河南中医药大学信息技术学院（智能医疗行业学院）产教协同课程建设成果

智能医学工程专业《医疗信息系统开发》课程

第05章：实现API服务开发

黄子杰

河南方和信息科技有限公司

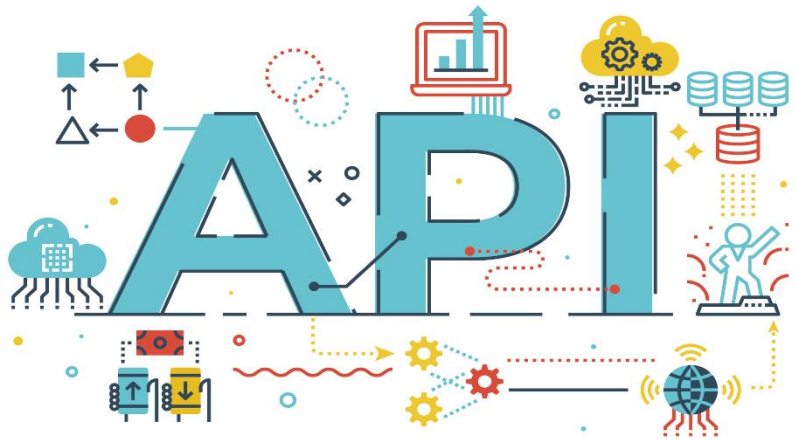
河南中医药大学信息技术学院（智能医疗行业学院）智能医疗教研室

https://aitcm.hactcm.edu.cn

2025/10/21

本章概要

- 需求分析
  - 核心基础思维
  - 技术与实现思维
- 构建软件系统
  - 工作是编码的基石
  - 保证代码质量
  - 生命周期的开始
- API服务开发
  - API服务概述
  - API设计原则
  - API开发生命周期
  - 核心组件详解



核心基础思维

这些是进行有效需求分析的底层能力，任何岗位都应具备的能力。

- 用户思维

  核心：永远站在最终用户的角度思考问题。不是“系统应该有什么功能”，而是“用户想用这个功能解决什么实际问题？”

  实践：创建用户画像、走进现场观察用户工作、进行用户访谈。问自己：“这个功能对用户的价值是什么？”

- 业务思维

  核心：理解需求背后的商业逻辑和业务目标。软件是服务于业务的工具。

  实践：学习业务术语、了解业务流程、明确项目的商业价值（如提升效率、增加收入、降低成本）。与业务方沟通时，能用他们的语言交流。

- 逻辑思维

  核心：将模糊、杂乱的用户需求，转化为清晰、结构化、无矛盾的逻辑模型。

  实践：梳理业务流程（流程图）、分析业务规则（决策表）、定义数据关系（ER图）。能发现需求中隐藏的逻辑漏洞和边界条件。

核心基础思维

- 抽象思维

  核心：忽略不必要的细节，抓住问题的本质和共性。防止陷入“过度工程”或“只见树木不见森林”的困境。

  实践：将多个具体需求归纳为一个抽象模型；识别出可复用的模块或组件；将复杂的系统分解为不同的层次（如表现层、业务层、数据层）。

- 批判性思维

  核心： 不盲目接受任何需求。对需求的合理性、真实性和可行性提出质疑。

  实践： 连续追问“为什么？”（深入挖掘问题）；挑战假设（“用户真的会这样操作吗？”）；识别并规避“解决方案式需求”（用户直接提解决方案，而非问题）。

技术与实现思维

这些思维帮助将需求高质量地转化为技术现实。

- 系统思维

  核心：理解需求不是孤立的，它会影响系统的其他部分，包括性能、安全性、可扩展性和现有功能。

  实践：进行系统影响分析；考虑非功能性需求（如并发数、响应时间）；设计松耦合的架构。

- 数据思维

  核心：从数据的角度审视需求。关注数据的产生、流动、存储、使用和消亡。

  实践：设计数据模型；定义数据校验规则；考虑数据一致性、隐私和合规性（如GDPR）。

- 可行性思维

  核心：评估需求在技术上的实现难度、成本和风险。

  实践：进行技术预研；评估第三方工具/API；给出多种技术方案并分析其利弊；对不切实际的需求提出预警和替代方案。

- 前瞻性思维

  核心：在满足当前需求的同时，为未来的扩展和变化留出空间。但这需要与“过度设计”取得平衡。

  实践：遵循设计原则（如开闭原则）；使用可扩展的架构；编写易于维护和修改的代码。

构建软件系统

编码确实只占整个项目生命周期中相对较小的一部分。

“30%”可能是一个比喻性的说法，具体比例会因项目类型、技术栈和团队成熟度而异，但它的核心思想是：**编写代码远不是软件从业者唯一的工作，甚至不是最耗时、最复杂的工作。**

- 这部分工作是编码的基石，直接决定了“要不要编码”、“为什么编码”以及“编码的方向”。
- 需求分析与梳理：正如我们上一个话题讨论的，需要与客户、用户、业务方反复沟通，挖掘真实需求，澄清模糊点，避免做错方向。
- 方案设计与架构：技术选型、系统架构设计、数据库设计、API设计、模块划分。这部分是软件的“蓝图”，糟糕的设计会导致编码事倍功半，甚至推倒重来。
- 技术预研与原型验证：对于不确定的技术难点，需要提前编写小规模的代码进行验证，但这部分代码通常不会进入最终产品。
- 任务分解与计划：将大的功能模块拆分成小的、可执行的任务，并评估工作量。

构建软件系统

这些活动与编码紧密交织，是保证代码质量的关键。

- 阅读和理解代码： 阅读别人的代码、阅读自己很久以前写的代码、阅读依赖库的文档和源码。程序员大部分时间是在“读代码”，而不是“写代码”。
- 调试与问题排查： 解决Bug、分析日志、使用调试工具。这个过程可能极其耗时，远比写入那行引发Bug的代码要长。
- 团队沟通与协作： 代码审查、与同事讨论技术方案、同步接口、站会等。
- 学习与研究： 学习新的框架、库、工具，查阅官方文档和技术博客。

构建软件系统

代码写完并提交到代码库，只是它生命周期的开始。

- 测试：单元测试、集成测试、系统测试、用户验收测试。测试人员会花费大量时间来验证代码的正确性。
- 部署与运维：搭建部署环境、配置服务器、监控系统运行状态、处理线上故障。
- 复盘与重构：项目复盘后，可能会对现有代码进行重构，以提升其可读性、可扩展性和性能。重构也是编码，但它属于“改进”而非“新增”。
- 文档编写与维护：编写技术文档、API文档、用户手册等。

一个生动的比喻：建造一座大桥

需求分析 & 设计：勘测地形、确定桥的用途（走人还是通车）、设计桥梁结构图纸。

编码：按照图纸，浇筑桥墩、铺设桥面。（是关键的实现步骤，但只是按图施工）

测试 & 调试：进行负载测试、质量检查，发现裂缝或问题并及时修补。

部署 & 运维：大桥通车，并安排日常维护、检修、应对突发事件。

API服务开发

- 什么是API服务

  定义：Application Programming Interface，应用程序编程接口。

  本质：软件组件之间的契约，定义如何相互通信

- API服务的核心价值

  解耦：前端与后端独立演进

  复用：同一服务支持多终端（Web、App、桌面）

  标准化：统一的交互模式降低学习成本

  生态建设：开放API构建开发者生态

API服务开发

- RESTful架构风格
  - REST的六个原则

客户端-服务器分离：关注点分离，各自独立演化

无状态： 每次请求包含所有必要信息，服务端不保存客户端会话状态

可缓存：响应必须明确标识是否可缓存，减少网络交互，提升性能

统一接口：资源标识、资源操作、自描述消息

分层系统：客户端不必知道是否连接到最终服务器，采用中间件（缓存、负载均衡、安全层）

按需代码（非必要）：服务端可以传输可执行代码

  - 资源导向设计

错误思维：基于动作

```
POST /createUser
GET /getUser?id=123
POST /updateUser
```

正确思维：基于资源

```
POST   /users          # 创建
GET    /users/123      # 获取
PUT    /users/123      # 更新（全量）
PATCH  /users/123      # 更新（部分）
DELETE /users/123      # 删除
GET    /users         # 列表
```

API服务开发

- 需求分析阶段

  1.解决关键问题

  谁是API的使用者？（移动端、Web端、第三方开发者）

  解决什么业务问题？

  预期的调用频率和性能要求？

  数据敏感度和安全要求？

  2.产出物

  API功能清单

  数据模型草图

  业务草图

API服务开发

- 设计阶段
  1.接口设计
  RESTful资源规划
  请求/响应格式定义
  错误码规范设计
  2.技术设计
  架构模式选择
  数据库选型和技术栈
  认证授权方案

API服务开发

- 实现阶段

  分层架构示例

```
表现层（Presentation Layer）
    ↓
业务逻辑层（Business Logic Layer）
    ↓
数据访问层（Data Access Layer）
    ↓
持久化层（Persistence Layer）
```

API服务开发

- 测试与文档
  - 单元测试（基础）
  - 集成测试（中间）
  - E2E测试（顶端）

API服务开发

- 路由设计

  RESTful资源命名规范：

  使用名词复数形式：/users 而非 /getUser

  层次关系：/users/{userId}/orders

  过滤分页：/users?page=1&limit=20&status=active

  避免动词：使用HTTP方法表达动作

- 请求/响应设计

请求设计：

```json
// 良好的POST请求体
{
  "user": {
    "name": "张三",
    "email": "zhang@example.com"
  }
}
```

响应设计：

```json
// 成功响应
{
  "code": 200,
  "data": {
    "id": "123",
    "name": "张三",
    "email": "zhang@example.com"
  },
  "message": "success"
}
```

```
// 错误响应
{
  "code": 400,
  "error": "INVALID_EMAIL",
  "message": "邮箱格式不正确",
  "details": {...}
}
```

本章作业

- 使用若依框架(前后端分离)完成患者健康档案管理功能

  要求:

  1.使用本地数据库并创建患者表。

  2.业务场景为医生为患者建档,包含增删改查功能。

  3.后端使用RESTful接口接收表单数据。

信创智能医疗系统研发课程体系

河南中医药大学信息技术学院（智能医疗行业学院）

河南中医药大学信息技术学院（智能医疗行业学院）智能医疗教研室

河南中医药大学医疗健康信息工程技术研究所