如何在软件开发生命周期中高效管理开源组件

OPPO / 朱红林

使用开源组件有哪些风险

开源组件免费、缩短软件开发周期、丰富软件产品的功能和业务，但蕴藏很多风险

安全技术风险

开源许可证合规风险

供应链风险

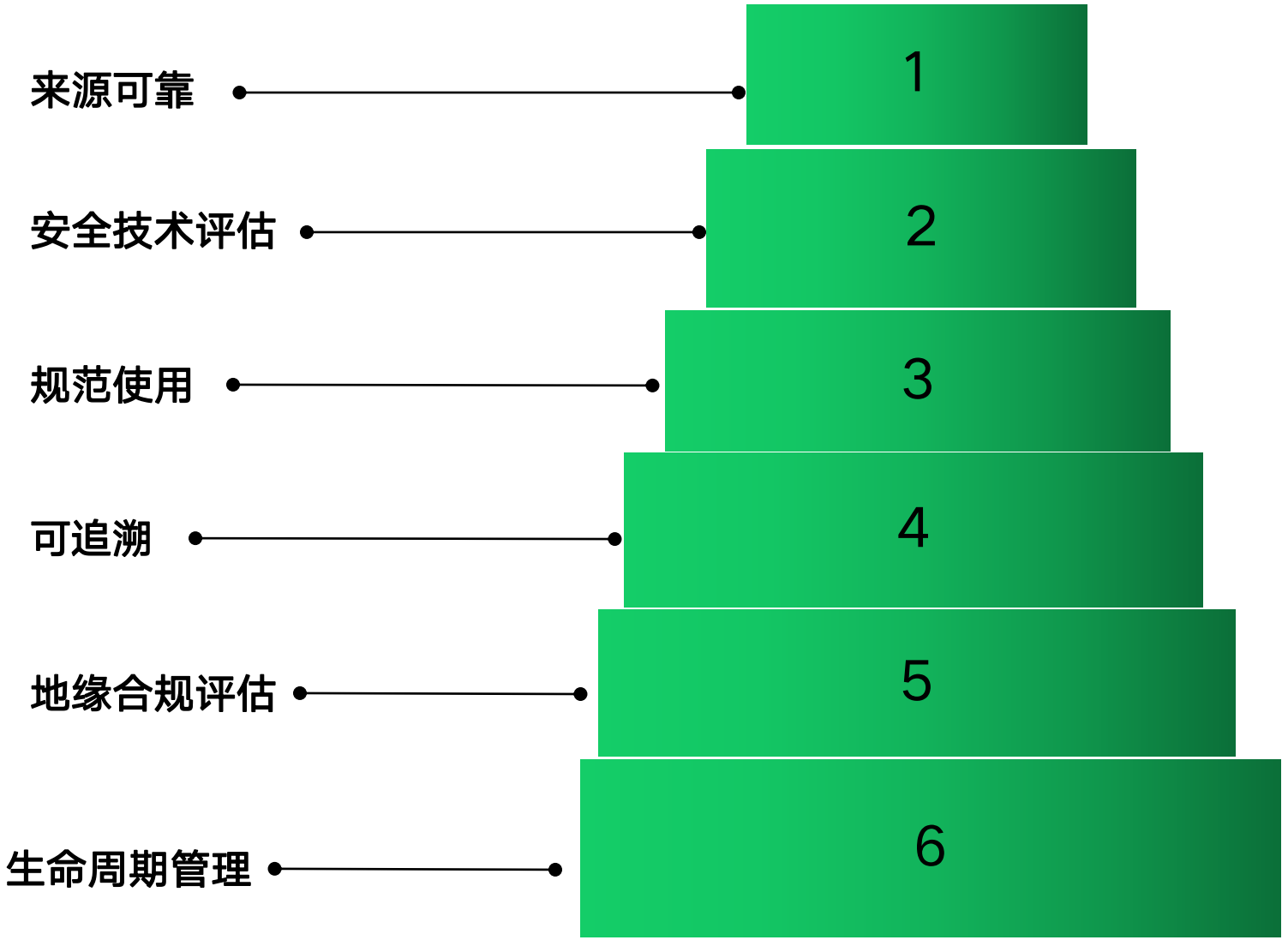
运维风险

专利侵权风险

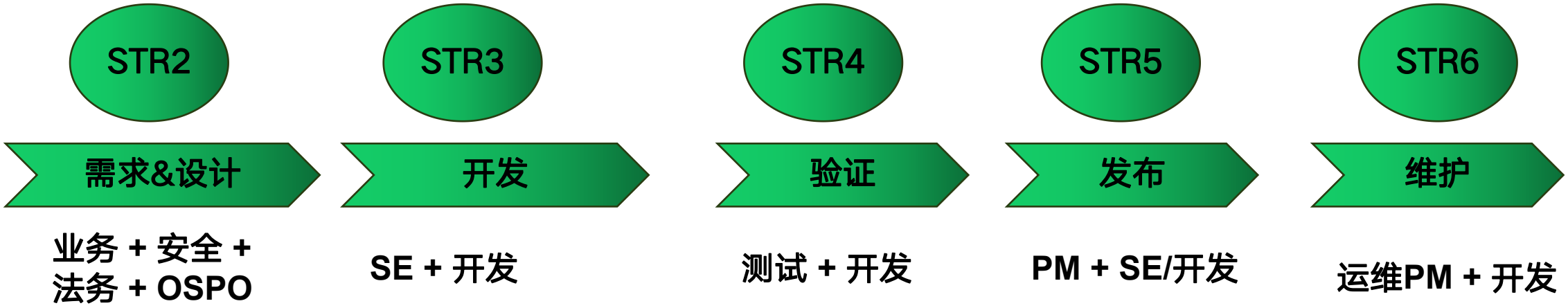
出口管制合规风险

地缘合规风险

如何管控开源组件的风险



如何基于SDL治理开源合规



- 开源风险评估
- 很选型&入库
- 使用申请
- BIS备案

- SCA工具扫描及治理
- Notice集成
- 使用申请更新

- 开源合规测试
- Notice更新
- 准出校验

- FSR
- 开源分发
- 应急响应计划

- 漏洞跟踪处理
- 生命周期维护
- 应急响应

IT基础设施　开源软件库、开源源码库/制品库、DevOps工具链、项目管理系统、PDM产品数据库、版本控制系统、故障跟踪系统.....

需求&设计阶段 - 选型评估

建立准入机制，从源头把控风险，是安全左移的重要举措，其中开源软件选型是关键活动之一。

	技术生态	开源许可证	安全	生命周期
考量因素	• 功能/性能满足程度 • 架构的高可用性、先进性、可靠性 • 兼容性 • 易用性 • 行业应用范围	• 许可证条款 • 专利陷阱 • 历史诉讼案例	• 已知安全漏洞 • 历史版本漏洞 • 历史漏洞处理情况	• 成立时间 • Start/Fork/Watching • 贡献者/FR • 版本更新频率

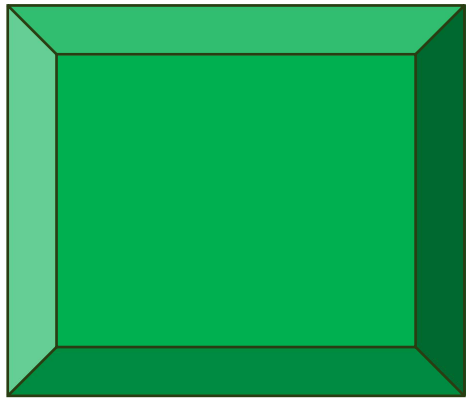
开源软件库

开源同源

- 官网同源
- 内网同源
- 版本同源

功能

- 开源软件库、许可证库、漏洞库、黑名单、台账
- 入库、使用申请、更新和推出
- 漏洞推出



开源软件库

运营

- Owner机制
- TMG/专家评审团
- 分级审核机制

需求 & 设计阶段 - 风险评估 & 使用申请

开源风险评估

依托开源合规库，进行开源风险评估

- 输入：开源合规调查
- 输出：风险项/合规需求
- 准出条件：所有项目/新增需求均已完成开源风险评估

开源软件使用申请

依托开源合规库，进行使用申请

- 先申请，后使用
- 从开源软件库申请

使用申请关注的信息

- 基本信息
- 使用信息

开发阶段 - 开源扫描 & 义务履行

规范使用

- 从开源软件库中引入
- 架构解耦
- 配置解耦
- 保留原始版权、许可证
- 禁止故意绕过工具检查
- 禁止代码片段引入
- 整包使用，不修改
- Patch管理

开源扫描

- SCA工具扫描
- SBOM生成
- SCA工具嵌入到CI/CD

问题整改

- 风险分析、整改计划
- 安全漏洞修复
- 合规治理

义务履行

- Notice集成
- 开源分发准备
- 修改说明

验证阶段 - 开源测试

测试计划

- 测试计划
- 测试用例

测试执行

- Notice集成测试
- 开源分发准备工作验收
- 其他义务履行情况验收
- 漏洞修复情况验收

准出校验

- 使用申请和SBOM一致
- 高风险问题处理完成
- Notice集成测试通过
- 开源分发准备就绪

发布阶段 - 开源发布

执行FSR，审核各项开源合规活动是否已执行完成，做好开源发布准备，并制定应急响应计划。

开源发布

主动开源

贡献整个项目

- 开源合规治理
- 确定许可证
- 其它相关审查
- Notice、发布说明

主动回馈社区

- 开源范围
- 编译通过
- 其他相关审查

被动开源

- 开源范围审核
- 不包含第三方专有代码
- 不包含公司商业秘密和其他敏感信息
- 出口管理审查
- 提供符合许可证要求的源码

维护阶段 – 漏洞跟踪和修复

漏洞收集

- 内部漏洞库
- SCA工具

漏洞跟踪和验证

- 快速定位
- 漏洞验证和分析
- 漏洞分发
- 产品漏洞预警

漏洞修复

- 临时紧急措施
- 源码补丁
- 跟随社区升级
- 现网产品漏洞修复支持

维护阶段 - 生命周期管理

及时升级到最新稳定版本

- 定期维护开源软件库
- 及时跟随社区升级

制定明确的推出机制和流程

- 定期维护黑名单
- 及时退出

IT系统支撑和工程能力建设

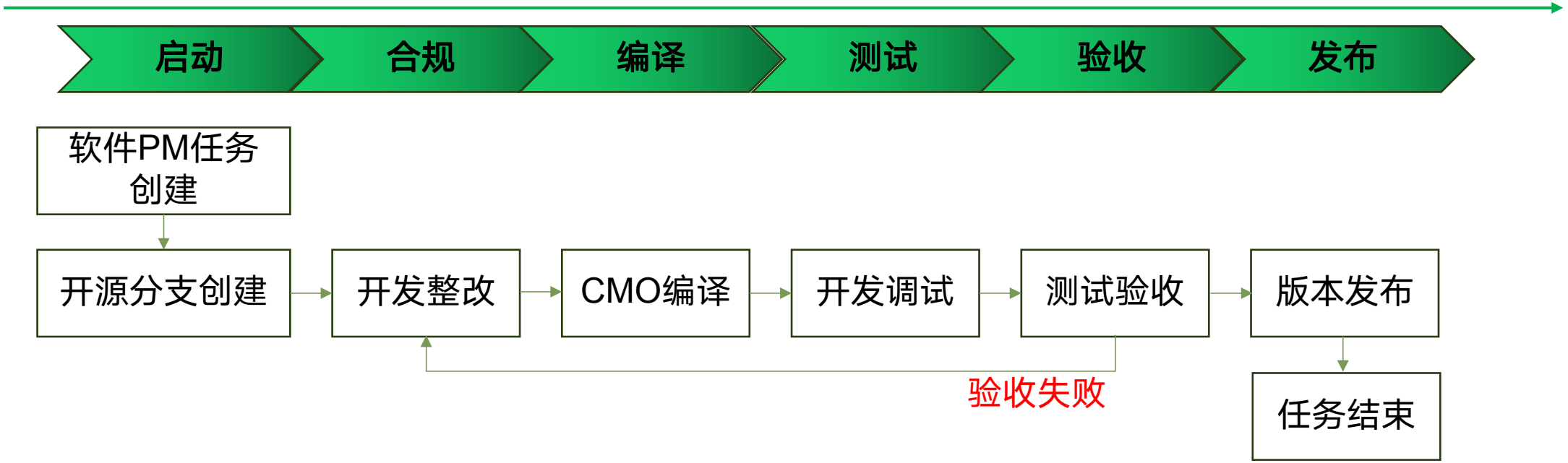
- 提供完善的IT系统支撑和较强的工程能力
- 溯源及自动化管理

高效治理开源组件的基本原理

- 开源各阶段实现自动化
- 管理过程做到可视化监控
- 总的目标：全程可控、问题闭环

旧的开源治理方法

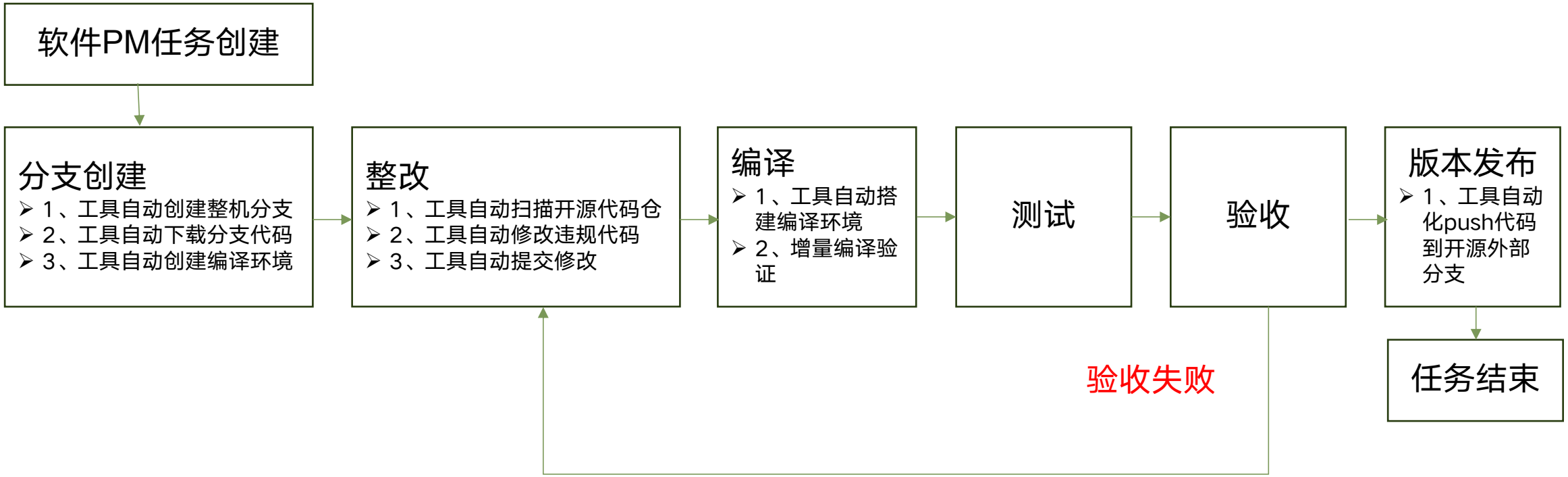
当前Kernel开源发布流程



痛点：

- 1、开源流程长，涉及工作领域比较多，人工操作比较多对，比较依赖经验，容易出错。
- 2、很多安全编码规范和开源规范，没有工具化，开源代码的质量难以得到保障。

新的开源治理方法

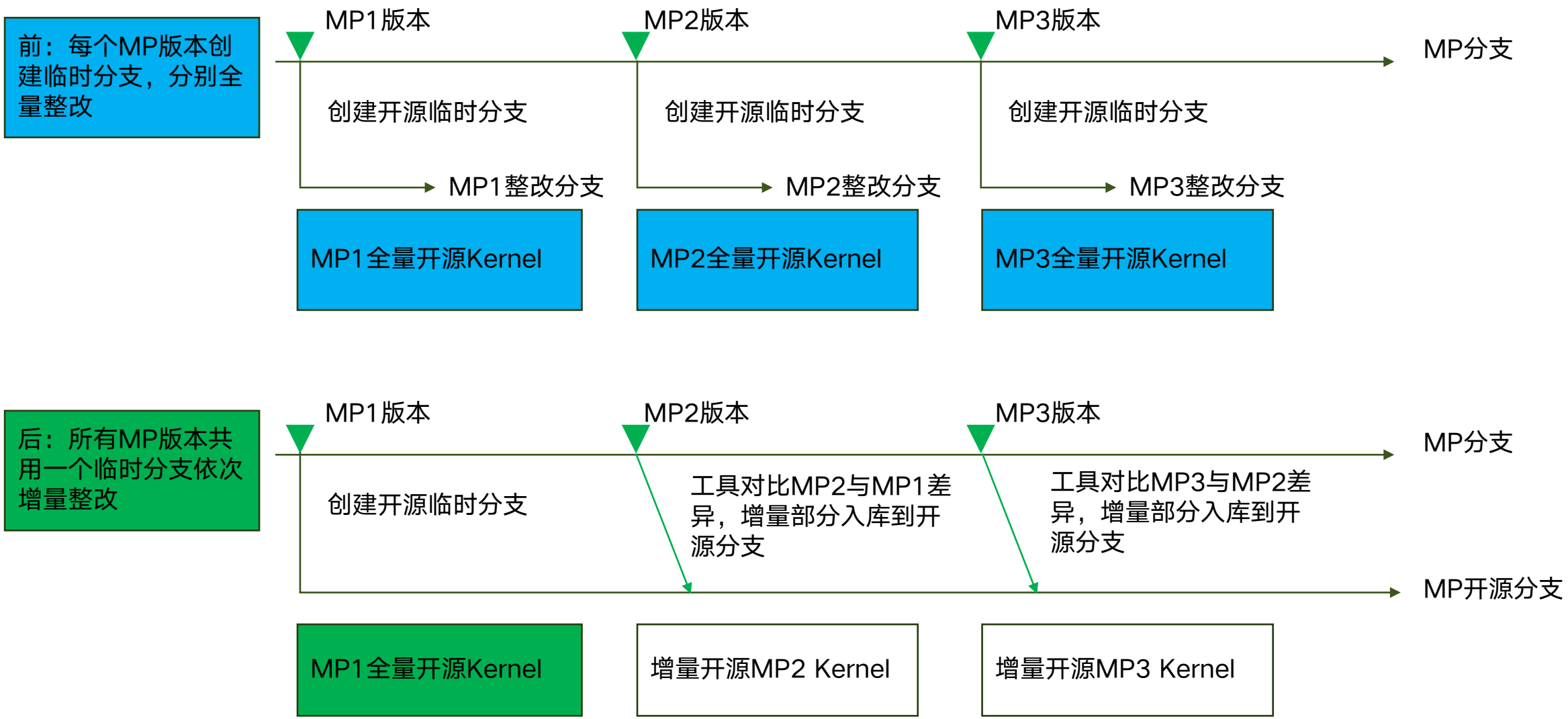


自动化实现开源过程说明：

➢ 1、将“分支创建”、“整改”、“编译”、“版本发布”等步骤实现工具自动化处理。

➢ 2、最大化减少人工操作，显著提高Kernel相关代码仓的开源效率和稳定性。

增量开源MP版本


前：每个MP版本创建临时分支，分别全量整改
MP1版本
MP2版本
MP3版本
MP分支
创建开源临时分支
创建开源临时分支
创建开源临时分支
MP1整改分支
MP2整改分支
MP3整改分支
MP1全量开源Kernel
MP2全量开源Kernel
MP3全量开源Kernel
后：所有MP版本共用一个临时分支依次增量整改
MP1版本
MP2版本
MP3版本
MP分支
创建开源临时分支
工具对比MP2与MP1差异，增量部分入库到开源分支
工具对比MP3与MP2差异，增量部分入库到开源分支
MP开源分支
MP1全量开源Kernel
增量开源MP2 Kernel
增量开源MP3 Kernel

想一想，我该如何把这些技术应用在工作实践中？

THANKS