需求跟踪是指跟踪一个需求使用期限的全过程，需求跟踪包括编制每个需求同系统元素之间的联系文档，这些元素包括其他类型的需求，体系结构，其他设计部件，源代码模块，测试，帮助文件等。需求跟踪为我们提供了由需求到产品实现整个过程范围的明确查阅的能力。需求跟踪概述

需求跟踪的目的是建立与维护“需求－设计－编程－测试”之间的一致性，确保所有的工作成果符合用户需求。

主要方式

需求跟踪有两种方式：

（1）正向跟踪。检查《产品需求规格说明书》中的每个需求是否都能在后继工作成果中找到对应点。

（2）逆向跟踪。检查设计文档、代码、测试用例等工作成果是否都能在《产品需求规格说明书》中找到出处。

正向跟踪和逆向跟踪合称为“双向跟踪”。不论采用何种跟踪方式，都要建立与维护需求跟踪矩阵（即表格）。需求跟踪矩阵保存了需求与后继工作成果的对应关系。

## 主要内容

跟踪能力（联系）链（traceability link）使你能跟踪一个需求使用期限的全过程，即从需求源到实现的前后生存期。跟踪能力是优秀需求规格说明书的一个特征。为了实现可跟踪能力，必须统一地标识出每一个需求，以便能明确地进行查阅。

[](https://baike.baidu.com/pic/%E9%9C%80%E6%B1%82%E8%B7%9F%E8%B8%AA/4574819/0/8474fbdd62137fa876c638ef?fr=lemma&ct=single)

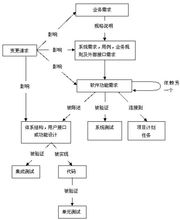
**图1：四类需求可跟踪能力**

图1说明了四类需求跟踪能力链。客户需求可向前追溯到需求，这样就能区分出开发过程中或开发结束后由于需求变更受到影响的需求。这也[确保](https://baike.baidu.com/item/%E7%A1%AE%E4%BF%9D" \t "_blank)了需求规格说明书包括所有客户需求。同样，可以从需求回溯相应的客户需求，确 认每个软件需求的源头。如果用使用实例的形式 来描述客户需求，图的上半部分就是使用实 例和功能性需求之间的跟踪情况。图的下半 部分指出：由于开发过程中系统需求转变为软件 需求、设计、编写等，所以通过定义单个需求和 特定的产品元素之间的（联系）链可从需求向前 追溯。这种联系链使你知道每个需求对应的产品 部件，从而确保产品部件满足每个需求。第四类 联系链是从产品部件回溯到需求，使你知道每个 部件存在的原因。绝大多数项目不包括与用户需 求直接相关的代码，但对于开发者却要知道为什 么写这一行代码。如果不能把设计元素、代码段 或测试回溯到一个需求，你可能有一个“画蛇添 足的程序”。然而，若这些孤立的元素表明了一 个正当的功能，则说明需求规格说明书漏掉了一项需求。

跟踪能力联系链记录了单个需求之间的父层、互连、依赖的关系。当某个需求变更（被删除或修改）后，这种信息能够确保正确的变更传播，并将相应的任务作出正确的调整。下图2说明了许多能在项目中定义的直接跟踪能力联系链。一个项目不必拥有所有种类的跟踪能力联系链，要根据具体的情况调整。

## 目的

在某种程度上，需求跟踪提供了一个表明与合同或说明一致的方法。更进一步，需求跟踪可以改善产品质量，降低维护成本，而且很容易实现重用。

[](https://baike.baidu.com/pic/%E9%9C%80%E6%B1%82%E8%B7%9F%E8%B8%AA/4574819/0/6a22e8242e9de2748644f9e2?fr=lemma&ct=single)

**图2：一些可能的需求跟踪能力联系链**

需求跟踪是个要求手工操作且[劳动强度](https://baike.baidu.com/item/%E5%8A%B3%E5%8A%A8%E5%BC%BA%E5%BA%A6)很大的任务，要求组织提供支持。随着系统开发的进行和维护的执行，要保持关联链信息与实际一致。跟踪能力信息一旦过时，可能再也不会重建它了。由于这些原因，应该正确使用需求跟踪能力。

下面是在项目中使用需求跟踪能力的一些好处： 审核（certification） 跟踪能力信息可以帮助审核确保所有需求被应用。 变更影响分析跟踪能力信息在增、删、改需求时可以确保不忽略每个受到影响的系统元素。 维护可靠的跟踪能力信息使得维护时能正确、完整地实施变更，从而提高[生产率](https://baike.baidu.com/item/%E7%94%9F%E4%BA%A7%E7%8E%87" \t "_blank)。要是一下子不能为整个系统建立跟踪能力信息，一次可以只建立一部分，再逐渐增加。从系统的一部分着手建立，先列表需求，然后记录跟踪能力链，再逐渐[拓展](https://baike.baidu.com/item/%E6%8B%93%E5%B1%95)。 项目跟踪在开发中，认真记录跟踪能力数据，就可以获得计划功能当前实现状态的记录。还未出现的联系链意味着没有相应的产品部件。 再设计（重新建造） 你可以列出传统系统中将要替换的功能，记录它们在新系统的需求和软件组件中的位置。通过定义跟踪能力信息链提供一种方法收集从一个现成系统的[反向工程](https://baike.baidu.com/item/%E5%8F%8D%E5%90%91%E5%B7%A5%E7%A8%8B)中所学到的方法。 重复利用跟踪信息可以帮助你在新系统中对相同的功能利用旧系统相关资源。例如：功能设计、[相关需求](https://baike.baidu.com/item/%E7%9B%B8%E5%85%B3%E9%9C%80%E6%B1%82)、[代码](https://baike.baidu.com/item/%E4%BB%A3%E7%A0%81)、测试等。 减小风险使部件互连关系文档化可减少由于一名关键成员离开项目带来的风险。 测试测试模块、需求、代码段之间的联系链可以在测试出错时指出最可能有问题的代码段。 　以上所述许多是长期利益，减少了整个产品生存期费用，但同时要注意到由于积累和管理跟踪能力信息增加了开发成本。这个问题应该这样来看，把增加的费用当作一项投资，这笔投资可以使你发布令人满意同时更容易维护的产品。尽管很难计算，但这笔投资在每一次修改、扩展或代替产品时都会有所体现。如果在开发工程中收集信息，定义跟踪能力联系链一点也不难，但要在整个系统完成后再实施代价确实很大。

[CMMI](https://baike.baidu.com/item/CMMI)要求具备需求跟踪能力。软件产品工程活动的[关键过程域](https://baike.baidu.com/item/%E5%85%B3%E9%94%AE%E8%BF%87%E7%A8%8B%E5%9F%9F" \t "_blank)有关于它的陈述，“在软件工作产品之间，维护一致性。工作产品包括软件计划，过程描述，分配需求，软件需求，软件设计，代码，测试计划，以及测试过程。”需求跟踪过程中还定义了一些关于一个组织如何处理需求跟踪能力的期望。

## 能力矩阵

表示需求和别的系统元素之间的联系链的最普遍方式是使用需求跟踪能力矩阵。下表展示了这种[矩阵](https://baike.baidu.com/item/%E7%9F%A9%E9%98%B5" \t "_blank)，这是一个“化学制品跟踪系统”实例的跟踪能力矩阵的一部分。这个表说明了每个功能性需求向后连接一个特定的使用实例，向前连接一个或多个设计、代码和测试元素。设计元素可以是模型中的对象，例如[数据流图](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E6%B5%81%E5%9B%BE" \t "_blank)、关系数据模型中的表单、或对象类。代码参考可以是类中的方法，源代码文件名、过程或函数。加上更多的列项就可以拓展到与其它工作产品的关联，例如在线帮助文档。包括越多的细节就越花时间，但同时很容易得到相关联的软件元素，在做变更影响分析和维护时就可以节省时间。

表1：一种需求跟踪能力矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例 | 功能[需求量](https://baike.baidu.com/item/%E9%9C%80%E6%B1%82%E9%87%8F" \t "_blank) | 设计元素 | 代码 | 测试实例 |
| UC-28  UC-29 | Catalog.query.sort  catalog.query.import | Class  Catalog  Class  catalog | Catalog.sort()  Catalog.import()  Catalog.validate() | Search.7  Search.8  Search.8  Search.13  Search.14 |

跟踪能力联系链可以定义各种系统元素类型间的一对一，一对多，多对多关系。表1中允许在一个表单元中填入几个元素来实现这些特征。这里是一些可能的分类：

一对一一个代码模块应用一个设计元素。 一对多多个测试实例验证一个功能需求。 多对多每个使用实例导致多个功能性需求，而一些功能性需求常拥有几个使用实例。 　手工创建需求跟踪能力矩阵是一个应该养成的习惯，即使对小项目也很有效。一旦确立使用实例基准，就准备在矩阵中添加每个使用实例演化成的功能性需求。随着软件设计、构造、测试开发的进展不断更新矩阵。例如，在实现某一功能需求后，你可以更新它在矩阵中的设计和代码单元，将需求状态设置为“已完成”。表示跟踪能力信息的另一个方法是通过矩阵的集合，矩阵定义了系统元素对间的联系链。例如：

一类需求与另一类需求之间。 同类中不同的需求之间。 一类需求与测试实例之间。 　可以使用这些矩阵定义需求间可能的不同联系，例如：指定/被指定、依赖于、衍生为以及限制/被限制。

下表2中说明了两维的跟踪能力矩阵。矩阵中绝大多数的单元是空的。每个单元指示相对应行与列之间的联系，可以使用不同的符号明确表示“追溯到”和“从.. 回溯”或其他联系。表2中使用一个箭头表示一个功能性需求是从一个使用实例追溯来的。这些矩阵相对于表16-6中的单跟踪能力表更容易被机器自动支持。

表2：反映使用实例与功能需求之间联系的需求跟踪能力矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能  需求 | 用例 |  |  |  |
| U C - 1 | U C - 2 | U C - 3 | U C - 4 |  |
| F R - 1 | √ |  |  |  |
| F R - 2 | √ |  |  |  |
| F R - 3 |  |  | √ |  |
| F R - 4 |  |  | √ |  |
| F R - 5 |  | √ |  | √ |
| F R - 6 |  |  | √ |  |

跟踪能力联系链无论谁有合适的信息都可以定义。下表3定义了一些典型的知识源，即关于不同种类源和目标对象间的联系链。定义了可以为[工程项目](https://baike.baidu.com/item/%E5%B7%A5%E7%A8%8B%E9%A1%B9%E7%9B%AE)提供每种跟踪能力信息的角色和个人。

表3：跟踪能力联系链可能的信息源

|  |  |  |
| --- | --- | --- |
| 链的源对象种类 | 链的目的对象种类 | 信息源 |
| 系统需求  用例  功能性需求  功能性需求  功能性需求  设计元素  功能性需求 | 软件需求  功能性需求  功能性需求  软件体系结构元素  其他设计元素  代码  测试实例 | 系统工程师  需求分析员  需求分析员  软件体系结构（设计）者  开发者  开发者  测试工程师 |

## 能力工具

由于联系链源于开发组成员的头脑中，所以需求跟踪能力不能完全自动化。然而，一旦已确定联系链，特定工具就能帮你管理巨大的跟踪能力信息。可以使用电子数据表来维护几百个需求的矩阵，但更大的系统需要更“鲁棒”的解决办法。

具有强大需求跟踪能力的商业需求管理工具均使用如表16 -7的跟踪能力矩阵。可以在工具的数据库中存储需求和其他信息，定义不同对象间的联系链，甚至包括同类需求的对等联系链。有一些工具需要区分“追溯到（跟踪进）”与“从..回溯（跟踪出）”关系，自动定义相对的联系链。这就是说，如果你指出需求R追溯到测试实例T，工具会自动定义相对的联系“ T从R回溯”。还有一些工具可以在联系链某端变更后将另一端标为“可疑”。可以让你检查确保知道变更的后续效果。

这些工具允许定义“跨项目”或“跨子系统”的联系链。一个有20个子系统的大项目，某些高层产品需求建立在多个子系统之上。有些情况下，分配给一个子系统的需求，实际上是由另一个子系统提供的服务完成的。这样的项目采用商业需求管理工具可以成功地跟踪这些复杂的跟踪能力关系。

## 能力过程

当你应用需求跟踪能力来管理工程时，可以考虑下列步骤：

决定定义哪几种联系链，可以参见图2来进行。 选择使用的跟踪能力矩阵的种类，是表1还是表2。 确定对产品哪部分维护跟踪能力信息。由关键的核心功能、高风险部分或将来维护量大的部分开始做起。 通过修订过程和核对表来提醒开发者在需求完成或变更时更新联系链。 制定标记性的规范，用以统一标识所有的系统元素，达到可以相互联系的目的。若必要，作文字记录，这样就可以分析系统文件，便于重建或更新跟踪能力矩阵。 确定提供每类联系链信息的个人。 培训项目组成员，使其接受需求跟踪能力的概念和了解重要性、这次活动的目的、跟踪能力数据存储位置、定义联系链的技术—例如，使用需求管理工具的特点。确保与会人员明白担负的责任。 一旦有人完成某项任务就要马上更新跟踪能力数据，即要立刻通知相关人员更新需求链上的联系链。 在开发过程中周期性地更新数据，以使跟踪信息与实际相符。要是发现跟踪能力数据没完成或不正确那就说明没有达到效果。

## 能力可行性

对有很多子系统的巨大产品进行跟踪[能力管理](https://baike.baidu.com/item/%E8%83%BD%E5%8A%9B%E7%AE%A1%E7%90%86)是一项巨大的工程，但这很必要。并不是所有的公司都会因为软件问题而造成严重的结果，然而应该严肃地对待需求跟踪，尤其对涉及你业务核心的[信息系统](https://baike.baidu.com/item/%E4%BF%A1%E6%81%AF%E7%B3%BB%E7%BB%9F)。考虑了应用技术的成本和不使用的风险后，才能决定是否使用任何改进的[需求工程](https://baike.baidu.com/item/%E9%9C%80%E6%B1%82%E5%B7%A5%E7%A8%8B)实践。随着软件的发展，要把时间投向回报丰厚的地方。