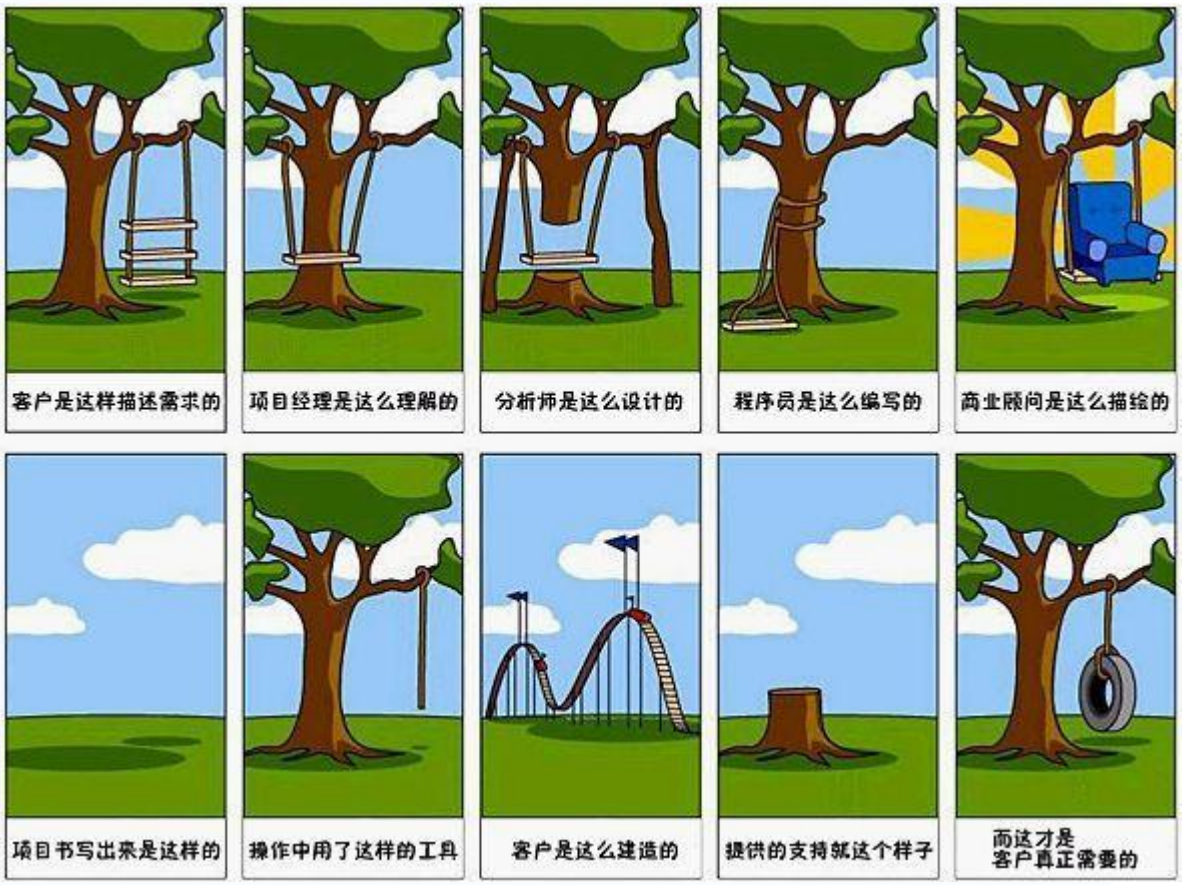
一. 需求评审与需求测试

在软件开发过程中，需求分析是最开始的工作，需求分析如果做得不够详细或者是偏离用户需求的话，往往会给项目带来灭绝性的灾难。因此如何保证需求分析的正确性，不偏离用户的需求就成了决定软件项目成败的关键。

需求工程师取得用户的显性需求后，要仔细的分析用户到底要求软件实现什么功能，用户的表达和需求工程师的理解有时间并不会一致，这样会导致用户所想的和需求说明书上所描述的有偏差。并且需求工程师取得用户的需求后必须做仔细透彻的分析，有时候用户的需求并不一定正确，可能是用户忽然的想法，并不可行。如果需求工程师不能对用户提出的需求进行判断的话，可能辛辛苦苦的实现了用户需求，结果被用户自己否决掉。用户绝对不会将责任揽到自己身上，他们只会说“你们是专家，怎么能怪我呢？”。

网上有一幅漫画形象地描述了信息在传递过程中产生的误差。



需求分析师是项目中直接与客户接触的人，需求做的好不好决定项目成败，因此对于需求规格说明书的正确性必须进行彻底的验证，将错误在开工前就消灭。

通常有两种手段来检查需求的正确性，分别是需求评审和需求测试。

1、 需求评审

需求评审可以分为正式评审与非正式评审，在需求规格说明书完成后，需求组必须自己对需求做评审。如果需求组递交的需求规格说明书在指导后面的工作的时候出现很明显的错误，我想拿高工资的需求分析人员是无法向老板交差的。为了需求分析人员的名誉，他们自己会对自己提交的内容进行审核，直到他们认为自己的工作成果足够好，才会将需求规格说明书提交给正式评审组。

正式评审组的成员一般由公司内经验最丰富，技术最牛的人（技术总监）来担任，当然参加评审的人中间还应该有项目经理、QA 人员、测试人员、架构师，他们仔细阅读需求规格说明书，并针对自己将要开展的工作内容进行检查，并提出问题

正式评审是最后一关，如果正式评审通过了，将进入系统设计阶段，如果在系统设计阶段再跨里程碑来修改需求的话，所花费的代价将大大增加。因此正式评审将是一个“鸡蛋里挑骨头”的过程，只有所有的人都认为需求已经没有什么可挑剔评审才能通过。

2、 需求测试

可以认为需求评审也属于需求测试范围，但是这里提的需求测试和评审不同，它是测部门来测试需求是否符合用户的要求。显然这是有难度的，传统的测试工作都是从单元测试开始，编码之前全部做得都是计划性工作。测试人员对需求分析进行测试？那么前提条件是测试人员必须熟悉需求分析，这对测试人员的要求提高了。将需求测试人员作为测试人员中的特殊种类来培养，能够对需求是否正确进行检查，这样就能够在需求阶段就引入测试。当然需求测试人员可以是经过培训的需求分析人员，但是他必须脱离需求组，加入测试部门，这样才能保证测试不是自己人测自己，以保证测试的效果。

需求测试不等同于后面阶段集成测试或者系统测试，后面的测试都是软件已经编写完成的条件下，判断软件是否会出错。而需求测试，只是验证需求是否真的是用户的。对于需求的功能测试，可以用 RAD 工具建立界面原型，用户通过原型的操作来确定是否需求跟他的期望相同。对于那些用户不合理的需求，测试人员要

能够分辨出来，并跟用户进行核对，确定用户的真实需求。可以说需求测试是需求测试人员和用户共同来执行的。

之所以将需求测试和需求评审并行进行，是因为需求评审是项目的各方干系人共同进行的检查工作，评审工作关注的焦点是分散的，很难将偏离用户的需求检查出来，并且涉及的人很多，因此不可能耗费太长时间。而需求测试执行的时间可以比评审时间长，有专门的关注方面，能够检查出不合理的需求分析，在项目前期进行错误纠正，往往比实现后纠正要节约几百甚至几千倍的成本。

二．测试人员眼中的需求分析

1 测试人员在需求分析过程中的职责测试人员的职责是对需求分析中产生的各种文档进行评审，确定其质量达到要求（尽量减少错误），提出修改意见，为今后的跟踪和测试打下基础。同时希望通过此文来丰富我们现在《文档审查检查表》的内容和可操作性

2 需求分析阶段的测试 1 首先检查所有的文档（或文档相关方面）是否齐全，包括项目范围、用户分析、实体和实体关系图、使用实例、软件需求规格说明文档。

2 检查所有的文档是否按照正确的模板来写？

3 数据字典和使用实例是否在项目范围之内？

4 软件需求规格说明文档是否与数据字典和使用实例相符，内容是否在项目范围之内？

5 项目范围是否清楚？
读完一份项目范围书后应该能清楚的标明哪些功能要作，哪些功能不作。

6 项目范围参考文档是否列清楚？
能从参考文档中查到对应的依据。

7 是否做过用户分析？
是否有对用户类型的描述，是否与对应的用户作过讨论，讨论的结果是否包含在使用实例中。

8 是否有实体联系图？
是否对用户数据进行详细的分类和统计，并对对应数据作详细的描述，有没有收集用户提到的相关报表的样式？

2.1 使用实例文档审查清单

1 使用实例是否是独立的分散任务？

2 使用实例的目标或价值度量是否明确？

3 使用实例给操作者带来的益处是否明确？

4 使用实例是否处于抽象级别上，而不具有详细的情节？

5 使用实例中是否不包含设计和实现的细节？

6 是否记录了所有可能的可选过程？

7 是否记录了所有可能的例外条件？

8 是否存在一些普通的动作序列可以分解成独立的使用实例？

9 是否简明书写、无二义性和完整地记录了每个过程的对话？

10 使用实例中的每个操作和步骤是否都与所执行的任务相关？

11 使用实例中定义的每个过程是否都可行？

12 使用实例中定义的每个过程是否都可验证？

13 使用实例是否有唯一的编号？

2.2 软件需求规格说明的审查清单 1 组织和完整性（对大项目尤其重要，因为大项目分割的模块很多，容易丢失一些功能）

2 正确性

3 质量属性

4 可跟踪性

5 特殊的问题

三. 需求说明的特征：

1. 完整性
每一项需求都必须将所要实现的功能描述清楚，以使开发人员获得设计和实现这些功能所需的所有必要信息。
2. 正确性
每一项需求都必须准确地陈述其要开发的功能。做出正确判断的参考是需求的来源，如用户或高层的系统需求规格说明。若软件需求与对应的系统需求相抵触则是不正确的。只有用户代表才能确定用户需求的正确性，这就是一定要有用户的积极参与的原因。没有用户参与的需求评审将导致此类说法："那些毫无意义，这些才很可能是他们所要想的。"其实这完全是评审者凭空猜测。
3. 可行性
每一项需求都必须是在已知系统和环境的权能和限制范围内可以实施的。为避免不可行的需求，最好在获取（ elicitation）需求(收集需求)过程中始终有一位软件工程小组的组员与需求分析人员或考虑市场的人员在一起工作，由他负责检查技术可行性。
4. 必要性
每一项需求都应把客户真正所需要的和最终系统所需遵从的标准记录下来。"必要性"也可以理解为每项需求都是用来授权你编写文档的"根源"。要使每项需求都能回溯至某项客户的输入，如使用实例或别的来源。
5. 划分优先级
给每项需求、特性或使用实例分配一个实施优先级以指明它在特定产品中所占的分量。如果把所有的需求都看作同样重要，那么项目管理者在开发或节省预算或调度中就丧失控制自由度。
6. 无二义性
对所有需求说明的读者都只能有一个明确统一的解释，由于自然语言极易导致二义性，所以尽量把每项需求用简洁明了的用户性的语言表达出来。避免二义性的有效方法包括对需
求文档的正规审查，编写测试用例，开发原型以及设计特定的方案脚本。
7. 可验证性
检查一下每项需求是否能通过设计测试用例或其它的验证方法，如用演示、检测等来确定产品是否确实按需求实现了。如果需求不可验证，则确定其实施是否正确就成为主观臆断，而非客观分析了。一份前后矛盾，不可行或有二义性的需求也是不可验证的。

四. 需求阶段的测试

首先，测试用例和测试工作本身是不断完善的，在开发过程的初期，可以认为是需求阶段，或者没有规范需求工作的设计阶段。如果有一个比较明确的需求文档，可以在这个阶段检查完了需求文档以后开始设计测试用例。这里，
对于需求文档的检查主要是两个方面：

1.检查需求文档描述的正确性，测试人员要对于真实的系统所涉及的业务非常熟悉，比如一个简单的财务软件，那么测试人员本身就要对会计工作熟悉，财务制度熟悉，在检查需求
文档的时候不要迷信所谓的"都是用户真实的需求"，这里存在两个问题，一是用户是否真的能正确地描述自己的需求，二是需求人员是否真的能正确地理解需求。另外，还有一个用户的嘘气是否符合行业规范的问题，如果不符合，那么是否要确认——这里存在一个隐患，用户可能会在开发的后期突然要求他们自己要走行业规范，让你的需求变动，所以要事先明确好。

2.检查需求文档描述的准确性。主要是考虑文档中是否存在描述的模糊的地方，对于自己不清楚的问题一定要明确。这个时候是要保证需求的可测试性——我得意思是说保证需求是可以完全为测试工作服务的。

那么在检查完了需求之后，就可以开始设计测试用例了，在这个阶段因为没有开始设计工作，所以对于测试用例的考虑不能仅仅从界面出发——虽然RUP 中对于用例的要求有这一项。因而测试用例的设计应该从业务角度出发，从实际业务出发来设计测试用例。当然，在测试用例的描述时，要尽量考虑怎样同应用程序脱离开而仍然具有有效性。

当然，这个阶段所实现的测试用例是不过完善的，只能涵盖某些内容，但是我认为这些用例不仅仅全部都是功能测试用例，而且在整个项目中都将有效。不过，当缺少需求文档时，那就要发挥测试人员自己的能动性了，要主动的工作，而不是被动的等待。要自己尝试着去熟悉实际业务，要尽量通过自己所能想到的方法来开展工作。
相信学过马克思哲学的朋友都知道什么是客观性和主观能动性吧。

当然，在设计阶段和最后的编码阶段，都还可以继续添加、修改或者剔除掉部分测试用例，使之更加完善。

一般软件测试都是编码阶段开始，而在现代软件测试中都是推崇使用测试驱动的方式进行软件开发的，以减少为修正错误而付出有时候甚至是惨重的代价。我们应该在需求阶段就重视和开始进行测试，这是因为系统的大多数的重要决定都是在需求阶段做出的，一旦一个即使是微小的错误在需求阶段产生，并延续到编码和集成测试结束之后才发现，其代价就是要重新确定需求再进行设计再编码

等。据估计，再验收测试阶段发现的错误的修复费用比能在需求阶段发现的错误要付出的多10倍。所以说，需求阶段的测试是必要的。

每个测试阶段都应该有负责人或者是负责组织，这里的负责人或负责组织不一定是负责着测试的任务，而是对测试的结果进行确认和经过确认后负有这个测试结果的责任。需求阶段的测试负责人应该是最好是用户，用户可以看见经过测试之后记录下来详细的需求变化做出决定，这样可以使用户更加明确了需求的内容。所以说服用户积极参加和承担需求测试责任是需求测试成功的第一步，也是系统开发顺利进行的第一步。

需求阶段的测试是在需求完成之后评审需求之前这段时间中进行的，应该由管理部门来提出测试要求，测试小组执行并向管理部门提交有关问题的可选方案及最佳方案等报告。需求阶段的目标一般有如下5点：

1） 需求已经文档化并且完整地描述了用户的需求期望；

2） 业务问题已经解决了；

3） 系统要求的性能价格比已经经过讨论研究并证实是合理的；

4） 已经明确了控制需求的方向（“控制”在这里的意思是包括所有的机制、方法及应用程序中用于保证其功能与期望相一致的过程。）；

5）确定需求阶段所有解决问题的方法和途径是有效和正确的。在测试过程中，一定要以测试目标为中心，去查找不充分的不完整的需求，直到所有的不充分的不完整的需求得到解决并经过再测试OK，需求阶段的测试才可以结束。在进行需求阶段的测试之前必须了解“测试因素”的概念，掌握一些测试技术和一些测试工具。为了了解“测试因素”

其实测试人员在需求阶段介入测试要求是非常高的，静态测试作为这一期间的主要测试行为，一般按照正式技术复审会议展开，一般情况下测试人员比较关注需求的整体性检查，可使用的检查表为：

01	**整体检查**	需求是否完整？			
02		所有的需求是否均可唯一确定？			
03		所有的需求的分级是否清晰而适当？			
04		需求是否具有一致性？（即，内部无矛盾）			
05		需求组合是否充分地提出了所有适当的例外情况？			
06		需求组合是否充分地提出了边界情况？			
07		需求是否可行？（即，该需求组合有解决方案）			
08		需求可否用已知的约束来实现？			
09		需求是否足够？（即，可以把			

		它送到一个规范的开发组织，并有一个生产出所需要产品的合理的可能性）			
10		反面的需求明确地规定了吗?			
11		这些最简单的需求组合达到了相关干系人的要求了吗?			
12		所有到其它需求的交叉引用是否正确?			
13		功能性和非功能性的需求都考虑到了吗?			
14		本条需求是否明确和准确?			
15		有没有尽可能地用简单的表格来规定本条需求？			
16		本条需求能否被测试或被验证?			
17		本条需求是否正确?			
18		需求是否在范围内?（即，只要有一个需求遗漏了，这个系统就被认为是不完整的）			
19		需求是否很容易被更改?			
20		需求是否是用客户的语言和术语来撰写的?			
21		这个需求可被所有相关涉及人员接受吗？			
22		这个需求是否是对所有相关涉及人员的需要的描述,而不是一个解决方案?			
23		需求可被追踪吗?			
24		是否必需?			
25		如本需求有信息缺失，这些信息是否记录于待决定列表并指定了解答者和解答期限了吗?			

需求阶段测试人员发挥的实际效果不大，如果参与的测试人员本身具备很高的业务知识或者本身就是业务专家则另当别论，一般参与正式技术复审会议通常按照两种形式展开：

内部抽取检查：

由项目人员内部抽取一定数量的需求在内部展开检查，参与人员可以是 SA、业务专家、资深开发、测试或者 QA，展开后按照需求检查列表一项一项的过。内部抽取的定义是尽可能的发现问题，然后根据会议记录后的问题经过业务方证

明，或者取消或者增加，主要排除 镀金需求等一系列不可实现或者实现后不符合主业务路线的需求，效果一般注重会议后的疑问跟踪，通常会议上会当场指定需求跟踪人员，必须在指定时间内清除疑问，否则容易丢失。这种会议容易跑题，需要会议组织者经常纠偏。

指定需求检查：

需要邀请需求最终承载用户参与，并且用户具备一定的 IT 实施能力，这种形式的会议通常会当场否定部分需求，或者大量增加需求实现。效果最好，但是会议交流比较激烈。通常这种会议后会跟上内部抽取检查会议。

需求文档检查

会议前由 QA 展开，比较形式不注重业务本身检查

做好需求的测试，对于测试人员的要求可就不那么简单了。除了要具备测试的基本素质之外，这个人还要有比较深的行业背景。对于行业业务了如指掌，对软件的逻辑结构的设计和掌握程度也要水平比较高，否则就不能够知道需求分析，是不是把客户需要的东西全部正确的实现了。也不能预见所有分析的功能是否真的都可以实现。这样自然达不到评审的效果。

五. 软件需求测试

软件测试 V 模型要求我们在需求阶段就开始制定系统测试的计划，开始考虑系统测试的方法。但这还不是足够的。全面的质量管理要求我们在每个阶段都要进行验证和确认的过程。因此在需求阶段我们还需要对需求本身进行测试。这个测试是必要的，因为在许多失败的项目中，７０％～８５％的返工是由于需求方面的错误所导致的。并且因为需求的缘故而导致大量的返工，造成进度延迟、缺陷的发散，这是一件及其痛苦的事情。因此我们要求在项目的源头（需求）就开始测试。

这类测试更多的还只是静态手工方面的测试，当然也有一些自动化的工具，但这些工具会要求我们按照某个固定的格式进行需求的表述（例如形式化的方法），因此在适用性上会受到限制。通过静态手工方法进行需求测试中最常使用的手段是同行评审。

通过评审来测试需求

同行评审是业界公认的最有效的排错手段之一。我们在需求测试过程当中，使用最多的也是同行评审（Peer Review），尤其是正规检视（Inspection）。正规检视是由Michael Fagan 在I B M 制定出来的一种非常严格的评审过程。

需求评审的参与者当中，必须要有用户或用户代表参与，同时还需要包括项目的管理者，系统工程师和相关开发人员、测试人员、市场人员、维护人员等。在项目开始之初就应当确定不同级别、不同类型的评审必须要有哪些人员的参与，否则，评审可能会遗漏掉某些人员的意见，导致今后不同程度的返工。

好的需求应当具有的特点

一个良好的需求应当具有一下特点：

完整性：每一项需求都必须将所要实现的功能描述清楚，以使开发人员获得设计和实现这些功能所需的所有必要信息。

正确性：每一项需求都必须准确地陈述其要开发的功能。

一致性：一致性是指与其它软件需求或高层（系统，业务）需求不相矛盾。

可行性：每一项需求都必须是在已知系统和环境的权能和限制范围内可以实施的。

无二义性：对所有需求说明的读者都只能有一个明确统一的解释，由于自然语言极易导致二义性，所以尽量把每项需求用简洁明了的用户性的语言表达出来。

健壮性：需求的说明中是否对可能出现的异常进行了分析，并且对这些异常进行了容错处理。

必要性：“必要性”可以理解为每项需求都是用来授权你编写文档的“根源”。要使每项需求都能回溯至某项客户的输入，如Use Case或别的来源。

可测试性：每项需求都能通过设计测试用例或其它的验证方法来进行测试。

可修改性：每项需求只应在 S R S 中出现一次。这样更改时易于保持一致性。另外，使用目录表、索引和相互参照列表方法将使软件需求规格说明书更容易修改。

可跟踪性：应能在每项软件需求与它的根源和设计元素、源代码、测试用例之间建立起链接链，这种可跟踪性要求每项需求以一种结构化的，粒度好（f i n e - g r a i n e d ）的方式编写并单独标明，而不是大段大段的叙述。

另外应当对所有的需求分配优先级。如果把所有的需求都看作同样的重要，那么项目管理者在开发或节省预算或调度中就丧失控制自由度

通过用例设计来测试需求

我们从V模型上可以知道，验收测试是以系统需求为基础的，系统测试是以功能测试为基础。每个开发阶段的活动都与相应的测试活动是并行进行的。在需求阶段进行系统（验收）测试计划和设计，除了能指导最终的系统测试和验收测试执行外，其本身也是对需求的一个验证过程。

通过阅读软件需求规格说明书，通常很难想象在特定环境下的系统行为。以功能需求为基础或者从使用实例派生出来的测试用例可以使项目参与者看清系统的行为。虽然没有在运行系统上执行测试用例，但是设计测试用例的简单动作可以解释需求的许多问题。如果你在部分需求稳定时就开始开发测试用例，那么就可以及早发现问题并以较少的费用解决这些问题。

设计概念性测试用例可以发现需求的错误、二义性、不可测性、遗漏等方面问题，为了获得最大的效果，要求测试人员能够独立的去对需求进行思维，从一个不同于开发的角度上进行分析，这可能会是一个逆向的思维过程，在这个过程中，测试人员可能会设计出不同于需求的测试用例，而这最终可能会有两个解释：

1、需求不完整或者需求有错；

2、遗漏了测试用例或者测试用例本身有错误；

不管是哪种解释，最终肯定会提高整个系统的质量。但这个质量的获得是通过冗余的人员来完成的，即：开发人员在对系统需求进行进一步分析的时候，有一组独立的测试人员也在对系统需求进行独立的思维，并从中获取测试用例。尽管这两种思维可能会出现重复，但由于思维的方式不同，最终肯定会使得需求变得更清晰和更完善。

需求建模测试

需求的建模包括把需求转换成图形模型或形式化语言模型。需求的图形化分析模型包括数据流图（Data Flow Diagram，DFD）、实体关系图（Entity-Relationship Diagram，ERD）、状态转化图（State-Transition Diagram，STD）、对话图（Dialog Map）和类图（Class Diagram）。这些图形化模型一般都需要借助一定的 CASE（Computer-Aided Software Engineering）工具。这样就可以借助于自动化分析工具本身提供的检测手段来对需求进行测试，而这类检测主要可以提供描述上的完整性检查，需求项之间的不一致性检查等方面的功能。同时，使用这类自动分析工具有助于获得需求的质量特性，包括：有效性、一致性、可靠性、可存活性、可用性、正确性、可维护性、可测试性、可扩展性、可交互性、可重用性、可携带性等。

写测试需求主要为了什么呢？我们的项目中基本都有很细致的功能规格说明，还有其他一些相关的概念设计文档，我们总是会看到这些文档的最新版本。然而，我们的项目多为迭代方式进行，分很多版本提交，1.0.1、1.0.2 等等。在这些版本中，我们并不是每个版本都要测试全部的功能，往往是测试一部分。有的版本主要测业务流程，有的主要测性能。测试需求就是说明，这个版本需要测试哪些东西。

测试需求按照功能性、可靠性、易用性、性能、可维护性、可移植性来分类。同时也要按照优先级来分类，有的是必须测试通过的，有的可以协商。

除了说明我们需要测试的内容以外，测试需求还有一个重要的作用：辅助说明测试接受标准。比如某个版本的功能测试需求有 100 个功能点，其中 30 个必须实现，其他 70 个实现 60 个即可，假如每个功能点 1 分，那么，功能测试接受标准就是：总分 90 分以上并且 30 个重要功能点必须测试通过。假如没有达到这个接受标准（只有 85 分），我们就可以负责的说：测试不通过，不能发布。如果要发布，可以，变更项目计划和测试计划。

测试需求最好能细致到功能点的粒度，这样对项目量化管理非常有好处，而且，我认为这是应该在项目版本计划中进行说明的，如果项目计划中没有说的很详细，那我们的测试计划就要写的详细一些。

六. 测试人员在需求阶段应做哪些工作

相信很多人跟我一样有过这样的困惑，那就是在需求阶段我们测试人员应该做些什么？下面就将看过的一篇文章贴出来与大家一块分享。

首先，测试用例和测试工作本身是不断完善的，在开发过程的初期，可以认为是需求阶段，或者没有规范需求工作的设计阶段。如果有一个比较明确的需求文档，可以在这个阶段检查完了需求文档以后开始设计测试用例。这里，对于需求文档的检查主要是两个方面：

1.检查需求文档描述的正确性，愚以为测试人员要对于真实的系统所涉及的业务非常熟悉，比如一个简单的财务软件，那么测试人员本身就要对会计工作熟悉，财务制度熟悉，在检查需求

文档的时候不要迷信所谓的"都是用户真实的需求"，这里存在两个问题，一是用户是否真的能正确地描述自己的需求，二是需求人员是否真的能正确地理解需求。另外，还有一个用户的嘘气是否符合行业规范的问题，如果不符

合，那么是否要确认——这里存在一个隐患，用户可能会在开发的后期突然要求他们自己要走行业规范，让你的需求变动，所以要事先明确好。

2.检查需求文档描述的准确性。主要是考虑文档中是否存在描述的模糊的地方，对于自己不清楚的问题一定要明确。这个时候是要保证需求的可测试性——我得意思是说保证需求是可以完全为测试工作服务的。

那么在检查完了需求之后，就可以开始设计测试用例了，愚以为，在这个阶段因为没有开始设计工作，所以对于测试用例的考虑不能仅仅从界面出发——虽然 RUP 中对于用例的要求有这一项。因而测试用例的设计应该从业务角

度出发，从实际业务出发来设计测试用例。当然，在测试用例的描述时，要尽量考虑怎样同应用程序脱离开而仍然具有有效性。

当然，这个阶段所实现的测试用例是不过完善的，只能涵盖某些内容，但是我认为这些用例不仅仅全部都是功能测试用例，而且在整个项目中都将有效。

不过，当缺少需求文档时，那就要发挥测试人员自己的能动性了，要主动的工作，而不是被动的等待。要自己尝试着去熟悉实际业务，要尽量通过自己所能想到的方法来开展工作。

七. 如何确定测试需求

相信做测试的大部分朋友都有同感，在很多时候，我们都要面临没有文档或文档混乱，残缺的系统进行测试，而这时往往距离系统提交的时间只有几周了。那么，这时我们应该如何确定测试需求呢？先放下心中的愤懑，让我们讨论一下。

对一个系统而言，就算是你对它一无所知，仍然可以对界面和普通功能点进行测试，比如添加，删除等。但这对一个系统而言是远远不够的，是否实现它预期的需求才是测试工作的重心。如果不是因为时间的限制和这个系统的提交不需要你负责，你要坚持你自己的意见，而不是听从所谓的功能基本都实现了。

首先，去发现需求。这时你要把自己想象成一个侦探。所测试的这个系统现在它要满足什么需求，并以此建立这个时间的测试需求版本。实际上，系统的开发也许已经经过了几个版本了，每个版本都有实现的需求，有些很重要，有些次之。现在，你是要尽可能的全部找出它们。

第一步：收集数据。

1、阅读文档。如果你手头还是有一些文档，不管它的版本是多么老或者残缺不全都比没有要好，它总会给你提供一些需求的线索。也可从用户手中得到的一些用户文档或发行的媒体文件。一般用户在系统设计之初都会给当时的调研人员留下一些纸制或电子类的文档以表述他们的系统需求期望。如果你的项目经理已经遗失了这些资料，用户那里一般也有备份。但你需要一种婉转的方式去询问，以免给用户造成系统是否不能正常完成的印象。这些可以帮你对这个系统总体来说应该要满足那些重要的功能提供资料。

2、检查系统的体系结构。找一些对这个系统体系结构比较了解的人解释给你，并告诉你为什么系统是这样的体系结构。我们常常能从定义系统能力的最高层的限制中发现一些薄弱的连接。如果你本身就有一些系统体系结构方面的知识，这方面的工作应该不是很困难。

3、执行程序。检查程序的执行，在每一个页面中检查功能是否能够全部实现，在此可以找出它是否存在一个上一个结点和下一个结点。做上记录。然后根据页面的标题和节点之间的逻辑推理，可以大致判断出有几个业务流，它们涉及哪些相关节点。哪些页面之间存在数据联系。

4、询问开发者。这是一个比较头痛的问题，如果开发人员正忙于赶工期，他们对你的轻视可能导致你的询问很难有所成效。所以，你要尽量的提问得仔细，

问题最好用是这样或不是这样回答，以免因为他们不想对你解释太多而敷衍你。所以，你要尽可能的做好前面的工作，而不是依赖于开发人员。首先，你需要在项目经理那里得到开发人员所做的模块清单。哪些模块被几个开发人员同时操作，找出现在的负责人。然后，整理出自己所知道的模块信息，与开发人员交流。如果你对这件事感到委屈，那么至少有两个方面你需要加强，一是学会善于沟通，与开发人员相处融洽。二是努力学习，获得足够的知识与开发人员平等交谈。

5、询问项目经理

项目经理是一个能给你提供自己最大帮助的人，因为这个测试可能往往就是他要求的。你在那里尽可能的去找出有关系统的信息和资料。如果他不配合你，那么这个系统可能存在着某些巨大的问题导致也许根本无法交付。所以你需要花更多的时间来做工作。

当你找到这些需求资料时，你应该没有责备过任何一个人，因为在那个时候，他们做了他们的工作，你不能去要求人们以前应该做什么。现在你拥有的优势超过了项目中的其他人，你不需要被他们的假定迷惑。你可以更客观的去看待这个系统，并且比较它和设计的初衷有什么不同。

第二部，将资料转化为系统需求。

现在你有一些经过整理的材料，可以将它们转变为需求了。每个一个功能在不同的人那里可能有不同的说法。当你浓缩它们并定义它们这些说明的价值将变得非常清晰。我们想确定每一个说明的本质。这说明是否相关角色，特性或功能？

首先确定你能够安排的工作时间，根据时间按主要和次要安排需求。然后可按以下步骤来整理：

1、确定系统拥有多少角色（业务），他们负责什么样的工作，在系统中体现在那些模块中。然后画出这些角色的用例，或者他们涉及的业务。一般来说，系统很少有角色会全程做完一个业务流程。你可以先把每一个角色所做的每一个功能点列出来，然后再将它们放到一个完整的业务流中去。或者先画出整个的业务流，然后再分配给角色。最后你能得到一个完整的图，它包含整个系统所有业务流程，并且有哪些角色在某个节点上能够做哪些操作（拥有哪些权限）都非常的了然，这将是你测试的重点。

2、确定系统管理员的工作内容，系统管理员一般对系统进行初始设置，角色定义，业务流定义等重要操作。

3、确定系统的数据流动，包括系统的内部模块间数据流动（可结合系统角色图）和系统间的数据流动接口，在这些地方一般都比较容易出问题。

4、确定公共部分需要测试的需求。系统中有一些部分为很多角色所共同拥有并且不涉及业务流程。将这部分内容整理，一般来说这些内容只会涉及界面和普通功能的测试，如定义系统界面风格。

5、确定系统的使用情况。系统有多少用户，稳定运行要求至少多少时间，什么时候会出现系统使用高峰期，高峰期的特点。系统对未来几年内的用户和数据增长是否提供足够的可扩展空间。

6、系统的安全确定。系统运行的环境要求什么样的安全级别，有什么具体要求。如：访客是否能访问到只有用户才能访问的功能；一个角色是否越权访问他不能访问的角色。系统是否存在没有指向的链接页面作为后门（这个比较难）。等等。

7、使用该系统的用户可能的硬，软件环境，比如机器类型，操作系统，常用软件等。

8、其他系统要求确定的需求。

做完这些工作后，你可以开始设计你的测试用例了。也许仍然存在你不知道的情况，但你可以确定它没有表现在系统的可视范围内。

八. 需求分析的 20 条法则

需求分析的 20 条法则

对商业用户来说，他们后面是成百上千个供应商，前面是成千上万个消费顾客。怎样利用软件管理错综复杂的供应商和消费顾客，如何做好精细到一个小小调料包的进、销、调、存的商品流通工作，这些都是商业企业需要信息管理系统的理由。软件开发的意义也就在于此。而弄清商业用户如此复杂需求的真面目，正是

软件开发成功的关键所在。

经理："我们要建立一套完整的商业管理软件系统，包括商品的进、销、调、存管理，是总部-门店的连锁经营模式。通过通信手段门店自动订货，供应商自动结算，卖场通过扫条码实现销售，管理人员能够随时查询门店商品销售和库存情况。另外，我们也得为政府部门提供关于商品营运的报告。"

分析员："我已经明白这个项目的大体结构框架，这非常重要，但在制定计划之前，我们必须收集一些需求。"

经理觉得奇怪："我不是刚告诉你我的需求了吗？"

分析员："实际上，您只说明了整个项目的概念和目标。这些高层次的业务需求不足以提供开发的内容和时间。我需要与实际将要使用系统的业务人员进行讨论，然后才能真正明白达到业务目标所需功能和用户要求，了解清楚后，才可以发现哪些是现有组件即可实现的，哪些是需要开发的，这样可节省很多时间。"

经理："业务人员都在招商。他们非常忙，没有时间与你们详细讨论各种细节。你能不能说明一下你们现有的系统？"

分析员尽量解释从用户处收集需求的合理性："如果我们只是凭空猜想用户的要求，结果不会令人满意。我们只是软件开发人员，而不是采购专家、营运专家或是财务专家，我们并不真正明白您这个企业内部运营需要做些什么。我曾经尝试过，未真正明白这些问题就开始编码，结果没有人对产品满意。"

经理坚持道："行了，行了，我们没有那么多的时间。让我来告诉您我们的需求。实际上我也很忙。请马上开始开发，并随时将你们的进展情况告诉我。"

风险躲在需求的迷雾之后

以上我们看到的是某客户项目经理与系统开发小组的分析人员讨论业务需求。在项目开发中，所有的项目风险承担者都对需求分析阶段备感兴趣。这里所指的风险承担者包括客户方面的项目负责人和用户，开发方面的需求分析人员和项目管理者。这部分工作做得到位，能开发出很优秀的软件产品，同时也会令客户满意。若处理不好，则会导致误解、挫折、障碍以及潜在的质量和业务价值上的威胁。因此可见——需求分析奠定了软件工程和项目管理的基础。

拨开需求分析的迷雾

像这样的对话经常出现在软件开发的过程中。客户项目经理的需求对分析人员来讲，像"雾里看花"般模糊并令开发者感到困惑。那么，我们就拨开雾影，分析一下需求的具体内容：

·业务需求——反映了组织机构或客户对系统、产品高层次的目标要求，通常在项目定义与范围文档中予以说明。

·用户需求——描述了用户使用产品必须要完成的任务，这在使用实例或方案脚本中予以说明。

·功能需求——定义了开发人员必须实现的软件功能，使用户利用系统能够完成他们的任务，从而满足了业务需求。

·非功能性的需求——描述了系统展现给用户的行为和执行的操作等，它包括产品必须遵从的标准、规范和约束，操作界面的具体细节和构造上的限制。

·需求分析报告——报告所说明的功能需求充分描述了软件系统所应具有的外部行为。"需求分析报告"在开发、测试、质量保证、项目管理以及相关项目功能中起着重要作用。

前面提到的客户项目经理通常阐明产品的高层次概念和主要业务内容，为后继工作建立了一个指导性的框架。其他任何说明都应遵循"业务需求"的规定，然而"业务需求"并不能为开发人员提供开发所需的许多细节说明。

下一层次需求——用户需求，必须从使用产品的用户处收集。因此，这些用户构成了另一种软件客户，他们清楚要使用该产品完成什么任务和一些非功能性的特性需求。例如：程序的易用性、健壮性和可靠性，而这些特性将会使用户很好地接受具有该特点的软件产品。

经理层有时试图代替实际用户说话，但通常他们无法准确说明"用户需求"。用户需求来自产品的真正使用者，必须让实际用户参与到收集需求的过程中。如果不这样做，产品很可能会因缺乏足够的信息而遗留不少隐患。

在实际需求分析过程中，以上两种客户可能都觉得没有时间与需求分析人员讨论，有时客户还希望分析人员无须讨论和编写需求说明就能说出用户的需求。除非遇到的需求极为简单；否则不能这样做。如果您的组织希望软件成功，那么必须要花上数天时间来消除需求中模糊不清的地方和一些使开发者感到困惑的方面。

优秀的软件产品建立在优秀的需求基础之上，而优秀的需求源于客户与开发人员之间有效的交流和合作。只有双方参与者都明白自己需要什么、成功的合作需要什么时，才能建立起一种良好的合作关系。

由于项目的压力与日俱增，所有项目风险承担者有着一个共同目标，那就是大家都想开发出一个既能实现商业价值又能满足用户要求，还能使开发者感到满足的优秀软件产品。

客户的需求观

客户与开发人员交流需要好的方法。下面建议 20 条法则，客户和开发人员可以通过评审以下内容并达成共识。如果遇到分歧，将通过协商达成对各自义务的相互理解，以便减少以后的磨擦（如一方要求而另一方不愿意或不能够满足要求）。

1、 分析人员要使用符合客户语言习惯的表达

需求讨论集中于业务需求和任务，因此要使用术语。客户应将有关术语（例如：采价、印花商品等采购术语）教给分析人员，而客户不一定要懂得计算机行业的术语。

2、分析人员要了解客户的业务及目标

只有分析人员更好地了解客户的业务，才能使产品更好地满足需要。这将有助于开发人员设计出真正满足客户需要并达到期望的优秀软件。为帮助开发和分析人员，客户可以考虑邀请他们观察自己的工作流程。如果是切换新系统，那么开发和分析人员应使用一下目前的旧系统，有利于他们明白目前系统是怎样工作的，其流程情况以及可供改进之处。s

3、 分析人员必须编写软件需求报告

分析人员应将从客户那里获得的所有信息进行整理，以区分业务需求及规范、功能需求、质量目标、解决方法和其他信息。通过这些分析，客户就能得到一份"需求分析报告"，此份报告使开发人员和客户之间针对要开发的产品内容达成协议。报告应以一种客户认为易于翻阅和理解的方式组织编写。客户要评审此报告，以确保报告内容准确完整地表达其需求。一份高质量的"需求分析报告"有助于开发人员开发出真正需要的产品。

4、 要求得到需求工作结果的解释说明

分析人员可能采用了多种图表作为文字性"需求分析报告"的补充说明，因为工作图表能很清晰地描述出系统行为的某些方面，所以报告中各种图表有着极高的价值；虽然它们不太难于理解，但是客户可能对此并不熟悉，因此客户可以要求分析人员解释说明每个图表的作用、符号的意义和需求开发工作的结果，以及怎样检查图表有无错误及不一致等。

5、 开发人员要尊重客户的意见

如果用户与开发人员之间不能相互理解，那关于需求的讨论将会有障碍。共同合作能使大家"兼听则明"。参与需求开发过程的客户有权要求开发人员尊重他们并珍惜他们为项目成功所付出的时间，同样，客户也应对开发人员为项目成功这一共同目标所做出的努力表示尊重。

6、 开发人员要对需求及产品实施提出建议和解决方案

通常客户所说的"需求"已经是一种实际可行的实施方案，分析人员应尽力从这些解决方法中了解真正的业务需求，同时还应找出已有系统与当前业务不符之处，以确保产品不会无效或低效；在彻底弄清业务领域内的事情后，分析人员就能提出相当好的改进方法，有经验且有创造力的分析人员还能提出增加一些用户没有发现的很有价值的系统特性。

7、 描述产品使用特性

客户可以要求分析人员在实现功能需求的同时还注意软件的易用性，因为这些易用特性或质量属性能使客户更准确、高效地完成任务。例如：客户有时要求产品要"界面友好"或"健壮"或"高效率"，但对于开发人员来讲，太主观了并无实用价值。正确的做法是，分析人员通过询问和调查了解客户所要的"友好、健壮、高效所包含的具体特性，具体分析哪些特性对哪些特性有负面影响，在性能代价和所提出解决方案的预期利益之间做出权衡，以确保做出合理的取舍。

8、 允许重用已有的软件组件

需求通常有一定灵活性，分析人员可能发现已有的某个软件组件与客户描述的需求很相符，在这种情况下，分析人员应提供一些修改需求的选择以便开发人员能够降低新系统的开发成本和节省时间，而不必严格按原有的需求说明开发。所以说，如果想在产品中使用一些已有的商业常用组件，而它们并不完全适合您所需的特性，这时一定程度上的需求灵活性就显得极为重要了。

9、 要求对变更的代价提供真实可靠的评估

有时，人们面临更好、也更昂贵的方案时，会做出不同的选择。而这时，对需求变更的影响进行评估从而对业务决策提供帮助，是十分必要的。所以，客户有权利要求开发人员通过分析给出一个真实可信的评估，包括影响、成本和得失等。开发人员不能由于不想实施变更而随意夸大评估成本。

10、 获得满足客户功能和质量要求的系统

每个人都希望项目成功，但这不仅要求客户要清晰地告知开发人员关于系统"做什么"所需的所有信息，而且还要求开发人员能通过交流了解清楚取舍与限制，一定要明确说明您的假设和潜在的期望，否则，开发人员开发出的产品很可能无法让您满意。

11、 给分析人员讲解您的业务

分析人员要依靠客户讲解业务概念及术语，但客户不能指望分析人员会成为该领域的专家，而只能让他们明白您的问题和目标；不要期望分析人员能把握客户业务的细微潜在之处，他们可能不知道那些对于客户来说理所当然的"常识"。

12、 抽出时间清楚地说明并完善需求

客户很忙，但无论如何客户有必要抽出时间参与"头脑高峰会议"的讨论，接受采访或其他获取需求的活动。有些分析人员可能先明白了您的观点，而过后发现还需要您的讲解，这时请耐心对待一些需求和需求的精化工作过程中的反复，因为它是人们交流中很自然的现象，何况这对软件产品的成功极为重要。

13、 准确而详细地说明需求

编写一份清晰、准确的需求文档是很困难的。由于处理细节问题不但烦人而且耗时，因此很容易留下模糊不清的需求。但是在开发过程中，必须解决这种模糊性和不准确性，而客户恰恰是为解决这些问题作出决定的最佳人选，否则，就只好靠开发人员去正确猜测了。

在需求分析中暂时加上"待定"标志是个方法。用该标志可指明哪些是需要进一步讨论、分析或增加信息的地方，有时也可能因为某个特殊需求难以解决或没有人愿意处理它而标注上"待定"。客户要尽量将每项需求的内容都阐述清楚，以便分析人员能准确地将它们写进"软件需求报告"中去。如果客户一时不能准确表达，通常就要求用原型技术，通过原型开发，客户可以同开发人员一起反复修改，不断完善需求定义。

14、 及时作出决定

分析人员会要求客户作出一些选择和决定，这些决定包括来自多个用户提出的处理方法或在质量特性冲突和信息准确度中选择折衷方案等。有权作出决定的客户必须积极地对待这一切，尽快做处理，做决定，因为开发人员通常只有等客户做出决定才能行动，而这种等待会延误项目的进展。

15、 尊重开发人员的需求可行性及成本评估

所有的软件功能都有其成本。客户所希望的某些产品特性可能在技术上行不通，或者实现它要付出极高的代价，而某些需求试图达到在操作环境中不可能达到的性能，或试图得到一些根本得不到的数据。开发人员会对此作出负面的评价，客户应该尊重他们的意见。

16、 划分需求的优先级

绝大多数项目没有足够的时间或资源实现功能性的每个细节。决定哪些特性是必要的，哪些是重要的，是需求开发的主要部分，这只能由客户负责设定需求优先级，因为开发者不可能按照客户的观点决定需求优先级；开发人员将为您确定优先级提供有关每个需求的花费和风险的信息。

在时间和资源限制下，关于所需特性能否完成或完成多少应尊重开发人员的

意见。尽管没有人愿意看到自己所希望的需求在项目中未被实现，但毕竟是要面对现实，业务决策有时不得不依据优先级来缩小项目范围或延长工期，或增加资源，或在质量上寻找折衷。

17、 评审需求文档和原型

客户评审需求文档，是给分析人员带来反馈信息的一个机会。如果客户认为编写的"需求分析报告"不够准确，就有必要尽早告知分析人员并为改进提供建议。

更好的办法是先为产品开发一个原型。这样客户就能提供更有价值的反馈信息给开发人员，使他们更好地理解您的需求；原型并非是一个实际应用产品，但开发人员能将其转化、扩充成功能齐全的系统。

18、 需求变更要立即联系

不断的需求变更，会给在预定计划内完成的质量产品带来严重的不利影响。变更是不可避免的，但在开发周期中，变更越在晚期出现，其影响越大；变更不仅会导致代价极高的返工，而且工期将被延误，特别是在大体结构已完成后又需要增加新特性时。所以，一旦客户发现需要变更需求时，请立即通知分析人员。

19、 遵照开发小组处理需求变更的过程

为将变更带来的负面影响减少到最低限度，所有参与者必须遵照项目变更控制过程。这要求不放弃所有提出的变更，对每项要求的变更进行分析、综合考虑，最后做出合适的决策，以确定应将哪些变更引入项目中。

20、 尊重开发人员采用的需求分析过程

软件开发中最具挑战性的莫过于收集需求并确定其正确性，分析人员采用的方法有其合理性。也许客户认为收集需求的过程不太划算，但请相信花在需求开发上的时间是非常有价值的；如果您理解并支持分析人员为收集、编写需求文档和确保其质量所采用的技术，那么整个过程将会更为顺利。

"需求确认"意味着什么

在"需求分析报告"上签字确认，通常被认为是客户同意需求分析的标志行为，然而实际操作中，客户往往把"签字"看作是毫无意义的事情。"他们要我在需求文档的最后一行下面签名，于是我就签了，否则这些开发人员不开始编码。"

这种态度将带来麻烦，譬如客户想更改需求或对产品不满时就会说："不错，我是在需求分析报告上签了字，但我并没有时间去读完所有的内容，我是相信你们的，是你们非让我签字的。"

同样问题也会发生在仅把"签字确认"看作是完成任务的分析人员身上，一旦有需求变更出现，他便指着"需求分析报告"说："您已经在需求上签字了，所以这些就是我们所开发的，如果您想要别的什么，您应早些告诉我们。"

这两种态度都是不对的。因为不可能在项目的早期就了解所有的需求，而且毫无疑问地需求将会出现变更，在"需求分析报告"上签字确认是终止需求分析过程的正确方法，所以我们必须明白签字意味着什么。

对"需求分析报告"的签名是建立在一个需求协议的基线上，因此我们对签名应该这样理解："我同意这份需求文档表述了我们对项目软件需求的了解，进一步的变更可在此基线上通过项目定义的变更过程来进行。我知道变更可能会使我们重新协商成本、资源和项目阶段任务等事宜。"对需求分析达成一定的共识会使双方易于忍受将来的摩擦，这些摩擦来源于项目的改进和需求的误差或市场和业务的新要求等。

需求确认将迷雾拨散，显现需求的真面目，给初步的需求开发工作画上了双方都明确的句号，并有助于形成一个持续良好的客户与开发人员的关系，为项目的成功奠定了坚实的基础。