

Martin Glinz

需求工程术语表

注意：该术语表仅与 CPRE 基础级大纲 3.0 版一致。

Version 2.0.0 2021 年 7 月 1 日

专业需求工程师（CPRE）认证的标准术语表，供学习和考试之用。



University of
Zurich^{UZH}

Department of Informatics



Requirements
Engineering
Research
Group



关于作者

马丁·格林兹 (Martin Glinz) 是瑞士苏黎世大学 (UZH) 的正教授。自 1993 年至 2017 年，他在苏黎世大学的信息学系担任教授。自 2007 年至 2016 年，担任系主任。他的研究范围涉及需求工程和软件工程——特别是在与建模、验证、质量和演化相关的工作。

1977 年，他在亚琛工业大学取得数学硕士学位，并在 1983 年获得计算机科学的自然科学博士学位。加入苏黎世大学之前，他在工业界工作了十年。这十年里，他主要从事软件工程研究、开发、培训和咨询。他于 2017 年夏季退休，但他依然活跃在需求工程的研究、教育和服务中。

马丁·格林兹在需求工程的学术界和工业界拥有超过 35 年的经验积累。他是软件和需求工程领域主要期刊及会议的编辑委员会和程序委员会的成员，并担任该领域顶级国际会议的总主席、程序主席、指导委员会主席和组织者。他是国际需求工程委员会 (IREB) 的正式成员，并主持 IREB 理事会。他在 2016 年获得了 ACM SIGSOFT 杰出服务奖和 IEEE 国际需求工程大会终身服务奖，并在 2017 年获得了 IEEE 国际需求工程大会最有影响力论文奖。

使用条款

如果承认版权并将其引用在专题讨论的材料中，个人和培训机构可以使用这个术语表作为专题讨论的基础。任何人想在广告中使用这个术语表，都需要 IREB e. V. 对此用途的书面许可。

在该术语表被正确引用的前提下，任何个人或团体都可以使用这个术语表作为论文、书籍或其它衍生出版物的基础。

© 2011 - 2020 国际需求工程委员会 IREB e. V. 和 Martin Glinz

本文版权归 IREB 所有。用于个人或培训目的的数字拷贝或硬拷贝是允许的。未经作者和 IREB e. V. 事先的书面许可，禁止以任何其他方式的复制、存储于检索系统中，或以任何形式或通过任何手段进行传播，如通过电子的、机械的、复印的、录制的或其它的方式。

鸣谢

非常感谢下列人员对于该术语表做出的贡献。与 Klaus Pohl, Chris Rupp 和 Thorsten Weyer 的讨论和合作形成该词汇表第一版中的几个定义。Karol Frühauf 评审了 2.0 版中所有定义的草稿。Karol 的评审建议以及他和我之间随后的讨论是改进的宝贵资源。

IREB 和 ISTQB 术语表之间的术语一致是在 IREB 的 Karol Frühauf 和我以及 ISTQB 的 Matthias Hamburg 和 Armin Born 之间的激烈讨论中实现的。

Xavier Franch 是 IREB 委员会中引领该术语表的人。他仔细审查了终稿并提供反馈，这些反馈在许多地方改进了终版文档。

许多人为将术语翻译成英语以外的其他语言做出了贡献。只有德语的翻译是我自己完成的。中文版术语表由夏勇校阅完成。

发布说明

该 2.0 版术语与 IREB CPRE 基础级 3.0 版和 CPRE 高级模块相一致。CPRE 基础级 2.2 版的用户应当使用先前 1.7 版的术语表。

翻译不再作为本文档的一部分。取而代之的是，每种翻译都作为相应语言的“需求工程术语字典”单独发布。

版本更新历史

1.0 版	2013 年 10 月：中文初始版本
1.1 版	2018 年 1 月：修改页眉/页脚、修正格式化和错别字问题、更正术语“需求规格说明”
1.2 版	2018 年 4 月：更正版本更新记录
2.0.0 版	2021 年 7 月：对本术语表涵盖的术语进行重大修订和扩展，包括 CPRE 高级级别中的重要术语。与 CPRE 基础级 3.0 中使用的术语相一致。实现了 IREB 和 ISTQB 术语之间的对齐。

目录

关于作者	2
使用条款	3
鸣谢	4
发布说明	4
版本更新历史	4
目录	5
前言	6
术语定义	7
缩略语表	24
来源	25
参考文献	26

前言

在 2011 年 5 月出版的第一版术语表的序言中，我写道：

如果想要查找需求工程术语的定义，你可以在网上搜索找到几乎所有术语的定义。然而，这样的搜索费时费力，而且结果的质量无法保证。通常情况下，查找到的不同来源的定义互相之间存在歧义。现有的需求工程的教科书中的术语表通常只针对本教科书所涵盖的主题。一直缺乏将术语系统地翻译成英语以外的主要语种。

本术语表的目的是收集需求工程术语的现有的知识并仔细地、连贯地定义核心术语。在超过一个定义的情况下，或从不同视角有不同的定义时，多个定义或视角都将包括在内。对于既具有一般的含义，又有需求工程上下文特定的含义的术语，两个含义都将包括在内。重要的术语将补充提示和附加信息。

该术语表弥补了上述空缺。不仅编译现有定义，而且仔细、始终如一地定义需求工程核心术语的原则也经受住了时间的考验。但是，自术语表首次发布以来，已经过了将近十年，是时候进行重大修订了。

好的术语表应该是稳定的制品：用户需要依靠通用术语——如果该术语不断变化，这就不可能实现。而另一方面，认为术语不会随着时间的推移而发展是不可取的想法。特别是，IREB CPRE 基础级别教学大纲的重大修订，要求对术语进行修改和扩展。此次重大修订也是为了包含 IREB CPRE 高级级别大纲中的重要术语（最初发布术语表时，还没有高级级别大纲）。最后，IREB 和 ISTQB（国际软件测试认证委员会）已在 2019 年达成一致，在各自的术语表中统一质量和测试术语。

在第一版术语表里定义的 128 个术语中，有 42 个（即大约三分之一）保持不变。对 67 个定义进行了微小或仅语法上的更改。我们重新编写了 17 个定义，删除了 2 个定义，并添加了 85 个新定义。主要增加的内容为有关敏捷、建模、原型设计和产品线的术语。我们也增加了一些基本的术语，如活动、方法、过程和技术。

许多重大变化是因为与 ISTQB 的术语统一。但是，我们还对基本术语进行了现代化处理：例如，我们简化了需求和需求工程的定义，并对系统定义中的注释进行了重大更改。术语表的重大修订还提供了在一个在所有定义中清楚标记解释性注释的机会，这将其与主要定义词组分开。

术语到其他语言的翻译是本术语表以前版本的组成部分，现已作为单独的术语字典发布。我非常感谢所有翻译人员所做的工作。

我非常感谢 Karol Frühauf 认真审查了我所有的定义草稿，并进行了颇具成果的讨论，使该术语表有了重大改进。也感谢 Xavier Franch 和 Stan Bühne 提供的建设性建议。最重要的是，我要感谢我的妻子 Angelika。没有她的爱、耐心和理解，我的大部分专业工作，包括这项工作，都不可能实现。

Martin Glinz

写于苏黎世，2020 年 10 月

术语定义

加粗的术语是 IREB CPRE Foundation Level 中必须知道的关键术语。

术语	定义
接受	评估一套↑系统是否满足其所有↑需求的过程。
可接受标准	<p>在敏捷中，为了使↑利益相关者能够接受，↑用户故事的实现所必须满足的标准。</p> <p>注意：用户故事以外的↑待办清单事项也可以编写可接受标准。</p>
验收测试	<p>评估一套↑系统是否满足其所有↑需求的测试。</p> <p>注意：通常↑客户用其来决定是否接受一个系统。</p>
活动	一个人或团队为了完成↑任务而进行的一个或一组动作。
活动模型	↑系统中的某些部分的动作流↑模型。
活动图	是↑UML 图中的一种类型。它模拟一个↑系统中某些部分的动作流，包括↑数据流和那些必要的责任划分。
角色	<p>处于某种↑角色的人，在所考虑的主体范围内，与该主体互动的↑系统或技术装置。</p> <p>注意：在需求工程中，该处于研究状态的对象通常为一个↑系统。在测试中，该对象可能为测试↑目标。</p>
（需求的）充分性	一个↑需求表达出↑利益相关者真正及认可的愿望和需要的程度（即他们在陈述需求时实际上在心里想的那些需求）。
敏捷	<p>1. 通常： (a) 能够快速、方便地移动。 (b) 快速、智能、聪明。</p> <p>2. 在软件开发中：一种产品开发方法，通过将工作划分成持续固定的时间段（时间盒）来↑增量↑迭代构建产品。</p> <p>注意：敏捷开发的特点是专注于在每次迭代中制品的交付和↑利益相关者间的合作。这些合作是在频繁反馈和在每次迭代后根据反馈及↑需求变更而调整计划中达成的。</p>
歧义性	与→“无歧义性”相反
应用领域	现实世界中那些确定↑系统↑环境的相关部分。
制品	↑“制品（work product）”的同义词
关联	在 UML 中：一↑个 UML↑类模型中两个↑类之间的关系。
属性	一个↑实体或↑目标的特性。

术语	定义
待办清单	见→"产品待办列表 (Product backlog)", →"冲刺待办列表 (Sprint backlog)"。
基线	<p>↑制品的一个稳定的、变更控制的↑配置。</p> <p>注意：基线用于↑版本规划和版本定义及项目管理，如估算工作量。</p>
行为	<p>↑系统对（外部或内部）刺激做出反应，改变其内部状态并产生可观察结果的方式。</p> <p>注意：刺激可能是事件，或是条件的变化。它们的源头可能是外部或系统内部的。</p>
行为模型	描述一个↑系统↑行为的↑模型，比如↑状态机活动图。
分支	<p>在某个时间点从主线（或从另一个分支）分出一列↑配置或↑制品↑版本。</p> <p>注意：分支是通过复制某些配置或制品版本而创建的，并且这份拷贝品是分支的根。分支可以在以后的某个时间点与主支或另一个分支合并。</p>
缺陷	见→“缺陷 (Defect)”
燃尽图	绘制在时间范围内仍需完成的工作项目的图表。
业务需求	<p>一项说明业务↑目标或组织需要的↑需求。</p> <p>注意：业务需求通常描述那些通过使用系统或↑系统集成应实现的业务目标和需求。</p>
基数	<p>1. 在建模中：表示↑对象在一种关系中的最小和最大数目。</p> <p>2. 在数学上：表示一个集合的元素数量。</p> <p>注意：在↑UML 中，用多重数 (Multiplicity) 表示基数 (Cardinality)。</p>
变更控制委员会	<p>由↑客户和↑供应商代表组成的委员会，负责决定↑变更请求。</p> <p>简称：CCB</p> <p>注意：变更控制委员会不应与变更咨询委员会混淆，变更咨询委员会是评估正在运行的↑系统的变更请求的委员会，通常没有决策权。</p>
变更管理	一种实现或拒绝↑制品变更请求的控制方式。
变更请求	在需求工程中：一个论据充分的请求，要求改变一个或多个已经设置了↑基线的↑需求。
易改变性	见→“可变更性” (Modifiability)
类	表示一组在结构上、操作方式和行为上相同类型的↑对象。
类图	一个↑类模型的图形表达。
类模型	包含一组↑类和它们之间关系的模型。

术语	定义
共性	↑产品线的所有构件共有的部分。
（需求的）完整性	<ol style="list-style-type: none">1. 对于一个单一的↑需求：需求的规格说明的独立程度。2. 对于包含多重需求的↑制品：制品包含与该制品范围相关的所有已知需求的程度。
遵从	↑制品对↑标准、约定、法规、法律或类似规定的遵守。
组件	<ol style="list-style-type: none">1. 一般而言指↑系统的可定界的部分。2. 在软件架构中指一套共同达成某些目的的相关联的↑对象或↑类。3. 在测试中指一个↑系统中可以进行孤立测试的部分。 <p>注意：孤立地看，组件本身是一个↑系统。</p>
（技术环境中的）构成	<ol style="list-style-type: none">1. 一个由一组实体组成的↑实体，形成一个“整体-部分”的关系。2. 由一组部件组成一个整体的动作。
配置	逻辑上一致的↑实体组成的集合。实体是在一个↑版本中的、各个可识别的↑制品，或制品的部分。
（关于需求的）冲突	见→“需求冲突 (Requirements conflict)”。
符合性	↑制品符合某些↑标准的程度。
（需求的）一致性	对一套↑需求的描述无前后矛盾程度。
（需求工程的）约束	一种↑需求，它限制解决方案在那些必要的满足给定的↑功能需求和↑质量需求之外的空间。
环境，上下文	<ol style="list-style-type: none">1. 一般是指对理解现象或言论所需要的思维和含义网。2. 在需求工程中特指↑系统周边对理解系统及其↑需求紧密相关的环境部分。 <p>注意：第二个含义也称为↑系统环境。</p>
环境边界	一个↑系统的↑环境和那些对↑系统及其↑需求无关紧要的↑应用范围之间的边界。
	<p>注意：环境边界将待开发的系统周边相关的部分从不相关的部分，即那些不影响系统的开发，因而在需求工程的过程中不必考虑的部分，分离开来。</p>
环境图，上下文图	<ol style="list-style-type: none">1. ↑环境模型的一种图形表达。2. 在↑结构化分析中，环境图（上下文图）是分层次↑数据流图的最顶层。
环境模型，上下文模型	描述↑系统在其↑环境中的↑模型。
控制流	一组动作的执行顺序。

术语	定义
正确性	<p>↑制品中所包含的信息被证明为真实的程度。</p> <p>注意：在需求工程中，正确性经常被用作↑“充分性”的同义词，尤其是在针对↑系统↑环境中的规定产品严格验证↑需求时。</p>
客户	<p>接受↑系统、↑产品或↑服务的个人或组织。</p> <p>也请参见↑“利益相关者”。</p>
客户需求说明	<p>从↑客户的角度对一个↑系统所需功能的粗略描述。</p> <p>注意：客户需求说明通常↑由客户提供。</p>
数据流	<p>自生产者向消费者传输的一系列数据项。</p>
数据流模型	<p>通过↑活动、数据存储和↑数据流描述↑系统↑功能的模型。</p> <p>注意：输入数据流触发处理进程，处理程序然后消费接收到的数据，进行转换，读写数据存储中的持久性数据，然后产生新的数据流。该数据流可能是中间结果将触发其它进程，或是最终结果而退出系统。</p>
数据流图	<p>↑数据流模型的图形表示。</p> <p>缩写：DFD</p>
决策表	<p>复杂决策的表格表示，根据条件值的可能组合需指定所需进行的行动。</p>
缺陷	<p>会损害预期用途的↑制品中的缺陷或不足。</p> <p>同义词： bug, fault</p>
设计	<ol style="list-style-type: none">1. 一个计划或图样，用来展示事物在被制作之前的外观、功能或结构。2. 创建设计的活动。3. 装饰图案[此含义不适用于软件工程↑领域]。 <p>注意：</p> <ol style="list-style-type: none">1. 在软件产品开发中，我们区分了塑造产品外观和感观的创意设计（如，可感知的形式、功能和质量）和决定产品内部结构的技术设计（也称为软件设计），特别是软件架构。2. 产品的创意设计也称为产品设计。3. 数字解决方案的创新设计称为数字设计。
文档模板	<p>一个为文档提供预定义的框架结构的模板。（见→“需求文档模板 requirements template”）</p> <p>注意：在需求工程中，文档模板可用于构建↑需求文档。</p>
领域，范围	<p>（对某些给定的事情）相关东西的范围，例如，一个↑应用程序范围。</p>

术语	定义
领域模型	<p>描述↑应用程序领域中现象的↑模型。</p> <p>注意：</p> <ol style="list-style-type: none">1. 在需求工程中，创建领域模型的目的是了解某将落地的计划中↑系统的↑应用程序领域。2. 静态领域模型在利益↑范围中陈述（业务）对象及其关系。3. 领域故事模型表示视觉故事，即角色在↑领域中如何与设备、工件和其他事物进行交互。
领域需求	↑系统↑环境中的为了使系统实现其目的而必须持有的↑领域属性。
有效性	<p>↑实体产生预期结果的程度。</p> <p>注意：在需求工程中，有效性通常是指↑系统使↑用户达到其↑目标的程度。</p>
效率	与所取得的成果相关的资源消耗程度。
（需求的）梳理	需求↑获取、↑协商和↑确认的总称。
（需求的）获取	见→“需求获取 (Requirements elicitation)”
终端用户	见→“用户 (User)”
实体	<ol style="list-style-type: none">1. 一般而言：任何可感知或可想象的东西（见→“实体 item”）。2. 在实体关系建模中：指单个↑对象——它具有一个标识，并且不依赖于另一个对象（见→“对象 object”）。
实体关系图	<p>↑实体关系模型的图形表示。</p> <p>缩写：ERD</p>
实体关系模型	<p>与一个↑系统或一个↑应用领域的数据库相关的数据↑模型，由一组实体类型组成，每个实体类型都有↑属性，并由关系连接。</p> <p>缩写：ER 模型</p>
史诗	在敏捷开发中：↑对利益相关者需求的抽象描述，通常大于单个↑迭代中可实现的内容。
错误	<ol style="list-style-type: none">1. 产生不正确结果的人为行动。2. 观察到的行为或结果和规定的↑行为或结果之间的差异。 <p>注意：在实践中，两种含义都被使用。如果需要，可以分别使用人为错误和观察到的错误或故障来消除“错误”的歧义。</p>
进化型原型	构成开发中的↑系统核心的先导系统。
探索型原型	用来建立共同理解、澄清↑需求或验证需求的一次性↑原型。
故障	见→“缺陷 (Defect)”

术语	定义
容错性	尽管有（硬件或软件上的）↑故障存在，↑系统仍可按预期操作的能力。 注意： 容错性可以作为↑质量需求。
（需求的）可行性	在现有的↑约束下可以实现↑系统↑需求的程度。
特性	为↑利益相关者提供价值的↑系统显著之处。 注意： 特征一般包括多种↑需求，并用于与↑利益相关者在更高度的抽象上的沟通以及表达可变的或供选的特征。
特征图	↑特征模型的图形表达。
特征模型	描述↑产品线可变特征的↑模型，包括它们之间的关系和依赖性。
表单模板	提供表单的模板，其中包含要填写的预定义字段。（见→"需求文档模板 requirements template"） 注意： 在需求工程中，表单模板可用于描述↑用例或↑质量需求。
功能性需求	关于一个↑系统的功能应提供的结果或↑行为的↑要求。
功能性	↑系统↑功能要求的说明能力。
术语表	在一些↑领域中相关术语定义的集合。 注意： 通常情况下，术语表还包含交叉引用词、↑同义词、↑同音异义词、首字母缩写词和缩略语词。
目标	事情的一个理想状态（即↑利益相关者想要达到的目标）。 注意： 目标描述利益相关者的意图。它们可能互相冲突。
目标模型	代表一组↑目标、子目标及两者之间关系的↑模型。 注意： 目标模型可能还包括实现目标所需的任务和资源、想要实现目标的参与者以及阻碍实现目标的障碍。
同形同音异义词	一个术语和另一个术语写法相同，但具有不同的含义。 注意： 例如，bill 为银行票据，而 bill 作为一个（材料的）列表，它们是同形同音异义词。
（软件开发）增量	对开发中的↑系统进行扩展、增强或重构（↑refactoring）系统现有部分的补充。 注意： 在↑敏捷开发中，每次↑迭代都会产生一个增量。
审查	由专家根据给定的标准，按照一定的程序对↑制品进行的正式↑评审。

术语	定义
实体	<p>任何可感知或可想象的东西。</p> <p>同义词：实体（entity），对象（object）</p>
迭代	<p>1. 一般而言：重复某些事，例如：一个程序、一个过程或一段程序代码。</p> <p>2. 在敏捷开发中：一个↑时间盒的工作单位，开发团队在其中对正在↑开发的产品实现↑增量。</p> <p>注意：在敏捷开发中，迭代和↑冲刺一般作为同义词。</p>
需求类型	<p>按需求类型将需求分类为↑系统需求（包括↑功能需求，↑质量需求和↑约束），项目需求和过程需求的分类方式。</p> <p>注意：</p> <ol style="list-style-type: none">1. 需求工程主要与系统需求相关。2. 质量需求和约束也称为↑非功能性需求。
语言	<p>一套结构化的表达和交流信息的符号。</p> <p>注意：符号是用于交流的元素：读写的文字或词组、记号、手势或声音等等。</p>
易维护性	<p>一套↑系统能够被预定的维护人员修改的难易程度。</p> <p>注意：易维护性可作为↑质量需求。</p>
方法	<p>一个或多个↑技术的系统应用，以实现某个目标或创建一个↑制品。</p>
方法论	<p>1. 系统地研究特定领域中的↑方法，特别是如何在特定的情况下系统地选择、应用或评估方法。</p> <p>2. 在某些组合中应用的一组↑方法。</p>
（数字系统的）模型	<p>一种中等还原↑原型，可演示用户界面的特征，但无需实现任何实际↑功能。</p> <p>注意：在需求工程中，该模型主要用于描述和确认用户界面。</p>
模型	<p>现实中现有的一部分或将要创造的现实的一部分的抽象表现。</p> <p>注意：</p> <ol style="list-style-type: none">1. 现实的概念包含了任何可能的元素、现象或概念集，包括其他模型2. 模型总是在特定环境中为特定目的而构建。3. 关于模型，现实的模型部分被称为原始部分（the original）。4. 在需求工程中，可以用模型描述↑需求。
建模语言	<p>以一种特定的形式来表达↑模型的↑语言。可以是文字、图形、符号或它们的某种组合。</p>
可变更性	<p>在不降低↑质量的情况下可以修改↑制品或↑系统的程度。</p>
多样性	<p>见→“基数（Cardinality）”</p>

术语	定义
原生原型	一种高保真↑原型，其实现↑系统关键部分的程度使得↑利益相关者可以使用原型来查看系统的原型部分是否可以正常工作并按预期方式工作。
自然语言	人们在日常生活中用于说话和写作的↑语言。 注意：这与人们为特定目的（例如编程或详述）特意创建的人工语言相反。
（需求的）必要性	某↑需求是↑系统↑需求规格的必要组成部分的程度。
协商	见→“需求协商（Requirements negotiation）”
非功能性需求	某个↑质量需求或↑约束。 注意：↑性能需求可被视为另一类的非功能性需求。在本术语表中，性能需求被认为是↑质量需求的一个子类。
对象	1. 一般而言：任何可感知或可想象的东西（见→“实体 item”）。 2. 在软件工程中：每个可辨别的↑实体的特征，在于其↑属性的价值，并且不依赖于另一个实体（→见“entity”）。
对象图	↑对象模型的图形表示。
对象模型	描述一组↑对象及它们之间关系的↑模型。
性能需求	描述性能特性（时间、速度、体积、容量、吞吐量）的↑需求。 注意：在本术语表中性能需求被认为是↑质量需求的一个子类。但也可以认为是它自己的一个↑需求类型。
角色	一个代表一组具有相似需求、价值观和习惯的↑用户的虚构角色，它们预期将以相同的方式使用↑系统。
句型模板	一个以自然语言表达某↑需求或↑用户故事语句的句法结构模板。（见→“需求文档模板 requirements template”）
可移植性	一套↑系统可以转移到另一个平台上而不影响其特征的难易程度。
实践	一种执行某些类型的↑任务或↑活动的有效方式。
优先级	根据一定的标准，实体（如↑需求或↑缺陷）所分配到的重要程度。
优先级处理	为一组↑实体分配优先级的过程。
问题	需要调查、研究或解决的困难、悬而未决的问题或不良状况。
过程	为处理信息或材料而按一定顺序进行的一组相互关联的↑活动。 注意：过程的概念包括业务过程（例如，如何委托和向↑客户发送订购商品），信息过程（例如，如何从与给定问题匹配的数据库中传送记录）和技术过程（例如，车内定速巡航装置）。

术语	定义
过程模型	描述某↑过程或一组相关过程的↑模型。
过程模式	一个抽象的、可重用的↑流程↑模型，可用于配置和实例化给定情况和↑环境的具体流程。
（软件环境中的）产品	一种基于软件的↑系统或↑服务，由↑供应商开发和销售并由↑客户使用。
产品待办列表	<p>在开发或演化↑系统时，开发团队按照优先级有序处理的工作项集合。</p> <p>注意：项目包括↑需求、要修复的↑缺陷或要完成的↑重构。</p>
产品线	<p>一组共同管理的系统（作为产品或服务提供），它们共享一个共同的核心，并具有一组可配置的↑变量集，以满足特定↑客户或市场细分的需求。</p> <p>注意：产品线中有多个↑变量可供选择的点称为↑变量点。</p> <p>同义词：产品族（Product family）</p>
产品负责人	<p>对↑产品功能、价值和↑风险方面负责的人员。</p> <p>注意：产品负责人维护并确定↑产品待办列表的优先级，确保在↑产品待办列表中，↑利益相关者的↑需求和市场的需求都得到了充分的记录，并在与开发团队沟通时代表利益相关者。</p>
原型	<p>1. 在制造业：在大规模生产开始前所制造的一件产品。</p> <p>2. 在软件系统工程中：对↑系统某些特征的初步部分实现。</p> <p>3. 在设计中：设计解决方案的初步、部分实例。</p> <p>注意：</p> <ol style="list-style-type: none">在需求工程中，原型是用于需求↑获取（见↑"specification by example"）和需求↑验证的一种手段。可将需求工程中的原型：<ol style="list-style-type: none">按照其还原度分类为↑原生原型、↑（Mock-up）模型和↑线框图；按照其目的，分类为↑探索型模型和↑进化型模型。
原型设计	涉及↑原型创建和评估的↑过程。
质量	<p>1. 一般而言：实体的一组固有特性满足↑需求的程度。</p> <p>2. 在软件系统工程中：↑系统满足其↑利益相关者表述和暗示的需求的程度。</p> <p>注意：在此定义中的质量表示符合↑需求中所述的预期用途。它不同于口语中意味着卓越品质的质量概念。</p>
质量需求	涉及质量的↑需求，是非功能性的需求。

术语	定义
重构	改进源代码的内部↑质量，特别是代码的结构，而不改变其外部表现行为。
冗余	相同的信息或资源的多次出现。
发布	已经发布了的用于↑客户安装和使用的↑配置。
可靠性	↑系统在特定条件下、特定时间段内执行特定功能的程度。 注意： 可靠性可作为↑质量需求。
需求	1. ↑利益相关者所认为的需要。 2. 一套↑系统必须具备的一项能力或性能。 3. 对需要、能力或性能的文档表达。
需求分析	1. 分析已经获取的↑需求以理解和编写这些需求的文档。 2. 需求工程的同义词。
需求基线	一组↑需求的↑基准线。
需求分支	见→"分支 (Branch)"
需求配置	见→"配置 (Configuration)"
需求冲突	1. 无法同时满足两个或多个↑需求的情况。 2. 两个或两个以上↑利益相关者不同意某些特定↑需求的情况。 注意： 需求冲突必须通过↑需求协商来解决。
需求获取	见→"需求获取 (Requirements elicitation)"
需求文档	一份由↑需求规格说明组成的文档。 注意： 需求文档常作为需求规格说明 (requirements specification) 的代名词。
需求获取	从现有的需求↑来源中寻找、捕获和强化↑需求的过程，可能包括重建或创建需求。
需求工程师	与↑利益相关者合作，进行需求获取、编制需求文档、验证和管理↑需求的人员。 注意： 在大多数情况下，需求工程师是一个↑角色而不是职位。
需求工程	对↑需求的↑规格化和管采取系统化和规范化的方法，目的是了解↑利益相关方的愿望和需求，并将交付不符合这些愿望和需求的↑系统的风险降到最低。 缩写： RE
需求管理	管理现有↑需求和与需求相关的↑制品的过程，特别包括需求的存储、变更和追踪（↑可追踪性）。

术语	定义
需求模型	以定义↑需求为目的而建立的模↑型。
需求协商	↑利益相关者正在努力达成一项协议以解决↑需求冲突的↑过程。
需求来源	导出↑需求的来源。 注意： 典型的来源有↑利益相关者、文档、现有的↑系统和观察。
需求规格说明	通常对一个满足给定条件的↑系统进行系统性描述的↑需求集合。 注意： <ol style="list-style-type: none">在某些情况下，我们区分↑客户需求说明书（通常由↑客户编写）和↑系统需求规格说明书或↑软件需求规格说明书（由供应商编写）。需求规格说明也可以表示对（↑获取、形成文档及↑确认）需求进行说明的↑过程。
需求文档模板	用于描述↑需求的模板。 注意： 在需求工程中，使用了几种形式的模板。↑语句模板用于描述各个↑需求或↑用户故事。↑表单模板可用于描述↑用例或↑质量需求。↑文档模板为↑需求文档提供了预定义的结构。
评审	个人或团体对↑制品的评估，以发现问题或提出改进建议。 注意： 可以对内容和符合性进行评估。
风险	可能威胁到成功的事件。 注意： 通常以其概率和潜在的损害做风险评估。
角色	<ol style="list-style-type: none">一个人在给定环境下所扮演的部分。在↑UML↑类模型中：链接的↑对象在↑关联中所扮演的部分。
安全性	在规定条件下，运行一套↑系统不会导致对生命、健康、财产或环境造成损害所达到的可接受的概率水平的能力。 注意： 安全性↑需求可以作为↑质量需求或↑功能性方面的需求加以说明。
场景，情景实例	<ol style="list-style-type: none">一般而言：描述一个潜在的能促成想要达到的（或不想要的）结果的事件序列。在需求工程中：合作伙伴之间，特别是↑系统和外部↑角色之间的一个有序的交互序列。可以是一个具体的序列（实例场景）或一组潜在的序列（类型场景、↑用例）
（系统开发）范围	在开发一个↑系统时，可以规划和设计的东西的范围。
Scrum	↑系统↑敏捷开发产品的流行↑过程框架。
安全性（信息的安全性）	↑系统在多大程度上保护其数据和资源不被未经授权的访问或使用，并确保其合法↑用户不受阻碍地访问和使用。 注意： 安全性需求可以表述为↑“质量需求”或↑“功能需求”。

术语	定义
语义	一个符号或一组符号在一种↑语言中的含义。
半正式的，半形式化的	<p>某些事情在一定程度上是正式的，但又不完全是正式的。</p> <p>注意： 如果一个↑制品包含正式的部分，但还没有全部正式化，这个制品称为半正式的。通常情况下，一个半正式的制品具有一个定义完整的↑句法，但↑语义定义不完整。</p>
顺序图	是↑UML 中的一个图类，它描述所选定的一组↑对象和（或）↑角色之间按顺序的交互。
服务	<p>提供者（系统、组织、团体或个人）向人或↑系统提供某些↑功能，从而为接收者带来价值。</p> <p>注意： 在系统、软件和需求工程中，服务通常由一个↑系统为↑用户或另一个系统提供。</p>
软件规格需求说明	<p>有关软件↑系统的↑需求规格说明。</p> <p>缩写：SRS</p>
（需求的）来源	→需求来源
规格说明	<ol style="list-style-type: none">1. 作为↑制品：对一个满足给定标准的实体（一个↑系统、一个设备等）的属性所作的系统性的描述。2. 作为过程：描述（↑获取、记录和↑确认）↑实体属性的过程。 <p>注意： 可以是关于需求属性（↑需求规格说明）或实现属性（例如，产品技术规格说明）的说明书。</p>
实例化规格说明	通过提供系统工作和行为的例子来明确↑系统↑需求的一种↑技术。
规格说明语言	一个为了表达↑规格说明书而创造的人造↑语言。
探针	在敏捷开发中：一项旨在了解或收集信息而不是产生↑产品↑增量的任务。
冲刺	↑敏捷开发中的↑迭代，特别是在使用↑Scrum 时。
冲刺待办列表	一组选择要在当前↑冲刺中实现的↑产品待办列表。
利益相关者	<p>一个对↑系统的↑需求有影响或被系统影响的个人或组织。</p> <p>注意： 影响也可以是间接的。例如，一些利益相关者必须遵守他们的管理者或组织所发布的指示。</p>
利益相关者需求	<p>表达↑利益相关者的愿望或需要的↑需求。</p> <p>注意： 利益相关者需求通常由利益相关者撰写及从他们的角度表达他们的愿望和需要。</p>

术语	定义
标准	关于如何解释、开发、制造或执行某些东西的一套正式的、可能是强制性的规定。 注意： 在需求工程中，有 ISO / IEC 和 IEEE 发行的与需求工程有关的标准。
状态机	一个用一组有限的 状态 和 状态转换 来描述 系统行为 的 模型 。状态转换是由事件触发，可以反过来触发动作和新的事件。
状态机图	状态机 的图示。
状态转换图，状态迁移图	见 状态机图 (State machine diagram) 。
状态图	具有分层和（或）正交分解状态的 状态机 。
指导委员会	负责督导项目的委员会。
（需求工程环境下的）故事	见 用户故事 (User story) 。
故事板	一系列使 方案执行情况 可视化的草图或图片。
故事地图	用户故事 的二维排列。 注意： 故事地图帮助理解 系统 的 功能 、识别差距以及计划发布。
结构式分析，结构化分析	一种基于 数据流图 的层次结构对 系统的功能 进行规格说明的方法。数据流以及持久性数据在数据字典中有定义。 环境图 模写输入 数据流 的来源和输出数据流的目的地。
供应商	为 客户 提供 产品 或 服务 的个人或组织。
同义词	一个与另一个有相同含义的词。
句法	在一种 语言 中构建有结构的符号的规则。

术语	定义
系统	<ol style="list-style-type: none">1. 通常情况下：指顺序和结构的原则。2. 在工程上：通过协调行动实现某种目的的一套连贯的、可划分的要素。 <p>注意：</p> <ol style="list-style-type: none">1. 一个系统可以包含其他系统或↑组件作为子系统。2. 系统能实现的目的有<ul style="list-style-type: none">• 在使用系统的地方部署系统；• 将系统作为↑产品向↑用户销售/供应；• 具有将系统功能作为↑服务向用户提供的供应商。3. 包含软件和物理↑组件的系统称为 <i>物联网</i>。4. 涵盖软件、硬件、人员和组织方面的系统称为 <i>社会技术系统</i>。 <p>重要提示：在本术语表中所有涉及系统的定义中，系统都是一个统称，包括：</p> <ul style="list-style-type: none">• 提供给↑客户的↑产品；• 向↑客户提供的↑服务；• 其他可帮助人员或组织实现目标的制品，例如 <i>设备、程序或工具</i>；• 系统的系统↑组件或↑构成。
系统边界	<p>↑系统和它周边↑环境之间的边界。</p> <p>注意：</p> <ol style="list-style-type: none">1. 系统边界限制了系统在实施和部署后的状态。2. 在系统边界，必须定义↑系统及其↑环境之间的外部接口。3. 系统边界通常与↑系统的↑范围重合（表示系统可以成形和设计的范围）。但是，情况并非总是如此：系统边界内可能有某些组件必须按原样重复使用（即，无法成形或设计），而在系统环境中，在系统开发时，有些组件可能需要重新设计（意味着它们在范围内）。
系统环境	与定义和理解待开发↑系统↑需求相关的↑系统环境部分。
系统需求	涉及一个↑系统的↑需求。
系统需求规格说明	<p>涉及↑系统的↑需求规格说明书。</p> <p>注意：系统需求规格说明经常看作是↑需求规格说明（requirements specification）的代名词。</p> <p>缩写：SyRS</p>
任务	连贯的、要完成的工作。
技术	完成↑任务或达成目标的记录下来的一系列的连贯行动。
主题	在敏捷开发中：相关↑用户故事的集合。
时间盒	完成一组↑任务的的固定的、不可扩展的时间量。

术语	定义
(软件工程) 工具	<p>一套帮助开发、运行和维护系统的（软件）↑系统。</p> <p>注意：在需求工程中，工具用来支持↑需求管理以及建模、编写文档和验证↑需求。</p>
可追踪性	<p>1. 一般而言：在相关的↑制品或制品中的↑实体之间建立明确关系的能力。</p> <p>2. 在需求工程中：追溯↑需求的能力：</p> <ul style="list-style-type: none">(a) 向后追溯到它的起源；(b) 向前追溯至其对应设计和相关测试；(c) 追踪其与其它需求之间依赖关系（反之亦然）。
UML	<p>统一建模语言（Unified Modeling Language）的缩写，一种为问题或解决方案建模的标准化的语言。</p>
(需求的) 无歧义性	<p>↑需求的表达不会让不同的人有不同理解的明晰程度。</p>
可理解性	<p>↑实体对目标用户的可理解程度。</p> <p>注意：典型实体包括：↑系统、↑制品或其一部分。</p>
可用性	<p>特定↑用户可以在特定的使用环境中使用↑系统来实现特定↑目标的程度。</p> <p>注意：可用性特别包括↑系统使目标↑用户对其理解、学习、使用和喜爱的能力。</p>
用例，用况	<p>外部↑角色和↑系统之间的一组可能的交互，这些交互为所涉及的角色提供了好处。</p> <p>注意：从用户（或其它外部角色）的角度来看，用例对系统做规格说明：每一个用例都描述一些系统必须向用例中所涉及的角色提供的↑功能。</p>
用例图，用况图	<p>是↑UML 中模写↑角色和↑系统↑用例的一类图表。</p> <p>注意：施动者和用例之间的边界构成了↑系统边界。</p>
用例模型，用况模型	<p>由一组↑用例组成的↑模型，通常同↑用例图一起使用。</p>
用户	<p>一个使用↑系统所提供的↑功能的人。</p> <p>注意：用户（也称终端用户）永远是↑系统的↑利益相关者。</p>
用户需求	<p>表达↑用户需要的↑需求。</p> <p>注意：用户需求通常关于系统应该为这些特定的用户做些什么，以及他们如何与系统交互。用户需求是↑利益相关者需求的一个子集。</p>

术语	定义
用户故事	<p>从↑用户的角度对需求的描述，以及满足该需求后的预期收益。</p> <p>注意：</p> <ol style="list-style-type: none">1. 用户故事通常使用给定的↑语句模板以↑自然语言编写且符合↑验收标准。2. 在↑敏捷开发中，用户故事是在↑产品负责人和开发团队之间传达需求的主要手段。
确认	<p>检查↑实体（↑系统、↑制品或它们中的一部分）是否与利益相关者的需求相匹配的↑过程。</p> <p>注意：在需求工程中，需求确认是确认记录在案的↑需求符合↑利益相关者需求的过程；换句话说：是否已描述出正确的需求。</p>
可变性	<ol style="list-style-type: none">1. ↑系统可以改变或定制的程度。2. 在产品线中：↑产品线的成员之间的↑特点可以有所不同。
变体	↑实体（例如：↑需求）具有的一种可能形式。
可变点	↑产品线中，可以从一组↑变体中选择产品线元素（通常是变量或↑特征）的点。
（需求的）可验证性	<p>可以检查已实施的↑系统满足↑需求的程度。</p> <p>注意：该↑验证可以通过如定义↑验收测试例、测量或↑审查程序等来执行。</p>
验证	<p>确认↑实体（系统、制品或其一部分）是否满足其↑规格说明的过程。</p> <p>注意：需求验证是确认↑需求已正确记录并满足需求的↑质量标准的过程；换句话说，这些需求是否已正确表述。</p>
版本	一个↑实体存在于多重的、有时序的事件中，每次事件的发生都是通过修改它的前导事件而创建的。
视图	<p>↑制品的摘录 – 仅包含当前感兴趣的部分。</p> <p>注意：视图可以抽象或汇总制品的一部分。</p>
观点，视角	<p>对↑系统↑需求的一个特定的视角。</p> <p>注意：典型的观点是↑利益相关者或利益相关者群的看法（如，最终用户的观点，运营商的观点）。然而，也可以是某个专题的观点，如：从安全性角度出发的观点。</p>
（对系统或产品的）愿景	对未来↑系统或↑产品的概念想象，描述其关键特性以及它将如何为↑用户创造价值。
走查，预排	↑制品的作者在↑评审中系统地带领评审者浏览制品，审阅者提出问题并就可能出现的问题发表意见。

术语	定义
线框图	<p>用简单的材料建立的低保真 ↑ 原型，主要用于讨论和验证需求、设计理念或用户界面概念。</p> <p>注意：在对数字系统进行原型设计时，通常用纸建立线框。这种原型也称为 <i>纸质原型</i>。</p>
制品	<p>工作 ↑ 过程中产生的受记录的中间或最终结果。</p> <p>同义词： ↑ 制品 (Artifact)</p>

缩略语表

CCB	变更控制委员会 (Change control board)
CPRE	专业需求工程师认证 (Certified Professional for Requirements Engineering)
DFD	数据流图 (Data flow diagram)
ER	实体关系 (Entity-relationship)
ERD	实体关系图 (Entity-relationship diagram)
IREB	国际需求工程委员会 (International Requirements Engineering Board)
RE	需求工程 (Requirements Engineering)
SRS	软件需求规格说明 (Software requirements specification)
SyRS	系统需求规格说明 (System requirements specification)
UML	统一建模语言 (Unified Modeling Language)

来源

我没有为单个定义标注引用来源，因为我特意决定不只是通过复制粘贴来从各种现有来源中编译定义，而是要谨慎地根据现今的使用情况重新统一所有定义。

有一些定义是基于我自己的工作[GI07]，[GIWi07]和[GI19]。敏捷领域中的大多数定义均来自 IREB RE @ Agile 术语表，是 IRE @ Agile 工作组与我的共同成果。IREB CPRE 基础级教学大纲[IREB20]的修订也提供了一些新的或更改的定义。

编写定义时，我参考了许多国际标准，如[IEEE610]，[IEEE730]，[IEEE830]，[IEEE1012]，[IEEE1028]，[ISO9000]，[ISO12207]，[ISO19770]，[ISO20246]，[ISO24765]，[ISO25000]，[ISO25010]，[ISO26550]，[ISO29148]，[ISO42010]。但是，由于这些标准中定义或使用的术语通常与需求工程术语表不一致或不充分，我没有从这些标准中逐字复制任何定义。

还有影响了一些定义的其他来源[GaWe89]，[My06]，[Po10]，[St73]，和[ZoCo05]。

为了交叉检验，我还查阅了在线韦氏大词典(<https://www.merriam-webster.com>)和维基百科(<https://en.wikipedia.org>)。

下面，我想对一些定义给予肯定，这些定义或多或少是我逐字从某个来源或与他人合作得来的。被引用定义的版权属于被引用作品的作者。共同作品的版权为本术语表的作者和以下提及的人员共同拥有。

术语引用

环境边界	和 Klaus Pohl, Chris Rupp, Thorsten Weyer 的共同成果，基于[Po10]，[PoRu11]和[We10]
功能性需求	和 Klaus Pohl, Chris Rupp, Thorsten Weyer 的共同成果
模型	和 Klaus Pohl, Chris Rupp 的共同成果，基于[PoRu11]
质量需求	和 Klaus Pohl, Chris Rupp, Thorsten Weyer 的共同成果，基于我在需求工程 I 课堂笔记中的定义
需求工程	对和 Klaus Pohl, Chris Rupp, Thorsten Weyer 的共同成果进行简化而形成的定义
需求规格说明	改编自 Pohl 和 Rupp [PoRu11]
系统边界	和 Klaus Pohl, Chris Rupp 的共同成果，基于[Po10]，[PoRu11]
系统环境	和 Klaus Pohl, Chris Rupp, Thorsten Weyer 的共同成果，基于[Po10]，[PoRu11]，[We10]

参考文献

- [GaWe89] Donald C. Gause and Gerald M. Weinberg (1989). *Exploring Requirements: Quality before Design*. New York: Dorset House.
- [Gl07] Martin Glinz (2007). On Non-Functional Requirements. *15th IEEE International Requirements Engineering Conference (RE' 07)*, Delhi, India. 21-26.
- [GlWi07] Martin Glinz and Roel Wieringa (2007). Stakeholders in Requirements Engineering (Guest Editors' Introduction). *IEEE Software* 24(2):18-20.
- [Gl19] Martin Glinz (2019). *Requirements Engineering I*. Course Notes, University of Zurich. <https://www.ifi.uzh.ch/en/rerg/courses/hs19/re-i.html#resources>. Last visited August 2020.
- [IEEE610] *IEEE Standard Glossary of Software Engineering Terminology*. IEEE Std 610.12-1990.
- [IEEE730] *IEEE Standard for Software Quality Assurance Processes*. IEEE Std 730-2014.
- [IEEE830] *IEEE Recommended Practice for Software Requirements Specifications*. IEEE Std 830-1998.
- [IEEE1012] *IEEE Standard for System, Software, and Hardware Verification and Validation*. IEEE Std 1012-2016.
- [IEEE1028] *IEEE Standard for Software Reviews and Audits*. IEEE Std 1028-2008.
- [IREB20] IREB (2020). *Certified Professional for Requirements Engineering - Foundation Level - Syllabus, Version 3.0*. <https://www.ireb.org/en/downloads/#cpre-foundation-level-syllabus-3-0>. Last visited September 2020.
- [ISO9000] *Quality Management Systems — Fundamentals and Vocabulary*. ISO Standard 9000:2015.
- [ISO12207] *Systems and Software Engineering — Software Life Cycle Processes*. ISO/IEC/IEEE Standard 12207:2017.
- [ISO19770] *Information Technology — IT Asset Management — Part 1: IT Asset Management Systems — Requirements*. ISO/IEC Standard 19770-1:2017.
- [ISO20246] *Software and Systems Engineering — Work Product Reviews*. ISO/IEC Standard 20246:2017.
- [ISO24765] *Systems and Software Engineering — Vocabulary*. ISO/IEC/IEEE Standard 24765:2017.

- [ISO25000] *Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*. ISO/IEC Standard 25000:2014.
- [ISO25010] *Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — System and Software Quality Models*. ISO/IEC Standard 25010:2011.
- [ISO26550] *Software and Systems Engineering — Reference Model for Product Line Engineering and Management*. ISO/IEC Standard 26550:2015.
- [ISO29148] *Systems and Software Engineering — Life Cycle Processes — Requirements Engineering*. ISO/IEC/IEEE Standard 29148:2018.
- [ISO42010] *Systems and Software Engineering — Recommended Practice for Architectural Description of Software-Intensive Systems*. ISO/IEC Standard 42010:2007.
- [My06] John Mylopoulos (2006). *Goal-Oriented Requirements Engineering: Part II*. Presentation slides of keynote talk at the 14th IEEE International Requirements Engineering Conference (RE' 06), Minneapolis, USA.
- [Po10] Klaus Pohl (2010). *Requirements Engineering: Fundamentals, Principles, and Techniques*. Berlin-Heidelberg: Springer.
- [PoRu11] Klaus Pohl, Chris Rupp (2011). *Requirements Engineering Fundamentals*. Santa Barbara, Ca.: RockyNook.
- [St73] Herbert Stachowiak (1973). *Allgemeine Modelltheorie*. (in German) Wien: Springer.
- [We10] Thorsten Weyer (2010). *Kohärenzprüfung von Verhaltensspezifikationen gegen spezifische Eigenschaften des operationellen Kontexts* (in German). PhD Dissertation, University of Duisburg-Essen.
- [ZoCo05] Didar Zowghi and Chad Coulin (2005). Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. In A. Aurum, C. Wohlin (eds.): *Engineering and Managing Software Requirements*. Berlin: Springer. 19 – 46.