ISC
CCS

**T/CDATS**

成 都 市 服 务 贸 易 协 会 团 体 标 准

T/CDATS 002-2023

# 对日软件外包交付标准

Delivery standards for software outsourcing to Japan

2023-XX-XX 发布　　2023-XX-XX 实施

成都服务贸易协会、音泰思计算机技术（成都）有限公司 、四川吉利学院联合发布

目次

前言.....4
引言.....5
1 范围.....6
2 需求分析阶段.....6
2.1 需求文档.....6
2.2 需求可追述矩阵.....6
2.3 需求验证.....6
3 设计阶段.....6
3.1 设计文档.....6
3.2 用户界面(UI)和用户体验(UX)设计.....6
3.3 系统架构图和数据流图.....6
3.4 设计评审.....6
3.5 测试计划.....6
4 开发阶段.....6
4.1 源代码.....7
4.2 单元测试.....7
4.3 代码审查.....7
4.4 版本控制.....7
4.5 集成.....7
4.6 编码文档.....7
5 测试阶段.....7
5.1 测试计划和测试用例.....7
5.2 测试结果和报告.....7
5.3 缺陷跟踪.....7
5.4 回归测试.....7
5.5 性能测试.....7
5.6 测试环境.....7
5.7 自动化测试.....8
6 文档.....8
6.1 需求文档.....8
6.2 设计文档.....8
6.3 代码文档.....8
6.5 测试文档.....8
6.6 安装和配置指南.....8
6.7 发布说明和版本 6.3 控制文档.....8
7 培训.....8
7.1 培训材料的制作.....8
7.2 培训课程的设计.....8
7.3 培训课程的实施.....9
7.4 培训效果的评估.....9
7.5 持续支持.....9
8 交付和部署.....9
8.1 预部署检查.....9

8.2 部署计划.....9
8.3 安装和配置.....9
8.4 数据迁移.....9
8.5 系统测试.....9
8.6 用户接受测试.....9
8.7 正式交付.....9
8.8 后续支持.....9
9 支持和维护.....10
9.1 用户支持.....10
9.2 问题跟踪.....10
9.3 错误修复.....10
9.4 版本更新.....10
9.5 系统优化.....10
9.6 文档更新.....10
9.7 持续学习.....10
10 各阶段交付清单.....10

# 前言

本文件按照GB/T 1.1—2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由成都服务外包协会、成都音泰思计算机技术（成都）有限公司、四川吉利学院提出。

本文件起草单位：成都音泰思计算机技术（成都）有限公司、成都服务贸易协会、四川吉利学院。

本文件由成都服务贸易协会归口。

本文件主要起草人：胡翔、阳宇、饶佳莹、曾芸芸、李志远、施一飞、胡荣

# 引言

软件的交付是继软件的需求，设计，编码，测试等阶段之后的一个核对用户需求，检验软件产品，面向客户实施应用的阶段。本阶段后期的工作主旨在于：通过对软件产品客户方安装，应用及维护，收集软件产品运行期出现的问题，及时反馈用户的使用信息，并转化为软件产品的升级换代的重要性材料。

这个重要的阶段如何顺利有效完整的进行，本文着重描述软件开发各个开发阶段以及开发相关过程的交付标准情况。

软件交付标准需要考虑多个因素的过程，包括开发流程、代码质量、测试和验证、文档、培训、支持和维护等。以下是详细阐述各个阶段交付内容，附带交付文档清单，可根据项目实际的工作范围和特性自行裁剪交付内容。

## 1. 范围

本文将按照对日软件外包常规阶段划分方式描述各阶段的主要交付工作及输出，包含需求分析、设计、开发、测试、文档、交付和部署、培训和维护几个阶段

## 2. 需求分析阶段

在需求分析和设计阶段，交付标准会帮助团队确保在完成这个阶段后，产出的工作成果能满足项目目标和需求。以下是一些可能的交付标准：

**2.1 需求文档：** 这是需求分析阶段的主要交付物。文档应包含清晰、明确、完整且可测试的需求。所有需求应详细说明，并且应能与相关利益相关者的需求对应起来。

**2.2 需求可追溯矩阵：** 需求可追溯矩阵用于跟踪需求，确保所有需求都得到满足，而且可以追溯到其来源。此外，它可以帮助确保每个需求都在设计和开发过程中得到解决。

**2.3 需求验证：** 需求应通过与关键利益相关者的审查会议进行验证，以确保它们准确地反映了所需的功能和性能。

## 3. 设计阶段

**3.1 设计文档：** 设计文档应详细说明系统的架构、组件、接口和数据模型。设计应清晰、一致且完整，以便开发团队可以根据其进行编码。

**3.2 用户界面(UI)和用户体验(UX)设计：** UI/UX 设计应该易于理解，直观，让用户可以简单地与系统进行交互。这可能包括原型，用户流程图，界面设计稿等。

**3.3 系统架构图和数据流图：** 这些图表提供了系统的视觉表示，帮助团队理解系统的工作原理，以及各部分如何协同工作。

**3.4 设计评审：** 设计应通过团队的评审，确保其符合需求并可以实现。设计评审还应检查设计的完整性、一致性和质量。

**3.5 测试计划：** 在设计阶段，应开始规划测试活动。这可能包括定义测试策略、制定测试计划和设计测试用例。

在达到这些交付标准后，你的团队就可以进入下一个阶段，即实施或编码阶段了。

## 4. 开发阶段：

开发阶段是软件开发过程中的核心阶段，主要涉及实际的编码和单元测试。以下是这个阶段的一些可能的交付标准：

**4.1 源代码：** 开发阶段的主要交付物是完成的源代码。代码应该清晰、整洁，符合编码规范，并且有充足的注释。这样，其他开发人员能够理解和维护代码。

**4.2 单元测试：** 对于每个功能模块或代码单元，都应编写和执行单元测试。测试结果应全部通过，以验证代码的正确性。

**4.3 代码审查：** 代码审查是一种质量控制措施，通过同行评审来查找并修复代码中的错误。代码审查应该在整个开发过程中定期进行。

**4.4 版本控制：** 所有的源代码应当存储在版本控制系统中，如 Git。这样，可以跟踪每个文件的修改历史，以及谁做了哪些修改。

**4.5 集成：** 代码应被整合到主分支，通过集成测试来确保各个模块可以正确地一起工作。构建和部署： 应创建自动化的构建和部署流程，以便可以快速并且一致地将代码从开发环境转移到测试或生产环境。

**4.6 编码文档：** 编码文档是一个重要的参考资料，它可以帮助其他开发人员理解代码的功能和工作原理。

以上是开发阶段的一些交付标准，不过具体标准可能会因项目的具体需求和团队的实践而有所不同。在开发过程中，一定要确保代码的质量，以便在后续阶段减少修改和调试的工作量。

5. **测试阶段：**

测试阶段是确保软件满足所有功能和非功能要求的关键阶段。以下是这个阶段的一些可能的交付标准：

**5.1 测试计划和测试用例：** 应该为软件的每一个功能和非功能要求编写详尽的测试计划和测试用例。这些文档应该清晰地描述了如何测试特定的功能或性能指标，并明确了预期的结果。

**5.2 测试结果和报告：** 执行测试后，应该生成测试结果和报告，明确指出测试的成功与否。如果测试失败，报告应该详述失败的原因，以及如何重现问题。

**5.3 缺陷跟踪：** 所有在测试中发现的缺陷都应该被记录在缺陷跟踪系统中，如 Jira。每个缺陷都应该有详细的描述、重现步骤，以及影响的严重性和优先级。

**5.4 回归测试：** 修复缺陷后，应该重新执行相关的测试用例，以确认问题已经被解决。这称为回归测试。

**5.5 性能测试：** 对于需要满足特定性能指标的软件，应该进行性能测试，如响应时间、并发处理能力、资源消耗等。

**5.6 测试环境：** 应提供与生产环境尽可能接近的测试环境，以降低环境差异带来的风险。

**5.7 自动化测试：** 尽可能使用自动化测试工具进行测试，以提高测试的效率和准确性。测试阶段的目标是确保软件在所有预期的使用条件下都能正常工作，以及及时发现和修复所有的缺陷。注意，测试阶段并非只发生在软件开发周期的最后阶段，而是应该贯穿于整个软件开发过程中。

6. **文档：**

在软件交付过程中，文档是一个非常关键的组成部分。以下是一些应该考虑的主要文档交付标准：

**6.1 需求文档：** 这是一个详细的文档，描述了开发软件的目的和目标，以及所有功能性和非功能性需求。它应包括需求的细节，如期望的用户输入、系统行为和输出。

**6.2 设计文档：** 设计文档应包括软件的整体架构、模块设计、数据模型、界面设计、算法和流程图。这个文档的目的是为开发团队提供一个清晰的软件构建蓝图。

**6.3 代码文档：** 这包括内嵌在源代码中的注释，以及可能的额外文档，描述 如何使用和维护代码。注释应清楚地描述函数、类和模块的目的，以及任何复杂代码块的工作原理。

**6.4 测试文档：** 测试文档应包括测试计划、测试用例和测试报告。这些文档应详细说明测试的目标、方法、结果和找到的问题。用户手册和在线帮助： 这些是给最终用户的文档，包括如何安装、配置、使用和排查软件问题的指南。

**6.5 安装和配置指南：** 这个文档应描述如何正确安装和配置软件以使其正常运行。
项目管理文档： 项目管理文档，如项目计划、风险管理、进度跟踪等，也是软件交付的一部分。

**6.6 发布说明和版本控制文档：** 对于每个软件发布版本，都应有一个发布说明，描述了新的功能、改进、已知问题和已修复的问题。版本控制文档应详细记录软件的版本历史和变更。

以上这些文档的目的是确保所有的利益相关方都能理解软件的需求、设计、实现和使用方法，以及对软件的未来开发和维护有一个清晰的理解。

7. **培训：**

在软件交付的过程中，培训是一个重要的环节，帮助用户或其他开发人员更好地理解和使用软件。以下是一些培训阶段可能需要考虑的关键因素：

**7.1 培训材料的制作：** 培训材料应该包括操作手册、教程、实例等，以便用户或者开发人员可以参考。这些材料需要清晰、详细，以便于理解。对于复杂的操作，可以制作步骤说明或者录制视频，让用户可以反复学习和理解。

**7.2 培训课程的设计：** 培训课程应该结构清晰，从基础功能开始，逐渐深入到高级功能。对于不同的用户群体，比如管理员、普通用户等，可以设计不同的培训课程。

**7.3 培训课程的实施：** 培训课程可以通过在线视频、现场培训、一对一指导等多种方式进行。培训时，需要耐心解答学员的问题，确保他们能够理解和掌握软件的使用方法。

**7.4 培训效果的评估：** 在培训结束后，可以通过测试、问卷等方式，对培训效果进行评估。这可以帮助改进未来的培训课程，也可以了解用户对软件的理解程度，为以后的软件更新和优化提供参考。

**7.5 持续支持：** 培训并不是一次性的，尤其是当软件更新或者新增功能时，可能需要进行进一步的培训。因此，持续提供支持和帮助，对于确保用户能够有效使用软件是非常重要的。

在软件交付过程中，培训是连接软件开发和用户的重要桥梁，因此在设计和实施培训计划时，需要认真对待，以确保用户能够顺利地使用和理解软件。

8. **交付和部署：**

在软件交付标准中，交付和部署阶段也是至关重要的。以下是一些可能需要考虑的关键因素：

**8.1.预部署检查：** 在部署之前，应该对软件进行最后一次全面的检查，确保所有的功能都能够正常工作，没有遗漏的错误或者缺陷。此外，还应该确认软件是否已经满足了所有的项目需求和目标。

**8.2 部署计划：** 应该创建一个详细的部署计划，包括部署的时间、地点、负责人、需要的资源等。部署计划应该考虑到可能遇到的风险和问题，并制定出应对策略。

**8.3 安装和配置：** 根据部署计划，进行软件的安装和配置。这可能包括设置服务器、数据库、网络等，安装软件，配置参数，等等。在安装和配置过程中，需要严格遵循相关的操作规程，确保系统的稳定和安全。

**8.4 数据迁移：** 如果需要，可能需要进行数据迁移。这包括将旧系统的数据迁移到新系统，或者将测试环境的数据迁移到生产环境。在数据迁移过程中，需要确保数据的完整和准确，避免数据丢失或者错误。

**8.5 系统测试：** 在部署完成后，应该进行系统测试，确认系统能够在真实的环境中正常工作。系统测试应该覆盖所有的功能，确保没有遗漏。

**8.6 用户接受测试：** 最后，用户应该进行用户接受测试，确认软件是否满足他们的需求，是否易于使用。用户接受测试的结果可以作为软件交付的一个重要指标。

**8.7 正式交付：** 当所有的检查和测试都通过后，就可以正式将软件交付给用户了。交付的方式可能包括提供安装包，或者提供在线访问的地址等。

**8.8 后续支持：** 在软件交付后，应该提供一段时间的后续支持，解答用户的问题，修复可能出现的错误，等等。这可以帮助用户顺利地过渡到新的系统，并增强用户的满意度。

交付和部署阶段是软件开发过程的最后一个阶段，但也是非常重要的阶段。只有成功的交付和部署，才能确保软件能够在实际环境中正常工作，满足用户的需求。

**9. 支持和维护：**

在软件交付标准中，支持和维护阶段是至关重要的环节，主要确保软件在交付后的正常运行和持续改进。以下是一些可能需要考虑的关键因素：

**9.1 用户支持：** 提供实时的用户支持是必不可少的。这可能包括为用户提供软件使用的指导，解答用户的问题，帮助用户解决使用过程中遇到的困难。用户支持可以通过电话、电子邮件、在线聊天或社区论坛等多种方式提供。

**9.2 问题跟踪：** 应建立问题跟踪机制，用于记录和跟踪用户报告的所有问题和错误。问题跟踪系统可以帮助我们了解软件的问题和改进点，也是评价软件质量和用户满意度的重要工具。

**9.3 错误修复：** 快速有效地修复软件错误是支持和维护的重要任务。错误修复应按照优先级进行，紧急且影响广泛的错误应优先处理。同时，修复错误的过程也应记录和追踪，以便进行进一步的分析和改进。

**9.4 版本更新：** 软件需要不断进行版本更新，以修复已知的问题，增加新的功能，改善用户体验。版本更新应当有规划，遵循一定的版本控制策略，并且每次更新都应有详细的更新日志。

**9.5 系统优化：** 根据系统运行情况和用户反馈，进行持续的系统优化，包括性能优化，安全性强化等，以提高软件的质量和用户满意度。

**9.6 文档更新：** 随着软件的更新和变化，相关的用户手册，操作指南等文档也需要及时更新，确保用户可以获取到最新准确的信息。

**9.7 持续学习：** 收集和分析用户的反馈和建议，持续学习和改进，以更好地满足用户的需求，提升用户满意度。

支持和维护阶段不仅需要保证软件的正常运行，还需要通过不断的学习和改进，提升软件的质量和用户满意度，实现软件的持续优化和提升。

**10. 各阶段交付清单：（请根据客户实际要求裁剪）**

10.1 需求阶段：

概要设计，外部设计，设计指摘管理表，设计进度管理表，设计 CHECKLIST，QA 表等

10.2 设计阶段：

内部设计书，详细设计书，设计指摘管理表，设计 CHECKLIST，设计进度管理表，阶段完了报告书，QA 票等

10.3 单体阶段：

代码，单体障害管理表，代码检查表，单体指摘管理表，单体开发进度管理表，单体测试计划书，单体测试检查表，单体测试成果物，单体测试方针，阶段完了报告书

10.4 测试阶段：

连接测试方针，连接测试计划书，连接测试检查表，连接测试进度计划，连接测试障害管理表，连接测试指摘管理表，连接测试成果物，系统运用手册，系统安装手册，阶段完了报告书等。