OPENCHAIN

课程

自由开源软件训练课程简报–搭配 OpenChain 规范书 1.1 版

本简报依美国法律进行说明。不同的司法管辖区域可能会有不同的法律要求。
此点在使用简报作为合规训练项目的一部分时，应被考量。

本简报内容并未包含法律建议

OpenChain 课程是什麼？

- OpenChain项目，协助确认及分享自由开源软件合规专案的核心构成要素。
- OpenChain项目的核心为其**规范书**。其确认并发布一个自由开源软件合规专案，所应满足的核心要项。
- 本OpenChain**课程**，透过提供自由可取得的训练素材，来支持规范书。
- 本简报协助商业公司满足规范书 1.2 项的要求。其亦可被用於一般合规训练上。

取得更多信息：https://www.openchainproject.org

内容

1. 什麼是知识财产？
2. 介绍自由开源软件许可
3. 介绍自由开源软件合规
4. 自由开源软件审核的关键软件观念
5. 进行自由开源软件审核
6. 端对端的合规管理（流程范例）
7. 避开合规陷阱
8. 开发者准则

自由开源软件政策

- <<此待补充的空白项，用以指示何处可以找到自由开源软件政策书(依OpenChain规范书 1.1 版第1.1.1项要求)>>

- 你可透过Linux Foundation开源合规专案取得一份自由开源软件政策书的范本 https://www.linux.com/publications/generic-foss-policy

章节一

什麼是知识财产？

什麼是「知识财产」？

- 著作权(版权)：保护著作具创作性之原始作品
  - 保护表达(不及於其後之思想)
  - 涵盖软件、书籍，及其他相类作品
- 专利：具新颖性及非属显著已知状态的实用发明
  - 抑制垄断以鼓励创新
- 营业秘密：保护具价值的保密信息
- 商标：保护用来辨识产品来源的标章（文字、图形记号、标语、颜色，等等。）
  - 保护消费者及品牌；避免消费者混淆及品牌淡化

本章节将聚焦在著作权及专利，

该领域和自由开源软件合规最为相关。

软件中的著作权(版权)概念

- 基本规则：著作权保护具创作性的作品
- 著作权一般适用於文学作品，例如书籍、电影、图片、音乐、地图
- 软件受到著作权保护
  - 软件被著作权保护的部分并非功能（这部份是被专利保护的）而是表达（实作细节中的创作性）
  - 包括二进位代码及源代码皆受到保护
- 著作权利人只对他/她创作的作品有控制地位，不及於他人的独立作品。
- 未经作者同意的复制可能导致侵权行为

著作权(版权)中与软件最相关的权利态样

- 复制软件的权利 - 制作复制件
- 创作「改编作品」的权利 - 进行修改
  - 此处改编作品一词引自美国著作权法
  - 其为「专用名词」，意指依法令的特定涵义来解释，而非依一般字典定义。
  - 一般来说，其指的是基於一个原作品来创作的新作品，过程中有足够的创作性加入，而让新作品表现为另一具创作性的作品，而非仅为原作品的复制件。
- 发行的权利
  - 发行一般被视为，在二进位代码或是源代码的形式下，提供软件一部分的复制件给其他的实体。（在你公司或是组织外的个人或是组织）

注意：对何者构成「改编作品」或「发行」的解释，在自由开源软件社区与自由开源软件法律圈中仍有争议

软件中的专利概念

- 专利保护功能性 – 这可以包含操作的方法，例如计算机程序
  - 不保护抽象思想、自然法则
- 为於特定地区取得专利，必须在该特定的司法管辖领域提出专利申请。倘该专利被核可，专利拥有者有权利可以阻止他人实施该专利的功能，不论该功能是否为独立创作。
- 其他希望利用该项技术者，或会洽询专利许可（该许可可能授与使用、制造、使制造、销售，及输入及输出该技术的权利）
- 即使其他人独立地创作相同的发明，仍可能导致侵权行为

许可

- 「许可」是著作权或专利拥有者，授与同意或权利给其他人的方法
- 许可得被限定在：
  - 被允许的使用类型（商业性/非商业性、发布、改编作品/制造、使制造、生产）
  - 专有或非专有的条件
  - 地理范围
  - 永久的或限定期间
- 许可得是附条件的授与，意指仅有在遵循某些义务性要求才会获得许可
  - 例如，提供署名，或给予互惠性许可
- 可能包含将保证、赔偿、支援、升级、维护相关的合同条件

检测你的了解程度

OPENCHAIN

- 著作权法保护何种素材？
- 对软件最重要的著作权权利是什麼？
- 软件能够是专利的标的吗？
- 专利给予专利拥有者什麼权利？
- 如果你独立开发了你的软件，你有可能会需要第三方的著作权许可或是专利许可吗？

章节二

介绍自由开源软件许可

自由开源软件许可

- 自由开源软件许可依定义来说，是让源代码能被容许修改及再次发布的协议
- 自由开源软件许可，可能会附随署名、保留著作权声明，或是提供书面源代码索取文件有关的条件
- 一组较普及的许可证是由开放源代码促进会 (OSI)依据其开放源代码定义 (OSD)核准的列表。完整的OSI核准许可证列表可参照右列键结 http://www.opensource.org/licenses/

宽松式的自由开源软件许可

- 宽松式的自由开源软件许可 – 此一词汇用来描绘最低限制程度的自由开源软件许可证
- 例如：BSD-3-Clause
  - 3款BSD许可证是宽松式自由开源软件许可的一则著例，其允许源代码或目标代码形式不受限制，基於任何目的之再行发行，只要其著作权声明以及许可证里的免责声明有被维持
  - 该许可证当中有一个条款，限制了在改编作品中，若未得到特别许可的话，不得使用贡献者的名字背书
- 其他例子：MIT、Apache-2.0

许可互惠性& Copyleft 许可证

- 某些许可证要求当改编作品（或相同文档的软件、同一个软件程序，或依其他界限划分的范围）被发行时，必须以原作品相同的许可证进行散布
- 此被称为 “copyleft” 或、「许可互惠性」的效应
- 以GPL-2.0的许可互惠性为例：

  你必须让任何你散布或发行的作品，其全部或一部含有 GPL-2.0 原生程序，或为 GPL-2.0 程序之改编，或前述原生改编的任何一部分，采，…，本许可之条款来迈行散布或发行。

- 带有互惠性或称Copyleft条文的许可证，包括所有版本的GPL、LGPL、AGPL、MPL，以及CDDL。

私有软件或闭源软件

- 私有软件许可证（或称商业许可、或称终端使用者许可协议），对软件的使用、修改，及／或发行具有限制性条件
- 私有许可证对每一个提供者都是独特的 – 有几个提供者就有几种私有许可证的变异，故每个私有许可证都应该被个别进行评估
- 即使自由开源软件及私有许可证都是基於知识财产，以给予该财产的授权许可，然自由开源软件的开发者常使用「私有 (proprietary)」这个字词，来形容商业性的非自由开源软件许可。

其他非自由开源软件的许可证情境

- 免费软件(Freeware)－采私有许可证以免费或非常低价进行发行的软件
  - 源代码可能或不能提供，创作改编作品通常是被限制的
  - 免费软件通常会提供完整功能（没有被上锁的特别功能），且可能永久使用（不受使用天数限制）
  - 免费软件许可证通常会对复制、发行和制作改编作品，以及使用目的（个人、商业用、学术用，...等）施加限制。
- 共享软件(Shareware)－就试用基础提供给使用者的私有软件，限定期间、免费但限制功能或限制特别功能
  - 共享软件的目的是提供潜在购买者使用此软件的机会，并在付费购买许可或完整版前先评估其实用性
  - 多数公司夺共享软件持怀疑态度，因为共享软件供应商，常会在共享软件免费流传在组织内部後，向这些公司要求索取高额的许可证费用。

其他非自由开源软件的许可证情境

- 「非商业性」–某些许可证具有多数自由开源软件许可证的特性，但却限制仅供非商业使用 (例如，CC 署名-非商业性 许可协议 / CC BY-NC)
  - 自由开源软件依定义，便不能限制软件的使用范畴
  - 商业使用即为一种使用范畴，故对商业的任何限制即阻却该许可成为自由开源软件许可证

公共领域

- **公共领域**一词是指不被法律保护的软件，因此公众不需要取得许可即可使用
- 开发者或会在他们的软件附上公共领域宣告
  - 例如，「在此软件中的所有代码及文件，已被作者贡献至公共领域。」
  - 公共领域宣告不等於自由开源软件许可证
- 公共领域宣告让开发者得尝试放弃或消除软件中，任何或有的知识产权，以明示其可不受任何限制的被使用，然宣告的可执行性於自由开源软件社区里仍有争议，且其法律上的有效性亦因司法管辖区域的不同而有所差异
- 公共领域宣告常会附带其他条文，例如免责条款；在这种情形，此软件通常会被视为依许可证提供，而非处於公共领域

许可相容性

- 许可相容性是确保许可证不冲突的程序
- 若有一个许可证要求你做一件事，但另一个许可证却禁止你做那件事，而倘若将两个软件模组合并将导致其结合须置於单一许可证的义务，那这两个许可证就是互相冲突且不相容的。
  - GPL-2.0 及 EPL-1.0 皆将其义务性规定延伸至被发行的「改编作品」
  - 若有一GPL-2.0 模组与一个EPL-1.0 模组被结合在一起，此合并的模组也被发行了，那麼该模组必须：
    - (依照 GPL-2.0) 仅依 GPL-2.0 来被发行，并且
    - (依照 EPL-1.0) 仅依 EPL-1.0 来被发行。
    - 发行者无法同时满足上列两个条件，故此模组也许不能被发行。
    - 此为一个*许可不相容(license incompatibility)*的例子。

「改编作品」的定义在自由开源软件社区中有不同看法，且其法律释义亦随司法管辖区域不同而有可能变化。

声明

声明，例如文档档头上的注解文字，通常会提供作者及许可证信息。自由开源软件许可证也可能会要求在源代码或文件里，或并随源代码或文件放置声明，以表彰作者 (署名)，或清楚指示该软件包括修改部件。

- **著作权声明 –** 置於作品复制件里，告诉世界其著作权归属状态的标示。例如： Copyright © A. Person (2016)
- **许可证声明 –** 说明及显示产品中自由开源软件的许可证与条件的声明。
- **署名声明 –** 於产品释出时，显示产品中自由开源软件的原始作者及/或其赞助者的声明。
- **修改声明 –** 指出你夺源代码里哪一个文档已作修改的声明，例如在文档最上方加入你的著作权声明即属之。

多重许可证

OPENCHAIN

- 多重许可证指的是，将软件发布同时置於二组或更多不同的许可证与条件下进行实作。
  - 例如，若软件是采「双重许可证」，则著作权利人是将二组许可证的选择权交给每一个软件的收受者
- 注意：此不应与授权人要求你必须遵从多於一组的所有许可证之状况混淆

检测你的了解程度

- 什麼是自由开源软件许可证？
- 宽松式的自由开源软件许可证，典型的义务性要求有哪些？
- 试列出一些宽松式的自由开源软件许可证。
- 许可互惠性意指什麼？
- 试列出一些 copyleft 类型的许可证。
- 使用依 copyleft 许可证授权的代码时，什麼是需要一并被发布的？
- 免费软件及共享软件是否会被视为自由开源软件？
- 什麼是多重许可证？
- 在自由开源软件的声明里你可能找到什麼信息，以及这些声明能被如何利用？

章节三

介绍自由开源软件合规

自由开源软件合规的目标

OPENCHAIN

- **了解对你的义务性要求。** 你应有一套能辨识及追踪，你的软件现存哪些自由开源软件组件之流程
- **满足许可证的义务性规定。**你的流程应要能够处理，因你组织的商业实作而带来的自由开源软件许可义务性规定。

哪些合规义务性规定必须被满足？

依据使用到的自由开源软件许可证，你的合规义务性规定或许包括：

- **署名与声明。**你也许需要提供或保留著作权声明及许可证文字到源代码，及/或产品的文件，或使用者操作介面里，好让下游使用者得知软件的来源，及在该许可证下赋予他们的权利。你也许需要提供将修改纪录有关的声明，或许可证文件的完整复制件。
- **源代码的提供。**你也许需要提供该自由开源软件本身、你所作的修改、供结合或键结的软件，以及控制建制流程的脚本之程序源代码。
- **互惠性。**你也许需要采与管理该自由开源软件组件完全相同的许可证，来维护其修改版本或改编作品。
- **其他条款。**该自由开源软件许可证或会限制其著作权利人姓名或商标之使用，也许会要求修改版本使用不同的名称来避免混淆，或在违反此要求时终止许可。

自由开源软件合规争议：发行

- 对外部组织散播素材
  - 应用程序被下载到使用者的机器或行动装置
  - JavaScript、网络服务客户端，或其他程序代码被下载到使用者的机器
- 对於某些自由开源软件许可证来说，透过网络存取可视为触发点。
  - 某些许可证对触发点的定义，包含对在伺服器上运行的软件提供存取 (例如：若该软件被修改过的话 – 所有 Affero GPL 版本皆作如此定义)，或是「使用者透过计算机网络远端与其互动」这种情境

自由开源软件合规争议：修改

- 对於当前既存的程序进行变动（例如：增加、删除文档里的程序代码，将组件结合在一起）
- 依某些自由开源许可证，修改也许会在发行时带来额外的义务性要求，例如：
  - 提供修改声明
  - 提供伴随的程序源代码
  - 依管理自由开源软件组件的同份许可证来授权该修改

自由开源软件合规专案

已於自由开源软件合规上取得成功的组织，会建立他们自己的自由开源软件合规专案 (包含政策、流程、训练，及和工具)，来达到下列目的：

1. 便利自由开源软件於其产品 (商业性或其他)里的采用效率
2. 尊重自由开源软件开发者/权利人的权利，及遵守其许可义务性规定
3. 贡献并参与自由开源软件社区

导入合规实作

准备好企划流程及足够的人力资源来应对：

- 辨识所有内部及外部软件的出处及许可证
- 开发流程里追踪自由开源软件
- 进行自由开源软件审核及辨识其许可义务性规定
- 在产品发送时实现许可义务性规定
- 监管自由开源软件合规专案、建立政策，及合规决策
- 内部教育训练

合规的好处

健全的自由开源软件合规专案带来的好处包括：

- 对自由开源软件的好处及其如何对你的组织产生影响，增加认识
- 对使用自由开源软件的成本及风险，增加认识
- 对可用的自由开源软件方案，增加知识
- 减低及管理侵权风险、增加对自由开源软件开发者/权利人许可决策的尊重
- 与自由开源软件社区及组织培养良好关系

检测你的了解程度

- 自由开源软件合规意指什麼？
- 自由开源软件合规专案的两个主要目标是什麼？
- 条列并说明自由开源软件合规专案的重要商业实作
- 自由开源软件合规专案的好处为何？

章节四

自由开源软件审核的关键软件概念

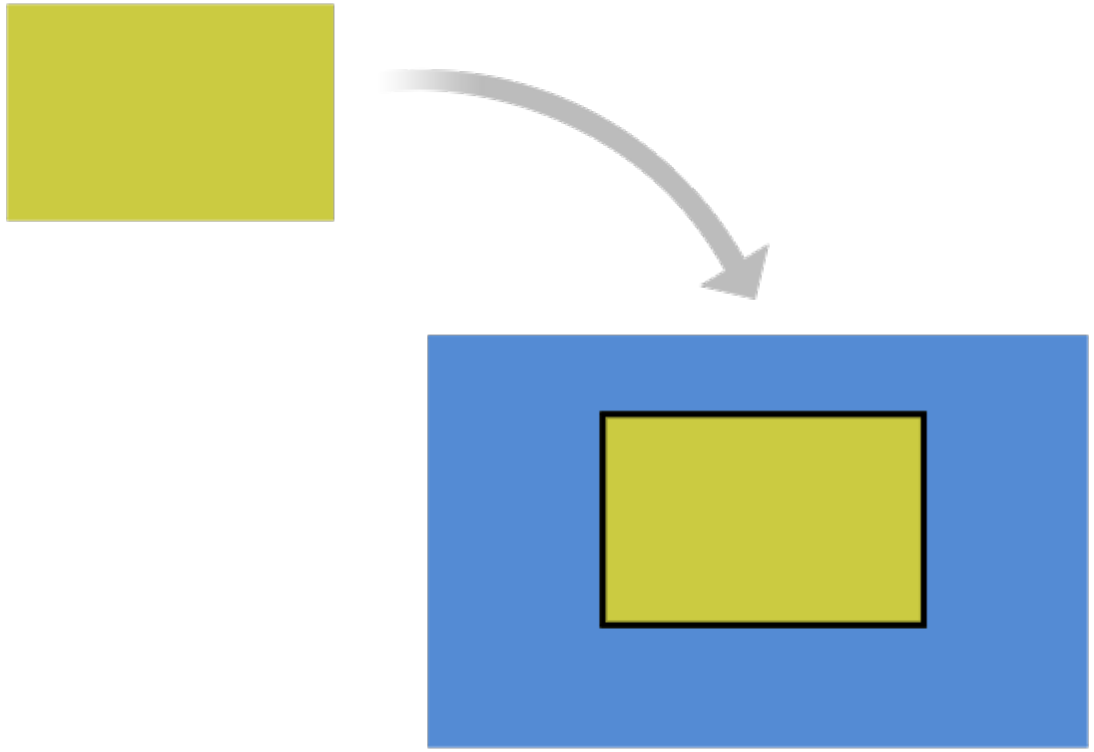
你想要如何使用自由开源软件组件？

常见的使用情境包括：

- 合并(Incorporation)
- 链结(Linking)
- 修改(Modification)
- 转变(Translation)

合并

开发人员可能会复制部分的自由开源软件，到你的软件产品之中。



相关的字词包括：

- 整合(Integrating)
- 融合(Merging)
- 贴上(Pasting)
- 改用(Adapting)
- 嵌入(Inserting)

键结

开发人员可能会链结或加入自由开源软件许可组件，与你的软件产品一起运作。

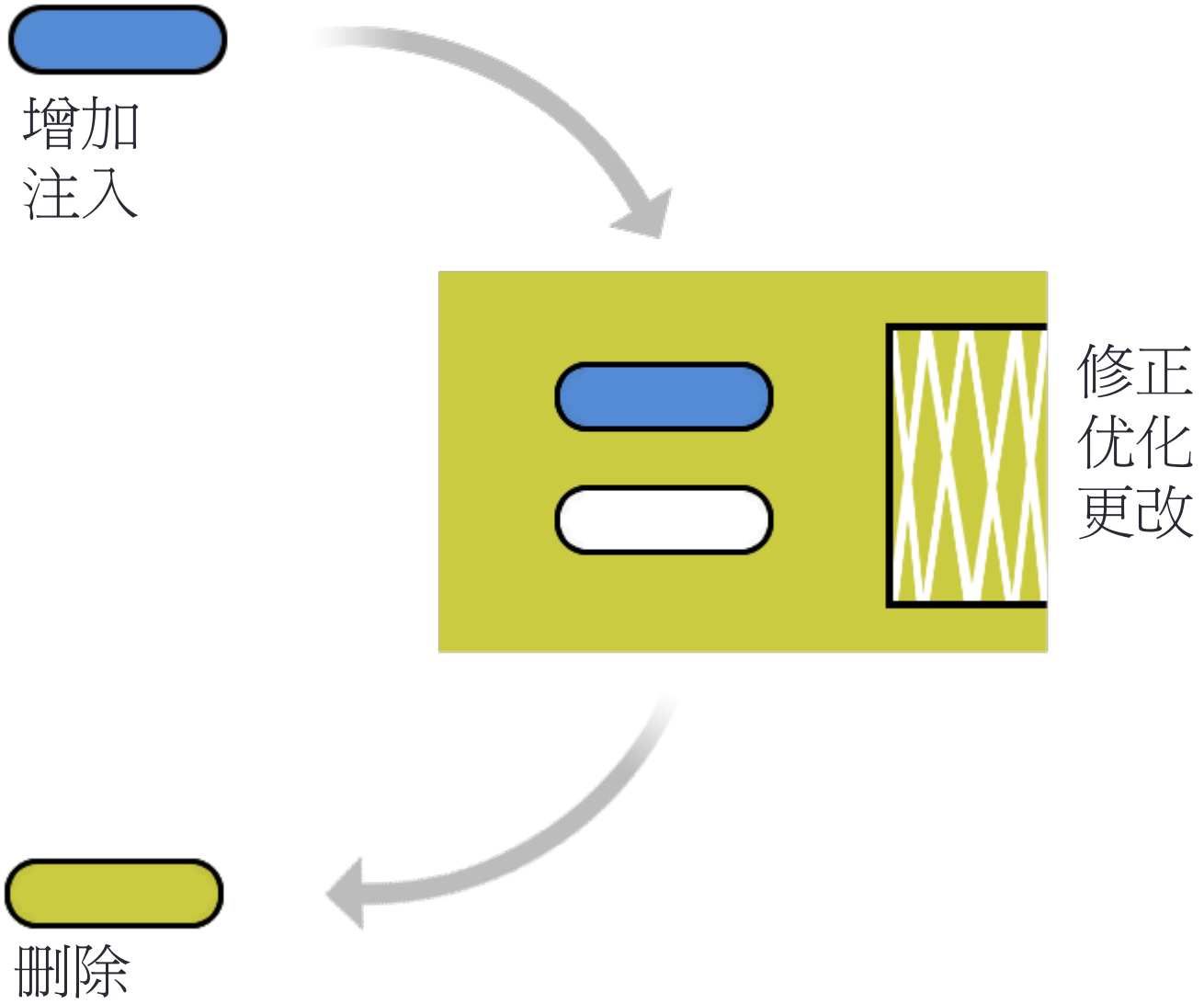
相关的字词包括：

- 静态/动态键结(Static/Dynamic Linking)
- 配对(Pairing)
- 结合(Combining)
- 利用(Utilizing)
- 打包(Packaging)
- 建立相依关系(Creating interdependency)

修改

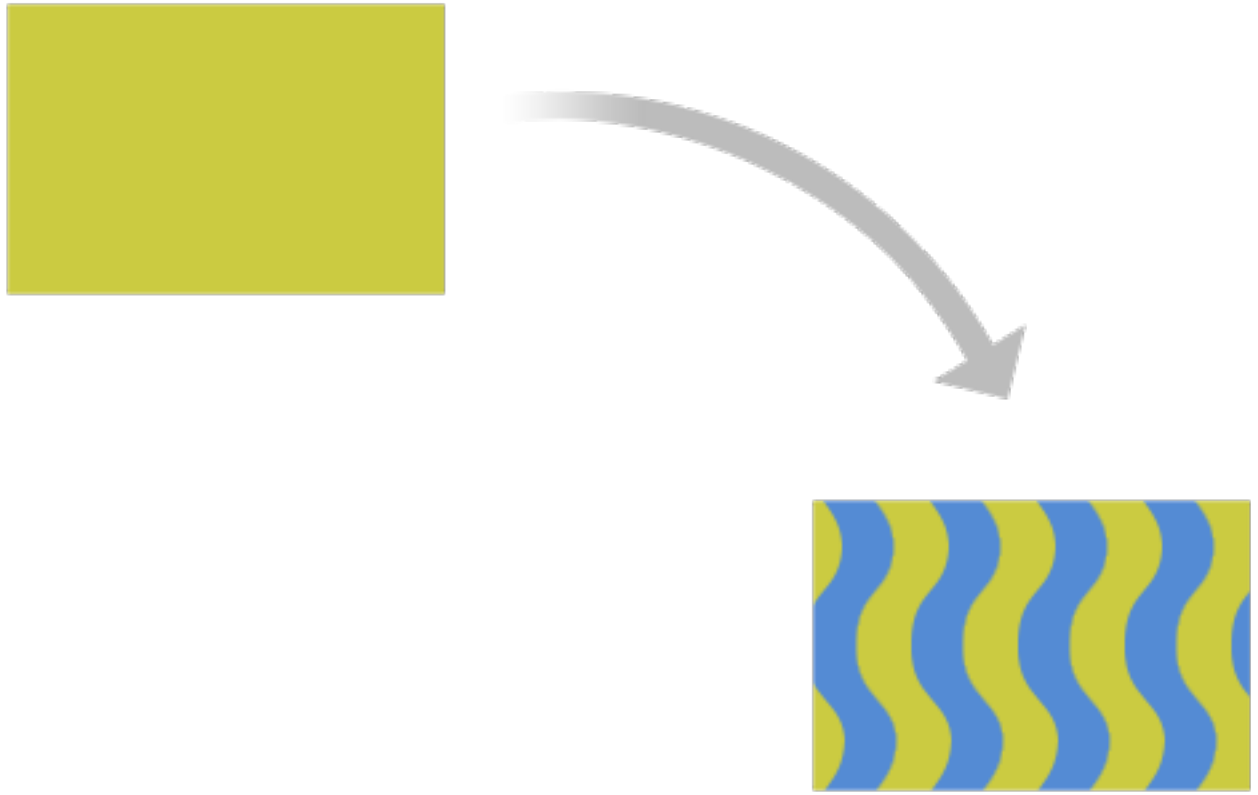
开发人员可能会对自由开源软件组件进行变动，包括：

- 增加/注入新的程序代码到自由开源软件组件里
- 对自由开源软件组件进行修正、优化，或更改
- 删除或移除程序代码



转变

开发者可能会转化程序代码的状态



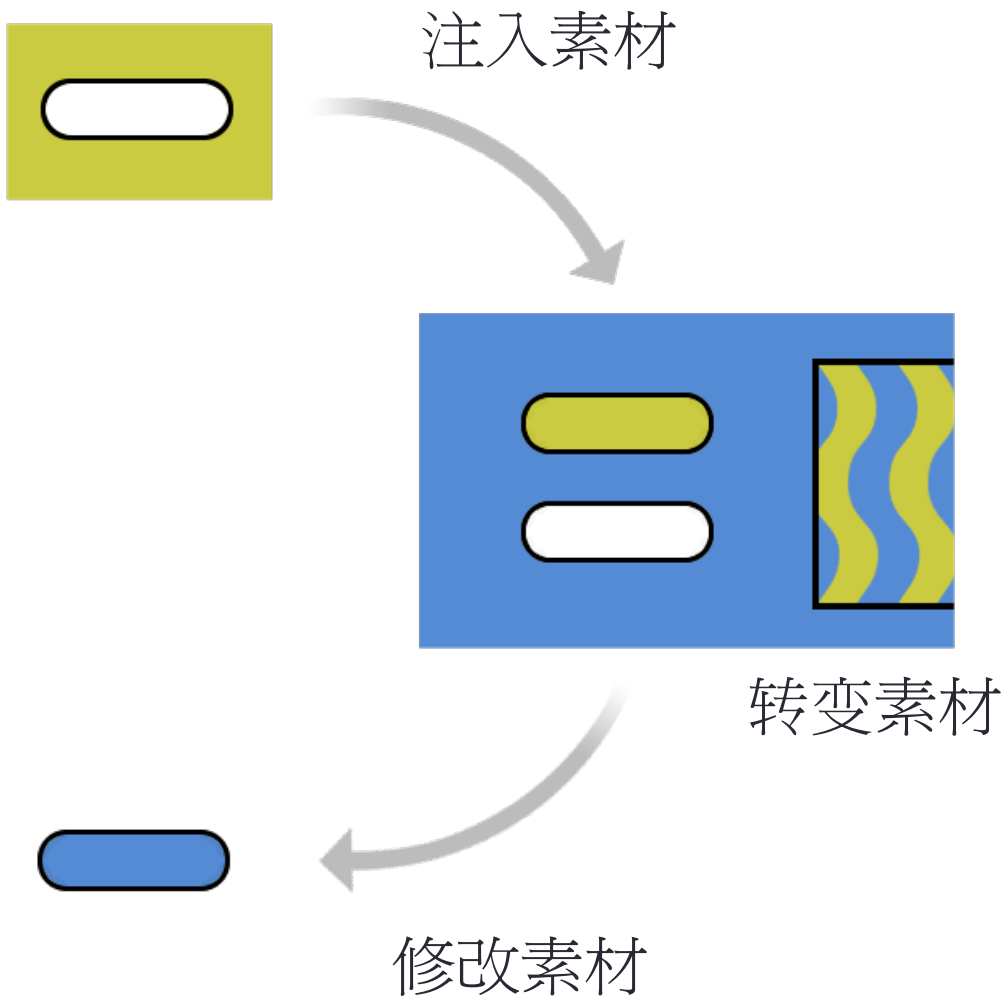
例子包括：

- 将中文翻译成英文
- 将C++转变为Java
- 编译成二进位代码

开发工具

开发工具可能会在幕後执行某些操作行为。

例如，开发工具可能会将其自身的部分程序代码注入至输出成果。



自由开源软件组件如何被发行？

- 谁会收受到这些软件？
  - 顾客/合作夥伴
  - 社区项目
  - 在商业团体范围内的另一个法人(这可能会被视为发行)
- 传递的形式是什麼？
  - 以程序源代码传递
  - 以二进位代码传递
  - 预载到硬体里

检测你的了解程度

- 合并(Incorporation)是什麼？
- 键结(Linking)是什麼？
- 修改(Modification)是什麼？
- 转变(Translation)是什麼？
- 评估发行的重要要素是什麼？

章节五

进行自由开源软件审核

自由开源软件审核

- 在专案及产品管理与工程师，已就推荐的自由开源软件组件进行可用性及品质的审核後，使用该选定组件牵涉到的权利与义务关系之审核，应被启动。
- 搜集相关信息
- *自由开源软件审核*流程，是自由开源软件合规专案的关键元素。透过此流程，公司得以分析其采用的自由开源软件，并理解其权利与义务关系。
- 自由开源软件审核流程，包含以下几个步骤：
  - 搜集相关信息
  - 分析并理解许可证的义务性规定
  - 提供与公司政策与商业目标相合的使用指导

启动自由开源软件审核

任何在公司里职司与自由开源软件有关的人，都应该能够启动自由开源软件审核，包括专案或产品管理人员、工程师，以及法务。

注意：此流程通常会在，基於自由开源软件的新软件被工程师或外部承包商选用时启动。

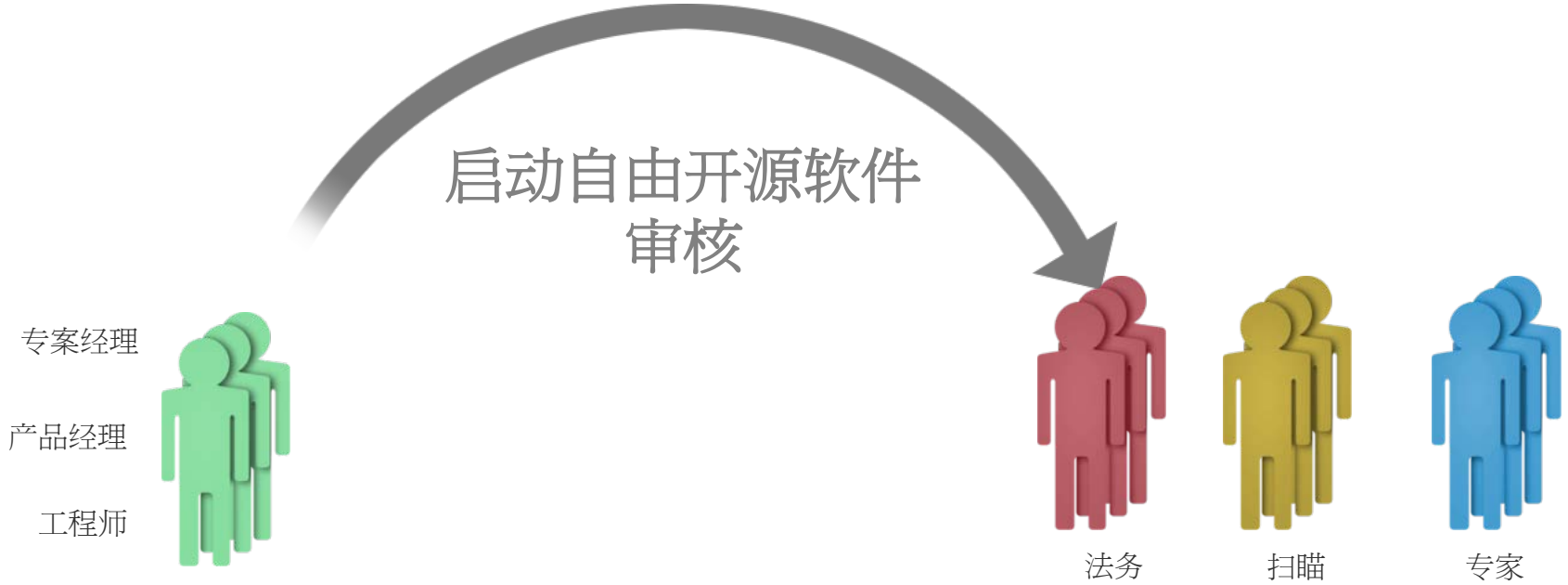
你需要搜集哪些信息？

在分析自由开源软件使用时，你需要搜集关於自由开源软件组件辨识信息，它的来源，及能被如何使用。这些信息可能包括：

- 套件名称(Package name)
- 与套件相关的社区状态(活动、各种成员状况、回应程度)
- 版本(Version)
- 下载或源代码网址(URL)
- 著作权利人
- 许可证及其许可证网址
- 署名及其他声明和其网址
- 对於修改部分的描述

- 相依性列表(List of dependencies)
- 在产品中的用途
- 第一个包含此套件的发行产品
- 程序源代码被维护的位置
- 在之前的其他脉络是否已经被同意使用
- 是否是由外部承包商处取得
- 开发团队的接触点
- 著作权声明、署名，供应商修改部分的程序源代码(若许可义务性要求须被满足的话)

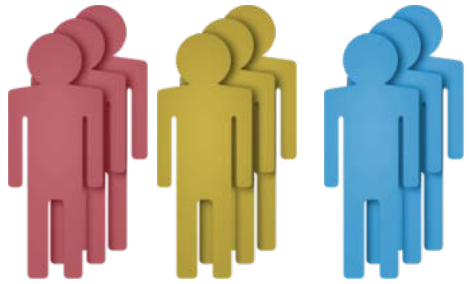
自由开源软件审核团队



自由开源软件审核团队，包括支援、指导、协调及审核自由开源软件使用的公司代表们。这些代表或会包含：

- 辨识与评估许可义务性规定的法务小组
- 支援源代码扫描及工具辅助，以协助辨识与追踪自由开源软件使用的扫描小组
- 与企业利益、商业许可证、出口规范等等部门共工，而可能会被自由开源软件使用影潜到的工程专家群

分析自由开源软件的使用提议



自由开源软件审核团队，在提供议题指导之前，应先评估已搜集信息。这可能包括扫描程序源代码，以确认信息的正确性。

自由开源软件审核团队应考量：

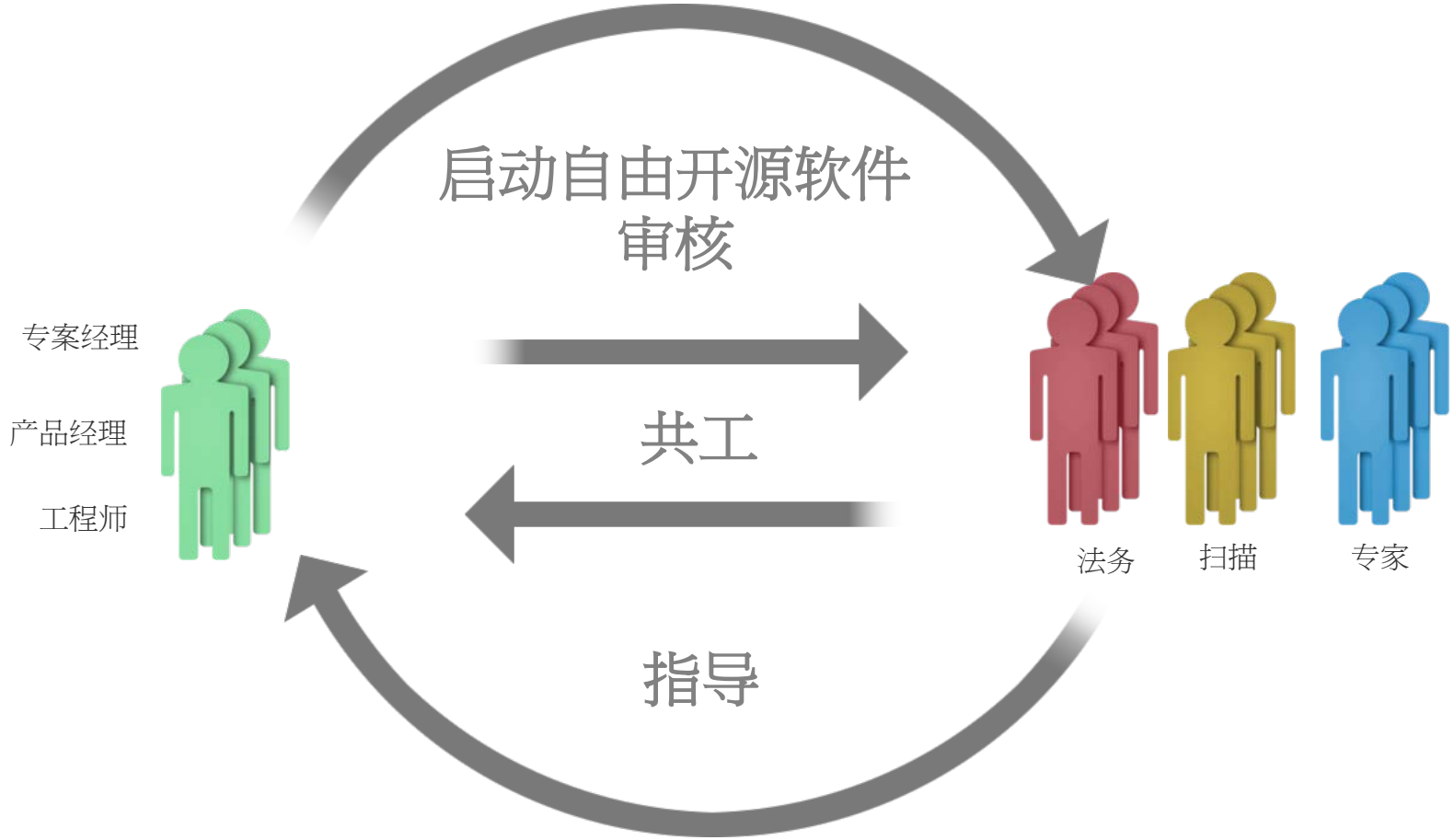
- 程序代码及其相关信息是否完整、一致，并且准确？
- 声称的许可证是否与程序代码文档里显示的一致？
- 该许可证是否容许与软件里的其他组件一起使用？

程序源代码扫描工具

- 有许多不同的自动化开放源代码扫描工具。
- 这些方案皆呼应到特定的需求 – 也因为这个原因 – 并无单一方案能解决所有可能的挑战
- 公司选择与他们特定市场领域与产品最相合的方案
- 许多公司兼采自动化工具及人工审核
- 自由供取用的开放源码扫描工具 FOSSology 是一个优良范例，这个项目是由 Linux Foundation 所主持：

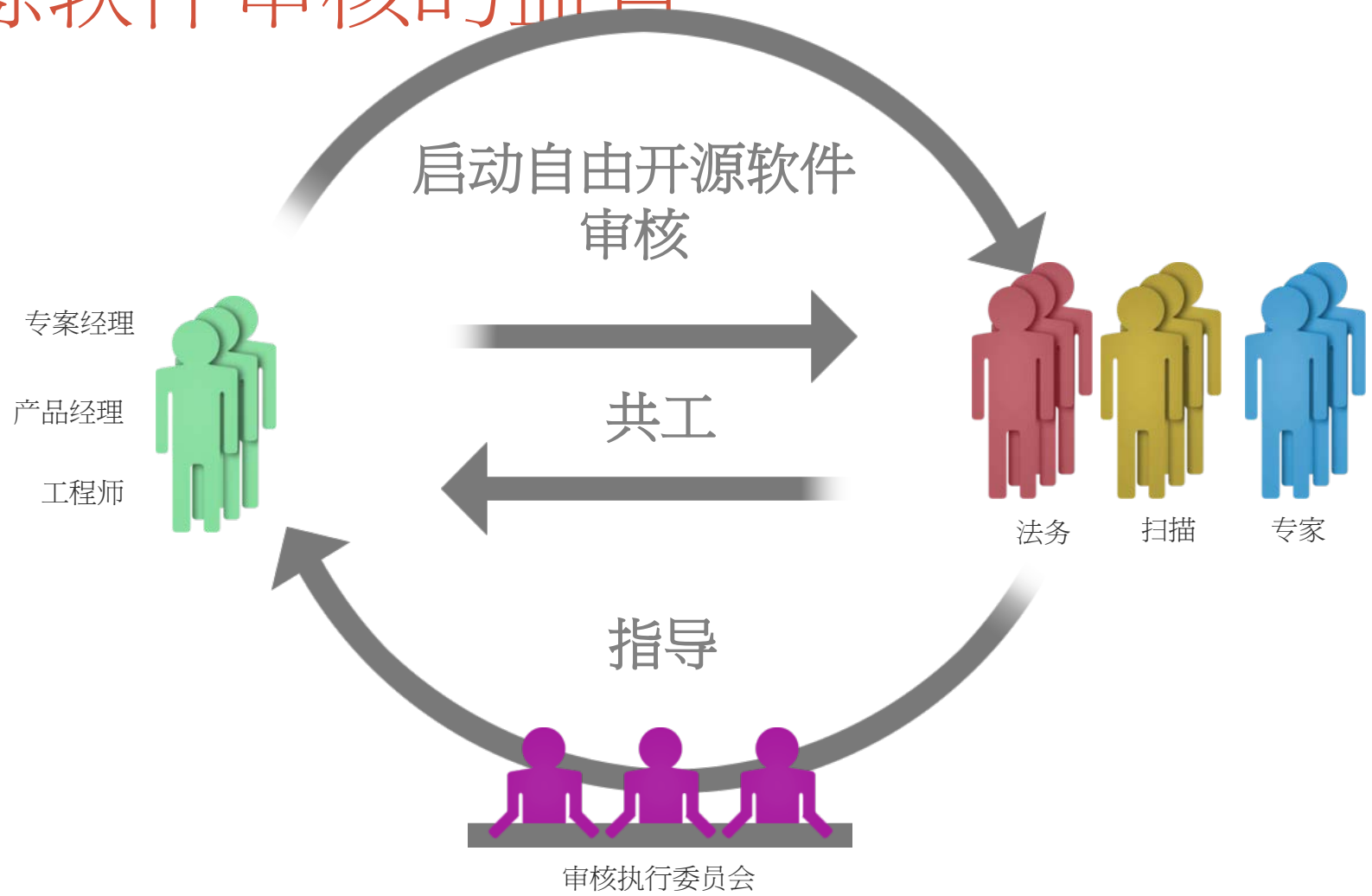
  https://www.fossology.org

透过自由开源软件审核进行共工



自由开源软件审核流程跨越学科领域，包括工程、商务，以及法务团队。它必须保持互动性，以确保所有那些团队皆对议题有正确理解，并能建立明确的共享性指导文件。

自由开源软件审核的监督



自由开源软件审核的流程，应该要有执行性的监督，以解决歧见，并核可最重要的政策。

检测你的了解程度

OPENCHAIN

- 自由开源软件审核的目的为何？
- 若你要使用自由开源软件组件，第一个应采行的行动为何？
- 如果你对使用自由开源软件有疑问，应该怎麼做？
- 为了自由开源软件审核，你可能需要搜集哪些种类的信息？
- 什麼信息可以协助辨识软件是被谁许可的？
- 当自由开源软件组件是从外部承包商而来，哪些额外信息对於审核它是重要的？
- 在自由开源软件审核里，可以采行哪些步骤，来评估所搜集信息的品质？

章节六

端对端的合规管理（流程范例）

中小型公司查核清单的范例

OPENCHAIN

持续性的合规工作事项：

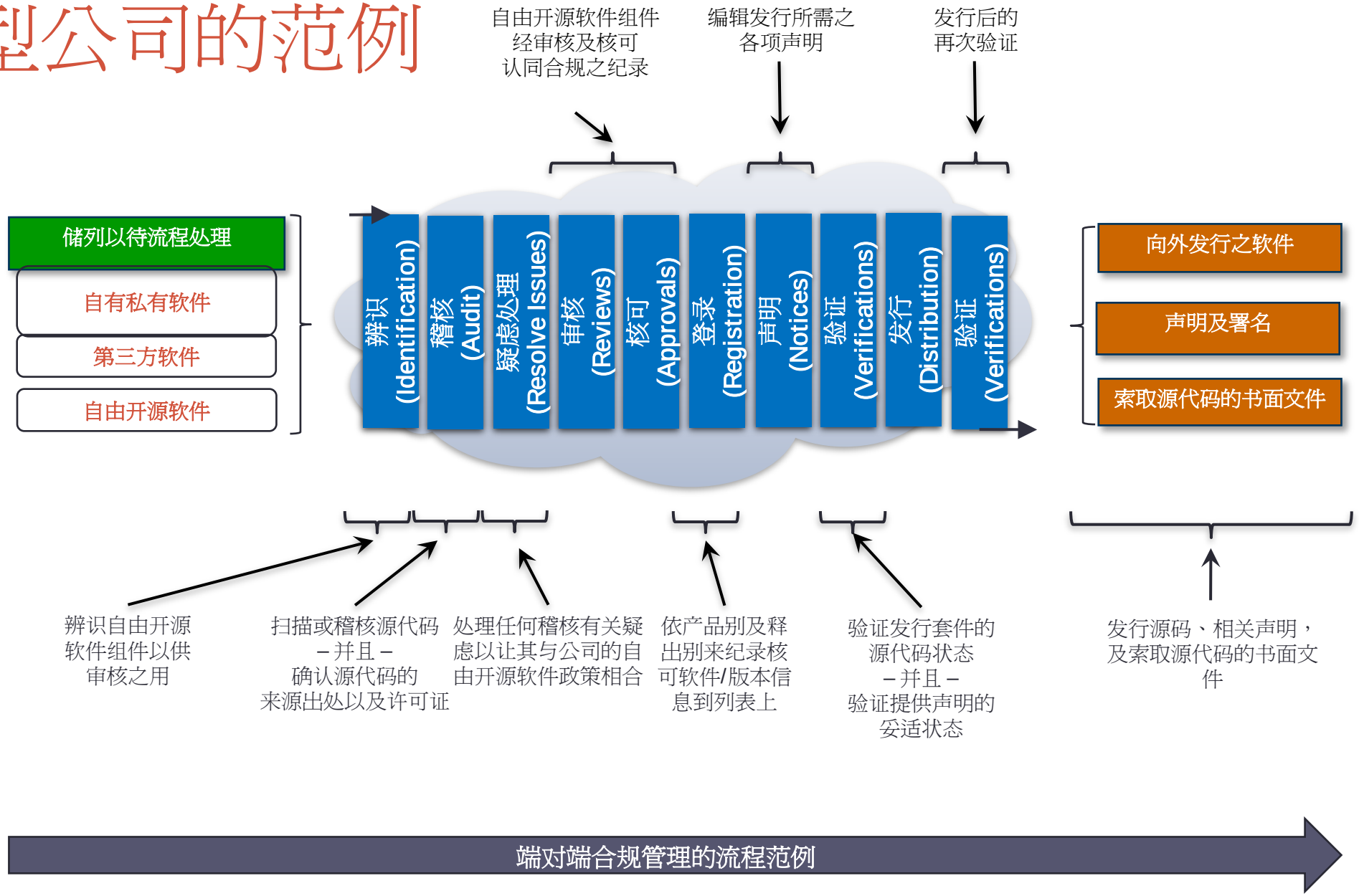
1. 在取得/开发的早期过程即发掘所有的自由开源软件
2. 审核及批准所有使用到的自由开源软件组件
3. 查验满足自由开源软件义务性要求的必要信息是否具足
4. 审核及批准任何对外部自由开源软件项目的贡献

需要的支援项目：

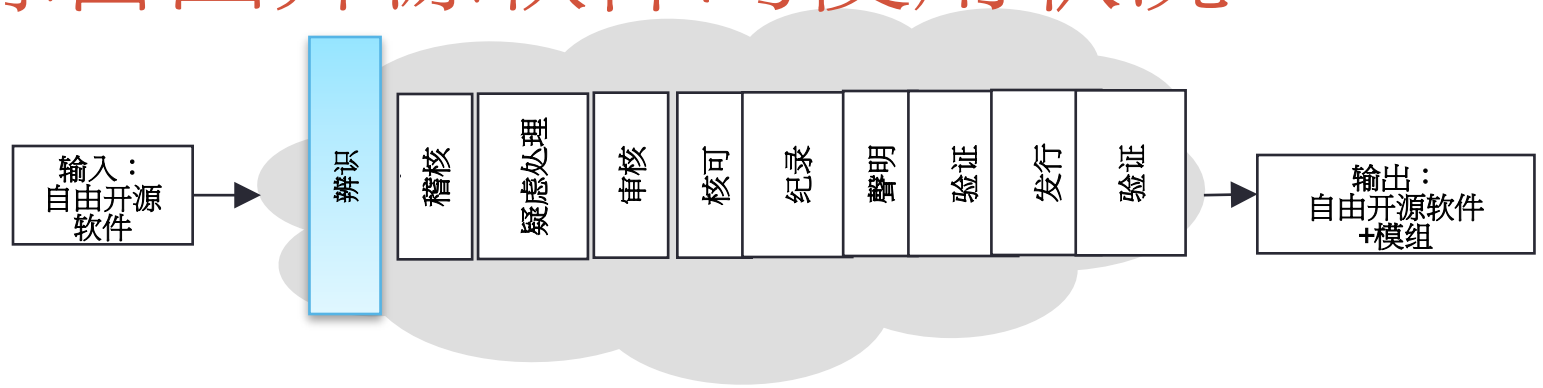
1. 确认胜任的合规工作人员，并就其职务责任指派清楚的界限
2. 采纳到既存的企业管理流程里，来支持自由开源软件合规专案
3. 提供组织的自由开源软件政策之训练课程给所有人
4. 对所有自由开源软件合规举措进行历程追踪

你可在此取得这些项目的详细查核清单: https://www.linuxfoundation.org/projects/opencompliance/self-assessment-compliance-checklist

企业型公司的范例


自由开源软件组件
经审核及核可
认同合规之纪录
编辑发行所需之
各项声明
发行后的
再次验证
储列以待流程处理
自有私有软件
第三方软件
自由开源软件
辨识
(Identification)
稽核
(Audit)
疑虑处理
(Resolve Issues)
审核
(Reviews)
核可
(Approvals)
登录
(Registration)
声明
(Notices)
验证
(Verifications)
发行
(Distribution)
验证
(Verifications)
向外发行之软件
声明及署名
索取源代码的书面文件
辨识自由开源
软件组件以供
审核之用
扫描或稽核源代码
– 并且 –
确认源代码的
来源出处以及许可证
处理任何稽核有关疑
虑以让其与公司的自
由开源软件政策相合
依产品别及释
出别来纪录核
可软件/版本信
息到列表上
验证发行套件的
源代码状态
– 并且 –
验证提供声明的
妥适状态
发行源码、相关声明，
及索取源代码的书面文
件
端对端合规管理的流程范例

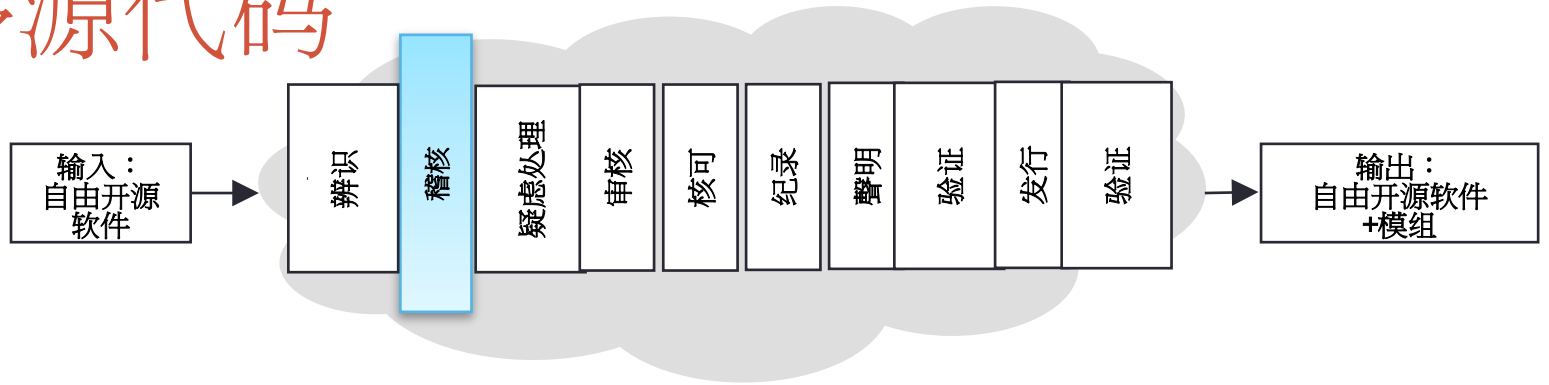
辨识及追踪自由开源软件的使用状况



辨识自由开源软件组件

- 步骤：
  - 来自工程师的输入要求
  - 扫描软件
  - 对第三方软件的发现尽相当努力(Due Diligence)
  - 将人工辨识之新组件信息加到知识库
- 成果：
  - 对该自由开源软件的合规纪录被建立或更新
  - 依自由开源软件政策所订，穷尽或限定范围内，对源代码审核之稽核将被要求。

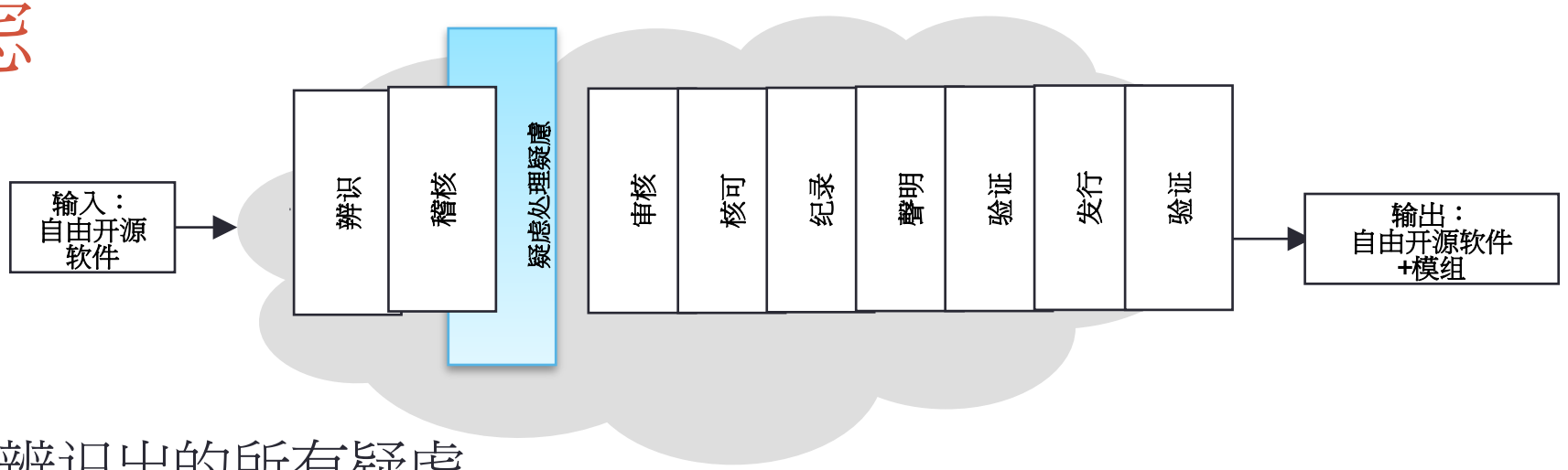
稽核程序源代码



辨识自由开源软件许可证

- 步骤：
  - 供稽核之程序源代码被辨识
  - 源代码或已使用软件工具进行扫描
  - 稽核或扫描的「成果(Hits)」被审核及验证，而得作为该程序代码适宜的来源信息
  - 稽核或扫描依该软件的开发与释出周期而被反覆操作
- 成果：
  - 稽核报告能用以辨识：
    1. 程序源代码的原始出处及其许可证
    2. 有待处理的疑虑

处理疑虑

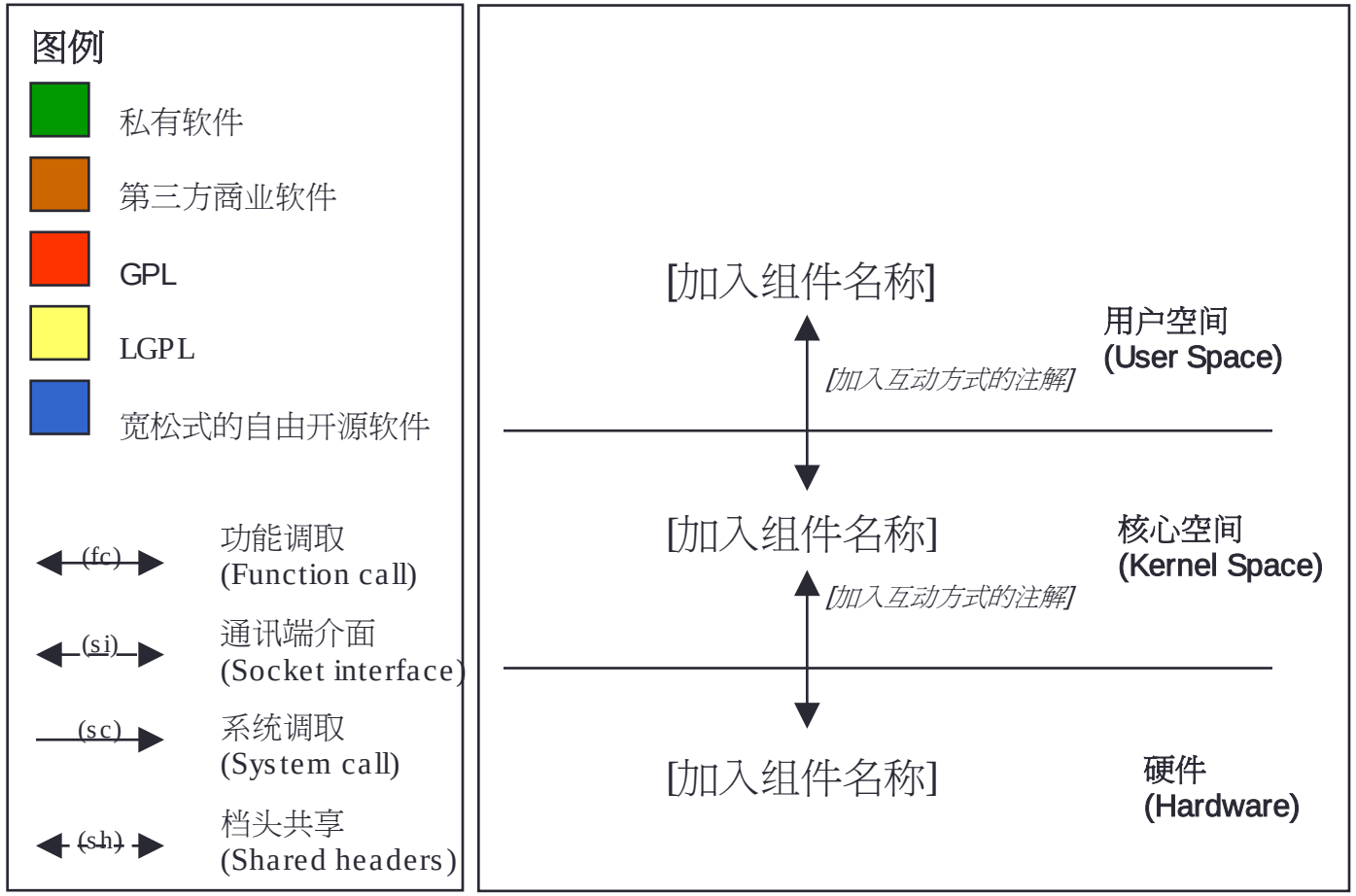


处理稽核过程辨识出的所有疑虑

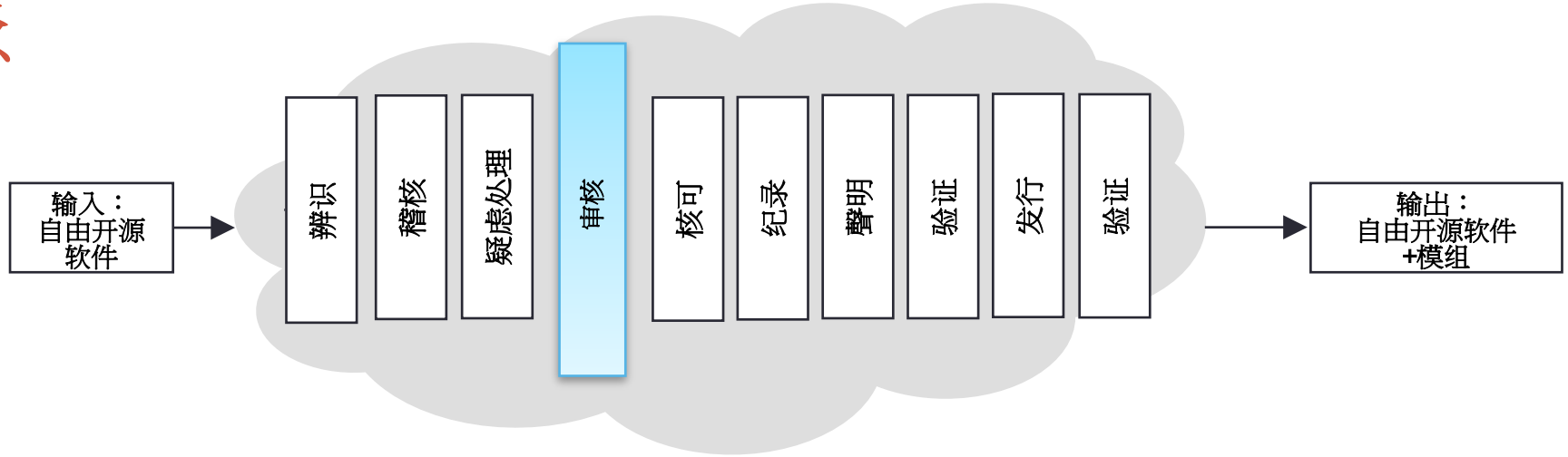
- 步骤：
  - 提供反馈予相应的工程师，以处理在稽核报告里，与你的自由开源软件政策冲突的疑虑
  - 该工程师接续就相关的程序源代码实施自由开源软件审核(样板可参阅下一张简报)
- 成果：

  对报告里每一个被标记的文档作处理，及对任何标记许可冲突的状况作处理

审核架构(样板范例)


图例
私有软件
第三方商业软件
GPL
LGPL
宽松式的自由开源软件
(fc)
功能调取
(Function call)
(si)
通讯端介面
(Socket interface)
(sc)
系统调取
(System call)
(sh)
档头共享
(Shared headers)
[加入组件名称]
用户空间
(User Space)
[加入互动方式的注解]
[加入组件名称]
核心空间
(Kernel Space)
[加入互动方式的注解]
[加入组件名称]
硬件
(Hardware)

执行审核



审核已处理之疑虑以确认其与你的自由开源软件政策相合

步骤：

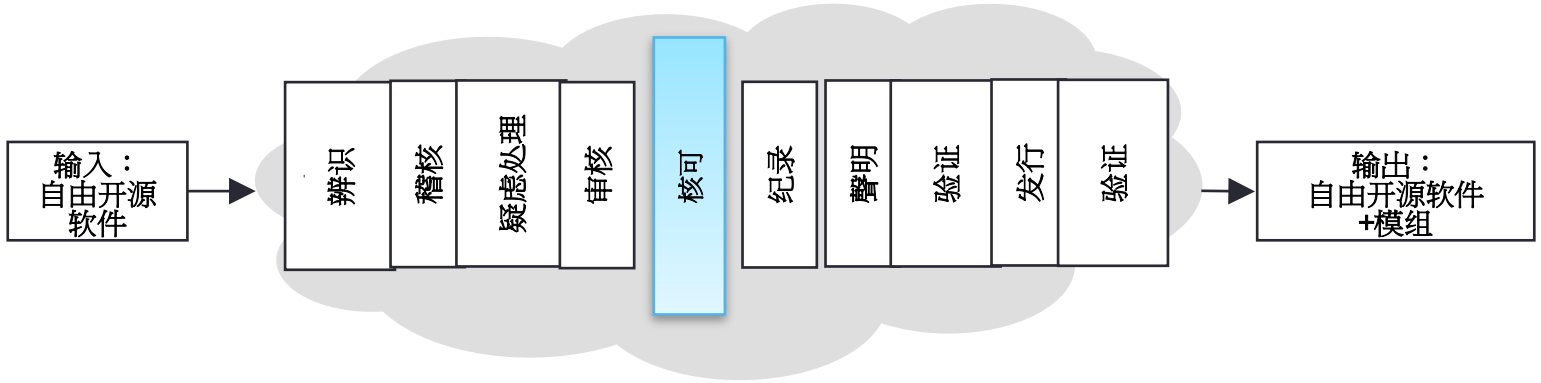
- 於审核工作人员里，应包含适切对应的管理阶层
- 依你的自由开源软件政策为参据来实施审核

成果：

- 确保在稽核报告里的软件与自由开源软件政策相合
- 准备往下一个步骤前(例如：核可前)，保存稽核报告的发现，并标注已处理的疑虑

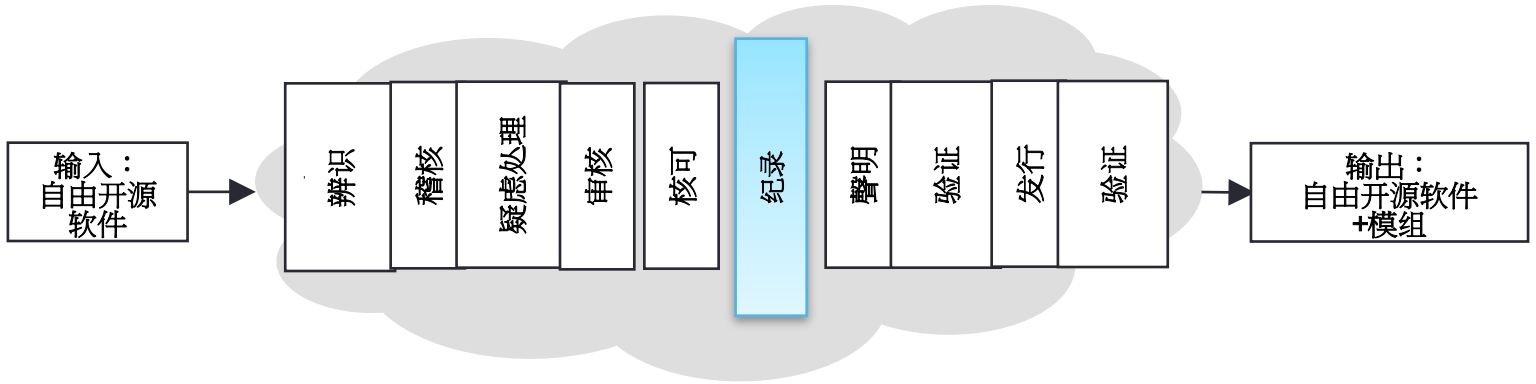
核可

- 根据上一个步骤的软件稽核及审核结果，软件可能被核可或可能不被核可使用
- 核可时，必须注明被核可的自由开源软件版本、被核可组件的使用模式，以及其他依自由开源软件许可证应施行的义务性要求
- 核可须对应到适宜的行政管理阶层来进行

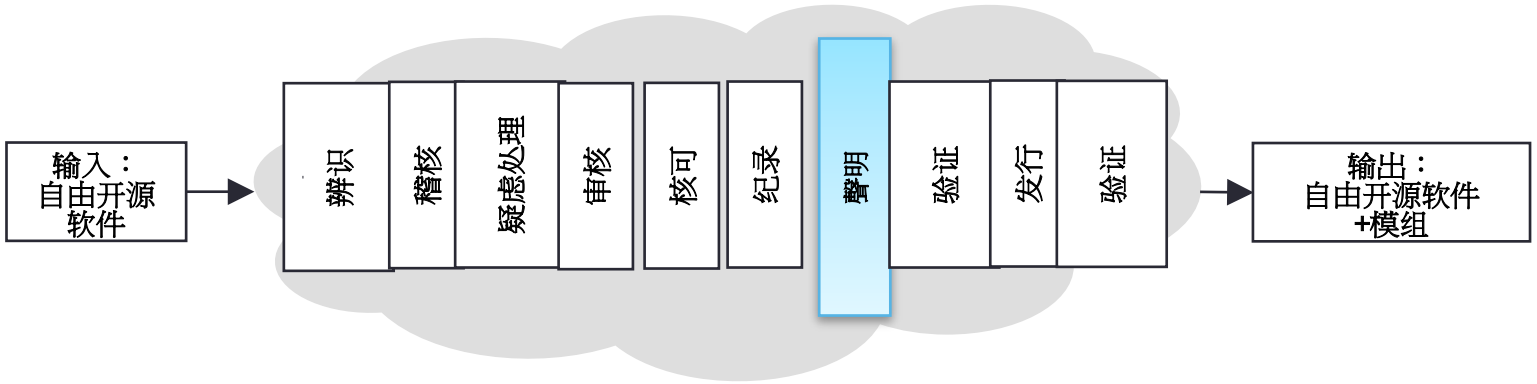


纪录 / 核可追踪

- 当一个自由开源软件组件被核可在产品中使用时，其应被加入该产品的软件清单
- 该项核可及核可的条件，必须被登记纪录在可追踪系统里
- 若新版本的自由开源软件组件或新的使用模式被提出时，该追踪系统必须清楚显示这需要一个新的核可



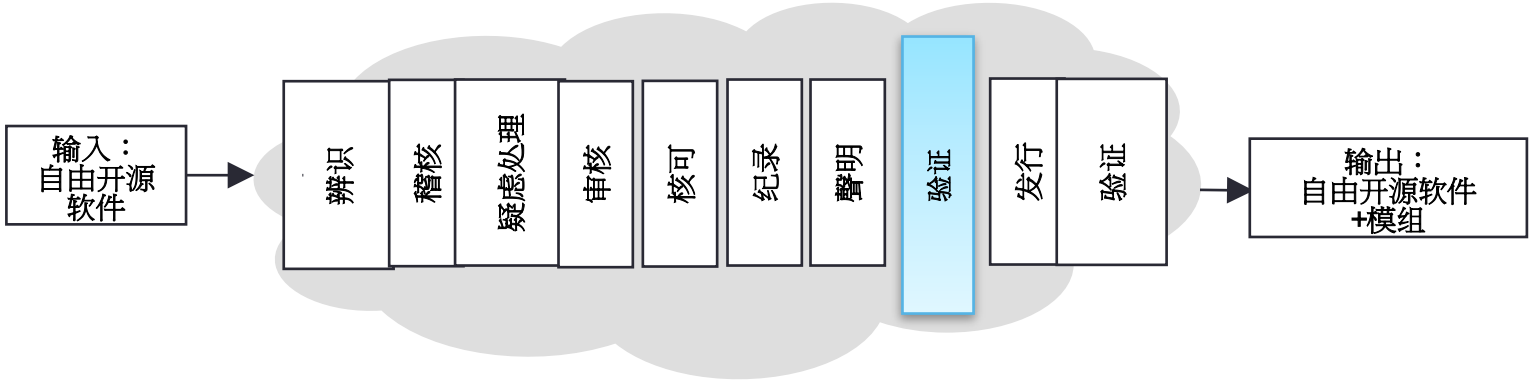
声明



为发行产品中任一自由开源软件备妥适宜的声明：

- 藉由完整著作权及署名声明之提供，来承认自由开源软件的使用
- 通知产品的终端使用者，如何获得自由开源软件程序源代码的复制件（当此要求适用时，例如 GPL 及 LGPL 即为此种状况）
- 应需求复制产品里自由开源软件程序代码之全部许可证协议文件

发行前的验证

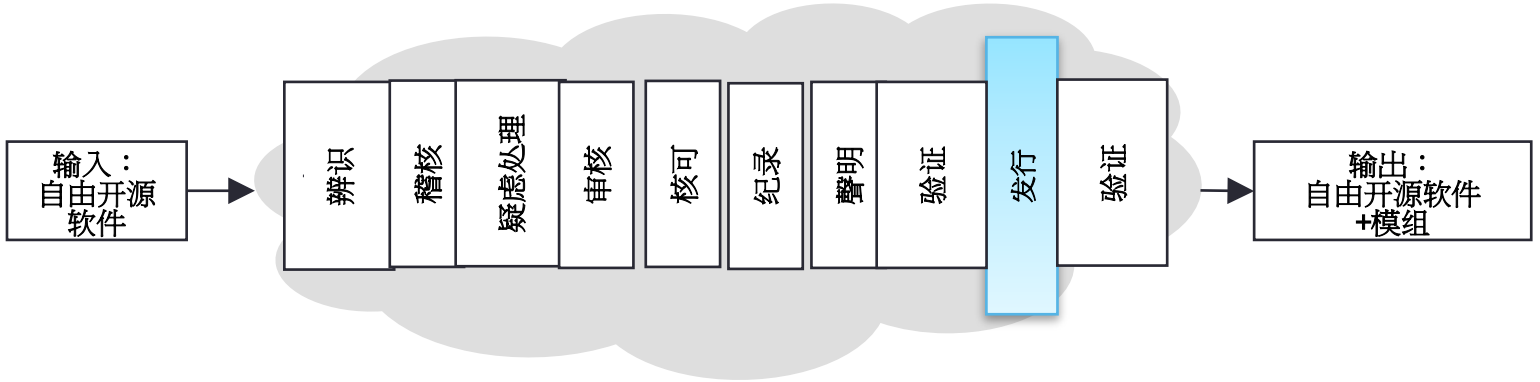


验证发行的软件已经过审核及核可

- 步骤：
  - 验证预计发行的自由开源软件套件已经过辨识及核可
  - 验证已经过审核的程序源代码与贩售产品里相对应的二进位代码是符合的
  - 确保已被审核的源代码符合相对应的产品执行文档
  - 验证所有相应的声明已被列入，以告知终端使用者其索取已被辨识之自由开源软件程序源代码的权利
  - 确保所有相应的声明已被纳入好让终端用户知道他们能获取相关自由开源软件的权益
  - 验证合规于其他已被辨识的义务性要求

- 成果：
  - 使发行的套件仅会包含已经过审核及核可的软件
  - 「供发行的合规稽证(Artifacts)」(依 OpenChain 规范书所定义)，包括被列入发行套件或其他投递模式所相应的声明文档

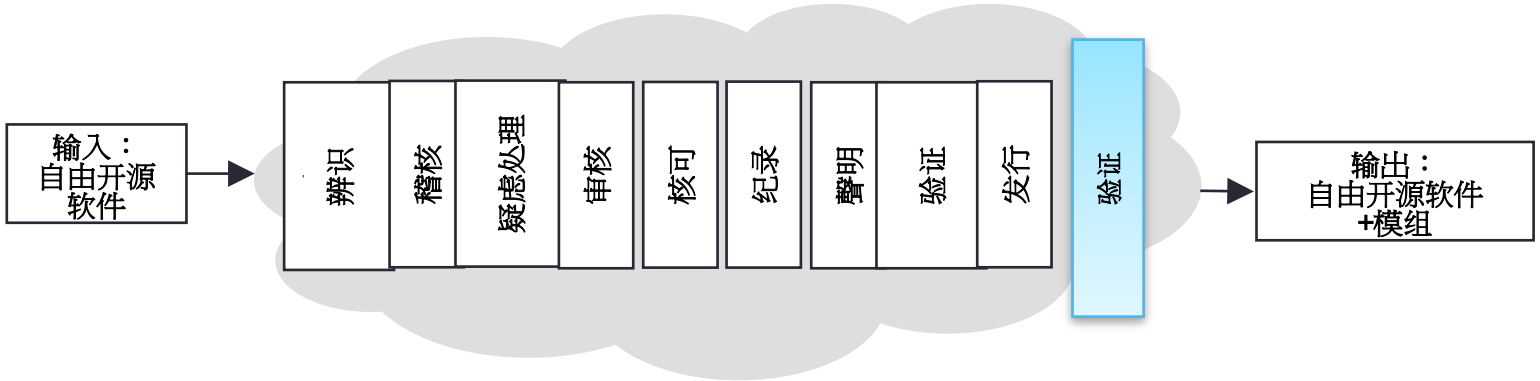
相应程序源代码的发行



依据要求提供相应的程序源代码

- 步骤：
  - 提供伴随任何相关联建置工具以及文件的对应程序源代码（例如，上传到发行网站上或列入发行套件里）
  - 此相应程序源代码应被辨识与标记，以和其产品及版本别对应
- 成果：
  - 提供相对应程序源代码的义务性规则被满足

最后验证



确认合规於许可证的义务性规定

- 步骤：
  - 验证相对应的程序源代码（若有的话）已经被正确地上传或发行
  - 确保其他许可证有被遵守
  - 验证上传或发行的程序源代码与经核可的版本是相对应的
  - 所需声明已被适当地发布与提供验证
  - 验证其他被辨识出的义务性要求已达到
- 成果：
  - 验证供发行的合规稽证(Artifacts)已被适切地提供

检测你的了解程度

- 在合规尽职工作(compliance due diligence)里牵涉到哪些事项？(就我们的范例流程里高项次的步骤作说明)
  - 辨识(Identification)
  - 稽核程序源代码(Audit source code)
  - 处理疑虑(Resolving issues)
  - 执行审核(Performing reviews)
  - 核可(Approvals)
  - 纪录/追踪核可(Registration/approval tracking)
  - 声明(Notices)
  - 发行前的验证(Pre-distribution verifications)
  - 相应程序源代码的发行(Accompanying source code distribution)
  - 验证(Verification)
- 什麼是结构性审核要追求的？

章节七

避开合规陷阱

合规陷阱

这个章节将会描述一些潜在的合规陷阱，以在合规程序中避免之：

1. 知识财产(IP)陷阱
2. 许可证合规陷阱
3. 合规流程陷阱

知识财产陷阱

类型 & 描述	发现	避免
意外地将 Copyleft 自由开源软件囊括进私有软件或第三方程序代码： 这种类型的错误发生在开发过程，当工程师将自由开源软件程序代码，添加到预定将采私有状态之程序源代码，造成与自由开源软件政策相抵触之情况。	此类型之错误可以透过程序源代码的扫描或稽核，以找出与下列的可能相合： • 自由开源软件程序源代码 • 著作权声明 可以使用自动化程序源代码扫描工具来完成此目标	此类型的错误可透过以下方式避免： • 为工程师人员提供合规疑虑、自由开源软件许可证差异，及列入自由开源软件到私有程序源代码的隐忧之相关训练 • 为建置环境里的所有程序源代码，定期执行程序源代码的扫描与稽核。

知识财产陷阱

类型 & 描述	发现	避免
意外地将 Copyleft 自由开源软件与私有软件的程序源代码键结在一起： 这种类型的错误发生在将许可证冲突或不相容的软件键结的结果。键结产生的法律效果为何，於自由开源软件社区里仍有争议。	这种类型的错误可以使用相依性追踪工具来发现，其可用来显示不同软件组件之间的键结性。	此类型的错误可透过以下方式避免： 1. 为工程人员提供相关训练，以避免键结到与你自由开源软件政策有所抵触的软件组件，将能在这些法律风险上站稳脚步 2. 在你的建置环境上，持续地在执行相依性追踪工具
透过修改程序源代码，将私有软件程序代码包含到 copyleft 自由开源软件里	此类型之错误，可以透过稽核或扫描来辨识及分析你采用到自由开源软件组件的程序源代码。	此类型的错误可透过以下方式避免： 1. 对工程人员提供教育训练 2. 执行周期性的程序代码稽核

许可证合规陷阱

类型 & 描述	避免
未能提供相应的程序源代码/适当的许可证、署名或声明信息	在产品上市前的产品释出循环，使程序源代码撷取及发布一个核对清单项目(checklist item)，可避免此类型错误。
相应程序源代码提供不正确的版本	透过在合规流程里添加验证的步骤，确保二进位版本的相对应程序源代码被发布，可避免此类型错误。
对自由开源软件组件修改部分未能提供相对应的程序源代码	透过在合规流程里添加验证的步骤，确保修改部分的程序源代码被发布，而非仅及於自由开源软件组件的原始程序源代码，可避免此类型错误。

许可证合规陷阱

类型 & 描述	避免
未对自由开源软件程序源代码的修改进行标注： 未能依自由开源软件许可证的要求，去标注自由开源软件程序源代码已经过变动。(或所提供与修改有关的信息，在细即及清楚级别不充份，无法满足许可证)	此类型的错误可透过以下方式避免： 1. 於程序源代码发行前，将程序源代码的修改标记(markings)添加为一个验证步骤 2. 为工程人员提供教育训练，以确保其对将要释出的所有自由开源软件或私有软件更新著作权标记(markings)或许可证信息

合规流程陷阱

描述	避免	预防
开发者未请求使用自由开源软件的核可	就公司自由开源软件政策及流程，提供工程人员教育训练，能避免此类型的错误。	此类型的错误可透过以下方式预防： 1. 定期执行软件平台的完整扫描以侦测任何「未经揭露」的自由开源软件是不是被使用了 2. 就公司的自由开源软件政策及流程，提供工程人员教育训练 3. 将合规事宜列入职员的绩效评估
未参与自由开源软件教育训练	确认将自由开源软件教育训练的完成，视为职员专业养成计画之一环，并将其完成视为绩效评估的一部份，能避免此类型的错误。	透过指示工程人员必须在特定日期前完成自由开源软件训练课程，能预防此类型的错误

合规流程陷阱

描述	避免	预防
未对程序源代码进行稽核	此类型的错误可透过以下方式避免： 1. 定期执行程序源代码的扫描/稽核 2. 确保稽核是开发流程反覆执行的里程碑	此类型的错误可透过以下方式预防： 1. 提供适当职员人力以免进度落后 2. 实施定期稽核
未对稽核发现进行处理（分析扫描工具或稽核所回报的「命中值(hits)」）	当稽核报告未终局结束时，不容许合规指派项目被标示已处理(例如关闭)，能避免此类型的错误。	在自由开源软件合规流程中，实施未经核可则阻断的机制，能预防此类型的错误
未在时限内取得自由开源软件的审核	即使工程师仍未决定采用该自由开源软件的程序源代码，仍及早开启自由开源软件审核的要求，能避免此类型的错误。	透过教育训练能预防此类错误

产品出货前确保合规

- 公司必须在任何产品（以任何形式）出货前确保合规的优先性
- 将合规顺位提高能促进：
  - 在你的组织中更有效率的使用自由开源软件
  - 与自由开源软件社区及组织建立更好的关系

建立社群關係建立社区关系

当公司在商业产品中使用自由开源软件时，最好要与自由开源软件社区建立及维持良好关系；尤其是，你公司在使用及部署的自由开源软件项目有关的特定社区。

此外，与自由开源软件组织的良好关系，在被建议采取合规最佳方案时非常有用，当你经历合规疑虑时也非常有帮助。

与软件社区的良好关系，可能对双向沟通也很有帮助：向上游推送(upstreaming)更新，以及从软件开发者取得协助。

检测你的了解程度

- 在自由开源软件合规里可能发生哪些类型的陷阱？
- 请举一个知识财产陷阱的例子。
- 请举一个许可证合规陷阱的例子。
- 请举一个合规流程陷阱的例子。
- 提高合规优先序位有何好处？
- 和社区建立良好关系有何好处？

章节八

开发者准则

开发者准则

- 从质优并具良好支援的自由开源软件社区选用程序代码
- 寻求指引
  - 就每一个你在使用的自由开源软件组件取得正式的核可
  - 不就未经审核的程序代码登录到任何内部的程序源代码库(source tree)
  - 就自由开源软件项目的外部贡献取得正式的核可
- 保留既存的许可信息
  - 不要从任何你所使用的自由开源软件组件，移除或采任何方式妨碍既存的自由开源软件著作权许可或其他许可信息。所有於自由开源软件组件里的著作权及许可信息都该被维持完整。
  - 除非依自由开源软件许可证的要求(例如，已经修改的版本需更换名称)，不要去更动自由开源软件组件的名称。
- 应自由开源软件审核流程所需来搜集及保留自由开源软件项目信息

预见合规流程的需求

- 列入所需时间以在工作计画依照已建立的自由开源软件政策来进行
  - 依照开发人员指导书来使用自由开源软件，特别是合并(incorporating)或键结(linking)自由开源软件程序代码到私有或第三方程序源代码时，反之亦然。
  - 审核结构规划，并避免混合受不相容自由开源软件许可证拘束的组件
- 永远更新合规的验证 – 对每个产品
  - 就不同产品(product-by-product)的基础上验证合规性：单单因为自由开源软件套件被核可使用在一个产品中，不必然代表它也会被核可使用在第二个产品里。
- 及对每个自由开源软件更新版本的升级
  - 确保同样自由开源软件组件的每一个新版本被审核并核可
  - 当你升级自由开源软件套件的版本时，确定新版本的许可证是与旧版本相同(於版本升级之际许可证的改变是可能发生的)
  - 若自由开源软件项目的许可证变更了，确定该合规纪录被更新且新的许可证不会制造冲突

将合规流程适用到所有的自由开源软件组件

- 收受软件
  - 采行步骤以了解从供应商处传递的软件里有什麽自由开源软件
  - 就所有将被包含到你产品里的软件评估你的义务性要求
  - 永远就你软件供应商取得的程序源代码进行稽核，或者替代方案是，让软件供应商必须就任何你取得的程序源代码，递交程序源代码稽核报告给你，作为一项公司政策。

检测你的了解程度

OPENCHAIN

- 列举一些开发者工作上采用自由开源软件可以实施的一般准则。
- 你需要移除或修改自由开源软件许可的档头信息吗？
- 列举一些在合规流程里的重要步骤。
- 一个之前已经审核自由开源软件组件的新版本能如何制造新的合规疑虑？
- 你应如何描述收受软件会有哪些风险？

透过 Linux Foundation 维护并免费提供的「给开发者的合规基础」来学习更多：

https://training.linuxfoundation.org/linux-courses/open-source-compliance-courses/ compliance-basics-for-developers