# 模版集萃

综述：

在程序员的日常工作中，除了编写代码之外，还免不了需要编写各种技术文档。一个编写良好的技术文档在项目中能够很好地建立沟通与协作，起到很积极的作用。因此，编写技术文档也就成为了程序员技能提升的很重要的一面。

为此，我们特意收集了一些在项目开发过程中经常用到的文档模板，这些模板包括格式和简单的写作说明，相信能够帮助大家编写出更加高效、实用的技术文档。在收集过程中，我们十分注重其实用性，以确保每个模板的价值，而且对于一些重要的文档提供了多个模板。

为了方便大家查找，我们将收录的 57 模板分为以下几类：

1） 项目及开发管理类：包括立项前的分析，立项后的计划、以及进度跟踪、风险控制方面的文档模板，共计 16 个；

2） 需求分析类：明确清晰的需求，是项目成功的基础，在此收集了在需求分析过程中所将使用到的文档模板，共计 14 个；

3） 系统分析与设计类：包括体系结构设计、高层设计、详细设计、数据库设计等 6 个相关文档模板；

4） 软件质量保证类：软件测试是质量保证的关键活动，在此收集了软件测试相关的 11 个文档模板；

5） 其它类：除此之外，还收集了关于用户手册、软件维护等方面的 10 个文档模板，其中还有一个软件过程规范的示例。

另外，值得说明的是，文档模板只是为文档的编写提供一个基础，在实际的编写过程中，你可以根据自己的需要进行必要的剪裁和增补。

## 一、 项目及开发管理类

### 可行性研究报告(ISO 标准)

编者说明：

在立项时，应该对项目进行综合分析，探讨项目的经济、社会、技术可行性，从而为决策提供基础。该模板为 ISO 标准文档模板，其不仅适用于软件项目，对于其它的系统项目也适用。

1. 引言

1.1 编写目的

[编写本可行性研究报告的目的，指出预期的读者。]

1.2 背景

a.[所建议开发的软件系统的名称；]

b.[本项目的任务提出者、开发者、用户及实现该软件的计算站或计算机网络；]

c.[该软件系统同其他系统或其他机构的基本的相互来往关系。]

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出用得着的参考资料。]

2. 可行性研究的前提

[说明对所建议开发的软件的项目进行可行性研究的前提。]

2.1 要求

[说明对所建议开发的软件的基本要求。]

2.2 目标

[说明所建议系统的主要开发目标。]

2.3 条件、假定和限制

[说明对这项开发中给出的条件、假定和所受到期的限制。]

2.4 进行可行性研究的方法

[说明这项可行性研究将是如何进行的，所建议的系统将是如何评价的，摘要说明所使用的基本方法和策略。]

2.5 评价尺度

[说明对系统进行评价时所使用的主要尺度。]

3. 对现有系统的分析

[这里的现有系统是指当前实际使用的系统，这个系统可能是计算机系统，也可能是一个机械系统甚至是一个人工系统。]

[分析现有系统的目的是为了进一步阐明建议中的开发新系统或修改现有系统的必要性。]

3.1 处理流程和数据流程

[说明现有系统的基本的处理流程和数据流程。此流程可用图表即流程图的形式表示，并加以叙述。]

3.2 工作负荷

[列出现有系统所承担的工作及工作量。]

3.3 费用开支

[列出由于运行现有系统所引起的费用开支。]

3.4 人员

[列出为了现有系统的运行和维护所需要的人员的专业技术类别和数量。]

3.5 设备

[列出现有系统所使用的各种设备。]

3.6 局限性

[列出本系统的主要局限性。]

4. 所建议的系统

4.1 对所建议系统的说明

[概括地说明所建议系统，并说明在第 2 条中列出的那些要求将如何得到满足，说明所使用的基本方法及理论根据。]

4.2 处理流程和数据流程。

[给出所建议系统的处理流程式和数据流程。]

4.3 改进之处

[按 2.2 条中列出的目标，逐项说明所建议系统相对于现存系统具有的改进。]

4.4 影响

[说明新提出的设备要求及对现存系统中尚可使用的设备须作出的修改。]

4.4.1.对设备的影响

[说明新提出的设备要求及对现存系统中尚可使用的设备须作出的修改]

4.4.2.对软件的影响

[说明为了使现存的应用软件和支持软件能够同所建议系统相适应，而需要对这些软件所进行的修改和补充。]

4.4.3.对用户单位机构的影响

[说明为了建立和运行所建议系统，对用户单位机构、人员的数量和技术水平等方面的全部要求。]

4.4.4.对系统运行过程的影响

[说明所建议系统对运行过程的影响。]

4.4.5.对开发的影响

[说明对开发的影响。]

4.4.6.对地点和设施的影响

[说明对建筑物改造的要求及对环境设施的要求。]

4.4.7.对经费开支的影响

[扼要说明为了所建议系统的开发，统计和维持运行而需要的各项经费开支。]

4.5 技术条件方面的可能性

[本节应说明技术条件方面的可能性]

5. 可选择的其他系统方案

[扼要说明曾考虑过的每一种可选择的系统方案，包括需开发的和可从国内国外直接购买的，如果没有供选择的系统方案可考虑，则说明这一点。]

5.1 可选择的系统方案 1

[说明可选择的系统方案 1，并说明它末被选中的理由。]

5.2 可选择的系统方案 2

[按类似 5。1 条的方式说明第 2 个乃至第 n 个可选择的系统方案。]

[……]

6. 投资及效益分析

6.1 支出

[对于所选择的方案，说明所需的费用，如果已有一个现存系统，则包括该系统继续运行期间所需的费用。]

6.1.1 基本建设投资

[包括采购、开发和安装所需的费用。]

6.1.2 其他一次性支出

6.1.3 非一次性支出

[列出在该系统生命期内按月或按季或按年支出的用于运行和维护的费用。]

6.2 收益

[对于所选择的方案，说明能够带来的收益，这里所说的收益，表现为开支费用的减少或避免、差错的减少、灵活性的增加、动作速度的提高和管理计划方面的改进等，包括：

6.2.1 一次性收益]

[说明能够用人民币数目表示的一次性收益，可按数据处理、用户、管理和支持等项分类叙述。]

6.2.2 非一次性收益

[说明在整个系统生命期内由于运行所建议系统而导致的按月的、按年的能用人民币数目表示的收益，包括开支的减少和避免。]

6.2.3 不可定量的收益

[逐项列出无法直用人民币表示的收益。]

6.3 收益/投资比

[求出整个系统生命期的收益/投资比值。]

6.4 投资回收周期

[求出收益的累计数开始超过支出的累计数的时间。]

6.5 敏感性分析

[是指一些关键性因素与这些不同类型之间的合理搭配、处理速度要求、设备和软件的配置等变化时，对开支和收益的影响最灵敏的范围的估计。]

7. 社会因素方面的可能性

7.1.[法律方面的可行性]

7.2.[使用方面的可行性]

8. 结论

[在进行可行性研究报告的编制时，必须有一个研究的结论]

# 软件项目商业性分析

批注[XuFeng1]： 注，改写自 RUP 模板中的《商业理由》

编者说明：

随着市场经济的不断发展，一个项目的商业价值、市场价值往往是衡量项目价值的最大依据。该文档模板十分适用于产品型项目，当你提出一个新的产品开发方向时，一份商业性分析是说服管理层的一个很好工具。

当然，如果是一些内部项目，也是可以借鉴该文档模板来论证该项目的商业价值。

1. 文档概述

[该部分主要描述该文档的目的、范围、术语以及参考资料等方面的内容。]

删除的内容： *注：一个更加现代的项目可行性分析*

1.1 目的

[说明该文档的作用。]

1.2 范围

[简要说明该与文档相关的其它事物与资料。]

1.3 术语

[列出所有将出现于本文档的新术语、缩略语等。]

1.4 参考资料

[在此应列出项目计划中引用的文档列表，对于引用的每个文档都应该列出其标题、文档编号、日期，并且指出这些文档的来源，以方便该计划的阅读者查找。]

1.5 概述

[本小节说明该文档所包括的内容，以及它的组织方式。]

2.系统说明

[在此简要地说明将要开发的系统，包括其名称、系统所解决的问题以及它的开发价值等，从而使得读者能够有一个直接的了解。并且在这处还应列出与在本文档中出现的缩略词的解释，以便读者更好地阅读。]

3.业务环境

[这一小节主要说明要开发的系统所处于的业务环境。它包括系统所面向的领域、用户。也可以在此指出它是产品型项目，还是用户定制型项目，同时如果该项目与原有的项目有紧密的联系，在此也应该把这些联系列出来。]

4.产品目标

[这一小节则用于深入说明为什么要开发该系统，它有什么价值。最好还应对进度计划、进度风险做一些评估。一个明确确定、表述清晰、可以度量的目标将为今后系统的开发工作

打下坚实的基础。]

5. 财务预测

[如果是产品型项目，那么其输出就是一个商业软件产品。对于这样的项目，在此应该包括对该项目的财务预测，最主要应该得出投资回报（ROI）指标。在做 ROI 分析时，应该针对不同的完成时间做出不同的预测，以让系统开发者对于进度延迟对投资回报的损伤有一个直观的了解。]

[在财务预测中，有一个基点就是对项目工作量、资源使用的估算，在这里还应给出估算的基础技术，当然这里的估算会随着项目的进展而逐步精化，应该这里还是应该估算出一个合理的范围。]

6. 约束

[任何事有利就有弊，在本小节则主要列举执行该项目时会遇到的一个诸如外部接口、标准、认证、特殊的技术等约束，这些约速将会对项目带来很大的执行风险，可能对项目的成本也带来巨大的影响。]

## 软件开发项目立项表

编者说明：

在许多开发组织中，开发立项请求通常来自市场部门，该表格的设计就是为了更好地理顺两个部门之间的沟通与协调，也使得开发立项流程化，你可以根据自己公司的实际情况，对该表格的格式做一些修改。

**删除的内容：** *注：一个挺古老的版本，但本质仍然未变*

| | |
|---|---|
| 项目名称(暂定)： | |
| 项目编号（开发部填写） | |
| 项目申请人： | 申请日期： |
| 项目优先级： | 最迟完成时间： |
| 问题/机会： | |
| 项目目标及成功标准： | |
| 目标描述： | |
| 假设、风险及障碍： | |
| 客户名单：<br>项目提出人：<br>项目决策人：<br>项目相关人员： | |
| 审批人意见：<br>签名：　　日期： | |

## 软件项目计划(ISO 标准)

编者说明：

拿破仑说过：“没有一场战役是按照计划打的，而胜利的战役没有一个是没有计划的。”，

战役尚且如此，软件项目也不例个。一个经过周密考虑，团队协作共同制订的项目计划是成功的关键。本文档模板是 ISO 标准模板，虽然时间有点久远，但还是十分有参考价值的。

1. 引言
    1.1 编写目的
        [说明编写这份项目开发计划的目的，并指出预期的读者。]
    1.2 背景
        a. 待开发软件系统的名称；
        b. 本项目的任务提出者、开发者、用户及实现该软件的计算中心或计算机网络；
        c. 该软件系统同其他系统或其他机构的基本的相互来往关系。
    1.3 定义
        [列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]
    1.4 参考资料
        [列出用得着的参考资料。]
2. 项目概述
    2.1 工作内容
        [简要地说明在本项目的开发中须进行的各项主要工作。 ]
    2.2 主要参加人员
        [扼要地说明参加本项目开发工作的主要人员的情况，包括他们的技术水平。]
    2.3 产品
        2.3.1 程序
            [列出需移交给用户的程序的名称、所用的编程语言及存储程序的媒体形式，并通过引用有关文件。逐项说明其功能和能力。]
        2.3.2.文件
            [列出需移交给用户的每种文件的名称及内容要点。]
        2.3.3.服务
            [列出需向用户提供的各项服务。 ]
        2.3.4.非移交的产品
            [说明开发集体应向本单位交出但不必向用户移交的产品。 ]
    2.4 验收标准
        [对于上述这些应交出的产品和服务，逐项说明或引用资料说明验收标准。]
    2.5 [完成项目的最迟期限]
    2.6 [本计划的批准者和批准日期]
3. 实施计划
    3.1 工作任务的分解与人员分工
        [对于项目开发中需完成的各项工作，从需求分析、设计、实现、测试直到维护，包括文件的编制、审批、打印、分发工作，用户培训工作，软件安装工作等，按层次进行分解，指明每项任务的负责人和参加人员。]
    3.2 接口人员
        [说明负责接口工作的人员及他们的职责。]
    3.3 进度
        [对于需求分析、设计、编码实现、测试、移交、培训和安装等工作，给出每项工作任务的预定的开始日期、完成日期及所需资源，规定各项工作任务完成的先后顺序以及表征每项工作任务完成的标志性事件。]
    3.4 预算

[逐项列出本开发项目所需要的劳务以及经费的预算和来源。]

3.5 关键问题

[逐项列出能够影响整个项目成败的关键问题、技术难点和风险，指出这些问题对项目的影响。]

4.支持条件

[说明为支持本项目的开发所需要的各种条件和设施。]

4.1 计算机系统支持

[逐项列出开发中和运行时所需的计算机系统支持，包括计算机、外围设备、通讯设备、模拟器、编译程序、操作系统、数据管理程序包、数据存储能力和测试支持能力等，逐项给出有关到货日期、使用时间的要求。]

4.2 需由用户承担的工作

[逐项列出需要用户承担的工作和完成期限，包括需由用户提供的条件及提供时间。]

4.3 需由外单位提供的条件

[逐项列出需要外单位分合同承包者承担的工作和完成的时间。]

5.专题计划要点

[说明本项目开发中需制订的各个专题计划的要点。]

## 软件项目计划模板(2)

编者说明：

大家可能都发现了 ISO 标准的项目计划缺少实用性，那是因为其未能很好地与 WBS、甘特图技术实现良好的结合。该文档模板则充分考虑到这一点，其简单、实用，适用于中小规模项目。

**删除的内容：** *注：一个简单实用的版本*

1.引言

1.1 计划的目的

1.2 项目的范围和目标

1.2.1 范围描述

1.2.2 主要功能

1.2.3 性能

1.2.4 管理和技术约束

2.项目估算

2.1 使用的历史数据

2.2 使用的评估技术

2.3 工作量、成本、时间估算

3. 风险管理战略

3.1 风险识别

3.2 有关风险的讨论

3.3 风险管理计划

3.3.1 风险计划

3.3.2 风险监视

3.3.3 风险管理

4.日程

4.1 项目工作分解结构

## 软件项目计划模板(3)

批注[XuFeng2]： 注，来源于《软件项目管理》一书。

编者说明：

如果项目规模较大，除了上一个模板中的内容之外，还应该加入许多分支内容，包括过程计划、组织计划、测试计划、变更及管理计划、文档计划等各多方面的问题，将这些内容的细化，将使项目计划更全面、更周密。

**删除的内容：** *注：一个较全面、现代的版本*

第 1 部分 概述

1.1 目标

[这部分的目标是总结整个项目计划。]

1.2 概述

[简要描述要做的工作。给出所有理解工作环境所需的背景。然后阐述在合同下的项目任务。紧接着，说明项目如何组织。然后，在项目的基础上列出假设和约束。]

1.3 详述

[说明项目的总体时间进度。包括项目中的所有主要工作，无论是你能控制的还是不能控制的。如果你计划发布多个版本，要说明如何安排进度。]

第 2 部分 过程计划

2.1 目标

[这部分的目标是对用一系列称为“过程”的时间段对开发活动加以定义，也就是确定该项目的开发将选用什么样的过程模型。]

2.2 概述

[定义你的开发生命周期，并且简要说明生命周期的每个过程。]

2.3 详述

2.3.1 定义过程

[主要目标：分析问题、制作项目计划、定义接收标准、选择项目工具。]

[次要目标：寻找人员、了解客户、形成试验性的设计思想。]

2.3.2 设计过程

[主要目标：设计操作性程序、设计支持性程序、改进项目计划、进行项目评审。]

[次要目标：准备集成环境、建立变更管理、制作模拟模型、为下一个过程寻找人员、准备程序员培训、出版程序员手册、初步准备系统测试、验收测试、现

场测试、建立项目资料库。]

2.3.3 编码过程

[主要目标：详细设计/编码和模块测试、模块集成、文档建立。]

[次要目标：详细地准备系统测试、验收测试、现场测试，准备客户培训、准备移植。]

2.3.4 系统测试过程

[主要目标：根据问题说明书进行系统测试、尽可能地“实况”测试、通过非程序开发人员测试。]

[次要目标：完成验收测试准备、培训客户、更新描述性文档、完成用户文档、再次分配人员。]

2.3.5 验收过程

[主要目标：执行和分析验收测试、签署正式的接收协议。]

[次要目标：完成客户培训、清理文档。]

2.3.6 移植过程

[主要目标：协助进行数据转换、建立数据转换标准、建立全面恢复计划、定义移植顺序、协助接入。]

[次要目标：与受影响组进行联系、支持评审过程。]

2.3.7 运行过程

[主要目标：协助初期运行。]

[次要目标：现场测试、继续维护和调整、评价项目。]

第 3 部分 组织计划

3.1 目标

[这部分的目标是定义项目的组织以及责任分配。]

3.2 概述

[说明建立组织的基本原因，画出组织内部的主要工作流程图，从问题的分析和设计开始，包括编码、测试、制作文档和交付。]

3.3 详述

[在每个子部分中，列出基于组织章程的部分以及每个部分的责任，然后再说明在每个过程中组织的结构图。]

3.3.1 部门及责任

[分析和设计部：编写问题说明书、设计说明书、变更管理、数据控制、模拟模型、制作用户文档、协作集成测试。]

[编程部：详细设计、编码、模块测试、集成测试、描述性文档。]

[测试部：制作系统测试说明书、制作验收和现场测试说明书、收集和制造测试数据、选择和获得测试工具、建立测试资料库、安排测试资源进度、执行测试、分析测试结果、制作测试结果文档。]

[行政部：资料管理、计算机时间控制、计划和安装终端和 PC、发放程序员手册、培训、特殊技术协助、技术联络、文档控制、报告控制、合同变更管理、提供杂务支持、维护项目历史信息。]

3.3.2 组织章程

第 4 部分 测试计划

4.1 目标

[这部分的目标是定义对软件系统的所有级别测试的工具、过程和责任。]

4.2 概述

[简要定义每个测试级别，并说明在一个测试层次上，不同级别如何组合在一起。]

4.3 详述

4.3.1 单元测试

[在与其它功能模块集成之前，针对单个程序模块的测试。在此应列出单元测试的目标、责任、过程、工具。]

4.3.2 集成测试

[逐步将通过测试的模块集成为更加复杂的集合，并且测试这些集合，直到整个软件都被集合在一起。在此应列出集成测试的目标、责任、过程、工具。]

4.3.3 系统测试

[在尽可能真实的环境下，重新测试完成的软件系统，应由非编程人员完成。在此应列出系统测试的目标、责任、过程、工具。]

4.3.4 验收测试

[在用户认可的条件下，试运行系统以验证系统满足了客户的需求。在此应列出验收测试的目标、责任、过程、工具。]

4.3.5 现场测试

[在不同的运行环境下测试软件系统，以确保运行准备就绪，这并不是每个项目都需要的。在此应列出现场测试的目标、责任、过程、工具。]

4.3.6 共同测试设备

[描述在几个或者所有级别的测试中共同的设备和工具，其中包括系统资料、计算机设备、桌面系统、操作系统、特殊语言、CASE 工具、仿真器。]

4.3.7 测试支持程序

第 5 部分 变更管理计划

5.1 目标

[这部分的目标是定义在软件系统开发过程中，变更控制的过程。]

5.2 概述

[描述建立你和客户都能够接受的关键基线文档以及控制与这些基线变化相关事件的需求。无论何时发生问题，基线文档都是参考的关键。]

5.3 详述

5.3.1 基线

[定义哪些文档在你的项目中是基线。]

5.3.2 变更申请

[列出可能会提出变更的人员类别，以及提供相应的变更申请文档。]

5.3.3 研究变更申请

5.3.4 变更的类型

[根据变更的基线影响的程序，设置不同的变更类型。]

5.3.5 变更管理会议

[明确变更管理会议的组成成员、召开时间以及具体的操作办法。]

5.3.6 建议类型

[定义变更建议的类型，通常包括接受和拒绝两种。]

5.3.7 执行变更

[定义执行变更的具体方法，通常包括评估变更成本、对变更进行审批、制作变更文档、对变更后的进度进行重新安排、测试变更结果。]

第 6 部分 文档计划

6.1 目标

[这部分的目标是定义出版周期所要求过程与资源，以及列出基础项目文档组的框架结构。]

6.2 概述

[强调所有的项目文档在这部分都列出结构框架。]

6.3 详述

6.3.1 发布过程和责任

[通常包括准备和批准、打字输入、校对和编辑、翻印、发放、电子存储等。]

6.3.2 项目文档大纲

[每个文档的都包括以下部分：]

[a.项目标志：用于标识项目文档之用；]

[b.文档名称：标识主题，如问题说明书、设计说明书……]

[c.文档编号：由项目资料员分配给文档的唯一标识；]

[d.批准：在作为正式版本之前，文档所需批准人的姓名。当然也不是所有文档都需要经过批准。]

[e.发行日期]

[f.文档主体：文档的内容。]

6.3.3 文档内容

[列出在该项目中将要使用的文档模板的结构性内容。]

**删除的内容：** *注：一个符合现代软件工程思想的版本*

## 软件项目计划模板(4)

**批注[XuFeng3]：** 注，对 RUP 模板的修改

编者说明：

随着现代软件工程思想的普及，迭代的、增量的开发生命周期已经被认识并付诸实践，针对这样的生命周期，其项目计划的格式也需要做出相应的调整。

1. 文档概述

[在此对整个文档进行概要性描述，另外还应列出该计划的目标、范围、定义、术语、参考资料等内容。]

1.1 目标

[在此描述本项目计划的目标。]

1.2 范围

[简要说明该计划所覆盖的范围，以及与其相关的项目，与该文档有联系的事物。]

1.3 定义与术语

[在此列出在该计划中所涉及的所有术语、定义、缩写词的解释，这些信息也可以引用项目词汇表来提供。]

1.4 参考资料

[在此应列出项目计划中引用的文档列表，对于引用的每个文档都应该列出其标题、文档编号、日期，并且指出这些文档的来源，以方便该计划的阅读者查找。]

1.5 概述

[说明该计划其它部分所包含的内容，以及文档的组织方式。]

2. 项目概述

2.1 项目目标

[指出该项目将会交付什么样的产品，能够帮助客户达到什么目标。]

2.2 假设与约束

[列举出制定该计划时所做的所有假设，以及列举出对该项目的解决方案的约束性要

求，如特定的操作系统平台、特定的时间、特定的经费范围等。]

2.3 项目交付物

[具体列出该项目完成后，将交付哪些东西，并可以列出每个交付时间。]

2.4 项目计划更新总结

[建议采用表格的形式，将计划的修订过程列出来。]

3. 项目组织

3.1 项目组织结构

[建议使用组织结构图的形式，将整个项目团队成员之间的关系与职责明确下来，甚至可以包括管理人员、各种委员会等。]

3.2 外部联系人

[列出开发组织之外的，所有与项目相关的外部人员的姓名、联系电话等资料。]

3.3 角色与职责

[明确项目开发各个任务的负责人或小组。]

4. 项目管理计划

4.1 项目估计

[给出关于项目成本、进度的估计值，这些估计值将是项目计划制定的基础，也是今后重新评估、修改计划的基础。你可以采用任何估算技术。]

4.2 项目计划

4.2.1 阶段计划

[主要包括工作结构分解（WBS）、显示各个阶段或迭代时间安排的甘特图、主要里程碑与其验收标准。]

4.2.2 迭代目标

[如果你采用的是迭代式的开发方法，那么在此列出每次迭代的计划，以及每次迭代计划实现的目标。]

4.2.3 发行计划

[列出软件开发过程中各个中间版本的发行时间，包括演示版、Alpha 版、Beta 版等。]

4.2.4 项目进度表

[使用甘特图或 PERT 图等方法，表示出该项目的进度计划。]

4.2.5 项目资源计划

[在此处应列出项目所需的人员、设备等资源情况。应指明所需人员的数量、技能要求，以及如何获取这些资源，是否要对人员进行必要的培训等。]

4.2.6 项目预算

[根据 WBS 和阶段计划分配成本，得到本项目的财务预算。]

4.3 迭代计划

[根据 4.2.2 小节的目标，具体列出每次迭代的详细计划。该部分可以视需要将其单列为专题计划。]

4.3.1 迭代一

4.3.1.1 计划

[列出此次迭代的时间线、小型里程碑等。]

4.2.1.2 资源

[列出此次迭代所需的人力、财力、设备等资源。]

4.2.1.3 用例

[列出此次迭代将要实现的用例。]

4.2.1.4 评估标准

[列出此次迭代的各项评测标准，包括功能、性能、容量、质量等。]

4.4 项目监督与控制

4.4.1 需求管理计划

[有针对性对制定各类需求元素的管理与跟踪办法。该部分可以视需要将其单列成为专题计划。]

4.4.2 进度控制计划

[说明如何对项目计划执行情况进行监控，将采用什么措施与管理手段。]

4.4.3 预算控制计划

[说明如何对项目的财务预算进行控制，以保证成本最小化。]

4.4.4 质量控制计划

[说明如何保证项目的质量，以及一些应急的应对措施。该部分可以视需要将其单列成为专题计划。]

4.4.5 报告计划

[说明项目开发过程中，整个项目团队的报告机制，什么时候、谁、报送什么数据，从而形成规则。]

4.4.6 评测计划

[制定项目开发过程中将要度量与评测的指标，说明如何评测，如何应对。该部分可以视需要将其单列成为专题计划。]

4.5 风险管理计划

[该部分可以视需要将其单列为专题计划。]

4.5.1 风险总述

[对项目所涉及的风险进行一个概要性描述。]

4.5.2 风险管理任务

[简要地说明在该项目中，风险管理所涉及的内容，可以包括用来确定风险的方法、对风险列表进行分析和确定优先级的方式、将采用的风险管理策略、对最严重的风险所计划的降低/规避或预防的策略、监测风险状态的方式、风险复审的时间表。]

4.5.3 风险管理的组织和职责

[列出与风险管理相关的个人或小组，并对其职责进行描述。]

4.5.4 工具与技术

[列出与风险管理将采用的工具软件或技术。]

4.5.5 纳入管理的风险项

[列出主要的风险项，并描述其影响以及应急措施。具体可以参考后面的《风险条目跟踪表模板》。]

4.6 收尾计划

[列出在项目后期将要做的事，包括材料存档、汇报总结等。]

5. 相关技术

5.1 开发案例

[给出本项目将采用的软件生命周期模型、过程规范等，从而对开发过程给予明确的指导。该部分可以视需要将其单列为一个专题文件。]

5.2 方法、工具和技术

[列出本项目中将运用的方法、工具和技术，并给出适当的工作指南和说明。]

5.3 产品验收计划

[列出本项目验收工作的一些细节计划，本部分内容可以视需要将其单列为一个专题计划。]

6．其它支持过程管理

6.1 配置管理计划

[在此列出该项目所采用的配置管理过程，通常是单列为一个专题。]

6.2 评估计划

[列出本项目评估时所使用的技术、标准、指标和过程。这里的评估包括走查、检查和复审。]

6.3 文档计划

6.4 质量保证计划

6.5 分包商管理计划

7.其他计划

8.附录

9.索引

## 风险条目跟踪表模板

编者说明：

对于中型以上的项目，风险控制的意义就犹为突出。要控制风险，就应该找到风险，并将风险记录下来，确定相关责任人，对于风险性高的、可能性大的还需要制订相关的应对措施。而最好的方法就是整理成为本模板中的表格，为每个潜在风险备个案。

| | |
|---|---|
| 序列号 | *<顺序号>* |
| 确定日期 | *<风险被识别出的日期>* |
| 撤消日期 | *<撤消风险确定日期>* |
| 描述 | *<以"条件-结果"的形式描述风险>* |
| 可能性 | *<风险转变为问题的可能性>*<br>*注：可用 0.1(极不可能)~1.0(肯定发生)来表示* |
| 影响 | *<如果风险变成了事实奖造成的损失>*<br>*注：可用 1(无甚么影响)~10(有很深、很大的影响)来表示* |
| 危害值 | *<可能性＊影响>* |
| 降低风险计划 | *<一种或多种用来控制、避免、最小化及降低风险的方法>* |
| 负责人 | *<解决风险的责任承担者>* |
| 截止日期 | *<完成降低风险措施的截止日期>* |

## 进度计划风险列表

批注[XuFeng4]： 注，来源于《快速软件开发》一书。

编者说明：

准确来说，本列表不是一个文档模板，而是一个参考文章。由于风险识别许多人都觉得无从入手，下面就是列出了与进度相关的风险条目，对于风险识别有很大的参考价值。

1.最常见的进度计划风险

1） 功能无限蔓延；

2） 需求镀金或开发人员镀金；

3） 质量不定

4） 计划过于乐观

5） 设计欠佳
6） 银弹综合症
7） 研发导向开发
8） 人员薄弱
9） 签约商失败；
10）研发人员与客户的磨擦。

2.进度计划风险完整列表

2.1 计划编制风险

1） 计划、资源和产品定义全凭客户或上层领导口头指令，并且不完全一致；
2） 计划是优化的，是“最佳状态”；
3） 计划忽略了必要的任务；
4） 计划基于使用特定的小组成员，而那个小组成员其实指望不上。
5） 在限定的时间内无法建成已定规模大小的产品；
6） 产品规模比估计的要大一些；
7） 工作量大于估算数；
8） 进度已经拖延的项目在重新评估时过于优化或忽视项目历史；
9） 过度的进度压力造成生产率下降；
10）目标日期提前，但没有相应地调整产品范围或可用资源；
11）一个任务的延迟导致相关任务的连锁反应；
12）涉足不熟悉的产品领域，花费在设计和实现上的时间比预期的要多。

2.2 组织和管理

1） 项目缺乏一个有凝聚力的最高领导人；
2） 由于前期乏力，项目长时间被搁置；
3） 解雇和削减开支导致项目小组能力下降；
4） 仅由管理层或市场人员进行技术决策，导致计划进度延长；
5） 低效的项目组结构降低生产率；
6） 管理层审查/决策的周期比预期时间长；
7） 预算削减打乱项目计划；
8） 管理层做出了打击项目组织积极性的决定；
9） 非技术的第三方的工作比预期延长（如审批，采购等）；
10）计划性太差，无法适应期望的开发速度；
11）项目计划由于压力而放弃，导致开发混乱、低效；
12）管理层强调英雄主义，而忽视客观确切的状态报告，这会降低发现和改正问题的能力。

2.3 开发环境

1） 设施没有及时到位；
2） 设施到位，但不配套；
3） 设施拥挤、杂乱或者破损；
4） 开发工具未能及时到位；
5） 开发工具不如期望那样有效，开发人员需要时间创建工作环境或切换新的工具；
6） 开发工具的选择不是基于技术需求，不能提供计划要求的性能；
7） 新开发工具的学习期比预期的长，内容繁多。

2.4 最终用户

1） 最终用户坚持新的需求；
2） 最终用户对于最后交付的产品不满意，要求重新设计和重做；
3） 最终用户不买进项目产品，无法提供后续支持；
4） 最终用户的意见未被采纳，造成产品最终无法满足用户期望，而必须重做。

2.5 客户

1） 客户坚持新的需求；
2） 客户对规划、原型和规格的审核/决策周期比预期长；
3） 客户没有或不能参与规划、原型和规格阶段的审核，导致需求不稳定和耗时的重复；
4） 客户答复的时间比预期长（如回答需求中需澄清的问题）；
5） 客户坚持技术决策而导致进度计划延长；
6） 客户对开发进度管理过细，导致实际进展变慢；
7） 客户提供的组件无法与开发的产品匹配，导致额外的设计和集成工作；
8） 客户提供的组件质量欠佳，导致额外的测试、设计和集成工作，以及额外的客户关系管理工作；
9） 客户要求的支持工具和环境不兼容、性能差或者功能不完善，导致生产率降低；
10）客户不接受交付的软件，尽管它满足了所有的规格；
11）客户期望的开发速度是开发人员无法达到的。

2.6 承包商

1） 承包商没有按承诺交付组件；
2） 承包商递交的组件质量低下无法接收，必须花时间改进质量；
3） 承包商没有买进项目开发需要的工具，进而无法提供需要的性能水平。

2.7 需求

1） 需求已经成为项目基准，但变化还在继续；
2） 需求定义欠佳，而进一步的定义会扩展项目范畴；
3） 添加额外的需求；
4） 产品定义含混的部分比预期需要更多的时间。

2.8 产品

1） 错误发生率高的模块需要比预期更多的测试、设计和实现工作；
2） 校正质量低下不可接受的产品，需要比预期更多的测试、设计和实现工作。
3） 在一个或多上新兴领域推广计算机技术使得计划进度的延长不可预期；
4） 由于软件功能的错误，需要重新设计和实现；
5） 开发额外不需要的功能（镀金）延长了计划进度；
6） 要满足产品规格与速度要求，需比预期更多时间，包括重新设计和实现的时间；
7） 严格要求与现有系统兼容，需要进行比预期更多的测试、设计和实现工作；
8） 要求与其他系统、复杂系统或不受本项目控制的系统相连，导致无法预料的设计、实现和测试工作。
9） 要求在不同操作系统下运行将花费比预期更长的时间；
10）在不熟悉或未经检验的软（硬）件环境中运行产生未预料的问题；
11）开发一种对组织全新的模块将比预期花费更长的时间；
12）依赖正在开发中的技术将延长计划进度。

2.9 外部环境

1） 产品依赖政府规章，而规章的改变将是不可预期的；
2） 产品依赖草拟中的技术标准，而最后的标准将是不可预期的。

2.10 人员

1） 招聘人员所花时间比预期的长；

2） 作为先决条件的任务不能按时完成（如培训、其它项目）；

3） 开发人员和管理层之间关系不佳导致决策缓慢，影响全局；

4） 项目组成员没有全身心投入项目，进而无法达到需要的产品性能水平；

5） 缺乏激励措施，士气低下，降低了生产能力；

6） 缺乏必要的规范，增加了工作失误与重复工作；

7） 某些人需要更多时间适应不熟悉的软件工具和环境、硬件环境、编程语言；

8） 项目结束前，合同制人员离开团队，或雇员辞职；

9） 项目后期加入新的开发人员，额外的培训和沟通降低现有成员的效率；

10）项目组成员不能有效地一起工作；

11）由于项目组成员间的冲突，导致沟通不畅、设计欠佳、接口错误和额外的重复工作；

12）有问题的成员没有调离项目组，损害了项目组其他成员的积极性；

13）项目的最佳人选未加入项目组；

14）项目的最佳人选已加入项目组，但因其他原因未能合理使用；

15）没有找到项目急需的具有特定技能的人；

16）关键人物只能兼职参与；

17）项目人员不足；

18）任务的分配与人员技能不匹配；

19）人员工作的进展比预期的慢；

20）项目管理人员怠工导致计划和进度失效；

21）技术人员怠工导致工作遗漏或质量低下，工作需要重做。

2.11 设计与实现

1） 设计过于简单，无法确定主要事件，并导致重新设计和实现；

2） 设计过于复杂，导致一些不必要的工作，影响实现效率；

3） 设计质量低下，导致重复设计和实现

4） 使用不熟悉的方法，导致额外的培训时间，并重犯前期使用这种方法时导致的错误；

5） 产品采用低级语言来实施，导致生产率比预期的低；

6） 一些必要的功能无法使用现有的代码和库实现，开发人员必须使用新库或自选开发所要的功能；

7） 代码和库质量低下，导致需要额外的测试、错误修正或重做；

8） 过高估计了增强型工具对计划进度的节省量；

9） 分别开发的模块无法有效集成，需要重新设计或重做。

2.12 过程

1） 大量的纸面工作导致进程比预期的慢；

2） 进程跟踪不准确，导致无法预知项目是否已落后于计划进度；

3） 前期的质量保证行为不真实，导致后期的重复工作；

4） 质量跟踪不准确，导致无法得知影响进度的质量问题；

5） 太不正规，导致沟通不足，质量问题和工作重做；

6） 过于正规，导致过多耗时无用的工作；

7） 向管理层撰写进度报告占用的开发人员的时间比预期的多；

8） 风险管理粗心，导致没有发现重大的项目风险；

9） 软件项目风险管理花费的时间比预期的多。

## 开发进度月报(ISO 标准)

编者说明：

计划需要跟踪进度来进行适当的调整，因此在开发组织内应该形成良好的进度汇报机制，ISO 标准模板也对这一块提供了参考。这一文档格式十分全面，不过也略显繁琐，适合于中型以上项目。

l. 标题

| |
|---|
| 开发中的软件系统的名称和标识符<br>分项目名称和标识符<br>分项目负责人签名<br>本期月报编写人签名<br>本期月报的编号及所报告的年月 |

2．工程进度与状态

2.1 进度

[列出本月内进行的各项主要活动，并且说明本月内遇到的重要事件，这里所说的重要事件是指一个开发阶段（即软件生存周期内各个阶段中的某一个，例如需求分析阶段）的开始或结束，要说明阶段名称及开始（或结束）的日期。]

2.2 状态

[说明本月的实际工作进度与计划相比，是提前了、按期完成了、或是推迟了？如果与计划不一致，说明原因及准备采取的措施。]

3．资额耗用与状态

3.1 资额耗用

[主要说明本月份内耗用的工时与机时。]

3.1.1 工时

[分为三类：]

[a. 管理用工时　包括在项目管理（制订计划、布置工作、收集数据、检查汇报工作等）方面耗用的工时；　]

[b. 服务用工时　包括为支持项目开发所必须的服务工作及非直接的开发工作所耗用的工时；]

[c. 开发用工时　要分各个开发阶段填写。]

3.1.2 机时

[说明本月内耗用的机时，以小时为单位，说明计算机系统的型号。]

3.2 状态

[说明本月内实际耗用的资源与计划相比，是超出了、相一致、还是不到计划数？如果与计划不一致，说明原因及准备采取的措施。]

4　经费支出与状态

4.1 经费支出

4.1.1 支持性费用

[列出本月内支出的支持性费用，一般可按如下七类列出，并给出本月支持费用的总和：]

[ a. 房租或房屋折旧费；]

[b. 员工工资、奖金、补贴；]

[c. 培训费包括给教师的酬金及教室租金；]
[d. 资料费包括复印及购买参考资料的费用；]
[e. 会议费召集有关业务会议的费用；]
[f. 旅差费；]
[g. 其他费用。]

4.1.2 设备购置费

[列出本月内支出的设备购置费，一般可分如下三类：]
[[a. 购买软件的名称与金额；]
[b. 购买硬设备的名称、型号、数量及金额；]
[c. 已有硬设备的折旧费。]

4.2 状态

[说明本月内实际支出的经费与计划相比较，是超过了。相符合、还是不到计划数？如果与计划不一致，说明原因及准备采取的措施。]

5．下个月的工作计划

6．建议

[本月遇到的重要问题和应引起重视的问题以及因此产生的建议。]

## 开发任务卡

编者说明：

项目中应该实现责任到人，项目的进度应该是每个项目成员个人进度表的总汇集，而开发任务卡则是项目与项目成员的约定，也是项目管理的一个好办法。大家可以根据自己的实际情况来修改该模板。

项目名：_____ 模块/类名：_____

安排时间：_____ 任务承担人：_____

相关模块/类情况：

| 模块/类名 | 负责人 | 开始时间 | 完成时间 | 状态 |
|---|---|---|---|---|
| | | | | |

任务描述：

估计完成时间：_____ 批准人：_____

## 个人开发进度月报

编者说明：

表格式的进度报表能够节省制作时间，缩短进度误差。对于中型以上项目，特别是成员的任务超过了 1 个月，那么让每个开发人员填写进度月报就是一个很好的管理办法。当然，如果成员的任务都较小，则无需使用该文档，只需对工作任务卡进行检查就可以了。

1．标题

| |
|---|
| 项目名称及标识： |
| 子项目名称及标识： |
| 开发阶段： |

| |
|---|
| 报告时间：　　年　　月　　日　至　　年　　月　　日 |
| 报告人：〈签名〉 |

2．进度

2.1 任务

| 任务：<任务名> |
|---|
| 任务描述： |
| 状态：　　☐完成　　☐未完成 |
| 与计划比较：　　☐提前　　☐按期　　☐推迟 |
| 推迟原因： |

3．资源耗费

| 总用工时： |
|---|
| 加班时间： |
| 机时： |
| 上网时间： |
| 硬件平台： |
| 软件环境和工具： |

4.下个月工作计划

| 任务：<任务名> |
|---|
| 任务描述： |
| 任务所属项目或子项目： |
| 性质：　　☐新　　☐续上月 |

5.建议

# 项目开发进度月报

编者说明：

项目进度月报是必须的管理机制，而长篇大论不仅浪费了大家的时间，而且也使得进度的收集与实际情况有一些时间上的误差，因而可以采用表格化的报表格式。

1.标题

| |
|---|
| 项目名称及标识： |
| 子项目名称及标识： |
| 本期月报编写人：〈签名〉 |
| 子项目负责人：〈签名〉 |
| 本期月报编号： |
| 月报日期：　　年　　月　　日 |

2. 进度

2.1 任务

| |
|---|
| 任务：〈任务名〉 |
| 任务描述： |
| 状态：　　□ 完成　　□未完成 |
| 与计划比较：　　□提前　　□按期　　□推迟 |
| 推迟原因： |

2.2 事件

| |
|---|
| 事件：〈事件名〉 |
| 事件标志： |
| 与计划比较：　　□提前　　□按期　　□推迟 |
| 推迟原因： |

3. 资源耗费

3.1 工时

| |
|---|
| 管理用工时： |
| 服务用工时： |
| 开发用工时： |
| 总　　计： |

3.2 机时

| | |
|---|---|
| 计算机类型： | 用时： |
| 计算机类型： | 用时： |
| 计算机类型： | 用时： |
| 总　　计： | 用时： |

4. 经费支出

4.1 支持性经费支出

| |
|---|
| 工资、奖金、补贴： |
| 培训费： |
| 资料费： |
| 会议费： |
| 差旅费： |
| 总计： |

4.2 设置购置费

| 设备名称 | 型号 | 数量 | 单价 | 金额 |
|---|---|---|---|---|
| | | | | |
| | | | | |
| 总计金额： | | | | |

5.下个月工作计划

5.1 任务

| |
|---|
| 任务：〈任务名〉 |
| 任务描述： |
| 开发阶段： |
| 性质：　　□新　　□续上月 |

5.2 事件

| |
|---|
| 事件：〈事件名〉 |
| 事件标志： |
| 性质：　□新　　□旧 |

6.建议

# 项目进度周报

编者说明：

月报通常需要较详细，而周报则应该更简洁，每周让项目经理花上 1-2 分钟将一周的项目进度情况做一个通报是很必要的。本文档模板就是一个例子，供大家参考。

周期：2003 年__月__日~2003 年__月___日

项目名称：_____ 项目编号：_____
项目经理：_____ 项目发起人：_____
项目成员：_____

项目计划开始时间：_____ 项目实际开始时间：_____
项目预计完成时间：_____ 现在预计完成时间：_____

项目处于： ☐ 初步计划阶段 ☐ 需求分析阶段 ☐ 开发阶段
项目状态： ☐ 按计划进度 ☐ 超计划进度 ☐ 进度延迟

项目预计投入人力：_____人/日 现在已投入人力：_____人/日
预计共需投入人力：_____人/日

项目遇到的困难和要解决的问题：

_____
_____

## 项目开发总结报告(GB 标准)

编者说明：

在项目中犯错误是正常的，但是犯同样的错误则是不可原谅的。因此，我们应该善于在项目中总结、在实践中总结。在项目结束的时候，所有的成员汇集在一起，回顾一下项目的过程，总结出错误，找到解决的办法，总结出经验，将这些经验复用到下一个项目中。然后形成本文档，共享给大家。

1．引言

1.1 编写目的

[说明编写这份项目开发总结报告的目的，指出预期的阅读范围。]

1.2 背景

[说明：]

[ a. 本项目的名称和所开发出来的软件系统的名称；]

[b. 此软件的任务提出者、开发者、用户及安装此软件的计算中心。]

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出要用到的参考资料，如：]

[a. 本项目的已核准的计划任务书或合同、上级机关的批文；]

[b. 属于本项目的其他已发表的文件；]

[c. 本文件中各处所引用的文件、资料，包括所要用到的软件开发标准。]

[列出这些文件的标题、文件编号、发表日期和出版单位，说明能够得到这些文件

资料的来源。]

2．实际开发结果

2.1 产品

[说明最终制成的产品，包括：]

[a. 程序系统中各个程序的名字，它们之间的层次关系，以千字节为单位的各个程序的程序量、存储媒体的形式和数量；]

[b. 程序系统共有哪几个版本，各自的版本号及它们之间的区别；]

[c. 每个文件的名称；]

[d. 所建立的每个数据库。如果开发中制订过配置管理计划，要同这个计划相比较。]

2.2 主要功能和性能

[逐项列出本软件产品所实际具有的主要功能和性能，对照可行性研究报告、项目开发计划、功能需求说明书的有关内容，说明原定的开发目标是达到了、未完全达到、或超过了。]

2.3 基本流程

[用图给出本程序系统的实际的基本的处理流程。]

2.4 进度

[列出原定计划进度与实际进度的对比，明确说明，实际进度是提前了、还是延迟了，分析主要原因。]

2.5 费用

[列出原定计划费用与实际支出费用的对比，包括：]

[a. 工时，以人月为单位，并按不同级别统计；]

[b. 计算机的使用时间，区别 CPU 时间及其他设备时间；]

[c. 物料消耗、出差费等其他支出。]

[明确说明，经费是超出了、还是节余了，分析其主要原因。]

3．开发工作评价

3.1 对生产效率的评价

[给出实际生产效率，包括：]

[a. 程序的平均生产效率，即每人月生产的行数；]

[b. 文件的平均生产效率，即每人月生产的千字数；]

[并列出原订计划数作为对比。]

3.2 对产品质量的评价

[说明在测试中检查出来的程序编制中的错误发生率，即每千条指令（或语句）中的错误指令数（或语句数）。如果开发中制订过质量保证计划或配置管理计划，要同这些计划相比较。]

3.3 对技术方法的评价

[给出对在开发中所使用的技术、方法、工具、手段的评价。]

3.4 出错原因的分析

[给出对于开发中出现的错误的原因分析。]

4．经验与教训

[列出从这项开发工作中所得到的最主要的经验与教训及对今后的项目开发工作的建议。 ]

编者说明：

当一个项目完成之后，应该将所有的文档、源程序、可执行文档进行整理打包，统一入库，而模块开发卷宗则是这些文档的封面。有了该文档，就可以使得下次找这些资料时更加方便。

第 1 章　模块开发情况

| | | |
|---|---|---|
| 模块名: | | 模块标识符 |
| 代码设计 | 计划开始日期 | 实际开始日期 |
| | 计划完成日期 | 实际完成日期 |
| 模块测试 | 计划开始日期 | 实际开始日期 |
| | 计划完成日期 | 实际完成日期 |
| 组装测试 | 计划开始日期 | 实际开始日期 |
| | 计划完成日期 | 实际完成日期 |
| 源代码行 | 预计行数 | 实际行数 |
| 目标模块大小 | 预计字节数 | 实际字节数 |
| 代码复查　（日期/签字） | | |
| 批　　准　（日期/签字） | | |

第 2 章　功能说明

| 输入 | 处理 | 输出 |
|---|---|---|
| | | |

第 3 章　设计说明

3.1　层次说明

| 模块名 | | 模块标识符 | |
|---|---|---|---|
| 调用模块 | | | |
| 被调用模块 | | | |

3.2　算法　（Ｎ－Ｓ图、ＰＡＤ图或ＰＤＬ语言）

3.3　外部数据结构

| 数据结构名称 | 关系 |
|---|---|
| | <生成/使用关系> |

3.4　出错信息

| 错误编号 | 错误名 | 描述 |
|---|---|---|
| | | |

第 4 章　源代码清单

第 5 章　测试说明

5.1　测试名称 1

| 测试标识符： | | 编号： | |
|---|---|---|---|
| 测试目的： | | | |

| 测试配置： | | | |
|---|---|---|---|
| 测试用例： | | | |
| 序号 | 输入 | 预期输出 | 实际输出 |
| | | | |
| | | | |

5.2 测试名称 2

……

第 6 章 复审结论

6.1 与需求说明的比较

6.2 与概要设计的比较

6.3 与详细设计的比较

6.4 一般结论

# 二、需求分析类

编者说明：

许多有经验的开发团队在开始需求调查的时候，总会将“软件客户需求权利书”和“软件客户需求义务书”提交给客户，让客户明确其权利与义务，将会对需求调研、分析的工作带来意想不到的效果，你可以一试。

**软件客户需求权利书**

1.要求分析人员使用符合客户语言习惯的表达；

2.要求分析人员了解客户系统的业务及目标；

3.要求分析人员组织需求获取期间所介绍的信息，并编写软件需求规格说明。

4.要求开发人员对需求过程中所产生的工作结果进行解释说明；

5.要求开发人员在整个交流过程中保持和维护一种合作的职业态度；

6.要求开发人员对产品的实现及需求都要提供建议，拿出主意。

7.描述产品使其具有易用、好用的特性；

8.可以调整需求，允许重用已有的软件组件；

9.当需要对需求进行变更时，对成本、影响、得失有个真实可信的评估；

10.获得满足客户功能和质量要求的系统，并且这些要求是开发人员同意的。

**软件客户需求义务书**

1.给分析人员讲解业务及说明业务方面的术语等专业问题；

2.抽出时间清楚地说明需求并不断完善；

3.当说明系统需求时，力求准确详细；

4.需要时要及时对需求做出决策；

5.要尊重开发人员的成本估算和对需求的可行性分析；

6.对单项需求、系统特性或使用实例划分优先级；

7.评审需求文档和原型；

8.一旦知道要对项目需求进行变更，要马上与开发人员联系；

9.在要求需求变更时，应遵造开发组织确定的工作过程来处理；

10.尊重需求工程中开发人员采用的流程（过程）。

## 软件项目视图和范围

编者说明：

项目所涉及的内容与所解决的问题都是有限的，而且项目应该是十分有目的性的，是为了实现某个可度量的目标而做的。因此，在需求分析的前期应该将“项目的目标与范围”这一项目的本质文档化，让每一个项目成员对其达成共识。该文档是十分重要，但却又是十分容易被忽视的。该文档模板比较适用于定制开发项目。

1.业务需求

[业务需求说明了提供给客户和产品开发商的新系统的最初利益。不同产品可能会有不同的侧重点。本部分描述了你为什么要从事此项项目的开发，以及它将给开发者和购卖者带来的利益。]

1.1 背景

[在这一部分，总结新产品的理论基础，并提供关于产品开发的历史背景或形势的一般性描述。]

1.2 业务机遇

[描述现存的市场机遇或正在解决的业务问题。描述商品竞争的市场和信息系统将运用的环境。包括对现存产品的一个简要的相对评价和解决方案，并指出所建议的产品为什么具有吸引力和它们所能带来的竞争优势。认识到目前只能使用该产品才能解决的一些问题，并描述产品是怎样顺应市场趋势和战略目标的。]

1.3 业务目标

[用一个定量和可测量的合理方法总结产品总结产品所带来的重要商业利润。关于给客户带来的价值在后面阐述，这里仅把重点放在给业务的价值上。这些目标与收入预算或节省开支有关，并影响到投资分析和最终产品的交付日期。]

1.4 客户或市场需求

[描述一些典型客户的需求，包括不满足现在市场上的产品或信息系统的需求。提出客户目前所遇到的问题在新产品中将可能（或不可能）出现的阐述，提供客户怎样使用产品的例子。确定了产品所能运行的软、硬件平台。定义了较高层次的关键接口或性能要求，但避免设计或实现细节。把这些要求写到列表中，可以反过来跟踪调查特殊用户和功能需求。]

1.5 提供给客户的价值

[确定产品给客户带来的价值，并指明产品怎样满足客户的需要。可以用下列言辞表达产品带给客户的价值：

- 提高生产效率，减少返工；
- 节省开支；
- 业务过程的流水线化；
- 先前人工劳动的自动化；
- 符合相关标准和规则；
- 与目前的应用产品相比较，提高了可用性或减少了失效程度。]

1.6 业务风险

[总结开发（或不开发）该产品有关的主要业务风险，例如市场竞争、时间问题、用户的接受能力、实现的问题或对业务可能带来的消极影响。预测风险的严重性，指明你

所能采取的减轻风险的措施。]

2.项目视图的解决方案

[文档中的这一部分为系统建立了一个长远的项目视图，它将指明业务目标。这一项目视图为在软件开发生存期中作出决策提供了相关环境背景。这部分不包括详细的功能需求和项目计划信息。]

2.1 项目视图陈述

[编写一个总结长远目标和有关开发新产品目的的简要项目视图陈述。项目视图陈述将考虑权衡有不同需求客户的看法。它可能有点理想化，但必须以现有的或所期待的客户市场企业框架。组织的战略方向和资源局限性为基础。]

[如："化学制品跟踪系统"可使科学家查询到化学制品仓库或供应商将提供的化学制品容器。系统可随时了解公司每一个化学制品容器所处的位置，容器中所剩余的药品剂量，任何时候每个容器所处的位置和用法的历史记录。通过充分利用公司内部的可用化学制品，废弃极少量已使用或过期失效的化学制品，使用标准的化学制品的购买过程等将在化学制品上节省 25%开支。"化学制品跟踪系统"还能产生符合政府部门规定所要求的全部报表，包括化学制品的使用、存储和废弃等报表。]

2.2 主要特征

[包括新产品将提供的主要特性和用户性能的列表。强调的是区别于以往产品和竞争产品的特性。可以从用户需求和功能需求中得到这些特性。]

2.3 假设和依赖环境

[在构思项目和编写项目视图和范围文档时，要记录所作出的任何假设。通常一方所持的假设应与另一方不同。如果你把它们都记录下来，并加以评论，就能对项目内部隐含的基本假设达成共识。比如，"化学制品跟踪系统"的开发者假设：该系统可以替代现有的仓库存货系统，并能与有关采购部门的应用相连接。把这些都记录下来以防止将来可能的混淆和冲突。还有，记录项目所依赖的主要环境，比如：所使用的特殊的技术、第三方供应商、开发伙伴及其它业务关系。]

3.范围和局限性

[项目范围定义了所提出的解决方案和概念和适用领域，而局限性则指出产品所不包括的某些性能。如果一般客户所提出的需求超出项目的范围时就应当拒绝它，除非这些需求是很有益的。记录这些需求以及拒绝它们的原因，以待查。]

3.1 首次发行的范围

[总结首次发行的产品所具有的性能。描述了产品的质量特性，这些特性使产品可以为不同的客户群提供预期的成果。应当避免将想到的每一个特性都包括到 1.0 版本产品中去。开发者应把重点放在能提供最大价值、花花费最合理的开发费用及普及率最高的产品上。]

3.2 随后发行的范围

[如果你想象一个周期性的产品演变过程，就要指明哪一个主要特性的开发将被延期，并期待随后版本发行的日期。]

3.3 局限性和专用性

[明确定义包括和不包括的特性和功能的界线是处理范围设定和客户期望的一个途径。列出风险承担者们期望的而你却不打算把它包括到产品中的特性和功能。]

4.业务环境

[这一部分总结了一些项目的业务问题。]

4.1 客户概貌

[客户概述明确了这一产品的不同类型客户的一些本质特点，以及目标市场部门和在

这些部门中的不同客户的特征。对于每一种客户类型，概述要包括：

- 各种客户类型将从产品中获得的主要益处；
- 它们对产品所持的态度；
- 感兴趣的关键产品的特性；
- 哪一类型客户能成功使用；
- 必须适应任何客户的限制。]

4.2 项目的优先级

[一旦明确建立项目的优先级，风险承担者和项目的参与者就能把精力集中在一系列共同的目标上。达到这一目的的一个途径是考虑软件项目的五个方面：性能、质量、计划、成本和人员。在所给的项目中，其每一方面应与下面三个因素之一相适应。

- 一个驱动----一个最高级别的目标；
- 一个约束----项目管理者必须操纵一个对象的限制因素；
- 一个自由度----项目管理能权衡其它方面，进而在约束限制的范围内完成目标的一个因素。

未必所有的因素都能成为驱动，或所有的因素都能成为约束因素。在项目开始时记录和分析哪一个因素适用于哪一类型，将有助于使每一个人的努力和期望与普遍认可的优先级相一致。]

5.产品成功的因素

[明确产品的成功是如何定义和测量的，并指明对产品的成功有巨大影响的几个因素。不仅要包括组织直接控制的范围内的事务，还要包括我部素。如果可能，可建立测量的标准，用于评价是否达到业务目标，如：市场股票、销售量及收入、客户满意度、交易处理量和准确度。]

# 项目构想

批注[XuFeng5]： 注，以 RUP《前景》为基础进行修改

编者说明：

这个文档模板与“软件项目视图与范围”文档的功能十分接近，只不过该文档更适合于产品型项目。其注重对项目的用户、市场进行分析，紧抓项目相关人员（也叫做风险承担者）的需求的本质。

1. 文档简介

[软件需求规格说明书的整个内容还是锁定于整个系统的操作、使用层面之上的功能性需求，只是解决了 How 的问题，而并未回答 Why 的问题。这使得系统在开发过程中，开发团队经常陷入知其然，而不知其所以然的困境，造成了不必要的误解与错误。因此，需要一个侧重于对项目的风险承担者、目标用户需要的文档，不仅要了解他们需要的功能，还要找到他们提出这些需求的原因。这就是“项目构想”文档所要描述的重要内容。]

[本节的内容主要是提供项目构想文档的目的、范围、定义、参考资料以及对其的摘要性概述。]

1.1 目的

[说明该文档的写作目的。]

1.2 范围

[范围主要用来说明该文档描述的项目内容，以及与其相关的其它东西。]

1.3 定义、首字母缩写词和缩略语

[与其它文档一样，该文档也需要将本文档中所涉及的所有术语、缩略语进行详细的定义。还有一种可简明的做法，就是维护在一个项目词汇表中，这样就可以避免在每个

文档中都重复很多内容。]

1.4 参考资料

[在这一小节中，应完整地列出该文档引用的所有文档。对于每个引用的文档都应该给出标题、标识号、日期以及来源，为阅读者查找这些文档提供足够详细的信息。]

1.5 概述

[在本小节中，主要是说明项目构想各个部分所包含的主要内容，就像一个文章摘要一样。同时也应该对文档的组织方式进行解释。]

2.定位

2.1 商业机会

[如果该项目是一个产品型项目，那么应该在本小节中描述该产品所针对的商业机会。如果是定制开发项目，那么可以省去本小节。]

2.2 问题说明

［使用表格的形式，将该项目将要解决的问题进行概要性地描述：］

| | |
|---|---|
| 存在的问题 | ［问题的简要说明］ |
| 受影响的人群 | ［该问题对哪些人群带来了影响］ |
| 导致的后果 | ［该问题带来的不利因素］ |
| 希望的解决方案 | ［列出解决方案所能够解决的问题，以及其相应的优点。］ |

2.3 产品定位说明

[如果是产品型项目，则该小节将以表格的形式对产品的定位进行明确，如果是定制开发项目，可以省略本小节。]

| | |
|---|---|
| 目标市场 | ［描述产品目标客户群体］ |
| 目标客户需求 | ［说明客户的需要或者潜在的机会］ |
| 产品类别 | ［说明该产品属于什么领域］ |
| 主要优点 | ［描述让目标客户产生兴趣和购买欲的理由］ |
| 主要竞争对手 | ［列出与该产品有竞争的其它厂商的产品］ |
| 主要优势 | ［针对竞争产品的分析］ |

[一个具有清晰定位的产品，在开发过程中，团队将更好地理解，更容易开发出满足目标市场的产品，因而该部分内容是十分重要的。]

3.项目相关人员和用户说明

［了解用户、了解所有与该项目相关的人员，是有效地满足他们对系统、产品需求的基础。你应该在本小节中将所有的项目相关人员以及用户收罗在一起，并对他们进行简要的描述，对他们的需求、习惯、角度进行说明。这些内容将有助于开发团队更好的理解用户的需求本质。］

3.1 产品用户分析

[如果是产品型项目，那么你应该本节中对目标客户进行分析。可以在市场调查的基础上，对其市场的规模和增长率进行研究，从而估计其潜在的用户数量。另外，还应结合目标市场的实际情况，分析你的组织是否在该市场上有拓展的优势，如何获得这些优势。如果是定制开发项目，可以省略这一小节。]

3.2 项目相关人员一览表

[使用下面的表格，对项目相关人员进行分析。]

| 人员类别 | 代表 | 作用 |
|---|---|---|
| [指明项目相关人员的类别] | [列举该类人员的代表] | [说明其对产品、项目开发的影响] |

3.3 用户一览表

[使用下面的表格，对项目、产品的用户进行分析。]

| 用户类型 | 说明 | 代表 |
| --- | --- | --- |
| [指明用户类别] | [简要说明他们在系统中代表的对象和充当的作用] | [列举出代表] |

3.4 用户环境

[了解用户在使用环境下使用系统或产品，是十分有意义的事，也是实现产品更好地满足需求，提供更加方便的使用界面的基础。例如：该任务由多少人来完成？是否总在变化？一个任务周期需要多长时间？执行每项活动要用多长时间？是否总在变化？是否有特殊的环境约束：移动、户外、乘机旅行等？目前使用的是哪些系统平台？以后会使用哪些平台？还在使用哪些应用程序？您的应用程序是否需要和这些应用程序集成？他们的计算机硬件系统的环境情况如何？他们都是在什么样的工作环境中使用系统的？]

3.5 项目相关人员的简要说明

[以下表的形式，将各类项目相关人员的基本情况进行说明，以帮助开发团队更好地了解他们的情况。为每一类人员生成一张表格。]

| | |
| --- | --- |
| **代表** | [列出该类项目相关人员的代表。] |
| **说明** | [对该类人员进行简要说明。] |
| **专业技能** | [描述本类人员的技能特长、技术背景以及电脑系统操作的熟练程度（可以分成业务用户、专家用户、熟练用户、初级用户等）] |
| **职责** | [描述本类人员对系统开发所承担的职责，以及应享有的利益。] |
| **验收标准** | [描述验证系统是否满足其职责的标准。] |
| **参与方式** | [该类人员是否参与系统开发，如果参与将以什么形式参加。] |
| **项目成果** | [说明该类项目相关人员是否参与项目成果的开发，是否有与其相关的项目成果。] |
| **意见/问题** | [列出与该类项目成员相关的问题与建议。] |

3.6 用户简要说明

[以下表的形式，将与系统相关的各种用户的信息整理出来，以方便开发团队针对性的工作。要注意的是，用户会有不同的类型，有些用户需要的是灵活性、方便快速操作的高级功能，而有些用户则侧重与用户界面的友好性。这些与该用户的基本情况直接相关，了解用户才能够真正地开发出符合用户习惯和水平的系统。为每类用户生成一张表。]

| | |
| --- | --- |
| **代表** | [列出该类用户的代表。] |
| **说明** | [对该类用户进行简要说明。] |
| **专业技能** | [描述该用户的技能特长、技术背景和对计算机系统操作的熟练程度。] |
| **职责** | [列出该用户对所开发的系统负有的关键职责，如记录详细信息、撰写报告、协调工作等。] |
| **验收标准** | [描述验证系统符合用户需求的标准。] |
| **参与方式** | [说明该类用户是否参与开发，如何参与。] |
| **项目成果** | [说明是否有依赖于该类用户的项目成果。] |
| **意见/问题** | [列出一些该类用户对系统提出的一个意见与建议，并 |

| | |
|---|---|
| | 且收集其认为该系统将遇到的问题。] |

3.7 关键的项目相关人员/用户需要

[列出项目相关人员提出的针对对于该解决方案的关键问题。对于列出的每个问题，需澄清：为什么会出现这一问题？目前的解决方案是什么？他们需要什么要的解决方案？或者对新的解决方案有什么样的预期？]

[还有一个很关键的内容就是，每个需求的优先级，这将对制定迭代计划时提供有效的基础，而优先级的确定，应该采用分级、累积投票等方法从用户、项目相关人员那里获得。应充分考虑项目客户方的要求。如果是产品型项目，则应该从产品经理、市场调查资料里获得。]

[经过整理后，将内容填入下表：]

| 需求 | 优先级 | 要点 | 目前解决方案 | 提议的解决方案 |
|---|---|---|---|---|
| | | | | |

3.8 备选方案和竞争

[如果是产品型项目，应在此小节列举出客户除了购买该产品这外的选择，其中包括购买竞争对手的产品、自行设计解决方案甚至是维持现状。对所有潜在的竞争产品做一个列表，并根据客户的实际情况来确认主要优缺点。]

[而如果是定制开发型项目，则应该了解竞争对手提供的解决方案，比在此进行相应的比较。]

4. 产品概述

［本节主要从产品级、系统级的视角，高度概括产品的功能、与其它应用程序的交互以及所需的系统配置等。］

4.1 产品总体效果

[本小节主要将产品话在用户环境、使用环境的角度来介绍。如果是自成一体，则说明用户将如何使用；如果是与其它的应用系统进行交互的，则在此小节说明如何与这些系统进行交互？它们之间采用什么样的通讯方式和接口。在这里最适合的方式是使用 UML 的部署图，让用户对系统最终的运行环境有一个较宏观的了解。]

4.2 主要功能

[本小节不是对系统或产品所有功能的罗列，而是将能够体现系统、产品主要优点和特性功能在此列出。在内容组织方面，应该直接与“客户能够通过产品获得的好处”相联系，使读者能够将系统的功能与客户的价值直接联系起来，在开发时能够从本质出发，构建出更加符合客户需要的系统。]

4.3 假设与依赖关系

[在此小节中，列出所有会影响该文档中所述特性的各种因素。也就是列举出所有可能让该文档发生变化的假设条件。]

4.4 成本与定价

[该小节主要是对该项目的成本进行核算，对给出相应的定价策略。对于定制开发的项目，其成本主要包括开发的人工成本、公司管理成本、项目额外开支、相关软硬件工具投资等方面。而对于产品型项目而言，还包括分销成本、用户手册制作、CD 制作等方面的成本。这里的成本核算为最终的合同价格以及产品的销售价值将提供一个基础的依据，因此也是十分重要的。]

4.5 许可与安装

[该小节中主要列出影响开发工作的一些许可和安装相关的问题。例如是否需要加密，如果验证用户合法性，安装界面的要求是什么。这方面对于产品型项目而言显得更加重要，也是对软件知识产权保护的一个重要措施。]

5.产品特性

[在本节中将列出系统或产品的特性，特性是指实现用户价值的系统功能。每一个特性都是一个所需的服务，通常是通过一系列操作实现预期结果。在 FDD 中，也就是特征。通常一个特征会由一个或多个用例来实现，通常系统的特性应该进行整合打包，以 25-99 项为合适。]

[本小节的描述应该能够让用户、操作人员、外部系统直接从系统的外边感受到每项特性，这些特性应该包括功能性说明以及一些可用性问题。但是要注意，在这里不要过早地引入设计的内容，这里说明的是 What，而不是 How。]

[另外，因在所有特性的描述中，确定其优先级。]

6.约束

[记录用户、项目相关人员提供出的一些约束条件，以及与其它系统之间的依赖关系，这是制订解决方案时必须考虑到的问题。]

7.质量要求

[对于整个系统的质量要求，如可靠性、可用性、性能、容错等质量要求，在这此节中详细地定义与描述。]

8.其他产品需求

[一些要求符合的标准、硬件基础要求、软件基础要求、环境要求等。]

8.1 适用的标准

[列出产品必须符合的所有标准。其中可能包括法律和法规（FDA、UCC）标准、通讯标准（TCP/IP、ISDN）、平台一致性标准（Windows、Unix 等）以及质量和安全标准（UL、ISO、CMM）。]

8.2 系统需求

[确定支持该应用程序所必需的任何系统需求。其中可能包括操作系统、网络环境、系统配置、内存大小、硬盘大小、外围设备和配套软件。]

8.3 性能需求

[本节用于详细说明性能需求。性能问题可能包括在各种负载条件下的用户负载因素、带宽或通信容量、吞吐量、精确度以及可靠性或响应时间。]

8.4 环境需求

[对于基于硬件的系统，环境因素可以包括温度、振荡、湿度、辐射等。对于软件应用系统，环境因素可以包括使用条件、用户环境、资源可用性、维护问题、错误处理和恢复。]

9. 文档需求

[列举用户所需的与该系统或产品相关的文档。]

9.1 用户手册

[用户手册的制作说明，例如手册篇幅、详细程序、是否需要图、主要关心的点、要不要建立索引、词汇表，采用教程式还是速查手册式。]

9.2 联机帮助

[联机帮助是一种用户界面友好的服务，它可以为用户提供实时的协助。]

9.3 安装指南、配置文件、自述文件

9.4 标签与包装

10. 功能需求属性

[为了在项目开发过程中，对每个功能需求进行跟踪管理，在此对所有的功能进行一个总体的描述。]

[可以生成一张功能需求属性表，每条记录代表一条功能，每个功能包括以下字段：]

1）状态：标识该功能的最新状态。

- 已提出：已经提出来，但是还没有经过正式的复审而确定的需求；
- 已批准：已经经过正式的渠道复审而确定，准备实施的需求；
- 已加入：已经加入到需求管理基线中的特性。

2）利益：根据客户的态度，确定每个需求的重要程序，也是确定系统开发优先级的基础数据。

- 关键：必不可少的特性，缺少这些特性的系统将无法满足客户的要求，这些特性通常会在最早安排到迭代开发中去；
- 重要：对于系统来说，该特性是十分重要的，很难以通过其它方式来弥补，如果这些特性没有第一时间实现，将会使得客户满意度大大降低。因此是第二优先实现的特性；
- 有用：这些是一些有效，但使用频率较低的功能特性。如果没有在第一时间实现，也不会对客户满意度造成很大的影响；
- 无用：对于系统来说是“镀金”需求，有也可以，没有也行的。

3）工作量：根据特性所需的时间和资源进行估算，给出团队开发的工作时间或个人开发的工作时间。也可以估算出代码行数或功能点数，这也将为迭代开发计划的制定提供良好的基础。

4）风险：列出该特性开发的最大风险，可以对这些风险进行级别细分，对于影响较大的风险还应该制定相应的应对措施。

5）稳定性：对该特性需求是否容易变化进行一个预估，以帮助设计人员在设计解决方案时更加有效地避免变化对体系结构的影响，从而节省时间。

6）基线：确定其是否已经纳入基线；

7）职责分配：列出负责实现该特性的团队；

8）原因：列出提出该特性的原因，也可以将与客户交流的记录等资料放在这里，以帮助开发团队更好的理解客户的本意。

## 需求规格说明书(ISO 标准版)

编者说明：

当需求调查、分析工作告一段落时，你就需要将这些需求进行规格化描述，整理成文，即软件需求规格说明书，也就是 SRS。这是在软件项目过程中最有价值的一个文档。ISO 所提供的标准虽然已经时间久远，但还是颇具参考价值的。

1．引言

1.1 编写的目的

[说明编写这份需求说明书的目的,指出预期的读者。]

1.2 背景

a. 待开发的系统的名称;
b. 本项目的任务提出者、开发者、用户；
c. 该系统同其他系统或其他机构的基本的相互来往关系。

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出用得着的参考资料。]

2．任务概述

2.1 目标

[叙述该系统开发的意图、应用目标、作用范围以及其他应向读者说明的有关该系统开发的背景材料。解释被开发系统与其他有关系统之间的关系。]

2.2 用户的特点

[列出本系统的最终用户的特点，充分说明操作人员、维护人员的教育水平和技术专长，以及本系统的预期使用频度。]

2.3 假定和约束

[列出进行本系统开发工作的假定和约束。]

3．需求规定

3.1 对功能的规定

[用列表的方式，逐项定量和定性地叙述对系统所提出的功能要求，说明输入什么量、经怎么样的处理、得到什么输出，说明系统的容量,包括系统应支持的终端数和应支持的并行操作的用户数等指标。]

3.2 对性能的规定

3.2.1 精度

[说明对该系统的输入、输出数据精度的要求，可能包括传输过程中的精度。]

3.2.2 时间特性要求

[说明对于该系统的时间特性要求。]

3.2.3 灵活性

[说明对该系统的灵活性的要求，即当需求发生某些变化时，该系统对这些变化的适应能力。]

3.3 输入输出要求

[解释各输入输出数据类型，并逐项说明其媒体、格式、数值范围、精度等。对系统的数据输出及必须标明的控制输出量进行解释并举例。]

3.4 数据管理能力要求（针对软件系统）

[说明需要管理的文卷和记录的个数、表和文卷的大小规模，要按可预见的增长对数据及其分量的存储要求作出估算。]

3.5 故障处理要求

[列出可能的软件、硬件故障以及对各项性能而言所产生的后果和对故障处理的要求。]

3.6 其他专门要求

[如用户单位对安全保密的要求，对使用方便的要求，对可维护性、可补充性、易读性、可靠性、运行环境可转换性的特殊要求等。]

4．运行环境规定

4.1 设备

[列出运行该软件所需要的硬设备。说明其中的新型设备及其专门功能，包括：

a. 处理器型号及内存容量

b. 外存容量、联机或脱机、媒体及其存储格式，设备的型号及数量

c. 输入及输出设备的型号和数量，联机或脱机；

d. 数据通信设备的型号和数量

e. 功能键及其他专用硬件]

4.2 支持软件

[列出支持软件，包括要用到的操作系统、编译程序、测试支持软件等。]

4.3 接口

[说明该系统同其他系统之间的接口、数据通信协议等。]

4.4 控制

[说明控制该系统的运行的方法和控制信号，并说明这些控制信号的来源。]

## 需求规格说明书(Volere 版)

编者说明：

Atlantic System Guild（www.atlsysguild.com）公司所提供的 Volere 需求过程与软件需求规格说明书模板则充分利用了现代软件工程思想与技术，是一个十分实用、完善的 SRS 模板。其所提供的 Volere 需求记录卡也十分实用，强烈推荐。

*注：从 Atlantic System Guild 公司网站 www.atlsysguild.com 上获得，并稍做修改*

1.产品的目标

1.1 该项目工作的用户问题或背景

[对引发开发任务的工作和情况的描述。同时也应描述用户希望用将要交付的软件来完成的工作。]

*[该节内容为该项目提供了合法的理由，你应该考虑用户的问题是否严重，是否应该解决和为什么应该解决。]*

1.2 产品的目标

[用一句话或很少的几句话来说明“我们希望该产品做什么？”换言之，即开发该产品的真正原因。

*[项目如果没有一个表述清晰、易于理解的目标，就会迷失在产品开发的沙漠中。产品必须带来某种优势。典型的优势是产品会增加组织在市场上的价值，减少运作成本，或提供更好的客户服务。这个优势应该是可度量的，这样才能够让您确定交付的产品是否达到目标。]*

2.客户、顾客和其它风险承担者

2.1 客户是为开发付费的人，并将成为所交付产品的拥有者

[这一项必须给出客户的姓名，三个以内是合理的。]

*[客户最终将接受该产品，因此必须对交付的产品满意。如果你无法找到一个客户的姓名，那么也许你就不应该构建该产品。]*

2.2 顾客是将花钱购买该产品的人

[也给出姓名和相关的信息]

2.3 其它风险承担者

[其他的一些人或组织的名称，他们或者受到产品的影响，或影响产品。]

1） 经理或项目负责人；
2） 业务领域专家；
3） 技术人员；
4） 系统开发者；
5） 市场人员；
6） 产品经理；
7） 测试和质量保证人员；
8） 审查员，诸如安全审查员或审计人员；
9） 律师；
10）易用性专家；
11）你所处行业的专业人员。

3.产品的用户

3.1 产品的用户

[产品的潜在用户或操作员的列表。针对每种类型的用户，提供以下信息：]

1） 用户分类

2） 用户工作的任务；

3） 主要相关的经验；

4） 技术经验；

5） 其他用户特征：包括身体、智力、工作态度、对技术的态度、教育程度、语言技能、年龄、性别等。

*[用户是为了完成工作而与产品交互的人，你了解用户，就越可能提交适合用户工作方式的产品。]*

3.2 对用户设的优先级

[在每类用户后面附上一个优先级，这区别了用户的重要性和优先地位：]

1） 关键用户：对产品的后续成功至关重要；

2） 次要用户：他们使用产品，但对产品的长期成功并无影响；

3） 不重要的用户：不常用、未授权和没有技能的用户。

*[如果认为某些用户对产品或组织更重要，那么应该写明，因为它会影响你设计产品的方式。]*

4.需求限制条件

4.1 解决方案限制条件

[此处明确了限制条件，它们规定了解决问题必须采取的方式。您可以认为它们是指令式的解决方案。仔细描述该解决方案，以及测试是否符合的度量标准。如果可能，您应该解释使用该解决方案的原因。]

*[换一句话说，就是要求软件解决方案满足哪些限制条件！]*

4.2 实现环境

[此处描述产品将被实施的技术环境和物理环境。]

*[该环境也将成为设计解决方案时的限制条件之一。]*

4.3 伙伴应用

[此处描述那些不属于产品的一部分，但产品却又必须与其协作的应用程序。]

4.4 COTS

[此处描述实现产品需求所必须使用的 COTS(商业组件)。]

4.5 预期的工作场地环境

[此处描述用户工作和使用该产品的工作场地。此处应该描述任何可能对产品设计产生影响的工作场地特征。]

4.6 开发者构建该产品需要多少时间

[任何已知的最后期限，或商业机会的时限，应在此处说明。]

4.7 该产品的财务预算是多少

[该产品的预算，以金钱的形式或可得资源的形式说明。]

5.命名标准和定义

[定义项目中使用到的所有术语，包括同义词。这里的内容就是一个字典，包括在需求规格说明书中使用的所有名称的含义。这个字典应该使用你的组织或行业使用的标准名称。这些名称也应该反映出在工作领域中当前使用的术语。该字典包括项目中用到的所有名称。请仔细地选择名称，以避免传达不同的、不期望的含义。为每个名字写下简明扼要的定义，这些定义必须经过相应的风险承担者同意。]

6.相关事实

[可能对产品产生影响的外部因素，但不是命令式的需求限制条件。]

7.假定

[列出开发者所做的假设。]

*[将所有的假设列在此的目的是让每一个项目成员都意识到这个假设。]*

8.产品的范围

8.1 工作的上下文范围

[上下文范围图用来表示将要开发的系统、产品与其它系统之间的关系，以确定系统边界。]

8.2 工作切分

[一个事件清单，确定系统要响应的所有业务事件。清单包括：]

1） 事件名称

2） 输入和输出

8.3 产品边界

[你可以使用用例图(use-case)来确定了用户与产品之间的边界。]

9.功能性需求与数据需求

9.1 功能性需求

[对产品必须执行的动作的描述。]

*[每个功能性需求必须有一个验收标准。]*

9.2 数据需求

[与产品/系统有密切关系的主题域相关的业务对象、实体、类的说明书。]

*[进行问题域建模，生成相应的类图。]*

10.观感需求

[一些与产品的用户界面相关的需求描述。]

11.易用性需求

11.1 易于使用

[描述如何构建符合最终用户期望的产品。]

11.2 学习的容易程序

[学习使用该产品应该多容易的说明。通常是有学习时间来衡量。]

12.性能要求

12.1 速度需求

[明确完成特定任务需要的时间，这常常指响应时间。]

12.2 安全性的需求

[对可能造成人身伤害、财产损失和环境破坏所考虑到的风险进行量化描述。]

12.3 精度需求

[对产品产生的结果期望的精度进行量化描述。]

12.4 可靠性和可用性需求

[本节量化产品所需的可靠性。这常常表述为允许的两次失败之间无故障运行时间，或允许的总失败率。]

12.5 容量需求

[本节明确处理的吞吐量和产品存储数据的容量。]

13.操作需求

13.1 预期的物理环境

[本节明确产品将操作的物理环境，以及这种环境引起的任何特殊需求。]

13.2 预期的技术环境

[硬件和其它组成新产品操作环境的设备的规范。]

13.3 伙伴应用程序

[对产品必须与之交互的其它应用程序的描述。]

14.可维护性和可移植性需求

14.1 维护该产品需要多容易

[对产品作特定修改所需时间的量化描述。]

14.2 是否存在一些特殊情况适用于该产品的维护

[关于预期的产品发布周期和发布将采取的形式的规定。]

14.3 可移植性需求

[对产品必须支持的其他平台或环境的描述。]

15.安全性需求

15.1 该产品是保密的吗?

[关于该被授权使用该产品，以及在什么样的情况下授权等方面的描述。]

15.2 文件完整性需求

[关于需要的数据库和其他文件完整性方面的说明。]

15.3 审计需求

[关于需要的审计检查方面的说明。]

16.文件和政策需求

[本节包括针对社会和政策的因素的规格说明，这些因素会影响产品的可接受性。如果你开发的产品是针对外国市场的，可能要特别注意这些需求。]

*[问一下是否产品的目标是你所不熟悉的文化环境，是否其它国家的人或其他类型的组织的人会使用该产品。人们是否有与你的文化不同的习惯、节日、迷信、文化上的社会行为规范。]*

17.法律需求

17.1 该产品是否受到某些法律的管制

[明确该产品的法律需求的描述。]

17.2 是否有一些必须符合的标准

[明确适用的标准和参考的详细标准的描述。]

18.Opend 问题

[对未确定但可能对产品产生重要影响的因素的问题描述。按照需求分析的术语还说，就是 TBD（To Be Define）的问题。]

19.COTS 解决方案

19.1 是否有一些制造好的产品可以购买

[应该调查现存产品清单，这些产品可以作为潜在的解决方案。]

19.2 该产品是否可使用制造好的组件

[描述可能用于该产品的候选组件，包括采购的和公司自己的产品。列出来源。]

19.3 是否有一些我们可以复制的东西

[其他相似产品的清单。]

20.新问题

20.1 新产品会在当前环境中带来什么问题

[关于新产品将怎样影响当前的实现环境的描述。]

20.2 新的开发是否将影响某些已实施的系统

[关于新产品将怎样与现存系统协同工作的描述。]

20.3 是否我们现有的用户会受到新开发的敌对性影响

[关于现有用户可能产生的敌对性反应的细节。]

20.4 预期的实现环境会存在什么限制新产品的因素

[关于新的自动化技术、新的组织结构方式的任何潜在问题的描述。]

20.5 是否新产品会带来其他问题

[确定我们可能不能处理的情况。]

21.任务

21.1 为提交该产品已经做了哪些事

[用来开发产品的生命周期和方法的细节。画一个高层的过程图展示各项任务和它们之间的接口，这可能是沟通这方面信息的最好办法。]

21.2 开发阶段

[关于每个开发阶段和操作环境中的组件的规格说明。]

22.移交

22.1 我们要让已有数据和过程配合新产品，有什么特殊要求

[一个移交活动的列表，一个实现的时间表。]

22.2 为了新产品，哪些数据必须修改/转换

[数据转换任务清单，同时确定新产品需要转换的数据。]

23. 风险

23.1 当你开发该产品时，要面对什么风险

23.2 你制定了怎样的偶然紧急情况计划

24.费用

[需求的其他费用是你必须投入到产品构建中去的钱或工作量。当需求规格说明书完成时，你可以使用一种估算方法来评估费用，然后以构建所需的资金或时间的形式表述出来。]

25.用户文档

[用户文档的清单，这些文档将作为产品的一部分交付。]

26.后续版本的需求

[这里记录下一些希望今后版本中实现的需求。]

## Volere 需求记录卡

编者说明：

正如前面所述，Atlantic System Guild 还提供了一个配套的 Volere 需求记录卡，这个记录卡十分实用。建议大家在需求调查、分析过程中，将需求记录在一系列的 Volere 需求记录卡上，这个卡让你能够很好的理清需求之间的关系，需求提出的背景，用户对需求的期望，有了这些素材，整理 SRS 时将变得更加简单。

| 需求#： | 需求类型： | 事件/用例#： |
| --- | --- | --- |
| 描述：<br><br>理由：<br><br>来源：<br>验收标准： | | |

*注：顾客满意度是指完成该项功能顾客满意的程度，而顾客不满意度则是指未实现该功能顾客不满意的程度。*

# 软件需求规格说明书

批注[XuFeng6]： 注，以 RUP《软件需求规约》进行修改。

编者说明：

如果在需求分析时采用了用例（Use case）技术，那么该需求规格说明书将更加符合你的需要。当然，你也可以结合 Volere 需求规格说明书对该模板进行必要的修改。

1. 文档概述

[该部分主要是对软件需求规格说明书文档进行基本的描述，包括该文档的目的、范围、术语定义、参考资料以及概要。]

[软件需求规格说明书用来系统、完整地记录系统的软件需求。该软件需求说明书的基础是用例分析技术。因此该文档中应包括用例模型、补充规约等内容。]

1.1 目的

[在此小节中，主要对软件需求规格说明书的目的做一概要性说明，通常软件需求规格说明书应详细地说明应用程序、子系统的外部行为，还要说明非功能性需求、设计约束，以及其它的相关因素。]

1.2 范围

[系统是有范围的，而不是无限扩展的，对于无限扩展的需求是无法进行描述的。因此，在本小节应该对该说明书所涉及的项目范围进行清晰的界定。指定该规格说明书适用的软件应用程序、特性或者其它子系统分组、其相关的用例模型。当然在此也需要列出会受到该文档影响的其它文档。]

1.3 定义、首字母缩写词和缩略语

[与其它文档一样，该文档也需要将本文档中所涉及的所有术语、缩略语进行详细的定义。还有一种可简明的做法，就是维护在一个项目词汇表中，这样就可以避免在每个文档中都重复很多内容。]

1.4 参考资料

[在这一小节中，应完整地列出该文档引用的所有文档。对于每个引用的文档都应该给出标题、标识号、日期以及来源，为阅读者查找这些文档提供足够详细的信息。]

1.5 概述

[在本小节中，主要是说明软件需求规格说明书各个部分所包含的主要内容，就像一个文章摘要一样。同时也应该对文档的组织方式进行解释。]

2. 整体说明

[在本节中，将对整个软件需求进行总体性的描述，以期让读者对整个软件系统的需求有一个框架性的认识。也就是说，该节中主要包括影响产品及其需求的一般因素，而不列举具体的需求。主要包括产品总体效果、产品功能、用户特征、约束、假设与依赖关系、需求子集等方面的内容。]

2.1 用例模型

[在本小节中，将列出该软件需求的用例模型，该模型处于系统级，对系统的特性进行宏观的描述。在此应该列出所有的用例和 Actor 的名称列表，并且对其做出简要的说明，以及在图中的各种关系。]

2.2 假设与依赖关系

[在软件系统的开发过程中，存在许多假设和依赖关系。在本小节中应列举出所有的重要的技术可行性假设、子系统或构件可用性假设，以及一些可行性的假设。]

3. 具体需求

[如果说第二章节是框架，那么本节就是血肉。在本节中，应该详细列出所有的软件需求，其详细程序应使设计人员能够充分理解并且进行设计的要求，同时也应该给予测试人员足够的信息，以帮助他们来验证系统是否满足了这些需求。整个需求的组织可以采用用例描述进行。]

3.1 用例描述

[如果你使用用例建模技术，那么你已经通过用例定义了系统的大部分功能性需求和一些非功能性需求。因此，在软件需求规格说明书只需将这些具体的用例描述，整理在一起，全部放在该小节之中。当然也可以将用例描述做为附件，在此列出引用，只是这样做并不利于阅读。建议在组织形式上采用以“软件需求”为线索，在每个需求中，填入对应的 1 个或几个用例描述。]

3.2 补充需求

[由于用例毕竟主要针对功能性需求，因此还会有一些其它的补充需求遗漏，因此在本小节中就是将这些东西补充出来。这些补充需求大部分集中在非功能需求之上，包括以下几个方面的内容：]

1） 易用性：例如指出普通用户和高级用户要高效地执行某个特定操作所需的培训时间；指出典型任务的可评测任务次数；或者指出需要满足的可用性标准（如 IBM 的 CUA 标准、Microsoft 的 GUI 标准。

2） 可靠性：包括系统可用性（可用时间百分比、使用小时数、维护访问权、降纸模式操作等）；平均故障间隔时间（MTBF，通常表示为小时数，但也可表示为天数、月数或年数）；平均修复时间（MTTR，系统在发生故障后可以暂停运行的时间）；精确度（指出系统输出要求具备的精密度、分辨率和精确度）；最高错误或缺陷率（通常表示为 bugs/KLOC，即每千行代码的错误数目或 bugs/function-point，即每个功能点的错误数目）；错误或缺陷率（按照小错误、大错误和严重错误来分类：需求中必须对“严重”错误进行界定，例如：数据完全丢失或完全不能使用系统的某部分功能）。

3） 性能：包括对事务的响应时间（平均、最长）；吞吐量（例如每秒处理的事务数）；容量（例如系统可以容纳的客户或事务数）；降级模式（当系统以某种形式降级时可接受的运行模式）；资源利用情况：内存、磁盘、通信等。

4） 其它：包括用户界面要求、联机帮助系统要求、法律许可、外购构件，以及操

作系统、开发工具、数据库系统等设计约束。

4.支持信息

[支持信息用于使软件需求规格说明书更易于使用。它包括：目录、索引、附录等。]

## 计算机软件需求说明编制指南

编者说明：

软件需求规格说明是十分重要的文档，因此为开发团队提供一份详细的编制指南是十分有意义和必要的。本文档就是一个编制指南的例子，你可以根据该指南，结合自己的实际情况进行修改。

### 1．引言

#### 1.1 目的和作用

本指南为软件需求实践提供了一个规范化的方法。本指南不提倡把软件需求说明（Software Requirements Specifications，以下简称 SRS）划分成等级，避免把它定义成更小的需求子集。

本指南适用对象：

1）软件客户（Customers），以便精确地描述他们想获得什么样的产品。

2）软件开发者（Suppliers），以便准确地理解客户需要什么样的产品。

对于任一要实现下列目标的单位和（或）个人：

1）要提出开发规范化的 SRS 提纲；

2）定义自己需要的具体的格式和内容；

3）产生附加的局部使用条款，如 SRS 质量检查清单或者 SRS 作者手册等。

SRS 将完成下列目标：

1） 在软件产品完成目标方面为客户和开发者之间建立共同协议创立一个基础。对要实现的软件功能做全面描述，帮助客户判断所规定的软件是否符合他们的要求，或者怎样修改这种软件才能适合他们的要求；

2） 提高开发效率。编制 SRS 的过程将使客户在设计开始之前周密地思考全部需求，从而减少事后重新设计、重新编码和重新测试的返工活动。在 SRS 中对各种需求仔细地进行复查，还可以在开发早期发现若干遗漏、错误的理解和不一致性，以便及时加以纠正；

3） 为成本计价和编制计划进度提供基础。SRS 提供的对被开发软件产品的描述，是计算机软件产品成本核算的基础，并且可以为各方的要价和付费提供依据。SRS 对软件的清晰描述，有助于估计所必须的资源，并用作编制进度的依据；

4） 为确认和验证提供一个基准。任何组织将更有效地编制他们的确认和验证计划。作为开发合同的一部分，SRS 还可以提供一个可以度量和遵循的基准（然而，反之则不成立，即任一有关软件的合同都不能作为 SRS。因为这种文件几乎不包括详尽的需求说明，并且通常不完全的）；

5） 便于移植。有了 SRS 就便于移值软件产品，以适应新的用户或新的机种。客户也易于移植其软件到其他部门，而开发者同样也易于把软件移植到新的客户；

6） 作为不断提高的基础。由于 SRS 所讨论的是软件产品，而不是开发这个产品的设计。因此 SRS 是软件产品继续提高的基础。虽然 SRS 也可能要改变，但是原来的 SRS 还是软件产品改进的可靠基础。

#### 1.2 范围

本指南适用于编写软件需求规格说明，它描述了一个SRS所必须的内容和质量，并且在第6章中提供了SRS大纲。

## 2．引用标准

GB 8566 计算机软件开发规范

GB 8567 计算机软件产品开发文件编制指南

GB/T 11457 软件工程术语

## 3．定义

GB/T 11457所列术语和下列定义适用于本指南。

合同（contract）：是由客户和开发者共同签署的具有法律约束力的文件。其中包括产品的技术、组织、成本和进度计划要求等内容。

客户（customer）：指个人或单位，他们为产品开发提供资金，通常（但有时也不必）还提出各种需求。文件中的客户和开发者也可能是同一个组织的成员。

语言（language）：是具有语法和语义的通信工具，包括一组表达式、惯例和传递信息的有关规则。

分割（partitioning）：把一个整体分成若干部分。

开发者（supplier）：指为客户生产某种软件产品的个人或集团。在本指南中，客户和开发者可能是同一个组织的成员。

用户（user）：指运行系统或者直接与系统发生交互作用的个人或集团。用户和客户通常不是同一些人。

## 4．编写SRS的背景信息

### 4.1 SRS的基本要求

SRS是对要完成一定功能、性能的软件产品、程序或一组程序的说明。对SRS的描述有两项基本要求：

1）必须描述一定的功能、性能；

2）必须用确定的方法叙述这些功能、性能。

### 4.2 SRS的环境

必须认识到SRS在整个软件开发规范（见GB 8566）所规定的有关阶段都起作用。正因为如此，SRS的起草者必须特别注意不要超出这种作用的范围。这意味着要满足下列要求：

1） SRS必须正确地定义所有的软件需求；

2） 除设计上的特殊限制之外，SRS中一般不描述任何设计、验证或项目管理细节。

### 4.3 SRS的特点

#### 4.3.1 无歧义性

当且仅当它对每一个需求只有一种解释时，SRS者是无歧义的。

2） 要求最终产品的每一个特性用某一术语描述；

3） 若某一术语在某一特殊的行文中使用时具有多种歧义，那么对该术语的每种含义作出解释并指出其适用场合。

需求通常是用自然语言编写的，使用自然语言的SRS起草者必须特别注意消除其需求的歧义性。提倡使用形式化需求说明语言。

#### 4.3.2 完整性

如果一个SRS能满足下列要求，则该SRS就是完整的：

1） 包括全部有意义的要求，无论是关系到功能的、性能的、设计约束的，还是关系到属性或外部接口方面的需求；

2） 对所有可能出现的输入数据的响应予以定义，要对合法和非合法的输入值

的响应做出规定；

3）要符合 SRS 要求。如果个别章节不适用，则在 SRS 中要保留章节号；

4）填写 SRS 中的全部插图、表、图示标记和参照，并且定义全部术语和度量单位。

#### 4.3.2.1 关于使用“待定”一词的规定

任何一个使用“待定”的 SRS 都是不完全的。

1）若万一遇到使用“待定”一词时，作如下处理：

- 对产生“待定”一词的条件进行描述，使得问题能被解决；
- 描述必须干什么事，以删除这个“待定”；

2）包含有“待定”一词的任何 SRS 的项目文件应该：

- 标识与此特定文件有关的版本号或叙述其专门的发布号；
- 拒绝任何仍标识为“待定”一词的 SRS 章节的许诺。

### 4.3.3 可验证性

当且仅当SRS中描述的每一个需求都是可以验证的，该SRS才是可以验证的；当且仅当在某一性能价格比可取的有限处理过程，人或机器能通过该过程检查软件产品能否满足需求时，才称这个需求是可以验证的。

### 4.3.4 一致性

当且仅当 SRS 中各个需求的描述是不矛盾时 SRS 才是一致的。

### 4.3.5 可修改性

如果一个 SRS 的结构和风格在需求有必要改变时是易于实现的、完整性的、一致的，那么这个 SRS 就是可以修改的。可修改性要求 SRS 具备以下条件：

1）具有一个有条不紊的易于使用的内容组织，具有目录表，索引和明确的交叉引用表；

2）没有冗余。即同一需求不能在 SRS 中出现多次。

- 冗余本身不是错误，但是容易发生错误。冗余可增加 SRS 的可读性，但是在一个冗余文件被更新时容易出现问题。例如：假设一个明确的需求在两个地方详细列出，后来发现这个需求需要改变，若只修改一个地方，于是 SRS 就变得不一致了。
- 不管冗余是否必须，SRS 一定要包含一个详细的交叉引用表，以便 SRS 具备可修改性。

### 4.3.6 可追踪性

如果每一个需求的源流是清晰的，在进一步产生和改变文件编制时，可以方便地引证每一个需求，则该 SRS 就是可追踪的。建议采用如下两种类型的追踪：

1）向后追踪（即向已开发过的前一阶段追踪）。根据先前文件或本文件前面的每一个需求进行追踪。

2）向前追踪（即是向由 SRS 派生的所有文件追踪）。根据 SRS 中具有唯一的名字和参照号的每一个需求进行追踪。

当 SRS 中的一个需求表达另一个需求的一种指派或者是派生的，向前、向后的追踪都要提供。例如：

1）从总的用户响应时间需求中分配给数据库操作响应时间；

2）识别带有一定功能和用户接口的需求的报告格式；

3）支持法律或行政上需要的某个软件产品（例如，计算税收）。在这种情况下，要指出软件所支持的确切的法律或行政文件。

当软件产品进入运行和维护阶段时，SRS 的向前可追踪性显得特别重要。当

编码和设计文件作修改时，重要的是要查清这些修改所影响的全部需求。

#### 4.3.7 运行和维护阶段的可使用性

SRS 必须满足运行和维护阶段的需要，包括软件最终替换。

1）维护常常是由与原来开发无联系的人来进行的。局部的改变（修正）可以借助于好的代码注释来实现。对于较大范围的改变。设计和需求文件是必不可少的，这里隐含了两个作用：
   - 如 4.3.5 条指出，SRS 必须是可修改的；
   - SRS 中必须包括一个记录，它记录那些应用于各个成分的所有具体条文。例如：它们的危急性（如故障可能危及完全或导致大量财政方面和社会方面的损失）；它们仅与暂时的需要相关（如支持一种可立即恢复原状的显示）；它们的来源（如某功能是由已存在的软件产品的全部拷贝复制而成）。

2）要求在 SRS 中清楚地写明功能的来源和目的，因为对功能的来源和引入该功能的目的不清楚的话，通常不可能很好地完成软件的维护。

### 4.4 SRS 的编制者

软件开发的过程是由开发者和客户双方同意开发什么样的软件协议开始的。这种协议要使用 SRS 的形式，应该由双方联合起草。这是因为：

1）客户通常对软件设计和开发过程了解较少，而不能写出可用的 SRS；

2）开发者通常对于客户的问题和意图了解较少，从而不可能写出一个令人满意的系统需求。

### 4.5 SRS 的改进

软件产品的开发过程中，在项目的开始阶段不可能详细说明某些细节，在开发过程中可能发现 SRS 的缺陷、缺点和错误之类的问题，所以可能要对 SRS 进行改进。

在 SRS 的改进中，应注意如下事项：

1）尽管可以预见校正版本的开发以后不可避免，而对需求还必须尽可能完全、清楚地描述。

2）一旦最初识别出项目的变化，应引入一个正式的改变规程来标识、控制、追踪和报告项目的改变。批准了的需求改变，用如下的方法编入 SRS 之中：
   - 提供各种改变后的正确的、完全的审查记录；
   - 允许对 SRS 当前的和被替代部分的审查。

### 4.6 SRS 的编制工具

编制 SRS 最显而易见的方法是用自然语言来描述。尽管自然语言是丰富多彩的，但不易精确，用形式化的方法较好。

#### 4.6.1 形式化说明方法

在 SRS 中是否使用形式化方法要依据下列因素：

1）程序规模和复杂性；

2）客户合同中是否要求使用；

3）SRS 是否是一个合同工具或仅仅是一个内部文件；

4）SRS 文件是否成为设计文件的根据；

5）具有支持这种方法的计算机设备。

#### 4.6.2 生产工具

软件产品生产中有多种生产工具。比如，计算机的字处理器就是非常有用的生产辅助工具。一个 SRS 通常有若干作者。可能经历若干版本，并且要进行多次重新组织内容。故生产工具是必要的。

#### 4.6.3 表达工具

在 SRS 中有许多词汇，特别是许多名词和动词，专门涉及到系统的实体和许多活动，所以表达 SRS 需要若干工具。比如：

1）可以验证实体或活动，无论在 SRS 中什么地方都是同一名字。

2）可以标识一个特殊的实体或动作在规格说明中的描述位置。

此外，可以使用若干种形式化方法，以便允许自动处理 SRS 内容，只要作某些限制就可以做到：

用一些表格或图示法来显示需求。

用详细分层体系自动检查 SRS 的需求，这里每一个分层自身是完全的，但是也可以扩展为下一层，或是上一层的一个组成成分。

自动检查 SRS 具有在 4.3 条描述的部分或全部特点。

## 5 软件需求

SRS 中每一个软件需求是要求开发软件产品的某些基本功能和性能的一个陈述。

### 5.1 表达软件需求的方法

软件需求可以用若干种方法来表达：

1）通过输入、输出说明；

2）使用代表性的例子；

3）用规范化的模型。

#### 5.1.1 输入、输出说明

用输入输出序列来描述一个软件产品所要求的特性是很有效的。

##### 5.1.1.1 途径

根据被描述的软件的性质，至少有三种不同的途径：

1）有些软件产品（如报表系统）要求着重说明输出。一般情况下，致力于输出的系统主要是在数据文卷上操作。用户的输入通常是致力于提供控制信息和启动数据文卷的处理；

2）有些软件产品需要着重说明输入、输出特性。关注输入、输出的系统主要是在当前的输入上操作，要求生成与输入相匹配的输出（类似于数据转换例行程序或一个数学函数包）；

3）还有一些系统（如过程控制系统）要求记忆它们的状态。可以根据本次输入和上一次输入进行应答。也就是说，它的行为如同一个有限状态机。在此种情况下，既要关注输入/输出对，又要关注这些输入/输出对的次序。

##### 5.1.1.2 困难

多数软件产品可能接收无限的序列作为输入，于是，为了通过输入输出序列完整地说明产品的特性，就要求 SRS 包括一个无限长的输入和所需的输出充列。然而，用这样的途径不可能完整地描述软件所要求的一切特性。

#### 5.1.2 典型例子

一种选择是用典型例子来说明要求的特性。例如，假设一个系统中当接收“0”时用“1”来回答。显然，要列出全部输入和输出序列是不可能的。然而，用典型的序列可以十分清楚地理解系统的特性。下面是一组四种对话的典型的例子，用它描述系统特性。

0101

010101010101

01

010101

这些对话仅提供了要求的输入和输出之间的关系，但是不能完全描述系统的特性。

### 5.1.3 模型

另一种表达需求的方法是模型的方式，这是表达复杂需求的精确和有效方法。至少可以提出三种可供使用的通用模型：数学型、功能型、计时型。应注意区别各种模型的应用场合，参考 5.1.3.5。

#### 5.1.3.1 数学模型

数学模型是使用数学关系描述软件特性的模型。数学模型对某些特殊应用领域是特别有用的。例如，导航、线性规划、计量经济、信号处理和气象分析等。

用数学模型能够对 5.1.2 中所讨论的典型例子描述如下：

（01）*。

这里，“*”号表示括号内的字符串可以重复一次或多次。

#### 5.1.3.2 功能模型

功能模型是提供从略语以输出映象的模型。象有限状态机或 Petri 网，这些功能模型可以有助于标识和定义软件的各种特点，或者可以表示系统所要进行的操作。

对前面用数学模型描述的例子。可用图 1 所示的有限状态机形式的功能模型来描述。图中进入的箭头表示启动状态。双线的方框表示接收状态。在各线记号 x /y 的含义是：x 代表接受的输入，而 y 是产生的输出。

#### 5.1.3.3 计时模型

计时模型是一种增加了时间限制的模型。这种模型对于表达软件特性的形式和细节特别有用。尤其是实时系统或考虑人为因素的系统。

计时模型可以把下列限制加到图 1 的模型中去：

1）激活因素 0 将在进入 S1 状态 30S 之内出现；

2）响应 1 将在进入 S2 状态 2S 之内出现。

#### 5.1.3.4 其他模型

除了上面提及的模型外。对一些特殊的应用还有一些特别有用的模型。例如，编译程序的说明可以使用属性文法，工资单系统可以使用表格。要注意的是，对 SRS 使用形式需求语言，通常含有使用特殊模型的意思。

#### 5.1.3.5 警告

无论使用哪一类型的模型，都要在 SRS 中或在 SRS 涉及到的一个文件中对它严格定义。这个定义应该规定：

1）模型中的参数所要求的范围；

2）使用时的限定值；

3）结果的精确度；

4）负载的能力；

5）要求的执行时间；

6）缺省或失败时的响应。

必须注意，在需求的定义域内要保持一个模型定义。每当一个 SRS 使用一个模型时：

1）它意味着此模型提供一个十分有效和精确的方法说明需求；

2）并不意味着软件产品的实现必须基于这个模型。

一个模型用于解释文件所写的需求是有效的，但是对于实际软件的实现可能并不是最适宜的。

### 5.2 软件需求的注释

有关软件产品的所有需求，并不是同等重要的。某些需求可能是基本的，例如是对于生命攸关的应用。而另一些可能并不那么重要。

SRS 中每一个需求必须进行注释，以便区别其重要的程度。

有这种方法注释需求，可以：

1） 帮助客户对每个需求给予更周密的考虑，通常可以在需求中澄清隐藏的假设；

2） 帮助开发者做出正确的设计决定，并对软件产品不同部分作出相应的努力。

#### 5.2.1 稳定性

注释需求的一种方法是使用稳定性量纲。当一个需求在软件预期的生存期间内描述不改变的话，可以认为该需求是稳定的，否则可以认为是易变的。

#### 5.2.2 必要性等级

注释的另一种方法是把需求分成必须保证级、期望级和任选级。

5） 必须保证是指软件必须和这些需求相一致，否则该软件不可能被接受；

6） 期望是指这些需求将提高软件产品的功能，但如果缺省的话也是可接受；

7） 任选是给开发者一个机会，可以提供某些超出 SRS 规定的目标。

#### 5.2.3 注意事项

在注释需求之前，必须彻底理解这种注释的实质性含义。

### 5.3 在表达需求时遇到的共同弊病

SRS 的基本点是它必须说明由软件获得的结果，而不是获得这些结果的手段。编写需求的人必须描述的基本问题是：

1） 功能——所设计的软件要做什么；

2） 性能——是指软件功能在执行过程中的速度、可使用性、响应时间、各种软件功能的恢复时间、吞吐能力、精度、频率等等；

3） 强加于实现的设计限制——在效果、实现的语言、数据库完整性、资源限制、操作环境等等方面所要求的标准；

4） 属性——可移植性、正确性、可维护性及安全性等方面的考虑因素；

5） 外部接口——与人、硬件、其他软件和其他硬件的相互关系。

编写需求的人应当避免把设计或项目需求写入SRS之中，应当对说明需求设计约束与规划设计两者有清晰的区别。

#### 5.3.1 在 SRS 中嵌入了设计

在 SRS 中嵌入设计说明，会过多地约束软件设计，并且人为地把具有潜在危险的需求放入 SRS 中。

1） SRS 必须描述在干什么数据上、为谁完成什么功能、在什么地方、产生什么结果。SRS 应把注意力集中在要完成的服务目标上。通常不指定如下的设计项目：

- 把软件划分成若干模块；
- 给每一个模块分配功能；
- 描述模块间的信息流程或者控制流程；
- 选择数据结构。

2） 把设计完全同 SRS 隔离开来始终是不现实的。安全和保密方面的周密考虑可能增加一些直接反映设计约束的需求。例如：

- 在一些分散的模块中保持某些功能；
- 允许在程序的某些区域之间进行有限的通讯；
- 计算临界值的检查和。

3） 通常应考虑到，若要为软件选择高层次的设计，就可能需要大量的资源（可能占整个产品开发成本的 10%-20%以上）。有两种选择：

- 不顾本指南的警告，在 SRS 中描述了设计。这意味着，或者将一个潜在不适当的设计作为一个需求进行描述（因为，若要得到好的设计，所花费的时间是不够的），或者在需求阶段花费了过多的时间（因为在 SRS 完成之前整个设计分析都要完成）；
- 采用本指南中 5.1.3 条中的建议，用模型设计描述需求，这种模型设计只用于辅助描述需求，而不使之成为实际的设计。

### 5.3.2 在 SRS 中嵌入了一些项目要求

SRS 应当是描写一个软件产品，而不是描述生产软件产品的过程。

项目要求表达客户和开发者之间对于软件生产方面合同性事宜的理解（因此不应当包括在 SRS 中）例如：

1） 成本；
2） 交货进度；
3） 报表处理；
4） 软件开发方法；
5） 质量保证；
6） 确认和验证的标准；
7） 验收过程。

项目需求在另外文件中描述。在 SRS 中提供的只是关于软件产品本身的需求。

## 6 SRS 大纲

本章着重讨论 SRS 的每一个基本部分，可以作为一个 SRS 的大纲。表 1 给出该大纲目录，表 2 至表 5 给出大纲中第 3 章的具体需求内容。各开发者和客户应当根据所描述的实际情况，按本指南有关规定编写自己的 SRS。

```
1 前言
    1.1 目的
    1.2 范围
    1.3 定义、缩写词、略语
    1.4 参考资料
2 项目概述
    2.1 产品描述
    2.2 产品功能
    2.3 用户特点
    2.4 一般约束
    2.5 假设和依据
3 具体需求
  （参阅本指南6.3.2 条中具体需求的组织形式）
附录
索引
```

### 6.1 前言（SRS 第 1 章）

本章提供整个 SRS 综述。

#### 6.1.1 目的（SRS 的 1.1 条）

在这一条包括下列内容：

1）描述实际 SRS 的目的；

2）说明 SRS 所预期的读者。

### 6.1.2 范围（SRS 的 1.2 条）

1）通常应考虑到，若要为软件选择高层次的设计，就可能需要大量的资源（可能占整个产品开发成本的 10%-20%以上）。有两种选择：

2）用一个名字标识被生产的软件产品。比如：×××数据库系统，报表生成程序等等；

3）说明软件产品将干什么，如果需要的话，还要说明软件产品不干什么；

4）描述所说明的软件的应用。应当：

- 尽可能精确地描述所有相关的利闪、目的、以及最终目标。
- 如果有一个较高层次的说明存在，则应该使其和高层次说明中的类似的陈述相一致（例如，系统的需求规格说明）。

### 6.1.3 定义、缩写词、略语（SRS 的 1.3 条）

本条中必须提供全部需求的术语、缩写词及略语的定义，以便对 SRS 进行适当的解释。这些信息可以由 SRS 的附录提供。也可以参考其他的文件。

### 6.1.4 参考资料（SRS 的 1.4 条）

本条应包括：

1）在 SRS 中各处参照的文件的全部清单，如经核准的计划任务书，上级机关批文、合同等；

2）列出其他参考资料，如属本项目的其他已发表的文件和主要文献等。每一个文件、文献要有标题，索引号或文件号，发布或发表日期以及出版单位；

3）详细说明可以得到该参考文件的来源。这个信息可以通过引用附录或其他文件提供。

## 6.2 项目概述（SRS 第 2 章）

本章应描述影响产品和其需求的一般因素，本章不说明具体的需求，而仅使需求更易于理解。

### 6.2.1 产品描述（SRS 的 2.1 条）

这一条是把一个产品用其他有关的产品或项目来描述。

1）如果这个产品是独立的，而且全部内容自含，应在此说明；

2）如果 SRS 定义的产品是一个较大的系统或项目中的一个组成部分，那么本条应包括如下内容：

- 要概述这个较大的系统或项目的每个组成部分的功能，并说明其接口；
- 指出该软件产品主要的外部接口。在这里，不要求对接口详细地描述，详细描述放在 SRS 其他章条中；
- 描述所使用的计算机硬件、外围设备。这里仅仅是一个综述性描述。

在本条的描述中，用一个方框图来表达一个较大的系统或项目的主要组成部分、相互联系和外部接口是非常有帮助的。

本条既不用来强迫进行设计方案的描述，也不是描述在解决问题时的设计约束。本条应对在以后具体需求一章中说明的设计约束提供理由。

### 6.2.2 产品功能（SRS 的 2.2 条）

本条是为将要完成的软件功能提供一个摘要。例如，对于一个记帐程序来说，SRS 可以用这部分来描述：客户帐目维护、客户财务报表和发票制作，而不必把功能所要求的大量的细节描写出来。

有时，如果存在较高层次的规格说明时，则功能摘要可直接从中取得，这个较高层次的规格说明为软件产品分配了特殊的功能，为了清晰起见，请注意：

1）编制功能的一种方法是制作功能表，以便客户或者第一次读这个文件的人都可以理解；

2）用方框图来表达不同的功能和它们的关系也是有帮助的。但要牢记，这样的图不是产品设计时所需求的，而只是一种有效的解释性的工具。

这一条不用作陈述具体需求，只是对后来 SRS 中具体需求一章中为什么要描述的某些需求提供理由。

### 6.2.3 用户特点（SRS 的 2.3 条）

本条要描述影响具体需求的产品的最终用户的一般特点。

许多人在软件生存周期的操作和维护阶段与系统相关。而这些人中有用户、操作员、维护人员和系统工作人员。这些人的某些特点，象教育水平、经验、技术、专长等，都是施加于系统操作环境的重要约束。

如果系统的大多数用户是一些临时用户，那么就要求系统包含如何完成基本功能的提示，而不是假设用户已经从过去的会议或从阅读用户指南中了解到这些细节。

这一条的内容不能用来陈述具体需求或强加若干特殊的设计约束，本条应对在 SRS 的具体需求一章之中的某些具体需求或设计约束的描述提供理由。

### 6.2.4 一般约束（SRS 的 2.4 条）

本条对设计系统阳限制开发者选择的其他一些项作一般性描述。而这些项将限定开发者在设计系统时的任选项。这些包括：

1）管理方针；

2）硬件的限制；

3）与其他应用间的接口；

4）并行操作；

5）审查功能；

6）控制功能；

7）所需的高级语言；

8）通信协议；

9）应用的临界点；

10）安全和保密方面的考虑。

本条不陈述具体需求或具体设计约束：而对 SRS 的具体需求一章中为什么要确定某些具体需求和设计约束提供理由。

### 6.2.5 假设和依据（SRS 的 2.5 条）

本条列出影响 SRS 中陈述的需求的每一个因素。这些因素不是软件的设计约束，但是它们的改变可能影响到 SRS 中的需求。例如：假定一个特定的操作系统是在被软件产品指定的硬件上使用的，然而，事实上这个操作系统是不可能使用的，于是，SRS 就要进行相应的改变。

## 6.3 具体需求（SRS 的第 3 章）

本章应包括软件开发者在建立设计时需要的全部细节。这是 SRS 中篇幅最大和最重要的部分。

1）根据本指南第 4 章所规定的准则（如可验证性、无歧义性等），对每一个需求细节作具体描述；

2）在 SRS 的前言、项目概述、附录部分的有关讨论中，要提供对任何一个具体

需求交叉引用的背景；

3） 具体需求分类的方法如下：

- 功能需求；
- 性能需求；
- 设计约束；
- 属性；
- 外部接口需求。

本章中要注意的二点是：

1） 符合逻辑的和可读的方式组织；

2） 详细描述每个需求，使该需求应达到目标能够用指定的方法进行客观的验证。

### 6.3.1 具体需求的内容

#### 6.3.1.1 功能需求

本条描述软件产品的输入怎样变换成输出。即软件必须完成的基本动作。 对于每一类功能或者有时对于每一个功能，需要具体描述其输入、加工和输出的需求。这通常由四个部颁组成：

1） 引言

这部分描述的是功能要达到的目标、所采用的方法和技术，还应清楚说明功能意图的由来和背景。

2） 输入

这部分应包括：

- 详细描述该功能的所有输入数据，如：输入源、数量、度量单位、时间设定、有效输入范围（包括精度和公差）；
- 操作员控制细节的需求。其中有名字、操作员活动的描述、控制台或操作员的位置。例如：当打印检查时，要求操作员进行格式调整；
- 指明引用接口说明或接口控制文件的参考资料。

3） 加工

定义输入数据、中间参数，以获得预期输出结果的全部操作。它包括如下的说明：

- 输入数据的有效性检查；
- 操作的顺序，包括事件的时间设定；
- 异常情况的响应，例如，溢出、通信故障、错误处理等；
- 受操作影响的参数；
- 降级运行的要求；
- 用于把系统输入变换成相应输出的任何方法（方程式、数学算法、逻辑操作等）；
- 输出数据的有效性检查。

4） 输出

这部分应包括：

- 详细描述该功能所有输出数据，例如：输出目的地、数量、度量单位、时间关系、有效输出的范围（包括精度和公差）、非法值的处理、出错信息；
- 有关接口说明或接口控制文件的参考资料。

此外，对着重于输入输出行为的系统来说，SRS 应指定所有有意义的输入、输出对及其序列。当一个系统要求记忆它的状态时，需要这个序列，使得它可以根

据本次输入和以前的状态作出响应。也就是说，这种情况犹如有限状态机。

#### 6.3.1.2 设计约束

设计约束受其他标准、硬件限制等方面的影响。

1）其他标准的约束：本项将指定由现有的标准或规则派生的要求。例如：报表格式、数据命名、财务处理、审计追踪等等。

2）硬件的限制：本项包括在各种硬件约束下运行的软件要求，例如，应该包括：硬件配置的特点（接口数，指令系统等）、内存储器和辅助存储器的容量。

#### 6.3.1.3 属性

在软件的需求之中有若干个属性，下面指出其中的几个（注意：对这些决不应理解为是一个完整的清单）。

1）可用性：可以指定一些因素，如检查点、恢复和再启动等，以保证整个系统有一个确定的可用性级别。

2）安全性：这里指的是保护软件的要素，以防止各种非法的访问、使用，修改、破坏或者泄密。这个领域的具体需求必须包括：

- 利用可靠的密码技术；
- 掌握特定的记录或历史数据集；
- 给不同的模块分配不同的功能；
- 限定一个程序中某些区域的通信；
- 计算临界值的检查和。

3）可维护性：这里规定若干需求以确保软件是可维护的。例如：

- 软件模块所需要的特殊的耦合矩阵；
- 对微型装置指定特殊的数据/程序分割要求。

4）可转移/转换性：这里规定把软件从一种环境移植到另一种环境所要求的用户程序，用户接口兼容方面的约束等等。

5）警告：指定所需属性十分重要，它使得人们能用规定的方法去进行客观的验证。

#### 6.3.1.4 外部接口要求

1）用户接口：提供用户使用软件产品是地的接口需求。例如，如果系统的用户通过显示终端进行操作，就必须指定如下要求：

- 对屏幕格式的要求；
- 报表或菜单的页面打印格式和内容；
- 输入输出的相对时间；
- 程序功能键的或用性。

2）硬件接口：要指出软件产品和系统硬部件之间每一个接口的逻辑特点。还可能包括如下事宜：支撑什么样的设备，如何支撑这些设备，有何约定。

3）软件接口：在这里应指定需使用的其他软件产品（例如，数据管理系统，操作系统，或者数学软件包），以及同其他应用系统之间的接口。 对每一个所需的软件产品，要提供名字、助记符、规格说明号、版本号、来源等内容。 对于每一个接口，这部分应说明与软件产品相关的接口软件的目的，并根据信息的内容和格式定义接口，这里不必详细描述任何已有完整文件的接口，只要引用定义该接口的文件即可。

4）通信接口：这里指定各种通信接口，例如，局部网络的协议等等。

#### 6.3.1.5 其他需求

根据软件和用户组织的特性等，某些需求放在下面各项中描述。

1）数据库：本项对作为产品的一部分进行开发的数据库规定一些需求，它们可能包括：

- 在 6.3.1.1 条中标识的信息类别；
- 用的频率；
- 存取能力；
- 数据元素和文卷描述符；
- 数据元素、记录和文卷的关系；
- 静态和动态的组织；
- 数据保存要求。

*注：如果使用一个现有的数据库包，这个包应在“软件接口”中命名，并在那里详细说明其用法。*

2）操作：这里说明用户要求的常规的和特殊的操作。

- 在用户组织之中各种方式的操作。例如，用户初始化操作；
- 交互作用操作的同期和无人操作的周期；
- 数据处理支持功能；
- 后援和恢复操作。

*注：这里的内容有时是用户接口的一部分。*

3）场合适应性需求：这里包括：

- 对给定场合、任务或操作方式的任何数据或初始化顺序的需求进行定义。例如，栅值，安全界限等等。
- 指出场合或相关任务的特点，这里可以被修改以使软件适合特殊配制的要求。

### 6.3.2 具体要求的组织

本条通常是 SRS 所有部分中最大并且最复杂的部分。

1）可以根据软件实现功能的基本类型，将本条分成若干段。例如：考虑一个大的交互记帐系统，在里层可以分为操作软件（它支持近乎实时的事务处理）、支撑软件（联机功能、磁盘备份、装入磁带等等）以及诊断软件（诊断硬件、通信等），外一层是应收款帐以及应付款帐等等；

2）结构细分的目的是提高 SRS 的可读性，而不是进行概要设计。

对于 SRS 中的第 3 章的具体需求部分的最好的组织方案取决于所说明的软件产品的应用范围和性质。 文中最后部分提供了四种可能的组织方案。

1）大纲 1 中首先说明全部功能需求，然后说明四种类型的接口要求，最后是其他需求；

2）大纲 2 中，把对应每个特定功能的四种接口需求和该功能需求放在一起描述，然后说明其他需求；

3）大纲 3 中，与功能需求有关的全部内容放在一起首先说明，然后是其他需求的描述。对每一种外部接口的需求重复上述过程；

4）大纲 4 中，接口需求和其余的需求作为每一个功能需求的附属部分来说明。

SRS 的具体需求的组织形式必须选择可读性最好的方法来描述。

## 6.4 支持信息

支持信息是指目录表，附录和索引。以便使 SRS 易于使用。

1）目录表和索引很重要，而且应按照可以接受的好的文件规则来编写。

2）对一个实际的需求规格说明来说，若有必要应该编写附录。附录中可能包括：

- 输入输出格式样本，成本分析研究的描述或用户调查结果；
- 有助于理解 SRS 的背景信息；
- 软件所解决问题的描述；
- 用户历史、背景、经历和操作特点；
- 交叉访问表。按先后次序进行编排，使一些不完全的软件需求得以完善（参见 4.3.2 条和 4.3.3 条）；
- 特殊的装配指令用于编码和媒体，以满足安全、输出、初始装入或其他要求。

3）6.4.3 当包括附录时，SRS 必须明确地说明附录是不是需求要考虑的部分。

**SRS 大纲 1**

3 具体需求
- 3.1 功能需求
  - 3.1.1 功能需求 1
    - 3.1.1.1 引言　3.1.1.2 输入　3.1.1.3 加工　3.1.1.4 输出
  - 3.1.2 功能需求 2
  - ……
- 3.1.n 功能需求 n
- 3.2 外部接口需求
  - 3.2.1 用户接口　3.2.2 硬件接口　3.2.3 软件接口　3.2.4 通信接口
- 3.3 性能需求
- 3.4 设计约束
  - 3.4.1 其他标准的约束　3.4.2 硬件的限制　……
- 3.5 属性
  - 3.5.1 安全性　3.5.2 可维护性　……
- 3.6 其他需求
  - 3.6.1 数据库　3.6.2 操作　3.6.3 场合适应性　……

**SRS 大纲 2**

3 具体需求
- 3.1 功能需求
  - 3.1.1 功能需求 1
    - 3.1.1.1 规格说明
      - 3.1.1.1.1 引言　3.1.1.1.2 输入　3.1.1.1.3 加工　3.1.1.1.4 输出
    - 3.1.1.2 外部接口
      - 3.1.1.2.1 用户接口　3.1.1.2.2 硬件接口　3.1.1.2.3 软件接口
      - 3.1.1.2.4 通信接口
  - 3.1.2 功能需求 2
  - ……

**SRS 大纲 3**

3 具体需求

3.1 功能需求

3.1.1 功能需求 1

3.1.1.1 引言　3.1.1.2 输入　3.1.1.3 加工　3.1.1.4 输出

3.1.1.5 性能需求

3.1.1.6 设计约束

3.1.1.6.1 其他标准的约束　3.1.1.6.2 硬件的限制　……

3.1.1.7 属性

3.1.1.7.1 安全性　3.1.1.7.2 可维护性　……

3.1.1.8 其他需求

3.1.1.8.1 数据库　3.1.1.8.2 操作　3.1.1.8.3 场合适应性

……

3.1.2 功能需求 2

……

3.1.n 功能需求 n

3.2 外部接口需求

3.2.1 用户接口

3.2.1.1 性能需求

3.2.1.2 设计约束

3.2.1.2.1 其他标准的约束　3.2.1.2.2 硬件的限制　……

3.2.1.3 属性

3.2.1.3.1 安全性　3.2.1.3.2 可维护性　……

3.2.1.4 其他需求

3.2.1.4.1 数据库　3.2.1.4.2 操作　3.2.1.4.3 场合适应性

……

3.2.2 硬件接口　3.2.3 软件接口　3.2.4 通信接口

3.1 功能需求 1

3.1.1 引言　3.1.2 输入　3.1.3 加工　3.1.4 输出

3.1.5 外部接口

3.1.5.1 用户接口　3.1.5.2 硬件接口

3.1.5.3 软件接口　3.1.5.4 通信接口

3.1.6 性能需求

3.1.7 设计约束

3.1.8 属性

3.1.8.1 安全性　3.1.8.2 可维护性　……

# 用例说明模板 1(经典模板)

编者说明：

随着 UML 的日益普及，用例（Use case）分析技术也在需求实践中广泛被采用。但是也有许多团队在使用该技术时，只画出了用例图，而缺少了用例说明，其实这是一个严重的误区。而本模板就将指导你编写该说明。

1.用例名称

1.1 简要说明

[简要说明用例的作用和目的。该小节的篇幅不要太长。]

2.上下文图

[在此小节中，有一个只包括本用例和所有与该用例相关的 Actor 和其它用例组成的，一个用例图的局部。]

3. 事件流

3.1 基本流

[当 Actor 采取行动时，用例也就随即开始。用例总是由 Actor 启动的，用例应说明 Actor 的行为及系统的响应，可按照 Actor 与系统进行对话的形式来逐步引入用例。]

[要注意的是，用例描述应该说明系统内发生的事情，而不是事件发生的方式与原因。如果进行了信息交换，则需指出来回传递的具体信息。例如，只表述主角输入了客户信息就不够明确。最好明确地说主角输入了客户姓名和地址。当然你也可以通过项目词汇表来定义这些信息，使得用例中的内容被简化，从而不致于让用例描述陷入过多的细节内容。]

[如果存在一些相对比较简单的备选流，只需少数几句话就可以说明清楚，那么也可以直接在这一部分中描述。但是如果比较复杂，还是应该单独放在备选流小节中描述。]

[一幅图胜过千言万语，因此建议在这一小节中，除了叙述性文字之外，你还可以引用 UML 中的活动图、顺序图、协作图、状态图等手段，对其进行补充说明。]

3.2 备选流

3.2.1 第一备选流

[正如前面所述，对于较复杂的备选流应单独地说明。]

3.2.1.1 备选支流

[如果能使表达更明确，备选流又可再分为多个支流。]

3.2.2 第二备选流

[在一个用例中很可能会有多个备选流。为了使表达更清晰，应将各个备选流

分开说明。使用备选流可以提高用例的可读性，并防止将用例分解为过多的层次。应切记，用例只是文本说明，其主要目的是以清晰、简洁、易于理解的方式记录系统的行为。]

4. 非功能需求

[在这个小节中，主要对该用例所涉及的非功能性需求进行描述。由于其通常很难以在事件流中进行表述，因此单列为一小节进行阐述。这些需求通过包括法律法规、应用程序标准、质量属性（可用性、可靠性、性能、支持性等）、兼容性、可移植性，以及设计约束等方面的需求。在这些需求的描述方面，一定要注意使其可度量、可验证，否则就容易流于形式，形同摆设。]

5. 前置条件

[用例的前置条件是执行用例之前必须存在的系统状态。]

6. 后置条件

[用例的后置条件是用例一执行完毕系统可能处于的一组状态。]

7. 扩展点

[此用例的扩展点，通常是用例图中的 extent 关系。]

## 用例说明模板 2(单列表格式)

编者说明：

如果你觉得文本描述不够清晰，也可以采用如本文档模板所示的表格式的描述方式。

| | | |
|---|---|---|
| **用例#** | [用例名应是一个动词短语，应让读者一目了然地从名字中就可以知道该用例的目标。] | |
| **使用语境** | [用例目标，是一个较长的描述，甚至包括触发条件。] | |
| **范围** | [用例的设计范围，在设计时将系统作为一个黑盒来考虑。] | |
| **级别** | [概要、用户目标、子功能三者之一。] | |
| **主执行者** | [也就是该用例的主 Actor，在此应列出其名称，并简要描述。] | |
| **项目相关人员利益** | **项目相关人员** | **利益** |
| | [项目相关人员名称] | [项目相关人员取得的利益] |
| | …… | …… |
| **前置条件** | [也就是激发该用例，所应该满足的条件。] | |
| **后置条件** | [也就是该用例完成之后，将执行什么动作。] | |
| **成功保证** | [描述当目标完成后，环境的变化情况。] | |
| **触发事件** | [什么引发用例，例如时间事件。] | |
| **描述** | **步骤** | **活动** |
| | 1 | [在这里写出触发事件到目标完成以及清除的步骤。] |
| | 2 | [……] |
| | 3 | |
| **扩展** | **步骤** | **分支动作** |
| | 1a | [引起分支的条件] |
| | | [活动或子用例名称] |
| **技术和数据变化** | | |
| | 1 | [变化列表] |

## 用例说明模板 3(双列表格式)

编者说明：

本模板是对上一模板的补充，如果你想更好地捕捉系统的响应，那么就可以采用本表格所示的格式。

有时，为了更好地捕获系统的响应，对于场景描述（主成功场景、扩展场景）在上表的基础上变成如下表所示的双列：

| 步骤 | 用户 | 系统 |
|---|---|---|
| | | |
| | | |

## 用例说明模板 4(文本式)

编者说明：

相信用过用例分析技术的，对用例应该多少细有很大的疑问，而 Alistair Cockburn 率先将其进行分级：概要、用户目标、子功能，如果你对他的思想有认同，则该模板就适合于你。

1.用例名：

[用例名应是一个动词短语，应让读者一目了然地从名字中就可以知道该用例的目标。]

2.使用语境：

[用例目标，是一个较长的描述，甚至包括触发条件。]

3.范围：

[用例的设计范围，在设计时将系统作为一个黑盒来考虑。]

4.级别：

[用来表示该用例是在描述哪个级别上的功能，通常包括概要、用户目标、子功能三种。这三种级别的划分是 Alistair Cockburn 在《编写有效用例》一书是提出的。]

5.主执行者：

[也就是该用例的主 Actor，在此应列出其名称，并给予简要描述。]

6. 项目相关人员利益

[说明该用例对项目相关人员能够带来什么好处。]

7. 前置条件：

[也就是激发该用例，所应该满足的条件。]

8. 后置条件：

[也就是该用例完成之后，将执行什么动作。]

9. 成功保证：

[描述当目标完成后，环境的变化情况。]

10. 触发事件：

[什么引发用例，例如时间事件。]

11. 主成功场景

[在这里写出触发事件到目标完成以及清除的步骤。]

[步骤编号＃：动作描述]

[步骤编号＃：动作描述]

……

12. 扩展：

[在这里写出扩展情况，每次写一个扩展，每个扩展都应指向主场景的特定步骤。]

[被改变步骤　条件：动作或子用例]

[被改变步骤　条件：动作或子用例]

……

13. 技术和数据变化列表

[在这里写出场景中因技术或数据变化而引起的可能分支。]

[步骤或变化编号＃：变化列表]

[步骤或变化编号＃：变化列表]

……

14. 相关信息

[项目所需要的所有附加信息。]

## 数据要求说明书(ISO 标准)

编者说明：

如果在你的项目中有大量要求数据存储、数据采集等方面的需求，那么你就应该专门将这些需求进行整理，以数据要求说明书的形式表现出来。

1．引言

1.1 编写目的

[说明编写这份数据要求说明书的目的，指出预期的读者。]

1.2 背景

a.待开发软件系统的名称；

b.列出本项目的任务提出者、开发者、用户以及将运行该项软件的计算站或计算机网络系统。

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出有关的参考资料。]

2．数据的逻辑描述

[对数据进行逻辑描述时可把数据分为动态数据和静态数据。]

2.1 静态数据

[列出所有作为控制或参考用的静态数据元素。]

2.2 动态输入数据

[列出动态输入数据元素。]

2.3 动态输出数据

[列出动态输出数据元素。]

2.4 内部生成数据

[列出向用户或开发单位中的维护调试人员提供的内部生成数据。]

2.5 数据约定

[说明对数据要求的制约。逐条列出对进一步扩充或使用方面的考虑而提出的对数据要求的限制。对于在设计和开发中确定是临界性的限制更要明确指出。]

3．数据的采集

3.1 要求和范围

[按数据元的逻辑分组来说明数据采集的要求和范围，指明数据的采集方法，说明数据采集工作的承担者是用户还是开发者。]

3.2 输入的承担者

[说明预定的对数据输入工作的承担者。如果输入数据同某一接口软件有关，还应说明该接口软件的来源。]

3.3 预期处理

[对数据的采集和预处理过程提出专门的规定，包括适合应用的数据格式、预定的数据通信媒体和对输入的时间要求等。对于需经模拟转换或数字转换处理的数据量，要给出转换方法和转换因子等有关信息，以便软件系统使用这些数据。]

3.4 影响

[说明这些数据要求对于设备、软件、用户、开发单位所可能产生的影响。]

# 三、系统分析与设计类

## 软件体系结构设计说明书

批注[XuFeng7]：注，以RUP《软件构架文档》为基础进行修改。

编者说明：

随着OO方法论地日臻成熟，其思想也从编程（OOP）到了设计（OOD）和分析（OOA），而软件体系结构则是从设计的最高层进行设计与规划的技术，本文档模板就是用来帮助你从用例视图、逻辑视图、进程视图、部署视图等方面对系统进行总体描述。

1.文档简介

[本节主要是描述软件体系结构设计说明书的目的、范围、相关术语、参考资料和本文档的摘要性介绍。软件体系结构设计属于高层设计文档，是符合现代软件工程要求的概要设计。]

1.1 目的

[软件体系结构设计说明书，将从设计的角度对系统进行综合的描述，使用不同的视图来描述其不同方面。在本小节中，将对该文档的结构进行简要的说明，明确该文档针对的读者群，指导他们正确的地使用该文档。]

1.2 范围

[说明该文档所涉及的内容范围，以及将影响的内容。]

1.3 定义、首字母缩写词和缩略语

[与其它文档一样，该文档也需要将本文档中所涉及的所有术语、缩略语进行详细的定义。还有一种可简明的做法，就是维护在一个项目词汇表中，这样就可以避免在每个文档中都重复很多内容。]

1.4 参考资料

[在这一小节中，应完整地列出该文档引用的所有文档。对于每个引用的文档都应该给出标题、标识号、日期以及来源，为阅读者查找这些文档提供足够详细的信息。]

1.5 概述

[在本小节中，主要是说明软件体系结构设计说明书各个部分所包含的主要内容，就像一个文章摘要一样。同时也应该对文档的组织方式进行解释。]

2. 体系结构表示方式

[本节说明软件体系结构在当前系统中的作用及其表示方式。它将列举其所必需的用例视图、逻辑视图、进程视图、部署视图或实施视图，并分别说明这些视图包含哪些类型的模型元素。]

3. 软件体系结构的目标和约束

[本节说明对软件体系结构具有某种重要影响的软件需求和用户目标，例如，系统安全

性、保密性、第三方组件的使用、可移植性、发布和重新使用。它还要记录可能适用的特殊约束：设计与实施策略、开发工具、团队结构、时间表、遗留系统等。]

4.用例视图

[本节使用用例分析技术所生成的系统用例模型，描述其中的一些用例或场景。在该模型中纳入用例或场景，应该是系统中最重要、最核心的功能部分。]

[另外，在本节中还应该选择一个主要的用例，对其进行描述与解释，以帮助读者了解软件的实际工作方式，解释不同的设计模型元素如何帮助系统实现。]

5. 逻辑视图

[逻辑视图主要是反映系统本质的问题领域类模型，在逻辑视图中将列出组成系统的子系统、包。而对每个子系统、包分解成为一个个类，并说明这些关键的实体类的职责、关系、操作、属性。这也是 OO 思想的体现，以类、类与类之间的协作、包、包与包之间的协作模型来表达系统的逻辑组织结构。]

5.1 概述

[在本小节中，列出逻辑视图的顶层图，该图将反映系统由哪些包组成，每个包之间的关系与协作，以及包的层次结构。使得读者对整个软件体系结构有一个整体的了解。]

5.2 影响软件体系结构的重要设计包

[在本小节中，将从逻辑视图中选择有重要意义的设计包，每个设计包有一个小节来描述，说明这些包的名称、简要的说明、该包中的主要类和相关的类图。对于包中的重要的类，还应该说明其名称、简要说明、主要职责、操作、属性等。]

6. 进程视图

[本节主要描述该软件体系结构下，系统运行态的情况。描述系统在执行时，包括哪些进程（包括线程、进程、进程组），以及它们之间是如何进行通信的、如何进行消息传递、接口如何。并且来说明如何进行组织。]

7.部署视图

[本节主要描述该软件系统部署后的样子，需要哪些硬件、支撑软件、网络环境。在每个物理节点上所运行的模块，它们之间是如何连接的，这些物理节点与进程之间的映射关系等等。]

8.实施视图

[本节主要从开发的角度来描述软件系统架构，包括其整体结构、层次结构、子系统，以及要使用的第三方控件，自定义控件，以及它们之间的接口。]

8.1 概述

[在本小节中，说明各个层的内容、边界与交互，通常用 UML 中的构件图进行表示。]

8.2 层

[本小节则是在上一小节的基础上，对每一个层进行说明，并给出每一个层的构件图，帮助读者分而治之。]

## 概要设计说明书(ISO 标准)

编者说明：

这是 ISO 提供的规范，是最原始的概要设计说明书的编写格式，其适用于结构化设计思想下的软件设计，不过其中还是有很多具有参考价值的内容。

1．引言

1.1 编写目的

[说明编写这份概要设计说明书的目的，指出预期的读者。]

1.2 背景

a.[待开发软件系统的名称；]

b.[列出本项目的任务提出者、开发者、用户。]

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出有关的参考资料。]

2．总体设计

2.1 需求规定

[说明对本系统的主要的输入输出项目、处理的功能性能要求。包括]

2.1.1 系统功能

2.1.2 系统性能

2.1.2.1 精度

2.1.2.2 时间特性要求

2.1.2.3 可靠性

2.1.2.4 灵活性

2.1.3 输入输出要求

2.1.4 数据管理能力要求

2.1.5 故障处理要求

2.1.6 其他专门要求

2.2 运行环境

[简要地说明对本系统的运行环境的规定。]

2.2.1 设备

[列出运行该软件所需要的硬设备。说明其中的新型设备及其专门功能。]

2.2.2 支持软件

[列出支持软件，包括要用到的操作系统、编译（或汇编）程序、测试支持软件等。]

2.2.3 接口

[说明该系统同其他系统之间的接口、数据通信协议等]

2.2.4 控制

[说明控制该系统的运行的方法和控制信号，并说明这些控制信号的来源。]

2.3 基本设计概念和处理流程

[说明本系统的基本设计概念和处理流程，尽量使用图表的形式。]

2.4 结构

[给出系统结构总体框图（包括软件、硬件结构框图），说明本系统的各模块的划分，扼要说明每个系统模块的标识符和功能，分层次地给出各模块之间的控制与被控制关系。]

2.5 功能需求与系统模块的关系

[本条用一张矩阵图说明各项功能需求的实现同各模块的分配关系。]

| | [系统模块 1] | [系统模块 2] | [……] | [系统模块 m] |
|---|---|---|---|---|
| [功能需求 1] | √ | | | |
| [功能需求 2] | | √ | | |
| [⋮] | | | | |

| | | | | |
|---|---|---|---|---|
| [功能需求 n] | | √ | | √ |

2.6 人工处理过程

[说明在本系统的工作过程中不得不包含的人工处理过程。]

2.7 尚未解决的问题

[说明在概要设计过程中尚未解决而设计者认为在系统完成之前必须解决的各个问题。]

3．接口设计

3.1 用户接口

[说明将向用户提供的命令和它们的语法结构，以及相应的回答信息。]

[说明提供给用户操作的硬件控制面板的定义。]

3.2 外部接口

[说明本系统同外界的所有接口的安排包括软件与硬件之间的接口、本系统与各支持系统之间的接口关系。]

3.3 内部接口

[说明本系统之内的各个系统元素之间的接口的安排。]

4．运行设计

4.1 运行模块组合

[说明对系统施加不同的外界运行控制时所引起的各种不同的运行模块组合，说明每种运行所历经的内部模块的支持软件。]

4.2 运行控制

[说明每一种外界的运行控制的方式方法和操作步骤。]

4.3 运行时间

[说明每种运行模块组合将占用各种资源的时间。]

5．系统数据结构设计

[不涉及软件设计可不包含]

5.1 逻辑结构设计要点

[给出本系统内软件所使用的每个数据结构的名称、标识符以及它们之中每个数据项、记录、文卷和系的标识、定义、长度及它们之间的层次的或表格的相互关系。]

5.2 物理结构设计要点

[给出本系统内软件所使用的每个数据结构中的每个数据项的存储要求，访问方法、存取单位、存取的物理关系、设计考虑和保密条件。]

5.3 数据结构与程序的关系

[说明各个数据结构与访问这些数据结构的各个程序之间的对应关系。]

| | [程序 1] | [程序 2] | [……] | [程序 m] |
|---|---|---|---|---|
| [数据结构 1] | √ | | | |
| [数据结构 2] | √ | √ | | |
| ⋮ | | | | |
| [数据结构 n] | | √ | | √ |

6．系统出错处理设计

6.1 出错信息

[用一览表的方式说明每种可能的出错或故障情况出现时，系统输出信息的形式、含意及处理方法。]

6.2 补救措施

[说明故障出现后可能采取的变通措施。包括：]

a. 后备技术 [说明准备采用的后备技术，当原始系统数据万一丢失时启用的副本的建立和启动的技术，例如周期性地把磁盘信息记录到磁带上去就是对于磁盘媒体的一种后备技术。]

b. 降效技术 [说明准备采用的后备技术，使用另一个效率稍低的系统或方法来求得所需结果的某些部分，例如一个自动系统的降效技术可以是手工操作和数据的人工记录。]

c. 恢复及再启动技术 [说明将使用的恢复再启动技术，使软件从故障点恢复执行或使软件从头开始重新运行的方法。]

6.3 系统维护设计

[说明为了系统维护的方便而在程序内部设计中作出的安排，包括在程序中专门安排用于系统的检查与维护的检测点和专用模块。]

## 概要设计说明书模板 2

编者说明：

这也是一个面向结构化设计思想的概要设计说明书模板，其在 ISO 规范的基础上提供了一些更加直观的方式，是一个很有价值的模板。

第 1 章 引言

1.1 编写目的

[说明对程序系统的设计考虑，包括程序系统的基本处理流图、程序系统的组织结构、模块划分、功能分配、接口设计、运行设计、数据结构设计和安全性设计等。为程序的详细设计奠定基础。]

1.2 术语

| 序号 | 术语或缩写词 | 说明性定义 |
|---|---|---|
| | | |
| | | |

1.3 参考文献

| 序号 | 资料名 | 文件编号 | 发表日期 | 出版单位 |
|---|---|---|---|---|
| | | | | |

第 2 章 系统概述

2.1 系统说明

任务提出单位：

开发单位：

预期用户：

2.2 系统任务

2.2.1 系统目标

2.2.2 运行环境

2.2.3 与其它系统关系

2.3 需求规定

2.3.1 功能需求

2.3.2 性能需求

2.3.3 数据要求

2.3.4 其它

第 3 章 总体设计

3.1 系统物理结构

3.1.1 系统流程图

3.1.2 设备清单

| 序号 | 设备名称 | 数　量 | 型号和规格 |
|---|---|---|---|
| | | | |

3.2 软件结构图

3.2.1 模块结构图

3.2.2 模块清单

| 编号 | 模块名称 | 模块标识 |
|---|---|---|
| | | |

第 4 章 模块功能描述

4.1 模块 1（标识符） 功能

| 模块编号： | 模块名称： | 模块标识符： |
|---|---|---|
| 输入 | 处理 | 输出 |
| | | |

4.2 模块 2 （标识符）功能

第 5 章 接口设计

5.1 用户界面

5.2 硬件接口

5.3 软件接口

5.4 通信接口

第 6 章 数据结构设计

6.1 数据结构 1 （标识符）

6.1.1 结构属性

| 结构名称 | | 逻辑标识 | | 物理标识 | |
|---|---|---|---|---|---|
| 结构类型 | | 存储模式 | | 存储介质 | |
| 访问模式 | | 读/写方式 | | | |
| 记录标识 | | 记录长度 | | | |

6.1.2 逻辑结构

6.1.3 物理结构

6.1.4 数据元素

6.2 数据结构 2 （标识符）

第 7 章 运行设计

7.1 运行 1

7.1.1 运行模块组合运行名称

| 模块集合 | 运行条件 | 支持软件 |
|---|---|---|
| | | |

7.1.2 运行控制操作

| 运行名称 | 控制方法 | 操作步骤 |
|---|---|---|
| | | |

7.1.3 运行时间

| 运行名称 | 所占资源 | 时间 |
|---|---|---|
| | | |

7.2 运行 2

第 8 章　系统安全

8.1　系统安全

[1、系统安全控制和物理保护措施；]

[2、用户身份鉴别机制；]

[3、用户对系统的访问权限和范围。]

8.2　数据安全

[1、数据用户身份鉴别；]

[2、访问主体、访问对象的控制策略和实现方法；]

[3、数据加密方法。]

8.3　后备与恢复

[1、系统后备；]

[2、数据后备；]

[3、系统恢复；]

[4、数据恢复。]

8.4　出错处理

[1、出错情况；]

[2、出错信息输出形式、信息含义、处理方法；]

[3、出错失效的后备措施。]

8.5　计算机病毒的防治措施

第 9 章　功能需求、数据结构和模块

9.1　功能需求与模块关系

| 功能<br>模块 | 功能 1 | 功能 2 | … |
|---|---|---|---|
| 模块 1 | | U | |
| 模块 2 | U | U | |
| 模块 3 | U | | |
| … | | | |

9.2　数据结构与模块关系

| 数据<br>模块 | 数据结构 1 | 数据结构 2 | … |
|---|---|---|---|
| 模块 1 | U | C | |
| 模块 2 | U | U | |
| | | | |

## 数据库设计说明书(ISO 版)

编者说明：

如果你的项目中有很多与数据库相关的内容，则你应该把数据库设计、表结构设计、视图设计统一整理形成数据库设计说明书。

1.引言

1.1 编写目的

[说明编写这份数据设计说明书的目的，指出预期的读者。]

1.2 背景

a.[待开发数据库的名称和使用此数据库的软件系统的名称；]

b.[列出本项目的任务提出者、开发者、用户。]

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出有关的参考资料。]

2．外部设计

2.1 标识符的状态

[联系用途，详细说明用于唯一地标识该数据库的代码、名称或标识符，附加的描述性信息亦要给出。如果该数据库属于尚在实验中、尚在测试中或是暂时使用的，则要说明这一特点及其有效时间范围。]

2.2 使用它的程序

[列出将要使用或访问此数据库的所有应用程序，对于这些应用程序的每一个，给出它的名称和版本号。]

2.3 约定

[陈述一个程序员或一个系统分析员为了能使用此数据库而需要了解的建立标号、标识的约定。]

2.4 专门指导

[向准备从事此数据库的生成、从事此数据库的测试、维护人员提供专门的指导。]

2.5 支持软件

[简单介绍同此数据库直接有关的支持软件。说明这些软件的名称、版本号的主要功能特性。列出这些支持软件的技术文件的标题、编号及来源]

3．结构设计

3.1 概念结构设计

[说明本数据库将反映的现实世界中的实体、属性和它们之间的关系等的原始数据形式，包括各数据项、记录、系、文卷的标识符、定义、类型、度量单位和值域，建立本数据库的每一幅用户视图。]

3.2 逻辑结构设计

[说明把上述原始数据进行分解、合并后重新组织起来的数据库全局逻辑结构。]

3.3 物理结构设计

[建立系统程序员视图。]

4．运用设计

4.1 数据字典设计

[对数据库设计中涉及到的各种项目一般要建立起数据字典，以说明它们的标识符、同义名及有关信息。]

4.2 安全保密设计

[说明在数据库的设计中，将如何通过区分不同的访问者、不同的访问类型和不同的数据对象，进行分别对待而获得的数据库安全保密的设计考虑。]

## 详细设计说明书(ISO 标准)

编者说明：

概要设计通常是项目中专门的人员完成，是对系统的高层描述，而详细设计的任务则通常由每一个任务实施人来完成，其是对某个具体的模块、类等局部元素的设计描述。该模板是 ISO 推荐的格式，其仍然是以结构化设计为主要思想。

1.引言

1.1 编写目的

[说明编写这份详细设计说明书的目的，指出预期的读者。]

1.2 背景

a. [待开发系统的名称；]

b. [列出本项目的任务提出者、开发者、用户。]

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出有关的参考资料。]

2. 系统的结构

[给出系统的结构框图，包括软件结构、硬件结构框图。用一系列图表列出系统内的每个模块的名称、标识符和它们之间的层次结构关系。]

3．模块 1（标识符）设计说明

[从本章开始，逐个地给出各个层次中的每个模块的设计考虑。以下给出的提纲是针对一般情况的。对于一个具体的模块，尤其是层次比较低的模块或子程序，其很多条目的内容往往与它所隶属的上一层模块的对应条目的内容相同，在这种情况下，只要简单地说明这一点即可。]

3.1 模块描述

[给出对该基本模块的简要描述，主要说明安排设计本模块的目的意义，并且，还要说明本模块的特点。]

3.2 功能

[说明该基本模块应具有的功能。]

3.3 性能

[说明对该模块的全部性能要求。]

3.4 输入项

[给出对每一个输入项的特性。]

3.5 输出项

[给出对每一个输出项的特性。]

3.6 设计方法（算法）

[对于软件设计，应详细说明本程序所选取用的算法，具体的计算公式及计算步骤。]

[对于硬件设计，应详细说明本模块的设计原理、元器件的选取、各元器件的逻辑关系，所需要的各种协议等。]

3.7 流程逻辑

[用图表辅以必要的说明来表示本模块的逻辑流程。]

3.8 接口

[说明本模块与其它相关模块间的逻辑连接方式，说明涉及到的参数传递方式。]

3.9 存储分配

[根据需要，说明本模块的存储分配。]

3.10 注释设计

[说明安排的程序注释。]

3.11 限制条件

[说明本模块在运行使用中所受到的限制条件。]

3.12 测试计划

[说明对本模块进行单体测试的计划，包括对测试的技术要求、输入数据、预期结果、进度安排、人员职责、设备条件、驱动程序及桩模块等的规定。]

3.13 尚未解决的问题

[说明在本模块的设计中尚未解决而设计者认为在系统完成之前应解决的问题。]

4．模块 2（标识符）设计说明

[用类似第 3 条的方式，说明第 2 个模块乃至第 N 个模块的设计考虑。]

## 详细设计说明书模板 2

编者说明：

该模板也是以结构化设计的主要思想，在 ISO 标准的基础上进行了适当的修改与完善。如果你采用的是面向对象的思想，那么通常该文档被类图、顺序图、交互图、活动图、状态图等描述类静态结构与动态行为的图表所代替，而不再专门的形成文档。

1.引言

1.1 编写目的

[说明软件系统各个层次的每个程序（每个模块或子程序）的设计考虑。]

1.2 系统说明

[任务提出单位：]

[开发单位：]

[预期用户：]

1.3 术语

| 序号 | 术语或缩写词 | 说明性定义 |
|---|---|---|
| | | |

1.4 参考资料

| 序号 | 资料名 | 文件编号 | 发表日期 | 出版单位 |
|---|---|---|---|---|
| | | | | |

2.软件结构

2.1 软件结构图

[模块结构图]

2.2 模块子结构图

[1.模块内部结构图；]

[2.子模块清单。]

| 编 号 | 子模块名称 | 子模块标识符 | 父模块名称 |
|---|---|---|---|
| | | | |

2.3 模块清单

| 编 号 | 模块名称 | 模块标识符 |
|---|---|---|
| | | |

3. 模块设计

3.1 模块 1 （标识符）

3.1.1 模块概述

[包括模块的简要情况以及属性。]]

3.1.2 功能和性能

3.1.2.1 （标识符）功能（IPO 图）

| 输入 | 处理 | 输出 |
|---|---|---|

| | | |
|---|---|---|
| | | |

3.1.2.2 性能

3.1.3 输入/输出项

3.1.3.1 输入项

| 名称 | 标识符 | 类型 | 介质 | 来源 | 描述 |
|---|---|---|---|---|---|
| | | | | | |

3.1.3.2 输出项

| 名称 | 标识符 | 类型 | 介质 | 来源 | 描述 |
|---|---|---|---|---|---|
| | | | | | |

3.1.4 数据结构

3.1.4.1 全局数据结构

| 名称 | 标识符 | 类型 | 使用方式 | 访问方式 | 描述 |
|---|---|---|---|---|---|
| | | | | | |

3.1.4.2 局部数据结构

| 名称 | 标识符 | 类型 | 使用方式 | 访问方式 | 描述 |
|---|---|---|---|---|---|
| | | | | | |

3.1.5 算法

[N－S 图、ＰＡＤ图或ＰＤＬ语言。]

3.1.6 限制条件

[模块的所有限制条件。]

3.1.7 测试计划

[1.驱动模块和桩模块；]

[2.前置条件；]

[3.测试用例：输入和预期结果。]

3.2 模块 2

# 四、软件质量保证类

## 测试计划

编者说明：

要想系统性地完成一件事，首先要做好计划，测试工作是十分重要的，因此测试计划也是十分必要的。该文档适用于集成测试、系统测试、验收测试的计划制订，并不适用于单元测试计划。

第 1 章 引言

1.1 综述

1.2 参考文献

| 序号 | 名称 | 文件标识/版本 | 出版单位 | 出版日期 |
|---|---|---|---|---|
| | | | | |

第 2 章 测试项

2.1 测试项

| 测试项名称 | 测试项标识 | 介质特性 | 变换要求 | 相关引用材料 |
|---|---|---|---|---|
| | | | | |

2.2 不测试的软件项

| 软件项名称 | 软件项标识 | 未测试原因 | 相关引用材料 |
|---|---|---|---|
| | | | |

第 3 章 被测试的特性

| 特性或组合名称 | 测试设计说明编号 |
|---|---|
| | |

第 4 章 不被测试的特性

| 特性或组合名称 | 测试设计说明编号 |
|---|---|
| | |

第 5 章 方法

5.1 <方法名称>

5.2 <方法名称>

第 6 章 项通过准则

第 7 章 暂停标准和再启动要求

7.1 暂停标准

7.2 再启动要求

第 8 章 应提供的测试文档

| 文档名称 | 标识符 |
|---|---|
| | |

第 9 章 测试任务

| 序号 | 任务 | 前期任务 | 特殊技能 | 责任人 | 工作量（天） | 完成日期 |
|---|---|---|---|---|---|---|
| | | | | | | |

第 10 章 环境要求

10.1 硬件

10.2 软件

10.3 安全性

10.4 工具

10.5 文档

第 11 章 职责

11.1 测试组

11.2 开发组

11.3 ……

第 12 章 人员和培训要求

12.1 人员

12.1.1 测试组

12.2 培训

第 13 章 进度

13.1 进度

| 序号 | 测试任务名称 | 工作量 | 开始日期 | 完成日期 |
|---|---|---|---|---|
| | | | | |

13.2 测试资源使用期限

第 14 章 风险和应急

## 测试日志

编者说明：

测试都有一个结果，而这些结果对于软件质量保证活动来说是十分重要的，因此应该将这些结果有序地记录下来，这就是测试日志模板所要解决的问题。

第 1 章　描述

1.1　测试项

| 序号 | 测试项名称 | 标识符 | 版本 | 相关传递报告 |
|---|---|---|---|---|
| | | | | |

1.2　测试的环境

1.2.1　　硬件

1.2.2　　软件

第 2 章　活动和事件条目

2.1　<日期>

| 时间 | 活动描述 | 事件 |
|---|---|---|
| | | |

2.2　<日期>

## 测试设计说明

编者说明：

如果说测试计划是对测试的活动、人员进行安排，那么测试设计则是对测试方法、测试技术的说明。

第 1 章　被测试的特性

1.1　单项特性

1.2　组合特性

1.3　引用文档

第 2 章　方法详述

2.1　方法描述

2.2　测试评价标准

2.3　测试用例选择原则

2.4　测试用例的共同属性和依赖关系

## 测试用例说明

编者说明：

测试计划解决的是怎么安排测试活动，测试设计说明是怎么测试，那么测试用例说明就是测试什么，也就是列出具体的测试项目，以使得测试有目的、有计划。

第 1 章　测试项

1.1　测试项名称

| 测试项名称 | 标识符 | 说明 |
|---|---|---|
| | | |

1.2　引用文档

| 编号 | 文档名称 | 章节名 |
|---|---|---|
| | | |

第 2 章　输入说明

| 序号 | 名称 | 值 | 类型 | 允许误差 | 输入方式 |
|---|---|---|---|---|---|
| | | | | | |

第 3 章　输出说明

| 序号 | 名称 | 值 | 类型 | 允许误差 | 输出方式 |
|---|---|---|---|---|---|
| | | | | | |

第 4 章　环境要求

4.1　硬件

4.2　软件

4.3　其它

第 5 章　特殊的规程要求

第 6 章　用例间的依赖关系

6.1　所依赖的用例

| 序号 | 用例名称或标识 |
|---|---|
| | |

6.2　依赖关系的性质

## 集成测试计划(ISO 标准)

编者说明：

前面的测试计划模板是一个通用性的，也可以是用于制定所有测试活动的计划，而本模块则是用来指导编写集成测试计划的。

1.引言

1.1 编写目的

[说明编写这份测试计划目的，指出预期的读者。]

1.2 背景

a. 待开发系统的名称；

b. 列出本项目的任务提出者、开发者、用户。

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出有关的参考资料。]

2．计划

2.1 系统说明

[提供一份图表，并逐项说明被测系统的功能、输入、输出等质量指标，作为叙述测试计划的提纲。]

2.2 测试内容

[列出集成测试和确认测试中的每一项测试内容的名称标识符、这些测试的进度安排以及这些测试的内容和目的。]

2.3 测试 1（标识符）

[给出这项测试内容的参与单位及被测试的部位。]

2.3.1 进度安排

[给出对这项测试的进度安排，包括进行测试的日期和工作内容。]

2.3.2 条件

[陈述本项测试工作对资源的要求。包括：]

a. 硬件

b. 软件

c. 人员

2.3.3 测试资料

[列出本项测试所需的资料。]

2.3.4 测试培训

[说明或引用资料说明为被测系统的使用提供培训的计划。规定培训的内容、受训的人员及从事培训的工作人员。]

2.4 测试 2（标识符）

[用与本测试计划 2。3 条相类似的方式说明用于另一项及其后各项测试内容的测试工作计划。]

[……]

3．测试设计说明

3.1 测试 1（标识符）

[说明对第一项测试内容的测试设计考虑。]

3.1.1 控制

[说明本测试的控制方式。]

3.1.2 输入

[说明本项测试中所使用的输入数据及选择这些输入数据的策略。]

3.1.3 输出

[说明预期的输出数据。]

3.1.4 过程

[说明完成此项测试的一个个步骤和控制命令。]

3.2 测试 2（标识符）

[用与本测试计划3。1 条相类似的方式说明第 2 项及其后各项测试工作的设计考虑。]

[……]

4．评价准则

4.1 范围

[说明所选择的测试用例能够检查的范围及其局限性。]

4.2 数据整理

[陈述为了把测试数据加工成便于评价的适当形式，使得测试结果可以同已知结果进行比较而要用到的转换处理技术；如果是用自动方式整理数据，还要说明为进行处理而要用到的硬件、软件资源。]

4.3 尺度

[说明用来判断测试工作是否能通过的评价尺度，如合理和输出结果的类型、测试输出结果与预期输出之间的容许偏离范围、允许中断或停机的最大数。]

## 软件集成测试工作流程指南

编者说明：

严格地说，该文档不属于文档模板，它只是一个工作指南。要想更好地完成集成测试工

作，你就需要为团队制定一个工作指南。你可以根据该文档，结合实际进行修改。

1. 简介

1.1 目的

本文详细阐述了集成测试流程，指导项目开发人员如何开展软件集成测试。

1.2 范围

此指南可运用于使用RUP 的任一软件项目的集成测试。

1.3 参考文件

Software Test Process

Rational Unified Process

1.4 定义与缩写

RUP：统一开发过程

SIT：软件集成测试

SEPG：软件工程过程小组

SQA：软件质量保证

2. 集成测试指南

2.1 简介

集成测试的目的是确保各单元组合在一起后能够按既定意图协作运行，并确保增量的行为正确。它所测试的内容包括单元间的接口以及集成后的功能。使用黑盒测试方法测试集成的功能。并且对以前的集成进行回归测试。

2.2 单元测试工作内容及其流程

| 活动 | 输入工件 | 输出工件 | 参与角色和职责 |
|---|---|---|---|
| 制定集成测试计划 | 设计模型<br>集成构建计划 | 集成测试计划 | 测试设计员负责制定集成测试计划 |
| 设计集成测试 | 集成测试计划<br>设计模型 | 集成测试用例<br>测试过程 | 测试设计员负责设计集成测试用例和测试过程。 |
| 实施集成测试 | 集成测试用例<br>测试过程<br>工作版本 | 测试脚本（可选）<br>测试过程（更新） | 测试设计员负责编制测试脚本（可选），更新测试过程。 |
| | | 驱动程序或稳定桩 | 设计员负责设计驱动程序和桩，实施员负责实施驱动程序和桩。 |
| 执行集成测试 | 测试脚本（可选）<br>工作版本 | 测试结果 | 测试员负责执行测试并记录测试结果 |
| 评估集成测试 | 集成测试计划<br>测试结果 | 测试评估摘要 | 测试设计员负责会同集成员、编码员、设计员等有关人员（具体化）评估此次测试，并生成测试评估摘要。 |

2.3 集成测试需求获取

集成测试需求所确定的是对某一集成工作版本的测试的内容，即测试的具体对象。集成测试需求主要来源于设计模型（Design Model ）和集成构件计划（Integration Build Plan ）。

集成测试着重于集成版本的外部接口的行为。因此，测试需求须具有可观测、可测评性。

1.集成工作版本应分析其类协作与消息序列，从而找出该工作版本的外部接口。

2.由集成工作版本的外部接口确定集成测试用例。

3.测试用例应覆盖工作版本每一外部接口的所有消息流序列。

注意：一个外部接口和测试用例的关系是多对多，部分集成工作版本的测试需求可映射到系统测试需求，因此对这些集成测试用例可采用重用系统测试用例技术。

2.4 集成测试工作机制

软件集成测试工作由产品评测部担任。需要项目组相关角色配合完成。如图示：

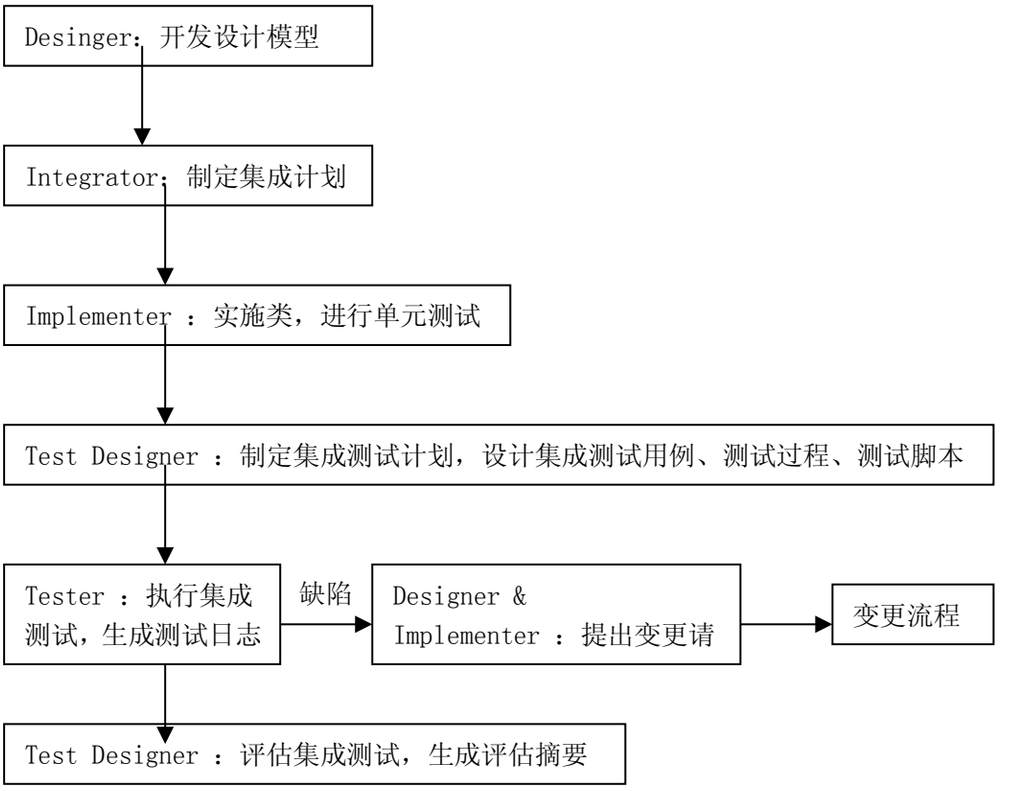
软件评测部：

| 角色 | 职责 |
|---|---|
| 测试设计员 | 负责制定集成测试计划、设计集成测试、实施集成测试、评估集成测试。 |
| 测试员 | 执行集成测试，记录测试结果。 |

软件项目组：

| 角色 | 职责 |
|---|---|
| 实施员 | 负责实施类（包括驱动程序和桩），并对其进行单元测试。根据集成测试发现的缺陷提出变更申请。 |
| 配置管理员 | 负责对测试工件进行配置管理。 |
| 设计员 | 负责设计测试驱动程序和桩。根据集成测试发现的缺陷提出变更申请。 |

集成测试工作内容及其流程工作流程：




2.5 集成测试产生的工件清单

1、软件集成测试计划

2、集成测试用例

3、测试过程

4、测试脚本
5、测试日志
6、测试评估摘要

# 软件系统测试工作指南

编者说明：
这是一个系统测试的工作指南。你可以根据该文档，结合实际进行修改。

1. 简介

1.1 目的

本文详细阐述了系统测试的类型以及各个类型的基本测试方法，指导项目开发人员进行软件系统测试。

1.2 范围

本文适用于使用 RUP 的所有软件项目的系统测试工作。

1.3 文档结构

第一部分：简介，介绍软件系统测试指南的目的，本指南的适用范围，以及在本文档中使用的术语的解释。

第二部分：描述系统测试指南。包括系统测试流程、系统测试需求的获取、系统测试侧策略选择、系统测试技术和方法等。

第三部分：列出本指南使用的参考文献。

1.4 词汇表

系统测试（System Testing）：系统测试是通过与系统的需求规格作比较，发现软件与系统需求规格不相符合或与之矛盾的地方。它将通过确认测试的软件，作为整个基于计算机系统的一个元素，与计算机硬件、外设、某些支持软件、数据和人员等其他系统元素结合起来，在实际运行（使用）环境下，对计算机系统进行的测试。

黑盒测试（Black-Box Testing）：黑盒测试是基于系统需求规格，在不知道系统或组件的内部结构的情况下进行的测试。通常又将黑盒测试叫做：基于规格的测试（Specification-Based Testing ）、输入输出测试（Input/Output Testing ）、功能测试（Functional Testing ）。

2. 系统测试指南

2.1 系统测试过程

| 活动名称 | 输入工件 | 输出工件 | 参与角色 |
|---|---|---|---|
| 制定系统测试计划 | 软件需求工件<br>软件项目计划 | 系统测试计划 | 测试设计员 |
| 设计系统测试 | 系统测试计划<br>软件需求工件 | 系统测试用例<br>系统测试过程 | 测试设计员 |
| 实施系统测试 | 系统测试计划<br>工作版本 | 系统测试脚本 | 测试设计员 |
| 执行系统测试 | 系统测试计划<br>系统测试用例<br>系统测试过程<br>系统测试脚本 | 测试结果 | 测试员 |

| | | | |
|---|---|---|---|
| 评估系统测试 | 测试结果 | 测试分析报告<br>变更请求 | 测试设计员<br>相关组 |

2.2 系统测试需求获取

系统测试需求所确定的是测试的内容，即测试的具体对象。系统测试需求主要来源于需求工件集，它可能是一个需求规格说明书，或是由前景、用例、用例模型、词汇表、补充规约组成的一个集合。

在分析测试需求时，可应用以下几条一般规则：

1） 测试需求必须是可观测、可测评的行为。如果不能观测或测评的测试需求，就无法对其进行评估，以确定需求是否已经满足。

2） 在每个用例或系统的补充需求与测试需求之间不存在一对一的关系。用例通常具有多个测试需求；有些补充需求将派生一个或多个测试需求，而其他补充需求（如市场需求或包装需求）将不派生任何测试需求。

3） 在需求规格说明书中每一个功能描述将派生一个或多个测试需求，性能描述、安全性描述等也将派生出一个或多个测试需求。

1. 功能性测试需求

功能性测试需求来自于测试对象的功能性说明。每个用例至少会派生一个测试需求。对于每个用例事件流，测试需求的详细列表至少会包括一个测试需求。对于需求规格说明书中的功能描述，将至少派生一个测试需求。

2. 性能测试需求

性能测试需求来自于测试对象的指定性能行为。性能通常被描述为对响应时间和资源使用率的某种评测。性能需要在各种条件下进行评测，这些条件包括：

1）不同的工作量和/或系统条件

2）不同的用例/功能

3）不同的配置

4）性能需求在补充规格或需求规格说明书中的性能描述部分中说明。

对包括以下内容的语句要特别注意：

1）时间语句，如响应时间或定时情况

2）指出在规定时间内必须出现的事件数或用例数的语句

3）将某一项性能的行为与另一项性能的行为进行比较的语句

4)将某一配置下的应用程序行为与另一配置下的应用程序行为进行比较的语句

5）一段时间内的操作可靠性（平均故障时间或 MTTF ）

6）配置或约束

应该为规格中反映以上信息的每个语句生成至少一个测试需求。

3. 其它测试需求

其它测试需求包括配置测试、安全性测试、容量测试、强度测试、故障恢复测试、负载测试等测试需求可以从非功能性需求中发现与其对应的描述。每一个描述信息可以生成至少一个测试需求。

2.3 系统测试策略

测试策略用于说明某项特定测试工作的一般方法和目标。系统测试策略主要针对系统测试需求确定测试类型及如何实施测试的方法和技术。

一个好的测试策略应该包括要实施的测试类型和测试的目标、所采用的技术、用于评估测试结果和测试是否完成的标准、对测试策略所述的测试工作存在影响的特殊事项等内容。

2.3.1 系统测试类型和目标

确定系统测试策略首先应清楚地说明所实施系统测试的类型和测试的目标。清楚地说明这些信息有助于尽量避免混淆和误解（尤其是由于有些类型测试看起来非常类似，如强度测试和容量测试）。测试目标应该表明执行测试的原因。

系统测试的测试类型一般包括：功能测试（Functional Testing）、性能测试（Performance Testing）负载测试（Load Testing）、强度测试（Stress Testing）、容量测试（Volume Testing）、安全性测试（Security Testing）、配置测试（Configuration Testing）、故障恢复测试（Recovery Testing）、安装测试（Installation Testing）、文档测试（Documentation Testing）、用户界面测试（GUI Testing）等等。

其中，功能测试、配置测试、安装测试等在一般情况下是必需的。而其它的测试类型则需要根据软件项目的具体要求进行裁剪。

2.3.2 采用的测试技术

系统测试主要采用黑盒测试技术设计测试用例来确认软件满足需求规格说明书的要求。

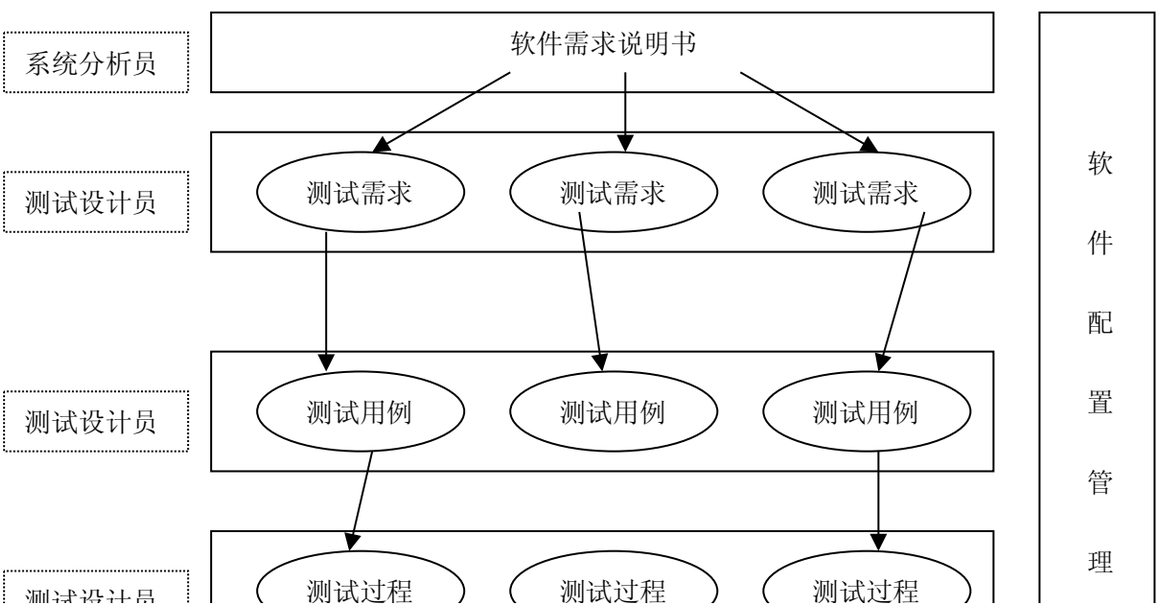
2.4 系统测试的工作机制

1）项目组为每一个软件项目成立测试组，确定测试经理（通常由测试设计员担任）一名，测试设计员和测试员若干。

| 角色 | 职责 |
|---|---|
| 测试设计员 | 制定系统测试计划、设计系统测试、实施系统测试以及评估系统测试 |
| 测试员 | 执行系统测试 |

2）项目组需要提供系统测试需要的输入，建立测试环境，以及对测试工件进行配置管理。

| 角色 | 职责 |
|---|---|
| 系统分析员 | 生成需求工件集，管理需求。为测试设计员提供测试需求。 |
| 配置管理员 | 对测试工件进行配置管理 |



2.5 系统测试产生的工件清单

1）软件系统测试计划

2）系统测试用例

3）系统测试过程

4）测试脚本（可选）

5）测试结果

6）测试分析报告

## 测试分析报告(GB 标准)

编者说明：

测试完成后，将会形成一些测试日志，对于每个测试用例也有了一个反馈的结果，那么从这个数据中看出问题、找到问题以及寻找解决问题的方法，那就是测试分析报告所要完成的事了。

1.引言

1.1 编写目的

[说明这份测试分析报告的具体编写目的，指出预期的阅读范围。]

1.2 背景

[说明：]

[ a. 被测试软件系统的名称；]

[ b. 该软件的任务提出者、开发者、用户及安装此软件的计算中心，指出测试环境与实际运行环境之间可能存在的差异以及这些差异对测试结果的影响。]

1.3 定义

[列出本文件中用到的专问术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出要用到的参考资料，如：]

[ a. 本项目的经核准的计划任务书或合同、上级机关的批文；]

[ b. 属于本项目的其他已发表的文件；]

[ c. 本文件中各处引用的文件、资料，包括所要用到的软件开发标准。]

[列出这些文件的标题、文件编号、发表日期和出版单位，说明能够得到这些文件资

料的来源。]

2.测试概要

[用表格的形式列出每一项测试的标识符及其测试内容，并指明实际进行的测试工作内容与测试计划中预先设计的内容之间的差别，说明作出这种改变的原因。]

3.测试结果及发现

3.1 测试 1（标识符）

[把本项测试中实际得到的动态输出（包括内部生成数据输出）结果同对于动态输出的要求进行比较，陈述其中的各项发现。]

3.2 测试 2（标识符）

[用类似本报告 3.1 条的方式给出第 2项及其后各项测试内容的测试结果和发现。]

4.对软件功能的结论

4.1 功能 1（标识符）

4.1.1 能力

[简述该项功能，说明为满足此项功能而设计的软件能力以及经过一项或多项测试已证实的能力。]

4.1.2 限制

[说明测试数据值的范围（包括动态数据和静态数据），列出就这项功能而言，测试期间在该软件中查出的缺陷、局限性。]

4.2 功能 2（标识符）

[用类似本报告 4.1 的方式给出第 2 项及其后各项功能的测试结论。]

.....

5 分析摘要

5.1 能力

[陈述经测试证实了的本软件的能力。如果所进行的测试是为了验证一项或几项特定性能要求的实现，应提供这方面的测试结果与要求之间的比较，并确定测试环境与实际运行环境之间可能存在的差异对能力的测试所带来的影响。]

5.2 缺陷和限制

[陈述经测试证实的软件缺陷和限制，说明每项缺陷和限制对软件性能的影响，并说明全部测得的性能缺陷的累积影响和总影响。]

5.3 建议

[对每项缺陷提出改进建议，如：]

[ a. 各项修改可采用的修改方法；]

[ b. 各项修改的紧迫程度；]

[ c. 各项修改预计的工作量；]

[ d. 各项修改的负责人。]

5.4 评价

[说明该项软件的开发是否已达到预定目标，能否交付使用。]

6 测试资源消耗

[总结测试工作的资源消耗数据，如工作人员的水平级别数量、机时消耗等。]

## 测试规程说明

编者说明：

软件测试就像生产线上的产品测试一样，需要专业的技能与工作方法，而测试规程则是

确保每次测试动作高度统一。

第 1 章　目的

1.1　一般目的

1.2　执行的测试用例

| 序号 | 测试用例名称或标识符 |
| --- | --- |
| | |

第 2 章　特殊要求

2.1　前继规程

| 序号 | 前继规程的名称或标识符 |
| --- | --- |
| | |

2.2　专门技能

2.3　特殊环境

2.4　其它

第 3 章　规程步骤

3.1　日志

3.2　准备

3.3　启动

3.4　处理

3.5　度量

3.6　暂停

3.7　再启动

3.8　停止

3.9　清除

3.10 应急

## 计算机软件测试文件编制规范

编者说明：

　　测试是一个复杂、系统化的工作，也是一个内容广泛的课题，其间将产生大量的文档。本文档就是一个指导所有这些文档编写的规范。你可以根据自己的实际，对其修改，以适用于你的开发团队。

### 1. 引言

#### 1.1 目的和作用

本规范规定一组软件测试文件。测试是软件生存周期中一个独立的、关键的阶段，也是保证软件质量的重要手段。为了提高检测出错误的几率，使测试能有计划地、有条不紊地进行地进行，就必须要编制测试文件。而标准化的测试文件就如同一种通用的参照体系，可达到便于交流的目的。文件中所规定的内容可以作为对测试过程完备性的对照检查表，故采用这些文件将会提高测试过程的每个阶段的能见度，极大地提高测试工作的可管理性。

#### 1.2 适用对象及范围

本规范是为软件管理人员、软件开发人员和软件维护人员、软件质量保证人员、审计人员、客户及用户制定的。

本规范用于描述一组测试文件，这些测试文件描述测试行为。本规范定义每一种基本文件的目的、格式和内容。所描述的文件着重于动态测试过程，但有些文件仍适用其它种类的测试活动。

本规范可应用于数字计算机上运行的软件。它的应用范围不受软件大小、复杂度或重要性的限制，本规范既适用于初始开发的软件测试文件编制，也适用于其后的软件产品更新版本的测试文件编制。

本规范并不要求采用特定的测试方法学、技术及设备或工具。对文件控制、配置管理或质量保证既不指明也不强制特定的方法学。根据所用的方法学，可能需要增加别的文件（如“质量保证计划”）。

本规范既适用于纸张上的文件，也适用于其它媒体上的文件。如果电子文件编制系统不具有安全的批准注册机制，则批准签字的文件必须使用纸张。

## 2.引用标准

GB/T 11457 软件工程术语

GB 8566 计算机软件开发规范

GB 8567 计算机软件产品开发文件编制指南

## 3.定义

本章定义本规范中使用的关键术语。

### 3.1 设计层 design level

软件项的设计分解（如系统、子系统、程序或模块）。

### 3.2 通过准则 pass criteria

判断一个软件项或软件特性的测试是否通过的判别依据。

### 3.3 软件特性 software feature

软件项的显著特性。(如功能、性能或可移植性等)。

### 3.4 软件项 software item

源代码、目标代码、作业控制代码、控制数据或这些项的集合。

### 3.5 测试项 test item

作为测试对象的软件项。

## 4.概述

### 4.1 主要内容

本规范确定了各个测试文件的格式和内容，所提出的文件类型包括测试计划、测试说明和测试报告。

测试计划描述测试活动的范围、方法、资源和进度。它规定被测试的项、被测试的特性、应完成的测试任务、担任各项工作的人员职责及与本计划有关的风险等。

测试说明包括三类文件：

（1）测试设计说明：详细描述测试方法，规定该设计及其有关测试所包括的特性，还规定完成测试所需的测试用例和测试规程，并规定特性的通过准则。

（2）测试用例说明：列出用于输入的具体值以及预期的输出结果，并规定在使用具体测试用例时，对测试规程的各种限制。将测试用例与测试设计分开，可以使它们用于多个设计并能在其它情形下重复使用。

（3）测试规程说明：规定对于运行系统和执行指定的测试用例来实现有关测试设计所要求的所有步骤。

测试报告包括四类文件：

（1）测试项传递报告：指明在开发组和测试组独立工作的情况下或者在希望正式开始测试的情况下为进行测试而被传递的测试项。

（2）测试日志：测试组用于记录测试执行过程中发生的情况。

（3）测试事件报告：描述在测试执行期间发生并需进一步调查的一切事件。

（4）测试 7 总结报告：总结与测试设计说明有关的测试活动。

这些文件同其它文件在编制方面的关系以及同测试过程的对应关系如图 1 所示。

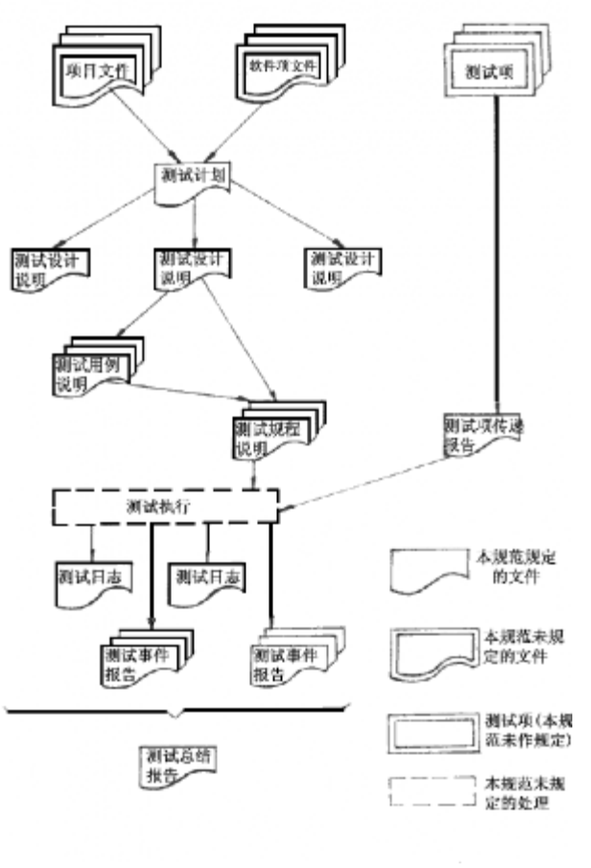
**4.2 实施灵活性**

在 GB 8567 中，涉及软件测试的文件有“测试计划”及“测试分析报告”。本规范中的八个测试文件是上述二个文件的补充和细化，这样可使文件的书定更具体、更有参照性，其中测试计划可细化为本规范的测试计划、测试设计说明、测试用例说明及测试规程说明，测试分析报告可细化为本规范的测试项传递报告、测试日志、测试事件报告及测试总结报告。

使用本规范的每个单位，要规定测试阶段所应有的特定文件，并在测试计划中规定测试完成后所能提交的全部文件。对于不同的设计层或不同规模的软件，所选文件的种类也可有所不同。

在所提供的每个标准文件中，每一章的内容对于具体的应用和特定的测试阶段可以有所增减。不仅可以调整内容，还可以在基本文件集中增加另外的文件。任何一个文件都可以增加新的内容，并且某章若无可写的内容，则可不写，但须保留该章的编号。使用本规范的每个单位应该补充规定对内容的要求和约定，以便反映自己在测试、文件控制、配置管理和质量保证方面所用的特定方法、设备和工具。

附录 A（参考件）中，将叙述文件编制实施及使用指南。




**4.3 总体要求**

以下将叙述各个测试文件的书写格式及内容。对于每一个文件而言各章应按指定的次序排列，补充的章可以放在最后或放在“批准”一章的前面（如果该文件最后一章是“批准”的话）。如果某章的部分或全部内容在另一文件中，则应在相应的内容位置上列出所引用的材料，引用的材料必须附在该文件后面或交给文件的使用者。

**5.内容要求**

**5.1 测试计划**

测试计划结构如下图所示。

1 测试计划名称
2 引言
3 测试项
4 被测试的特性
5 不被测试的特性
6 方法
7 项通过准则
8 暂停标准和再启动要求
9 应提供的测试文件
10 测试任务
11 环境要求
12 职责
13 人员和训练要求
14 进度
15 风险和应急
16 批准

下面给出每一章的详细内容：

**5.1.1 测试计划名称（本计划的第 1 章）**

为本测试计划取现代战争专用的名称。

**5.1.2 引言（本计划的第 2 章）**

归纳所要求测试的软件项和软件特性，可以包括系统目标、背景、范围及引用材料等。

在最高层测试计划中，如果存在下述文件，则需要引用它们：项目计划、质量保证计划、有关的政策、有关的标准等。

**5.1.3 测试项（本计划的第 3 章）**

描述被测试的对象，包括其版本、修订级别，并指出在测试开始之前对逻辑或物理变换的要求。

**5.1.4 被测试的特性（本计划的第 4 章）**

指明所有要被测试的软件特性及其组合，指明每个特性或特性组合有关的测试设计说明。

**5.1.5 不被测试的特性（本计划的第 5 章）**

指出不被测试的所有特性和特性的有意义的组合及其理由。

**5.1.6 方法（本计划的第 6 章）**

描述测试的总体方法，规定测试指定特性组志需的主要活动、、技术和工具，应详尽地描述方法，以便列出主要的测试任务，并估计执行各项任务所需的时间。规定所希望的电低程度的测试彻底性，指明用于判断测试彻底性的技术（如：检查哪些语句至少执行过一次）。指出对测试的主要限制，例如：测试项可用性、测试资源的可用性和测试截止期限等。

**5.1.7 项通过准则（本计划的第 7 章）**

规定各测试项通过测试的标准。

**5.1.8 暂停标准和再启动要求（本计划第 8 章）**

规定用于暂停全部或部分与本计划有关的测试项的测试活动的标准。规定当测试再启动时必须重复的测试活动。

**5.1.9 应提供的测试文件（本计划的第 9 章）**

规定测试完成后所应递交的文件，这些文件可以是前述八个文件的全部或者部分。

**5.1.10 测试任务（本计划的第 10 章）**

指明执行测试所需的任务集合，指出任务音的一切依赖关系和所需的一切特殊技能。

**5.1.11 环境要求（本计划的第 11 章）**

规定测试环境所必备的和希望的的性质。包括：硬件、通信和系统软件的物理特征、使用方式以及任何其它支撑测试所需的软件或设备，指出所需的特殊测试工具及其它测试要求（如出版物或办公场地等）。指出测试组目前还不能得到的所有要求的来源。

**5.1.12 职责（本计划的第 12 章）**

指出负责管理、设计、准备、执行、监督、检查和仲裁的小组。另外指出负责提供

5.1.3 中指出的测试项和在 5.1.11 中指出的环境要求的小组。

这些小组可以包括开发人员、测试人员、操作员、用户代表、数据管理员和质量保证人员。

**5.1.13 人员和训练要求（本计划的第 13 章）**

指明测试人员应有的水平以及为掌握必要技能可供选择的训练科目。

**5.1.14 进度（本计划的第 14 章）**

包括在软件项目进度中规定的测试里程碑以及所有测试项传递时间。

定义所需的新的测试里程碑，估计完成每项测试任务所需的时间，为每项测试任务和测试里程碑规定进度，对每项测试资源规定使用期限。

**5.1.15 风险和应急（本计划的第 15 章）**

预测测试计划中的风险，规定对各种风险的应急措施（如：延期传递的测试项可能需要加夜班来赶上规定的进度。）

**5.1.16 批准（本计划的第 16 章）**

规定本计划必须由哪些人（姓名和职务）审批。为签名和填写日期留出位置。

## 5.2 测试设计说明

测试设计说明如下图所示。

| |
|---|
| 1 测试设计说明名称 |
| 2 被测试的特性 |
| 3 方法详述 |
| 4 测试用例名称 |
| 5 特性通过准则 |

下面给出本说明每一章的详细内容。

**5.2.1 测试设计说明名称（本说明第 1 章）**

给每一个测试设计说明取一个专用名称。如果存在的话，也可引用有关的测试计划中给出的名称。

**5.2.2 被测试的特性（本说明的第 2 章）**

规定测试项，描述作为本设计测试目标的特性和特性的组合，其它特性可以论及，但不必测试。

**5.2.3 方法详述（本说明的第 3 章）**

将测试计划中规定的方法进行细化，包括要用的具体测试技术，规定分析测试结果的方法（如比较程序或人工观察）。

规定为选择测试用例提供合理依据的一切分析结果。例如：可以说明容错的条例（如：区别有效输入和无效输入的条件）。

归纳所有测试用例的共同属性，可以包括输入约束条件，共享环境的要求，对共享的特殊规程的要求及任何共享的测试用例间的依赖关系。

**5.2.4 测试例名称（本说明的第 4 章）**

列出与本设计有关的每一测试用例的名称和简要说明。某个特定的测试用例可能在多个测试设计说明中出现，列出与本测试设计说明有关的规程及其简要说明。

**5.2.5 特性通过准则（本说明的第 5 章）**

规定用于判别特性和特性组合是否通过测试的准。

**5.3 测试用例说明**

测试用例说明结构如下图所示。

| |
|---|
| 1 测试用例说明名称 |
| 2 测试项 |
| 3 输入说明 |
| 4 输出说明 |
| 5 环境要求 |
| 6 特殊的规程说明 |
| 7 用例间的依赖关系 |

由于测试用例可能被由多个小组长期使用的多个测试设计说明引用，所以在测试用例说明中必须包含足够具体的信息以便重复使用。

下面给出本说明每一章的详细内容。

**5.3.1 测试用例说明名称（本说明的第 1 章）**

给本测试用例说明取一个专用名称

**5.3.2 测试项（本说明的第 2 章）**

规定并简要说明本测试用例所要涉及的项和特性、对于每一项、可考虑引用以下文件：需求说明书、设计说明书、用户手册、操作手册。

**5.3.3 输入说明（本说明的第 3 章）**

规定执行测试用例所需的各个输入。有些输入可以用值（允许适当的误差）来规定。而另一些输入，如常数表或事务文件可以用名来规定。规定所有合适的数据库、文件、终端信息、内存常驻区域和由操作系统传送的值。规定各输入间所需的所有关系（如时序关系等）。

**5.3.4 输出说明（本说明的第 4 章）**

规定测试项的所有输出和特性（如：响应时间）。提供各个输出或特性的正确值（在适当的误差范围内）。

**5.3.5 环境要求（本说明的第 5 章）**

**5.3.5.1 硬件**

规定执行本测试用例所需的硬件特征和配置（如：80 字符×24 行的显示终端）。

**5.3.5.2 软件**

规定执行本测试用例所需的系统软件和应用软件。系统软件可以包括操作系统、编译程序、模拟程序和测试工具等。

**5.3.5.3 其它**

说明所有其它的要求，如特种设施要求或经过专门训练的人员等。

**5.3.6 特殊的规程要求（本说明的第 6 章）**

描述对执行本测试用例的测试规程的一切特殊限制。这些限制可以包括特定的准备、操作人员干预、确定特殊的输出和清除过程。

**5.3.7 用例间的依赖关系（本说明的第 7 章）**

列出必须在本测试用例之前执行的测试用例名称，归纳依赖性质。

**5.4 测试规程说明**

测试规程说明结构如下图表示


下面给出本说明每一章的详细内容。

**5.4.1 测试规程说明名称（本说明的第 1 章）**

给每个测试规程说明取一个专用名称，给出对有关测试设计说明的引用。

**5.4.2 目的（本说明的第 2 章）**

描述本规程的目的。如果本规程执行测试用例，则引用各有关的测试用例说明。

**5.4.3 特殊要求（本说明的第 3 章）**

指出执行本规程所需的所有特殊要求，包括作为先决条件的规程、专门技能要求和特殊环境要求。

**5.4.4 规程步骤（本说明的第 4 章）**

**5.4.4.1 日志**

说明用来记录测试的执行结果、观察到的事件和其它与测试有关事件（见 5.6 条测试日志和 5.7 条测试事件报告）的所有特殊方法或格式。

**5.4.4.2 准备**

描述新任务执行规程所必需的动作序列。

**5.4.4.3 启动**

描述开始执行规程所必需的动作。

**5.4.4.4 处理**

描述在规程执行过程中所必需的动作。

**5.4.4.5 度量**

描述如何进行测试度量（如描述如何用网络模拟程序来充其量远程终端的响应时间）。

**5.4.4.6 暂停**

描述因发生意外事件暂停测试所必需的动作。

**5.4.4.7 再启动**

规定所有再拨动点和在启动点上重新启动规程所必需的动作。

**5.4.4.8 停止**

描述正常停止执行时所必需的动作。

**5.4.4.9 清除**

描述恢复环境所必需的动作。

**5.4.4.10 应急**

描述处理执行过程中可能发生的异常事件所必需的动作。

## 5.5 测试项传递报告

测试项传递报告结构如下图所示。


下面给出本报告每一章的详细内容。

**5.5.1 传递报告名称（本报告的第 1 章）**

为本测试项传递报告取一个专用名称。

**5.5.2 传递项（本报告的第 2 章）**

规定被传递的项及其版本/修订级别。提供与传递项有关的项文件和测试计划的相关信息，指出对该传递项负责的人员。

**5.5.3 位置（本报告的第 3 章）**

规定传递项的位置及其所在媒体。

**5.5.4 状态（本报告的第 4 章）**

描述被传递的测试项的状态，包括其与项文件、这些项的以往传递以及测试计划的差别。列出希望由被传递项解决的事件报告。

**5.5.5 批准（本报告的第 5 章）**

规定本传递报告必须由哪些人（姓名和职务）审批，并为签名和日期留出位置。

**5.6 测试日志**

测试日志结构如下图所示。

| |
|---|
| 1 测试日志名称 |
| 2 描述 |
| 3 活动和事件条目 |

下面给出本报告每一章的详细内容。

**5.6.1 测试日志名称（本日志的第 1 章）**

为本测试日志取一专用的名称。

**5.6.2 描述（本日志的第 2 章）**

除了在日志条目中特别注明的以外，用于日志中所有条目的信息都包括在本章中。应该考虑有以下信息：

（1）规定被测试项及其版本/修订级别。如果存在的话，引用各项的传递报告。

（2）规定完成测试的环境属性，包括设备说明、所用的硬件、所用的系统软件及可用存储容量等可用资源。

**5.6.3 活动和事件条目（本日志的第 3 章）**

对每个事件（包括事件的开始和结束），记录发生的日期和时间，并说明记录者。应考虑以下各项信息。

**5.6.3.1 执行描述**

记录所执行的测试规程的名称，并引用该测试规程说明。记录执行时在场人员，包括：测试者、操作员和观察员，还要说明每个人的作用。

**5.6.3.2 测试结果**

对每次执行，记录人工可观察到的结果（如：产生的错误信息、异常中止和对操作员动作的请求等），还要记录所有输出的位置（如磁带号码），记录测试的执行是否成功。

**5.6.3.3 环境信息**

记录本条目的的一切特殊的环境条件。

**5.6.3.4 意外事件**

记录意外事件及其发生前后的情况（如请求显示总计，屏幕显示正常，但响应时间似乎异常长，重复执行时响应时间也同样过长）。记录无法开始执行测试或无法结束测试的周围环境（如电源故障或系统软件问题）。

**5.6.3.5 事件报告名称**

每产生一个测试事件报告时，记录其名称。

**5.7 测试事件报告**

测试事件报告结构如下图所示。

1 测试事件报告名称
2 摘要
3 事件描述
4 影响

下面给出本报告每一章的详细内容。

**5.7.1 测试事件报告名称（本报告的第 1 章）**

为本测试事件报告取一个专用名称。

**5.7.2 摘要（本报告的第 2 章）**

简述事件，指出有关测试项及其版本/修订级别。引用有关的测试规程说明、测试用例说明及测试日志。

**5.7.3 事件描述（本报告的第 3 章）**

对事件进行描述。该描述应包括以下各项：

输入
预期结果
实际结果
异常现象
日期和时间
规程步骤
环境
重复执行的意图
测试者
观察者

该描述应该包括有助于确定事件发生原因及改正其中错误的有关浩劫及观察。例如，描述可能对此事件有影响的所有测试用例执行情况，描述与已公布的测试规程之间的一切差异等。

**5.7.4 影响（本报告的第 4 章）**

在所知道的范围内指出本事件对测试计划、测试设计说明、测试规程说明或测试用例说明所产生的影响。

**5.8 测试总结报告**

规定本报告必须由哪些人（姓名和职务）审批，并为签名和日期留出位置。

# 单元测试报告

编者说明：

单元测试是每一个开发人员都必须去做的事，它将采用白盒方法来进行，为了跟踪单元测试的效果，对开发人员进行督促，对于一些重要的模块进行测试是很必要的。该表格就是用来让开发人员填写单元测试的结果的文档。

填表日期：_____ 编号：_____

| 开发项目名称 | | 开发项目编号 | | 第一责任人 | |
|---|---|---|---|---|---|
| 单元名称 | | 责任人 | | 单元所属子系统 | | 开发周期 | |
| 代码测试检查： | | | | | |
| 代码测试内容 | 测试人员 | 测试结果 | | 备注 | |
| 路径测试 | | | | | |

| | | | |
|---|---|---|---|
| 声明测试 | | | |
| 循环测试 | | | |
| 边界测试 | | | |
| 接口测试 | | | |
| 界面测试 | | | |
| 数据确认测试 | | | |
| 代码走查 | | | |

功能测试：

| 序号 | 功能名称 | 操作方法 | 结果 | 建议 | 测试人员 | 备注 |
|---|---|---|---|---|---|---|
| | | | | | | |

| | | | |
|---|---|---|---|
| 测试结论 | | | |
| 责任人 | | 项目第一责任人 | |
| 审核 | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 项目组 | | 测试组 | | 总工办 | | 总工程师 | |

## 五、其它类模块开发说明(ISO 标准)

编程说明：

该文档将与模块开发卷宗结合使用，卷宗是对整个系统进行整理，而模块开发说明则是对具体的模块进行说明；其作用于归档阶段。

**删除的内容:**
**编码实现类**

1．标题

[系统名称和标识符]

[模块名称和标识符]

[程序编制员签名]

[卷宗的修改文本序号]

[修改完成日期]

[卷宗序号]

[编排日期]

2．模块开发情况表

3．功能说明

[扼要说明本模块的功能,主要是输入、要求的处理、输出。可以从系统设计说明书中摘录。同时列出在需求说明书中对这些功能的说明的章、条、款。]

4．设计说明

[说明本模块的设计考虑]

5．硬件部分的设计结果

1） 经项目组调试通过的硬件成品 1 件

2）设计文件：

《原理图》

《PCB 图》

《BOM 清单》

《可编程器件及烧录进制文件》

《必要测试点波形图或硬件指标评细说明》

《原理详细说明》

《与系统内其他部分接口软硬件详细说明》

这些文件可以附件的形式列后。

6．软件的设计结果

[要给出所产生的本模块的第一份无语法错的源代码清单以及已通过全部测试的当前有效的源程序代码。]

7．测试说明

[说明直接要经过本模块的每一项测试，包括这些测试各自的标识符和编号、进行这些测试的目的、所用的配置和输入、预期的输出及实际的输出。]

8．复审的结论

[把实际测试的结果，同需求说明书、系统设计说明书中规定的要求进行比较和给出结论。]

## 用户手册概要(GB 标准)

编者说明：

为用户提供一个使用手册，是提升软件可用性的必要措施。用户手册的作用是让用户对整个软件系统有一个宏观的认识。解决安装问题，告知运行环境，介绍主要功能等。

1 引言

1.1 编写目的

[说明编写这份用户手册的目的，指出预期的读者。]

1.2 背景

[说明：]

[a. 这份用户手册所描述的软件系统的名称；]

[b. 该软件项目的任务提出者、开发者、用户（或首批用户）及安装此软件的计算中心。]

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

删除的内容：五、其它类

1.4 参考资料

[列出有用的参考资料，如：]

a. 项目的经核准的计划任务书或合同、上级机关的批文；

b. 属于本项目的其他已发表文件；

c. 本文件中各处引用的文件、资料，包括所要用到的软件开发标准。

[列出这些文件资料的标题、文件编号、发表日期和出版单位，说明能够取得这些文件资料的来源。]

2 用途

2.1 功能

[结合本软件的开发目的逐项地说明本软件所具有各项功能以及它们的极限范围。]

2.2 性能

2.2.1 精度

[逐项说明对各项输入数据的精度要求和本软件输出数据达到的精度，包括传输中的精度要求。]

2.2.2 时间特性

[定量地说明本软件的时间特性，如响应时间，更新处理时间，数据传输、转换时间，计算时间等。]

2.2.3 灵活性

[说明本软件所具有的灵活性，即当用户需求（如对操作方式、运行环境、结果精度、时间特性等的要求）有某些变化时，本软件的适应能力。]

2.3 安全保密

[说明本软件在安全、保密方面的设计考虑和实际达到的能力。]

3 运行环境

3.1 硬设备

[列出为运行本软件所要求的硬设备的最小配置，如：]

a. 处理机的型号、内存容量;

b. 所要求的外存储器、媒体、记录格式、设备的型号和台数、联机／脱机；

c. I／O 设备（联机／脱机？）；

d. 数据传输设备和转换设备的型号、台数。

3.2 支持软件

[说明为运行本软件所需要的支持软件，如：]

a. 操作系统的名称、版本号；

b. 程序语言的编译／汇编系统的名称和版本号；

c. 数据库管理系统的名称和版本号；

d. 其他支持软件。

3.3 数据结构

[列出为支持本软件的运行所需要的数据库或数据文件。]

4 使用过程

[在本章，首先用图表的形式说明软件的功能同系统的输入源机构、输出接收机构之间的关系。]

4.1 安装与初始化

[一步一步地说明为使用本软件而需进行的安装与初始化过程，包括程序的存储形式、安装与初始化过程中的全部操作命令、系统对这些命令的反应与答复。表征安装工作完成的测试实例等。如果有的话，还应说明安装过程中所需用到的专用软件。]

4.2 输入

[规定输入数据和参量的准备要求。]

4.2.1 输入数据的现实背景

[说明输入数据的现实背景，主要是：]

a. 情况--例如人员变动、库存'缺货；

b. 情况出现的频度--例如是周期性的、随机的、一项操作状态的函数.

c. 情况来源-一例如人事部门、仓库管理部门；

d. 输入媒体---例如键盘、穿孔卡片、磁带；

e. 限制--出于安全、保密考虑而对访问这些输入数据所加的限制；

f. 质量管理--例如对输入数据合理性的检验以及当输入数据有错误时应采取的措施，如建立出错情况的记录等；

g. 支配--例如如何确定输入数据是保留还是废弃，是否要分配给其他的接受者等。

4.2.2 输入格式

[说明对初始输入数据和参量的格式要求，包括语法规则和有关约定，如：]

a. 长度-一例如字符数／行，字符数／项；

b. 格式基准--例如以左面的边沿为基准；

c. 标号--例如标记或标识符；

d. 顺序--例如各个数据项的次序及位置；

e. 标点--例如用来表示行、数据组等的开始或结束而使用的空格、斜线、星号、字符组等。

f. 词汇表--给出允许使用的字符组合的列表，禁止使用＊的字符组合的列表等；

g. 省略和重复--给出用来表示输人元素可省略或重复的表示方式；

h. 控制--给出用来表示输入开始或结束的控制信息。

4.2.3 输入举例

[为每个完整的输入形式提供样本，包括：]

a. 控制或首部--例如用来表示输入的种类和类型的信息，标识符输入日期，正文起点和对所用编码的规定；

b. 主体--输入数据的主体，包括数据文件的输入表述部分；

c. 尾部--用来表示输入结束的控制信息，累计字符总数等；

d. 省略--指出哪些输入数据是可省略的；

e. 重复--指出哪些输入数据是重复的。

4.3 输出

[对每项输出作出说明.]

4.3.1 输出数据的现实背景，

[说明输出数据的现实背景，主要是：]

a. 使用--这些输出数据是给谁的，用来干什么；

b. 使用频度--例如每周的、定期的或备查阅的；

c. 媒体--打印、CRI 显示、磁带、卡片、磁盘，

d. 质量管理-一例如关于合理性检验、出错纠正的规定；

e. 支配--例如如何确定输出数据是保留还是废弃，是否分配给其他接受者等。

4.3.2 输出格式

[给出对每一类输出信息的解释，主要是：]

a. 首部--如输出数据的标识符，输出日期和输出编号；

b. 主体--输出信息的主体，包括分栏标题；

c. 尾部--包括累计总数，结束标记。

4.3.3 输出举例

[为每种输出类型提供例子。对例子中的每一项，说明：]

a. 定义--每项输出信息的意义和用途；

b. 来源--是从特定的输入中抽出、从数据库文件中取出、或从软件的计算过程中得到；

c. 特性--输出的值域、计量单位、在什么情况下可缺省等。

4.4 文件查询

[这一条的编写针对具有查询能力的软件，内容包括：同数据库查询有关的初始化、准备、及处理所需 要的详细规定，说明查询的能力、方式，所使用的命令和所要求的控制规定。]

4.5 出错处理和恢复

[列出由软件产生的出错编码或条件以及应由用户承担的修改纠正工作。指出为了确保再启动和恢 复的能力，用户必须遵循的处理过程。]

4.6 终端操作

[当软件是在多终端系统上工作时，应编写本条，以说明终端的配置安排、连接步骤、数据和参数输入步骤以及控制规定说明。通过终端操作进行查询、检索、修改数据文件的能力、语言、过程以及辅助性程序等。]

## 操作手册 (GB 标准)

编者说明：

操作手册也是一个很常见的针对用户的文档，其帮助用户更好地操作软件，使用软件。

1 引言

1.1 编写目的

[说明编写这份操作手册的目的，指出预期的读者。]

1.2 前景

[说明： ]

a.这份操作手册所描述的软件系统的名称；

b.该软件项目的任务提出者、开发者、用户及安装该软件的计算中心。

1.3 定义

[列出本文件中用到的专门术语的定义和外文首字母组词的原词组。]

1.4 参考资料

[列出有用的参考资料，如： ]

a. 本项目的经核准的计划任务书或合同、上级机关的批文；

b. 属于本项目的其他已发表的文件；

c. 本文件中各处引用的文件、资料；

[包括所列出的这些文件资料的标题、文件编号、发表日期和出版单位，说明能够得到这些文件资料的来源。 ]

2 软件征述

2.1 软件的结构

[结合软件系统所具有的功能包括输入、处理和输出提供该软件的总体结构图

表。 ]

2.2 程序表

[列出本系统内每个程序的标识符、编号和助记名。]

2.3 文件表

[列出将由本系统引用、建立或更新的每个永久性文件，说明它们各自的标识符、编号、助记名、存储媒体和存储要求。 ]

3 安装与初始化

[一步一步地说明为使用本软件而需要进行的安装与初始化过程，包括程序的存载形式，安装与初始化过程中的全部操作命令，系统对这些命令的反应与答复，表征安装工作完成的测试实例等。如果有的话，还应说明安装过程中所需用到的专用软件。 ]

4 运行说明

[所谓一个运行是指提供一个启动控制信息后，直到计算机系统等待另一个启动控制信息时为止的计算机系统执行的全部过程。]

4.1 运行表

[列出每种可能的运行，摘要说明每个运行的目的，指出各自所执行的程序。 ]

4.2 运行步骤

[说明从一个运行转向另一个运行以完成整个系统运行的步骤。 ]

4.3 运行 1（标识符）说明

[把运行 1 的有关信息，以对操作人员为最方便最有用的形式加以说明。]

4.3.1 运行控制

[列出为本运行所需要"的运行流向控制的说明。]

4.3.2 操作信息

[给出为操作中心的操作人员和管理人员所需要的信息，如：]

a. 运行目的；

b. 操作要求；

c. 启动方法 如应请启动（由所遇到的请求信息启动）、预定时间启动、…等；

d. 预计的运行时间和解题时间； 操作命令；

f. 与运行有联系的其他事项。

4.3.3 输入一输出文件

[提供被本运行建立、更新或访问的数据文件的有关信息，如：]

a. 文件的标识符或标号；

b. 记录媒体；

c. 存留的目录表；

d. 文件的支配如确定保留或废弃的准则、是否要分配给其他接受者、占用硬设备的优先级以及保密控制等有关规定。

4.3.4 输出文件

提供本软件输出的每一一个用于提示、说明、或应答的文段（包括"菜单"）的有关信息，如：

a. 文段的标识符；

b. 输出媒体（屏幕显示、打印、……）；

c. 文字容量；

d. 分发对象；

e. 保密要求。

4.3.5 输出文段的复制

[对由计算机产生，而后需用其他方法复制的那些文段提供有关信息，如：]

a. 文段的标识符；

b. 复制的技术手段；

c. 纸张或其他媒体的规格；

d. 装订要求；

e. 分发对象；

f. 复制份数。

4.3.6 恢复过程

[说明本运行故障后的恢复过程。]

4.4 运行 2（标识符）说明

[用与本手册 1.4.3 条相类似的方式介绍另一个运行的有关信息。]

5 非常规过程

[提供有关应急操作或非常规操作的必要信息，如出错处理操作、向后备系统的切换操作以及其他必须向程序维护人员交待的事项和步骤。]

6 远程操作

[如果本软件能够通过远程终端控制运行，则在本章说明通过远程终端运行本软件的操作过程。]

## 软件维护需求表

编者说明：

软件维护工作是一个持续的过程，该表格用于在客户服务部门与开发部门之间的联系。它将维护的要求详细地提供给开发部门，以帮助他们更好地有针对性地安排维护工作。

| |
|---|
| 维护需求的编制者：_____<br>申请者：_____<br>模块/程序名：_____<br>完成人员：_____<br>紧急程度：[ ] 紧急 [ ] 高 [ ] 中 [ ] 低 |
| 问题/需求描述： |
| 维护案例的标志：_____<br>维护活动的标志：_____<br>估计成本：_____<br>维护工作开始时间：_____<br>预计维护工作结束时间：_____<br>累计成本：_____ |

| |
|---|
| 对产品和修改的模块所产生的影响/注释： |
| 接收/拒收：_____<br>完成的维护工作：_____<br>日期/签名：_____ |

# 软件问题报告表

编者说明：

软件维护通常从问题报告开始，上门维护的人员有义务将问题进行整理，以便于开发人员找到原因，提供解决方案。

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 软件问题报告 | | | 登记号： | | | | | | |
| | | | 登记日期： | | | | | | |
| | | | 时间： | | | | | | |
| 阶段： | | | 状　态： | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 报告人资料 | 姓　名 | | 电　话 | | | | | | |
| 地　　址 | | | | | | | | | |
| 问题：　　程　序[　]　　数据库[　]　　文　档[　] | | | | | | | | | |

| | | |
|---|---|---|
| 文档/模块： | 版本号： | 磁带： |
| 数据库： | 文档： | |
| 测试用例： | 硬件： | |

| |
|---|
| 问题描述/影响：<br><br>签名：　　日期： |
| 软件开发部意见：<br><br>签名：　　日期： |

| |
|---|
| 附注： |

## 软件问题解决记录表

编者说明：

该表格用于上门维护人员，记录其发现问题之后的解决过程，将其备案对于维护工作有很重要的价值。

| | |
|---|---|
| 软件问题报告号： | |
| 软件维护人： | 维护时间： |
| 软件解决过程：<br><br>签名：　　日期： | |
| 软件用户意见：<br><br>签名：　　日期： | |
| 软件开发部意见：<br><br>签名：　　日期： | |
| 备注：<br><br>签名：　　日期： | |

## 软件维护报告表

编者说明：

该表格用于开发部门对软件所做出的维护性修改，将其记录在案，是十分必要的，防止文档的不一致性带来的维护麻烦。

维护案例的标志：_____

维护活动的标志：_____

维护需求的类型：[　]改正　　[　]改编　　[　]调整　　[　]扩充

需要维护的原因和维护后产生的影响：

| | 原　因 | 影　响 |
|---|---|---|
| 需求定义 | | |
| 设计 | | |
| 软件环境 | | |
| 硬件环境 | | |
| 优化 | | |
| 其它 | | |

所有维护过的模块和系统的结果及成本/工作：

| 模块标志 | 维护的行数不清 | | | 工作 |
|---|---|---|---|---|
| | 源码 | 文档 | 总计 | (人小时) |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

**删除的内容：** *注：该方案通常用于投标*

对所做维护工作的注释：

维护人签名：

日期：

## 系统方案书模板

编者说明：

现在许多系统都需要进行招标，所以投标用的技术方案是开发团队经常编写的，本文档则从许多优秀的系统方案书中总结出一个基础的框架，相信对于大家很有帮助。

1.前言

1.1 项目简介

[在本小节对该项目的基本情况做一简单的介绍。]

1.2 背景分析

[在本小节应列举一些与该项目相关的背景资料以及相关的分析，可以包括国际、国内的动向，当地应用的基本情况，客户群体背景等方面的内容。]

1.3 建设目标

[对该项目的建设目标做出一个概要性的描述，以帮助读者能够很快地抓住主题，对项目的意义与远景有一个共识。]

1.4 系统设计原则

[说明该方案中的设计方案的设计原则，通常包括先进性、安全性、实用性、经济性，或者是诸如什么统一规划、统一协调之类的大方向。]

1.5 遵循的标准与规范

[如果业主方有需求，或者你的设计方案是符合某个国际标准、国家标准、行业标准的话，应该在本小节中列出这些规范，并且说明你在设计方案中是如何满足这些规范的，这样做将带来什么样的好处。]

2.系统远期规划

[如果你所做的系统将是一个长期性大项目的第一期，或者是其中的一期，那么你应该从整个系统的远期规划着手，描述该项目的长远目标和远景。然后从中导出你所做的这一期的建设规划，从而使业主明白你的设计方案与长远规划的一致性。]

3.项目建设计划

[在本节中，你应该对本期建设的组织结构、实施进度计划等方面的内容进行阐述，这样让业心明白你需要多少时间来完成本期项目。]

4.需求分析

[该部分内容主要来源于招标文件，你可以在招标文件提出的系统需求的基础上，进行扩展性描述，也可以对其进行合理的重新组织，使得其更加的规格化。然后对其需求进行分析，为系统总体设计打下铺垫。]

5.系统总体设计

[这是系统设计方案中的最重要的一部分，通过该部分的描绘，你将为业主构建一个良好的框架，让其对你的设计思路有一个总体上的了解。]

4.1 系统架构

[在该小节中，你应该给出一个系统总体结构图，这个图是一个让用户直观地获得整个系统结构的示意图。然后对该图进行一些必要的补充说明，帮助读者更好地理解总体结构。同时你应该结构系统总体结构图对各个子系统的功能以及它们之间的关系进行描述。如果需要的话，还可以分小节进行描述。]

4.2 技术说明

[在该小节中，你可以对选用的主要技术进行必要的解释和说明，帮助读者了解这些技术的特点，以及其优势和采用的原因。]

4.3 系统的设计相关考虑

[在该小节中，你可以对总体设计是考虑到的一些非功能因素进行描述，例如安全性、兼容性、对原有资源的利用等。]

5.主机系统设计方案

[如果在你所设计的系统中有使用到主机，那么就应专僻一个章节来说明。这部分的内容主要是主机的选型，主机的详细技术参数，你的选择考虑、理由，也就是要达到说服业主采用你的选型方案。你可以从主机系统的产品技术白皮书上获取这些资料，并且从性价比，使用情况等方面进行深入的描述。]

[如果需要，也可以提供多种不同的方案建议，并充分说明这些选择的优劣点，供业主

根据实际的需要进行选择。]

[通常情况下，主机的性能、功能要求在招标书中会列出，因此你在描述的时候应该结合这些内容进行针对性的说明。]

6.软件系统设计方案

[如果项目中包括计算机软件应用系统的开发，则你应该专门单列出个章节进行描述。在这里，你应该列出软件系统的总体设计，各个子系统之间的关系，每个系统的设计考虑，以及将采用什么操作系统、数据库、中间件、开发工具，并且说明理由。]

[对于你所选择的操作系统、数据库、中间件、开发工具应该列举出详细的技术资料，你可以从相关的产品白皮书中获得这些资料，并结合它们的优势在该项目中应用的好处的角度多做一些阐述。]

[通常本章节中应该包括软件系统总体设计，各个子系统的说明等小节。]

7.具体应用系统设计

[如果项目中包括一些如视频会议系统、邮件系统、WEB 系统等，则应该单独地进行描述，可以根据其规模决定是都放在一个章节还是多个章节。主要内容就是产品选型，技术说明、推荐理由等。]

8.网络架构方案

[如果系统中有计算机网络部分，则应专门列出章节进行描述。在本章节中应该包括网络架构的拓扑结构设计图。并对其进行说明。]

[另外，还应该对于网络中的各种设备，如路由器、交换机等提供相应的设计的选型、产品技术参数、推荐理由，分析的依据等内容。]

[如果涉及到综合布线，则应该分一小节进行描述。]

9.系统的安全性与扩展性

[对于一些系统安全性、扩展性等项目十分关注的非功能因素，应该单独地放置在一个章节内进行统一描述。主要包括采用的技术、措施，能够达到的效果，以及与系统的综合考虑。]

10.技术培训与支持

[最后，还应该对项目的技术培训与售后技术服务进行说明。]

10.1 项目培训

[通常包括培训目标、培训人员、培训内容、培训方式等内容。]

10.2 售后服务

[通常包括应用软件维护、现场维护、巡检维护、备件管理、安装规范以及售后服务响应体系的介绍。]

## 软件过程规范示例

编者说明：

软件过程管理中的一个很重要的工作就是制定项目、组织的过程规范，它是软件开发组织行动的准则与指南。该文档就是一个实际的过程规范的实例，通过该实例，相信对大家根据自身情况制定符合要求的项目过程规范、组织过程规范有很好的借鉴作用。

### 1.总则

最大限度提高 Q&P（质量与生产率），提高 Q&P 的可预见性，是每一个软件开发机构的最大目标。而 Q&P 依赖于三个因素：过程、人和技术，因此要实现 Q&P 的提高，除了加强技术能力，引进、培育更多优质技术人才之外，规范、改进机构的过程是一个十分重要的手段。我们希望通过在制定软件过程规范标准，并在软件开发实践中不断地完善、修订，

提高 Q&P 和 Q&P 的可预见性。

本规范采用 CMM（软件过程成熟度模型）的指导，吸收 RUP、XP、MSF、PSP、TSP 等过程规范指南的思想、方法及实践，充分结合 xxx 技术开发部的实际情况，引入先进的技术、方法、工具，为公司的软件开发工作提供一部详细、可操作的过程指南。在本规范的第一版本中，主要包括管理过程和开发过程两个部分，管理过程中包括项目管理过程、需求变更管理过程、配置管理过程。对于软件开发项目中的其它的一些过程将在实践中逐步补充、完善。

## 2.项目管理过程规范

项目管理过程主要包括三个阶段：项目立项与计划、项目实施、项目关闭。

### 2.1 项目立项与计划

**参与人员：**技术开发部指定的项目负责人（包括前期负责人、正式的项目经理）、立项申请人、[相关最终客户]以及实施该项目的开发组队成员；

**入口准则：**接到经公司总经理或副总经理批准的市场部门的《软件开发立项申请表》；

**出口准则：**立项申请人签字确认了经修订正后的正式《软件项目计划》，并通过《工作任务卡》下达了开发任务，开发工作正式开始；

**输入：**经审批的《软件开发立项申请表》、与需求相关的业务资料；

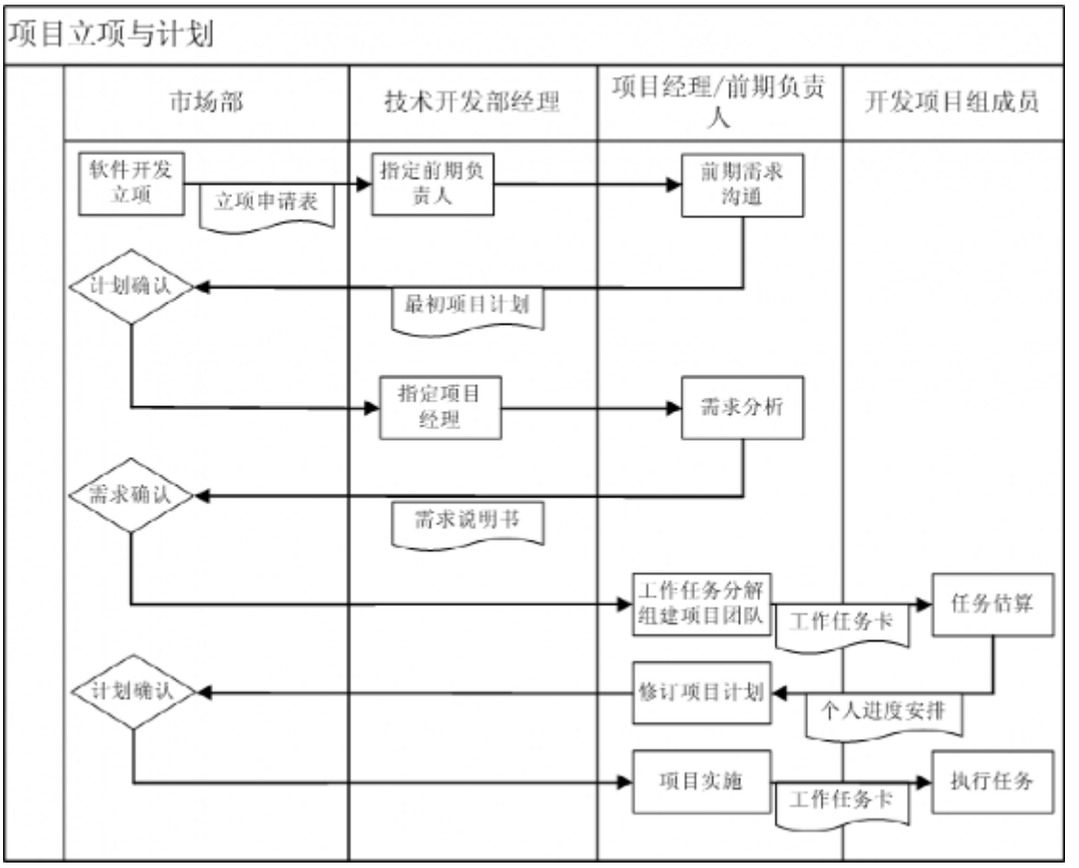
**输出：**《软件项目计划》、《软件需求规格说明书》、《开发任务卡》；

**活动：**

1. 接到《软件开发立项申请表》后，技术开发部经理指定前期负责人，并告知立项申请人；
2. 前期负责人阅读《软件开发立项申请表》后，通过与立项申请人的沟通、阅读立项申请人提交的材料、通过立项申请人与客户直接交流等方式，了解项目目标、范围与基本需求；并形成最初的《软件需求规格说明书》；
3. 前期负责人会同技术开发部经理以及其它相关人员，制定最初的《软件项目计划》，并组织评审；
4. 向立项申请人提交最初的《软件项目计划》；
5. 最初的《软件项目计划》通过立项申请人的确认后，项目经理计划安排需求分析；
6. 需求分析完成后，形成正式的《软件需求说明书》，提交立项申请人确认；（需求分析过程参见开发过程规范部分）
7. 根据立项申请人确认后的《软件需求说明书》，项目经理组织进行软件高层设计，并对工作任务进行分解，并根据实际需要向技术开发部经理申请资源，组建项目组队；
8. 项目经理根据工作任务分解，下发《工作任务卡》，并协同组队成员进行任务估算；

   *注：工作任务包括模块开发任务、其它任务（如安装）；模块开发任务主要包括：详细设计、编码和单元测试*
9. 任务估算完成后，组队成员向项目经理提交《个人进度安排》（以甘特图的形式表示），项目经理根据每个组队成员的《个人进度安排》修订《软件项目计划》（必须包括总的计划甘特图），并提交立项申请人确认；
10. 立项申请人确定后，项目经理根据软件项目计划基线，补充《工作任务卡》，下发到每个组队成员，开发工作开始。

项目立项与计划过程的工作流程如下图所示：




*图表 1　项目立项与计划工作流程图*

**相关模板：**

《软件需求规格说明书》、《软件项目计划》、《工作任务卡》

**说明：** 如果计划确认、需求确认未通过，立项申请人与项目经理进行协商，进行修正，无法达成共识的，提交部门经理、总经理协调；

**2.2 项目实施**

**参与人员：**项目经理，项目组成员；

**入口准则：**项目计划基线已建立，并通过立项申请人确定，带有工作进度要求的《工作任务卡》已下发到每个项目成员；

**出口准则：**立项申请人在《验收报告》上签字确认；

**输入：**《软件需求规格说明书》、《软件项目计划》、《工作任务卡》；

**输出：**经验收测试的可交付的程序、源代码及相关文档。

**活动：**

1、 在开发期间，项目成员每周需上交一份《时间日志》、《缺陷日志》，每天向项目经理汇报工作任务进度；

2、 在开发期间，项目经理负责填写《项目进度周报》报于技术开发部经理、立项申请人（格式不同，交予立项申请人的只需周报的第一页，报予技术开发部经理的项目进度周报的第二页为“跟踪甘特图”）；

3、 项目经理必须根据实际的进度情况，及时调整项目计划，若发现进度延误，需采取措施。

**相关模板：**

《软件项目计划》、《开发任务卡》、《时间日志》、《缺陷日志》、《项目进度周报》

### 2.3 项目关闭

**参与人员：**技术开发部经理或经理助理、项目经理，项目组成员、立项申请人、[相关客户、公司总经理、公司副总经理]；

**入口准则：**立项申请人在《验收报告》上确认；

**出口准则：**形成《项目总结》，完成项目绩效考核，项目数据存入“过程数据库”；

**输入：**《时间日志》、《缺陷日志》、《项目开发计划》；

**输出：**《项目总结》、已完成的《项目绩效考核表》、过程数据库中的该项目记录；

**活动：**

1、 项目经理主持召开项目总结会，交流项目实施过程中的心得体会，对项目实施中的成功处、不足处进行总结，并由项目经理形成《项目总结》；
2、 由技术开发部经理组织对该项目进行绩效考核，并填写相应的《项目绩效考核表》；
3、 项目经理组织所有成员对项目过程中的文档、源程序等资料进行整理、归档；
4、 由项目经理根据过程数据库的需要，整理相应的数据，提交技术开发部经理，存入过程数据库。

**相关模板：**

《项目总结》、《项目绩效考核表》

## 3.开发过程规范

开发过程是提炼用户需求，设计、构建和测试满足这些需求的软件并最终将其交付给客户的过程。是软件过程中的主体过程之一。当开发新的应用或计划为现有的应用进行重要的增强时，需使用本规范所定义的开发过程执行。

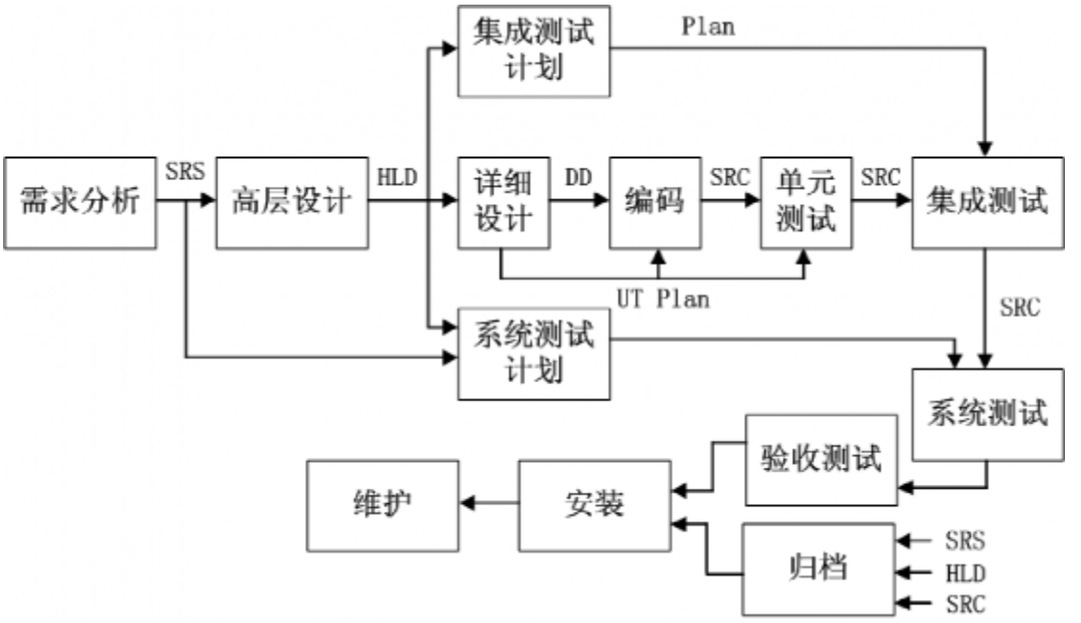
项目管理过程是对开发过程进行计划、监控/管理、总结的辅助过程，但由于项目管理是保证进度、质量的重要手段，因此在软件项目中也是十分重要的过程之一。而需求管理过程与配置管理过程则是次重要的辅助过程，需求管理过程是一个需求变更管理的过程，以对变更进行统一的管理；配置管理过程的最重要工作就是版本控制，使得开发过程中的各种交付物能够有机地形成一个个整体。

因此以上四个过程是交织进行的，均是为成功完成软件项目的保障过程。

### 3.1 过程总述

现在比较通行的开发过程模型包括：瀑布模型、演化模型、原型模型、螺旋模型等。根据公司的项目特点、队伍规模、组队情况等实际因素，决定选择最为简单、易于掌握的瀑布模型为基础，根据公司特点，进行合理的修改，使其成为公司本阶段的软件开发过程。

正如下图所示，本规范将整个开发过程分为：需求分析、高层设计、详细设计、编码和单元测试、集成计划与测试、系统测试、验收测试与安装、维护等八个阶段。




*图表 2　开发过程总图*

*注：SRS：软件需求规格　　HLD：高层设计　　DD：详细设计*

*SRC：代码　　UT Plan：单元测试计划*

*注：“归档”在配置管理过程统一说明。*

**3.2 需求分析阶段**

需求分析的主要目的是生成一个正确说明客户所有需求的文档。换言之，软件需求规格（Software Requirement Specification，SRS）文档是该阶段的主要输出。正确的需求分析和确定需求规格对一个项目的成功是非常关键的。许多在系统和验收测试时发现的缺陷是在需求阶段产生的。在验收阶段去掉需求阶段产生的一个错误将比在需求阶段本身去掉该错误要多花 100 多倍的费用。很明显，在执行这阶段时，正确地生成具有最少缺陷的 SRS 是非常必要的。

**参与人员：**项目经理，[分析员]，立项申请人，[客户，最终用户]；

**入口准则：**项目立项，最初的项目计划已得到立项申请人的确认。

*注：这里所说明的**需求分析**阶段是进行开发过程的需求分析阶段，在技术开发部出具初步的项目计划之前的需求沟通工作，不是该过程规范所定义的。最初的需求沟通工作可以参考本过程规范。*

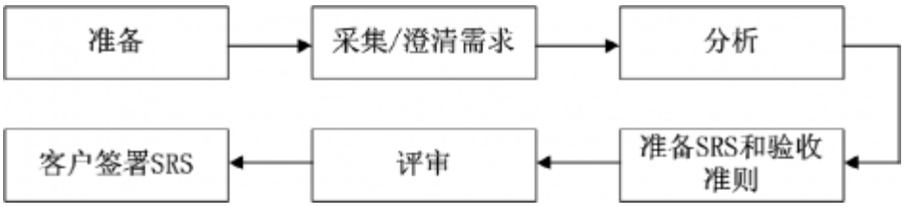
**出口准则：**立项申请人、[客户]在《软件需求规格说明书》上签字确认；

**输入：**《项目立项申请表》、最初的《项目计划》，需求相关的资料；

**输出：**经确认的《软件需求规格说明书》；

**活动：**

整个需求分析过程主要包括以下几个步骤：




*图表 3　需求分析阶段活动总图*

1、 首先，项目经理与分析员一块，做好需求分析的准备，包括阅读相关的背景

资料，熟悉客户的实际情况，准备用户访谈计划，准备会谈问题清单等；

2、 然后通过面谈、专题讨论会等形式与客户进行沟通，采集需求的详细内容，澄清每一个需求点；从而界定出系统的目标和范围；
3、 对所采集和澄清的需求进行分析，构建需求模型，从功能性、非功能性两个方面进行需求分析，深入领会客户需求；
4、 形成《软件需求规格说明书》，建立软件需求基线，并为软件需求评审做好准备；
5、 由项目经理安排软件需求评审，协同立项申请人、[客户]进行需求评审；
6、 立项申请人[或客户]在《软件需求规格说明书》上确认。

**相关模板：**

《软件需求规格说明书》

### 3.3 高层设计阶段

高层设计是软件开发过程中的一个重要阶段，在这个阶段将从计算机实现的逻辑角度开发针对用户需求的解决方案。这一解决方案是一个高级的抽象方案。高层设计要设计出各主要部分，并说明他们在技术上如何工作：1）相互间的协作；2）所需外在的硬件和软件环境；3）内在环境。也就是说，高层设计确定了组成产品的构件，定义了每个构件的功能任务，并且定义了构件间的接口及构件到运行环境的外部接口。

**参与人员：**项目经理，项目组员（设计团队）；

**入口准则：**《软件需求规格说明书》已通过立项申请人的确认；

**出口准则：**形成高层设计，实现任务分解，所有的问题得到解决；

**输入：**《软件需求说明书》

**输出：**《高层设计说明书》（功能与数据库设计）、详细设计、编码、文档和用户接口标准；

**活动：**

1、 制定详细设计、编码、文档和用户接口的标准；
2、 根据项目特点选择运行的目标平台和开发工具；
3、 制定软件的体系结构，定义逻辑和物理的对象模型，包括确定类、类的属性、类方法、类之间的关系和对象间的动态交互。若采用结构化设计，则该活动应为功能设计；
4、 从需求规格说明书中的数据模型中得到物理数据库结构，进行物理数据库设计：包括确定表/记录类型、域和其他部分。
5、 生成高层设计说明书，并组织设计评审。

**相关模板：**

《高层设计说明书》

### 3.4 详细设计阶段

在详细设计阶段，高层设计阶段开发出的整体应用被分成几个模块（或构件）和程序。为每个程序（或构件）进行逻辑设计，然后归档作为程序规格，同时为每个程序（或构件）生成一个单元测试计划。详细设计阶段的重要活动包括通用例程和程序的确定、框架程序的开发以及用于提高生产率的实用程序和工具的开发。

在详细设计阶段负责每个程序、模块（或构件）的内部设计，确定其程序流程，并且可以通过使用设计语言、图形流程图（如活动图、状态图）等，或通过简单地写叙述而将设计文档化。

**参与人员：**每个模块（或构件）的任务承担人；

**入口准则：**《高层设计说明书》已通过评审；

**出口准则：**完成详细设计，所有的问题得到解决，详细设计与单元测试计划文档化；

**输入：**《软件需求规格说明书》、《高层设计说明书》、详细设计标准

**输出：**《详细设计说明书》、《单元测试计划》

**活动：**

1、 将高层设计中的每个程序（或构件）细分成小的组件；
2、 对每个小组件进行详细设计，包括确定调用方法、输入和输出、程序逻辑、数据结构等；
3、 根据组件的逻辑，制定单元测试计划，包括确定单元测试环境、测试用例、测试数据等；
4、 向项目经理（或高层设计者）提交详细设计与单元测试计划；

**相关模板：**

《详细设计说明书》、《单元测试计划》

**剪裁说明：**对一些小项目，详细设计阶段的活动 1、2 可以省略。

### 3.5 编码和单元测试

在编码子阶段，根据详细设计用编程语言编写所需的程序。这个阶段根据合适的编码规范产生源代码、可执行代码以及数据库（如果使用了数据库）。这个阶段的输出是随后测试和验证的主体。而单元测试子阶段则是根据详细设计阶段所制定出来的单元测试计划进行测试，验证每一个组件正确、可用。

**参与人员：**每个模块（或构件）的任务承担人；

**入口准则：**《详细设计说明书》已通过批准，编码规范已建立；

**出口准则：**成功执行所有单元测试计划中的测试用例；

**输入：**《软件需求规格说明书》、《高层设计说明书》、《详细设计说明书》、《单元测试计划》编码、用户接口标准；

**输出：**测试数据、源代码、可执行代码、《单元测试报告》

**活动：**

1、 根据详细设计，按照编码、用户接口规范编写程序；
2、 对程序进行代码复查、编译、调试，直到程序运行通过，符合详细设计的要求；
3、 根据单元测试计划进行单元测试，生成单元测试报告。

**相关模板：**

《单元测试报告》

### 3.6 集成计划与测试

集成是把设计阶段制定的，已通过单元测试的模块构建成一个完整软件结构的系统方法。可采用很多方式进行集成，集成计划必须指定模块集成的顺序。在该阶段，同时进行测试，以发现与接口相关的缺陷。集成按照集成计划中制定的顺序进行，并执行每个集成阶段的相应测试用例。集成计划描述了集成顺序、额外需要的软件、测试环境和资源需求。集成计划与集成测试计划通常一起完成。

**参与人员：**项目经理，集成团队；

**入口准则：**经批准的《高层设计说明书》；

**出口准则：**集成计划和集成测试计划经过评审和授权；

**输入：**《高层设计说明书》、源程序

**输出：**《集成计划》、《集成测试计划》

**活动：**

1、 确定集成所需的环境，包括硬件的物理特性、通信和系统软件、使用模式等；

2、 决定集成规程，确定将要集成的关键模块，集成的顺序，需要测试的接口等；
3、 开发集成测试计划，确定测试用例和执行用例的规程，确定测试数据，确定期望输出等。

**相关模板：**

《集成计划》、《集成测试计划》

**剪裁说明：**对一些小项目，集成计划与测试阶段可以省略。

### 3.7 系统测试

系统测试是依据需求规格验证软件产品有效性的活动。这个阶段是为了发现那些只有通过测试整个系统才能暴露的缺陷。就像外部接口、性能、安全、配置敏感性、共存、恢复以及可靠性等属性只有在这个阶段才能判断其是否有效。可以使用具有不同测试目的的一系列测试来验证所有系统元素都已经正确地集成，系统能够执行所有功能并满足所有非功能需求。系统测试开始之前，必须在系统测试计划阶段详细地制定计划。

系统测试计划工作从需求分析结束后就可以开始，一直到编码时结束。

**参与人员：**项目经理，系统测试团队；

**入口准则：**经确认的《软件需求规格说明书》和经批准的《高层设计说明书》；

**出口准则：**系统测试计划经过评审和授权，成功执行所有系统测试计划中的测试用例；；

**输入：**《软件需求规格说明书》、《高层设计说明书》

**输出：**《系统测试计划》、《系统测试报告》

**活动：**

1、 决定所需的测试环境；
2、 决定系统测试的规程，包括：确定测试特性，如用户接口、软硬件接口、通信接口、主要业务过程；确定不需要测试的重要特性以及不测试的原因；确定关键测试；
3、 开发测试用例，包括确定每个测试用例以及执行它的规程，确定每个输入、输出数据的要求，确定预期的结果。

**相关模板：**

《系统测试计划》、《系统测试报告》

**剪裁说明：**对一些小项目，系统测试阶段可以省略，直接准备验收测试，在验收测试之前，开发组队按验收测试计划做一次没有立项申请人、[客户]参加的预测试。

### 3.8 验收测试与安装

验收测试和安装阶段的主要任务是将软件产品集成到它的操作环境中，并在这个环境中经受测试，以确保它按需求执行。这个阶段包括两个基本任务：使软件得以验收和客户处安装软件。验收指的是由立项申请人、[客户]根据早期准备的《验收报告》而进行正式的测试，并对测试结果进行分析，以确定系统是否满足验收准则。当分析结果满足验收测试时，用户接受软件。安装指的是把接受的软件置于实际产品环境中。

*注：《验收报告》应附有验收测试计划*

**参与人员：**项目经理，安装团队、立项申请人、[客户]；

**入口准则：**成功地完成了系统测试（或成功地完成了验收预测试）；

**出口准则：**立项申请人或客户在《验收报告》上签署确认意见；

**输入：**《软件需求说明书》、测试后的软件和《验收报告》

**输出：**签署了确认意见的《验收报告》和安装后的软件；

**活动：**

1、 根据《软件需求说明书》，编写验收报告；
2、 与立项申请人、[客户]一起按《验收报告》执行验收测试，包括：在验收环境下安装软件、进行实况运行、协助客户进行验收测试、改正验收缺陷、更新文档以反映所有变更、获得客户的验收确认；
3、 执行安装，包括：在产品环境下安装软件、搭建产品环境、载入软件和数据、进行实况运行、修改安装缺陷、执行用户培训。

**相关模板：**

《验收报告》

### 3.9 维护

维护支持阶段是指已安装的应用得到支持，直至其在生产环境中稳定运行的阶段。

**参与人员：**项目经理，系统安装人员；

**入口准则：**软件在生产中运行；

**出口准则：**合同中指定的维护支持阶段终止；

**输入：**安装后的应用、用户文档和《软件维护申请表》；

## 4.需求变更管理过程规范

需求变更，这是个永恒的真理。需求变更的一个重要原因是系统周围的世界在变化，从而要求系统适应这个变化。在项目生命周期的任何时候或者项目结束之后都可以有需求变更。与其希望变更不会来临，不如希望初始的需求在某种程度上做得很好而使得没有变更需求，最好是项目准备时想到对付这些变更，以防变更真的到来。不管做多少准备和计划都不可能阻止变更，说项目在需求冻结后再开始不过是个神话罢了。

### 4.1 过程总述

需求变更管理过程定义了一系列活动，当有新的需求或对现有需求进行变更（我们可以称它们都是需求变更）时就会执行这些活动。需求变更可以在项目执行的任何一个点上发生。需求变更会影响项目进度，甚至会影响已经生产出来的产品。越是在生命周期后期的需求变更，对项目的影响越严重。不可控的需求变更导致对成本、进度以及项目质量的负面影响，这些极可能严重危害项目成功的概念。

需求变更管理过程用来控制需求变更并减少他们对项目的影响。这个目标需要理解需求变更请求的隐含意义，以及变更带来的总影响。同样，也需要立项申请人、[客户]意识到变更对项目影响的后果，使得可以友好地将变更反映到协商好的条款中。需求变更管理过程，从某种程序上说，试图保证在需求变更影响下项目依然可以成功。

需求变更管理有两个方面，一方面与立项申请人、[客户]就怎样处理变更达成一致，一方面是实际进行变更的过程。处理变更的整体方法必须与立项申请人、[客户]达成一致。一般来说，它制定怎样进行变更请求，当需要正式的批准时，为处理变更估计留出冗余空间等等。在整个方法的背景下，当需求变更到来时，需要执行需求变更管理过程。

### 4.2 过程规范

**参与人员：**项目经理，立项申请人、[客户]、开发团队；

*注：项目经理对将变更纳入项目中所需的过程执行负主要责任。立项申请人、[客户]以及开发队伍也需要参与这个过程。*

**入口准则：**收到立项申请人提交的《需求变更请求单》

**出口准则：**变更已列入新的《软件需求说明书》，并体现在新的《软件项目计划中》；

**输入：**《需求变更请求单》

**输出：**根据《需求变更请求单》，在充分协商与的基础上，提交新的《软件需求说明书》，并提交《软件项目计划变更表》；

**活动：**

1、 记录需求变更请求，记录项中应包括变更请求数、变更的简要描述、变更的影响、变更请求的状态和关键数据；
2、 分析变更请求对工作的影响；
3、 估计变更请求需要的工作量；
4、 修改项目计划，重新估计交付时间；
5、 对总的成本花费的影响进行估计；
6、 将修改过的项目计划提交立项申请人，并获得确认。

**相关模板：**

《项目计划变更表》

## 5. 配置管理过程规范

软件项目在其执行过程会产生大量的工件，包括各种文档、程序、数据和手册。所有这些工件都是易于改变的。这是软件一个独有的特点。正如“需求变更管理”章节中所述，在软件项目中，在项目执行过程中的任何时候，需求本身都会发生变更。为避免项目在变更时失控，正确控制和管理变更是很必要的。配置管理（Configuration Management，CM）又称为软件配置管理，是项目管理中专用于关注系统地控制项目进行中发生的变更的那些部分，由用来识别机构软件产品并控制其修改的一系统活动构成。

配置管理需要满足项目基本目标之一：为客户提交高质量的软件产品。这个提交的产品，包括各种资源以及构成资源或目标代码的目标文件，还包括以这些文件来构建工作系统的脚本以及相关文档。在项目中，资源和文档通常以很多独立文件的方式来维护。

当项目进展时，文件发生了改变，产生了不同的版本。在种情况下，即使将项目的各部分组合起来，构建成系统，也是很困难的任务，怎样保证合并的是源程序的正确版本以及没有遗漏任何源程序？还有，怎样保证传送的文档的版本是正确的，该版本和最终交付的软件是一致？对于这类型的情况，必须正确跟踪软件开发过程中的各种中间产品、其版本以及软件产品的版本。没有这些信息，交付最终系统就成为繁重的任务。这个活动不是由开发过程完成的，而需要一个独立的过程，那就是配置管理过程。

### 5.1 配置管理的目标

配置管理过程，需要达到以下目标：

1） 能够随时给出程序的最新版本；
2） 能够处理并发的文档、程序的更新/修改请求；
3） 能够根据需要撤消程序的修改；
4） 能够有效防止未授权的程序员对文档、程序进行变更或删除；
5） 能够有效地显示变更的情况。

### 5.2 配置管理过程规范

配置管理过程包括两个主要阶段：配置管理计划、实施配置管理。

#### 5.2.1 配置管理计划

**参与人员：**项目经理，配置管理团队；

**入口准则：**《软件需求规格说明书》已经确认；

**出口准则：**完成项目配置管理计划；

**输入：**《软件需求规格说明书》

**输出：**《配置管理计划》

**活动：**

1、 识别配置项，配置项的典型例子包括需求规格、设计文档、源代码、测试计划、测试脚本、测试规程、测试数据、项目使用的编码、用户接口规范、验收报告等；

2、 定义为配置项命名和编号的计划：如果使用 CM 工具，那么有时由工具处理版本编号，否则，在项目中必须明确地进行版本编号；
3、 定义 CM 所需的目录结构；
4、 定义访问控制；
5、 定义变更控制规程；
6、 确定 CM 工作人员的责任和权利；
7、 定义跟踪配置项状态的方法；
8、 定义备份制度
9、 定义发布制度；
10、 确定将配置项转移到基线的原则。

**相关模板：**

《软件配置管理计划》

#### 5.2.2 实施配置管理

**参与人员：**项目经理，配置管理团队、开发项目组队成员；

**入口准则：**《软件配置管理计划》已批准，项目开始；

**出口准则：**项目结束；

**输入：**《软件配置管理计划》

**活动：**

1、 接受变更请求；
2、 Check out 需要变更、修改的配置项，并进行修改；
3、 Check in 变更、修改过的配置项。

## 6. 附件

附件包括各种文档模板与工作指南。所有附件以单独的文档形式存储，文档名为 xxxx 模板、xxxx 工作指南。具体包括：

### 6.1 文档模板

#### 6.1.1 项目管理类

《软件项目计划模板》、《工作任务卡模板》、《时间日志模板》、《缺陷日志模板》、《项目进度周报模板》、《项目总结模板》、《项目绩效考核表模板》、《项目计划变更表模板》、《软件配置管理计划》

#### 6.1.2 开发过程类

《软件需求规格说明书模板》、《高层设计说明书模板》、《详细设计说明书模板》、《单元测试计划模板》、《单元测试报告模板》、《集成计划模板》、《集成测试计划模板》、《集成测试报告模板》、《系统测试计划模板》、《系统测试报告模板》、《验收测试报告模板》。

### 6.2 工作指南

《软件需求分析工作指南》、《软件项目计划工作指南》、《软件需求管理工作指南》、《软件配置管理工作指南》

**网站定量评估的度量指标**

| 度量指标 | 描述 | 获得方法 | 评估注意事项 |
|---|---|---|---|
| 点击数 | 访问服务器上某个文件的请求 | 日志文件 | 目前该指标的有效性大大降低，因为一个页面可能伪造成多个点击。 |
| 页面访问次数 | 访问服务器上一个 HTML 页面的请求 | 日志文件 | 注意主机、代理服务器和缓存可能会误报页面访问次数 |

| | | | |
|---|---|---|---|
| 唯一用户数 | 有不同IP地址或Cookie的用户 | 日志文件/数据库分析 | 动态分配IP会导致统一数大于实际数，而代理服务器会导致统一数小于实际数 |
| 用户会话数 | 与网站连接的时间超过30分钟而且没有中断的对话 | 日志文件 | 可以在网络服务器上设定超时时间，默认设置为30分中。然而通过修改这个设置，使采集的结果产生偏差 |
| 用户会话 | 平均的用户会话长度 | 日志文件 | 由于多个用户共享一台计算机或由于使用代理服务器上网，因此很难判断一个用户产生了会话，期间他离开了，而由另一个用户接替这个会话 |
| 访问网站的顶级路径 | 当用户访问网站时，大多数人访问页面的顺序 | 日志文件 | 如果使用框架的话，可能会到“无人访问的结果” |
| 进入和离开页面 | 大多数用户进入或离开网站的页面 | 日志文件 | 有助于了解用户是否从外部的链接进入网站，或从一个标签页进入，而非从网站主要进入 |
| 与其它网站互相链接的点击数 | 常常用来测量广告链条的情况 | 日志文件(广告服务器) | 用来评测一个广告条的吸引力，不能用来评测品牌的知名度或链接的效果 |
| 涉及网站 | 联系最紧密的网站 | 日志文件 | 用来了解网站最主要的流量从哪里来 |
| 繁忙时间段 | 网站网络服务器使用率最高的时刻 | 日志文件 | 用于计划网站升级和对网络进行维护、管理的依据 |
| 客户/服务错误类别 | 客户或服务器产生错误的详细情况 | 日志文件 | 很多错误不一定是真正的错误 |
| 用户浏览器和操作系统 | 用户使用浏览器及操作系统的类型和版本 | 日志文件 | 用来了解设计的技术规格和实际情况间的差距，可以对未来的规划提供依据 |
| 用户地域分布 | 地域分布统计 | 日志文件 | 基于国家域名后缀的统计不是很可靠 |
| 网站响应时间 | 页面下载时间和数据搜索时间 | 性能监控软件 | 是一个很好的基准，网站随着流量的增加，可以了解其性能变化 |
| 服务器正常运行时间 | 网站可用时间的百分比 | 性能监控软件 | 应该尽量使这个指标接近100%，对于那些以广告为主要业务的网站更重要 |
| 注册用户数 | 带有用户个人详细信息的注册用户总数 | 数据库 | 要求用户使用时需注册和登录，操作更麻烦 |
| 个性化应用 | 专家报告功能 | 数据库分析 | 包括分析购买趋势、销售转化率、用户生活方式等。 |