业务需求：

表示组织或客户高层的目标。

用户需求：

用户希望系统必须能完成的任务。

功能需求：

开发人员必须在产品中实现的软件功能，用户利用这些功能来完成任务，满足业务需求。

需求分析之所以重要,就因为：

它具有决策性,方向性,策略性的作用,

它在软件开发的过程中，具有举足轻重的地位.在一个大型软件系统的开发中,它的作用要远远大于程序设计. 因此，一定要对需求分析具有足够的重视.

需求分析的任务就是

解决“做什么”的问题,就是要全面地理解用户的各项要求。并准确地表达所接受的用户需求.

什么是需求分析

分析软件用户的需求,细致的进行调查,把用户“做什么”的要求，最终转换为一个完全的、精细的软件逻辑模型。并写出软件的需求规格说明。准确地表达用户的要求.

需求分析阶段的工作,可以分为四个方面:

问题识别 分析与综合 制订规格说明 评审.

不好的需求：

用户参与不足，用户需求扩展，有歧义的需求，镀金问题，过于抽象的需求，忽略了某类用户，不准确的计划。

优秀需求：

完整性，正确性，可行性，必要性，有优先次序，无歧义，可验证性。

需求规格说明书的特点：

完整性，一致性，可修改性，可跟踪性。

需求工程推荐方法

知识：

需求管理：

项目管理：

需求获取：

需求分析：

编写规格说明书：

需求验证：

知识：

培训需求分析人员

培训用户代表和管理人员

对开发人员进行应用领域的培训

创建项目术语表

需求管理：

确定变更控制过程

建立变更控制委员会

进行变更影响分析

编写需求文档的基准版本和控制版本

维护变更历史记录

跟踪需求状态

衡量需求稳定性

使用需求管理工具

创建需求跟踪矩阵

项目管理：

选择合适的生存周期

确定项目的基本计划

发生需求变更时，协商项目约定

编写文档和管理与需求相关的风险

跟踪需求工程所耗的工作量

积累经验

需求获取：

编写项目视图与范围

确定需求开发过程

用户群分类

选择产品代表

建立典型的用户核心队伍

让用户代表确定使用实例

召开应用程序开发联系会议

分析用户工作流程

确定质量属性和其它非功能需求

检查问题报告，完善需求

跨项目重用需求

需求分析：

绘制关联图

创建用户接口原型

分析需求可行性

确定需求优先级

为需求建立模型

编写数据字典

使用质量功能调配（QFD）

编写规格说明书：

采用软件需求规格说明模版

指明需求来源

为每项需求注上标号

记录业务规范

创建需求跟踪能力矩阵

需求验证：

审查需求文档

编写测试用例，测试需求

确定合格的标准

需求分析员：

—— 是对软件项目设计的需求进行收

集、分析、记录和验证等工作的

主要承担者。

—— 是用户群体和软件开发团队之间

进行需求沟通的桥梁。

——是收集和传播的中心角色。

1）定义业务需求

2）确定项目承担者和用户类别

3）获取需求

4）分析需求

5）编制需求规格说明书

6）为需求建模

7）主持对需求的验证

8）引导对需求的优先级划分

9）管理需求

项目视图和范围文档的模板：

a. 业务需求

a.1 背景

a.2 业务机遇

a.3 业务目标

a.4 客户或市场需求

a.5 提供给客户的价值

a.6 业务风险

b. 项目视图的解决方案

b.1 项目视图陈述

b.2 主要特性

b.3 假设和依赖环境

c. 范围和局限性

c.1 首次发行的范围

c.2 随后发行的范围

c.3 局限性和专用性

d. 业务环境

d.1 客户概貌

d.2 项目优先级

e. 产品成功的因素

a. 业务需求

a.1 背景

a.2 业务机遇

a.3 业务目标

a.4 客户或市场需求

a.5 提供给客户的价值

a.6 业务风险

b. 项目视图的解决方案

b.1 项目视图陈述

b.2 主要特性

b.3 假设和依赖环境

c. 范围和局限性

c.1 首次发行的范围

c.2 随后发行的范围

c.3 局限性和专用性

d. 业务环境

d.1 客户概貌

d.2 项目优先级

e. 产品成功的因素

软件需求的典型来源：

1. 访问并与有潜力的用户探讨

2. 把对目前的或竞争产品的描述，写成文档

3. 系统需求规格说明

4. 对当前系统的问题分析，并增强要求

5. 市场调查和用户问卷调查

6. 观察正在工作的用户

7. 用户工作的情景分析

8. 事件和响应

业务规则：

是对业务的某个方面进行定义或约束的语句。

需求开发的最终成果是：

—— 客户和开发小组对将要开发的产品，达成一致协议。

—— 这一协议综合了业务需求、用户需求和软件功能需求。

—— 而使用用例文档，则只包含了用户需求。

—— 必须应用文档把他们表示出来。

编写软件需求规格说明，有三种方法：

用好的结构化和自然语言编写文本型文档

建立图形化模型方法

---- 模型：可以描绘转换过程、系统状态和它们之间的变化、数据关系、逻辑流或对象类和它们的关系。

编写形式化规格说明

---- 这可以通过使用数学上精确的形式化逻辑语言来定义需求。

数据字典

定义应用程序中，使用的所有数据元素和结构的含义、类型、数据大小、格式、度量单位、精度以及允许取值范围的共享仓库。

数据字典

----- 可以把不同的需求文档和分析模型紧密结合在一起

—— 如果所有的开发人员在数据字典上取得一致意见，那么就可以缓和集成性问题。

—— 而并不是在每个需求出现的地方定义每一个数据项。

—— 数据字典的维护独立于软件需求规格说明，并且在产品的开发和维护的任何阶段，各个风险承担者都可以访问数据字典。

数据流图：是结构化系统分析的基本工具：确定了系统的转化过程、系统所操纵的数据或物质的存储：可以在一个抽象的广泛范围内表示系统

数据流图：是分层次的，高层数据流图提供一个整体的统览，是对软件需求规格说明的精确、详细叙述的补充。

数据流图：描述了功能需求怎样和使用户相结合

数据流图：反馈的信息，有助于理解所探讨的任务流，进行提炼加工。

实体联系图：描绘了系统的数据关系

---- 实体联系图：有助于对业务或系统数据组成的理解和交互。

---- 用一个实体联系图和一个数据字典，来记录数据关系，可以为新的业务过程，提供一个数据组成的概念性框架

状态转换图：为状态提供了一个简洁、完整、无二义性的表示。

状态转换图：表示处理结果可能的状态转换

---- 对于软件系统中只能存在于特定状态的那一部分，可以使用状态转换图来建模。

状态转换图：有助于开发者理解系统的预期行为。

---- 测试者：可以从转换路径的状态转换图中获得测试用例。

---- 用户：只要稍微学一些符号就可以读懂状态转换图。

对话图（dialog map）：一种状态转换图

对话图在较高的抽象层次上表示用户界面的设计，它展示了系统的对话元素及这些元素之间的导航连接，但没有展示详细的屏幕设计。

在对话图中将每个对话元素表示为一个状态（用矩形框表示），将每个允许的导航选项表示为一个转换（用箭头表示）。触发用户界面导航的条件表示为转换箭头上的文本标签。

对话图是表示用例中所描述的参与者与系统之间的交互的很好的方法。

软件质量属性或质量引述是系统非功能性需求的一部分。

非功能需求(none-functional requirements):描述系统展现给用户的行为和执行的操作等。包括:产品必须遵循的标准、规范和合约,外部界面的具体细节性能要求设计或实现的约束条件

质量属性(quality attribute):通过多种角度对产品的特点进行描述，从而反映产品性能。

可用性有效性灵活性完整性互操作性可靠性健壮性易用性可维护性可移植性可重用性可测试性

一个软件原型：

——是所提出的新产品的部分实现

使用原型有三个主要目的：

 明确并完善需求

——原型，作为一种需求工具，它初步实现所理解的系统的一部分。

探索设计选择方案

——原型，作为一种设计工具，用它可以探索不同的用户界面技术，使系统达到最佳的可用性。

发展为最终的产品

——原型，作为一种构造工具，是产品最初子集的完整功能实现。

水平原型

——也叫“行为原型”或“演示性模型”

——水平原型，显示出用户界面的正面像，但是它仅包含少量的功能，并没有真正实现所有的功能。

——水平原型，可以使用户判断是否有遗漏、 错误或不必要的功能。

——可以使用不同的屏幕设计工具或甚至使用纸和铅笔来建立水平原型。

垂直原型：

——也叫“结构化原型”或“概念性模型”

——它实现了一部分应用功能

——当不能确信所提出的构造软件的方法是否完善

——或者当需要优化算法，评价一个数据库的图表或测试临界时间需求时。

抛弃型原型：

——在原型达到预期目的以后，将它抛弃，所以，可以花最小的代价，尽快地建立该原型。

——抛弃型原型，忽略了很多具体的软件构造技术。

——不能将抛弃型原型中的代码，移植到产品系统中。

——否则，将在软件生存期中遭遇种种麻烦。

——当遇到需求中的不确定性、二义性、不完整性或含糊性时。

——最合适的方法，是建立抛弃型原型。

——原型，可帮助用户和开发者。

——想象如何实现需求和发现需求中的漏洞。

——原型，还可以使用户判断出需求是否可以完成必要的业务过程。

进化型原型：

——在已经清楚地定义了需求 的情况下，进化型原型为产品提供了坚实的构造基础。

——进化式模型，一开始就必须具有健壮性和产品质量级的代码。

——建立进化型原型比建立抛弃型原型，所花的时间要多。

——一个进化型原型必须设计为易于升级和优化的

——从测试和首次使用中获得的信息，将引起下一次软件原型的更新，正是这样不断增长并更新，使软件才能从一系列进化型原型，发展为实现最终完整的产品。

——这种原型提供了快速获得有用功能的方法。

书面原型：

——即，模仿计算机的人，就会把关于显示方面的纸张和索引卡给用户看。

用户就可以判断这些界面是否是所期望的响应，并且还可以判断所显示的项是否正确。

如果有错误，只要用一张新纸或索引卡，重画一张就可以了。

不管你建立原型的工具多么高效，在纸张上勾画界面是最快的。

软件原型法：

——提供了一套强有力的技术

——可以缩短开发进度

——增加用户的满意程度

——生产出高质量的产品

——可以减少需求错误和用户界面的缺陷。

设定优先级:

——有助于项目经理解决冲突、安排阶段性交付，并且，做出必要的取舍。

——下面将讨论设定需求优级的重要性

——并且提出一个基于价值、费用和风险的设定优先级方案

优先级 = 价 值 %

（费用%×费用权值）+（风险%×风险权值）

需求确认：

是指开发方和客户方共同对《产品需求规格说明书》进行评审，双方对需求达成共识后作出承诺。需求确认包含两个重要工作：“需求评审”和“需求承诺”。

需求确认，确保了需求规格说明的良好特性。是对渐增型软件需求规格说明的反复评审，将贯穿着反复获取需求、分析和编写规格说明的整个过程。可以减少返工，并加快系统测试，从而真正缩短了开发时间。

需求确认的主要活动有以下几方面：

● 软件需求规格说明是否正确描述了预期的系统行为和特征？

● 从系统需求或其它来源中得到软件需求的正确性。

● 需求是否是完整的和高质量的？

● 所有对需求的看法是否一致？

● 需求为继续进行产品设计、构造和测试是否提供了足够的基础？

管理需求，是因为想让项目获得成功。满足项目需求，即为成功打下了基础。若无法管理需求，达到目标的几率就会降低。

管理需求，避免失败就是一个很充分的理由。提高项目的成功率和需求管理所带来的其他好处同样也是理由。

需求管理：

——是一种获取、组织并记录系统需求的系统化方案，以及一个使客户与项目团队对不断变更的需求达成并保持一致的过程。

需求管理：

——是对所有相关活动的规划和控制。

需求管理：

——准确地强调了追踪变更以保持涉众与项目团队之间共识的重要性

需求管理：

——包括：在工程进展过程中维持需求约定集成性和精确性的所有活动。

需求管理强调：

• 控制对需求基线的变动。

• 保持项目计划与需求一致。

• 控制单个需求和需求文档的版本情况。

• 管理需求和联系链之间的联系或管理单个需求和其它项目之间的依赖关系。

• 跟踪基线中需求的状态。

需求管理的困难性

需求不总是显而易见的，而且它可来自各个方面。

需求并不总是能容易用文字明白无误地表达。

存在不同种类的需求，其详细程度各不相同。

如果不加以控制，需求的数量将难以管理。

需求之间相互关联，而且需求也和软件工程流程中的其他可交付工件有关。

需求有唯一的特征或特征值。例如，它们的重要性和容易满足的程度都各不相同。

需求涉及众多相关方面。

需求会变更。

需求管理的主要活动：

确定需求变更控制过程

建立变更控制委员会

进行需求变更影响分析

跟踪所有受需求变更影响的工作产品

建立需求基准版本和需求控制版本文档。

维护需求变更的历史记录。

跟踪每项需求的状态

衡量需求稳定性，记录基准需求的数量和每周或每月的变更（添加、修改、删除）数量。

使用需求管理工具。

风 险

—— 是指可能给项目的成功，带来威胁或损失的情况

风险管理 ：

—— 就是在风险给项目带来损失之前，指明、评估并对风险加以控制。

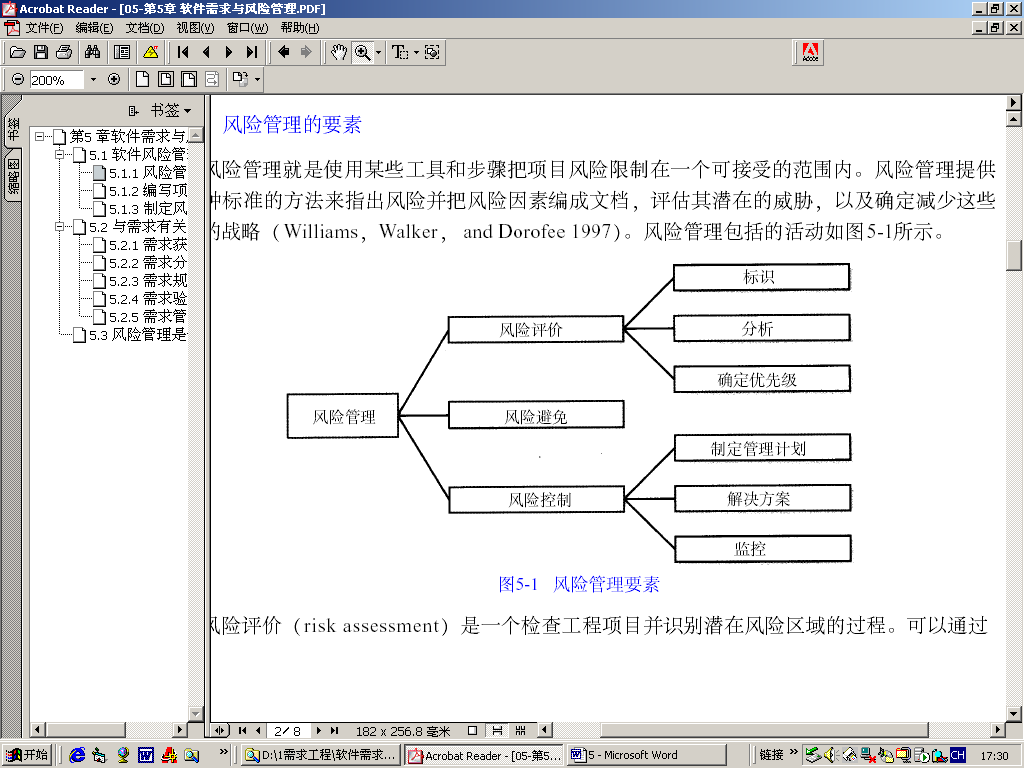
典型的需求风险：

—— 包括:对需求的误解、不恰当的用户参与、不确定、 随意变更项目范围和目标以及持续变更需求提供了一种标准的方法，来指出风险，并把风险因素编成文档。

——风险管理

评估其潜在的威胁

确定控制和减少风险的战略



运用风险管理，提高对造成项目损

失条件的警惕

如果用户一直没有参与，风险危害

值，将会扩大，以至影响项目的成功。