello!BelgacomProximus	0	1	0	0
VKontakteKateMobile	0	1	0	0
NH스마트뱅킹	0	1	0	0
SFBio	1	1	0	0
遠通電收ETC	0	1	0	0

Outsurance	1	1	0	0
Veikkaus	0	1	0	0
CommBank	1	1	1	0
Sparkasse	1	1	0	0
FNBBankingApp	1	1	0	0
StandardBankMobileBanking	1	1	0	0
Telstra24x7	1	0	1	0
WestpacMobileBanking	1	0	1	0
Swedbank	1	0	1	0
BarclaysMobileBanking	1	0	1	0
NAB	1	0	0	0
ErsteBank/Sparkassenetbanking	1	0	0	0
TD(Canada)	1	0	0	0
OP-mobiili	1	0	0	0
MobiilipankkiFI	1	0	0	0
MyING.be	1	0	0	0
Bradesco	1	0	0	0
BankofAmerica	1	0	0	0
laCaixa	1	0	0	0
BanamexMobile	1	0	0	0
RBCMobile	0	1	1	0
OK캐쉬백	0	1	1	0
BancodoBrasil	0	1	0	0
KB국민금융스타뱅킹	0	1	0	0
Swishpayments	0	1	0	0
ChaseMobile	0	1	0	0
NewBelfiusDirectMobile	0	0	1	0
GoldenScreenCinemas	1	0	0	0
CineplexMobile	0	1	0	0
Ster-Kinekor	0	1	0	0
haivl.com2.0	0	0	1	0
BibleOffline	0	0	1	1
VDMOfficiel	0	1	0	0
Yahoo!JAPAN	0	1	0	0
네이버-Naver	0	1	0	0
WeHeartIt	0	1	0	0
Shazam	1	1	1	0
GooglePlayMusic	1	0	1	0
Bloom.fm-Themusicapp	0	1	0	1
Pandora®internetradio	0	1	0	1
Deezer	0	1	0	1
Spotify	0	1	0	1
Hitradio Ö3	Ö3 für Android	0	1	0	0
카카오뮤직	0	1	0	0
PalcoMP3	0	1	0	0

ZingMp3	0	1	0	0
NhacCuaTui	0	1	0	0
KKBOX	0	1	0	0
Ask.fm	0	1	0	0
SoundCloud-Music&Audio	0	1	0	0
RuntasticSixPackAbsWorkout	0	0	0	2
クックパッド	0	1	1	1
Marmiton	0	1	0	0
IMDbMovies&TV	0	1	0	0
Duolingo:LearnLanguagesFree	0	1	0	0
おかあさんといっしょ？　みいつけた！こども向け　リズムあそび	0	1	0	1
鬼から電話	0	0	0	1
TurmadaGalinhaPintadinha	0	0	0	1
NHLGameCenter™	0	1	0	1
LEquipe.fr	0	1	0	1
CricbuzzCricketScores&News	0	1	0	1
LigTV	0	1	0	1
Mackolik Canlı Sonuçlar	0	1	0	1
theScore:Sports&Scores	0	1	0	0
Résultats Foot en Direct	0	1	0	0
MARCA.com	0	1	0	0
Shpock:mobileyardsale	1	1	1	1
segundamano.es	1	1	1	0
Wallapop	1	1	1	0
eBayKleinanzeigenforGermany	1	1	1	0
Groupon-DailyDeals,Coupons	1	1	1	0
eBay	1	1	1	0
ricardo.ch	1	1	0	0
Gumtree	1	0	1	0
ImmobilienScout24	1	0	1	0
Nettiauto	1	0	0	0
AutoScout24-usedcarfinder	1	0	0	0
mobile.de - mobile Auto B.rse	0	1	0	0
AmazonモバイルAndroidアプリ	0	1	0	0
MercadoLibre	0	1	0	0
MobileRecharge,DTHandBills	0	1	0	0
Flipkart	0	1	0	0
AmazonMobile	0	1	0	0
쿠팡	0	1	0	0
半價美食、住宿券·GOMAJI團購網	0	1	0	0
위메프	0	1	0	0
티몬	0	1	0	0
옥션	0	1	0	0

Gmarket	0	1	0	0
GSSHOP	0	1	0	0
Qoo10SG	0	1	0	0
Amazon Mobil für Android	0	1	0	0
Amazon Mobile for Android	0	1	0	0
autoTRADER.ca-AutoTrader	0	0	1	0
Woolworths	0	0	1	0
ヤマダ電機　ケイタイde安心	0	0	0	1
EnglishChineseDictionary	0	0	0	1
Dictionary.com	0	1	0	1
Dictionary-Merriam-Webster	0	0	1	0
EnglishTagalogDictionaryFr	0	0	1	0
Tu Dien Viet Anh-Từ Điển TFLAT	0	0	1	0
OxfordDictionaryofEnglishT	0	0	0	1
realestate.com.au	1	0	0	0
LiveTotoSingapore	0	1	0	1
Live4DSingapore	0	1	0	0
Live4DResults!(MY&SG)	0	1	0	0
tv.nu	0	1	0	0
ProgramTV	0	1	1	0
YY－－你在播，我在看	0	1	0	0
HongKongToolbar	0	1	0	0
iHeartRadio-InternetRadio	0	1	0	0
MeRadio	0	1	0	0
TuneInRadio	0	1	0	0
VTM	1	0	1	0
YouTube	0	1	1	0
Programme TV par Télé Loisirs	0	1	0	1
MYTF1	0	1	0	1
Télé 7 programme TV	0	1	0	1
nexGTv-MobileTVLiveTV	0	1	0	1
Кинотеатрivi.ruфильмыонлайн	0	1	0	1
SPBTV	0	1	0	1
RTLXL	0	1	0	0
電視連續劇	0	1	0	0
電視綜藝	0	1	0	0
TVOveral/TVPartout	0	1	0	0
AlloCine	0	1	0	0
6play	0	1	0	0
SFRTV	0	1	0	0
RaiTV	0	1	0	0
HDFilmIzle	0	1	0	0

아프리카TV-AfreecaTV(Korean)	0	1	0	0
myTV	0	1	0	0
Netflix	0	1	0	0
TVSPIELFILM-TVProgramm	0	0	0	1
ZattooLiveTV	0	0	0	1
千尋影視	0	0	0	1
FreeBooks&Stories-Wattpad	0	1	0	0
GooglePlayBooks	0	0	1	0
20minutes(CH)	1	1	1	0
Aftonbladet	1	1	1	0
20Minuten(CH)	1	1	0	0
Ilta-Sanomat	0	1	0	0
NU.nl	0	1	0	0
NOS	0	1	0	0
BBCNews	0	1	0	0
台灣蘋果日報	0	1	0	0
HLN.beMobile	0	1	0	0
NewsHunt:IndiaNews|Jobs	0	1	0	0
다음	0	1	0	0
Apple Daily 蘋果動新聞	0	1	0	0
Báo Mới - Doc Bao Sieu Dep	0	1	0	0
HungryJacks®Shake&WinApp	1	1	1	0
Domino'sPizza	1	0	0	0
マクドナルド公式アプリ	1	0	0	0
MPost	0	1	0	0
Market+Mobile	0	1	0	0
蘋果投資王	0	1	0	0
XECurrency	0	0	0	1
NavitelNavigator	0	0	0	1
iChangi	0	1	0	0
도와줘!타로forKakao	0	1	0	1
GALAXYGift	0	1	0	0
4shared	0	1	0	0
MobileNumberTrackerIndia	1	0	0	0
StudioKUMACallFilter	0	1	0	1
Truecaller-CallerID&Block	0	1	0	1
WhosCall-CallerID&Block	0	1	0	0
AutomaticCallRecorder	0	0	1	0
KimArıyor?	0	0	0	1
CIA-CallerIdentificationApp	0	0	0	1
Bredbandskollen	1	0	0	0
Stocard-LoyaltyCards	0	1	1	0
MobileSecurity&Antivirus	0	1	1	0
Dropbox	0	1	0	1
BkavSecurity-AntivirusFree	0	1	0	0

N드라이브	0	1	0	0
AdobeAIR	0	1	0	0
Super-BrightLEDFlashlight	0	1	0	0
GoogleDrive	0	0	1	0
ZDbox(RootTaskKiller)	0	0	0	1
RemoteControlCollection	0	0	0	1
App2SD&AppManager	0	0	0	1
Dr.Webv.8Anti-virus	0	0	0	1
CoolReader	0	0	0	1
AppLock	0	0	0	1
GOLauncherEX	0	0	0	1
CityMaps2GoOfflineMaps	1	0	0	1
EzanVakti/NamazSaatiPro	0	1	1	0
صلاتك Salatuk (Prayer time)	0	1	0	0
Al-MoazinLite(PrayerTimes)	0	0	0	1
KaaveFalı	0	1	0	1
QRコードリーダー EQS	0	1	0	0
QRDroidTM	0	1	0	0
Chaozhuyin	0	0	0	1
GOKeyboard	0	0	0	1
JorteCalendar	0	1	0	1
JSBackup	0	1	0	0
PicsArt-PhotoStudio	1	1	1	1
Cymera-Camera&PhotoEditor	1	1	1	0
LINEcamera	1	1	0	1
PhotoEditorbyAviary	1	0	0	1
Crop&collagepapelook	0	1	0	1
DECOPIC,KawaiiPhotoEditingApp	0	1	0	1
PhoneThemeShop	0	1	0	1
PicCollage	0	1	0	1
Perfect365:BestFaceMakeup	0	1	0	1
earthphoto&diary	0	1	0	0
Petapic	0	1	0	0
InstaMag-MagazineCollage	0	1	0	0
HDwallpapers,themeworld	0	1	0	0
PicMix	0	1	0	0
Ticker	0	1	0	0
Bitstrips	0	1	0	0
FotoRus	0	1	0	0
FaceChanger	0	0	0	1
PhotoEditor	0	0	0	1
PhotoWonder	0	0	0	1
Repix	0	0	0	1
TheWalkingDeadDeadYourself	0	1	0	0
TalkingTomCat2Free	0	1	0	1

DJStudio5	0	0	0	1
TalkingAngela	0	1	0	1
TalkingTom&BenNewsFree	0	1	0	1
TalkingGinger	0	1	0	1
TalkingBentheDogFree	0	1	0	1
TalkingTomCatFree	0	1	0	1
TomLovesAngela	0	1	0	0
Speedtest.net	1	0	1	1
MobileSecurity&AntiVirus	0	1	1	1
BES10Client	0	1	0	0
DUBatterySaver&Widgets	0	0	1	1
NQSecurityMulti-language	0	0	0	1
NQMobileSecurity&Antivirus	0	0	0	1
GOBatterySaver&PowerWidget	0	0	0	1
ChromeBrowser-Google	0	1	1	0
2G|3G|4GFastInternetBrowser	0	1	0	0
barcoo-Barcode&QRScanner	0	1	0	0
CamScanner-PhonePDFCreator	0	1	0	1
FreeTelec Téléc Freebox	0	0	0	1
Socialcam	0	1	0	0
Atresplayer	0	0	0	1
MyCalendar	0	1	0	1
LichVanSu(LịchÂmDương)	0	1	0	0
PayPal	1	1	1	0
MobileISPService	0	1	0	0
EndomondoSportsTracker	1	1	1	1
SportsTracker	1	0	0	0
TwitCastingViewer-(Free)	0	1	0	1
FREECallswithmagicJack	0	0	0	0
FreePP:FreeCall&FreeMMS	0	0	0	0
DiDi - FreeCalls&Texts	0	0	0	0
QQ空間	0	0	0	0
YahooMail	0	0	0	0
Outlook.com	0	0	0	0
QQHDmini2013	0	0	0	0
ChikkaTextMessenger	0	0	0	0
QQ	0	0	0	0
Yahoo!Messenger	0	0	0	0
微博	0	0	0	0
Instagram	0	0	0	0
Blocket	0	0	0	0
發票小幫手(統一發票對獎)	0	0	0	0
search.ch	0	0	0	0
TheWeatherNetwork	0	0	0	0
Yahoo!天氣．災害	0	0	0	0

AndroidWeather&ClockWidget	0	0	0	0
FMIWeather	0	0	0	0
SMHI Väder	0	0	0	0
BBCWeather	0	0	0	0
墨迹天气	0	0	0	0
Weatherbyeltiempo.es	0	0	0	0
wetter.com	0	0	0	0
CitybusNWFB	0	0	0	0
ADACSpritpreise	0	0	0	0
Yahoo! 乗換案内	0	0	0	0
ComfortDelGroTaxiBooking	0	0	0	0
네이버지도/교통	0	0	0	0
Hitta.se	0	0	0	0
百度地圖	0	0	0	0
SGBuses	0	0	0	0
Yanosik	0	0	0	0
Yandex.Maps	0	0	0	0
Yandex.Metro	0	0	0	0
Yandex.Navigator	0	0	0	0
GoogleEarth	0	0	0	0
willhaben.at	0	0	0	0
Leboncoin	0	0	0	0
dubizzleEgypt-دوبيزلمصر	0	0	0	0
Tokobagus.com(OfficialApp)	0	0	0	0
Sulit.com.phBuyandSell	0	0	0	0
GumtreeSouthAfrica	0	0	0	0
sahibinden.com	0	0	0	0
easyJet	0	0	0	0
MeinA1	0	0	0	0
3Kundenzone	0	0	0	0
ALDITALK	0	0	0	0
Comviq	0	0	0	0
MyO2	0	0	0	0
Play24	0	0	0	0
Minafordon	0	0	0	0
배달의민족	0	0	0	0
TheXFactorUK	0	0	0	0
ArabsGotTalent	0	0	0	0
ANZgoMoneyTM	0	0	0	0
Nordea Mobile Bank - Finland	0	0	0	0
RaboBankieren	0	0	0	0
INGBankieren	0	0	0	0
Easybanking	0	0	0	0
Banque	0	0	0	0
Accès CompteLa Banque Postale	0	0	0	0

Itaú	0	0	0	0
StateBankFreedom	0	0	0	0
ABNAMROMobielBankieren	0	0	0	0
Handelsbanken	0	0	0	0
MinaSidor	0	0	0	0
Nordea Mobile Bank - Sweden	0	0	0	0
支付宝錢包	0	0	0	0
OctoCheck	0	0	0	0
IBKONE뱅킹개인-스마트뱅킹	0	0	0	0
AXSPayment	0	0	0	0
DBSmBanking	0	0	0	0
Maybank2uMalaysia	0	0	0	0
СбербанкОнЛ@й н	0	0	0	0
Tradera	0	0	0	0
TVCableChile	0	0	0	0
Demotywatory	0	0	0	0
Bible	0	1	0	0
AlQuranBahasaIndonesia	0	0	0	0
İnternetsiz Kuran-ı Kerim	0	0	0	0
Kapsel	0	0	0	0
네이버웹툰-NaverWebtoon	0	0	0	0
Songza	0	0	0	0
千千靜听2012（音樂播放器）	0	0	0	0
Cubomusica	0	0	0	0
Baixar Músicas - SimplesMP3	0	0	0	0
네이버뮤직	0	0	0	0
Rdio	0	0	0	0
코레일톡	0	0	0	0
멜론	0	0	0	0
Musicmp3zaycev.net	0	0	0	0
iKara-SingKaraoke	0	0	0	0
KaraokeVietnam	0	0	0	0
酷我音樂播放器	0	0	0	0
TTPod	0	0	0	0
KugouMusic	0	0	0	0
SimpleMp3MusicDownload	0	0	0	0
Simplemp3Downloader	0	0	0	0
Chefkoch.deRezepte	0	0	0	0
PeliculasWifiGratis:Movies	0	0	0	0
trucospou-monedasyniveles	0	0	0	0
БилетыПДД2013РФ	0	0	0	0
Colo-Colo Móvil	0	0	0	0
SuperSport	0	0	0	0
Carsales.com.au	0	0	0	0
AutoScout24Schweiz	0	0	0	0

GumtreeAustralia	0	0	0	0
Marktplaats	0	0	0	0
CraigslistMobile	0	0	0	0
Yapo.cl	0	0	0	0
DanMurphys	0	0	0	0
KijijiFreeLocalClassifieds	0	0	0	0
Hofer	0	0	0	0
樂天市場	0	0	0	0
ZalandoMobile	0	0	0	0
Allegro	0	0	0	0
McDkuponydoMcDonalds	0	0	0	0
PicknPay	0	0	0	0
Webxicon.org	0	0	0	0
有道詞典	0	0	0	0
英漢字典ECDictionary	0	0	0	0
Polish-Englishofflinedict.	0	0	0	0
Wielojęzyczny słownik polski	0	0	0	0
ThaiDict	0	0	0	0
İngilizce Türkçe Sözlük	0	0	0	0
LEOdictionary	0	0	0	0
Sky+	0	0	0	0
ilizyonNumaraları	0	0	0	0
radiko.jpforAndroid	0	0	0	0
SverigesRadioPlay	0	0	0	0
OpenFM	0	0	0	0
SharkMovie	0	0	0	0
YLEAreena	0	0	0	0
Ruutu	0	0	0	0
ElisaViihde	0	0	0	0
無料動畫 GyaO!	0	0	0	0
ClanenRTVE.es	0	0	0	0
Telefe	0	0	0	0
TeletextforSwitzerland	0	0	0	0
Dumpert	0	0	0	0
NPO	0	0	0	0
ZiggoTV	0	0	0	0
TVGiDS.tv2.0	0	0	0	0
SVTPlay	0	0	0	0
TV4Play	0	0	0	0
Kanal5Play	0	0	0	0
SwisscomTV	0	0	0	0
TeleboyTV	0	0	0	0
ITVPlayer	0	0	0	0
TVCatchup	0	0	0	0
4oD	0	0	0	0

SkyGo	0	0	0	0
PPTV网絡電視（手机版）	0	0	0	0
TVBfun	0	0	0	0
YeloTV	0	0	0	0
IndonesiaTVLive	0	0	0	0
IndonesiaLiveTV	0	0	0	0
ipla	0	0	0	0
ВКонтактеМузыкаиВидео	0	0	0	0
TVThailand	0	0	0	0
TVOnline	0	0	0	0
ThaiTVOnlineดูทีวีออนไลน	0	0	0	0
XemtiviHDVietNam(TVphim)	0	0	0	0
Youku	0	0	0	0
PPS影音(HD版)	0	0	0	0
PPS(forMobile)	0	0	0	0
Kindle	0	0	0	0
Iltalehti.fi	0	0	0	0
NOSTeletekst	0	0	0	0
SkySportsLiveFootballSC	0	0	0	0
SkySportsNews	0	0	0	0
NeteaseNews	0	0	0	0
detikcom	0	0	0	0
News24	0	0	0	0
DowcipyXXL	0	0	0	0
캐시슬라이드-첫화면에혜택을담다	0	0	0	0
QuienLlama-WhoisCalling	0	0	0	0
CallTimer	0	0	0	0
360手机衛士	0	0	0	0
CallerNameTalker	0	0	0	0
AnTuTuBenchmark	0	0	0	0
OFCABroadbandPerformanceTst	0	0	0	0
dodolPhone(data,call,SMSu	0	0	0	0
BSReaderS	0	0	0	0
充電が超長持ち高性能バッテリー	0	0	0	0
Fastweb.Alice.WiFi.Recovery	0	0	0	0
OldPhoneDialer	0	0	0	0
ASTROFileManagerwithClouds	0	0	0	0
BankID säkerhetsapp	0	0	0	0
BrightestFlashlightFree®	0	0	0	0
EmojiKeyboard	0	0	0	0
WeSync	0	0	0	0
fqrouter2	0	0	0	0
Link2SD	0	0	0	0
ScreenCaptureShortcutFree	0	0	0	0
WiFiPassRecovery&Backup	0	0	0	0

GOSMSProEmojiPlugin	0	0	0	0
EasyBatterySaver	0	0	0	0
FileManager(Explorer)	0	0	0	0
BatteryHD	0	0	0	0
AlwaysOnSamsung	0	0	0	0
InstantButtons	0	0	0	0
MP3Cutter&RingtoneMaker!!	0	0	0	0
GOLocker	0	0	0	0
BatteryDoctor(BatterySaver)	0	0	0	0
Quickoffice	0	0	0	0
FlashlightHDLED	0	0	0	0
ESFileExplorer File Manager	0	0	0	0
ZEDGETM	0	0	0	0
TinyFlashlight+LED	0	0	0	0
AdobeReader	0	0	0	0
SingaporeMap	0	0	0	0
WifiPassword	0	0	0	0
Simeji (JapaneseKeyboard)	0	0	0	0
搜狗手机輸入法	0	0	0	0
我查查-條碼比价、快遞查詢、二維碼掃描	0	0	0	0
GooglePinyinInput	0	0	0	0
GoogleCantoneseInput	0	0	0	0
GoogleZhuyinInput	0	0	0	0
KeyboardManMan	0	0	0	0
GoTiengViet3Vietnameseinput	0	0	0	0
GoTiengViet2	0	0	0	0
BarcodeScanner	0	0	0	0
PeriodCalendar/Tracker	0	0	0	0
[HighQuality]silentcamera	0	0	0	0
SilentCamera	0	0	0	0
InstaEyesPic-AnimalEyes	0	0	0	0
LovelyPhotoFrame	0	0	0	0
PhotoFunia	0	0	0	0
FaceSwitch	0	0	0	0
RetricaViewer	0	0	0	0
AgingBooth	0	0	0	0
QuickPic	0	0	0	0
Tablica	0	0	0	0
PicFrames	0	0	0	0
PhotoArt-ColorEffects	0	0	0	0
PhotoArtCollage	0	0	0	0
PicStitchCollageMaker	0	0	0	0
FxCollage-PhotoCollage	0	0	0	0

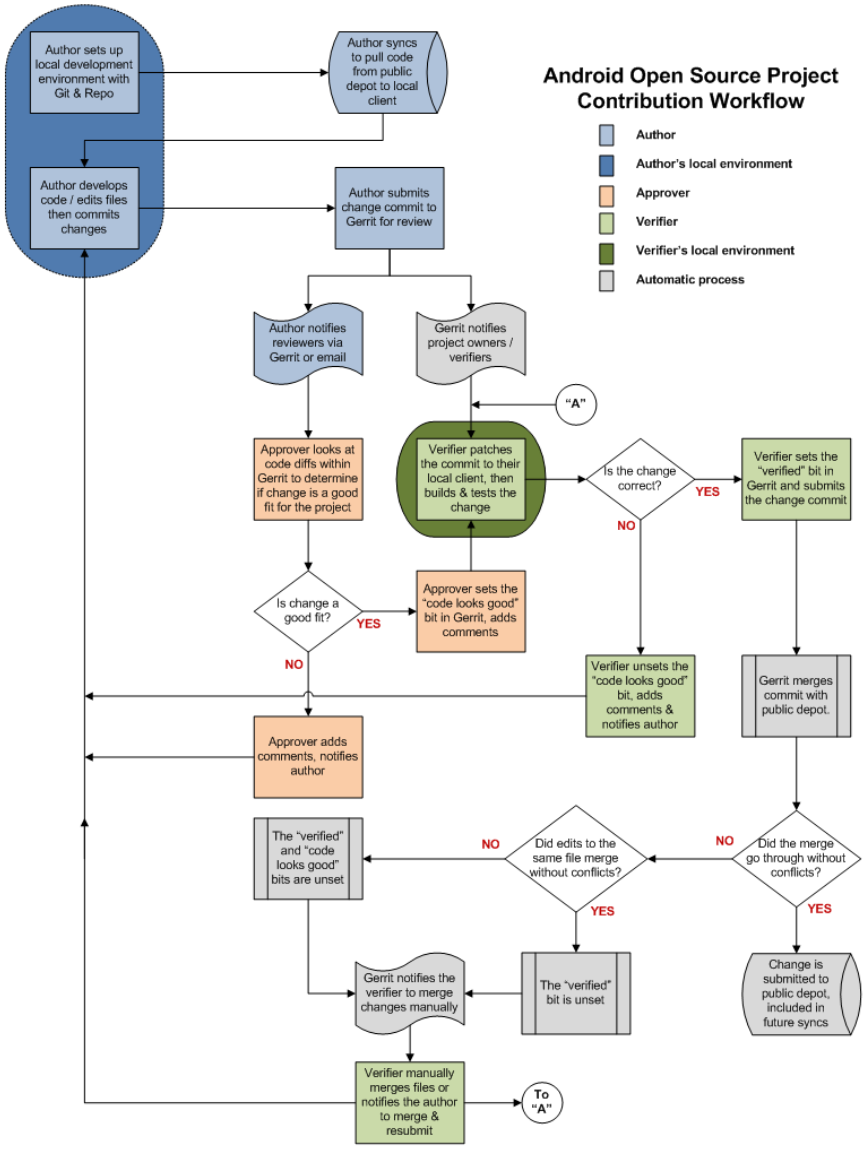
PhotoEffectsPro	0	0	0	0
PhotoCollagesCamera	0	0	0	0
PhotoPainterFree	0	0	0	0
WondersharePowerCam	0	0	0	0
Camera360Ultimate	0	0	0	0
PhotoGrid-CollageMaker	0	0	0	0
MP3Cutter&RingtoneMaker	0	0	0	0
SleepTimer	0	0	0	0
RealGuitarFree	0	0	0	0
RingtoneMaker	0	0	0	0
MusicVolumeEQ	0	0	0	0
UCBrowserMini	0	0	0	0
UC瀏覽器	0	0	0	0
Yahoo!スマホ最適化ツール	0	0	0	0
V3MobilePlus2.0	0	0	0	0
AdvancedDownloadManager	0	0	0	0
ScreenOffandLock	0	0	0	0
Dr.Webv.7Anti-virusLight	0	0	0	0
UCBrowserforAndroid	0	0	0	0
OperabrowserforAndroid	0	0	0	0
FirefoxBrowserforAndroid	0	0	0	0
OperaMinibrowserforAndroid	0	0	0	0
ElectricScreenLiveWallpaper	0	0	0	0
FLASHCODE:QRCodeLecteur	0	0	0	0
KingsoftOffice5.8(Free)	0	0	0	0
HDWallpapersforAndroid	0	0	0	0
CrackedScreenLiveWallpaper	0	0	0	0
BackgroundsHD(30MDownloads)	0	0	0	0
WhatsAppWallpaper	0	0	0	0
PVSTAR+(YouTubeMusicPlayer)	0	0	0	0
MYTF1Connect	0	0	0	0
MXPlayerCodec(ARMv6VFP)	0	0	0	0
네이버미디어플레이어	0	0	0	0
BBCiPlayer	0	0	0	0
BBCMediaPlayer	0	0	0	0
WondersharePlayer	0	0	0	0
FVD-FreeVideoDownloader	0	0	0	0
AVDDownloadVideoDownloader	0	0	0	0
LINESnapMovie	0	0	0	0
VLCforAndroidBeta	0	0	0	0
TubeVideoDownloader	0	0	0	0
MXPlayer	0	0	0	0
ZDcalendar-Chinesecalendar	0	0	0	0
OCDAPP(Official)	0	0	0	0
GoogleTranslate	0	0	0	0

<표 27> 어플리케이션의 구글코드 사용 여부
(1: 사용이 확인됨, 0: 사용이 확인되지 않음)

안드로이드 버젼	공개일자	API 버젼	버전 코드명
1.0	2008년 9월 23일	1	없음
1.1	2009년 2월 9일	2	없음
1.5	2009년 4월 30일	3	Cupcake
1.6	2009년 9월 15일	4	Donut
2.0	2009년 10월 26일	5	Eclair
2.0.1	2009년 12월 3일	6	
2.1	2010년 1월 12일	7	
2.2	2010년 5월 20일	8	Froyo
2.2.1	2011년 1월 18일		
2.2.2	2011년 1월 22일		
2.2.3	2011년 11월 21일		
2.3	2010년 12월 6일	9	Gingerbread
2.3.1	2010년 12월		
2.3.2	2011년 1월		
2.3.3	2011년 2월 9일	10	
2.3.4	2011년 4월 28일		
2.3.5	2011년 7월 25일		
2.3.6.	2011년 9월 2일		
2.3.7	2011년 9월 21일		
3.0	2011년 2월 22일	11	Honeycomb
3.1	2011년 5월 10일	12	
3.2	2011년 7월 15일	13	
3.2.1	2011년 9월 20일		
3.2.2	2011년 8월 30일		
3.2.3			
3.2.4	2011년 12월		
3.2.5	2012년 1월		
3.2.6	2012년 2월		
4.0	2011년 10월 19일	14	Ice Cream Sandwich
4.0.1	2011년 10월 21일		
4.0.2	2011년 11월 28일		
4.0.3	2011년 12월 16일	15	
4.0.4	2012년 5월 29일		
4.1	2012년 7월 9일	16	Jelly Bean
4.1.1	2012년 7월 23일		
4.1.2	2012년 10월 9일		
4.2	2012년 11월 13일	17	
4.2.1	2012년 11월 27일		
4.2.2	2013년 2월 11일		
4.3	2013년 7월 24일	18	
4.3.1	2013년 10월 3일		
4.4	2013년 10월 31일	19	KitKat

4.4.1	2013년 12월 5일		
4.4.2	2013년 12월 9일		

<표 28> 안드로이드 운영체제의 변화



<그림 30> 안드로이드 코드 반영 절차

(출처: http://source.android.com/source/life-of-a-patch.html, 2014년 6월 18일 방문)

Abstract

Expansion of Power Using Open Information Technology

Soojong Kim
Department of Sociology
The Graduate School
Seoul National University

This study analyzes the mechanism how open information technology expands specific actor's power. For this purpose, the case of Android operation system, the mobile operating system primarily led by Google, is chosen and studied in the perspective of actor-network theory and technological system theory.

The Android system, the network between human and non-human actors related to the Android operating system, is analyzed in the three domains and structures of dominance are identified from each domain. First, in the Android operating system domain, one can freely participate in the production process; however, Google controls merge and publication procedures and dominates the product as well as the production process. Second, in the mobile application domain, there are technological structures promoting production of a wide array of developers and actually use and development of Android applications are influenced by regional and cultural factors throughout the world. In contrast, most Android applications are technologically dependent on Google's information infrastructure. Third, in the domain of smartphone devices, thanks to the open-source license, manufacturers who want to modify open and free operating system to make their own, private smartphones prefer Android. Also, Android smartphones with various technological specifications are produced by a number of manufacturers. However, Google dominates their products using compatibility standards

which manufacturers should obey to gain Google's approval to install Google Play Store, the mobile application market software.

These structures of dominance combine and interact in the whole Android system. Google sets its private technology, Google Play Store, as an obligatory passage point and attracts device manufacturers and application developers into its technological system. The status of actors changes and Google possesses dominant position in the system. As the central actor of the open technological system, Google utilizes stability and compatibility as instruments to execute and expand its power.

The structures promoting openness and dominance can co-exist in a technological system. However, there can be various types of mixture, such as fragmented, politicized, closed types. In the era of technological openness, with the advent of Internet of Things (IoT), we should give attention to structures of dominance usually hidden behind openness. Social and governmental intervention is needed to avoid excessive expansion of private power and technological subordination which might damage publicness of technology and people's technological independence.

Keywords : Actor-Network Theory, Technological System, Open Technology, Power, Google, Android

Student Number : 2012 - 20081