操作系统片段引用治理

丁紫薇 麒麟软件合规研发工程师

开放原子开源基金会 OPENATOM FOUNDATION | OpenEuler

目录

- 片段引用概念解析
- 片段引用弊端
- 片段引用案例分析
- 自研软件片段引用自动分析
- 非自研软件片段引用治理分析

片段引用概念解析

片段引用概念解析

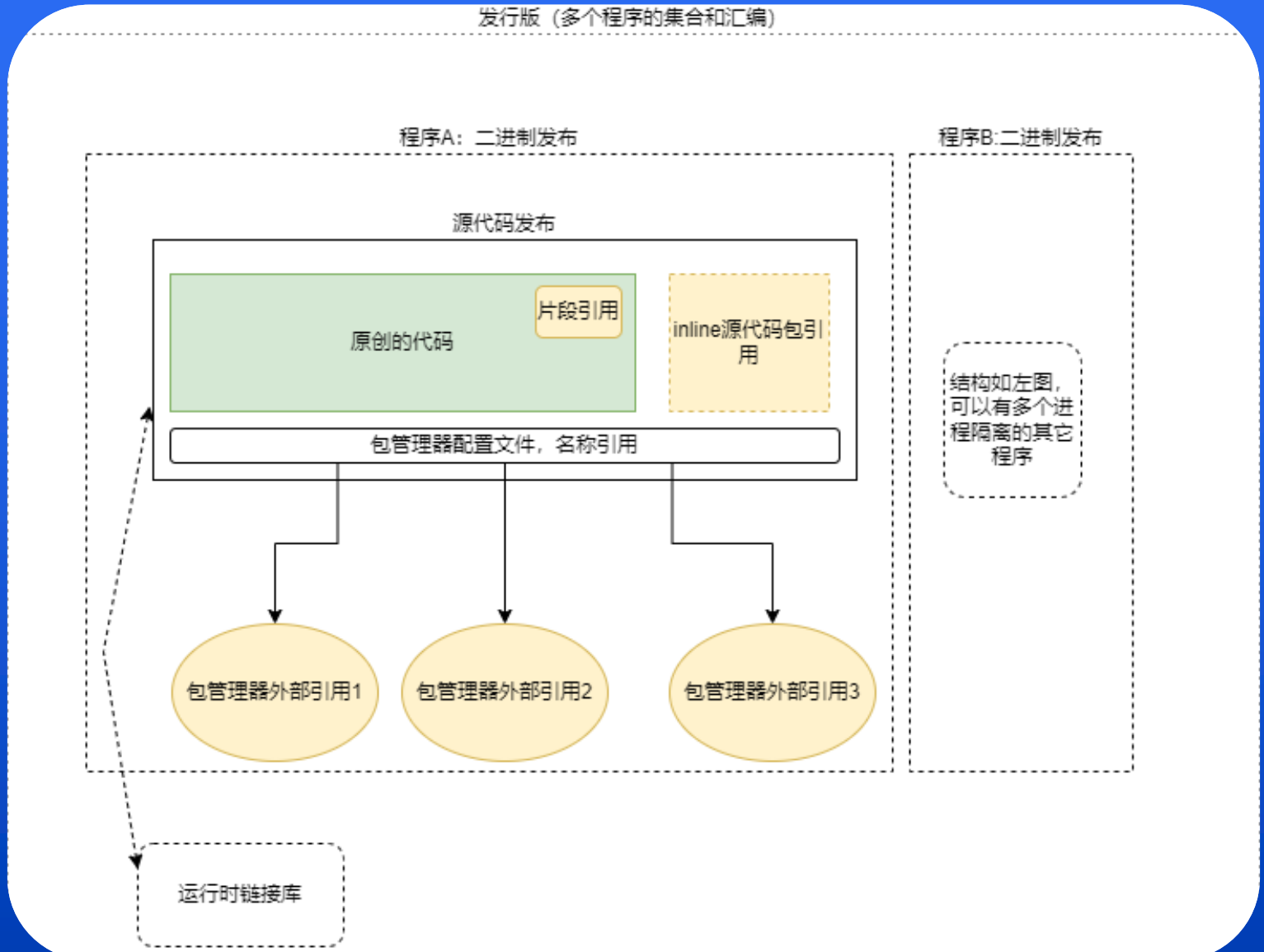
1、片段引用

2、整包inline引用

3、包管理器引用

4、运行时链接库

定义：在原有的代码文件中引用（原样复制或简单修改）第三方开源软件代码片段，或项目中添加第三方开源软件部分代码文件。



片段引用弊端

片段引用弊端介绍

不易溯源风险 1

可能导致代码库的可读性下降，增加维护难度，特别是在需要了解和修改引用代码的情况下。

3 法律风险

可能面临法律责任，包括罚款、赔偿以及停止使用侵权代码。

难以进行升级和缺陷修复风险 2

影响： 项目可能会错过重要的功能更新、性能改进或安全补丁，因为维护代码变得困难。

4 声誉风险

可能失去用户信任，项目的可持续性和成功可能受到负面影响。

片段引用弊端-情况说明

01 不同的分支或衍生项目可能会存在相似的代码片段，导致代码的重复

02 开源组织通常会维护多个项目，这些项目可能相互依赖，但也可能包含相似的功能或实现，引发代码片段引用的问题

03 某位贡献者可能在多个项目中都有代码贡献，导致代码的片段引用

04 不同的贡献者可能专注于不同的领域和任务，他们可能在解决类似问题时使用相似的代码实现，导致在项目中存在相似的代码片段

05 长期存在的开源项目可能经历多个迭代和演进，新功能的引入或旧功能的重构可能导致存在多个版本的相似代码片段。

片段引用弊端——基本应对策略

- **针对片段引用的情况，有一些基本策略：**

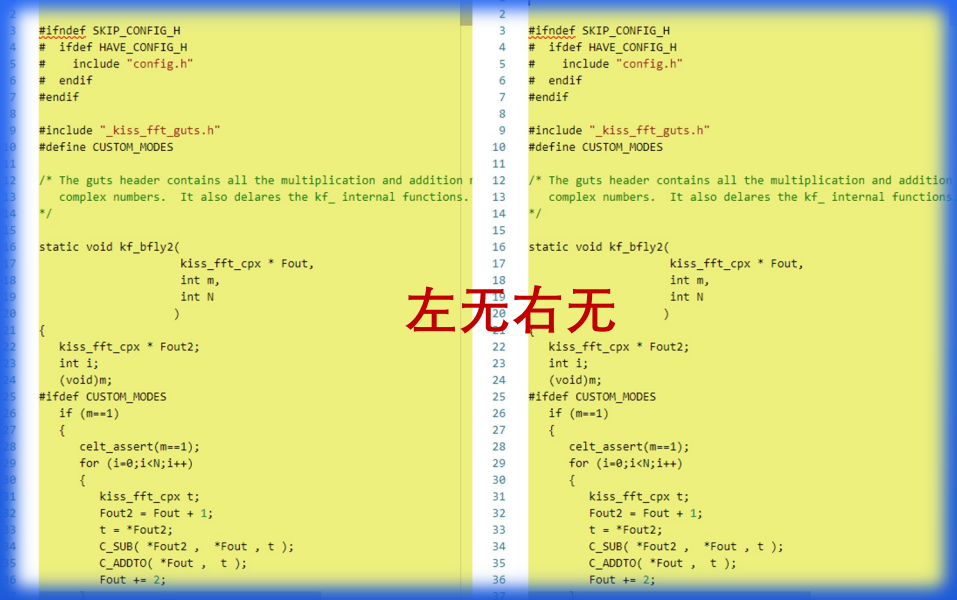
不鼓励但允许片段引用（如果能迁移成包管理器引用更好）

严格防止变更片段Copyright声明或License声明的情况

严格防止引入片段的License和项目的License不兼容的情况

片段引用案例分析

片段引用案例分析——案例情况复杂



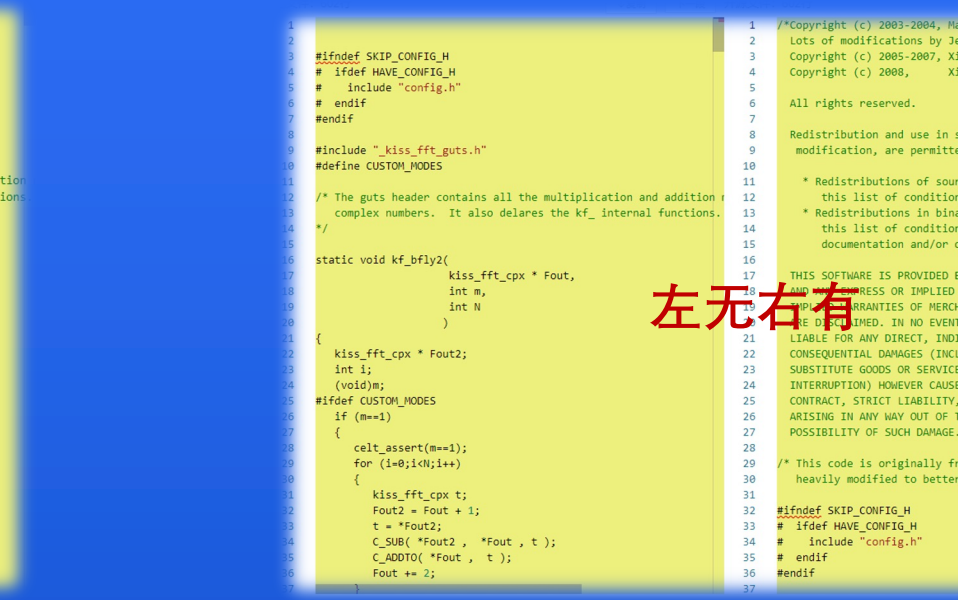
```
#ifndef SKIP_CONFIG_H
#  ifdef HAVE_CONFIG_H
#    include "config.h"
#  endif
#endif

#include "_kiss_fft_guts.h"
#define CUSTOM_MODES

/* The guts header contains all the multiplication and addition
   complex numbers.  It also delares the kf_ internal functions.
*/

static void kf_bfly2(
                    kiss_fft_cpx * Fout,
                    int m,
                    int N
                   )
{
   kiss_fft_cpx * Fout2;
   int i;
   (void)m;
#ifdef CUSTOM_MODES
   if (m==1)
   {
      celt_assert(m==1);
      for (i=0;i<N;i++)
      {
         kiss_fft_cpx t;
         Fout2 = Fout + 1;
         t = *Fout2;
         C_SUB( *Fout2 ,  *Fout , t );
         C_ADDTO( *Fout ,  t );
         Fout += 2;
```

左无右无



```
#ifndef SKIP_CONFIG_H
#  ifdef HAVE_CONFIG_H
#    include "config.h"
#  endif
#endif

#include "_kiss_fft_guts.h"
#define CUSTOM_MODES

/* The guts header contains all the multiplication and addition
   complex numbers.  It also delares the kf_ internal functions
*/

static void kf_bfly2(
                    kiss_fft_cpx * Fout,
                    int m,
                    int N
                   )
{
   kiss_fft_cpx * Fout2;
   int i;
   (void)m;
#ifdef CUSTOM_MODES
   if (m==1)
   {
      celt_assert(m==1);
      for (i=0;i<N;i++)
      {
         kiss_fft_cpx t;
         Fout2 = Fout + 1;
         t = *Fout2;
         C_SUB( *Fout2 ,  *Fout , t );
         C_ADDTO( *Fout ,  t );
         Fout += 2;
```

左无右有

```
# Copyright 2022 Hu[redacted]., Ltd
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
# ==============================================================================
import numpy as np


def _pdist(a, b):
    """Compute pair-wise squared distance between points in `a` and `b`.

    Parameters
    ----------
    a : array_like
        An NxM matrix of N samples of dimensionality M.
    b : array_like
        An LxM matrix of L samples of dimensionality M.

    Returns
```

左有右无


```
# vim: expandtab:ts=4:sw=4
import numpy as np


def _pdist(a, b):
    """Compute pair-wise squared distance between points in `a` and `b`.

    Parameters
    ----------
    a : array_like
        An NxM matrix of N samples of dimensionality M.
    b : array_like
        An LxM matrix of L samples of dimensionality M.

    Returns
    -------
    ndarray
        Returns a matrix of size len(a), len(b) such that eleeme
        contains the squared distance between `a[i]` and `b[j]`.

    """
    a, b = np.asarray(a), np.asarray(b)
    if len(a) == 0 or len(b) == 0:
        return np.zeros((len(a), len(b)))
    a2, b2 = np.square(a).sum(axis=1), np.square(b).sum(axis=1)
    r2 = -2. * np.dot(a, b.T) + a2[:, None] + b2[None, :]
    r2 = np.clip(r2, 0., float(np.inf))
    return r2
```

```
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
# ==============================================================================

#ifndef SKIP_CONFIG_H
#  ifdef HAVE_CONFIG_H
#    include "config.h"
#  endif
#endif

#include "_kiss_fft_guts.h"
#define CUSTOM_MODES

/* The guts header contains all the multiplication and addition
   complex numbers.  It also delares the kf_ internal functions.
*/

static void kf_bfly2(
                    kiss_fft_cpx * Fout,
                    int m,
                    int N
                   )
{
   kiss_fft_cpx * Fout2;
   int i;
```

左有右有

片段引用案例分析——治理情况

- 需要关注4个要素：本地项目的License/Copyright(以下简称L/C), 本地文件的L/C, 远程项目（被引用的开源项目，以下统称远程项目）的L/C, 远程文件的L/C
- 如果文件级的License或Copyright未声明，则它会继承项目级的L/C声明，片段的L/C声明会继承文件的。

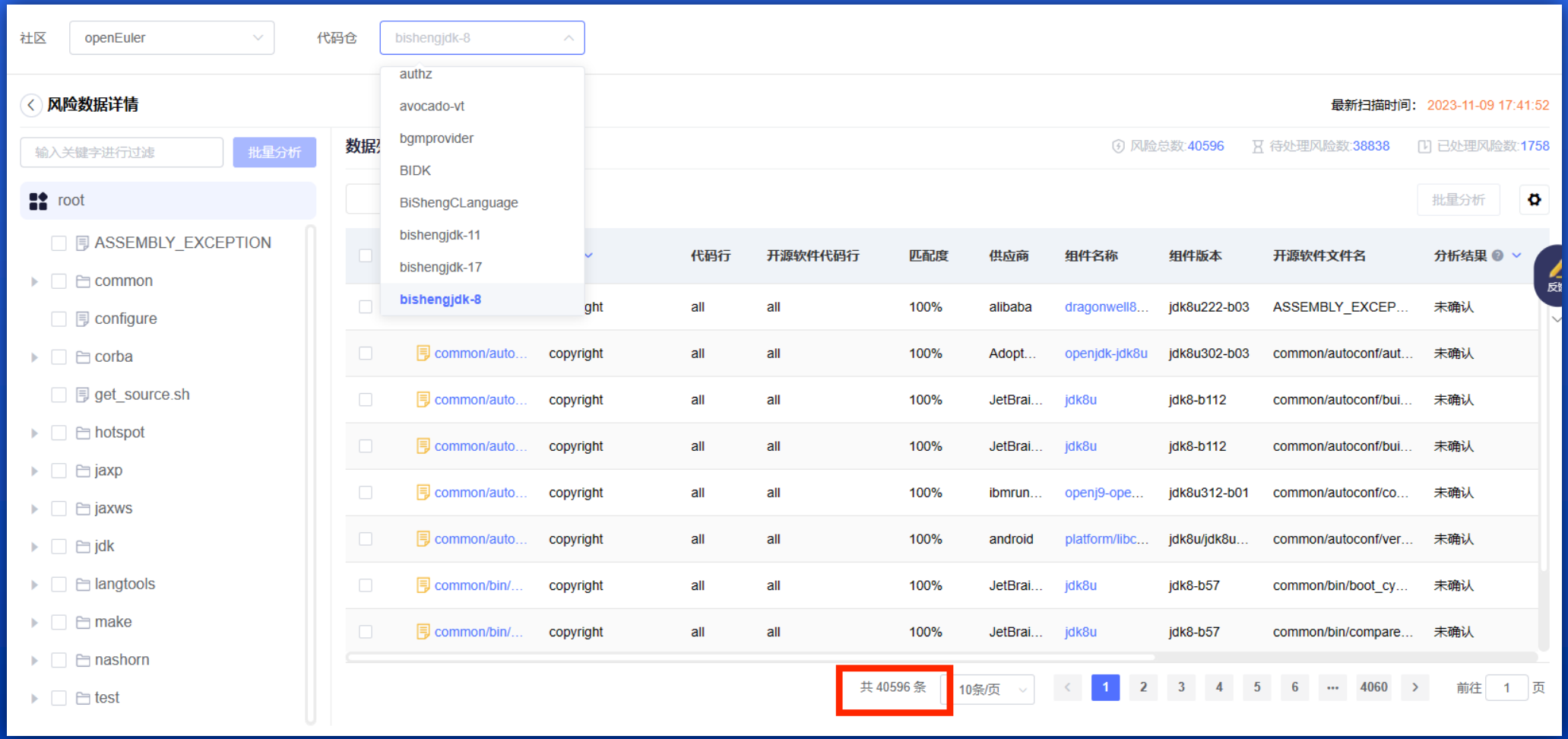
本地文件 L/C声明	远程文件 L/C声明	本地项目的声明 和远程项目的L/C声明	本地文件“实际的” L/C 声明	远程文件“实际“ L/C 声明	关注的风险	合规措施
空	空	LPL (Local Project License) LPC (Local Project Copyright) RPL (Referenced Project License) RPC (Referenced Project Copyright)	LPL, LPC	RPL, RPC	LPL 大概率不等于 RPL LPC 一定不等于RPC	本地文件添加RPL, RPC声明，并增加出处声明
空	有		LPL, LPC	RFL, RFC	LPL 大概率不等于 RFL LPC 一定不等于RFC	本地文件保留RFL, RFC，不要删除
有	空		LFL，LFC	RPL, RPC	检查：本地文件的L/C声明是否远程项目信息L/C一致。	如有风险：修改本地文件的L/C声明
有	有		LFL, LFC	RFL, RFC	检查： LFL = RFL and LFC = RFC	如有风险：修改本地文件的L/C声明

表格来源--openEuler合规sig 郑志鹏

自研软件片段引用自动分析

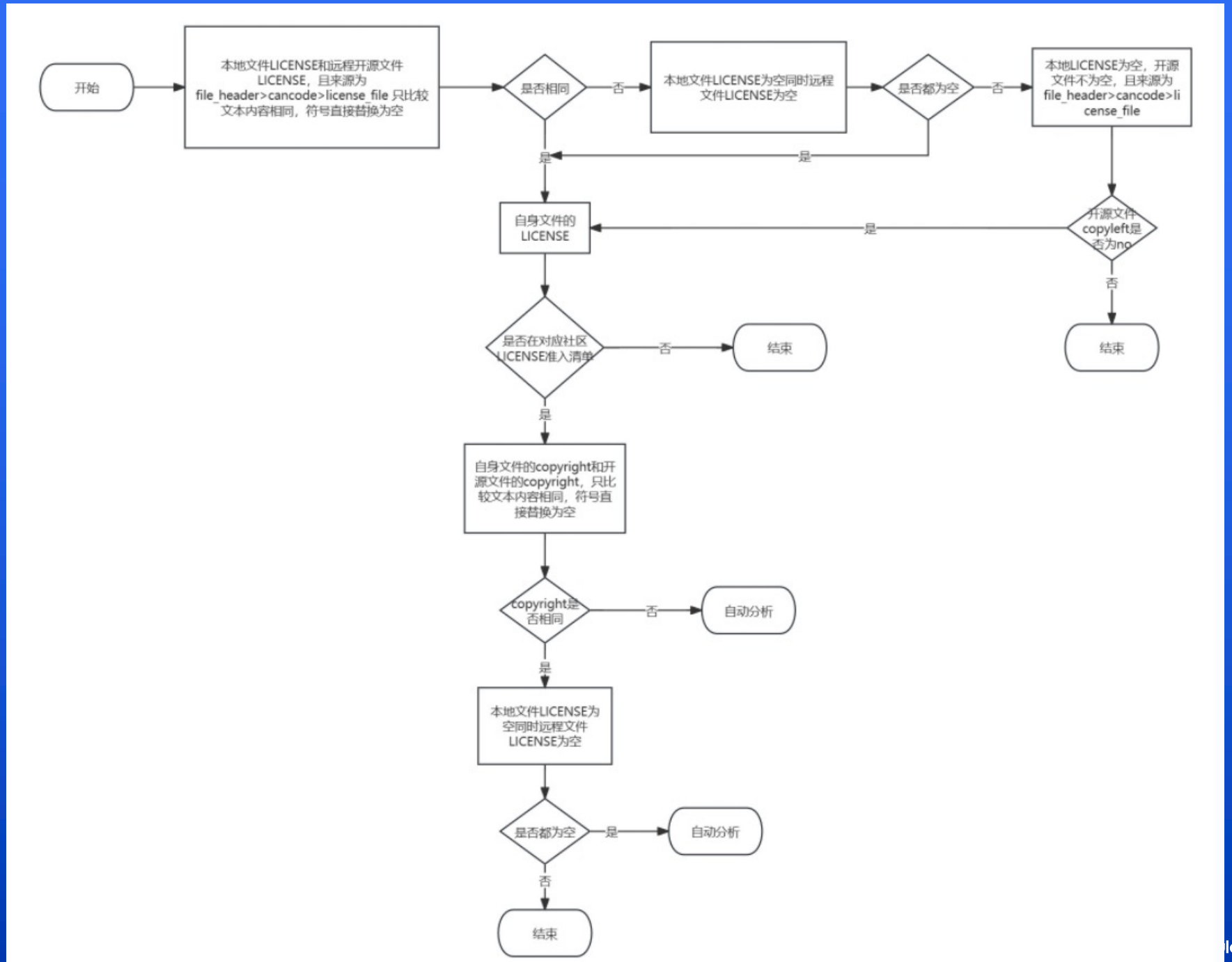
自研软件片段引用自动分析

识别到的片段引用过多，光靠人力是无法做到的，需要考虑自动分析



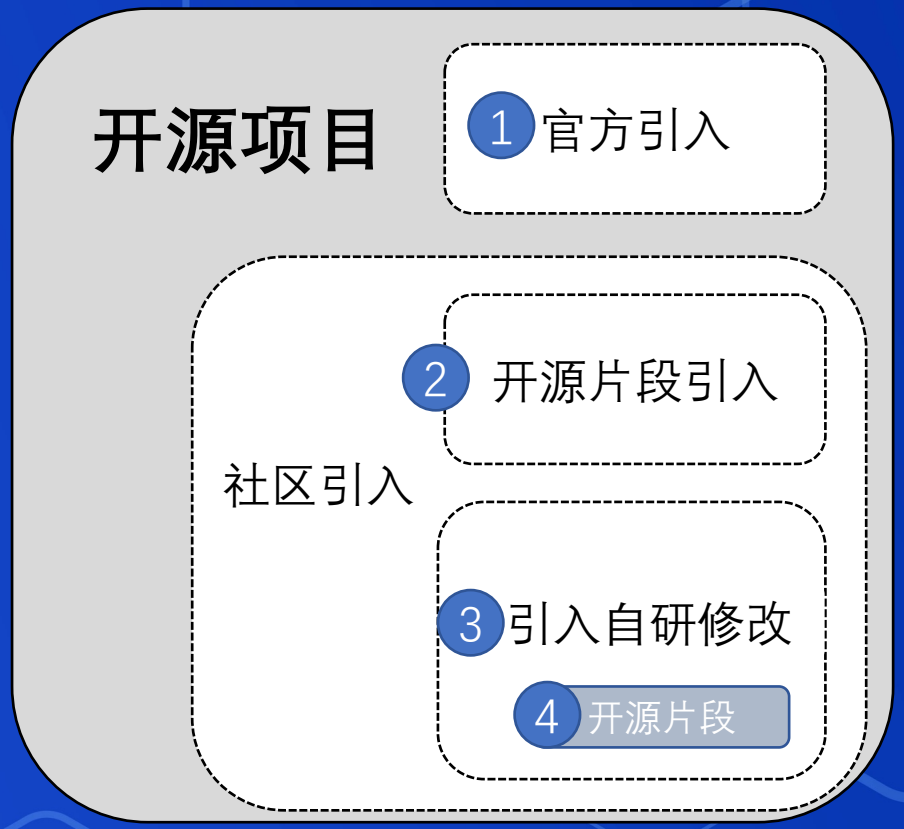
自动分析流程

比较本地文件与远程文件的License和Copyright，其中License来源为file_header>scancode>license_file， 制定一些评判标准



非自研软件片段引用治理分析

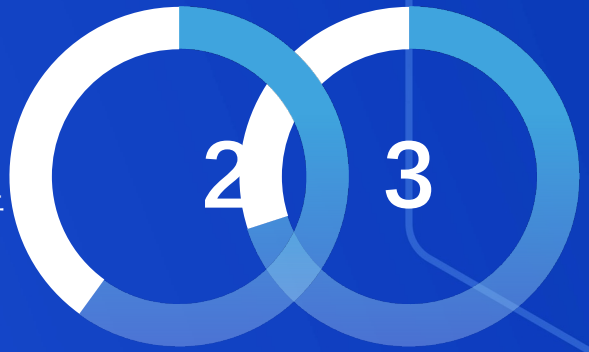
非自研软件片段引用治理分析



非自研软件片段引用治理分析

情况1为官方引入

比对片段引用的远程文件与开源项目的L/C是否兼容



情况2和3为社区开源项目的修复和特性增强

片段引用第三方代码，需要根据本地文件L/C和远程文件L/C进行判断，确定如何引入无风险

需比对两方

情况4是情况3中的一种情况，是社区引入的自研修改中存在开源片段引用的情况

这种情况需要查看开源软件，自研代码本身与自研代码的片段引用三方面的L/C,考虑他们之间的兼容性

需比对三方

THANKS

开放原子开源基金会
OPENATOM FOUNDATION
|
OpenEuler

开放原子开源基金会
OPENATOM FOUNDATION
|
OpenEuler