匡信科技日常开发支持环境

# 目录

[匡信科技日常开发支持环境 1](#_Toc398652914)

[支持环境综述 2](#_Toc398652915)

[代码仓库 2](#_Toc398652916)

[产品仓库 2](#_Toc398652917)

[事务跟踪系统 2](#_Toc398652918)

# 综述

在日常开发的流程管理中，需要多个支持环境协作管理流程中的各个节点和相关物件。各支持环境与开发流程的协同和交互详见[《匡信科技开发流程规范》](匡信科技日常开发流程规范.docx)。

## 代码仓库

负责管理所有源代码和原始资源的存储、共享、版本管理、分支管理、访问权限控制。当前使用subversion作为代码仓库，可以通过命令行、客户端、集成开发环境（IDE）、浏览器访问。

## 产品仓库

负责管理所有编译构建产出物的存储、共享、版本管理、权限控制。当前使用基于Sonatype Nexus的Maven仓库，可以通过命令行、集成开发环境（IDE）、浏览器访问。

## 事务跟踪系统

负责管理完整开发生命中期中所有事务流程、节点、时间、成本。当前使用JIRA系统管理各类事务，包括任务、缺陷、需求、支持等。

# 代码仓库

代码仓库以subversion服务形式提供，建立在subversion 1.8版本基础上，提供svn://协议与https://协议访问。

由代码仓库管理的内容是源代码、文档以及其他原始媒质和资料。所有经过构建处理的内容（如编译得到的二进制文件、打包得到的产品发布包等）均不应通过代码仓库管理。

## 组织和结构

### 仓库（repository）

仓库是代码仓库的顶级管理单元，对应开发中的一个独立发布、运行、使用和维护的产品，完成一套完整的业务逻辑组合。由于subversion的版本号（revision/rev.）基于仓库粒度，亦即，每个仓库有一个全局版本号，建议每个产品占据单个仓库，以便版本号更客观的反应产品代码版本变更。

每个仓库由subversion服务url下的一级路径组成，路径名为产品名称或名称缩写，例如：

xxxx产品的代码仓库：https://svn.hzcominfo.com/xxxx/

### 分支目录（folder）

每一个产品代码仓库下有标准化的顶级子目录结构：

* trunk

主开发分支，容纳当前开发的版本所有组件（component）代码。其子目录为组成产品的各个组件名称或简称。

* branches

并行开发分支，例如，产品某个版本的演示版本与开发版本将会并行开发、发布、运行。

并行开发分支可能被合并回到主开发分支或其源开发分支，从而结束其生命周期。此后该分支可能被删除或存档。

branches目录内应平坦的定义各组件并行开发分支，以“组件名称+版本号+并行任务描述后缀”形式命名。

* tags

版本发布/备份标签分支，供产品开发过程中备份使用，如版本发布后的备份。

tags目录内所有代码均应只读，不应有任何变更和提交。如需要对被备份的代码做变更，应从标签中建立并行开发分支，在其中进行开发。开发完成后建立新的标签。

tags目录内应平坦的定义各组件备份标签，以“组件名称+版本号+标签描述后缀”形式命名。

* doc：产品文档（包括无需编译构建的内容，例如，数据库创建和测试数据生成脚本）。

### 项目目录（project）

每一产品的主开发分支中，应根据产品设计定义多个项目目录，每个项目对应开发环境中的一个项目，同时对应产品设计中的一个组件，亦即：一个独立运行的程序。

除设计中的各组件项目外，开发分支中应包含测试用例分支，对应该项目的所有测试用例实现。

测试用例分支名称为testcase（/trunk/testcase）。

## 权限管理

下表描述了一个产品代码仓库权限管理矩阵，其中所有路径均相对于项目根目录（svn/xxxx）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 角色 | 项目决策 | 项目经理 | 开发人员 | 测试人员 | 部署人员 |
| / | R/W | R/W | N | N | N |
| /trunk | N | N |
| /trunk/testcase | R | RW |
| /trunk/component1 | R（根据任务分配） | N |
| /trunk/component2 |
| /branches | N |
| /branches/component1-branch1 | R（根据任务分配） |
| /branches/component2-branch1 |
| /tags | N |
| /tags/component1-tag1 |
| /tags/component2-tag2 |
| /doc | RW | RW |

## 使用规范

### 命名规范

#### 项目名称

项目名称应由可读的、意义能描述项目功能的英语单词组成且尽量简短。

在代码仓库中组成项目名称的多个单词应全部**小写**，以短横线（减号）分隔，以符合url规范。

如果项目名称过长，可以以首字母缩写全部或者部分单词。若仅缩写部分单词，则缩写部分和其他单词同样以减号分隔。

#### 版本号

项目版本号遵循产品仓库（MAVEN）的推荐版本号规范，以圆点（小数点）分隔的三部分组成。三部分均由数字组成，分别是：

主版本号（Major #）：标识产品功能定义。

次版本号（Minor #）：标识产品功能升级和扩展。

开发版本号（Dev #）：标识一个完整的开发过程。

#### 并行分支描述/标签描述

描述作为并行分支或标签中子目录命名后缀使用，由可读的、意义能描述并行开发目的或标签目的的英语单词组成且尽量简短。可以使用首字母缩写。命名中各个部分应全部**大写**，以减号分隔。

### 提交规范

1. 每次提交均应包含提交注释（comment）。建议以简短的英语描述（因为svn对多字节码注释可能产生乱码问题）。
2. 每次提交所包含的代码变更应保证不影响整个产品中所有项目的成功编译、构建和当前运行正常的业务。亦即，不影响当前运行正常的测试用例。
3. 每次提交所包含的所有变更应该是“完整的”。亦即，能完整实现提交注释中描述的提交目的。
4. 提交粒度应符合任务粒度。亦即，理想状态下一次提交应对应一个任务（如功能点实现、缺陷修复等），或该任务可以恰当切分和描述的一个子任务。
5. 在流程管理系统中变更任务状态时，应附带完成该任务过程中所有代码提交变更集编号。

# 产品仓库

代码仓库以maven仓库服务形式提供，建立在sonatype nexus 1.8版本基础上，提供http://协议访问。

产品仓库管理的是所有经过编译、构建、打包产生的产出物。包括二进制编译产出、产品发布包等。

## 组织和结构

## 权限管理

## 使用规范

# 事务跟踪系统

使用JIRA问题跟踪系统管理开发流程中的相关事务。具体机制、规划可以参考[《匡信科技日常开发流程规范》](匡信科技日常开发流程规范.docx)。