

在现代软件开发过程中,软件需求不仅是用户对系统的期望与需求的表达,更是软件开发团队在设计、开发和测试中遵循的指南。随着技术的快速发展和市场需求的不断变化,软件需求呈现出动态变化的特性,因此需求的获取、分析、文档化、确认、验证和管理成为软件工程中不可或缺的环节。

需求工程作为一门系统化的方法学,涵盖了从需求获取到需求管理的全过程。通过系统化的需求工程实践,开发团队能够深入理解用户需求,识别潜在的需求冲突,并通过有效的沟通和协作来确保需求的准确性和一致性。

本章将详细探讨软件需求的基本概念、分类、定义及其在软件开发过程中的重要性。通过对需求工程过程的深入分析,我们将了解如何通过系统化的方法获取、分析和管理需求,以确保软件产品既符合用户的期望,又符合项目的预期目标。

本章目标

- 理解软件需求的基本概念和定义。
- 掌握软件需求的分类与层次结构。
- 了解软件需求在软件开发过程中的重要性和特点。
- 了解需求工程的一般步骤,包括需求获取、分析、文档化、确认、验证和管理。
- 了解敏捷软件开发中的软件需求。

5.1 什么是软件需求

本节的重点在于使读者理解软件需求的本质、分类和层次结构,了解软件需求是如何驱动软件开发项目的成功。

5.1.1 软件需求的本质

软件需求的本质是指在特定的环境下,为了解决某个问题或达成某个目标,用户或系统必须满足的条件或能力。它是软件开发过程中极为关键的一环,因为软件需求直接决定了软件产品的功能、性能和质量标准。理解软件需求的本质,有助于我们更准确地捕捉用户的真实软件需求,从而开发出能够满足用户期望的软件产品。

软件需求的本质反映在以下几方面。

(1) 用户期望的反映:软件需求是用户对于软件产品的期望和需求的具体表述,它反映了用户希望软件产品能够帮助他们完成什么样的任务,解决什么样的问题。

(2) 问题解决的方向:软件需求描述了如何解决用户面临的问题,或者如何帮助用户

达成他们的目标。它为软件产品的开发提供了明确的方向和目标。

(3) 软件产品的标准：软件需求定义了软件产品的功能、性能和质量标准,它是评价软件产品是否成功的重要依据。

(4) 动态变化的特性：随着环境的变化、用户需求的改变和技术的发展,软件需求可能会发生变化。因此,软件需求不是静态的,而是动态变化的。

总体来说,软件需求的本质是对用户期望、解决方案和产品标准的描述。理解软件需求的本质,有助于我们更准确地捕捉用户的真实需求,从而开发出能够满足用户期望的软件产品。

5.1.2 软件需求的特点与重要性

软件需求具有一些显著的特点,这些特点对于理解软件需求的重要性和如何进行有效的软件需求管理至关重要。

1. 特点

软件需求的特点如下。

(1) 多样性：软件需求来源于各种不同的利益相关者,包括终端用户、项目赞助者、开发团队、法规制定者等。因此,软件需求可能涵盖多种多样的功能、性能、安全性、可用性等。

(2) 动态性：随着市场变化、技术进步和用户需求的演进,软件需求可能会发生变化。这就需要一个有效的软件需求变更管理过程来处理这些变化。

(3) 复杂性：软件需求往往涉及复杂的业务流程和技术问题,需要通过深入的分析 and 理解才能准确地定义。

(4) 模糊性：由于语言的不精确性和利益相关者对软件需求的理解可能存在差异,软件需求有时会存在模糊性。这就需要通过清晰、准确的软件需求表述和持续的沟通来解决。

2. 重要性

软件需求的重要性如下。

(1) 项目成功的关键：明确、准确的软件需求是项目成功的关键。如果软件需求定义不清,那么可能导致开发的软件不能满足用户的实际需要,以致项目失败。

(2) 质量保证：软件需求是质量保证的基础。通过软件需求,我们可以定义出什么是一个好的软件,从而制定出相应的测试策略和质量标准。

(3) 资源规划：软件需求可以帮助我们进行有效的资源规划。通过了解软件需求的复杂性和范围,我们可以估计项目的工作量,从而进行人力和时间的规划。

(4) 沟通工具：软件需求是项目团队和利益相关者之间的重要沟通工具。通过软件需求,所有人都可以对软件的目标和期望有一个清晰的理解。

因此,对软件需求的理解和管理是软件开发过程中的重要环节,关系到项目的成功与否,以及软件的质量和用户满意度。

5.1.3 软件需求的分类与层次

在软件工程中,软件需求可以按照不同的标准进行分类,以便于更好地理解和管理。常见的软件需求分类方法包括按照软件需求的来源(业务需求、用户需求和系统需求)以及按照软件需求的性质(功能性需求和非功能性需求)进行分类,如图 5-1 所示。

此外,软件需求可以按照不同的层次进行划分,常见的软件需求层次包括高层软件需求和详细软件需求。高层软件需求通常是抽象的、全局的,描述了系统的总体目标和范围;而详细

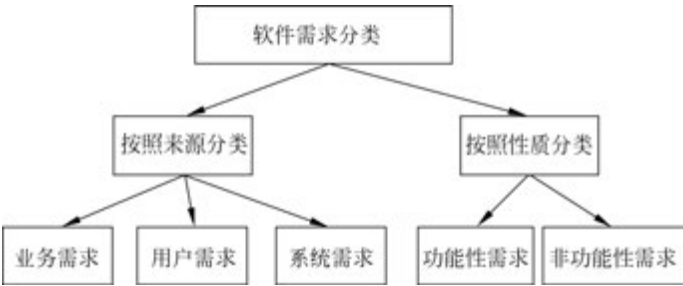


图 5-1 软件需求分类

软件需求则是具体的、详细的,描述了系统的具体功能和操作。理解软件需求的分类和层次有助于我们更好地理解和管理软件需求,从而更有效地开发出满足用户需求的软件产品。

软件需求有 3 种不同的阶段:业务需求、用户需求和功能性需求。此外,每个系统都包含某种类别的非功能性需求。实际上,图 5-1 中的不同种类的需求彼此之间是有一定关联的,如图 5-2 所示。

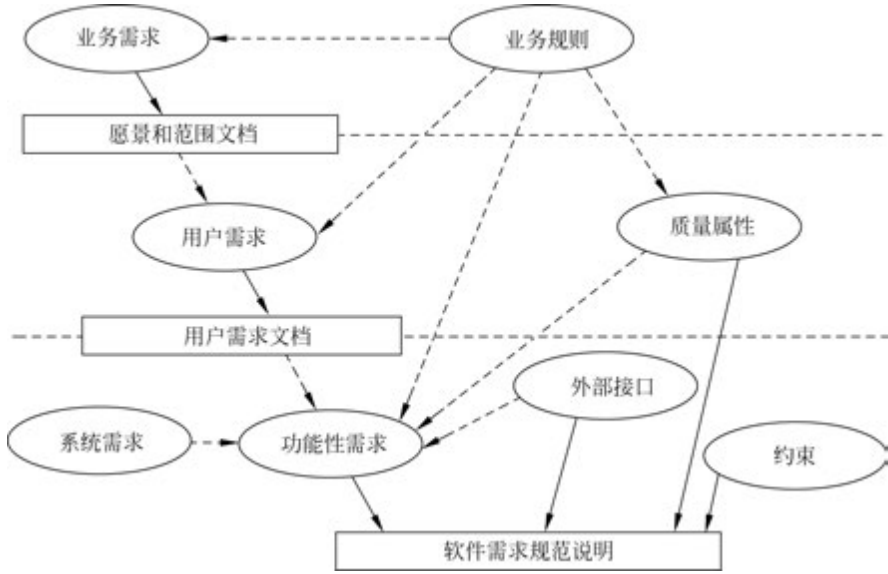


图 5-2 不同需求彼此之间的关联

- 在图 5-2 中,我们用不同形状的图形来表示不同的信息,具体如下。
- (1) 椭圆代表需求信息的种类,也就是我们需要收集的各类需求。
 - (2) 长方形代表存储信息的文件,也就是实际存放这些需求的地方。
- 图形之间的连线也有特殊含义,具体如下。
- (1) 实线箭头(→): 表示这类需求通常存放在箭头指向的文件中。例如,某个椭圆指向一个长方形,就说明这类需求通常存在那个文件中。
 - (2) 虚线箭头(-->): 表示两类信息之间存在关联,可能是某种需求来源于另一种信息,或者会受到另一种信息的影响。
- 这里需要强调的是,业务规则、系统需求应该和软件需求中的其他需求分开存放,不要混在一起。下面将详细讲述业务需求、用户需求、系统需求、功能性需求和非功能性需求。

而图 5-2 中其他需求信息的简要描述如表 5-1 所示。

表 5-1 图 5-2 中其他需求信息的简要描述

需求信息	简要描述
质量属性	一种非功能性需求,它主要描述系统或产品在性能方面有哪些特点(例如,一款在线文档编辑软件的质量属性可以是响应速度快,用户单击操作后 0.5 秒内就能看到界面反馈,极大提升了使用体验)
业务规则	策略、纲领、标准或制度,能对业务的某些方面起到界定或限制作用(例如,对于电商平台,业务规则规定新注册用户首单消费可享 8 折优惠,且下单后 48 小时内发货)
约束	在设计和打造产品时,开发人员需要遵守的各种限制(例如,在开发财务报表系统时,需遵循《企业会计准则》,所有数据计算和报表生成,都必须使用符合准则要求的特定算法和格式)
外部接口	软件系统跟用户、其他软件系统及硬件设备之间是怎么关联的(例如,跨境电商平台对接海关报关接口,自动提交商品进出口报关数据,实现清关流程自动化)
数据需求	在特定的业务场景或系统设计中,用户或组织为了实现其目标、完成相关任务以及满足对产品特性的期望,而对数据的采集、存储、管理、查询、分析和呈现等方面提出的具体要求(例如,在线教育平台为精准推荐课程和评估教学效果,需收集学生的注册信息、课程学习时长、作业完成情况、考试成绩等数据,并对这些数据进行安全存储和多维度分析)

注:在图 5-2 中,没有专门把数据需求呈现出来。这是因为业务需求、用户需求和功能性需求这三个层次都离不开数据需求。

1. 业务需求

业务需求是软件开发项目中至关重要的一环,它们直接关系到软件系统是否能够成功地满足企业、组织或利益相关者的特定业务目标或解决业务问题。下面将详细介绍业务需求的定义、特点以及识别过程等。

1) 业务需求的定义

业务需求源自组织的战略目标和业务需求,它们反映了企业的核心目的和期望的业务成果。例如,假设一家电商公司希望提升在线销售额,他们的业务需求可能包括增加网站流量和提升用户转化率等。这些业务需求是与公司长期发展战略紧密相关的,能够直接推动公司的业务增长。

2) 业务需求的特点

业务需求具有以下几个主要特点。

(1) 高层次:业务需求关注的是期望的商业效益和结果,而不是具体的实现细节。例如,一家快递公司的业务需求可能是提升服务质量和准时送达率,而不是要求系统在技术上如何实现。

(2) 战略性:业务需求与组织的长期目标和策略密切相关,是组织未来发展的基石。例如,一家银行可能制订业务需求,以提升客户满意度和扩大市场份额为目标,这与银行长期的市场竞争策略息息相关。

(3) 度量性:业务需求的实现程度需要通过某种方式度量,通常是通过一些关键指标来衡量。例如,一家保险公司的业务需求可能包括提升理赔处理效率和降低客户投诉率,这些需求的实现程度可以通过理赔处理时间和客户投诉量来度量。

3) 业务需求的识别

识别业务需求的过程包括以下几个步骤。

(1) 业务目标分析: 团队首先需要深入理解组织的商业目标及其转化为软件需求的方式。例如, 一家医院可能希望提升患者就诊体验, 因此他们的业务需求可能包括提升预约挂号系统的效率和改善就诊流程。

(2) 市场研究: 进行市场研究, 分析市场趋势、竞争对手和客户需求, 以便更好地理解市场需求和行业趋势。例如, 一家餐饮连锁企业可能通过市场研究发现, 消费者对于线上订餐服务的需求正在增加, 因此他们的业务需求可能包括开发一个方便快捷的在线订餐平台。

(3) 利益相关者访谈: 与关键利益相关者进行访谈, 获取他们的视角和需求。例如, 一家零售企业可能与供应商、销售团队和客户进行沟通, 了解他们对系统功能和性能的期望, 从而确定业务需求。

(4) 法规遵从性审查: 进行法规遵从性审查, 确保需求符合所有相关的法律和规定。例如, 一家金融机构的业务需求可能受到金融监管机构的严格监管, 因此在确定业务需求时需要确保符合相关的法规和规定。

(5) 风险评估: 进行风险评估, 识别可能影响业务目标实现的潜在风险。例如, 一家航空公司的业务需求可能包括提升客户满意度和安全性, 因此需要对系统的安全性和稳定性进行风险评估, 以确保系统能够安全可靠地运行。

4) 业务需求的例子

一家快递公司为了提升市场竞争力, 决定开发一个订单跟踪系统, 以实现包裹的精确定位和配送的高效管理。该系统的业务需求包括以下几项。

(1) 实时追踪包裹位置: 客户能够通过系统实时查看包裹的当前所在位置。

(2) 提高配送速度: 系统应支持对快递路线的优化, 以确保包裹能够以最快的方式送达。

(3) 减少丢件率: 通过包裹签收确认功能, 确保客户收到包裹, 减少丢件的发生率。

(4) 提高客户满意度: 系统需具备推送通知功能, 实时向客户提供包裹的状态更新(如已发货、派送中和已签收等)。

(5) 业务数据分析: 系统应能够自动生成报表, 帮助公司分析每月的配送效率、客户满意度以及未达标订单情况。

2. 用户需求

用户需求是软件开发过程中的一个重要环节, 它们直接来源于最终用户对软件产品的期望和使用需求。下面将详细介绍用户需求的定义、特性以及识别方法等。

1) 用户需求的定义

用户需求指的是用户对软件产品的功能、性能、界面和操作的具体要求。这些需求直接反映了用户在使用软件产品时的期望, 包括他们希望软件做什么、不希望软件做什么, 以及如何与软件交互。用户需求可能涉及产品的所有方面, 包括但不限于用户界面设计、功能实现、性能优化、安全保障等。例如, 办公软件用户需求: 能快速将 Excel 表格数据转换成图表, 打开文件的时间不超过 3 秒, 界面按钮清晰, 输入内容自动保存, 防止数据丢失。

2) 用户需求的特点

用户需求具有以下几个主要特点。

(1) 用户中心: 用户需求以用户为中心, 考虑的是用户的感受、体验和满意度。这意味着用户需求需要深入理解用户的需求, 尽可能地满足用户的期望。例如, 为老年群体优化购

物 App,简化界面按钮、放大字体、添加语音搜索,还设“一键联系子女代付”功能,贴合老年群体的操作习惯,提升使用满意度。

(2) 具体性:与业务需求的高层次性相比,用户需求更加具体和详细,通常描述了用户在软件中可以执行的具体操作。这意味着用户需求需要具体到每个功能、每个操作、每个界面。例如,电商 App 用户需求:商品详情页需有“一键收藏”按钮,单击后弹窗提示“收藏成功”,且收藏夹能按“添加时间”或“价格”筛选商品。

(3) 易于理解:用户需求应该用非技术的语言表述,以便所有利益相关者,包括非技术背景的人员,都能理解。这意味着用户需求需要避免使用复杂的技术术语,尽可能地使用通俗易懂的语言。例如,外卖 App 用户需求:用户点餐后能随时看到外卖员到哪了,预计还有多久送到,不用自己打电话问,也不用总刷新页面等。

3) 用户需求的识别

识别用户需求是一个复杂的过程,需要采用多种方法和技术进行数据收集和分析,具体如下。

(1) 访谈与调研:通过与用户的直接访谈、问卷调查或用户工作坊来收集软件需求。这种方法可以获取到用户的直接反馈,了解他们在使用产品时可能遇到的问题和期望的改进。例如,为优化生鲜配送 App,团队访谈 20 名用户、发 150 份问卷,发现 85% 用户希望增加“预约收货时段”功能,还反馈缺货提醒不及时。

(2) 观察:在用户的自然环境中观察他们的工作流程,以识别他们可能尚未意识到的软件需求。这种方法可以帮助我们更深入地理解用户的行为和软件需求。例如,观察社区医院护士工作,发现她们常反复切换系统录入患者数据,记录时需先手写再录入,由此识别出“系统数据同步”的潜在需求。

(3) 用户故事:编写用户故事,描述用户在使用产品时可能遇到的场景。这种方法可以帮助我们从一个用户的角度理解产品的使用场景和软件需求。例如,作为外卖骑手,我希望 App 能自动规划避开拥堵路段的取送路线,这样能减少超时,且多接几单,无须自己手工查路况。

4) 用户需求的例子

为了更好地理解用户需求,我们将通过一个在线购物网站的开发项目来进行说明。假设我们正在开发一个在线购物网站,用户需求可能包括如下内容。

(1) 用户能够通过关键词搜索商品,搜索结果应在 2 秒内显示,并按相关性排序。

(2) 用户可以查看商品的详细信息,包括产品介绍、价格、库存和用户评价等。

(3) 用户可以将商品添加到购物车,并在购物车中修改商品数量。

(4) 用户可以选择不同的支付方式进行支付,如信用卡、支付宝、微信等。

(5) 用户可以查看订单状态,包括订单处理进度、物流信息等。

(6) 用户可以申请退款或退货,并在网站上跟踪退款或退货的进度。

(7) 网站应提供用户注册和登录功能,用户可以在个人账户中查看购买历史、管理收货地址和支付方式等。

通过这个具体的例子,我们可以看到用户需求的具体性和翔实性,这将有助于我们在后续的设计和开发过程中,更好地满足用户的需求,提高产品的用户满意度。

3. 系统需求

系统需求是指软件系统的硬件、软件开发技术及架构要求。它们决定了软件产品在何种技术环境下运行、如何部署以及如何与其他系统进行交互。系统需求不仅是软件本身的要求,还涉及整个技术基础设施的规划,从而确保系统能够稳定、高效、安全地运行。系统需求是软件需求的一部分,是实现用户需求和业务需求的技术基础。例如,企业考勤系统需求:需兼容 Windows/Linux 系统,支持与企业 OA 系统数据对接,采用云服务器部署,确保同时 1000 人在线打卡时响应延迟 ≤ 1 秒。下面将详细介绍系统需求的主要组成部分、特点和识别等。

1) 系统需求的主要组成部分

系统需求的主要组成部分包括如下。

(1) 硬件需求:系统运行所需要的物理设备,如服务器、存储设备、网络设备等。硬件需求的定义主要依据系统的规模、用户数量、数据处理量以及响应时间要求。例如,一个在线零售平台需要支持上千个并发用户,则服务器的处理能力、内存和存储空间需要充分考虑,以确保系统能够在高负载下仍然保持稳定的性能表现。

(2) 软件开发技术:系统需求中包括用于开发、运行和维护软件的开发技术和工具。不同类型的软件系统可能需要使用特定的编程语言、框架和数据库技术。例如,开发一个企业级 Web 应用可能需要使用 Java 或 .NET 框架,而数据库可以选择 MySQL、PostgreSQL 或 Oracle。技术需求的选择通常基于系统的复杂性、开发团队的熟悉程度和项目的可扩展性要求。

(3) 系统架构:系统架构是指软件系统的总体设计和结构,它决定了系统的模块如何相互交互、数据如何流动以及系统如何进行扩展。系统架构设计包括如何选择单体架构、微服务架构和分布式架构。例如,对于一个大规模的电商平台,使用微服务架构能够将系统分解为多个独立模块,各模块可以独立开发和部署,从而增强系统的可扩展性和维护性。此外,云架构或本地部署是系统需求的一部分。云环境可能要求系统设计为高可用性、容错和弹性扩展的模式。

2) 系统需求的特点

系统需求具有以下几个主要特点。

(1) 硬件依赖性:系统需求直接受硬件设备的支持和限制。不同规模和类型的软件系统对硬件有不同的要求。例如,一个企业级客户关系管理(Customer Relationship Management, CRM)系统可能需要高性能的服务器、大量存储空间和高速网络来确保数据处理的快速性和稳定性。此外,一个移动应用可能对硬件要求较低,但仍需考虑移动设备的性能和兼容性。

(2) 架构性:系统需求明确了软件的总体架构,包括系统如何模块化设计、如何扩展以及各个组件如何集成和通信。常见的架构类型有单体架构、分布式架构和微服务架构。例如,对于一个全球电子商务平台,分布式或微服务架构可以确保系统在不同地区的数据处理效率和扩展能力。

(3) 技术依赖性:系统需求包括使用的编程语言、数据库、框架和中间件等技术要求。例如,一个数据密集型的系统可能需要选择高性能的关系数据库(如 Oracle 或 PostgreSQL),而一个实时通信应用可能选择 Node.js 或 Go 作为后端开发语言,以处理大量的并发请求。

(4) 安全性与可扩展性:系统需求通常规定了安全措施和未来系统扩展的能力。对于

需要处理敏感数据的应用(如银行系统),系统需求必须包含加密标准、访问控制以及合规性要求。同时,系统需求应考虑系统未来的扩展需求,以确保在用户增长时不需要对系统进行大规模重构。

3) 系统需求的识别

识别系统需求的过程包括以下几个步骤。

(1) 硬件和基础设施评估: 确定系统运行所需的硬件设备,包括服务器、存储设备和网络设备等。团队需要评估业务需求对系统硬件的要求,并根据系统的复杂性选择合适的硬件配置。例如,对于一个需要处理大量交易请求的在线支付系统,团队可能需要选择性能强大的服务器集群和高可用的网络架构。

(2) 架构设计: 系统需求通常由架构设计引导,确定系统的核心结构、模块化方案和交互模型。架构设计需要确保系统能够应对当前业务需求,并具有应对未来增长的弹性。例如,对于一个在线零售平台,分布式架构可能更适合通过增加新的服务器节点来提高负载能力。

(3) 技术选择: 根据业务需求和用户需求,团队需要确定合适的开发语言、数据库、框架和工具链。例如,在开发一个实时数据处理系统时,团队可能选择使用 Kafka 作为消息队列,Hadoop 作为大数据处理平台。技术选择应考虑团队对技术的熟悉程度、项目时间表和系统性能要求。

(4) 性能与安全需求分析: 系统需求必须包含详细的性能和安全要求,确保系统在高负载下稳定运行,并能够抵御潜在的安全威胁。例如,电信公司可能要求其系统处理大量并发用户请求,且不出现性能瓶颈。同时,系统必须符合行业标准的安全要求,确保用户数据不被非法访问或窃取。

4) 系统需求的例子

以下是一些典型的系统需求示例。

(1) 硬件需求如下。

- 处理器: 系统要求至少使用 2.5GHz 四核处理器。
- 内存: 系统运行至少需要 8GB 的随机存取器(Random Access Machine, RAM),推荐 16GB 的 RAM 以保证最佳性能。
- 存储: 系统要求至少 100GB 的可用硬盘空间,用于安装和数据存储。
- 显卡: 系统需支持 NVIDIA GeForce GTX 1060 或以上的独立显卡。

(2) 操作系统需求如下。

- Windows: 系统应支持 Windows 10 或以上版本的操作系统。
- Linux: 系统需兼容 Ubuntu 18.04 LTS 或更高版本的操作系统。
- macOS: 系统需支持 macOS 10.15 及更高版本的操作系统。

(3) 软件需求如下。

- 数据库: 系统需与 MySQL 8.0 或 PostgreSQL 13.0 及以上版本兼容。
- Web 服务器: 系统应要求 Apache HTTP Server 2.4 或 Nginx 5.18 及以上版本。
- 开发框架: 系统必须支持 Spring Boot 2.5.0 或更高版本(用于 Java 项目),或者 Django 3.2.0 及以上版本(用于 Python 项目)。

(4) 网络需求如下。

- 带宽: 系统应在最低 100Mbps 的网络带宽下正常运行,建议使用 1Gbps 带宽以确

保高并发情况下的良好表现。

- 连接协议：系统应支持超文本传输安全协议(Hypertext Transfer Protocol Secure, HTTPS),以确保数据传输的安全性。

(5) 安全需求如下。

- 认证：系统应支持 OAuth 2.0 或 SAML 2.0,用于用户身份认证。
- 加密：所有敏感数据(如用户密码)应使用 AES-256 加密进行存储。
- 日志审计：系统必须记录所有用户登录和关键操作日志,并至少保留 12 个月。

(6) 性能需求如下。

- 响应时间：系统在峰值负载时的响应时间不得超过 2 秒。
- 并发用户数：系统应能够支持至少 5000 名并发用户在线操作而不会出现性能下降。
- 高可用性：系统应保证 99.9%的可用性,平均故障间隔时间(Mean Time Between Failures MTBF)不应少于 1000 小时。

(7) 备份与恢复需求如下。

- 自动备份：系统需支持每日自动备份并保留最近 30 天的备份数据。
- 恢复时间：系统在灾难恢复后应能够在 1 小时内恢复所有功能。

(8) 可扩展性需求如下。

- 系统应设计为支持横向扩展,能够通过添加服务器节点来提升处理能力。
- 系统架构需支持未来用户数量增加至 10 倍的扩展性。

4. 功能性需求

功能性需求是对系统应提供哪些功能的描述。它们是对系统行为的规定,定义了系统应该如何响应特定的输入,以及在特定情况下的行为。下面将详细介绍功能性需求的定义、特点、识别和收集等。

1) 功能性需求的定义

功能性需求是对系统或系统部分的行为描述,它们定义了系统在给定输入下的行为,或者在特定条件下的行为,或者系统必须满足的特定属性。这些需求描述了系统应该做什么,而不是如何做。例如,系统应该能够接受用户输入,处理数据,执行计算,或者与其他系统进行交互等。

2) 功能性需求的特点

功能性需求具有以下几个特点。

(1) 明确性：功能性需求应该明确、具体,没有歧义。每个需求应详细描述系统在特定情况下的预期行为,以便开发团队能够准确地实现它。例如,电商 App 功能性需求：用户提交退货申请后,系统需在 1 小时内发送审核结果短信,审核通过则自动生成带物流单号的退货标签。

(2) 可测性：功能性需求应该是可测的。这意味着对于每个功能性需求,都应该有一种方法来验证系统是否正确地实现了该需求。例如,办公软件功能性需求：用户单击“文档导出为 PDF”按钮后,系统需 5 秒内完成导出,测试时可通过该操作检查 5 秒内是否生成可正常打开的 PDF 文件。

(3) 完整性：功能性需求应该是完整的。这意味着所有的功能性需求应该覆盖系统的所有功能,没有遗漏。例如,外卖配送系统功能性需求：含商家接单、骑手抢单/派单、订单

实时跟踪、用户催单/确认收货、费用结算,覆盖从下单到完成交易全流程功能。

3) 功能性需求的识别和收集

识别和收集功能性需求是软件需求工程的重要环节。以下是一些常用的方法。

(1) 访谈:与系统的潜在用户、管理人员、开发人员等进行访谈,了解他们对系统的期望。例如,为开发校园图书管理系统,访谈10名学生问借书流程痛点、5名管理员聊库存管理需求、3名开发人员确认技术实现难点,收集核心期望。

(2) 调研:通过调查问卷、在线调研等方式收集大量用户的需求。例如,为优化视频剪辑App,在线发放500份问卷,问“常用功能”“操作难点”等,回收420份有效问卷,发现68%的用户希望增加“一键去水印”功能。

(3) 文档分析:分析相关的业务文档,例如业务流程图、操作手册等,以理解系统的业务流程和规则。

(4) 原型设计:设计原型系统,通过用户反馈来收集软件需求。例如,为优化旅行规划App,制作含行程添加、酒店比价模块的原型,让50名用户试用,收集到“需增景点门票预订入口”等反馈以完善需求。

4) 功能性需求的例子

以机票预订系统为例,功能性需求可能包括如下内容。

(1) 航班搜索:用户输入出发地、目的地、日期等信息,系统2秒内按价格、时长等维度返回搜索结果。

(2) 航班详情:单击航班,可查看航空公司、起降时间、退改签政策、剩余座位等详细信息及用户评价。

(3) 机票预订:选择航班和座位后加入订单,支持为不同类型的旅客预订,确认页展示票价及费用明细。

(4) 支付功能:支持信用卡、支付宝、微信等支付方式,能够实时处理交易,成功后生成电子机票并更新订单状态。

(5) 订单管理:可查看各状态的订单,对符合条件的订单进行退票、改签操作,实时更新订单状态。

(6) 账户管理:用户可注册登录,管理个人信息、常用乘机人和支付方式等,会员还能累计里程享权益。

5. 非功能性需求

非功能性需求也被称为质量属性、服务等级需求或系统属性,是对系统性能、安全性、可靠性、可用性等方面的规定,它们定义了系统如何工作,而不是系统应该做什么。

非功能性需求是对系统的某些质量属性的期望,通常包括以下几个类别。

(1) 性能需求:关于系统的响应时间、处理能力、资源使用等方面的软件需求。

(2) 可靠性需求:关于系统的稳定性、错误恢复能力、持久性等方面的软件需求。

(3) 可用性需求:关于系统的易用性、一致性、可理解性等方面的软件需求。

(4) 安全性需求:关于系统的访问控制、数据保护、隐私保护等方面的软件需求。

(5) 可维护性需求:关于系统的可修改性、可扩展性、可测试性等方面的软件需求。

(6) 可移植性需求:关于系统的可安装性、可替换性、适应性等方面的软件需求。

上述类别的具体介绍如下。

1) 性能需求

性能需求涵盖了系统在特定工作负载和数据量下的行为和响应。这通常包括系统的响应时间、处理能力和资源使用等方面。例如,一个在线购物网站可能有这样的性能需求:在黑色星期五这样的高流量时期,网站应能处理每秒 1000 个请求,而平均响应时间不应超过 3 秒。性能需求对于提供流畅、无延迟的用户体验至关重要,因为如果网站加载时间过长,那么用户可能会选择离开。

性能需求的验证通常需要进行性能测试。例如,使用负载测试工具先模拟高流量,然后测量系统的响应时间。

2) 可靠性需求

可靠性需求关注系统的稳定性和故障恢复能力。这涉及系统的正常运行时间、故障频率、恢复时间等方面。例如,一个在线购物网站可能有这样的可靠性需求:网站应有 99.9% 的正常运行时间,这意味着每年的故障时间不超过 8.76 小时。可靠性需求对于建立用户对系统的信任至关重要,因为频繁的故障和长时间的恢复会损害用户的信任。

可靠性需求的验证通常需要进行可靠性测试。例如,使用故障注入技术先模拟系统故障,然后观察系统的恢复能力。

3) 可用性需求

可用性需求涉及系统的易用性、一致性和可理解性。这包括用户界面的设计、错误消息的清晰性、用户文档的质量等方面。例如,一个在线购物网站可能有这样的可用性需求:网站应具有直观的用户界面和详细的用户帮助文档,使用户能轻松地找到他们想要的商品并完成购买。可用性需求对于提供愉快的用户体验至关重要,因为复杂难用的界面和模糊的指导会让用户感到沮丧。

可用性需求的验证通常需要进行可用性测试。例如,邀请目标用户先进行用户体验测试,然后收集他们的反馈。

4) 安全性需求

安全性需求涉及系统的访问控制、数据保护和隐私保护等方面。这包括用户身份验证、数据加密、隐私政策等方面。例如,一个在线购物网站可能有这样的安全性需求:用户的支付信息应通过安全的加密技术进行传输和存储,以防止数据被窃取。安全性需求对于保护用户信息和防止恶意攻击至关重要,因为数据泄露和隐私侵犯会严重损害用户的信任和公司的声誉。

安全性需求的验证通常需要进行安全性测试。例如,使用渗透测试工具先测试系统的安全漏洞,然后修复找到的漏洞。

5) 可维护性需求

可维护性需求涉及系统的可修改性、可扩展性和可测试性等方面。这包括代码的质量、模块的独立性、测试的便利性等方面。例如,一个在线购物网站可能有这样的可维护性需求:网站的代码应遵循一定的编码规范,模块应尽可能独立,以便进行单元测试和维护。可维护性需求对于快速修复错误、添加新功能和提高代码质量至关重要,因为难以维护的代码会增加开发成本和风险。

可维护性需求的验证通常需要进行代码审查和测试。例如,检查代码的质量和可读性,以便改进不合格的代码。

6) 可移植性需求

可移植性需求涉及系统的可安装性、可替换性和适应性等方面。这包括系统的平台独立性、数据格式的标准性、接口的兼容性等方面。例如,一个在线购物网站可能有这样的可移植性需求:网站应能在各种主流浏览器(如 Chrome、Firefox、Safari)上正常运行,而不需要特定的插件或设置。可移植性需求对于扩大用户群和提高用户满意度至关重要,因为用户能在他们选择的任何设备和平台上使用系统。

7) 非功能性需求的例子

某在线银行系统需要满足以下非功能性需求。

(1) 性能需求:系统在高峰期需支持每秒处理 1000 笔交易,且每笔交易的响应时间不能超过 3 秒;页面加载时间需控制在 2 秒以内,以保证用户的使用体验。

(2) 安全需求:系统需符合支付卡行业数据安全标准(Payment Card Industry Data Security Standards, PCI DSS),所有用户支付数据必须加密存储和传输;用户登录需使用多因素认证(Multi-Factor Authentication, MFA),以增加账户访问的安全性;必须定期进行安全漏洞扫描和渗透测试,以确保系统无重大安全风险。

(3) 可用性需求:系统的可用性要求达到 99.99%,保证全年无重大停机事故;需要设置 24 小时自动备份机制,防止数据丢失。

(4) 可维护性需求:系统架构设计需模块化,便于未来功能扩展及维护;系统需提供完善的日志记录功能,供技术人员诊断和分析故障。

(5) 可移植性需求:系统需兼容不同的浏览器(如 Chrome、Safari、Firefox),并支持个人计算机(Personal Computer, PC)端和移动端设备的访问;系统应能够轻松迁移至不同的云计算平台,支持横向扩展以应对业务增长。

需要注意的是,非功能性需求一般关心系统的整体特性,而不是个别的系统特性。因此,非功能性需求比功能性需求对系统更关键。如果一个功能性需求没有满足,那么可能会降低系统的能力;而如果一个非功能性需求没有满足,那么可能使整个系统无法使用。例如,界面不美观尚可忍受,但系统的响应时间太慢(如超过 10 秒),那么这个软件系统就几乎无人愿意使用了。

5.1.4 软件需求与需求工程

1. 需求工程

需求工程是指软件开发过程中,用于定义、分析、记录、验证和管理软件需求的一系列活动和流程。它是将利益相关者的想法和需求转化为清晰的、可实现的软件规范的关键过程。

2. 软件需求与需求工程的关系

软件需求与需求工程的关系如下。

(1) 相互依存:软件需求是需求工程的核心对象,需求工程的任务是获取、分析、记录、验证和管理这些软件需求。没有软件需求,需求工程将失去其存在的意义;没有需求工程,软件需求将无法得到有效的组织、分析和管理。

(2) 目标不同:软件需求关注的是描述系统必须满足的功能和特性,它是开发团队的目标。需求工程则是一套系统化的方法和流程,旨在确保这些需求能够被准确识别、理解、记录并管理,是实现软件需求的手段。

(3) 覆盖范围:软件需求是需求工程活动的产出,具体描述系统的各个方面。需求工

程则涵盖了需求的整个生命周期,包括从获取到验证、管理的全过程。

(4) 动态与静态的关系: 软件需求本身是相对静态的,代表用户对系统的期望和规范。需求工程则是动态的,贯穿软件开发的整个生命周期,确保需求在开发、测试、维护等阶段能够持续满足项目要求。

3. 需求工程对软件需求的重要性

需求工程对软件需求的重要性主要体现在以下几个方面。

(1) 确保完整性和准确性: 通过系统化的方法,需求工程能够帮助开发团队深入理解并准确捕获用户的需求,避免遗漏或误解。

(2) 处理需求变更: 需求工程中的需求管理活动可以有效处理需求在项目生命周期中的变化,确保软件产品始终符合利益相关者的最新需求。

(3) 降低项目风险: 需求工程有助于在项目早期发现并解决潜在的需求问题,减少后期修改和返工的可能性,降低项目风险和成本。

总之,软件需求是对系统功能、性能和限制的详细描述,关注的是“系统应该做什么”。需求工程是一套系统化的方法,用于获取、分析、记录、验证和管理软件需求,确保开发团队能够准确理解并实现这些需求。

5.2 需求工程的过程

本节内容旨在为读者提供一个全面的视角,理解如何系统性地获取、分析和管理需求,帮助读者在实践中高效地应对复杂的需求问题,确保软件产品能够满足用户需求并符合项目的预期目标。

5.2.1 需求工程的一般步骤

需求工程是一个系统的、迭代的过程,涉及多个步骤(如图 5-3 所示),包括软件需求获取、软件需求分析、软件需求文档化、软件需求确认和验证以及软件需求管理。以下是这些步骤的主要内容。

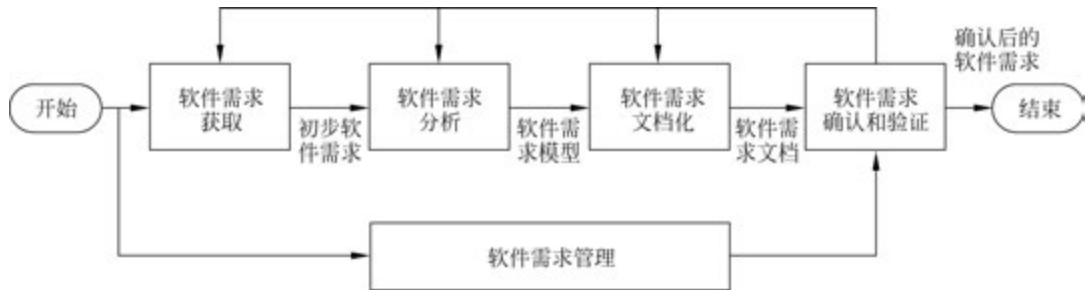


图 5-3 需求工程的一般步骤

(1) 软件需求获取: 这是软件需求过程的起始点,涉及与用户、客户、利益相关者进行交流,理解他们的软件需求和期望。这个过程可能会使用访谈、问卷调查、观察法、原型开发与迭代验证等方法。

(2) 软件需求分析: 在收集到初步的软件需求信息后,需要进行分析,理解软件需求的实际含义,识别软件需求的冲突和缺陷,确定软件需求的优先级。这个过程可能会使用模型、图表、矩阵等工具。

(3) 软件需求文档化：在对软件需求进行了深入的分析后,需要将软件需求转化为具体的文档,这些文档描述了系统必须实现的功能和性能。这个过程可能会生成软件需求规格文档、用例图、状态图等制品。

(4) 软件需求确认和验证：在完成了软件需求文档化后,需要进行确认和验证,确保软件需求规格正确、完整、一致、可实现。这个过程可能会使用审查、测试等方法。

(5) 软件需求管理：整个软件需求过程中需要进行软件需求管理,处理软件需求的变更,维护软件需求的追溯性,评估软件需求的影响。这个过程可能会使用软件需求管理工具、追溯性矩阵、变更控制板等工具。

这些步骤并非是线性的,而是相互关联的。实际的软件需求过程中可能需要反复进行这些步骤,以适应软件需求的变化和项目的发展。

5.2.2 软件需求获取

软件需求获取涉及收集和理解用户、客户、利益相关者的需求和期望。这个过程的主要目标是创建一个全面、详细、准确的软件需求集,这些软件需求可以指导后续的软件需求分析和文档化。以下是软件需求获取的主要方法。

(1) 访谈：访谈是一种需求获取的方法,通过面对面的对话从利益相关者那里收集需求信息。访谈可以是正式的或非正式的,结构化的或非结构化的。它是一种直接、互动的交流方式,能够深入了解用户的需求和期望、业务流程以及潜在的挑战。

(2) 问卷调查：问卷调查是需求获取的常用方法,通过编制标准化、结构化问卷,面向大量用户与利益相关者批量发放,侧重原始需求数据的批量采集。问卷可采用纸质或电子形式,允许受访者自主安排时间填写,适合快速在大范围内收集统一格式的需求信息。

(3) 观察法：观察法是一种通过直接观察用户在其实际工作环境中如何使用现有系统或执行相关任务来获取需求信息的方法。这种方法能够帮助需求工程师(也称需求分析师)理解用户的实际工作流程、操作习惯和遇到的问题。观察法特别适用于了解用户日常工作中的非言语行为和潜在需求,这些需求往往是通过访谈或问卷调查难以获取的。

(4) 工作坊：工作坊是一种通过组织利益相关者和开发团队在特定时间和地点进行集中讨论、交流和协作的需求获取方法。工作坊旨在通过面对面的互动,快速、有效地收集需求、解决问题、达成共识,促进团队之间的沟通与协作。工作坊通常由专业的主持人引导,采用多种互动方式,如头脑风暴、小组讨论、角色扮演等,确保所有参与者积极参与、充分表达意见。

(5) 原型开发与迭代验证：原型开发与迭代验证是软件开发过程中的一种关键方法,它通过创建系统的简化版本(原型)来展示系统的主要功能和界面。这些原型可以是低保真度的草图,可以是高保真度的可交互模型,其目的是让用户和利益相关者在开发初期就能够了解系统的外观和功能,从而提供反馈和意见。

(6) 焦点小组：焦点小组是一种定性研究方法,通常由一个训练有素的主持人带领一组特定的参与者进行深入讨论。其目的是通过互动和讨论,深入了解参与者的观点、感受、需求和动机。这种方法特别适用于需求获取过程中的探索性研究,以便更好地理解用户的真实需求和期望。

(7) 调查分析：调查分析是在调研采集基础上开展的需求梳理方法,以问卷、访谈等各类调研素材为依据,对收集到的原始需求数据进行整理、归纳、统计与深度研判。侧重数据



观看视频

加工、规律提炼和需求量化分析,从中挖掘用户潜在期望与共性诉求,为需求筛选和规格定义提供依据。

(8) 文档研究:文档研究是通过查阅和分析现有文档资料来获取需求的方法。这些文档包括用户手册、业务流程文档、系统规格说明书、技术报告、市场研究报告等。文档研究是一种有效的辅助方法,可以为需求获取提供重要背景信息和参考资料。

(9) 用户故事和敏捷方法:用户故事和敏捷方法是获取需求的有效工具,通过简洁的用户故事描述需求,使用敏捷方法迭代开发和验证系统功能,能够灵活应对需求变化,快速响应用户反馈,提高系统设计的准确性和用户满意度。

(10) 数据分析法:数据分析法是一种通过对现有数据进行收集、整理、分析和解释,来获取软件需求的方法。它通过对各种数据源的深入分析(包括系统日志、用户行为数据、业务数据、市场数据等),来发现用户需求、使用模式和潜在改进点。这种方法特别适用于大型系统和平台,在用户基数大、数据量丰富的情况下,利用数据分析可以提供精确的需求洞察。

这些方法并非相互排斥,而是相互补充的。在实际的软件需求获取过程中,我们可能需要结合使用这些方法,以确保软件需求的全面性和准确性。

近年来,需求获取中的智能化辅助技术开始流行起来。需求获取中的智能化辅助技术是利用人工智能、机器学习、自然语言处理等先进技术,通过分析用户行为数据、挖掘用户反馈信息、理解用户自然语言表述等方式,自动识别和提取用户需求,为需求获取过程提供智能支持,以提高需求获取的效率和准确性,减少人工干预和错误的一系列技术手段。需求获取中常见的智能化辅助技术有以下几种。

(1) 自然语言处理技术:能理解和分析用户输入的自然语言文本,比如分析用户在产品论坛、客服对话中的反馈,自动提取关键需求信息。

(2) 数据挖掘技术:通过分析大量用户行为数据(如用户的操作日志、购买记录等),发现用户的潜在需求和行为模式。

(3) 机器学习技术:利用机器学习算法对历史需求数据进行训练,建立需求预测模型,从而预测用户未来的需求。

(4) 智能推荐系统:根据用户的偏好、历史行为等信息,为用户推荐可能感兴趣的产品功能或服务,同时收集用户对推荐内容的反馈,以进一步挖掘用户需求。

5.2.3 软件需求分析

软件需求分析是需求工程的关键阶段,它涉及对收集到的软件需求进行深入的理解和解析。这个过程的主要目标是识别软件需求的实质,解决软件需求的冲突,确定软件需求的优先级,为后续的软件需求文档化提供基础。以下是软件需求分析的主要活动。

(1) 软件需求理解:这是软件需求分析的基础,涉及对软件需求的深入理解和解析。软件需求工程师需要理解软件需求的含义,理解软件需求的背景,理解软件需求的目标。这个过程可能会使用到模型、图表、矩阵等工具。

(2) 软件需求冲突解决:在收集到的软件需求中,可能存在一些冲突,例如,某些软件需求可能相互矛盾,某些软件需求可能与项目的目标或限制冲突。软件需求工程师需要识别这些冲突,通过协商、调整和削减等方法来解决这些冲突。

(3) 软件需求优先级确定:在收集到的软件需求中,可能存在一些优先级的问



观看视频

如,某些软件需求可能比其他软件需求更重要,某些软件需求可能比其他软件需求更紧急。软件需求工程师需要确定这些优先级,通过排序、打分和分组等方法来确定这些优先级。

(4) 软件需求模型创建:为了更好地理解和管理软件需求,软件需求工程师可能需要创建一些软件需求模型,如用例模型、对象模型、动态模型等。这些模型可以帮助软件需求工程师更好地理解软件需求的结构和动态,更好地管理软件需求的变更和追溯。

在软件需求分析阶段,需求工程师需要对需求进行深入的分析 and 理解,识别并解决潜在的需求冲突,确定需求的优先级,并将这些需求以模型的形式表达出来。这个阶段的工作为后续的需求文档化和验证提供了坚实的基础。

5.2.4 软件需求文档化

软件需求文档化的主要目标是将分析过程中得出的结果以明确、一致、可理解的方式记录下来。软件需求文档是项目团队、利益相关者和最终用户之间沟通的重要工具,也是项目进展的关键参考。以下是软件需求文档化的主要步骤。

(1) 软件需求规格说明书编写:软件需求规格说明书是记录软件需求的主要文档。它详细地描述了软件的功能需求、性能需求、接口需求等,为软件设计和开发提供了基础。规格说明书应该清晰、完整、一致,避免歧义。

(2) 软件需求模型文档化:在软件需求分析阶段,软件需求工程师可能创建了一些软件需求模型,如用例模型、对象模型、动态模型等。这些模型需要以适当的形式文档化,以便于理解和沟通。

(3) 软件需求追溯矩阵创建:软件需求追溯矩阵是一种用于管理软件需求变更和追溯软件需求来源的工具。它可以帮助软件需求工程师了解软件需求的来源,理解软件需求的影响,管理软件需求的变更。

软件需求文档化不仅是记录软件需求的过程,也是沟通和管理软件需求的过程。有效的软件需求文档化,可以确保所有的利益相关者对软件需求有共同的理解,可以有效地管理软件需求的变更,也可以提高项目的成功率。9.5节的“案例:‘小型网上书店系统’的需求规格说明书”就是一个实际的软件需求文档化的例子。

5.2.5 软件需求确认和验证

软件需求确认和验证可以确保最终的软件需求集合能够准确反映用户的需求,并且是可实施的。这个阶段的目的是验证软件需求的正确性、一致性、完整性和可测试性。以下是软件需求确认和验证的主要活动。

(1) 软件需求审查:通过正式的软件需求审查会议,邀请项目团队成员、利益相关者以及用户代表等参与,共同检查软件需求文档。审查的目的是发现软件需求中可能存在的错误、遗漏、不一致和歧义。

(2) 软件需求测试:软件需求测试是指对软件需求文档进行的一系列测试活动,以确保软件需求的可实施性和可测试性。这包括从软件需求角度出发设计测试用例,验证软件的实际行为是否与软件需求相符。

(3) 软件需求验证技术应用:应用多种技术来验证软件需求,如原型、模型和模拟。原型可以快速地获取用户反馈,模型可以确保软件需求的逻辑正确性,而模拟则可以在软件开

发前评估软件需求的实际表现。

(4) 软件需求追溯性确认: 确认软件需求与源需求、系统设计和测试用例之间的追溯性。这确保了任何软件需求的变更都能够被跟踪, 相关的设计和测试用例也能相应更新, 保持软件需求的一致性。

(5) 用户验收测试: 在软件开发的后期, 用户会进行验收测试, 以验证软件是否满足了他们的软件需求。这是最终用户确认软件需求已被准确实现的重要步骤。

软件需求确认和验证是一个迭代的过程, 可能会导致软件需求的进一步细化或修改。持续的确认真和验证活动, 可以确保软件产品能够满足用户的实际软件需求, 并且在整个软件开发生命周期中保持软件需求的质量。

5.2.6 软件需求管理

软件需求管理的目标是确保软件开发项目能够满足用户需求, 同时能够应对软件需求变更。以下是软件需求管理的主要活动。

(1) 软件需求变更管理: 软件需求变更是软件开发过程中常见的情况。软件需求变更管理是一个系统的过程, 包括变更提出、变更评估、变更决策、变更实施和变更验证。

(2) 软件需求跟踪: 软件需求跟踪是持续监视软件需求状态和进度的过程, 以确保软件需求在整个生命周期中得到满足。软件需求报告则是将软件需求的状态、进度和问题报告给项目团队和利益相关者。

软件需求管理是一个持续的过程, 它需要在整个软件开发生命周期中进行。有效的软件需求管理, 可以确保软件产品能够满足用户的实际软件需求, 同时能有效地应对软件需求变更, 提高项目的成功率。

5.3 敏捷软件开发中的软件需求

在敏捷软件开发中, 软件需求通常是指开发团队需要满足的功能性和非功能性需求, 这些需求可以在项目的不同阶段逐步定义和改进。

1. 敏捷软件开发中软件需求的例子

几个敏捷软件开发中的软件需求的例子如下。

1) 用户故事

用户故事是对软件功能的简短、非技术性描述, 通常来自最终用户的需求。例如, 作为一名用户, 我希望能够通过邮箱重置密码, 这样我就可以找回我的账户; 作为管理员, 我希望能够查看用户的活动日志, 以便监控系统的安全性。

2) 功能性需求

功能性需求是软件应该实现的具体功能。例如, 用户可以注册一个新账户, 并通过电子邮件进行验证; 系统应该能够支持实时聊天功能, 允许用户和客服代表进行即时沟通。

3) 非功能性需求

非功能性需求描述了软件应该如何运行, 而不是软件应该做什么。例如, 系统应在高负载情况下能够支持至少 1000 个并发用户; 软件的页面加载时间不应超过 3 秒。

4) 验收标准

验收标准是用来定义用户故事或功能是否完成的标准。例如, 对于用户注册功能, 验收



观看视频

标准可能包括：用户可以成功提交注册信息；收到验证邮件并单击链接后，账户被激活。

在敏捷软件开发中，这些需求通常是通过短周期的迭代逐步实现和完善的，而且需求是不断变化和调整的，因此需要灵活应对。在每个迭代周期中，项目团队与客户或最终用户密切合作，以确保需求的实现符合实际需求和期望。

2. 用例与用户故事

在软件开发中，用例和用户故事都是描述系统需求的方式，但存在诸多不同，具体如下。

1) 定义与侧重点

用例更侧重于详细描述参与者（可以是用户或其他系统）与系统之间的交互方式。它像是一份严谨的系统交互说明书，会明确列出触发条件、前置条件、正常流程、替代流程以及后置条件等，精确刻画系统在不同情况下的行为逻辑。例如，自动取款机（Automatic Teller Machine, ATM）系统“取款”用例会按步骤说明用户插入银行卡、输入密码、输入提款金额、提取现金、退出系统取回银行卡等操作流程。

用户故事从用户视角出发，以更偏向业务价值和非技术角度的方式，描述用户使用系统想要达成的目标或期望获得的产品属性。用户故事的作用是备忘功能，不是功能描述，也不是文档。它更关注“为什么”使用系统，强调功能能为用户带来的价值，而非具体实现步骤，例如“作为仓库操作人员，我想通过使用手持设备来完成仓库的拣选商品操作，以便提高工作效率”。

2) 格式与详细程度

用例的格式较为严格、形式化，拥有主成功场景、失败路径及其他特定场景等详细内容。它对系统功能性需求的定义详尽，能为系统分析师、开发人员及测试人员提供深入、精确的信息，可直接用于指导测试和验证系统功能。

用户故事的格式相对简洁，一般遵循“身份—需求—目的”模式，常用“作为<角色>，我想<实现某功能>，以便<达成某个商业目标>”的形式表述。它故意遗漏很多技术细节，保持在较高层次描述，方便非技术背景人员理解，促进开发团队与利益相关者沟通。

3) 应用场景与项目阶段

用例常用于传统软件开发模型，如瀑布模型。在这类模型中，需求相对固定，用例详尽的描述有助于在项目前期固化需求，明确系统功能和行为边界，减少开发过程中的需求变更。但在需求频繁变更的项目中，其灵活性欠佳。

用户故事在敏捷软件开发中广泛应用，尤其适用于项目早期需求捕捉与定义阶段。它鼓励团队从用户角度思考功能，注重灵活性和反馈，可根据项目进展随时迭代修改，更好地适应需求不断变化演进的现代软件开发环境。

4) 寿命与稳定性

用例通常是持久的工作制品，在产品开发和维护阶段一直存在，是对系统功能性需求相对稳定、固化的定义。

用户故事往往只存在于实现该故事的迭代中。虽然用户故事卡等记录形式可以存档，但很多团队在迭代完成后会将其销毁，因为它更强调在当前迭代中对需求的沟通和理解。

总体来说，用例更注重描述系统如何工作，而用户故事则更多关注“为何”需要实现某个功能。实际中二者可结合使用，例如可先用用户故事梳理优先级，再用用例细化关键功能。关于用例和用例图，本书将在第8章、第9章和第10章详细讲述。

5.4 软件需求实例(1)

【例 5-1】 针对在线教育平台的需求获取。

【解析】 项目背景：某公司计划开发一款面向 K-12(中小学)学生的在线教育平台，支持直播授课、作业提交、学习进度跟踪等功能。需求获取过程如下。

1. 用户访谈

用户访谈(关键利益相关者)的过程如下。

(1) 目标：了解教师、学生、家长和管理员的核心需求。

(2) 访谈记录如下。

• 教师：

“需要支持 PPT 和手写板同步演示，方便讲解数学题。”

“希望平台能自动批改选择题，减少重复劳动。”

• 学生：

“想要课后能回放课程视频，方便复习。”

“作业提交后，希望老师快速反馈，最好有错题解析。”

• 家长：

“需要定期收到学习报告，了解孩子进度。”

“希望有‘家长监控模式’，限制孩子使用娱乐功能。”

(3) 输出需求如下。

- 直播课支持演示文稿(PowerPoint,PPT)和手写板双画面。
- 自动批改作业(选择题和填空题)。
- 课程回放功能。
- 学习报告自动生成并推送给家长。

2. 问卷调查

问卷调查(扩大样本)的过程如下。

(1) 目标：量化需求优先级，覆盖更多潜在用户。

(2) 问卷问题示例如下。

- “您最需要的 3 项功能是什么？(直播互动、作业批改、课程回放和学习社区等)”
- “您能接受的直播卡顿时间是多少？(1 秒以内,1~3 秒,不低于 3 秒)”

(3) 结果分析如下。

- 85%的教师认为“自动批改作业”是核心需求。
- 70%的学生希望有“错题本功能”。

(4) 输出需求如下。

- 高优先级：作业自动批改、错题本。
- 低优先级：学习社区(仅 30%用户需要)。

3. 竞品分析

竞品分析(参考现有产品)的过程如下。

(1) 目标：避免重复设计，挖掘差异化功能。

(2) 分析的竞品：Zoom、腾讯课堂、猿辅导。

(3) 发现：Zoom 缺乏教育专用功能(如作业管理)；猿辅导有 AI 口语评测功能,但仅限英语。

(4) 输出需求：增加数学公式识别功能,差异化竞争。

4. 原型测试

原型测试(验证需求可行性)的过程如下。

(1) 目标：通过低保真原型获取用户反馈。

(2) 测试场景如下。

- 教师试用作业批改界面,发现手工标注错题步骤烦琐。
- 学生尝试课程回放功能,建议增加倍速播放和章节标记。

(3) 输出需求如下。

- 优化批改界面,支持一键圈画错误。
- 回放功能增加 1.5 倍速、2 倍速和书签。

5. 文档分析

文档分析(合规性需求)的过程如下。

(1) 目标：确保符合教育行业规范。

(2) 分析文件如下。

- 根据《未成年人网络保护条例》,该平台需具有家长控制模式。
- 根据《直播类在线教育平台安全保障要求》,开发团队需要对该平台进行数据加密和实名认证。

(3) 输出需求如下。

- 强制家长绑定学生账号。
- 聊天区敏感词过滤。

最终需求列表(部分)如表 5-2 所示。

表 5-2 最终需求列表(部分)

类 别	需 求 描 述	来 源
核心功能	直播课支持 PPT 和手写板	用户访谈(教师)
学习工具	自动批改作业(选择题和填空题)	问卷调查和访谈
用户体验	课程回放和倍速播放	原型测试
家长端	每周学习报告推送	用户访谈(家长)
安全合规	实名认证和聊天敏感词过滤	文档分析

总之,通过多种方法结合,开发团队明确了核心功能(如直播互动、自动批改作业)、差异化需求(数学公式识别)和合规要求(家长监控),为后续开发提供了清晰方向。

5.5 软件需求实例(2)：大语言模型如何赋能软件需求

扫描下方二维码即可阅读。



拓展阅读：高斯数据库 ——以工匠精神铸就国产数据库标杆

扫描下方二维码即可阅读。



本章小结

本章深入探讨了软件需求的基本概念、分类、定义及其在现代软件开发中的核心作用等。我们了解到,软件需求不仅是用户期望和问题解决方向的具体体现,还是软件产品标准和动态变化的反映。

首先,本章详细介绍了软件需求的不同分类方法,包括按照来源(业务需求、用户需求、系统需求)和性质(功能性需求、非功能性需求)进行分类,并讨论了软件需求的层次结构,从高层次的抽象需求到详细的具体需求。

其次,本章探讨了需求工程的过程,涵盖了需求获取、分析、文档化、确认、验证和管理等关键步骤。通过这些步骤的系统化方法,开发团队可以提高需求管理的效率和准确性,确保软件项目的成功交付。

最后,本章介绍了敏捷软件开发中软件需求的处理方法,突出了需求的灵活性和迭代性,为理解和实施软件需求提供了全面的框架与方法。

习 题

本书提供在线测试习题,扫描下方二维码可以获取本章习题。

