上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

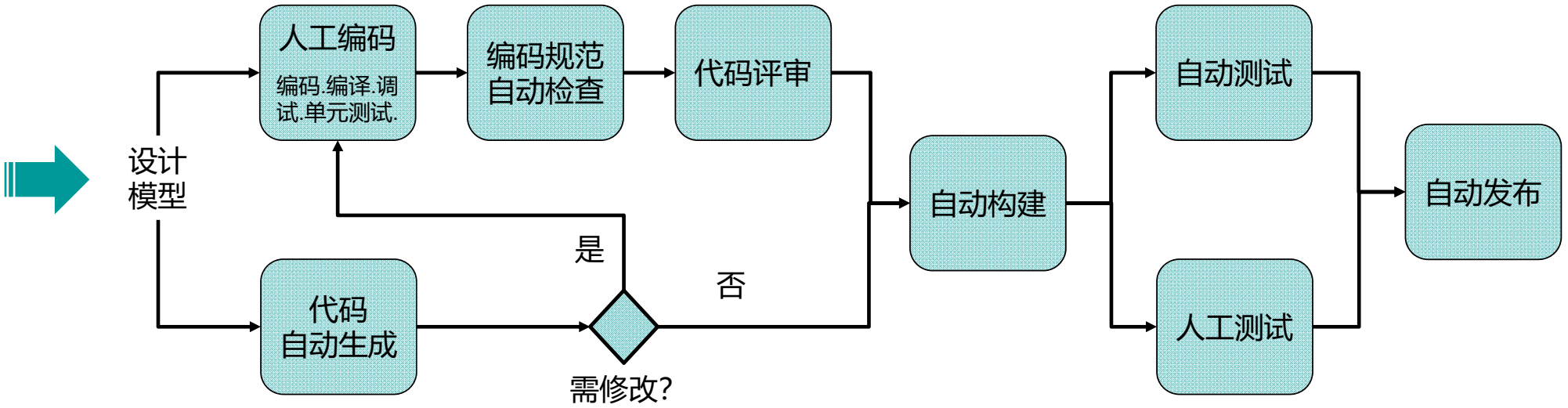
软件工程原理与实践
SOFTWARE ENGINEERING

编码和版本管理

沈备军

饮水思源 · 爱国荣校

软件设计后...



- 通过持续集成工具进行工具链集成
- 所有的模型、代码、脚本和数据由版本管理工具进行管理

从 宏观上 看百度程序员的工作

大 纲

01-编码

02-编码规范和代码评审

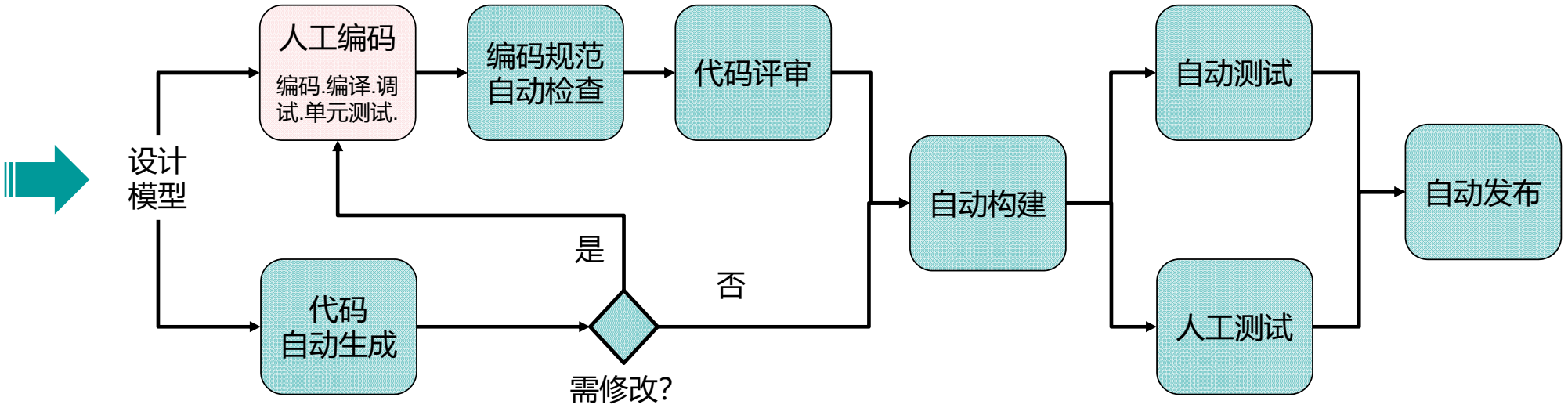
03-代码自动生成

04-软件版本管理

05-软件持续集成

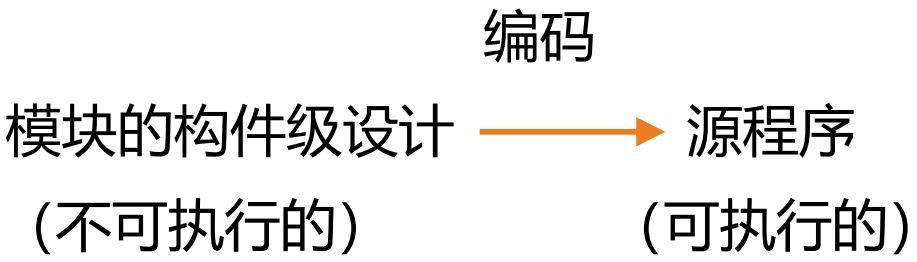
@第12.3.4节.教材

编码



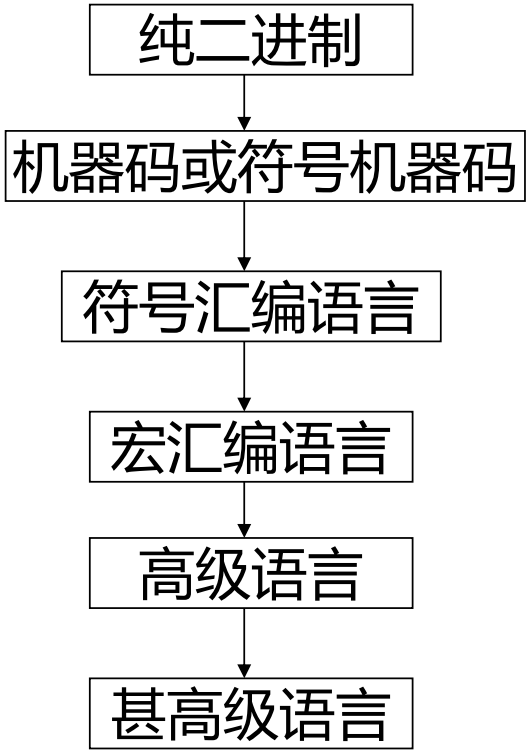
- 通过持续集成工具进行工具链集成
- 所有的模型、代码、脚本和数据由版本管理工具进行管理

编码的目的和质量要求



- 编程语言的特性和编程风格会深刻地影响软件的质量和可维护性。
- 为了保证程序编码的质量，程序员必须深刻理解、熟练掌握并正确地运用编程语言的特性。此外，还要求源程序具有良好的结构性和良好的编程风格。

计算机上语言的层次



Byte或word，指令、数据不分

用一些符号来代表指令，如sub代表减，Add代表加等，机器地址用十进制。有时汇编语言等同于符号机器码。

变量名用符号，地址也可用符号而非数字。
编制的程序称为汇编语言程序。

用户可以定义新指令及子程序

源程序编译为目标程序，或解释执行

高级程序语言的高层规约语言，提供比高级程序语言更高级的语言设施。又称为“可执行的规约语言”。有时也不区别于高级语言。

高级程序设计语言的分类

- 命令式（imperative） 语言
  - 结构化语言 C, Ada, Fortran …
  - 面向对象语言 Smalltalk, Eiffel, C++, Java …
  - 脚本语言 Perl, Python, PHP…
- 说明式（declarative）语言
  - 函数式 Lisp/Scheme, ML, Haskell, Clean, Erlang, Miranda…
  - 数据流 Id, Val …
  - 逻辑式 或基于约束的 Prolog, spreedsheets …
  - 基于模板的 XSLT …
- 量子（quantum）编程语言
  - Q#, Quipper, Sliq

语言的选择

- 选择编码语言的标准
  - 应用领域
  - 算法、计算、数据结构的复杂性
  - 运行效率、开发效率、可移植性、安全性等考虑
  - 库
  - 生态、大厂的支持

TIOBE Programming Community Index

（http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html）

语言选择举例

- 如果编写对性能要求苛刻，或和操作系统结合紧密的程序，必然选择c。
- 如果需要跨平台，又要广泛的支持的话，选择java。
- 编写服务器端程序，用Java 、JavaScript 、php、perl、python、asp。
- 编写客户端程序，最常用的是JavaScript。
- 编写机器学习软件采用python、R、Java
- 编写数据库程序最简单的语言是vb或delphi。
- 编写知识的处理程序用prolog。
- 编写最灵活，最模糊的程序用lisp。
- 编写office程序用vba。
- 如果只作为简单应用的工具语言，python和ruby是更好的选择，他们的跨平台移植性好，应用也比较广泛，语言简单，生产率高。

编码准则

- Follow structured programming practice.
- Select data structures that will meet the needs of the design.
- Create interfaces consistent with the software architecture.
- Keep conditional logic as simple as possible.
- Create nested loops in a way that makes them easily testable.
- Select meaningful variable names and follow other local coding standards.
- Write code that is self-documenting.
- Create a visual layout that aids understanding.

编码的风格

- 追求“聪明”和“技巧” → 提倡“简明”和“直接”
- 使用标准的控制结构
- 清晰的前提下求取效率
  - Make it right before you make it faster.
  - Make it clear before you make it faster.
  - Keep it right when you make it faster.

    (求快不忘保持程序正确)
  - Keep it simple to make it faster.

    (保持程序简单以求快)
  - Don’t sacrifice clarity for “efficiency”.

    (书写清楚，不要为“效率”牺牲清楚)

源程序的文档化 (code documentation)

- 有意义的变量名称
- 适当的注释
- 标准的书写格式
  - 用分层缩进的写法显示嵌套结构的层次；
  - 在注释段与程序段、以及不同程序段之间插入空行；
  - 每行只写一条语句；
  - 书写表达式时，适当使用空格或圆括号等作隔离符；

好的源程序本身就是详细设计文档！

大 纲

01-编码

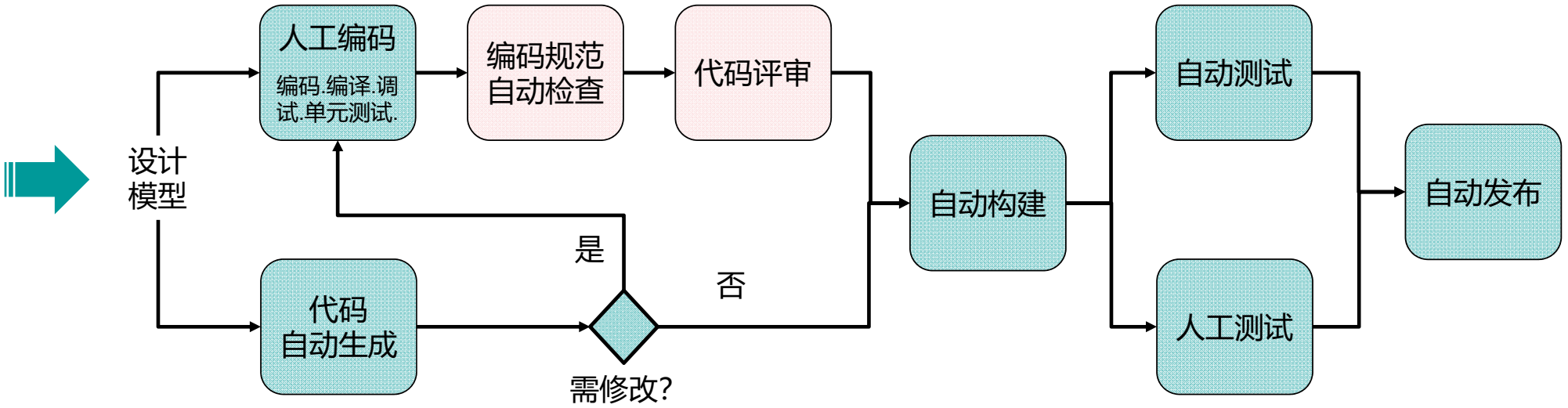
02-编码规范和代码评审

03-代码自动生成

04-软件版本管理

05-软件持续集成

编码规范和代码评审



- 通过持续集成工具进行工具链集成
- 所有的模型、代码、脚本和数据由版本管理工具进行管理

编码规范

- 目的：编写出简洁、可维护、可靠、可测试、高效、可移植的代码
- Google编码规范

  英文：https://github.com/google/styleguide

  中文：https://github.com/zh-google-styleguide/zh-google-styleguide
- 编码规范检查工具
  - FindBugs
  - PMD
  - CheckStyle
  - SonarQube
  - .....

示例：Google C++ 编程规范

- 编程规范的大纲

1. 头文件
2. 作用域
3. 类
4. 函数
5. 来自 Google 的奇技
6. 其他 C++ 特性
7. 命名约定
8. 注释
9. 格式

- 编程规范规则的例子

- 除少数特定环境外, 不要重载运算符. 也不要创建用户定义字面量.
- 将数据成员声明为 private, 除非是 static const 类型成员.
- 函数体尽量短小, 紧凑, 功能单一.
- 所有按引用传递的参数必须加上 const.
- 函数命名, 变量命名, 文件命名要有描述性; 少用缩写.
- 每个类的定义都要附带一份注释, 描述类的功能和用法.

代码评审(code review)

- 所有代码在进入代码主库之前，都必须要经过评审。
- 代码评审员从以下方面评价代码的质量：设计、功能、复杂性、测试、命名、评价质量和代码风格等。
- 工具举例
  - Gerrit 是一个免费、开放源代码的代码评审工具，出自google团队。它利用网页浏览器，同一个团队的软件程序员，可以相互审阅彼此修改后的程序代码，决定是否能够提交，退回或者继续修改。Gerrit 是使用 Git 作为底层版本控制系统，通过网页界面，能方便地进行代码评审的一个轻量型框架。

大 纲

01-编码

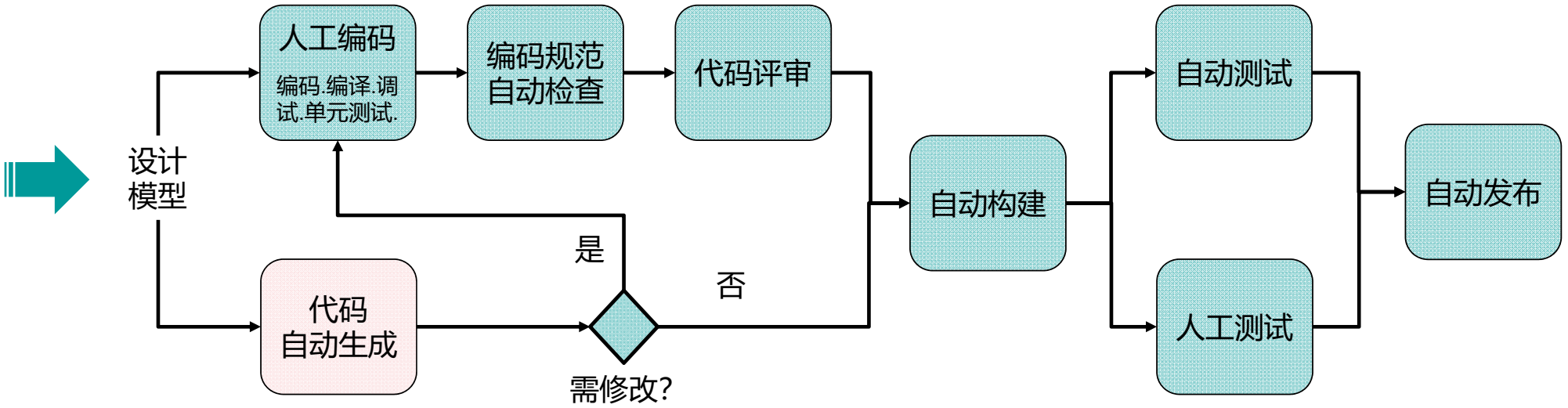
02-编码规范和代码评审

03-代码自动生成

04-软件版本管理

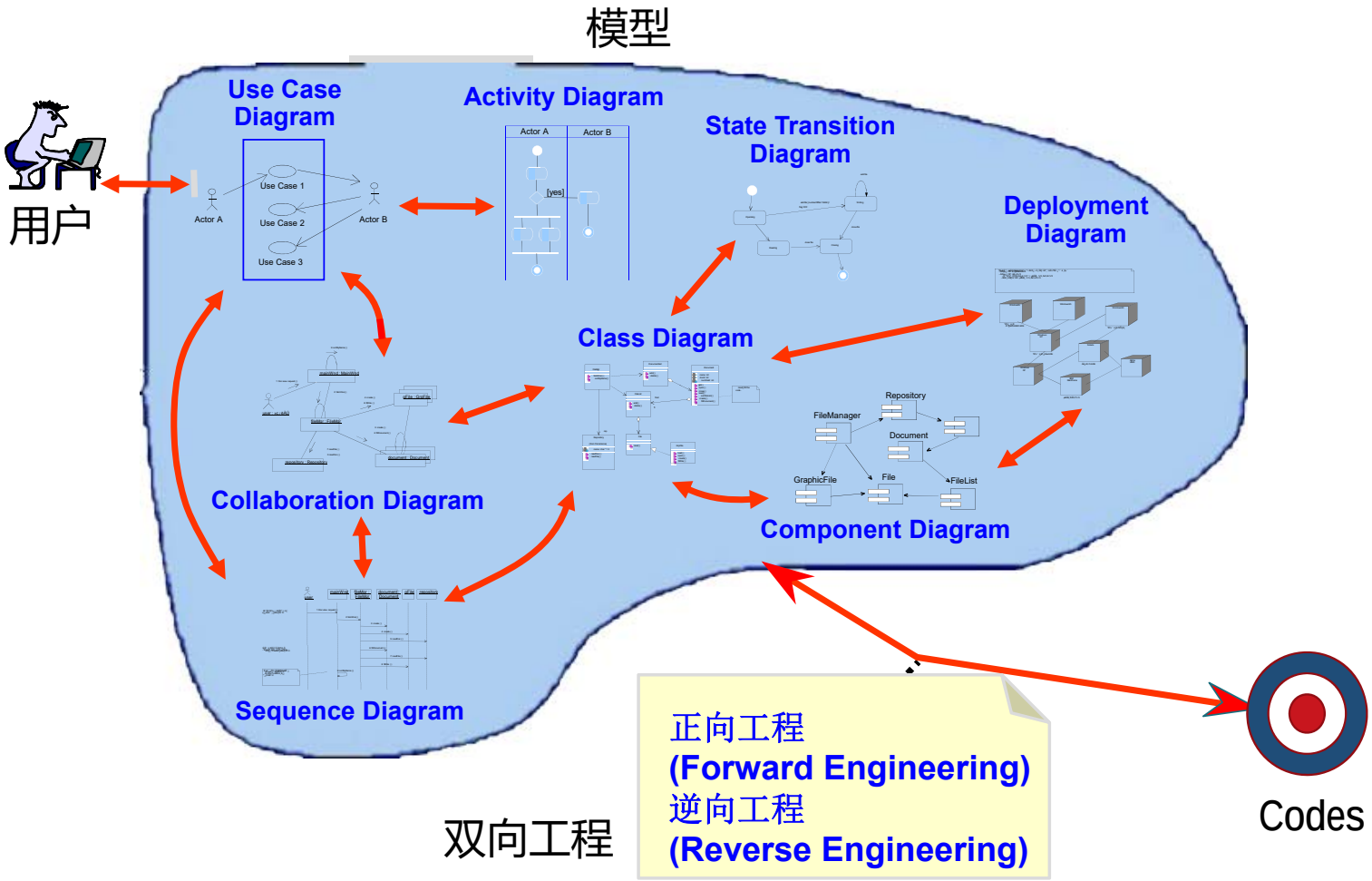
05-软件持续集成

代码自动生成



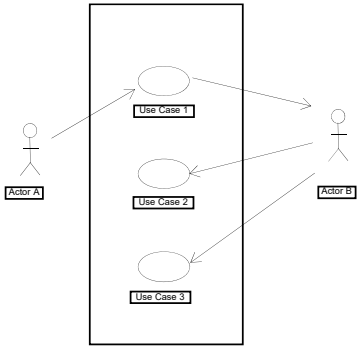
- 通过持续集成工具进行工具链集成
- 所有的模型、代码、脚本和数据由版本管理工具进行管理

从模型中生成代码


模型
用户
Use Case Diagram
Activity Diagram
State Transition Diagram
Deployment Diagram
Class Diagram
Collaboration Diagram
Component Diagram
Sequence Diagram
Actor A
Actor B
Use Case 1
Use Case 2
Use Case 3
[yes]
Repository
FileManager
Document
GraphicFile
File
FileList
双向工程
正向工程
(Forward Engineering)
逆向工程
(Reverse Engineering)
Codes

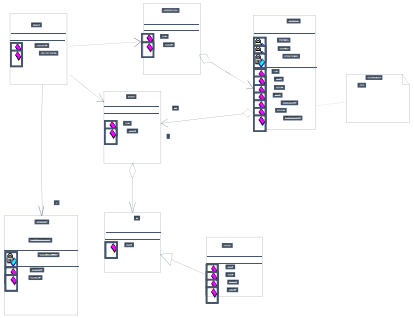
确保模型和代码的一致

需求



一致

设计

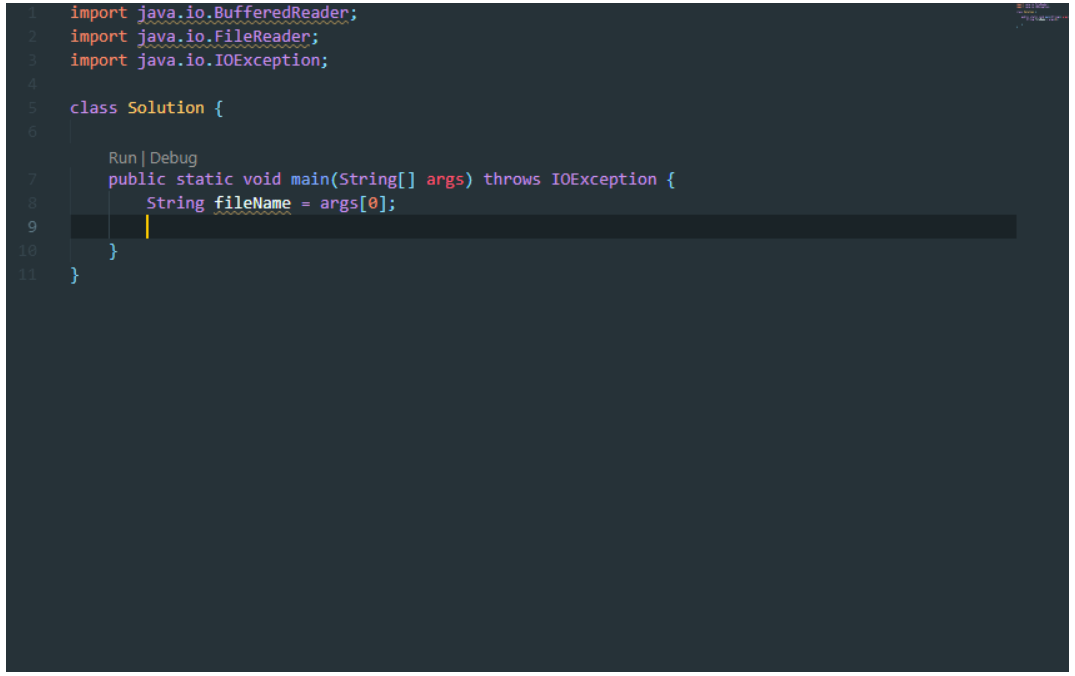


一致

代码

其他的代码自动生成技术

- Text2Code
- UI2Code
- Code2Code
- Sample2Code
- ……



大 纲

01-编码

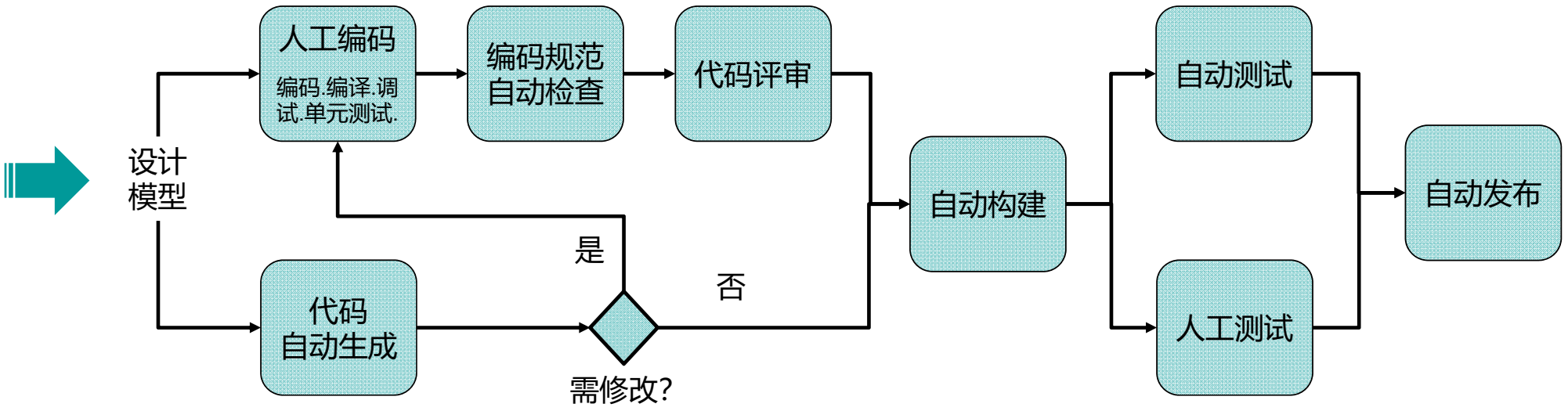
02-编码规范和代码评审

03-代码自动生成

04-软件版本管理

05-软件持续集成

软件版本管理



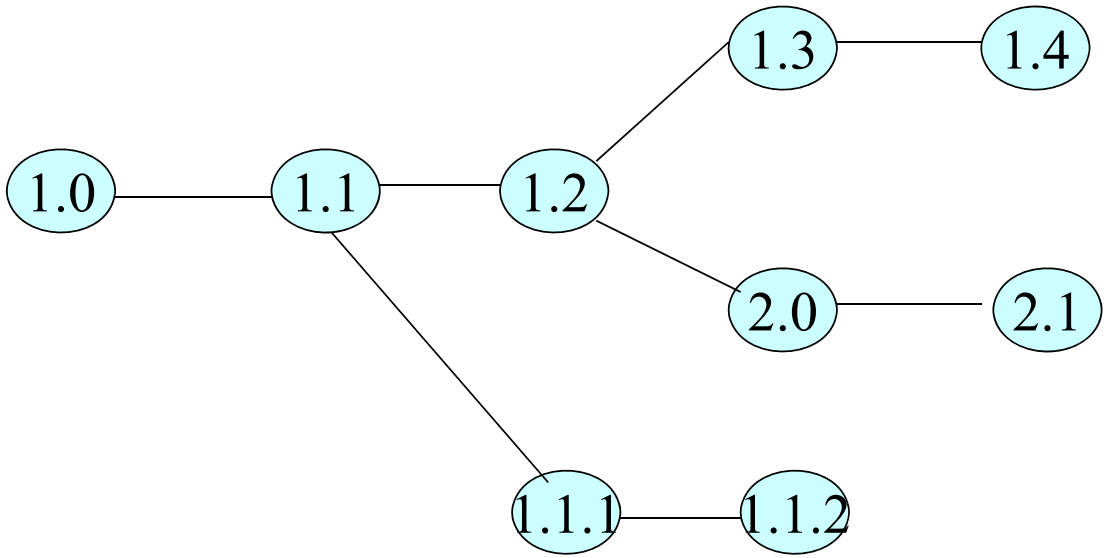
所有的模型、代码、脚本和数据由版本管理工具进行管理

软件配置项

- 软件版本管理的基本实体是软件配置项 (Software Configuration Item, SCI)。
- 一个软件配置项是在软件生存周期所产生或使用的一个工作产品或一组相关的工作产品，包括代码、文档、模型、数据等。

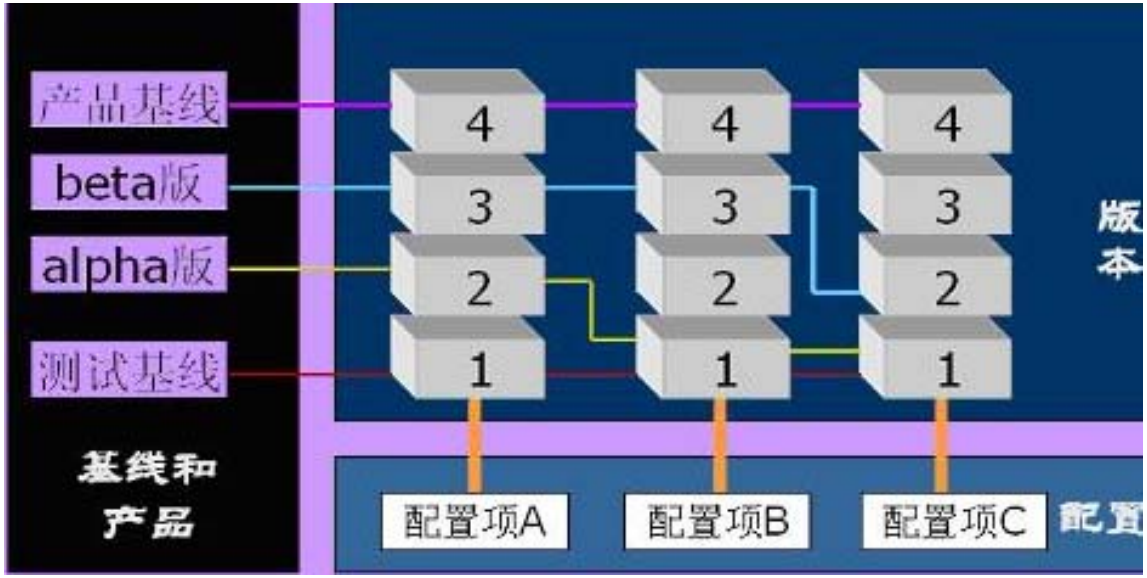
版本 version

- 配置项在软件开发过程中会不断变更，形成多个版本(Version)。
- 每个配置项的版本演化历史可以形象地表示为图形化的版本树(Version Tree)。



基线 baseline

- 基线是项目储存库中每个配置项版本在特定时期的一个“快照”，它提供一个正式标准，随后的工作基于此标准，并且只有经过授权后才能变更这个标准。
- 通常将交付给客户的基线称为一个“发布”（Release），为内部开发用的基线则称为一个“构建”（Build）。

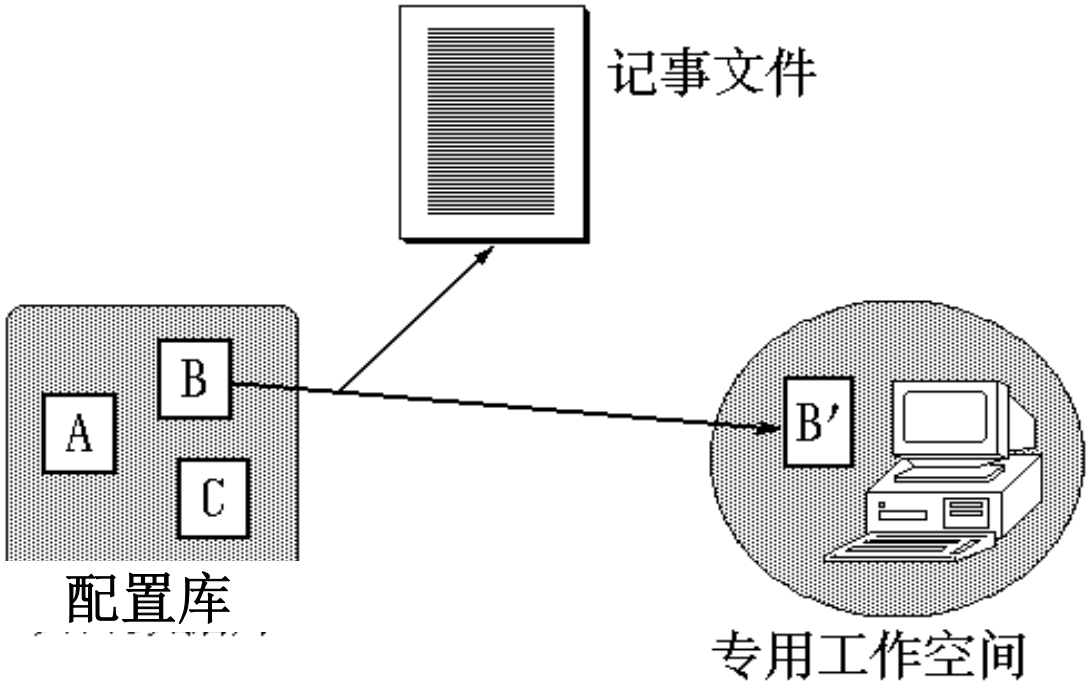


版本控制

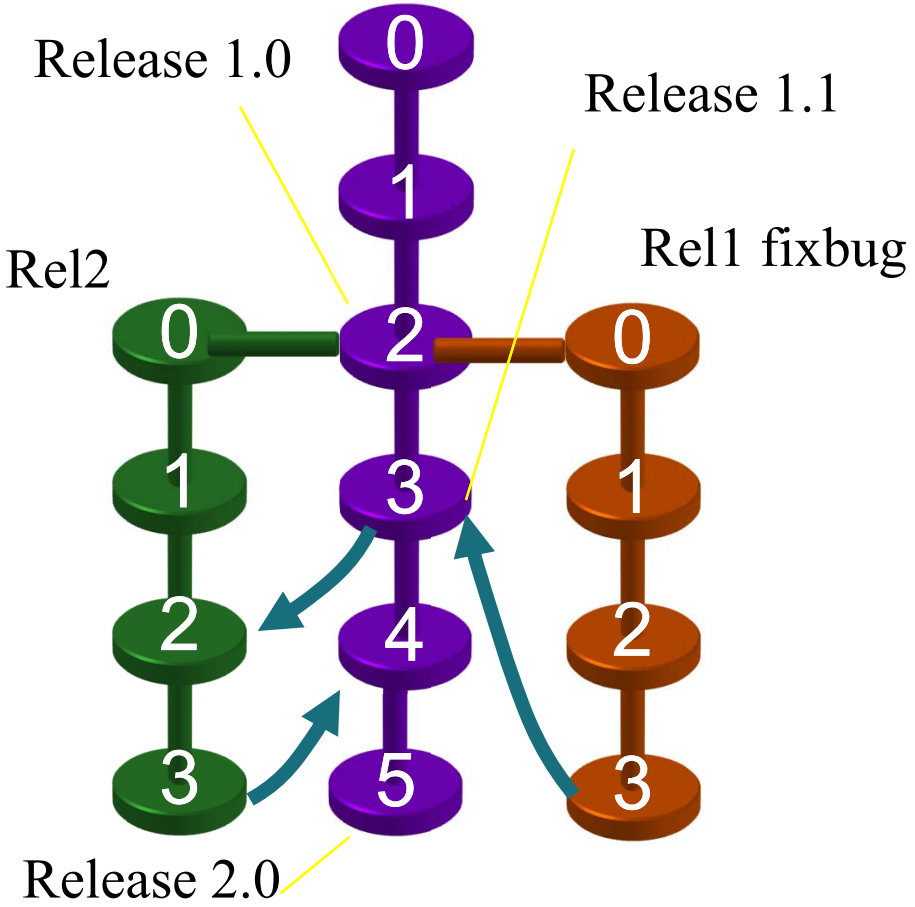
- 版本管理记录每个配置项的变更履历，并控制基线的生成。
- 目的：
  - 对软件开发过程中配置项的各个版本提供有效的追踪手段，保证在需要时可回到旧版本，避免文件丢失和相互覆盖；
  - 通过对版本库的访问控制避免未经授权的访问和修改，达到有效保护软件资产和知识产权的目的；
  - 实现团队并行开发、提高开发效率。

配置库CM Repository

- 存储配置管理信息及软件软件配置项的版本信息。

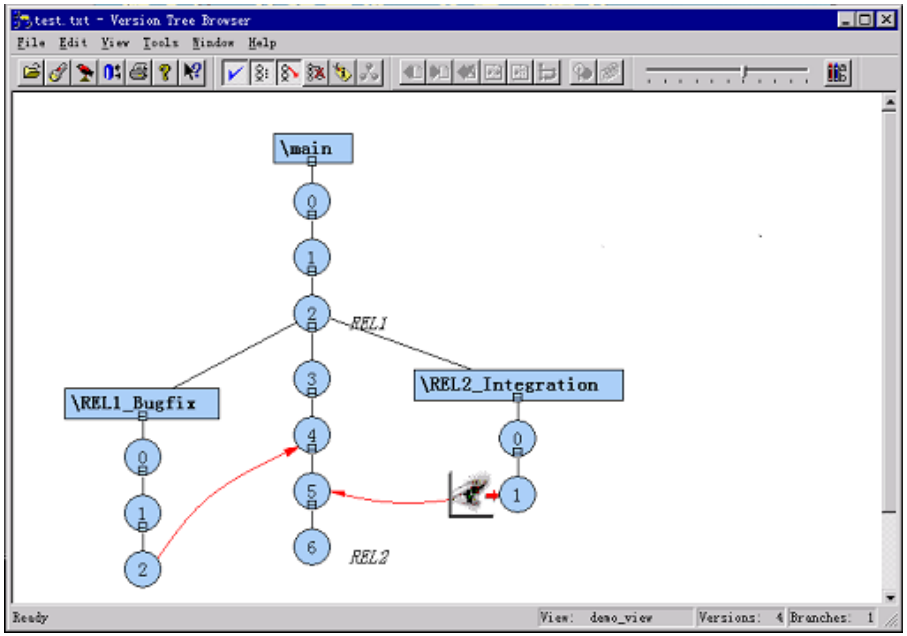


并行开发


Release 1.0
Release 1.1
Rel2
Rel1 fixbug
Release 2.0
0
1
2
3
4
5
0
1
2
3
0
1
2
3

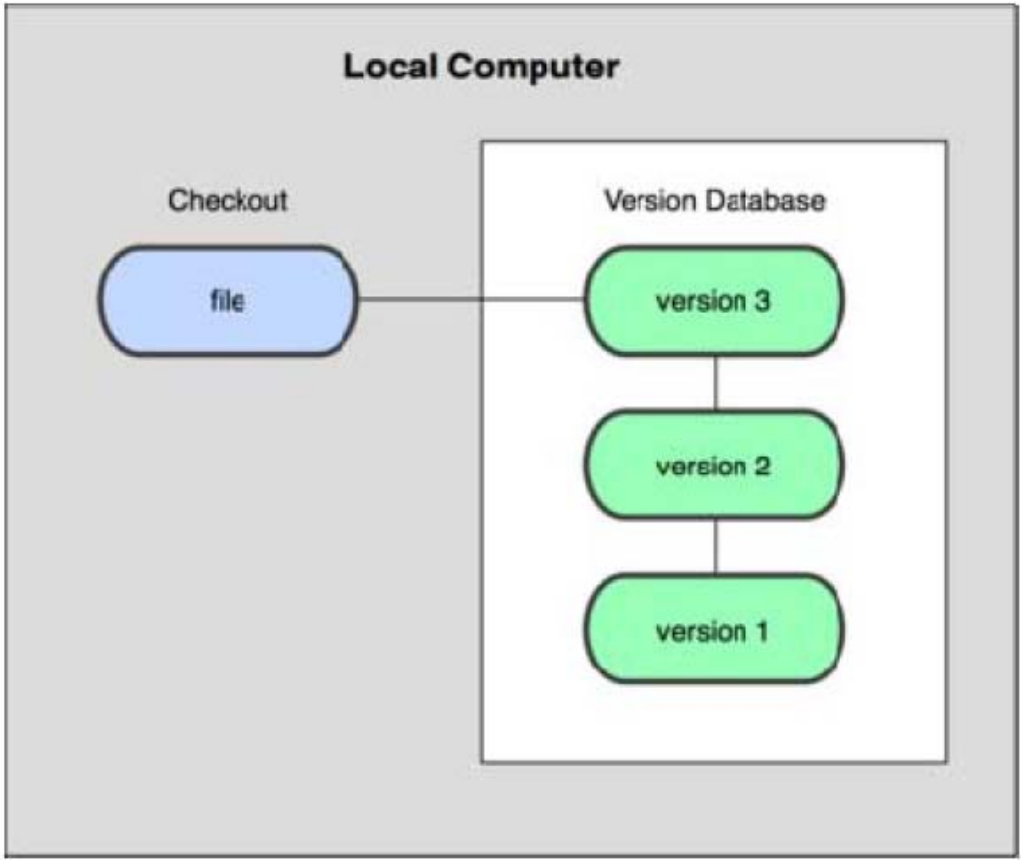
版本控制工具

- Merant PVCS
- IBM Rational Clearcase
- Microsoft Visual Source Safe
- CVS (Freeware)
- SVN (Freeware)
- Git (Freeware)
- GitHub 和 Gitee (云平台)
- …



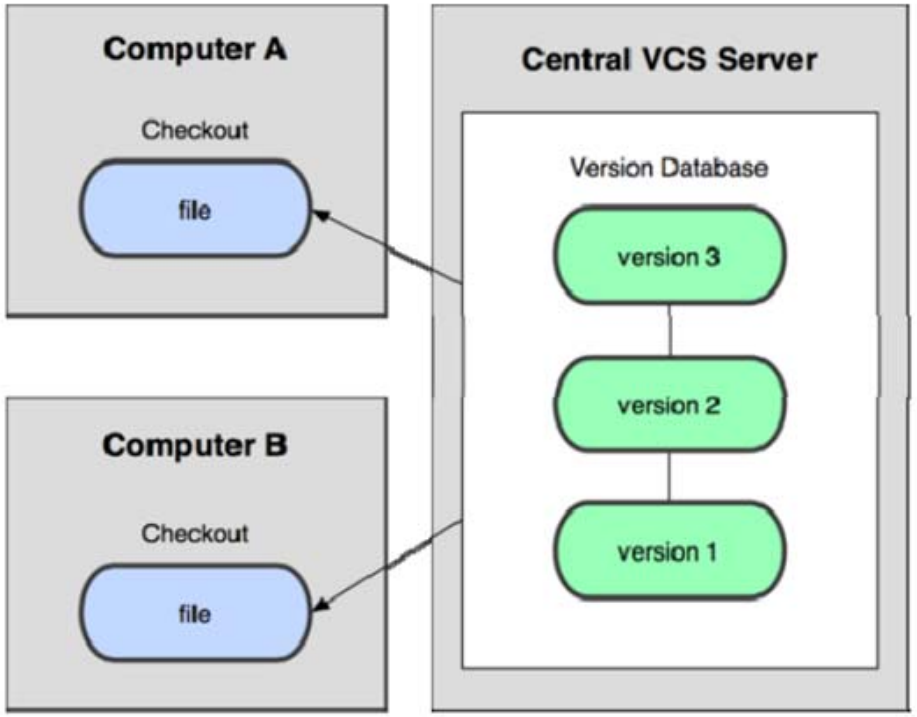
本地版本控制系统

- 适合于单人开发



集中化的版本控制系统

- 适合于局域网小团队开发
- CVS, Subversion以及Perforce等，有一个单一的集中管理的服务器，保存所有文件的修订版本，而协同工作的开发者通过客户端连到这台服务器，取出最新的文件或者提交更新。



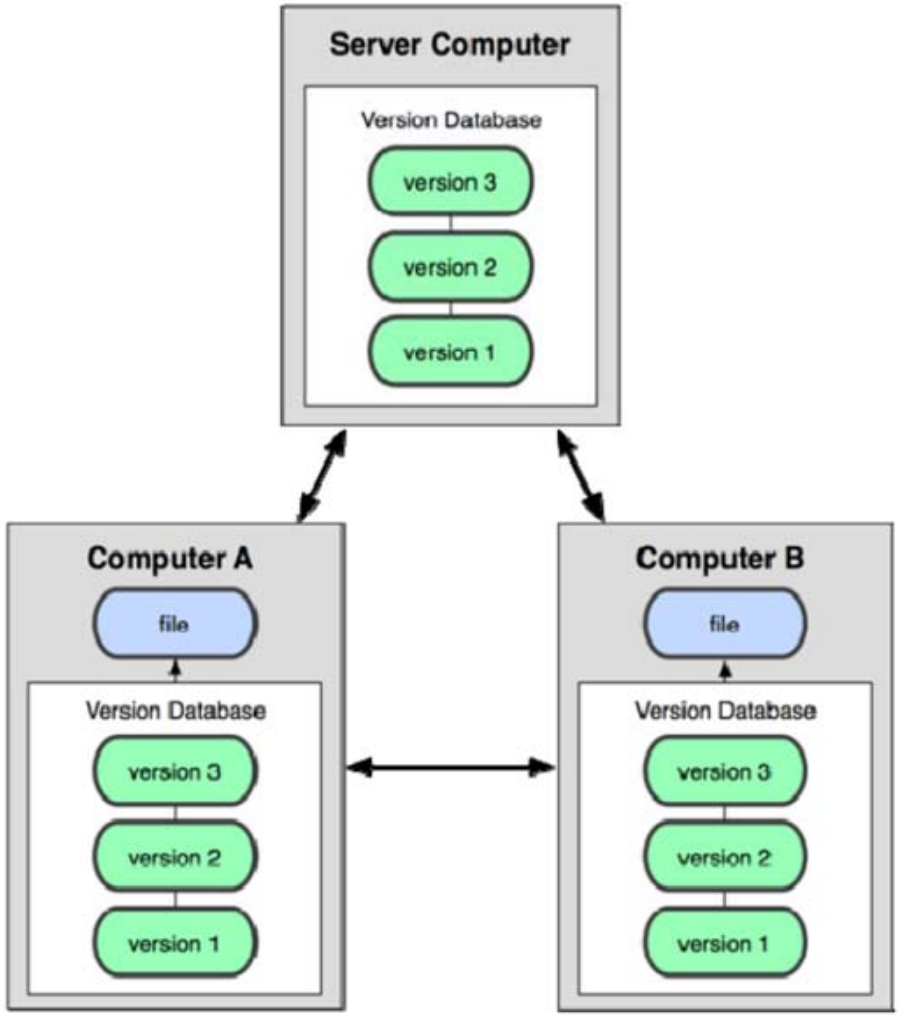
分布式版本控制系统

- 适合于任何规模的团队开发

  支持Internet或局域网等各种网络

  - 客户端并不只提取最新版本的文件快照，而是把原始的代码仓库完整地镜像下来。
  - 任何一处协同工作用的服务器发生故障，事后都可以用任何一个镜像出来的本地仓库恢复。
  - 每一次的提取操作，实际上都是一次对代码仓库的完整备份。

- 当前最流行的版本控制系统 -- Git



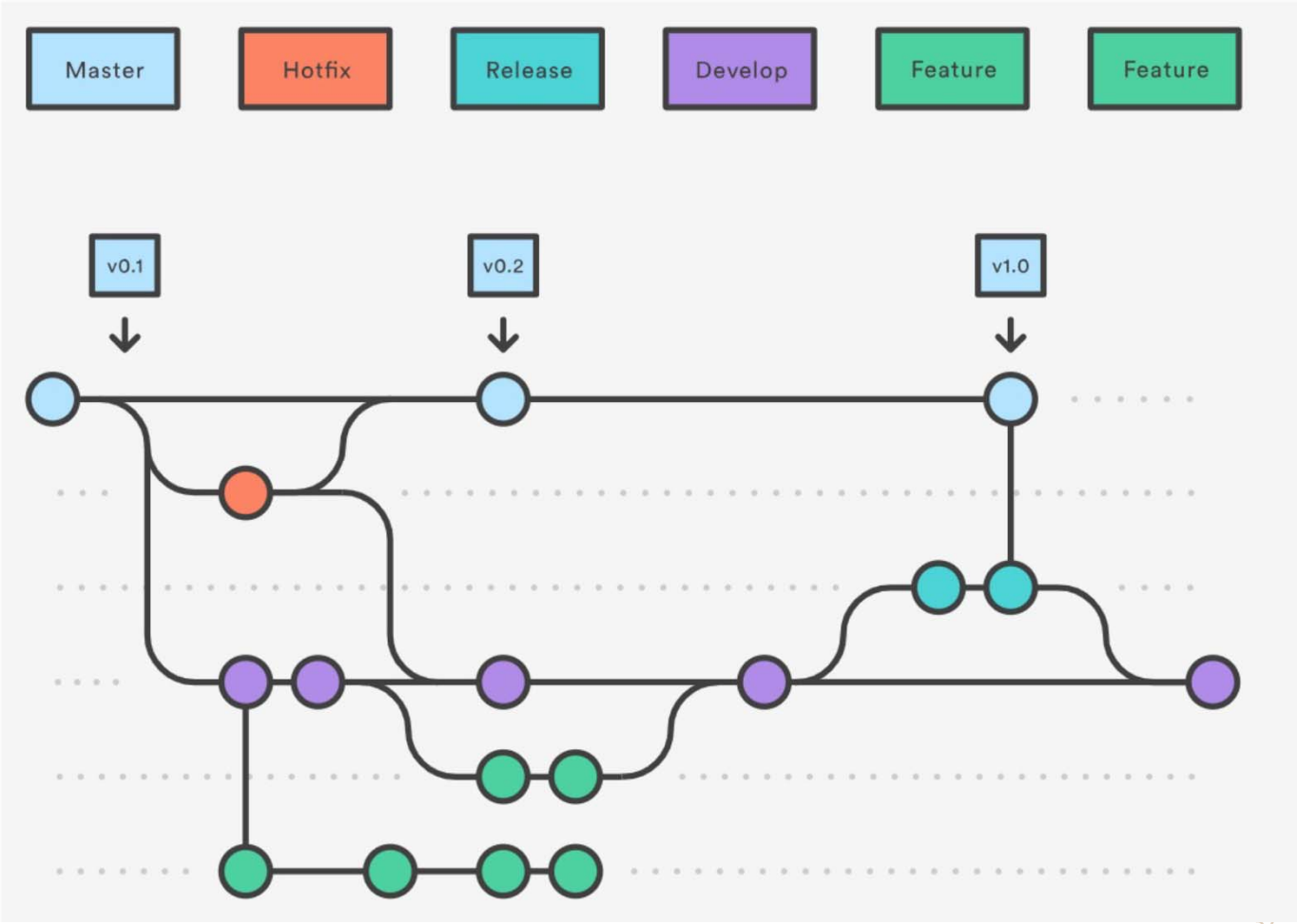
Git 多人协同工作流1：Github Flow

- 主分支：Master分支
- 短分支：开发新功能或修复缺陷时，创建一个新分支，完成后通过Pull Request (PR) 合并至Master，删除分支

GitHub flow Docs

Git 多人协同工作流2：Gitflow Workflow

- 两个主分支：
  - Master分支保存官方的发布历史
  - Develop分支用于集成各种功能开发的分支
- 在创建新的功能开发分支时，父分支应该选择Develop



Gitflow Workflow Tutorial

大 纲

01-编码

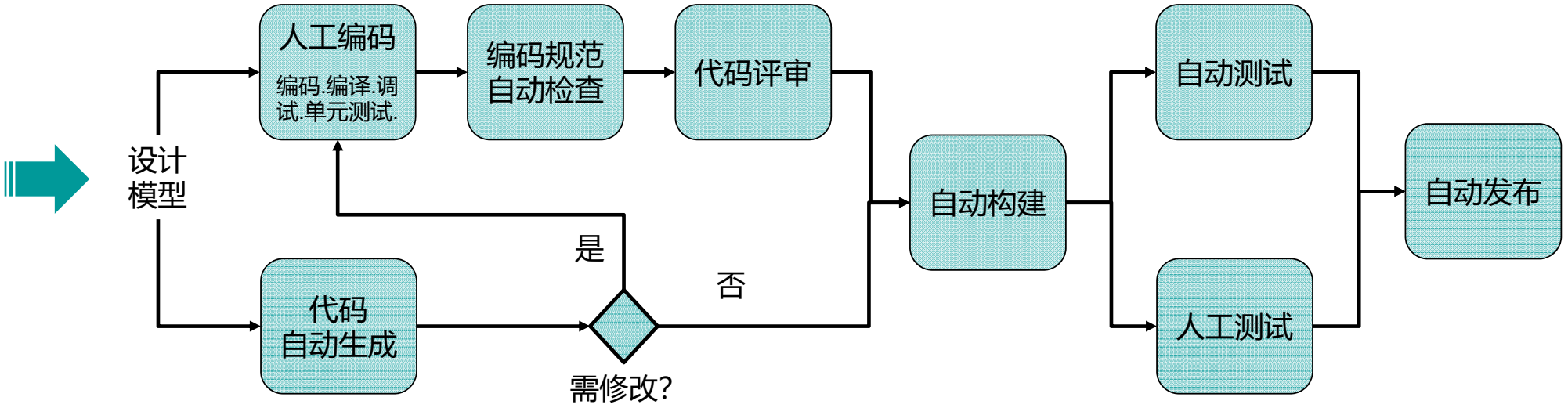
02-编码规范和代码评审

03-代码自动生成

04-软件版本管理

05-软件持续集成

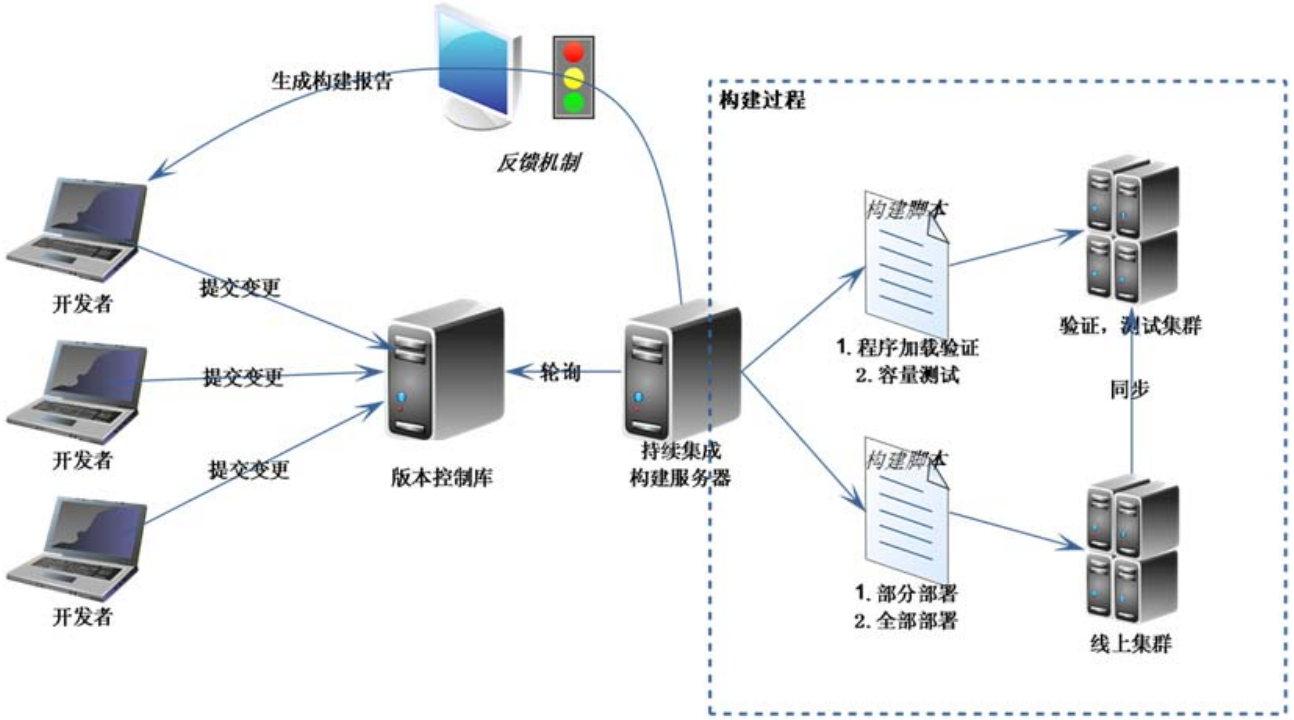
软件持续集成



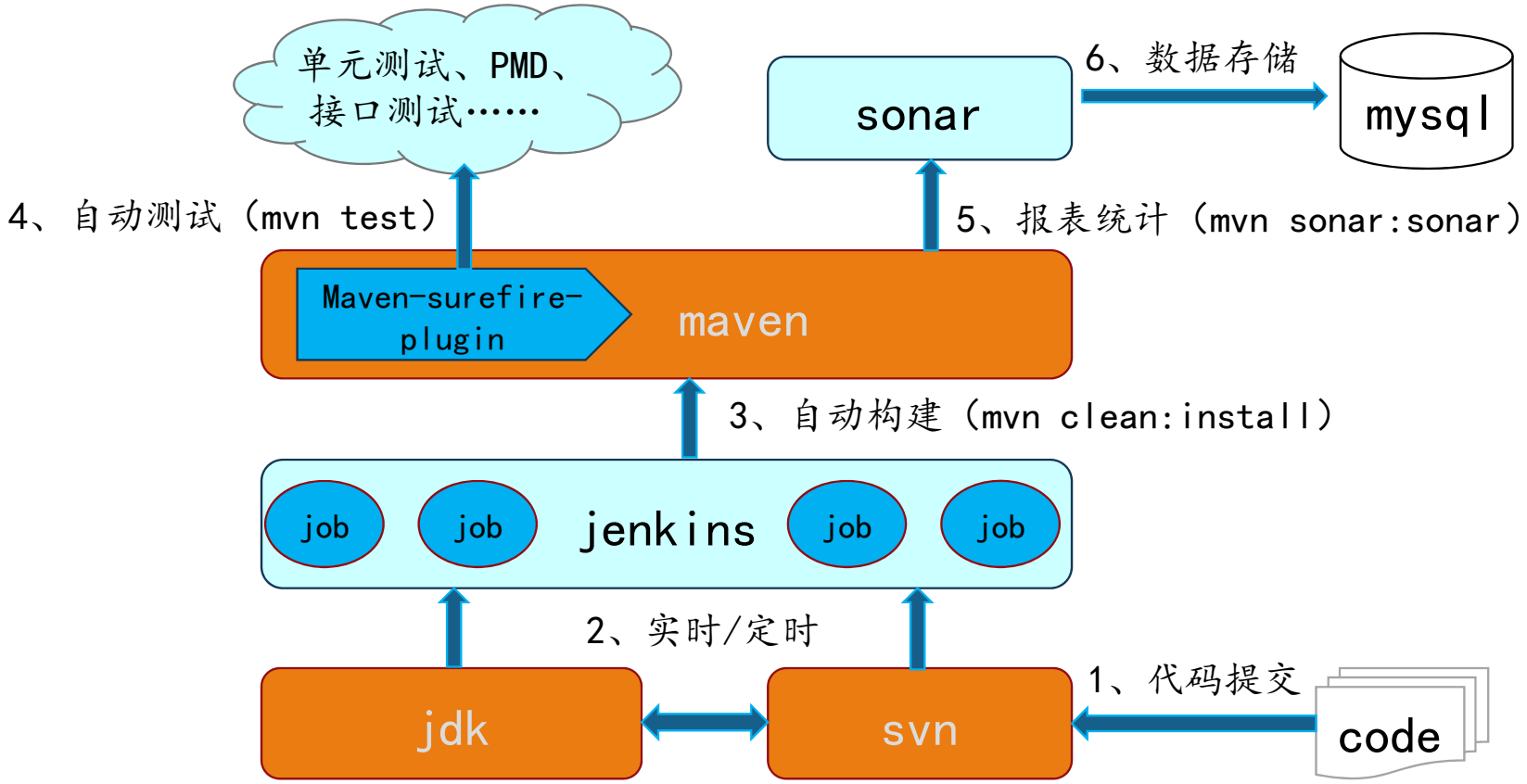
通过持续集成工具进行工具链集成

软件持续集成

- 持续集成是一种软件开发实践，即团队开发成员经常集成它们的工作，通常每个成员每天至少集成一次，也就意味着每天可能会发生多次集成。
- 每次集成都通过自动化的构建（包括编译，发布，自动化测试)来验证，从而尽快地发现集成错误。



持续集成示例


单元测试、PMD、
接口测试……
sonar
6、数据存储
mysql
4、自动测试（mvn test）
5、报表统计（mvn sonar:sonar）
Maven-surefire-plugin
maven
3、自动构建（mvn clean:install）
job
job
jenkins
job
job
2、实时/定时
1、代码提交
jdk
svn
code