代码上传日志规范及开发目录规范

# **1. 先更新，再提交**

SVN更新的原则是要随时更新，随时提交。

当完成了一个小功能，能够通过 编译并且自己测试之后，谨慎地提交。 如果在修改的期间别人也更改了svn的对应文件，那么commit就可能会 失败。如果别人和自

己更改的是同一个文件，那么update时会自动进行合并，如果修改的是同一行，那么合并时会产生冲突，这种情况就需要同之前的开发人员联系，两个人一起协商解决冲突，解决冲突之后，需要两人一起测试保证解决冲突之后，程序不会影响其他功能。 在更新时注意所更新文件的列表，如果提交过程中产生了更新，则也是 需要重新编译并且完成自己的一些必要测试，再进行提交。这样既能了解别人修改了哪些文件，同时也能避免SVN合并错误导致代码有错。

# **2. 一个提交尽量对应一个逻辑问题**

一次提交多个问题的修改或者一次提交多个模块的不同改动是不被允许的，这样也就失去了版本管理的意义。我们要尽量做到一次提交对应一个问题，涉及一个模块。 如果某次修改必须涉及到多个模块，请谨慎检查代码和库上的代码情况，确保不会因为你的上传导致无法回退。

# **3. 多提交 每次提交的间歇尽可能地短**

以几个小时的开发工作为宜。例如在更改UI 界面的时候，可以每完成一个UI界面的修改或者设计，就提交一次。在开发功能模块的时候，可以每完成一个小细节功能的测试，就提交一次，在修改bug的时候，每修改掉一个bug并且确认修改了这个bug，也就提交一次。我们提倡多提交，也就能多为代码添加上保险。

# **4. 不要提交不能通过编译的代码**

代码在提交之前，首先要确认自己能够在本地编译。如果在代码中使用了第三方类库，要考虑到项目组成员中有些成员可能没有安装相应的第三方类库。项目经理在准备项目工作区域的时候，需要考虑到这样的情况，确保开发小组成员在签出代码之后能够在统一的环境中进行编译。

# **5. 每次提交必须书写明晰的标注**

在一个项目组中使用SVN，如果提交空的标注或者不确切的标注将会让项目组中其他的成员感到很无奈，项目经理无法很清晰的掌握工作进度，无法清晰的把握此次提交的概要信息。在发现错误后也无法准确的定位引起错误的文件。所以，在提交工作时，要填写明晰的标注，能够概要的描述所提交文件的信息，让项目组其他成员在看到标注后不用详细看代码就能了解你所做的修改。

# **6. 对于不同类型的提交需要使用不同类型的前缀作为标记**

删除某个文件 -: 增加文件 +: 修改文件 \*: 解决BUG,需要加上fixbug：BUGID

# **7. 提交时注意不要提交本地自动生成的文件**

例如eclipse中的.classpath文件，Windows生成的缩略图Thumbs.db，项目编译生成的临时文件.obj, .class等等。如果项目中没有进行这方面的配置来强行禁止提交这样的文件，请自觉不要提交这样的文件。提交了这样的文件后，别人在更新后就可能与本地的环境冲突从而影响大家的工作。

# **8. 不要提交自己不明白的代码**

代码在提交入SVN之后，你的代码将被项目成员所分享。如果提交了你不明白的代码，你看不懂，别人也看不懂，如果在以后出现了问题将会成为项目质量的隐患。因此在引入任何第三方代码之前，确保你对这个代码有一个很清晰的了解。

# **9. 慎用锁定功能**

在项目中要慎用锁定的功能，在你锁定了一个文件之后别人就无法继续修改提交该文件，虽然可以减少冲突的发生率，但是可能会影响项目组中其他人员的工作。平时只有在编辑那些无法合并的文件（例如图片文件，flash文件等）时，才适当的采用锁定操作。

# **10. 使用TAG功能为**

release版本做标记 用Release Tags 标记你版本发布点的代码。 Release Tag 永远是相应发布分支的副本。 Release Tag命名规则：“REL-”前缀加上版本号。

# **备注：**

## **1、svn日志描述**

在我们的代码完成一个任务并且测试没有问题之后，应该尽快将代码提交到svn版本库中，而在提交的时候svn工具会提示我们输入日志。日志当然可以随便输入，但是我们要说的是如何规范日志的写法。这一点很重要，因为日后的开发过程中不免要回头看看上一个版本中我们修改了哪些内容。也许前面一两个版本的内容开发人员会记得比较清楚，但是再往前便不一定了。所以，在每次上传代码到svn版本库的时候一定要详细的记录本次修改了哪些内容，这就体现在日志如何填写了。根据我个人的总结，以及参考其他人的写法，得到了一个简介的日志模板：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Start：SVN日志模板\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

【提交类型】： Bug修复/阶段性提交/追加递交/新功能/需求修改/解决编译不过/代码整理/初次提交

【问题描述】：

1、添加了xxx功能；

2、解决了xxx导致的xxx问题；

......

【修改内容】：

1、xxx.h文件中添加/修改了xxx；

2、xxx.hc文件中添加/修改了xxx；

......

【相关单号】： Bug号/库版本号/......

【评 审 人】： xxx

【需要测试】： 是/否

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*End：SVN日志模板\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

举个例子，假如说本次代码时第一次上传svn，那我们在填写日志时可以这样填写：

【提交类型】： 初次提交

【问题描述】： 初次提交DIL模块代码

【修改内容】： 所有代码均为初次提交

【相关单号】： DIL库版本号1.00.000

【评 审 人】： 张三

【需要测试】： 否

## **2、svn日志模板化**

svn工具提供了方便的日志模板管理，我们可以在使用之前将模板加入svn的属性中，然后填入模板，这样每次提交代码的时候便不用手动输入模板，而且不容易产生手动输入的错误。以TortoiseSVN为例，此操作的步骤如下。

（1）、在本地svn目录下，右键选择TortoiseSVN->Properties

（2）、在弹出的窗口中，点击New，选择Other

（3）、在弹出的窗口中，点击下拉款，选择tsvn:logtemplate，在Property value窗口中填入模板即可。