第 4 章——需求工程

第1讲

涵盖的主题

- 4.1 功能性和非功能性需求
- 4.2 软件需求文档
- 4.3 需求规范
- 4.4 需求工程过程
- 4.5 需求获取和分析
- 4.6 需求验证
- 4.7 需求管理

需求工程

- 建立客户从系统中要求的服务及其运行和开发的约束的过程。
- 需求本身是对需求工程过程中产生的系统服务和约束的描述。

什么是要求?

- 它的范围可以从服务或系统约束的高级抽象声明到详细的数学功能规范。
- 这是不可避免的，因为需求可能具有双重功能
    - 可能是合同投标的基础——因此必须接受解释；
    - 可能是合同本身的基础——因此必须详细定义；
    - 这两个陈述都可以称为要求。

需求抽象（Davis，1993）

“如果一家公司希望为大型软件开发项目签订合同，它必须以足够抽象的方式定义其需求，而不是预先定义解决方案。必须编写要求，以便多个承包商可以投标该合同，提供满足客户组织需求的不同方式。一旦授予合同，承包商必须为客户编写更详细的系统定义，以便客户理解并验证软件将做什么。这两个文件都可以称为系统的需求文件。”

需求类型

- 用户要求
    - 自然语言语句加上系统提供的服务及其操作约束的图表。写给客户。
- 系统要求
    - 详细说明系统功能、服务和操作限制的结构化文件。定义应该实施的内容，以便成为客户和承包商之间合同的一部分。

用户和系统要求

不同类型需求规范的读者

4.1 功能性和非功能性需求

- 功能要求
  - 系统应该提供的服务的陈述，系统应该如何对特定输入做出反应以及系统在特定情况下应该如何表现。
  - 可以说明系统不应该做什么。
- 非功能性需求
  - 对系统提供的服务或功能的约束，例如时间约束、对开发过程的约束、标准等。
  - 通常适用于整个系统而不是单个功能或服务。
- 域要求
  - 操作域对系统的约束

4.1.1 功能要求

- 描述功能或系统服务。
- 取决于软件类型、预期用户和使用该软件的系统类型。
- 功能性用户需求可能是系统应该做什么的高级声明。
- 功能系统需求应详细描述系统服务。

MHC-PMS 的功能要求

- 用户应能够搜索所有诊所的预约列表。
- 系统应每天为每个诊所生成一份预计当天就诊的患者名单。
- 每位使用该系统的员工都应通过其 8 位员工编号进行唯一标识。

要求不精确

- 当需求没有被准确表述时就会出现问题。
- 开发人员和用户可能会以不同的方式解释不明确的需求。
- 考虑要求 1 中的术语“搜索”
  - 用户意图——在所有诊所的所有预约中搜索患者姓名；
  - 开发人员解释 - 在单个诊所中搜索患者姓名。用户选择诊所然后搜索。

需求完整性和一致性

- 原则上，需求应该是完整和一致的。
- 完全的
  - 它们应包括对所有所需设施的描述。

- 持续的
    - 系统设施的描述不应有冲突或矛盾。
- 在实践中，不可能产生完整且一致的需求文档。

4.1.2 非功能性需求

- 这些定义了系统属性和约束，例如可靠性、响应时间和存储要求。约束是 I/O 设备能力、系统表示等。
- 还可以指定过程要求，强制要求特定的 IDE、编程语言或开发方法。
- 非功能性需求可能比功能性需求更重要。如果不满足这些条件，则系统可能无法使用。

非功能性需求的类型

非功能性需求实现

- 非功能性需求可能会影响系统的整体架构，而不是单个组件。
    - 例如，为了确保满足性能要求，您可能必须组织系统以尽量减少组件之间的通信。
- 单个非功能性需求，例如安全性需求，可能会产生许多相关的功能性需求，这些功能性需求定义了所需的系统服务。
    - 它还可能生成限制现有需求的需求。

非功能性需求分类

- 产品要求
    - 要求指定交付的产品必须以特定方式运行，例如执行速度、可靠性等。
- 组织要求
    - 作为组织政策和程序的结果的要求，例如使用的过程标准、实施要求等。
- 外部要求
    - 由系统及其开发过程之外的因素引起的要求，例如监管要求、立法要求、道德要求等。

MHC-PMS 中的非功能性需求示例

产品要求MHC-PMS 应在正常工作时间（周一至周五，08:30–17:30）提供给所有诊所。任何一天在正常工作时间内的停机时间不得超过五秒。**组织要求** MHC-PMS 系统的用户应使用其卫生当局身份证进行身份验证。**外部要求** 系统应实施 HStan-03-2006-priv 中规定的患者隐私条款。

目标和要求

- 非功能性需求可能很难精确表述，不精确的需求可能难以验证。
- 目标
    - 用户的一般意图，例如易用性。

- 可验证的非功能性需求
    - 使用可以客观测试的某种度量的陈述。
- 目标对开发人员很有帮助，因为它们传达了系统用户的意图。

可用性要求

- 该系统应易于医务人员使用，并应以最大限度减少用户错误的方式进行组织。（目标）
- 医务人员经过四个小时的培训后，应能使用所有系统功能。本次培训后，有经验的用户在系统使用中的平均错误次数不应超过每小时两次。（可测试的非功能性需求）

用于指定非功能性需求的指标

财产	措施
速度	处理的事务/秒用户/事件响应时间屏幕刷新时间
尺寸	兆字节ROM芯片数量
便于使用	训练时间帮助框架的数量
可靠性	平均故障时间不可用的概率故障发生率可用性
稳健性	失败后重新启动的时间导致失败的事件百分比失败时数据损坏的概率
可移植性	目标相关陈述的百分比目标系统数量

域要求

- 系统的操作域对系统提出了要求。
    - 例如，列车控制系统必须考虑不同天气条件下的制动特性。
- 领域需求是新的功能需求、对现有需求的约束或定义特定的计算。
- 如果不满足域要求，则系统可能无法运行。

列车保护系统

- 这是列车保护系统的域要求：
- 列车的减速度应计算为：
    - Dtrain = Dcontrol + Dgradient
    - 其中 Dgradient 是 9.81m/s2 * 补偿梯度/alpha，其中 9.81m/s2 /alpha 的值对于不同类型的列车是已知的。
- 非专业人士很难理解这其中的含义以及它如何与其他需求相互作用。

域需求问题

- 易懂
    - 需求以应用领域的语言表达；
    - 开发系统的软件工程师通常不理解这一点。
- 隐含性
    - 领域专家非常了解该领域，以至于他们不会考虑明确领域需求。

关键点

- 软件系统的需求规定了系统应该做什么，并定义了对其操作和实现的约束。
- 功能需求是对系统必须提供的服务的陈述，或者是对必须如何执行某些计算的描述。
- 非功能性需求通常会限制正在开发的系统和正在使用的开发过程。
- 它们通常与系统的涌现属性有关，因此适用于整个系统。

第 4 章——需求工程

第二讲

4.2 软件需求文档

- 软件需求文档是对系统开发人员的要求的官方声明。
- 应包括用户需求的定义和系统需求的规范。
- 它不是设计文档。尽可能地，它应该设置系统应该做什么而不是它应该如何做。

敏捷方法和要求

- 许多敏捷方法认为，由于需求变化如此之快，因此生成需求文档是在浪费时间。
- 因此，该文件总是过时的。
- XP 等方法使用增量需求工程并将需求表达为“用户故事”（在第 3 章中讨论）。
- 这对于业务系统来说是实用的，但对于需要大量交付前分析的系统（例如关键系统）或由多个团队开发的系统来说是有问题的。

需求文档的用户

需求文档可变性

- 需求文档中的信息取决于系统类型和使用的开发方法。
- 通常，增量开发的系统在需求文档中的细节较少。
- 已经设计了需求文档标准，例如IEEE 标准。这些主要适用于大型系统工程项目的要求。

需求文档的结构

章节	描述
前言	这应该定义文档的预期读者群并描述其版本历史，包括创建新版本的基本原理和每个版本所做更改的摘要。
介绍	这应该描述系统的需求。它应该简要描述系统的功能并解释它将如何与其他系统一起工作。它还应该描述系统如何适应委托软件的组织的整体业务或战略目标。
词汇表	这应该定义文档中使用的技术术语。您不应该对读者的经验或专业知识做出假设。
用户需求定义	在这里，您描述为用户提供的服务。本节还应描述非功能性系统要求。此描述可能使用客户可以理解的自然语言、图表或其他符号。应规定必须遵循的产品和过程标准。
系统架构	本章应提供预期系统架构的高级概述，显示跨系统模块的功能分布。应突出显示重用的架构组件。

需求文档的结构

章节	描述
系统需求规范	这应该更详细地描述功能性和非功能性需求。如有必要，还可以向非功能性需求添加更多细节。可以定义与其他系统的接口。
系统型号	这可能包括显示系统组件与系统及其环境之间关系的图形系统模型。可能模型的示例是对象模型、数据流模型或语义数据模型。
系统进化	这应该描述系统所基于的基本假设，以及由于硬件进化、不断变化的用户需求等而导致的任何预期变化。本节对系统设计人员很有用，因为它可以帮助他们避免会限制未来可能对系统进行更改的设计决策。
附录	这些应提供与正在开发的应用程序相关的详细、具体的信息；例如，硬件和数据库描述。硬件要求定义了系统的最小和最佳配置。数据库需求定义了系统使用的数据的逻辑组织以及数据之间的关系。
指数	可以包括文档的多个索引。除了正常的字母索引外，还可能有图表索引、功能索引等。

4.3 需求规范

- 在需求文档中写下用户和系统需求的过程。
- 没有技术背景的最终用户和客户必须能够理解用户要求。
- 系统要求是更详细的要求，可能包括更多的技术信息。
- 要求可能是系统开发合同的一部分
    - 因此，这些尽可能完整是很重要的。

编写系统需求规范的方法

符号	描述
自然语言	要求是使用自然语言中的编号句子编写的。每个句子应表达一个要求。
结构化自然语言	需求以自然语言写在标准表格或模板上。每个字段提供有关需求的一个方面的信息。
设计描述语言	这种方法使用类似于编程语言的语言，但具有更抽象的特征，通过定义系统的操作模型来指定需求。这种方法现在很少使用，尽管它对于接口规范很有用。
图形符号	图形模型，辅以文本注释，用于定义系统的功能需求；UML 用例和序列图是常用的。
数学规格	这些符号基于数学概念，例如有限状态机或集合。尽管这些明确的规范可以减少需求文档中的歧义，但大多数客户并不了解正式的规范。他们无法检查它是否代表了他们想要的东西，也不愿意接受它作为系统合同。

要求和设计

- 原则上，需求应该说明系统应该做什么，而设计应该描述它是如何做到这一点的。
- 在实践中，需求和设计是分不开的
    - 可以设计系统架构来构建需求；
    - 该系统可以与产生设计要求的其他系统互操作；
    - 使用特定架构来满足非功能性需求可能是领域需求。
        - 这可能是监管要求的结果。

4.3.1 自然语言规范

- 需求写成自然语言句子，并辅以图表和表格。
- 用于编写需求，因为它具有表现力、直观和通用性。这意味着用户和客户可以理解需求。

写作要求指南

- 发明一种标准格式并将其用于所有要求。
- 以一致的方式使用语言。使用“shall”表示强制性要求，“should”表示理想的要求。
- 使用文本突出显示来识别需求的关键部分。
- 避免使用计算机术语。
- 包括对为什么需要某项要求的解释（理由）。

自然语言的问题

- 缺乏清晰度
    - 在不使文档难以阅读的情况下，精度是很困难的。
- 需求混乱
    - 功能性和非功能性需求往往会混淆。
- 需求合并
    - 几个不同的要求可以一起表达。

胰岛素泵软件系统的示例要求

3.2 如果需要，系统应每 10 分钟测量一次血糖并输送胰岛素。*（血糖的变化相对较慢，因此不需要更频繁的测量；不频繁的测量可能会导致不必要的高血糖水平。）*3.6 系统应每分钟运行一次自检程序，其中包含要测试的条件和定义的相关操作。*（自检例程可以发现硬件和软件问题，并提醒用户可能无法正常操作。）*

4.3.2 结构化规格

- 一种编写需求的方法，其中需求编写者的自由受到限制，并且需求以标准方式编写。
- 这适用于某些类型的需求，例如嵌入式控制系统的需求，但有时对于编写业务系统需求来说过于严格。

基于表单的规范

- 功能或实体的定义。
- 输入及其来源的描述。
- 输出的描述及其去向。
- 有关计算所需的信息和使用的其他实体的信息。
- 要采取的行动的描述。
- 前置和后置条件（如果适用）。
- 函数的副作用（如果有的话）。

胰岛素泵要求的结构化规范

胰岛素泵要求的结构化规范

表格规格

- 用于补充自然语言。
- 当您必须定义许多可能的替代行动方案时特别有用。
- 例如，胰岛素泵系统根据血糖水平的变化率进行计算，表格规范解释了如何计算不同场景的胰岛素需求。

胰岛素泵计算的表格规范

健康）状况	行动
糖水平下降 (r2 < r1)	复合剂量 = 0
糖水平稳定 (r2 = r1)	复合剂量 = 0
糖水平增加和增加率降低 ((r2 – r1) < (r1 – r0))	复合剂量 = 0
糖水平增加和增加率稳定或增加 ((r2 – r1) ≥ (r1 – r0))	CompDose = 轮 ((r2 – r1)/4)如果舍入结果 = 0 那么 CompDose = 最小剂量

4.4 需求工程过程

- 用于 RE 的流程因应用领域、涉及的人员和开发需求的组织而异。
- 但是，所有流程都有许多通用活动
    - 需求获取；
    - 需求分析;
    - 需求验证；
    - 需求管理。
- 在实践中，RE 是一种迭代活动，其中这些过程是交错的。

需求工程过程的螺旋视图

4.5 需求获取和分析

- 有时称为需求引出或需求发现。
- 涉及与客户一起工作的技术人员，以了解应用领域、系统应提供的服务以及系统的操作限制。
- 可能涉及最终用户、经理、参与维护的工程师、领域专家、工会代表等。这些被称为*利益相关者*。

需求分析的问题

- 利益相关者不知道他们真正想要什么。
- 利益相关者以他们自己的方式表达要求。
- 不同的利益相关者可能有相互冲突的要求。
- 组织和政治因素可能会影响系统要求。
- 需求在分析过程中发生变化。新的利益相关者可能会出现，商业环境可能会发生变化。

需求获取和分析

- 软件工程师与一系列系统涉众一起工作，以了解应用领域、系统应提供的服务、所需的系统性能、硬件约束、其他系统等。
- 阶段包括：
    - 需求发现，
    - 需求分类和组织，
    - 需求优先级和协商，
    - 要求规范。

需求获取和分析过程

流程活动

- 需求发现
    - 与利益相关者互动以发现他们的需求。在此阶段还会发现域需求。
- 需求分类和组织
    - 获取非结构化的需求集合，将相关需求分组，并将它们组织成连贯的集群。
- 优先排序和协商
    - 确定需求的优先级并解决需求冲突。
- 要求规范
    - 需求被记录并输入到下一轮螺旋中。

需求获取问题

- 利益相关者不知道他们真正想要什么。
- 利益相关者以他们自己的方式表达要求。
- 不同的利益相关者可能有相互冲突的要求。
- 组织和政治因素可能会影响系统要求。
- 需求在分析过程中发生变化。新的利益相关者可能会出现，商业环境也会发生变化。

关键点

- 软件需求文档是对系统需求的一致同意的声明。应该对它进行组织，以便系统客户和软件开发人员都可以使用它。
- 需求工程过程是一个迭代过程，包括需求获取、规格说明和验证。
- 需求获取和分析是一个迭代过程，可以表示为一系列活动——需求发现、需求分类和组织、需求协商和需求文档。

第 4 章——需求工程

第三讲

4.5.1 需求发现

- 收集有关所需系统和现有系统的信息并从该信息中提取用户和系统要求的过程。
- 利益相关者的范围从系统的最终用户到管理人员，再到外部利益相关者，例如监管者，他们证明系统的可接受性。
- 系统通常有一系列利益相关者。

MHC-PMS 的利益相关者

- 信息记录在系统中的患者。
- 医生 谁是负责评估和治疗的病人。
- 与医生协调会诊并进行一些治疗的护士。
- 管理患者预约的医疗接待员。
- 负责安装和维护系统的 IT 人员。

MHC-PMS 的利益相关者

- 必须确保系统符合当前患者护理道德准则的医学道德经理。
- 从系统中获取管理信息的卫生保健管理者。
- 负责确保系统信息可以得到维护和保存，并且记录保存程序已正确实施的病历工作人员。

4.5.2 采访

- 与利益相关者的正式或非正式访谈是大多数可再生能源流程的一部分。
- 面试类型
    - 基于预先确定的问题列表的封闭式面试
    - 与利益相关者探讨各种问题的公开访谈。
- 有效的面试
    - 思想开放，避免对需求有先入为主的想法，并愿意倾听利益相关者的意见。
    - 使用跳板问题、需求提案或通过在原型系统上合作来提示受访者进行讨论。

实战面试

- 通常是封闭式和开放式面试的混合。
- 访谈有助于全面了解利益相关者做什么以及他们如何与系统交互。
- 面试不利于理解领域需求
    - 需求工程师无法理解特定领域的术语；
    - 一些领域知识是如此熟悉，以至于人们发现很难表达或认为它不值得表达。

4.5.3 场景

- 场景是如何使用系统的真实示例。
- 他们应该包括
    - 开始情况的描述；
    - 对正常事件流的描述；
    - 可能出错的描述；
    - 有关其他并发活动的信息；
    - 场景完成时的状态描述。

在 MHC-PMS 中收集病史的场景

在 MHC-PMS 中收集病史的场景

4.5.4 用例

- 用例是 UML 中基于场景的技术，它识别交互中的参与者并描述交互本身。
- 一组用例应该描述与系统的所有可能的交互。
- 高级图形模型辅以更详细的表格描述（见第 5 章）。
- 序列图可用于通过显示系统中事件处理的序列来向用例添加细节。

MHC-PMS 的用例

4.5.5 民族志

- 社会科学家花费大量时间观察和分析人们的实际工作方式。
- 人们不必解释或阐明他们的工作。
- 可以观察到重要的社会和组织因素。
- 民族志研究表明，工作通常比简单系统模型所暗示的更丰富、更复杂。

民族志的范围

- 源自人们实际工作方式的需求，而不是流程定义表明他们应该工作的方式。
- 源于合作和了解他人活动的要求。
    - 意识到其他人在做什么会导致我们做事的方式发生变化。
- 人种学对于理解现有流程很有效，但无法识别应添加到系统中的新功能。

重点民族志

- 在研究空中交通管制过程的项目中开发
- 将民族志与原型设计相结合
- 原型开发导致未解决的问题，这些问题侧重于人种学分析。
- 民族志的问题在于它研究现有的实践，这些实践可能具有一些不再相关的历史基础。

用于需求分析的人种学和原型设计

4.6 需求验证

- 关注证明需求定义了客户真正想要的系统。
- 需求错误成本很高，因此验证非常重要
    - 交付后修复需求错误的成本可能高达修复实现错误的成本的 100 倍。

需求检查

- 有效性。系统是否提供最能支持客户需求的功能?
- 一致性。是否有任何要求冲突?
- 完整性。是否包括客户要求的所有功能?
- 现实主义。在可用的预算和技术的情况下，这些要求能否得到实施
- 可验证性。可以检查要求吗?

需求验证技术

- 需求评审
    - 对需求进行系统的手动分析。
- 原型制作
    - 使用系统的可执行模型来检查需求。第 2 章中介绍。
- 测试用例生成
    - 为需求开发测试以检查可测试性。

需求评审

- 在制定需求定义时应进行定期审查。
- 客户和承包商员工都应参与审查。
- 审查可以是正式的（有完整的文件）或非正式的。开发人员、客户和用户之间的良好沟通可以在早期解决问题。

复核支票

- 可验证性
    - 需求是否可实际测试?
- 可理解性
    - 是否正确理解了要求?
- 可追溯性
    - 是否明确说明了要求的来源?

- 适应性
    - 是否可以在不影响其他需求的情况下更改需求?

4.7 需求管理

- 需求管理是在需求工程过程和系统开发过程中管理不断变化的需求的过程。
- 新的需求随着系统的开发和投入使用而出现。
- 您需要跟踪各个需求并维护相关需求之间的链接，以便您可以评估需求更改的影响。您需要建立一个正式的流程来提出变更建议并将这些建议与系统需求联系起来。

不断变化的要求

- 系统的业务和技术环境在安装后总是会发生变化。
    - 可能会引入新硬件，可能需要将系统与其他系统连接，业务优先级可能会发生变化（随之而来的是所需系统支持的变化），并且可能会引入系统必须遵守的新立法和法规。
- 为系统付费的人和该系统的用户很少是同一个人。
    - 由于组织和预算限制，系统客户提出了要求。这些可能与最终用户的要求相冲突，并且在交付后，如果系统要实现其目标，则可能必须添加新功能以提供用户支持。

不断变化的要求

- 大型系统通常具有多样化的用户社区，许多用户具有不同的需求和优先级，这些需求和优先级可能相互冲突或矛盾。
    - 最终的系统需求不可避免地是它们之间的折衷，并且根据经验，经常发现必须改变给予不同用户的支持的平衡。

需求演变

4.7.1 需求管理计划

- 建立所需的需求管理细节级别。
- 需求管理决策：
    - *需求标识*每个需求都必须唯一标识，以便可以与其他需求交叉引用。
    - *变更管理流程*这是一组评估变更影响和成本的活动。我将在下一节中更详细地讨论这个过程。
    - *可追溯性策略*这些策略定义了每个需求之间以及需求和应该记录的系统设计之间的关系。
    - *工具支持*可以使用的工具范围从专业需求管理系统到电子表格和简单的数据库系统。

4.7.2 需求变更管理

- 决定是否应接受需求变更
    - *问题分析和变更说明*
        - 在此阶段，将分析问题或变更建议以检查其是否有效。此分析将反馈给变更请求者，变更请求者可

能会以更具体的需求变更提案进行响应，或者决定撤回请求。

- *变更分析和成本核算*
  - 使用可追溯性信息和系统要求的一般知识来评估提议更改的影响。完成此分析后，将决定是否继续进行需求更改。
- *变更实施*
  - 需求文档以及系统设计和实现在必要时被修改。理想情况下，应组织该文档，以便可以轻松实施更改。

需求变更管理

关键点

- 您可以使用一系列技术来获取需求，包括访谈、场景、用例和人种学。
- 需求验证是检查需求的有效性、一致性、完整性、现实性和可验证性的过程。
- 业务、组织和技术的变化不可避免地会导致软件系统需求的变化。需求管理是管理和控制这些变更的过程。