版本管理规范

V1.0 (草案)

修订记录：

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 修订时间 | 修订人 | 修订内容 |
| 1.0 | 2011/4/25 | 苑旭 | 起草 |

目录

[引言 2](#_Toc291487464)

[开发库规范 2](#_Toc291487465)

[推荐产品 4](#_Toc291487466)

[参考文献 5](#_Toc291487467)

# 引言

**文档目的**

本文档规定项目团队采用Git作为版本控制工具的使用规范。如项目团队采用SVN作为版本控制工具，请遵照OTPQ-PM-2b 版本管理规范(SVN)的规范执行。

**规范适用**

虽然本规范以产品开发团队为主要对象，同样适用于其他团队(如运营、个人)的版本管理。

**规范的表示方法**

我们通过**要**、**考虑**、**避免**、**不要**这些词把规范组织成一条条简单的建议。每条规范都描述了一种好的或者不好的做法，并用统一的方式来表示。对于好的做法，在其前面会用🗸表示，与此对应，不好的做法则用🗴表示。每一条规范的措辞也会明确表示出这条规范的重要性。

“**要…**” 描述的是必须遵循的规范。(必须这一措辞有点太强。虽然有些规范我们的确应该始终遵守。另一方面，在一些非常特殊的情况下，你可能需要违反一些“要…”规范。)

“**考虑…**” 描述的是在一般情况下应该遵守的规范，但如果完全理解规范背后的道理，并有很好的理由不遵守他时，也不要畏惧打破常规。

“**不要…**” 描述的是一些几乎绝对不应该违反的规范。

“**避免…**” 就没有那么绝对，他描述的做法虽然通常并不好，但却存在一些一直的可以违反该规范的情况。

**术语表**

* 产品库

指PMO存档开发团队最终产品和历次升级、BUG修复维护的版本库。

* 开发库

指开发团队为了在开发期间所使用的版本库。

# 开发库规范

**🗸要** 采用以下示例组织版本库结构

**🗸要** 用以下规则命名分支和标签

|  |  |  |
| --- | --- | --- |
|  | 命名风格 | 示例 |
| 发布分支 | RB-rel | RB-1.0  RB-1.0.a |
| 发布 | REL-rel | REL-1.0  REL-1.0.a |
| Bug修复分支 | BUG-track | BUG-3315  BUG-1087 |
| Pre-bug fix | PRE-track | PRE-3315  PRE-1087 |
| Post-bug fix | POST-track | POST-3315  POST-1087 |
| 实验分支 | TRY-initials-desc | TRY-MGM-cache-pages  TRY-MR-neo-persistence |

**🗸要** 在每次对外发布时创建一个发布标签

**🗸要** 在采用迭代/增量交付的生命周期的项目中，为每个预发布版本(Beta)创建一个发布分支，对此版本的所有修复工作均在此发布分支完成，新版本/增量的开发在主分支进行；发布分支正式发布时应将所有Bug修复合并到主分支，并创建发布标签且删除发布分支。

**🗸要** 为项目周期中的重要事件建立标签

**🗸要** 每个小团队均应有自己的主工作分支(Git中的一个分支或master)，只有一个团队的项目可以使用master作为主公做分支。

**🗸要** 为每个特性创建一个分支，测试通过后合并到主分支。

**🗸要** 在每次提交时编写备注，以反映提交的内容和/或目的

**🗸考虑** 为复杂的Bug修订创建一个分支，并创建一个修复前和一个修复后的标签。修复完毕后合并到主分支和其他有同样Bug的发布分支。

**🗸考虑** 保持小增量式的提交，尽量在完成某些特性后尽快提交，而不是等到某个固定时间点(如下班)一次性提交

**🗸考虑** 为文档创建一个单独的doc分支

**🗴避免**在分支上创建分支

**🗴避免** 创建过多的分支

**🗴避免** 在标签上修改，而应保持标签的只读性

# 推荐产品

MSysGit

TortoiseGit(可选)

GitSccProvider(Visual Studio插件)(可选)

EGit(Eclipse插件)(可选)

# 参考文献

版本控制之道——使用Git

.NET 设计规范——.NET约定、惯用法与模式 人民邮电出版社 2006.7