**Git版本管理工具**

**操作规范**

**2017年8月**

**修改记录**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **编号** | **版本** | **描述** | **日期** | **制定/修订** | **审核** | **备注** |
| 1 | V1.0 | 新建文档 | 20170512 | 刘田  李凯靖 | / |  |
| 2 | V1.1 | 调整文档结构，增加站点容器系统的git操作规范相关内容，增加各系统工作流说明 | 20170522 | 李凯靖 | 彭欢  刘田 |  |
| 3 | V1.2 | 1）3.1.2.5增加注意事项  2）3.1.3.4增加注意事项  3）增加3.1.3.5要求  4）修改4.2.2要求 | 20170830 | 刘爱信 |  |  |
|  |  |  |  |  |  |  |

目录

[１ 引言 - 6 -](#_Toc9558)

[１.１ 文档目的 - 6 -](#_Toc26602)

[１.２ 适用对象 - 6 -](#_Toc547)

[１.３ 适用范围 - 6 -](#_Toc27979)

[２ 分支命名规范 - 6 -](#_Toc8008)

[３ 操作规范 - 7 -](#_Toc3106)

[３.１ OA 系统 - 7 -](#_Toc26005)

[３.１.１ 工作流 - 7 -](#_Toc16359)

[３.１.２ 开发人员日常操作规范 - 7 -](#_Toc26851)

[３.１.２.１ 克隆分支 - 7 -](#_Toc16971)

[３.１.２.２ 创建新分支 - 8 -](#_Toc1133)

[３.１.２.３ 提交修改内容 - 8 -](#_Toc12453)

[３.１.２.４ 推送自己的开发分支到远端 - 8 -](#_Toc31510)

[３.１.２.５ 合并自己的开发分支到主开发分支 - 9 -](#_Toc31144)

[３.１.３ 测试人员日常操作 - 9 -](#_Toc17076)

[３.１.３.１ 克隆分支 - 9 -](#_Toc15866)

[３.１.３.２ 创建新分支操作 - 9 -](#_Toc6871)

[３.１.３.３ 合并开发人员的代码 - 10 -](#_Toc25193)

[３.１.３.４ 合并自己的测试分支到主测试分支 - 10 -](#_Toc21156)

[３.１.３.５ 提交分支时排除配置文件 - 10 -](#_Toc18635)

[３.１.４ 运维人员日常操作 - 11 -](#_Toc32450)

[３.１.４.１ 克隆分支 - 11 -](#_Toc3272)

[３.１.４.２ beta环境发布 - 11 -](#_Toc10087)

[３.１.４.３ 正式环境发布 - 11 -](#_Toc19494)

[３.２ 站点容器系统 - 12 -](#_Toc30723)

[３.２.１ 工作流 - 12 -](#_Toc32337)

[３.２.２ 开发人员操作规范 - 12 -](#_Toc1314)

[３.２.３ 测试人员操作规范 - 13 -](#_Toc15664)

[３.２.３.１ 发送安装包给运维 - 13 -](#_Toc8823)

[３.２.３.２ 合并开发分支到生产环境主分支 - 13 -](#_Toc12673)

[４ 操作重点注意事项 - 14 -](#_Toc21042)

[４.１ 开发人员注意事项 - 14 -](#_Toc10078)

[４.１.１ 合并代码操作注意事项 - 14 -](#_Toc24590)

[４.１.２ 解决代码合并冲突注意事项 - 14 -](#_Toc13203)

[４.１.３ 配置文件操作注意事项 - 15 -](#_Toc3510)

[４.２ 测试人员注意事项 - 15 -](#_Toc22760)

[４.２.１ 解决冲突注意事项 - 15 -](#_Toc9076)

[４.２.２ 配置文件操作注意事项 - 15 -](#_Toc21655)

[４.２.３ 分支合并注意事项 - 15 -](#_Toc3917)

[４.３ 运维人员注意事项 - 16 -](#_Toc13179)

[４.３.１ 分支合并注意事项 - 16 -](#_Toc19952)

[４.４ 个人账号管理 - 16 -](#_Toc7620)

[５ 定责标准 - 16 -](#_Toc29729)

[５.１ 对日常工作造成影响 - 16 -](#_Toc29999)

[５.２ 造成公司经济损失 - 17 -](#_Toc9356)

# 引言

## 文档目的

本文档旨在制定统一的 Git 版本管理工具的日常操作规范，解决因不规范操作而引起的系统异常问题，提高开发人员、测试人员、运维人员的工作效率与质量，提升系统运行的稳定性。

## 适用对象

本文档适用于所有开发人员、测试人员、运维人员等相关技术人员使用。

## 适用范围

本文档适用于 Git 版本管理工具（OA 系统与站点容器系统）。

# 分支命名规范

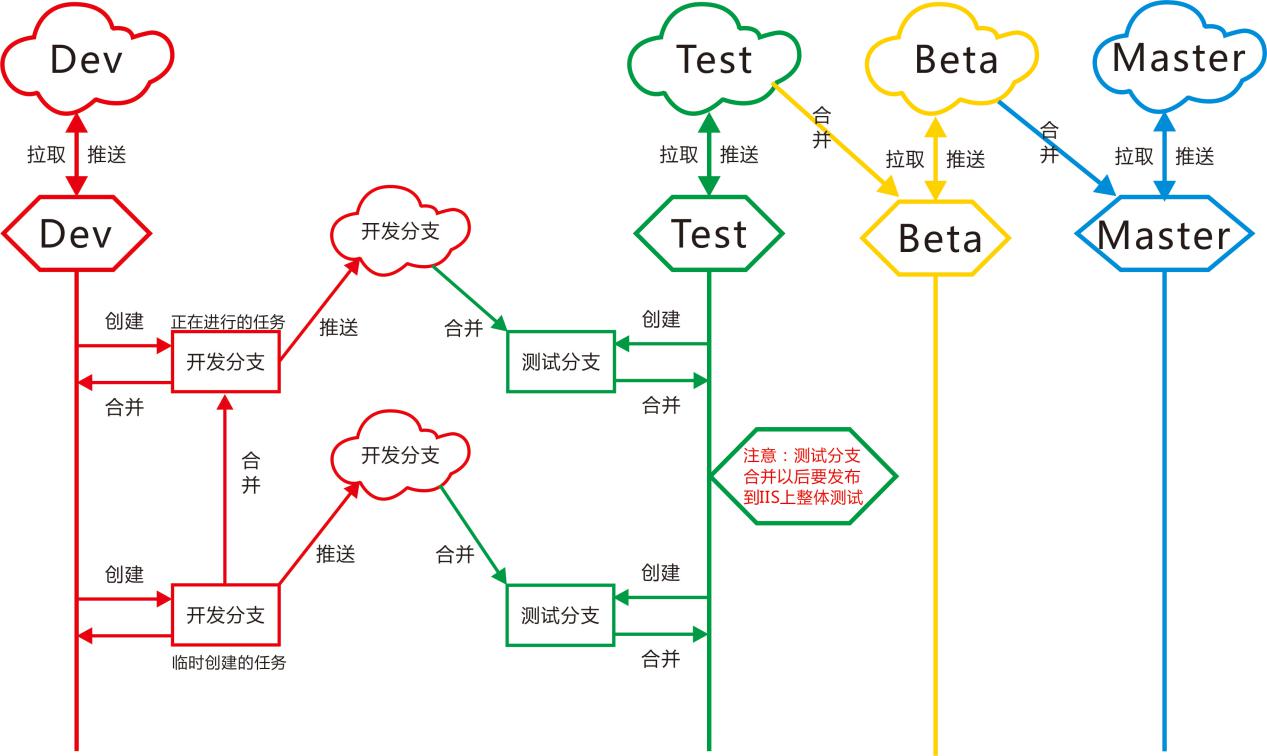
|  |  |
| --- | --- |
| 分支 | 命名规范 |
| 开发分支 | Dev |
| 测试分支 | Test |
| Beta分支 | Beta |
| 生产环境分支 | Master |
| 新开发分支 | 姓名拼音\_[bug|task]\_编号\_简要描述 |
| 新测试分支 | 姓名拼音\_[bug|task]\_编号\_简要描述 |

# 操作规范

## OA 系统

### 工作流

* OA系统采用四条主分支的管理方式，分别为开发分支Dev、测试分支Test、Beta分支以及生产环境分支Master，其工作流如下图所示：



OA系统工作流示意图

### 开发人员日常操作规范

#### 克隆分支

* 适用场景：开发人员第一次操作或后期特殊情况需要重新克隆分支时执行此操作。
* 操作步骤：右键 Git 克隆-》输入源码地址，设置分支名称（开发人员固定为Dev）-》确定完成。

#### 创建新分支

* 适用场景：开发人员有新的任务或BUG时需要创建新分支处理时执行此操作
* 操作步骤：在主开发分支下右键拉取-》右键创建分支（分支命名规范：姓名拼音\_[bug|task]\_编号\_简要描述），填写相应描述-》右键“切换/检出”切换到新添加的分支。

#### 提交修改内容

* 适用场景：开发人员创建的开发分支完成或暂停任务时执行此操作
* 操作要求：不要求每做一次操作都进行一次提交，但是当需要切换分支或是进行推送、拉取分支时，要求必须提交当前的操作（新增文件一定要记录加入到 Git 上）。
* 提示：提交只会提交到本地不会影响别人的代码或是主开发分支。

#### 推送自己的开发分支到远端

* 适用场景：开发人员将一个任务或 BUG 处理完，需要将分支推送到远端时执行此操作。
* 操作步骤：在当前自己的开发分支下右键推送（先提交后推送）-》输入远端分支名称（与当前本地分支名保持一致）-》确定完成。

#### 合并自己的开发分支到主开发分支

* 适用场景：当测试人员反馈安装包己经测试通过，此时开发人员需第一时间将已通过测试的开发分支合并到主开发分支。
* 操作步骤：提交自己手上正在处理的分支-》切换到主开发分支上-》右键拉取主开发分支-》右键合并（找到对应的开发分支）-》确定完成（如发生冲突则先解决冲突）-》右键推送（合并后的代码必须要推送到远端主开发分支上）。
* 注意事项：推送者有义务有责任确保自己的东西是推送成功的，如未推送成功一切后果自己负责。

### 测试人员日常操作

#### 克隆分支

* 适用场景：测试人员第一次操作或后期特殊情况需要重新克隆分支时执行此操作。
* 操作步骤：右键 Git 克隆-》输入源码地址，设置分支名称（测试人员固定为 Test ） -》确定完成。

#### 创建新分支操作

* 适用场景：测试人员需要对开发人员的代码进行测试时必须创建新分支来处理。
* 操作步骤：在主测试分支下右键拉取-》右键创建分支（分支命名建议：姓名拼音\_[bug|task]\_编号\_简要描述）,填写相应描述-》右键“切换/检出”切换到新添加的分支。

#### 合并开发人员的代码

* 适用场景：测试人员需要先合并开发人员的代码，再开展测试工作。
* 操作步骤：在当前自己的测试分支下右键合并-》搜索并选择相应开发人员的分支-》确定合并（如果出现冲突，请告知开发人员自行解决冲突后，再重新提交）。

#### 合并自己的测试分支到主测试分支

* 适用场景：测试通过后测试人员需要将代码合并到主测试分支上。
* 操作步骤：右键切换到主测试分支-》右键拉取最新的主测试分支-》右键合并（找到对应的测试分支）-》确定完成-》将程序发布到 IIS 上再进行一次测试-》测试完成后右键推送（合并上来的代码必须要推送到远端主测试分支上）。
* 注意事项：推送者有义务有责任确保自己的东西是推送成功的，如未推送成功一切后果自己负责。

#### 提交分支时排除配置文件

* 适用场景：突发紧急测试项目，测试人员提交正在测试的分支
* 操作步骤：右键提交分支-》编写提交信息-》提交（勿勾选配置文件）

### 运维人员日常操作

#### 克隆分支

* 适用场景：运维人员第一次操作或后期特殊情况需要重新克隆分支时执行此操作。
* 操作步骤：右键 Git 克隆-》输入源码地址（Beta环境需要设置分支名称为Beta，Master环境无需设置分支名称） -》确定完成。

#### beta环境发布

* 适用场景：运维人员需要进行beta环境版本发布时。
* 操作步骤：右键拉取最新的Beta分支-》右键合并（找到主测试分支）-》确定完成-》提交到Beta本地仓库-》推送到远端Beta仓库

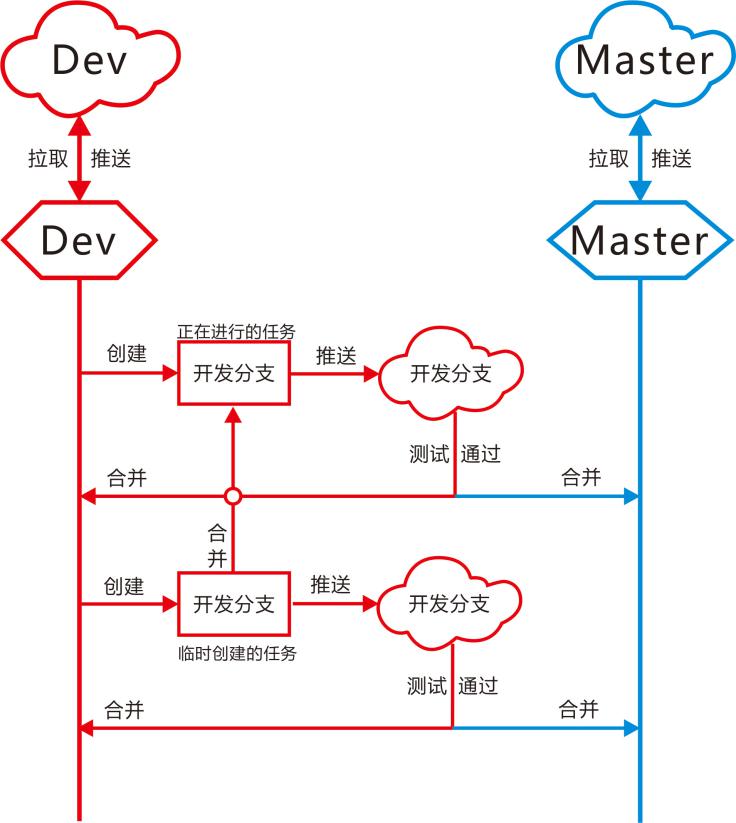
#### 正式环境发布

* 适用场景：运维人员需要进行正式生产环境版本发布时。
* 操作步骤：右键拉取最新的Master分支-》右键合并（找到主Beta分支）-》确定完成-》提交到Master本地仓库-》推送到远端Master仓库

## 站点容器系统

### 工作流

* 站点容器系统采用两条主分支的管理方式，分别为开发分支Dev与生产环境分支Master，其工作流如下图所示：



站点容器工作流示意图

### 开发人员操作规范

|  |  |
| --- | --- |
| 操作项目 | 详情 |
| 克隆新分支 | 参考3.1.2.1 |
| 创建新分支 | 参考3.1.2.2 |
| 提交修改内容 | 参考3.1.2.3 |
| 推送自己的开发分支到远端 | 参考3.1.2.4 |
| 打包安装包 | 开发人员需在模块根目录创建readme.md文件，并将相关信息打包为安装包提交测试人员测试 |
| 合并自己的开发分支到主开发分支 | 参考3.1.2.5，但如果修改的代码涉及多个模块，需相关模块都测试通过，才能进行开发分支的合并操作 |

### 测试人员操作规范

#### 发送安装包给运维

* 适用场景：开发分支打包为安装包，对涉及模块进行测试并测试通过后执行该操作
* 操作步骤：按照发布计划将安装包提交给运维人员，进行发布

#### 合并开发分支到生产环境主分支

* 适用场景：开发分支打包为安装包，对涉及模块进行测试并测试通过后执行该操作
* 操作步骤：右键切换到Master主分支-》右键拉取最新的Master主分支-》右键合并（找到对应的开发分支）-》确定完成-》推送到远端（合并上来的代码必须要推送到远端Master分支上）。

# 操作重点注意事项

## 开发人员注意事项

### 合并代码操作注意事项

* 合并代码前，必须从主分支上拉取最新的代码，再进行合并。
* 进行代码合并时，要求开发人员不能覆盖他人代码，如出现代码覆盖导致系统异常的问题，发现人需自行确认责任人，未及时查找责任人，则由发现人承担责任。
* 责任人需协助问题发现人进行问题修复，不允许出现不理睬不处理。
* 提示：出现代码覆盖问题，目前推荐的解决方案由被覆盖了代码的责任人重新提交本地代码。
* 如果修改的代码涉及多个模块，需相关模块都测试通过，才能进行开发分支的合并操作。
* 当测试人员告知开发人员己经测试通过时，开发人员要第一时间将分支合并到主开发上，以免出现其他人拉不到最新的代码造成后期代码合并测试时产生冲突。

### 解决代码合并冲突注意事项

* 当出现冲突时，由当前开发人员负责解决冲突，如需要其他人协助解决冲突，请积极与相关人员进行沟通。
* 如果开发人员在解决冲突时需要另一个开发人员协助，另一个开发人员需全力协助，不允许出现不理睬不处理。
* 当测试人员合并代码时出现冲突，开发人员需积极解决冲突，但不能影响测试人员的正常工作。

### 配置文件操作注意事项

* 开发人员不允许添加配置文件，包括但不仅限于以下配置文件：

|  |  |  |
| --- | --- | --- |
| App\_Data | Web.config | bgservice的Web.config |
| hibernate.cfg.xml | hibernate\_194.cfg.xml | SellerCube.SQLServer.dbmdl |
| SellerCube.SQLServer.jfm |  |  |

## 测试人员注意事项

### 解决冲突注意事项

* 如果程序出现冲突必须由开发人员进行解决，测试人员不能擅自修改程序代码，不能私自解决，更不能为了解决冲突而还原开发人员的代码。

### 配置文件操作注意事项

* 测试人员允许删除配置文件，且提交时，不提交配置文件，如出现跟配置文件相关的报错，请进行还原配置文件操作。

### 分支合并注意事项

* 合并分支前，必须拉取最新的分支，再进行合并。
* 测试通过后测试分支，成功合并到主测试分支上后，测试人员需第一时间通知开发人员，当前分支已测试通过，便于开发人员进行开发分支合并到主开发分支操作。
* 测试通过后的开发分支，需按照发布计划，将安装包发给运维人员进行发布，同时应及时将相应的开发分支，合并到Master主分支上。

## 运维人员注意事项

### 分支合并注意事项

* OA系统进行分支合并前，必须从（Beta/Master）环境拉取最新的分支，再进行合并。

## 个人账号管理

* 各系统对角色、个人的账号都有相应的权限管理机制，请妥善管理个人账号，不得转借他人使用。

# 定责标准

## 对日常工作造成影响

* 对于违反 Git 操作规范者，将综合实际情况，参考以下标准进行处罚：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 影响范围  分值 | 1分 | 2分 | 3分 | 4分 |
| 修复时间 | 0-1H | 1-2H | 2-4H | 4H以上 |
| 直接影响人（其他开发团队） | 0-3人 | 4-6人 | 7-9人 | 10人或以上 |
| 投诉量（用户） | 0-3人 | 4-6人 | 7-9人 | 10人或以上 |

|  |  |  |
| --- | --- | --- |
| 严重程度 | 分值 | 处罚标准 |
| 严重 | 11-12 | 100 |
| 中等 | 9-10 | 50 |
| 一般 | 6-8 | 30 |
| 轻微 | 3-5 | 10 |

* 若出现问题不在本规范定义内，将以公告形式进行警告，后续增加对应的规范。
* 系统因git操作不规范引起异常时，相关人员如不配合修复问题，或故意延长修复时间，则以当前严重程度为标准进行定责（例如由于A人员合并代码时，覆盖他人代码造成异常，严重程度为轻微，但其不配合做修复，或故意延长修复时间，则以双倍进行处罚）

## 造成公司经济损失

* 对于违反 Git 操作规范，或因非正常操作对 Git 内项目造成损坏，直接造成公司经济损失的，将按照实际情况评定处罚结果。