版本发布管理规范

文件编号：IFENG-2011-TS-PR-01

版 本 号：V1.0

编 制：金明岩

审 核：

批 准：

发布日期： 2011年 月 日 实施日期： 2011年 月 日

凤凰新媒体-技术部

目录

[版本发布管理规范 1](#_Toc291512425)

[1. 目的及约束 3](#_Toc291512426)

[1.1 目的 3](#_Toc291512427)

[1.2 约束范围 3](#_Toc291512428)

[2. 版本分类 3](#_Toc291512429)

[2.1 按照测试状态的分类 3](#_Toc291512430)

[2.1.1 dev版本 3](#_Toc291512431)

[2.1.2 alpha版本 3](#_Toc291512432)

[2.1.3 beta版本 4](#_Toc291512433)

[2.1.4 ga版本 4](#_Toc291512434)

[2.1.5 测试状态版本迁移过程 4](#_Toc291512435)

[2.2 按照功能集合分类 5](#_Toc291512436)

[2.2.1 DEMO版本 5](#_Toc291512437)

[2.2.2 ENHACE版本 5](#_Toc291512438)

[2.2.3 FULL版本 5](#_Toc291512439)

[3. 版本计划 5](#_Toc291512440)

[3.1 场景 5](#_Toc291512441)

[3.2 版本计划 5](#_Toc291512442)

[3.3 版本演进历史 7](#_Toc291512443)

[4. 版本命名规范 8](#_Toc291512444)

[4.1 名称约定 8](#_Toc291512445)

[4.1.1 命名与缩写 8](#_Toc291512446)

[4.1.2 命名的限制 8](#_Toc291512447)

[4.2 分支命名规范 8](#_Toc291512448)

[4.3 版本命名规范 8](#_Toc291512449)

[4.4 补丁(patch)命名规范 9](#_Toc291512450)

[5. 发布物内容规范 9](#_Toc291512451)

[5.1 必须包含的内容 9](#_Toc291512452)

[5.1.1 ReleaseNotes 9](#_Toc291512453)

[5.1.2 安装包 10](#_Toc291512454)

[5.2 发布包的命名规则 11](#_Toc291512455)

[6. 发布工作流程 11](#_Toc291512456)

[6.1 发布dev版本 11](#_Toc291512457)

[6.2 发布alpha版本 12](#_Toc291512458)

[6.3 发布beta版本 12](#_Toc291512459)

[6.4 发布ga版本 12](#_Toc291512460)

[7. 发布服务器 12](#_Toc291512461)

**本文档修订记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **修订时间** | **修订人** | **修订内容** |
| V0.1 | 2011-4-25 | 金明岩 | 初稿 |
|  |  |  |  |

# 目的及约束

## 目的

为了规范软件产品版本标识；规范软件产品发布、测试及上线过程，特制订本规范，自实施之日起，请相关部门相关人员严格按照此规范进行版本发布的命名及相关过程。

## 约束范围

版本发布规范对下列过程存在约束：

1. 软件产品版本发布过程。
2. 软件产品测试过程。
3. 软件产品上线安装实施过程。
4. 软件产品缺陷修改及再发布过程。
5. 软件产品缺陷跟踪过程。
6. 软件产品需求变更或追加过程。

# 版本分类

## 按照测试状态的分类

软件产品按照测试状态分为如下三个类别：

### dev版本

dev即development,dev版本即开发版本，用于在开发过程中未提交测试时使用的版本，一般用于开发团队内部阶段性工作标示、内部集成测试以及其他可能的应用场合。

### alpha版本

alpha版本即产品alpha测试版本，对应于alpha测试活动的输入。

alpha测试是指在技术团队内部，在一个测试环境（非线上环境）里，针对用户的使用要求进行的测试。测试的过程需要记录相关问题，并后续修正和评估这些问题。测试在受控的环境中进行。

### beta版本

beta版本即产品beta测试版本，对应beta测试活动的输入。

beta测试是指软件在实际使用环境下进行的测试。beta测试不能由程序员或测试员完成。因而，beta测试是在开发者无法控制的环境下进行的软件现场应用。在beta测试中，由用户记下遇到的所有问题，定期向开发者报告，开发者在综合用户的报告后，做出修改，最后将软件产品交付给全体用户使用。beta测试着重于产品的支持性，包括文档、客户培训和支持产品的生产能力。只有当alpha测试达到一定的可靠程度后，才能开始beta测试。

### ga版本

ga即General Availability,ga版本即正式发布的版本。

ga版本并不一定针对某中测试活动的输出物，只是软件在线上系统的一个标志。软件需要经历的测试过程，应该有软件需求和开发活动的管理者，在针对软件的功能、性能、影响程度等各方面表现进行综合考虑以后，针对现有的软件缺陷跟踪情况做出上线评估给出。

### 测试状态版本迁移过程



1. 每一个版本类型都应该有自己的演进序列。

2. 非dev版本必须依赖一个dev版本。

3. 版本需要按照演进过程给予一个顺序编号。

4. 非dev版本需要在编号中后缀一个依赖的dev版本的编号。

## 按照功能集合分类

### DEMO版本

包含正式软件的部分功能，可以体现软件的基本操作，为后续版本的开发提供可感知的需求及设计影响能力。

DEMO 版本的属性一定是部分的、可运行的、可体验的。

### ENHACE版本

增强版（加强版），针对DEMO能力的扩展，提供了更多的用户体验和功能约束维度的实现，加强版在某些场景下已经可以在线上应用。

ENHACE版本的属性一定要包含DEMO版本的属性，且在体验和功能约束维度有改进。

### FULL版本

完全版，应该包含需求范围内的全部功能，如果不包含全部功能的，需要在上线评估之前一并修正需求文档中针对功能的描述，将不包含的功能修正为可选的以后，才可以发布完全版。

# 版本计划

## 场景

在有开发活动时，就应该有开发活动对应的输出物，有输出物就应该有对应的版本计划。

常见的开发活动出现在如下的场景中：

1. 开发一个新的产品。
2. 已有的产品要支持新的需求。
3. 已有的产品发现缺陷。

## 版本计划

需求分析

确定开发计划

确定版本计划

1． 需求分析：对需求进行分析，分解，确定产品要实现的需求。

2． 确定开发计划：根据需求分析的结果安排开发计划，包括设计、开发、测试等过程；在开发计划中要明确迭代过程，明确阶段性工作目标。

3． 确定版本计划：确定开发小组推出新版本频率及名称，一般对应于开发活动中的阶段性目标的完成，针对目标的下一环节确定版本面向测试状态的分类；针对目标完成时功能的情况确定产品面向功能集合的分类。

## 版本演进历史



# 版本命名规范

为了规范公司的版本管理，首先需要规范版本的名称，在创建分支和发布版本时，都要遵循以下命名规范。

## 名称约定

### 命名与缩写

1. 版本状态

版本针对测试的状态包：dev、alpha、beta、ga ,原则上线上版本必须为ga版本。

版本针对功能集合的状态包括：DEMO、ENHACE、FULL，原则上DEMO版本不能为上线版本。

1. 补丁

为了满足线上故障的紧急修复需要，可能为某一个版本发布一个临时的补丁，补丁中一般只包含针对对应版本的修正物，包括但不限于：程序、配置、文档等。

### 命名的限制

1. 分支和tag标记不支持中文，请统一使用英文缩写。
2. 除了固定名称外，请一律使用字母小写。
3. 使用“-”作为一个域中不同文字的分隔符，使用“\_”作为域分隔符。

4 . 分支名称是区分大小写的。

## 分支命名规范

分支必须在SVN上有对应的tag.

分支tag的命名格式为:[产品名称]\_BRANCH\_[branch-name]

其中：

BRANCH:为分支的固定标记，英文大写。

[branch-name] 分支名称，小写的英文或拼音缩写。

例如：

iCdn\_BRANCH\_working

## 版本命名规范

为了记录版本，保持版本的可追溯性，必要时能够复原版本的内容，在发布版本时，必须在svn上该版本包含的所有文件的正确version上建立一个版本标签。

**版本标签的格式为：**

[产品名称]**\_BUILD\_[ dev or alpha or beta or ga]\_[DEMO or ENHACE or FULL]【build-name】\_【build-number】\_[dev-build-number]**

1. BUILD

固定名称，英文大写。

1. 测试状态及功能集合分类，请按照要求选择对应的分类。
2. 【build-name】

可选项，版本内容的名称，小写的英文或拼音缩写。

1. **【build-number】**

build的编号，当版本为非dev版本时使用。对于每个产品，其对应测试状态的编号都是从1开始，以1递增。

1. **[dev-build-number]**

为方便检索，对alpha、beta、ga的build号添加相应的dev版本号域，以dev的build号作为唯一索引，但alpha、beta、ga的版本号各自独立递增排序。

**示例**

iCDN\_ BUILD \_dev\_DEMO\_5

iCDN\_ BUILD \_alpha\_DEMO\_2\_5

依赖iCDN\_ BUILD \_dev\_\*\_5的alpha版本。

iCDN\_BUILD\_beta \_FULL\_isyn-server\_1\_15

说明：iCDN产品针对isyn-server的经过测试部测试后提交beta测试的版本，beta版本号为1，对应的开发版本号为15

## 补丁(patch)命名规范

PATCH的标签格式为：

版本号\_PATCH\_[patch编号]

1. PATCH:固定名称，英文大写。

2. 编号都是从1开始，以1递增。

# 发布物内容规范

## 必须包含的内容

### ReleaseNotes

它是一个发布历史的记录，里面包含有一个产品的所有发布历史，介绍了产品的版本演进和变化情况，这对产品的后续开发和维护非常重要。

#### 如何写ReleaseNotes

在一个文本文档中按照如下格式书写：

[产品名称] Release Notes

Version:[版本名称]

发布日期:[YYYY-MM-DD]

-----------------------------------------------------------------------------------------------------

一、主要内容

1) 版本依赖关系

2) 合并的patch

3) 变更的功能和接口

4) 解决的缺陷

5) 升级安装步骤

6) 支持信息

7) 版本历史

------------------------------------------

二、版本依赖关系

-----------------------------------------------------------------------------------------------------

三、 合并的patch

无

-----------------------------------------------------------------------------------------------------

四、变更的功能和接口

接口：

功能：

-----------------------------------------------------------------------------------------------------

五、解决的缺陷

-----------------------------------------------------------------------------------------------------

六、升级安装步骤

-----------------------------------------------------------------------------------------------------

七、支持信息

-----------------------------------------------------------------------------------------------------

当多个版本时，请在文档开始增加最后一个版本的信息。也就是从上到下，最近到远的顺序。

### 安装包

必须包含产品的安装包，如果有手册类文档，应该包含在安装包中。

测试和运维人员可以根据ReleaseNotes中的安装说明，解压缩安装包以后根据指引完成安装过程。

## 发布包的命名规则

文件名为版本名称加.UP后缀，后缀为压缩类型。

比如：

iCDN\_BUILD\_beta \_FULL\_isyn-server\_1\_15.UP.tar.gz

如果升级包很小，只需要打包，可以不压缩。

# 发布工作流程



## 发布dev版本

1.按照模板编写ReleaseNotes文件，在svn上标记dev版本标签（参见4.3），包含所有相关的程序和文档。注意主干/分支和文件的版本。  
2. 从svn上按照版本标签检出源程序到一个干净的目录下，编译，之后将相关内容打包为一个安装包，上传到发布服务器。  
3. 给本研发小组、其它需要使用此版本的人员发邮件，通报版本的发布。

注意事项：

在把文件提交到svn以及在svn上创建版本标签时，要检查是否在正确的主干或者分支上，不要使用错误的分支。

对于一个非补丁版本，版本标签最好包含该软件的所有文件，每个文件包含在此标签中的version都是正确的。

对于一个补丁，则只需要包含本补丁修改过的几个文件以及ReleaseNotes.txt。

## 发布alpha版本

1. 修改ReleaseNotes文件，在svn上标记alpha版本标签（参见4.3,注意依赖的dev版本号）。  
2. 根据产品的特定情况，生成安装包。  
3. 把与版本名称相同的安装包上传到发布服务器的指定目录下，安装包中必须包含ReleaseNotes文件和系统测试报告。  
4. 给本研发小组、其它需要使用此版本的人员发邮件，通报版本的发布。  
注：发布alpha版本时，安装维护手册、用户使用手册应该齐全。

## 发布beta版本

1. 开发负责人审核系统测试报告。如果主流程、主要功能能够成功运行，没有严重影响的功能性错误，同意发布。  
2. 修改ReleaseNotes文件，在svn上标记beta版本标签（参见4.3,注意依赖的dev版本号），把安装包上传到发布服务器。  
3. 发出邮件通知所有相关人员。

## 发布ga版本

1. 开发负责人审核系统测试报告。如果主流程、主要功能能够成功运行，没有严重影响的功能性错误，同意发布。  
2. 修改ReleaseNotes文件，在svn上标记ga版本标签（参见4.3,注意依赖的dev版本号），把安装包上传到发布服务器。  
3. 发出邮件通知所有相关人员。

# 发布服务器

发布服务器，具体规范参见配置管理系统和版本管理系统的约束，本文只约束版本名称相关规范。