**源代码管理约定**

为推动敏捷S实践，基于现有项目特点，现定义源代码管理约定。本着敏捷迭代的思想，本约定先试用，后续再依于实践反馈再持续改进和优化。

1. **基本约定**

* 使用git作为源代码管理软件。
* 在特性分支上开发和测试。
* 在特性分支完成测试后再合并至主干分支，确保主干分支代码的稳定性。
* 发布时先更新数据库，再更新应用程序。

1. **环境约定**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **环境名称** | **英文名称**  **(统一小写)** | **使用者** | **维护者** | **数量** | **用途** |
| 生产环境 | **prod** | 用户 | 运维 | 1 | 实现商业价值业务运行环境 |
| 用户验收测试环境 | **uat** | 测试、产品经理 | 测试 | 1 | 测试同学进行集成测试和用户功能验收的环境 |
| 测试环境 | **test** | 测试 | 测试 | 1 | 测试同学进行测试验证的环境 |
| 开发环境 | **dev** | 开发 | 开发 | N | 开发同学进行开发调试的环境（可以是本地独立的环境，或开发共享的环境） |

1. **git分支约定**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **分支名称** | **对应环境** | **维护者** | **数量** | **备注** | **生命周期** |
| master | 生产环境 | 运维 | 1 | 线上master分支（稳定版本） | 长期 |
| uat | 用户验收测试环境 | 运维 | 1 | uat分支（预生产环境）,定期从生产环境数据库同步数据 | 长期 |
| test | 测试验收测试环境 | 测试 | 1 | test分支（集成测试环境）,用作测试人员集成测试 | 长期 |
| dev | 开发验收测试环境 | 开发 | 1 | dev分支（集成开发环境）,用作开发人员集成测试 | 长期 |
| test\_<CRID> | 测试环境 | 测试 | N | 每个CR功能需求对应的代码分支（用于测试） | 短期 |
| dev\_<CRID> | 开发环境 | 开发 | N | 每个CR功能需求对应的代码分支（用于开发） | 短期 |
| bug\_<BUGID> | 开发环境 | 开发 | 0..N | 从master checkout的代码分支（用于bug修复） | 短期 |

注：1. CRID为CR在录入Jira时分配的唯一ID

1. BUGID为BUG在录入Jira时分配的唯一ID，

如果开发人员涉及多个BUG需同时修复，分支命名时多个BUGID可以 + 相连， 如：bug\_BUGID1+BUGID2+BUGID3+...

**Git分支管理原则**

1. 保持master稳定性（生产环境发布版本基于此分支构建）
2. 灵活创建开发分支
   1. 正常开发：按项目分配的CRID命名新分支进行开发
   2. 紧急Bug修复：从master创建新分支
3. 正常需求开发代码合并流程
4. 开发人员负责合并dev\_CRID到dev分支（可选）
5. 开发人员负责合并dev\_CRID到test\_CRID分支
6. 开发人员负责合并test\_CRID到test分支（可选）
7. 开发人员负责合并test\_CRID到uat分支
8. 运维人员负责合并uat到master分支
9. 紧急Bug修复代码合并流程

a）开发人员负责合并bug\_BUGID到dev分支（可选）

b）开发人员负责合并bug\_BUGID到test分支

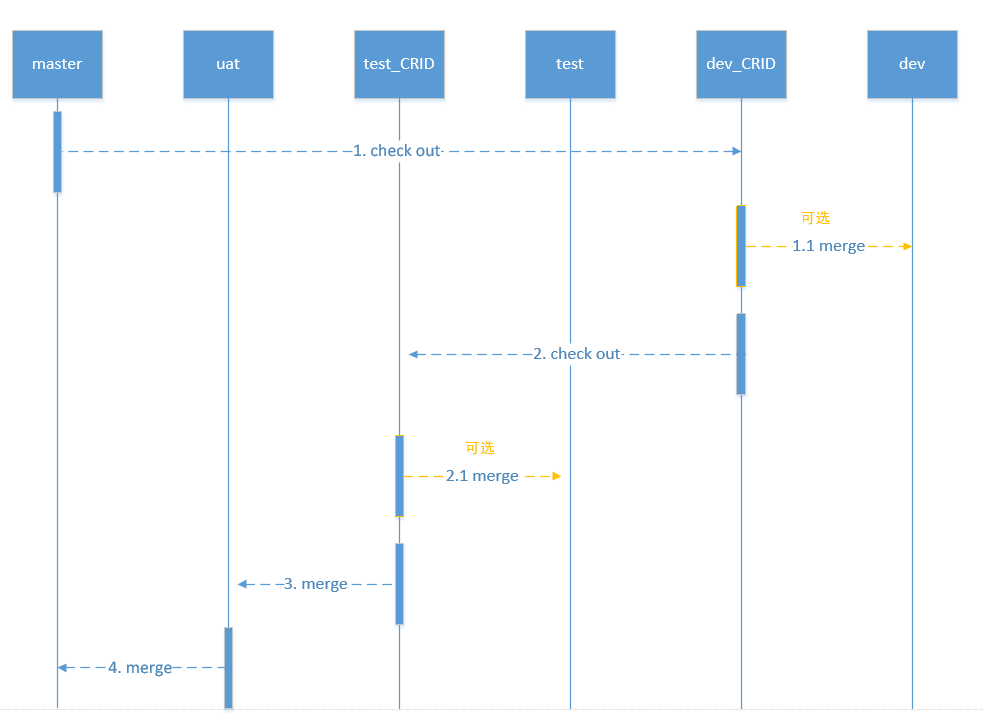
c）开发人员负责合并bug\_BUGID到uat分支

e）运维人员负责合并bug\_BUGID到master分支

1. 及时清除过期分支（减少过多垃圾分支）

* **开发流程**

1. 以启动一个新的CR功能开发为例 (下图省略了分支push和pull操作的描述)



1. 生产环境紧急bug处理 (下图省略了分支push和pull操作的描述)

