OPENCHAIN

CURRICULUM

Core FOSS Compliance Version 1

Designed for Version 1 of the OpenChain Specification

Contents

CHAPTER 1

Intellectual Property란?

“Intellectual Property”란?

- 저작권: 저자의 원저작물을 보호
  - (근본적인 아이디어가 아닌) 표현을 보호
  - Software, 책, 시청각 자료, 반도체 마스크
- 특허: 새롭고, 유용하며, 너무 뻔하지 않은 (non-obvious) 발명
  - 혁신을 장려하기 위한 제한 독점
- 영업 비밀 : 기밀의 가치 있는 정보를 보호
- 상표 : 품의 출처를 나타내는 상표(단어, 로고, 슬로건, 색상 등)를 보호
  - 소비자와 브랜드 보호; 소비자를 혼란스럽게 하거나 상표 가치 하락하는 것을 방지하기 위함

이 장에서는 FOSS Compliance와 주로 관련이 있는 저작권과 특허를 중점으로 설명한다

Software에서의 저작권 개념

OPENCHAIN

- 기본 규칙: 저작권으로 창작 저작물을 보호한다
- 저작권은 일반적으로 도서, 영화, 그림, 음악, 지도와 같은 문학 작품에 적용된다.
- Software는 저작권에 의해 보호되며 기능(기능은 특허에 의해 보호)이 아닌 표현 (세부 구현 내용의 창의성) 이 보호된다.
- 저작권 소유자는 자신이 만든 저작물에 대해서만 통제권을 갖는다. (타인의 독립적인 저작물에 대해서는 통제권이 없음)

Software와 가장 관련 있는 저작권 권리

OPENCHAIN

- Software를 복제할 수 있는 권리 - 복제
- 파생저작물을 만들 수 있는 권리 - 수정
  - '파생저작물'이란 어떠한 한 저작물을 기반으로 만들었지만, 충분히 독창적인 부분이 추가되어서, 단지 복사가 아닌 저자의 새로운 저작물로 간주할 수 있는 것을 의미한다. (US law에서 사용하는 용어임에 유의)
- 배포할 수 있는 권리
  - 일반적으로 Software 일부를 복사하여 Source Code 혹은 Binary Form으로 다른 주체(개인 혹은 회사/기관 밖의 기관)에 제공하는 것을 배포라고 본다

참고: "파생저작물"이나 "배포"가 무엇인지에 대한 해석은 FOSS community나 FOSS 법률가들 사이에서 논의의 대상이다

Software에서의 특허 개념

OPENCHAIN

- 특허는 기능을 보호함 - Computer Program과 같은 동작 방법 등
  - 추상적인 아이디어나 자연법칙을 보호하지는 않는다
- 특허 소유권자는 누군가 독창적으로 만든 것과 상관없이 해당 기능의 수행을 금지할 권리가 있다
- 해당 기술을 사용하고자 한다면 특허 License (해당 기술을 사용, 제작, 판매, 수입할 수 있는 권리)를 취득해야 한다

License

- "License"는 저작권자, 특허권자가 사용허가 혹은 권리를 부여하는 것을 말한다
- License는 다음 사항들에 따라 제한 될 수 있다 :
  - 허용되는 사용 형태 (배포, 파생저작물 제작)
  - 독점 혹은 비독점 조항
  - 관할 지역
  - 영구적 혹은 시간제한
- License는 권한을 부여하기 위한 조건을 가질 수 있다. 즉, 해당 조건을 준수하는 경우에만 License를 취득할 수 있다는 의미이다
  - 예, 저작자 표시 제공, 동일 License 적용
- 보증, 면책, 지원, 유지 보수 등과 관련하여 계약상의 조항이 포함되는 경우도 있다

Check Your Understanding

- 저작권법은 어떤 유형의 자료를 보호하는가?
- Software에서 가장 중요한 저작권 권리는 무엇인가?
- Software가 특허의 대상이 될 수 있는가?
- 특허는 특허 소유자에게 권리를 부여하는가?
- 독자적으로 개발한 Software에 대해 제3자로부터 저작권 License 취득이 필요할 수도 있는가? Patent License는?

CHAPTER 2

FOSS Licenses 소개

FOSS License

- FOSS(Free and Open Source Software) License는 일반적으로 수정 및 재배포를 허용하는 조건하에 Source Code를 공개한다
- FOSS License는 저작자 표시, 저작권 진술 보존, Source Code 제공을 위한 약정서 제공과 같은 조건을 포함할 수 있다
- OSI (Open Source Initiative)는 OSD(Open Source Definition)를 정해놓고, 이에 부합하는 FOSS License를 승인해준다. OSI-approved FOSS License는 http://www.opensource.org/licenses/ 에서 확인할 수 있다

Permissive FOSS License

- Permissive FOSS license - 주로 제약이 아주 적은 FOSS License를 설명하기 위해 자주 사용되는 용어
- 예: BSD-3-Clause
  - BSD License는 저작권 고지와 License의 보증부인을 유지하는 조건으로 제한 없는 재배포를 허락하는 Permissive License의 한 예이다.
  - 이 License는 파생저작물의 홍보에 별도의 승낙 없이 기여자 이름의 사용하는 것을 제한하는 조항을 포함한다
- 다른 예: MIT, Apache-2.0

License 상호주의 (Reciprocity) & Copyleft License

- 어떤 License는 파생저작물 (혹은 동일 File, 동일 Program이나 다른 Boundary 내 Software) 배포 시 원저작물과 같은 조항으로 배포할 것을 요구한다
  - 이러한 원리는 "Copyleft", "상호주의", "유전" 효과라고 불린다
- GPL-2.0 내 상호주의 조항 예 :

"You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed...under the terms of this License."

- 예: GPL, LGPL, AGPL, MPL, CDDL의 모든 version

Proprietary Licens

- Proprietary Software License (or 상용 License or EULA)는 Software의 사용, 수정, 배포에 대한 제한을 갖고 있다
- Proprietary License에는 비용 지급, License 금액 등에 대한 내용이 포함되기도 한다
- Proprietary License는 Vendor 별로 고유하다. Vendor의 수만큼 다른 Proprietary License가 있을 것이고, 이를 사용하기 위해서는 각각 개별 검토가 필요하다
- FOSS 개발자들은 상업적 non-FOSS License를 설명하기 위해 "Proprietary License"라는 용어를 사용하기도 한다

다른 형태의 License

OPENCHAIN

- Freeware - Proprietary License이지만, 무료 혹은 아주 저렴하게 배포된 Software
  - 일반적으로 Source Code는 공개되지 않고, 파생저작물의 생성을 금지한다
  - Freeware Software는 대개 모든 기능이 동작하고 (사용 불가 기능 없음), 기한 제한 없이 사용할 수 있다
  - Freeware Software는 복사, 배포, 파생저작물 생성에 대해서 뿐만 아니라 사용 형태 (개인, 상용, 학업 등)에 대해서도 제한을 부과한다
- Shareware - 시험판으로 사용자에게 제공되는 Proprietary Software로써, 사용 기간에 제한이 있고, 제한된 기능을 무료로 사용할 수 있게 한다
  - Shareware의 목적은 잠재적인 고객에게 Program 사용 기회를 제공하여 full version software의 구매를 결정할 수 있게 하는 것이다
  - 일부 Shareware vendor들은 일단 무료로 사용하게 하여 확산을 시킨 후 고비용의 License 금액을 청구하는 경우가 있으므로 대부분 기업은 Shareware를 상당히 경계한다.
- Freewware와 Shareware는 FOSS가 아니다

Public Domain

- Public Domain은 법에 따라 보호받지 않는 지적 재산을 가리키며 누구든지 License 없이 사용할 수 있다
- 개발자들은 자신의 Software에 대해 Public Domain에 해당한다는 선언을 포함할 수 있다
  - 예, "All of the code and documentation in this software has been dedicated to the public domain by the authors."
  - Public Domain 선언이 FOSS License와 같은 것은 아니다
- 이러한 Public Domain 선언의 법적 집행 가능성은 FOSS Community 내에서 논쟁이 있는 부분이기도 하다
- 종종 Public Domain 선언이 보증 부인 등 다른 조항과 함께 사용될 수 있는데, 이 경우는 Software가 Public Domain 상태라기보다는 하나의 License 하에 있다고 볼 수 있다

License Compatibility

- License Compatibility는 License 조항들이 충돌하지 않음을 보장하는 Process이다.
  - 만약 한 License가 요구하는 내용을 다른 License가 금지한다면, 두 개의 Software 모듈의 결합이 License 의무 사항을 유발할 경우, 이 License들은 충돌하는 것이고 호환되지 않는다.
- 한 예로 GPLv2는 "파생저작물"에도 의무 사항을 부여한다.
  - 만약, GPLv2 Licensed 모듈과 결합한 두번째 Software 모듈이 GPLv2 Licensed 모듈의 파생저작물이 아니라면, 이 두번째 Software 모듈은 GPLv2의 대상이 아니다. "파생저작물"의 정의는 FOSS Community의 관점에 따라 다르다.
- Free Software Foundation은 License Compatibility를 설명하기 위해 다음 사례를 제공한다 :

  License p로 배포된 저작물은 q의 조항으로도 배포할 수 있다면,

  License p는 License q와 호환(혹은 q-compatible)된다.
- 예: GPL compatibility
  - MIT와 같은 많은 FOSS License들 그리고 LGPL은 GPL-compatible하다. 즉, 이런 FOSS License 하의 Source Code는 GPL하의 Source Code와 충돌 없이 결합할 수 있다; 이러한 결합으로 생성된 새로운 Program은 GPL이 적용되어야 할 것이다.
  - 다른 FOSS License와 Proprietary Software License는 GPL의 조항과 조건이 충돌하기 때문에 GPL-compatible 하지 않다. 단, 이러한 충돌은 GPL-2.0 Software의 파생저작물을 생성하는 결합하는 경우에 한해서만 고려될 필요가 있다.
  - 참고 : http://www.fsf.org/licensing/licenses/

고지 (Notice)

OPENCHAIN

파일 상단의 Text 주석과 같은 고지는 주로 작성자 및 License 정보를 제공한다. FOSS License는 또한 저작자 정보의 제공이나 수정 내용을 포함시키기 위해 Source Code 혹은 문서 내에 고지를 포함할 것을 요구하기도 한다.

- **저작권 고지** - 저작권자가 누구인지를 알리기 위해 저작물의 사본에 붙이는 식별자. 예: Copyright © A. Person (2016).
- **License 고지**- 제품에 포함된 FOSS의 License 조항과 조건을 알리기 위한 고지.
- **귀속정보 (Attribution) 고지** - 제품에 포함된 FOSS 원저작자의 신원을 나타내기 위해 제품 배포 시 포함하는 고지.
- **수정 내용 고지** – 파일 상단에 새로운 저작권 표시를 추가하는 등, 파일의 Source Code를 수정하였음을 알리는 고지.

Multi-Licensing

- Multi License란 둘 이상의 다른 조항과 조건으로 Software를 배포하는 경우를 말한다
  - 예를 들어, Software가 "Dual License"인 경우, 수신자는 두 License 중 하나를 선택하여 Software를 사용하거나 배포할 수 있다
- 참고 : Licensor가 두 개 이상의 License를 부과하는 상황과 혼동해서는 안된다. 이런 경우는 모든 License를 준수해야 한다

Check Your Understanding

- FOSS License란 무엇인가?
- Permissive FOSS License의 주요 의무 사항은 무엇인가?
- Permissive FOSS License에는 무엇이 있는가?
- License 상호주의는 무엇을 의미하는가?
- Copyleft-style License에는 무엇이 있는가?
- Freeware와 Shareware를 FOSS로 간주되는가?
- Multi-License란 무엇인가?

CHAPTER 3

FOSS Compliance 소개

FOSS Compliance 목표

OPENCHAIN

- **의무 사항이 무엇인지 파악한다.(FOSS의 사용을 발견하고 추적하라.)** Software를 개발하면서 포함하는 모든 FOSS Component(및 각각 확인한 License)를 식별, 추적, 보관하는 Process가 있어야 한다.

- **사용한 FOSS의 모든 License 의무 사항을 준수한다. 귀사의** Program은 조직의 사업 형태에서 비롯되는 일반적인 FOSS의 Use Case를 식별하고 처리해야 한다.

준수해야 할 Compliance 의무 사항은 무엇인가?

OPENCHAIN

관련된 License에 따라 다음의 의무 사항이 있다 :

- **저작자 표시 및 고지.** 저작권 및 License Text를 Source Code 및/또는 제품 문서 혹은 User Interface에 포함하여 후속 사용자들이 Software의 출처와 License에 따른 권리를 알 수 있게 한다
- **Source Code 공개.** 원저작물, 결합저작물, 수정사항에 대해 Source Code뿐만 아니라 Build Script (Build Process를 제어하는 Script)를 제공한다

이러한 의무 사항은 다음과 같은 주요 Event에 따라 유발될 수 있다 :

- 외부 배포
- 수정 여부

FOSS 조건 및 제한 사항

OPENCHAIN

사용하는 FOSS License에 따라 다음 조건 및 제한사항을 하나 이상 준수해야 할 수 있다. :

- 저작권 (및 기타) 고지 유지
- License 사본 제공
- 수정 내용 고지 제공
- 수정 version은 혼란을 방지하기 위해 다른 이름을 사용해야만 함
- (수정 여부와 관계없이) 소스 코드에 대한 접근 권한 제공
- 수정 version (파생 저작물)에 대해 같은 License 유지
- 저작자 표시 제공
- Project 또는 저작권자 이름이나 상표를 사용하지 말 것
- 원 License 하에 부여된 권리를 타인에게 제한하지 말 것
- 종료 조항 (위반할 경우, License를 잃음)

FOSS Compliance Trigger : 배포 (Distribution)

OPENCHAIN

- 외부 주체로 자료 전파
  - User의 컴퓨터 혹은 모바일 장치에 다운로드 된 Application
  - User의 컴퓨터에 다운로드 된 Javascript, Web Client 또는 다른 코드
- 일부 FOSS License의 경우, 컴퓨터 Network을 통한 접근도 "Trigger Event"가 될 수 있다. Trigger는 “컴퓨터 Network을 통해 원격으로 상호작용하는 User”이다..
  - 일부 License는 Server에서 동작하는 Software에 대해서도 Access를 허용하도록 Trigger Event를 정의한다. (예: Software가 수정된 경우 Affero GPL의 모든 version)

FOSS Compliance Triggers: 수정

OPENCHAIN

- 기존 Program의 변경 (예: File내 Code의 추가, 삭제, Component 결합)
- 수정하는 것으로 파생저작물이 만들어 질 수 있으며, FOSS 저작자는 수정하는 것에 대해 의무를 제한하거나 부과할 수 있다.
- 수정은 다음과 같은 FOSS 의무 사항을 유발시킬 수 있다.:
  - 수정 내용 고지
  - 동반 Source Code 제공

FOSS Compliance Program

FOSS Compliance에 성공한 조직은 자신만의 FOSS Compliance Program (Policy, Process, Training, Tool 등으로 구성)을 갖고 있다. 이를 통해 :

1. 상용 제품에 FOSS를 효과적으로 사용할 수 있게 한다.
2. FOSS 개발자의 권리를 존중하고, License 의무 사항을 준수한다.
3. Open Community에 참여하고 기여한다.

Compliance를 위한 실천 사항

OPENCHAIN

다음 사항들을 처리하기 위한 Business Process와 충분한 인력을 준비한다:

- FOSS Software의 출처와 License 식별
- 개발 Process 내에서 FOSS Software 추적
- FOSS 검토 수행 및 License 의무 사항 확인
- 제품 배포 시 License 의무 사항 이행
- FOSS Compliance Program, 정책 수립 및 Compliance 의사 결정에 대한 감독
- 교육

Compliance의 이점

OPENCHAIN

강력한 FOSS Compliance Program의 이점은 다음과 같다 :

- FOSS의 이점과 조직에 미치는 영향에 대한 이해 증진
- FOSS의 이용과 관련된 비용과 위험에 대한 이해 증진
- FOSS Community와 FOSS 기관들과의 관계 개선
- 사용 가능한 FOSS Solution들에 대한 지식 증가

Check Your Understanding

OPENCHAIN

- FOSS Compliance는 무엇을 의미하는가?
- FOSS Compliance Program의 두 가지 주요 목표는 무엇인가?
- FOSS Compliance Program의 중요한 Business 실천 사항을 열거하고 설명하시오.
- FOSS Compliance Program의 이점은 무엇인가?

CHAPTER 4

FOSS Review를 위한 주요 Software 개념

어떤 정보를 수집해야 하나?

FOSS의 사용법을 분석할 때, FOSS Component의 정체, 출처 및 어떤 방법으로 사용될 것인지에 대한 정보를 수집한다. 여기에는 다음 사항이 포함될 수 있다:

- Package 이름
- Version
- 원본 Download URL
- License와 License URL
- 설명
- 수정 내용 설명
- Dependency List
- 제품 내 사용 방법
- Package가 포함된 제품의 초도 배포일
- Source Code 입수 가능성
- Source Code를 유지할 곳
- Package가 이전 다른 환경에서 사용 승인된 적이 있는지 여부
- 수출 통제 대상 기술 포함 여부
- *외부 vendor로부터 입수한 경우:*
  - 개발 팀의 연락처
  - License 의무 사항을 충족시키기 위한 저작권 고지, 귀속 정보, vendor 수정 사항에 대한 Source Code

Component를 어떻게 사용할 것인가?

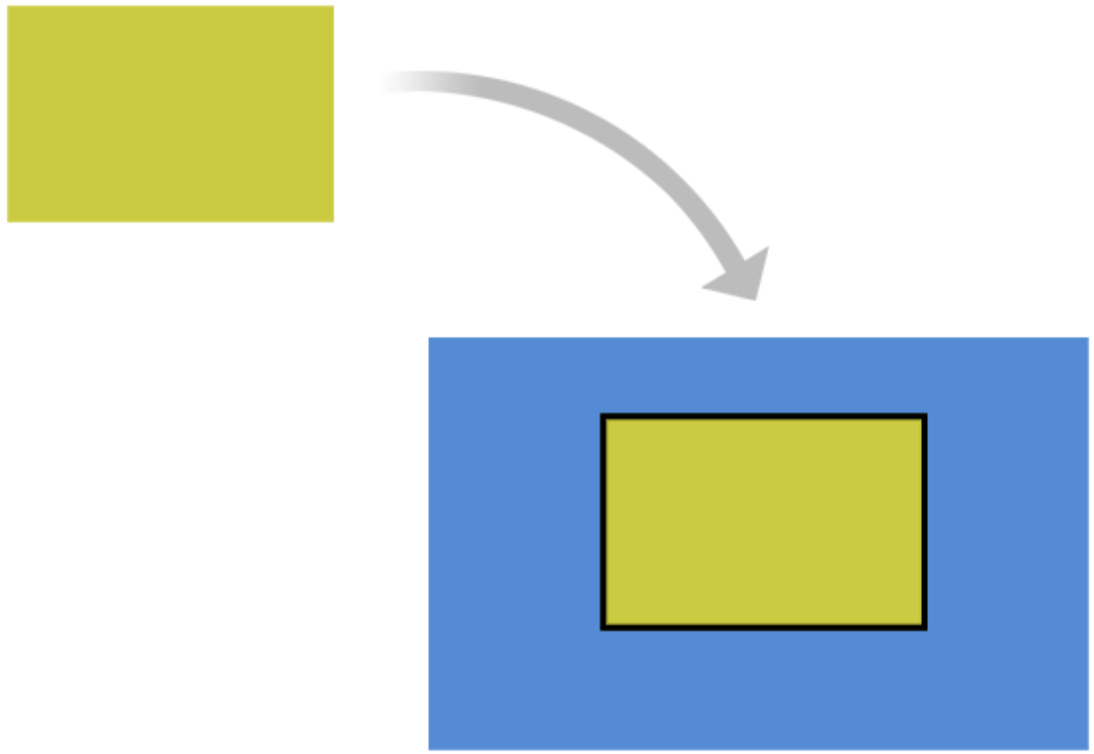
OPENCHAIN

일반적인 시나리오는 다음과 같다 :

- 통합 (Incorporation)
- Linking
- 수정 (Modification)
- 번역 (Translation)

통합 (Incorporation)

개발자는 FOSS Component의 일부를 Software 제품에 복사 할 수 있다.



관련 방식은 다음과 같다:

- 통합 (Integrating)
- 병합 (Merging)
- 붙여넣기 (Pasting)
- 개작 (Adapting)
- 삽입 (Inserting)

Linking

개발자는 FOSS Component를 Software 제품에 Link 혹은 Join할 수 있다.

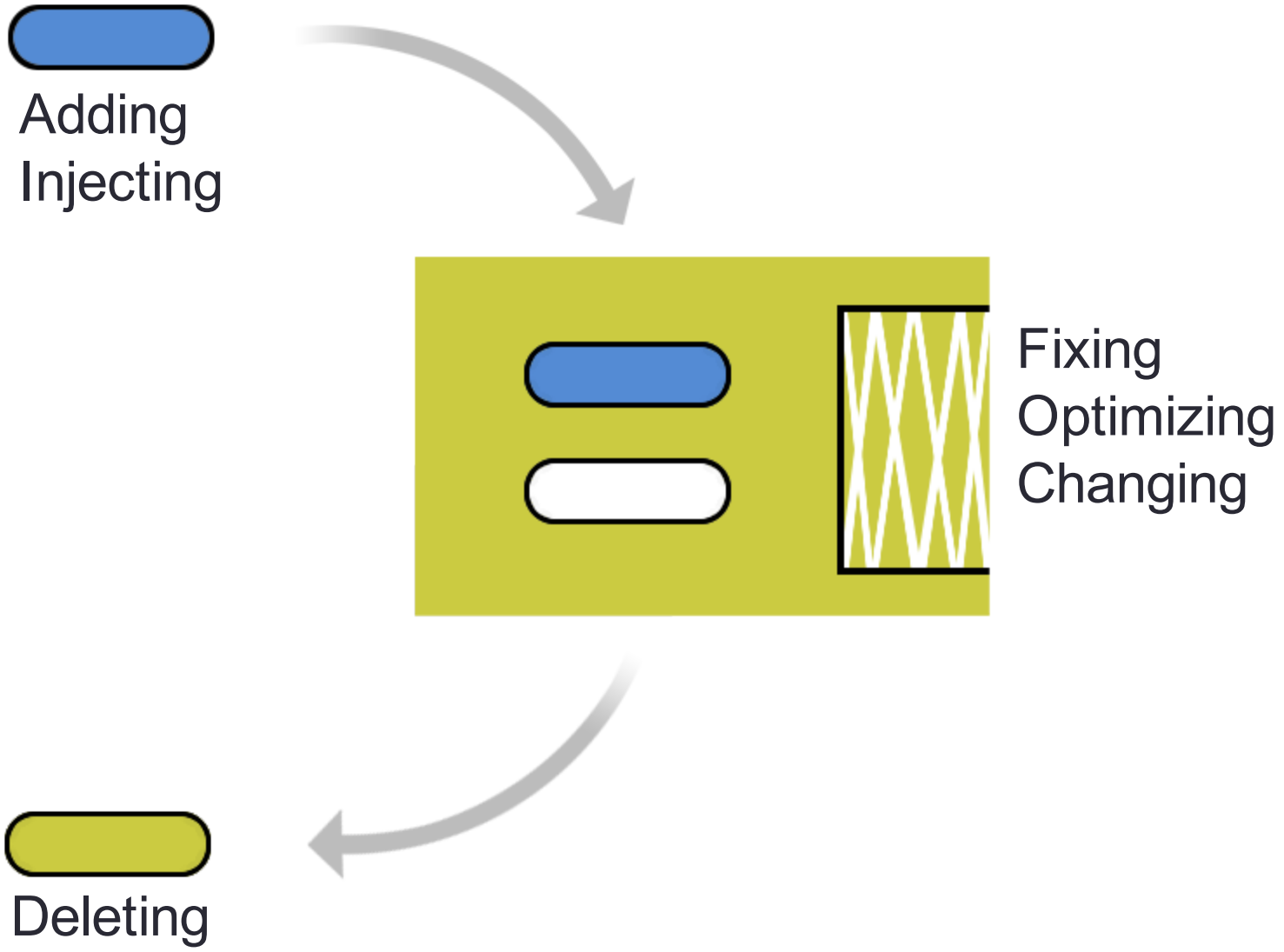
관련 방법은 다음과 같다:

- 정적/동적 Linking
- Pairing
- 결합 (Combining)
- 활용 (Utilizing)
- Packaging
- 상호 의존성 만들기

수정

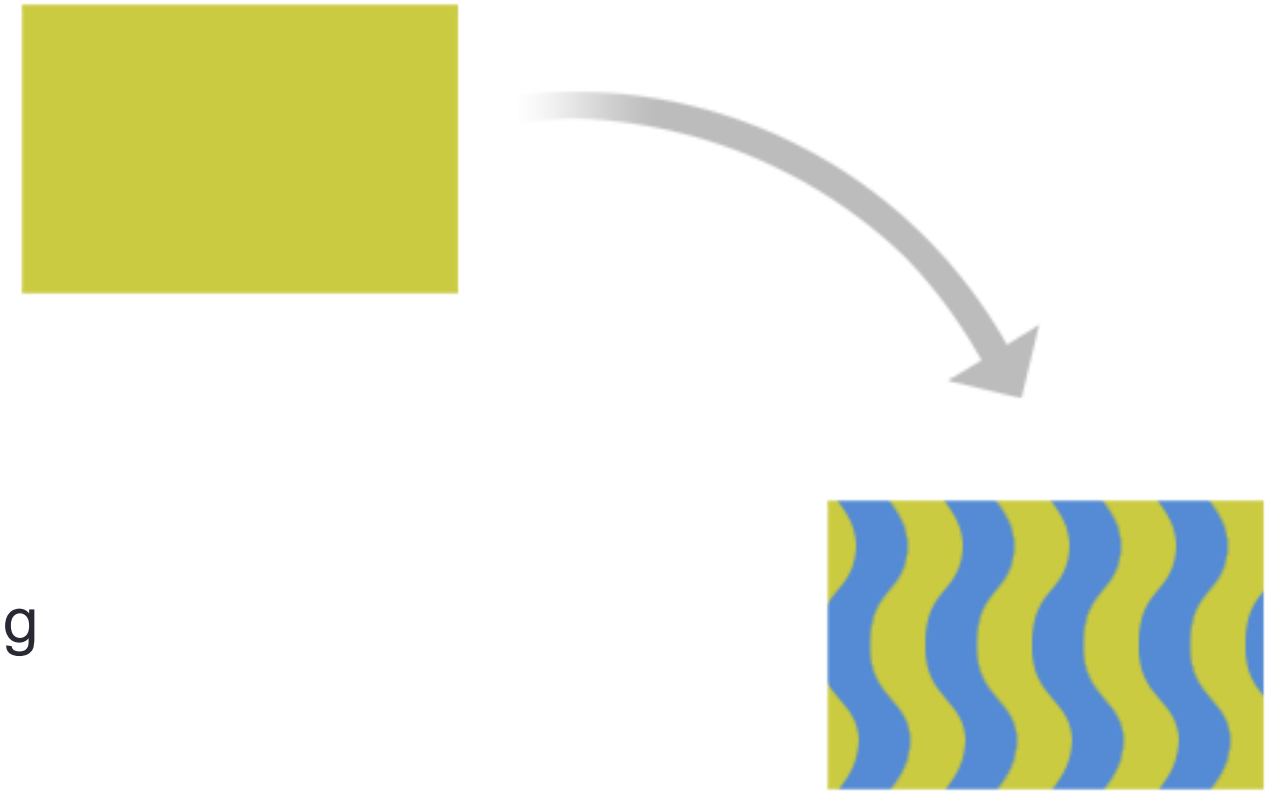
개발자는 다음과 같이 FOSS Component를 수정할 수 있다:

- FOSS Component에 새로운 Code 추가/삽입
- FOSS Component의 수정, 최적화 혹은 변경
- Code 삭제 혹은 제거



번역

개발자는 Code를 한 상태에서 다른 상태로 변환할 수 있다.



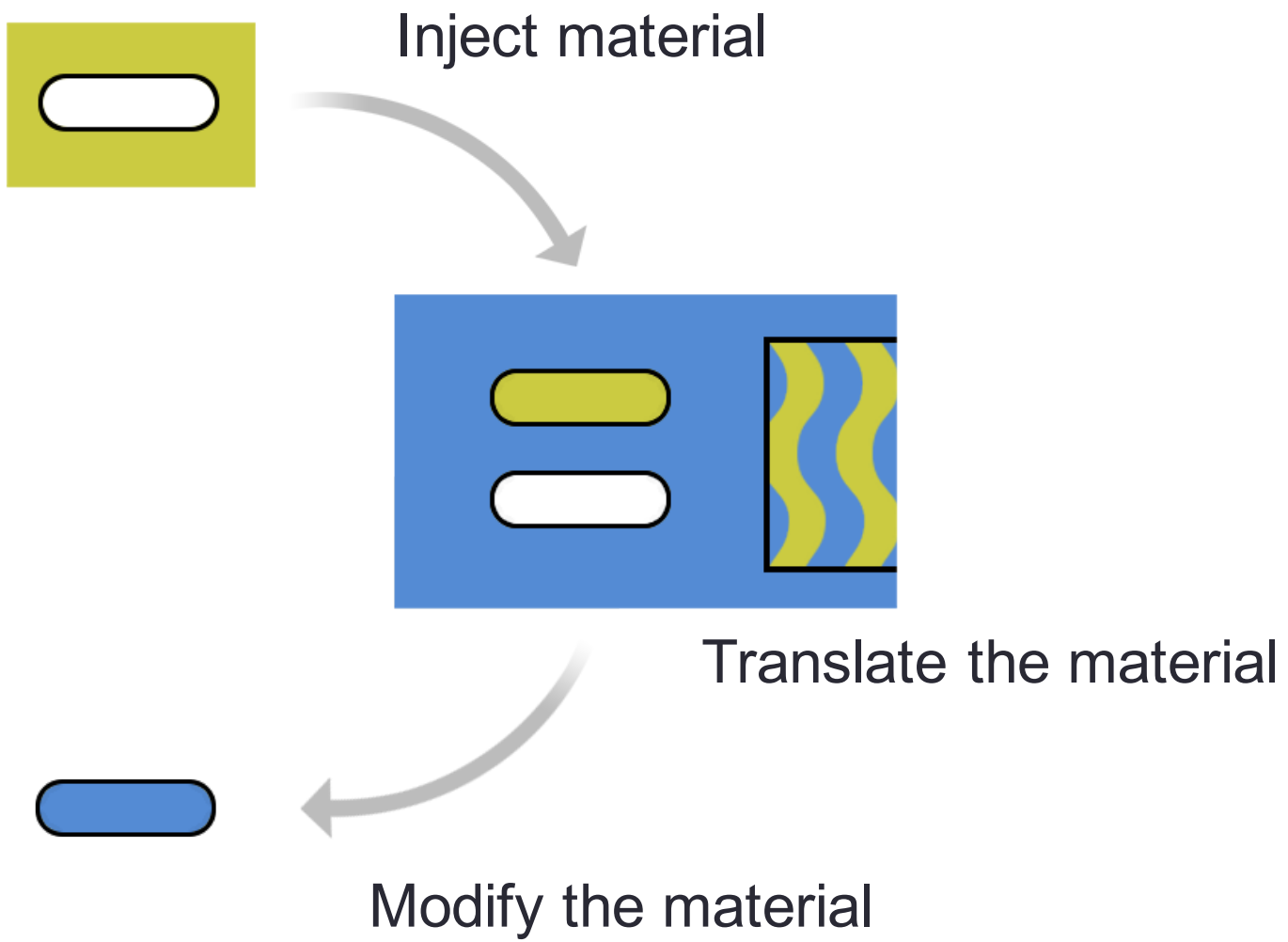
예:

- 중국어에서 영어로 번역
- C++에서 Java로 변환
- Mask나 Net List에서 VHDL Compiling
- Binary로 Compiling

개발 Tools

개발 Tool이 뒤에서 이러한 동작 중 일부를 수행할 수도 있다.

예를 들어, Tool이 동작하면서 산출물 내에 자체 Code의 일부를 추가시킬 수 있다.



FOSS Component는 어떻게 배포되는가?

OPENCHAIN

- Software를 받는 사람은 누구인가?
  - 고객/파트너
  - Community Project

- 전달 형태는 무엇인가?
  - Source Code 전달
  - Binary 전달
  - Hardware에 Pre-loaded

Check Your Understanding

OPENCHAIN

- Software의 License가 무엇인지 이해하는데 도움이 되는 정보는 무엇인가?
- 누가 Software에 License를 부여하는지 확인하기 위해 도움이 되는 정보는 무엇인가?
- 통합 (incorporation)이란?
- 수정 (modification)이란?
- Linking이란?
- 번역 (translation)이란?
- 배포에 대해 검토할 때 중요한 요소는 무엇인가?

CHAPTER 5

FOSS Review 실행

FOSS Review

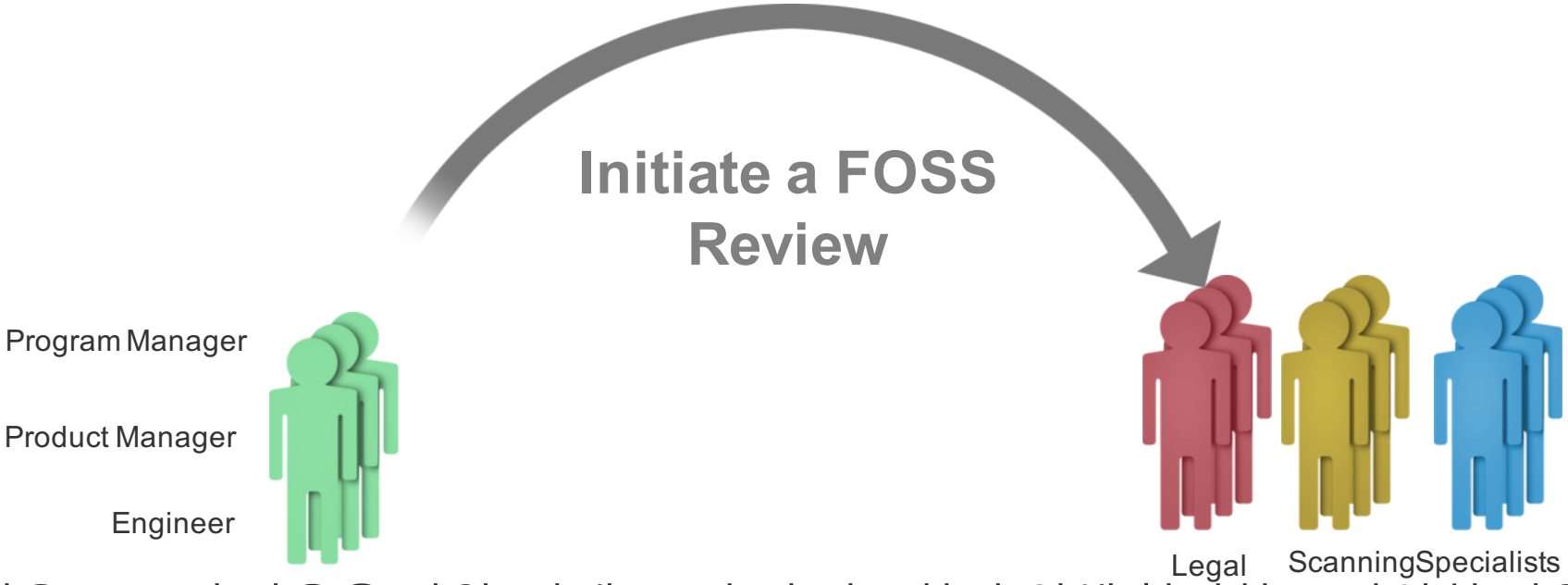
- FOSS Compliance Program의 핵심 요소는 FOSS Review Process로서, 이를 통해 회사는 FOSS 의무 사항을 분석하고 결정할 수 있다
- FOSS Review Process는 다음 과정을 포함한다 :
  - 관련 정보 수집
  - License 의무 사항 분석 및 결정
  - 회사 정책 및 Business 목표를 고려한 가이드 제공

FOSS Review 시작

FOSS Review Process는 Program/Product Manager, Engineer 및 FOSS와 관련하여 일하는 사람들이 Access 할 수 있어야 한다.

참고 : 이 Process는 외부 vendor로부터 FOSS-based Software를 제공받을 때도 시작할 수 있다.

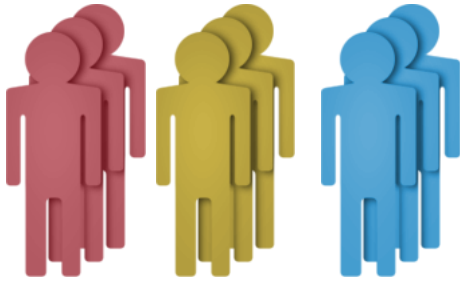
FOSS Review Team



FOSS Review 팀은 FOSS의 사용을 지원, 안내, 조정 및 검토하기 위해 협력하는 다양한 지원 그룹에 중요성을 알리고 참여시킨다. 이 팀은 다음을 포함할 수 있다:

- License 의무 사항을 식별하고 확인하는 법무팀
- FOSS 사용을 식별하고 추적하는 것을 도와주는 Scanning 및 Tooling 지원팀
- FOSS의 사용에 영향을 받을 수 있는, 사업 이익, 상용 License, 수출 Compliance 등을 담당하는 전문가

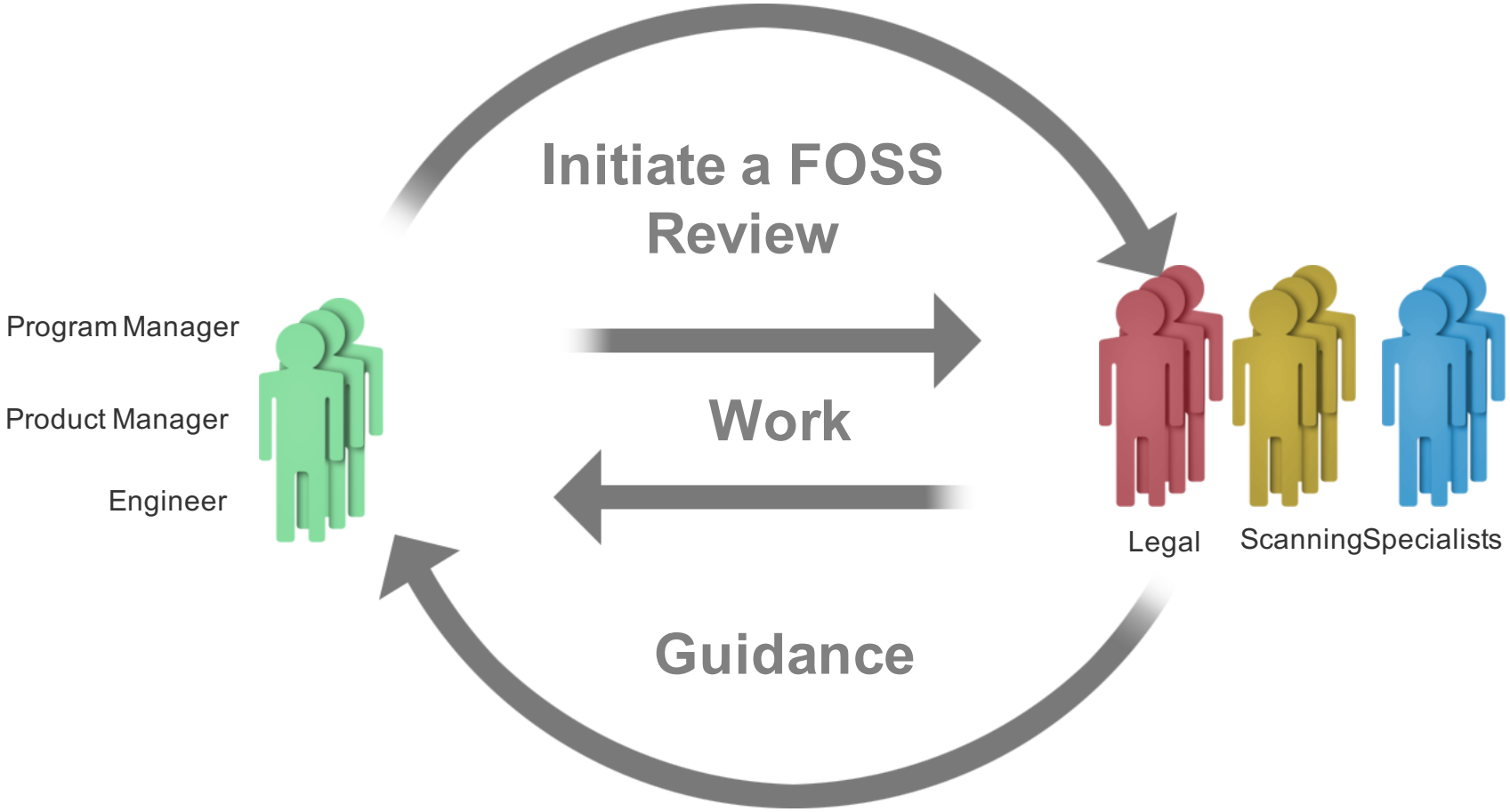
제안된 FOSS 사용 분석



FOSS Review Team은 가이드를 제공하기 전에 다음과 같은 이슈를 포함하여 수집한 정보를 평가해야 한다:

- 완전성, 일관성, 정확성 (감춰진 FOSS의 사용을 조사하는데 Code Scanning Tool이 사용될 수 있음)
- 선언된 License가 Code File에 있는 것과 일치하는가?
- License가 Software의 제안된 사용을 실제 허용하는가?

FOSS Review 작업



FOSS Review Process를 통한 업무는 상호작용을 한다. 이 업무는 Engineering, 사업 및 법무팀을 포함한 분야를 넘나들며, 모든 당사자가 근본적인 이슈를 이해할 수 있도록 Follow-up 논의를 요구할 수 있다. 궁극적으로 이 Process는 FOSS 사용에 대한 명확한 지침을 제공해야 한다.

FOSS Review 감독

FOSS Review Process는 관련 당사자간에 이견이 있거나 결정이 특별히 중요한 경우에 대해 충분한 감독을 해야 한다.

Check Your Understanding

OPENCHAIN

- FOSS Review의 목적은 무엇인가?
- FOSS Component를 사용하기 원한다면, 제일 처음 어떤 활동을 취해야 하는가?
- FOSS Review를 위해서 수집할 수 있는 정보는 무엇인가?
- 외부 vendor로부터 입수한 FOSS Component를 검토할 때 추가적으로 중요한 정보는 무엇인가?
- 이러한 정보의 품질을 평가하기 위해 취할 수 있는 조치는 무엇인가?
- FOSS를 사용함에 있어서 문의가 있을 경우, 무엇을 해야 하는가?

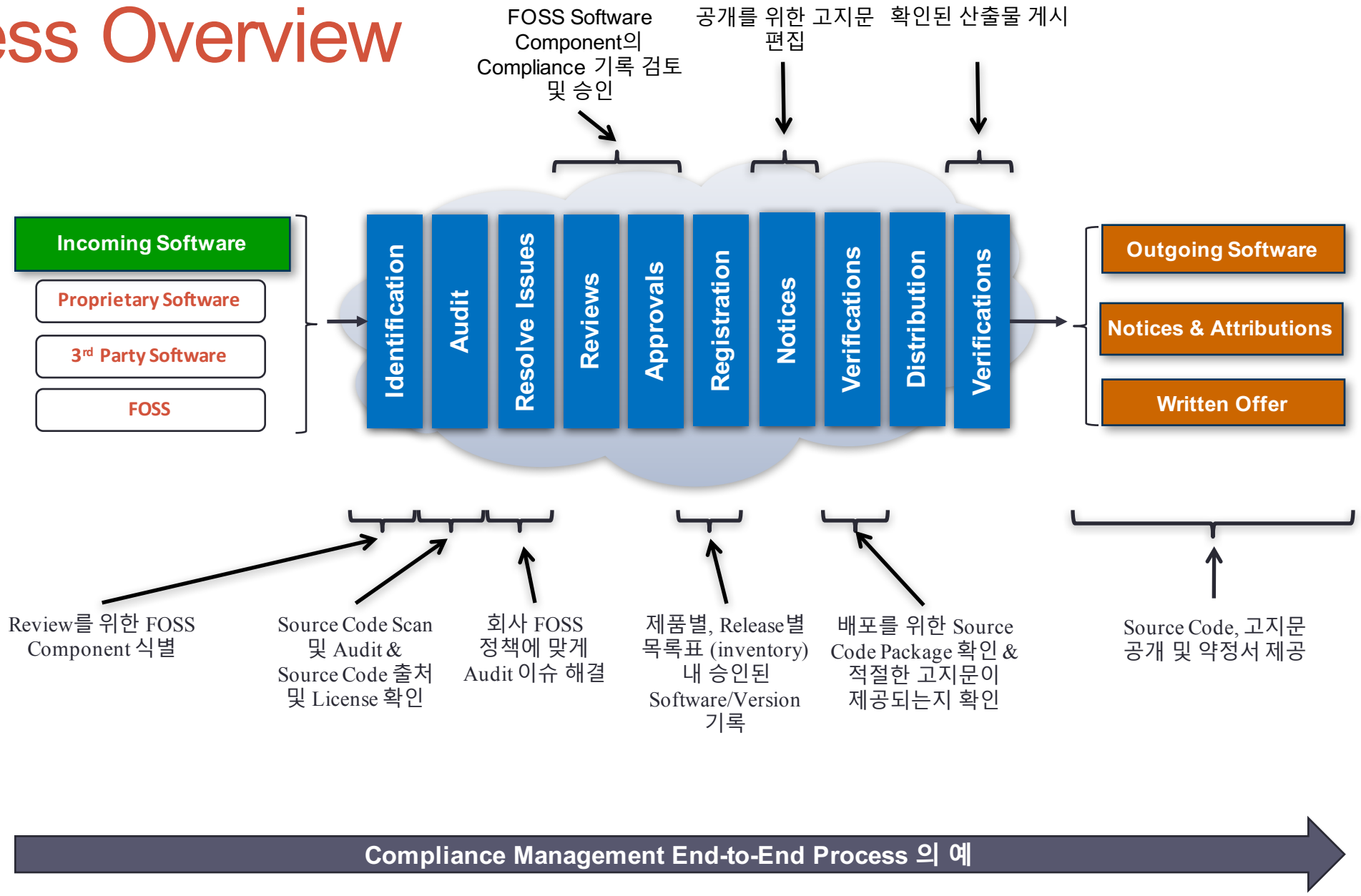
CHAPTER 6

End to End Compliance Management (Process 예)

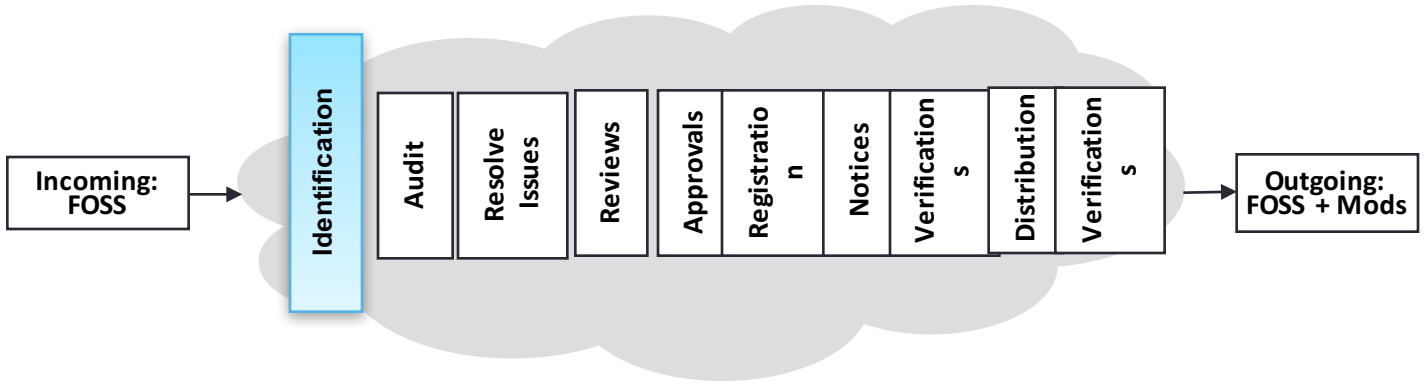
소개

- Compliance Management는 제품 (혹은 OpenChain Specification 에서의 "Supplied Software") 에 사용된 FOSS의 유입 및 배포를 제어하는 일련의 활동으로 구성된다.
- Compliance를 위한 기업 실사(due diligence)의 결과는 Supplied Software내 사용된 모든 FOSS를 식별하고, 모든 FOSS License 의무 사항이 충족된다고 보장하는 것이다.
- 이 장에서는 이러한 Process의 예를 제공하여 내부 Process를 구축하거나 개선하기 위한 도움을 제공한다.

Process Overview



FOSS 사용의 확인 및 추적



모든 Sourced에 대해 FOSS를 확인하고 추적한다

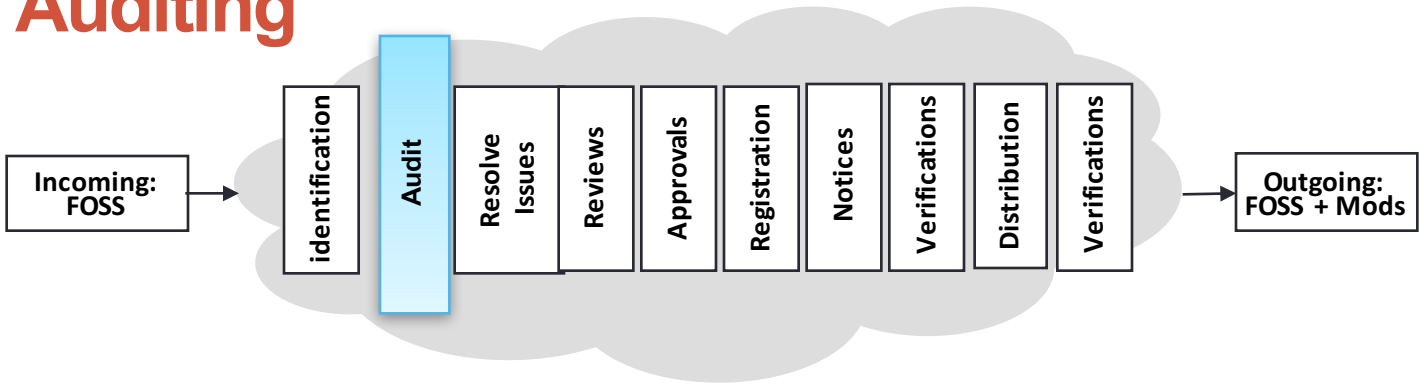
- 전제조건:

  Process는 다음 중 하나의 이벤트로 시작할 수 있다:
  - 개발팀에서 FOSS Component의 검토 혹은 외부 배포 요청
  - 적절한 승인 없이 사용된 FOSS의 발견
  - 3rd Party Software의 일부로 사용된 FOSS의 발견

- 단계:
  - 접수된 요청 기록
  - 전체 Platform에 대한 Scan을 수행할 수 있음
  - 3rd Party 제공 Software에 대한 실사 (Due Diligence)
  - 접수된 요청 없이 repository에 추가된 모든 FOSS Component에 대해 인식 및 검토

- 결과:
  - FOSS에 대한 Compliance 기록 생성 (혹은 갱신)
  - Source Code를 Scan하거나 Review하도록 Audit 요청

Source Code Auditing

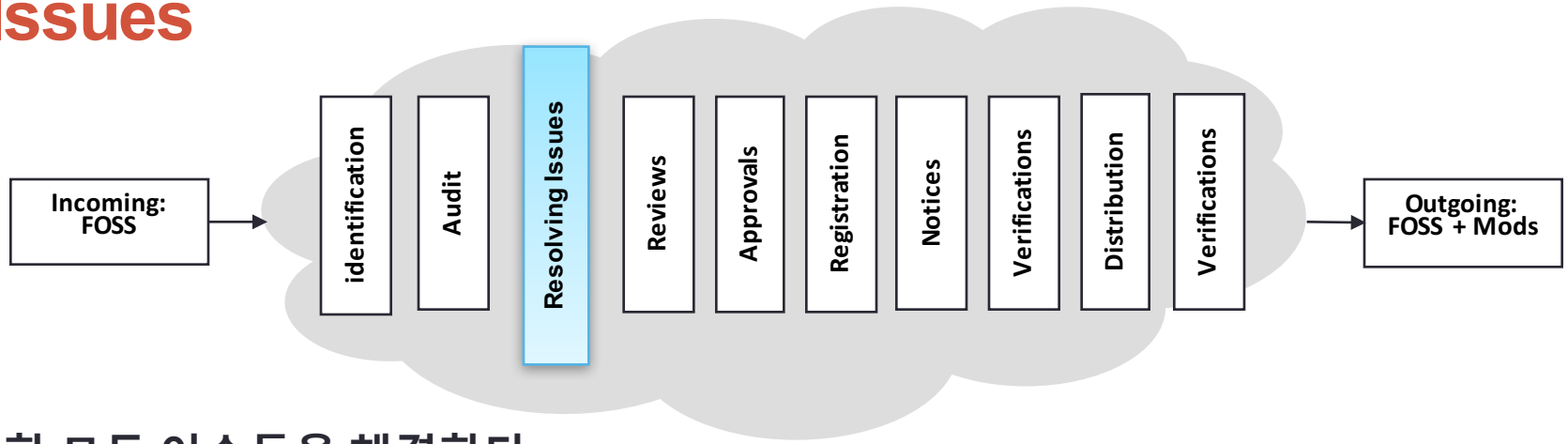


FOSS Component를 식별하여 출처 및 License를 확인한다

- 전제조건:
  - 개발팀은 FOSS 사용에 대한 정보가 담긴 Compliance 기록을 제공한다
  - 개발팀에서 제공한 기록이 없을 경우, 발견되는 FOSS Component로 기록을 만들 수 있다
- 단계:
  - Audit을 위한 Source Code 식별
  - Source Code Scan을 위해 Software Tool의 사용이 가능하다
  - Audit 혹은 Scan을 통해 발견된 부분은 Code의 적절한 출처에 대해 검토 및 확인되어야 한다
  - Audit 혹은 Scan은 Software 개발 및 Release Lifecycle에 따라 반복적으로 수행되어야 한다
- 결과:

  Source Code의 출처와 License를 식별한 Audit Report

Resolving Issues

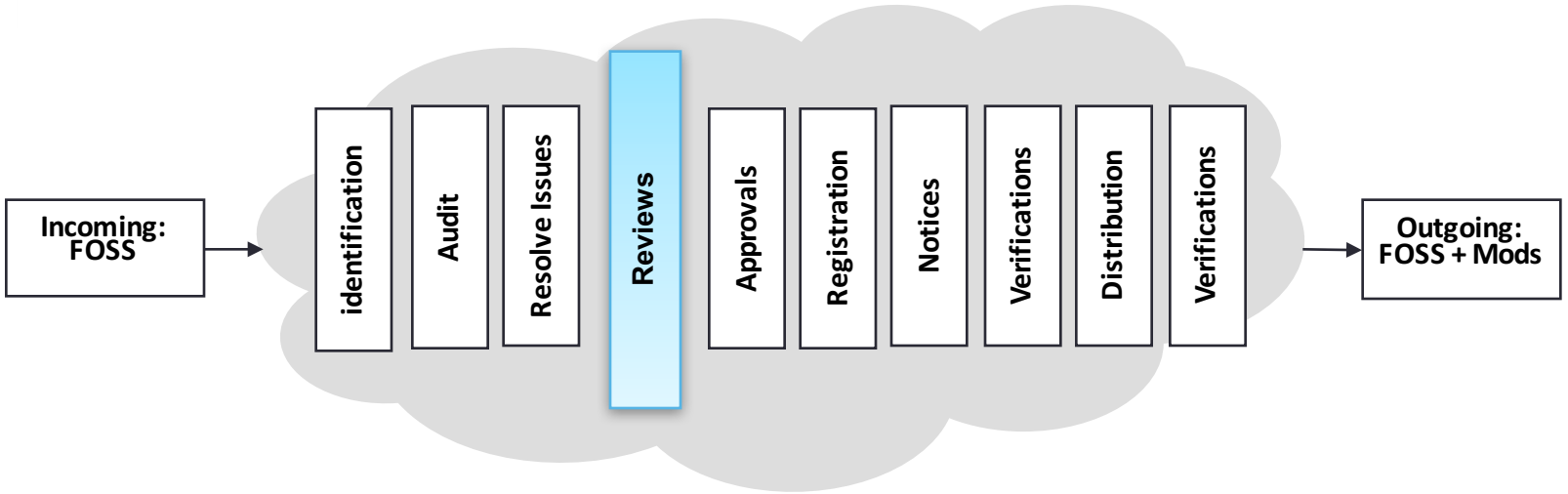


Audit을 통해 확인한 모든 이슈들을 해결한다

- 전제조건:
  - Source Code Audit 혹은 Scan을 완료하였다.
  - Audit Report는 Source Code의 출처 및 License를 식별하고, 추가 조사가 필요한 file에 대해 표시하였다.
- 단계:
  - FOSS 정책과 충돌하는 Audit Report내의 이슈 해결을 위해 적합한 Engineer에게 피드백을 제공한다.
  - Engineer와 상의하여 문제가 해결되었는지를 확인한다.
- 결과:

  Report내 표시된 각 파일의 해결 및 충돌되는 License의 해결

Review 수행



Audit Report를 검토하고 발견된 이슈들이 해결되었는지를 확인한다

전제조건:

- Source Code가 Audit되었다
- 모든 확인된 이슈들은 해결되었다

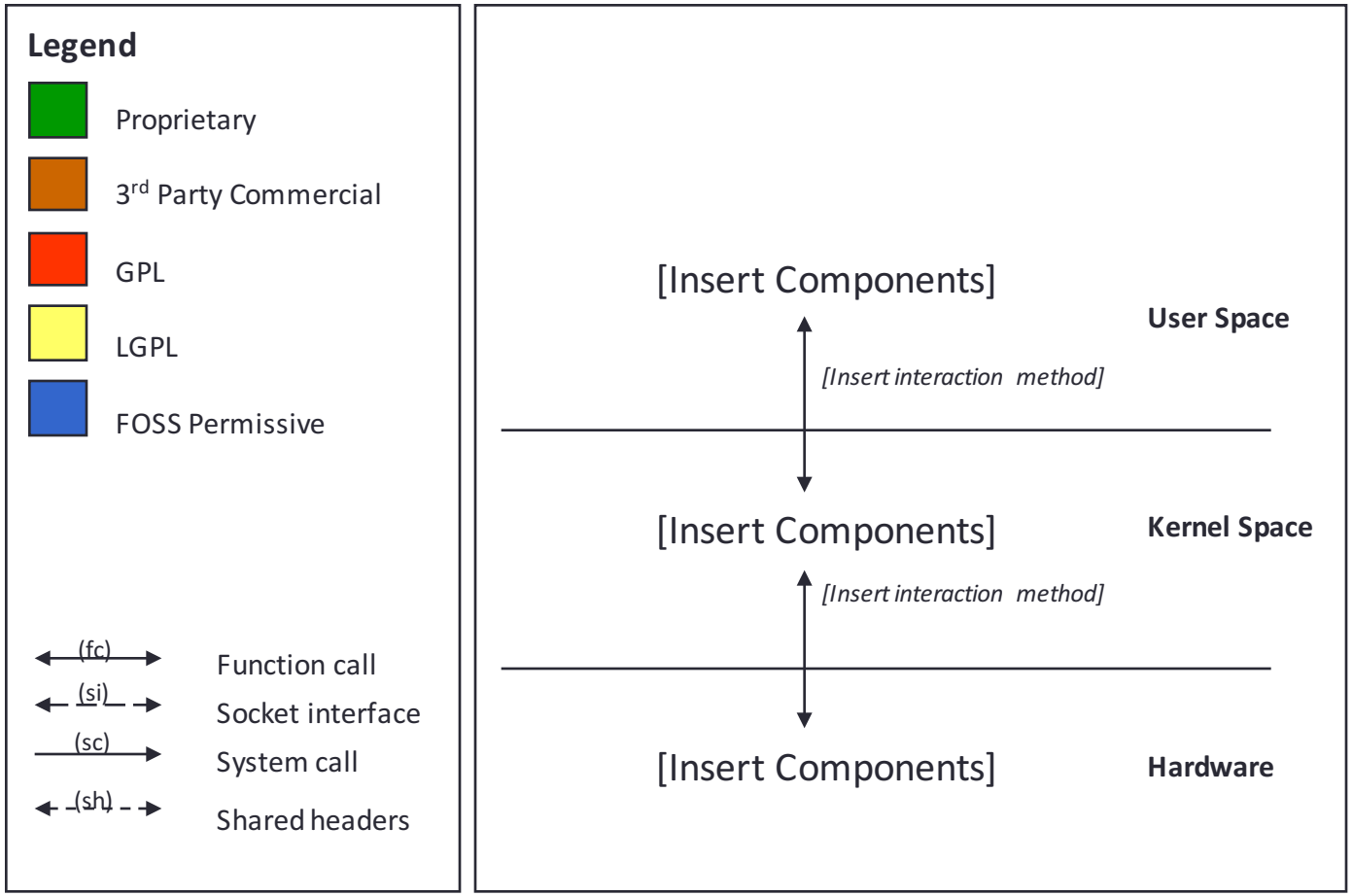
단계:

- 검토하는 직원에게 적절한 수준의 권한을 제공한다.
- Audit된 Source Code에 대해 FOSS Review를 수행하고, Software Architecture 및 FOSS 사용을 검토한다. (다음 Slide의 template참고)
- FOSS License들의 의무 사항을 확인한다.

결과:

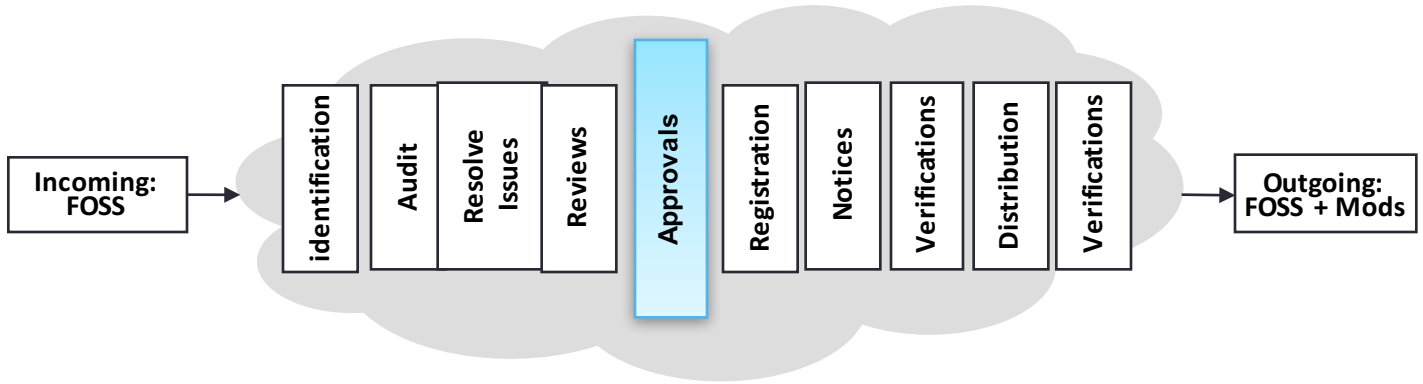
- Audit Report의 Software가 FOSS 정책에 순응하는지 확인한다
- Audit Report 결과를 보존하고, 다음 단계 (Approval) 를 위해 해결된 이슈들을 표시해둔다

Architecture Review (Example Template)



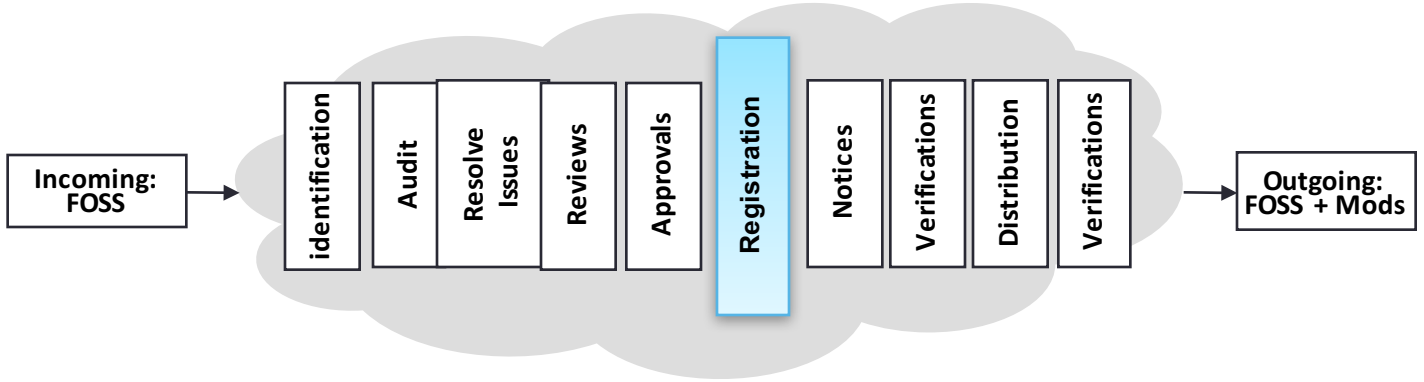
Approvals

- 이전 단계에서의 Software Audit 및 Review 결과에 따라 Software 사용이 승인될 수도, 되지 않을 수도 있다
- 승인된 FOSS Component의 Version, 승인된 Component의 사용 모델 및 FOSS License에 따른 의무사항들이 명시되어야 한다
- 승인은 적절한 수준의 권한에서 이루어져야 한다

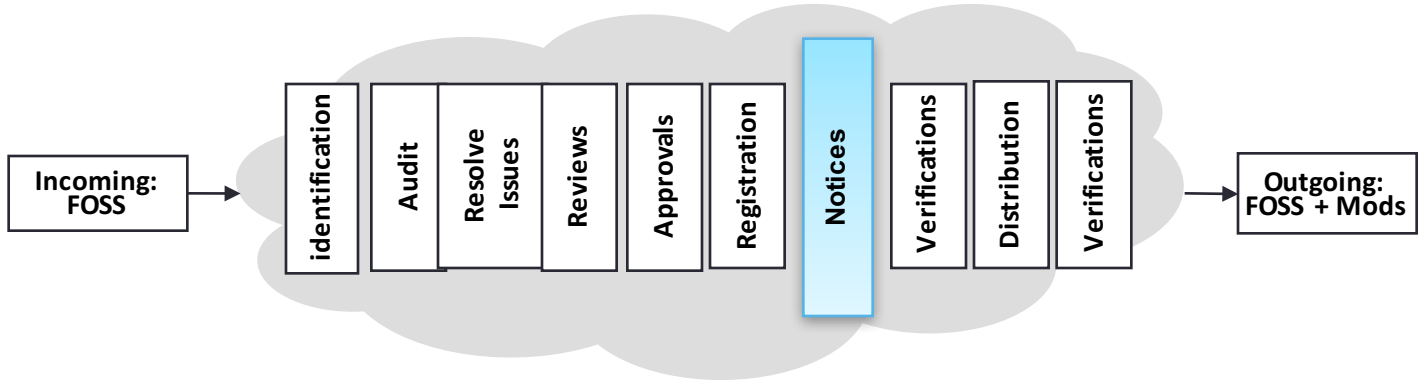


Registration / Approval Tracking

- FOSS Component가 제품에서의 사용이 승인되면, 해당 제품의 Software 목록표에 추가되어야 한다.
- 승인 및 이에 대한 조건은 Tracking System에 등록되어야 한다.
- Tracking System은 FOSS Component의 신규 version 혹은 새로운 사용 모델이 제안된 경우 새로 승인이 될 수 있도록 해야 한다.

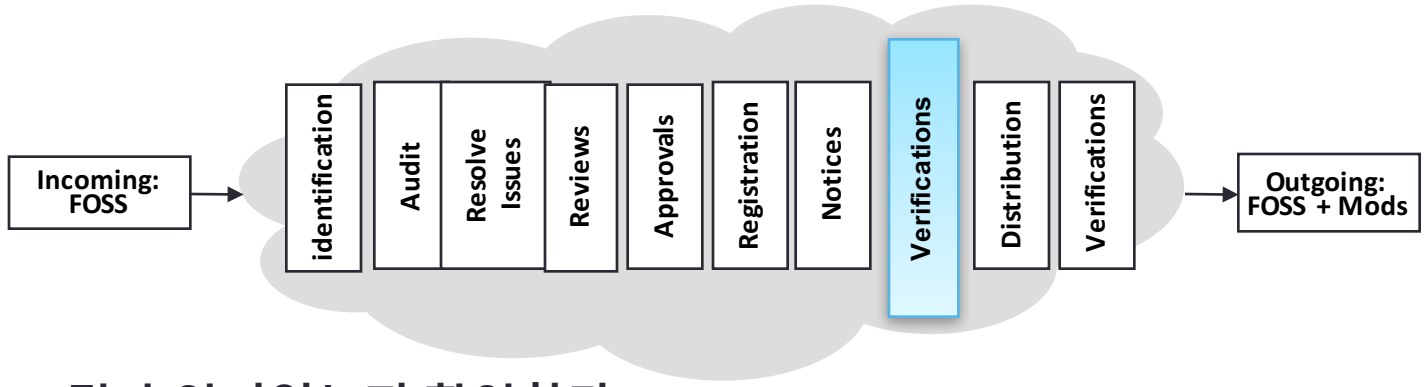


Notices



- **Release하는 제품에 사용된 모든 FOSS에 대해 적절한 고지를 준비한다:**
  - 모든 저작권 및 저작자 고지를 제공하기 위해 FOSS의 사용을 표시한다
  - 제품의 최종 사용자에게 FOSS Source Code 사본을 어떻게 얻을 수 있는지 알린다 (GPL, LGPL과 같이 Source 공개가 필요한 경우)
  - 필요에 따라 제품에 포함된 FOSS Code에 대한 License Agreement의 전체 text를 제공한다

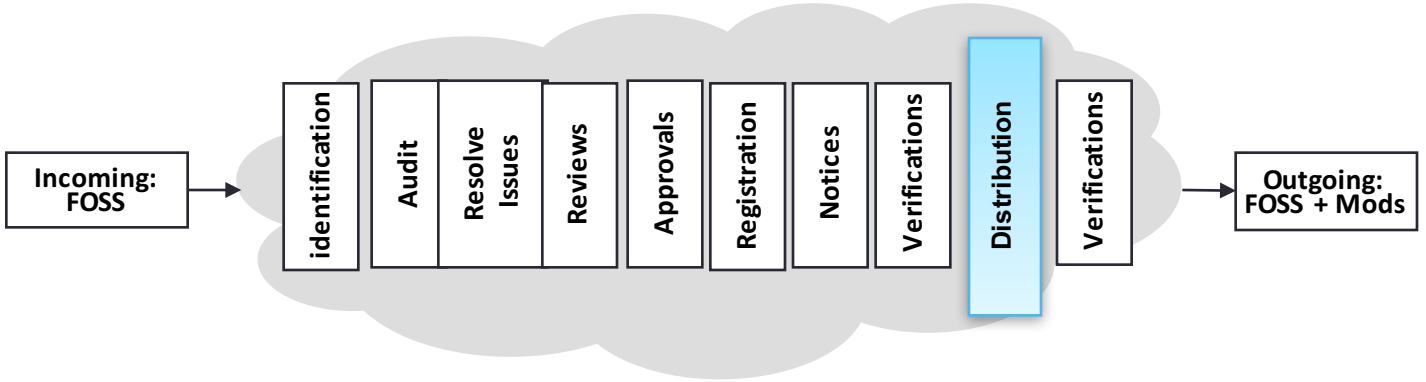
Pre-Distribution Verifications



배포된 Software가 Review 및 승인되었는지 확인한다

- 전제조건:
  - FOSS Component가 사용 승인 되었음
  - FOSS Component가 Release되는 Software의 목록표에 등록되었음
  - 적절한 고지문이 준비되었음
- 단계:
  - 배포할 FOSS Package들이 식별되고 승인되었는지 확인한다
  - 검토된 Source Code가 제품 내의 Binary와 일치하는지 확인한다
  - 식별된 FOSS에 대해 Source Code 요청 권리가 있음을 최종 소비자에게 알리기 위한 적절한 고지문이 포함되었는지 확인한다.
- 결과:
  - 배포 Package에는 검토 승인된 Software만 포함한다
  - 적절한 고지 File을 포함한 "배포 준수 산출물 (Distributed Compliance Artifacts : OpenChain 스펙에 정의)"을 배포 Package 혹은 다른 형태로 전달될 수 있도록 포함한다

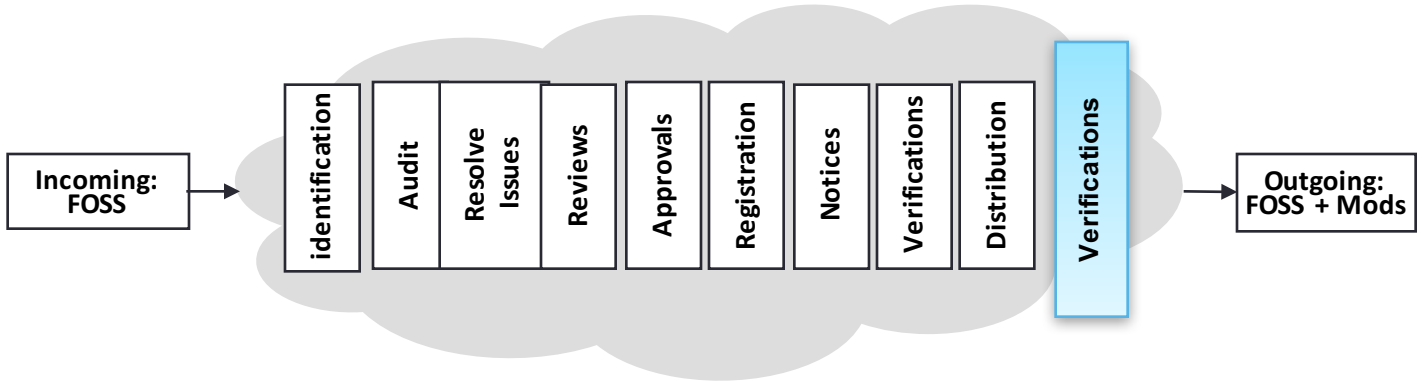
동반하는 Source Code Distribution



요구에 따라 동반하는 Source Code 제공

- 전제조건:
  - 배포 전 모든 사항이 확인 완료되었으며, 이슈는 발견되지 않음
- 단계:
  - 동반하는 Source Code를 관련 Build Tool 및 문서와 함께 제공한다 (예: 배포 사이트에 upload하거나 배포 Package내에 포함시키는 방법으로 제공)
  - 동반하는 Source Code는 제품 및 Version을 라벨(label)로 표시하여 식별되게 한다
- 결과:
  - 동반하는 Source Code를 제공하기 위한 의무 사항을 충족한다

Final Verifications



License 의무사항 준수 여부를 인증한다

- 전제조건:
  - 동반하는 Source Code를 필요에 따라 제공한다
  - 적절한 고지문을 준비 하였다
- 단계:
  - 동반하는 Source Code가 (필요한 경우) 올바르게 upload되거나 배포되었는지 확인한다
  - Upload 혹은 배포된 Source Code가 승인된 version과 동일한지 확인한다
  - 고지문이 적절하게 게시 되었는지와 접근 가능한지 확인한다
  - 이외 다른 의무 사항들이 준수되었는지 확인한다
- 결과:
  - 배포되는 Compliance 산출물들이 적절히 제공되었는지 확인한다

Check Your Understanding

OPENCHAIN

- Compliance를 위한 기업 실사(Due Diligence)를 위해 필요한 것은 무엇인가? (high level에서의 단계를 설명하라)
- Compliance Management의 일부로 해결해야 할 이슈 유형은 무엇인가?
- Audit 결과 검토에 누가 참여해야 하는가?
- Architecture 검토는 무엇을 확인하기 위해서인가?
- FOSS 고지문에는 무엇이 포함되어야 하는가?
- Copyleft License하의 Code는 무엇이 배포되어야 하는가?

CHAPTER 7

Compliance 위험 회피 방법

Compliance Pitfalls

OPENCHAIN

이 장에서는 Compliance Process에서 피해야 할 몇 가지 잠재적인 위험에 대해 설명한다 :

1. 지적 재산 (IP) 관련 위험
2. License Compliance 관련 위험
3. Compliance Process 관련 위험

지적 재산 관련 위험

유형 및 내용	발견 방법	회피 방법
Proprietary 혹은 3rd Party Code내에 의도치 않은 Copyleft FOSS의 포함: 이러한 유형의 오류는 개발 과정에서 개발자가 FOSS 정책과 맞지 않는 형태로 FOSS Code를 Proprietary Source Code에 추가(혹은 잘라서 붙여넣기)할 때 발생한다.	이러한 유형의 오류는 다음과 일치할 가능성이 있는지 Source Code의 Scanning 혹은 Auditing을 하여 발견할 수 있다: • FOSS Source Code • 저작권 고지 이 때, 자동화 Source Code Scanning Tool을 사용할 수도 있다	이러한 유형의 오류는 다음과 같은 방법으로 피할 수 있다: • 개발자들이 Compliance 이슈, 다양한 유형/종류의 FOSS License, Proprietary Source Code내에 FOSS Source Code를 포함시키는 것의 결과에 대해 인식할 수 있도록 교육을 제공한다 • Build 환경의 모든 Source Code (Proprietary, 3rd Party, FOSS)에 대해 정기적인 Source Code Scan 및 Audit을 수행한다

지적 재산 관련 위험

유형 및 내용	발견 방법	회피 방법
의도치 않게 Copyleft FOSS를 Proprietary Source Code에 Linking하는 경우 (혹은 그 반대 경우): 이런 유형의 오류는 충돌 혹은 호환되지 않는 License의 Software(FOSS, Proprietary, 3rd Party)를 서로 Linking하는 결과로 발생한다. Linking에 대한 법률적 효과는 FOSS Community에서 논쟁의 대상이다	이런 유형의 오류는 Dependency Tracking Tool(Software Component들 간의 연결관계를 보여주는 Tool)을 이용해 발견할 수 있다	이런 유형의 오류는 다음과 같은 방법으로 피할 수 있다: 1. 개발자들이 FOSS 정책과 맞지 않는 형태로 Software Component를 linking하는 것을 피할 수 있도록 교육을 제공한다 2. Build 환경에서 Dependency Tracking Tool을 지속적으로 실행한다
Source Code 수정을 통해 Proprietary Code를 Copyleft FOSS에 포함	이런 유형의 오류는 Audit 혹은 Scan을 통해 FOSS Component에 추가한 Source Code를 식별 및 분석하여 발견할 수 있다	이런 유형의 오류는 다음의 방법을 통해 피할 수 있다: 1. 개발자에게 교육 제공 2. 주기적인 Code Audit 수행

License Compliance 관련 위험

유형 및 내용	회피 방법
동반하는 Source Code 제공 실패	이런 유형의 오류는 Source Code 공개를 제품 출시를 위한 Checklist 항목으로 관리함으로 피할 수 있다
잘못된 Version으로 동반하는 Source Code 제공	Binary version과 일치하는 Source Code 공개를 보장하기 위한 확인 단계를 Compliance Process에 추가함으로 이런 유형의 오류를 피할 수 있다
수정한 FOSS Component에 대해 동반하는 Source Code 제공 실패	FOSS Component의 원본 Source Code 뿐만 아니라 수정한 Source Code에 대해서도 공개를 보장하기 위한 확인 단계를 Compliance Process에 추가함으로 이런 유형의 오류를 피할 수 있다

License Compliance 관련 위험

유형 및 내용	Avoidance
FOSS Source Code에 수정 내용 표시 실패: FOSS Source Code에 수정 내용을 표시하지 않거나 수정사항에 대한 설명을 포함하지 않음	이런 유형의 오류는 다음의 방법을 통해 피할 수 있다: 1. Source Code 수정 내용 표시 과정을 Source Code 공개 전 확인 단계로 추가 2. 개발자들에게 일반 대중에게 공개할 예정인 모든 FOSS 혹은 Proprietary Software에 대해 Copyright 표시 및 License 정보 update를 수행할 수 있도록 교육 제공

Compliance Process 관련 위험

내용	회피 방법	예방 방법
개발자가 FOSS 사용 승인을 받지 않음	이런 유형의 오류는 개발자에게 회사의 FOSS 정책 및 Process에 대해 교육을 제공함으로 피할 수 있다	이런 유형의 오류는 다음 방법을 통해 예방할 수 있다: 1. Software Platform에 대해 정기적으로 Full Scan을 수행하여 확인되지 않은 FOSS의 사용을 발견한다 2. 개발자에게 회사의 FOSS 정책 및 Process에 대해 교육을 제공한다 3. 직원 성과 평가 시 Compliance 항목을 추가한다
FOSS 교육을 수강하지 않음	FOSS 교육 수강을 직원의 전문 개발 계획의 일부임을 보장하고 직원 성과의 일부로 수강 여부를 감독함으로 이런 유형의 오류를 피할 수 있다	이런 유형의 오류는 개발자들에게 특정 날짜까지 FOSS 교육을 수강하도록 요구함으로써 예방할 수 있다

Compliance Process 관련 위험

내용	회피 방법	예방 방법
Source Code Audit 미수행	이런 유형의 오류는 다음의 방법을 통해 피할 수 있다: 1. 정기적인 Source Code Scan/Audit 수행 2. Audit을 반복적인 개발 Process내 하나의 주요한 단계로 보장	이런 유형의 오류는 다음의 방법으로 예방할 수 있다: 1. 예정보다 늦어지지 않도록 적절한 인원 제공 2. 정기적인 Audit 실시
Audit을 통해 발견한 결과(Scan Tool 혹은 Audit에 의해 보고된 "hits"의 분석에 의한 결과)를 해결하지 않음	Audit 보고서가 완료되지 않은 경우, 문제가 해결되었다는 Compliance 증명서를 발급하지 않음으로 이러한 유형의 오류를 피할 수 있다	이런 유형의 오류는 FOSS Compliance Process에 승인 절차를 추가함으로 예방할 수 있다
FOSS에 대한 검토를 적절한 시기에 진행하지 않음	개발자가 FOSS Source Code의 채택을 아직 결정하지 않았더라도 조기에 FOSS Review 요청을 시작함으로써 이런 유형의 오류를 피할 수 있다	이런 유형의 오류는 교육을 통해 예방할 수 있다

제품 출하 전 Compliance의 보장

OPENCHAIN

- 기업은 (어떤 형태로든) 제품 출하 이전에 Compliance를 우선시하여야 한다
- Compliance를 우선시하면 다음 사항이 개선된다 :
  - 조직 내에서 보다 효과적으로 FOSS를 사용함
  - FOSS Community 및 FOSS 기관들과의 관계 개선

Community와의 관계 확립

OPENCHAIN

상용 제품에 FOSS를 사용하는 기업이라면 FOSS Community (특히, 제품에 사용 및 배포하는 FOSS Project와 관계된 FOSS Community)와 좋은 관계를 유지하는 것이 대단히 유리하다

게다가, FOSS 기관과의 좋은 관계는 Compliance를 위한 최선의 방안에 대해 도움을 받을 수 있고, Compliance 이슈가 발생할 경우에도 도움이 된다

또한, Software Community와의 좋은 관계는 양방향 Communication에 도움이 될 수 있다 : Upstreaming 개선 및 Software 개발자로부터의 지원을 받을 수 있다

Check Your Understanding

- FOSS Compliance 관련 어떤 유형의 위험이 있을 수 있나?
- 지적 재산 관련 실패 사례를 제시하시오.
- License Compliance 관련 실패 사례를 제시하시오.
- Compliance Process 관련 실패 사례를 제시하시오.
- Compliance에 우선순위를 두었을 때의 장점은 무엇인가?
- Community와 좋은 관계를 유지하는 것의 장점은 무엇인가?