清華大學
TSINGHUA UNIVERSITY

12/25/2019

(1) 软件：程序+数据+文档

② 特性：复杂性，一致性，可变性，不可见性

③ 挑战：
- 需求
- 风险
- 过程与监控
- 团队

④ 产生 1968 NATO

⑤ 概念：系统，规范，定量的方法用于软件

目标：创造足够好的软件

⑥ 基本要素：
- 方法：技术
- 过程：开发环节控制、管理
- 工具：支持环境

⑦ 基本目标：
- 适用
- 可靠
- 高效
- 优化折中

⑧ 质量：价值
- 正确软件，为用户价值
- 正确地做：质量，成本，周期

质量：
- 功能
- 可靠
- 易用
- 效率
- 可维护
- 可移植

⑨ 质量系统：
开发 构建 执行
仅从测试而已
- 设计
- 编码
- 测试

(2. ① 过程：流程 开发活动：
问题定义—需求开发—设计—构造—测试

② 过程模型：瀑布模型：严格文档与阶段开发
- 原型化：做原型（需求不清）
- 增量：+模块
- 迭代：扩展，完善

③ 敏捷方法：能适应迭代

scrum: ⇒ sprint
- 1-4 迭代
- 需求不改变
- 产品完成发布

④ scrum 制品
- 产品待办列表
- 迭代待办：开发任务
- 可工作软件

⑤ scrum 团队：产品负责人、敏捷教练、开发团队 ⇒ 无领导团队化、团队

⑥ 沟通：
- 看板
- 燃尽图

Backlog：大块：排序

⑦ 会议：
- 迭代计划：迭代开始，分析设计
- 站会：每日（干了什么，困难，要干什么）
- 评审：用户展示
- 回顾：开始新迭代

⑧ 敏捷估算：评估工作量
- 故事点：相对
- 理想日：小时

工作量：分解、讨论、更新、迭代速度

速度：同类估算、1-2 sprint

工作量越大越困难

(3. ① 需求：
- 业务—战略
- 用户（用户需求 ~）（自然语言）
- 系统（系统需求）（专业）

② 类型：
- 功能
- 质量（性能，非功能）

约束：
- 设计（技术）
- 过程

③ 获取的问题：
- 问题复杂
- 理解偏差
- 用户不知道
- 开发者知识

④ 需求分析

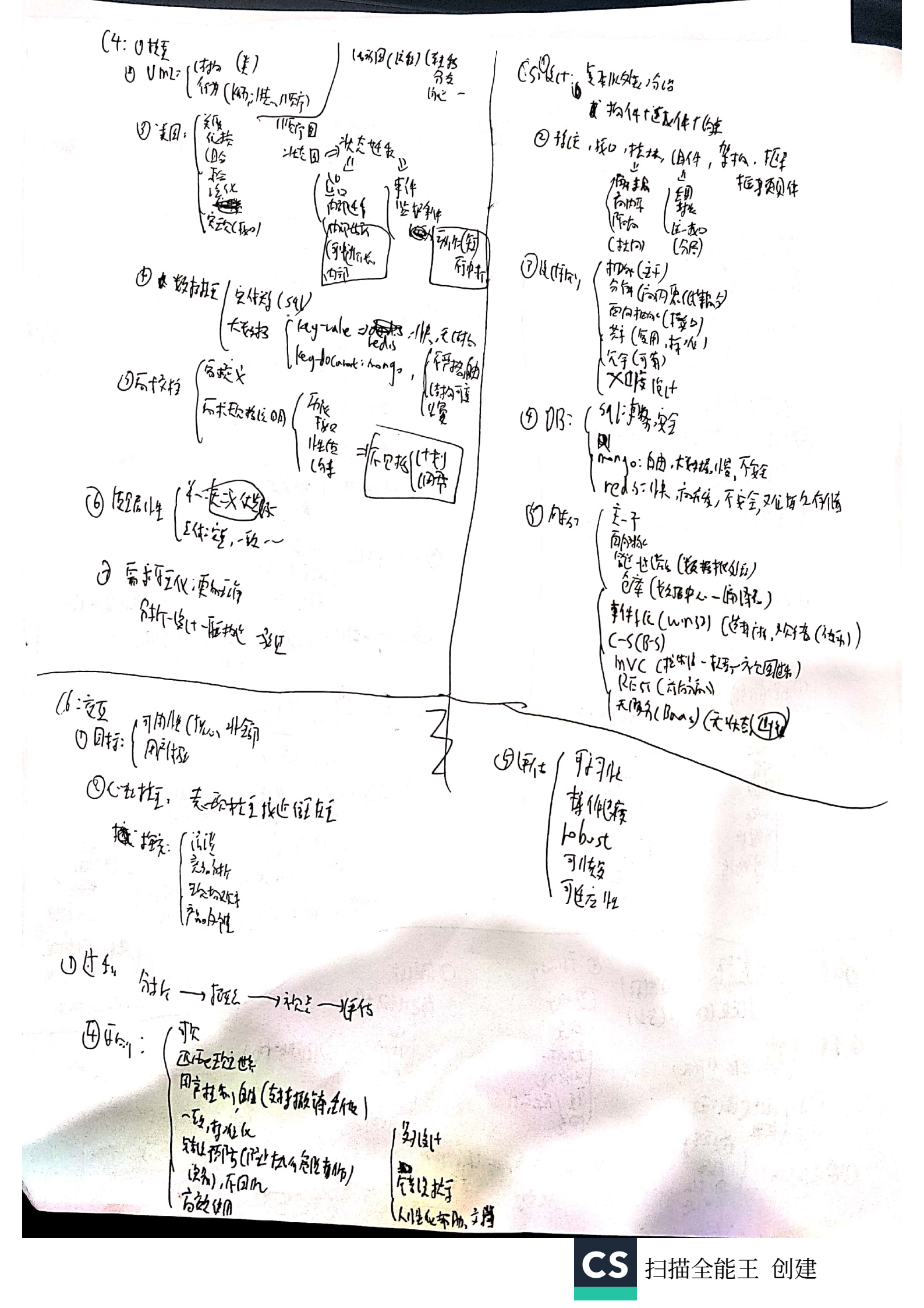
⑤ 验证
- 可读
- 需求分析
- 性能需求
- 可验证（需求分析）
- 问卷

⑥ 过程：获取需求，分析，定义，迭代验证

优先级 ——— 重要 ——— 紧急

⑦ 用户故事：
角色+活动+价值
- 独立
- 可协商
- 有价值
- 可估算
- 短小
- 可测试

⑧ 用户故事地图
重要价值
细化



(7: ① 配置管理: 控制修改

目的: 错误最小化 提高生产率

② 配置项: 文档 代码 程序 text

版本: 某个时间点上某配置项

X.Y.Z

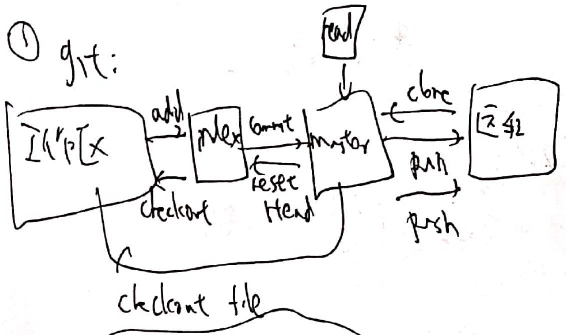
主次版

基线: 稳定版本 => 保证配置项完整正确

配置库: 存储配置项的仓库

开发库, 主库...

① git:



② init: 创建 clone: 克隆到本地 config: 环境配置

add → 暂存区 commit → 本地仓

branch, checkout {分支 / 历史版本}, merge 合并,

fetch, pull, push

③ 协作: 集中分支: 一个 central master 分支

功能分支: 维持多个分支, merge

git flow: 多分支 {master / hotfix / release / develop}

④ issue {bug / 需求 / 任务 / 功能} {优先级 / 标签} 看板: 进度管理

* 里程碑: issue 的集合

(8: 重构, 设计模式

① 改进内部结构, 不变功能

② 重复代码 过长函数 | 参数过多 (封装) 拆 | 数据类修改 (合)

⑤ 模式: 单件

工厂: 子类创建对象

策略: 多算法互换

观察者:

Facade 外观 => 封装复杂子系统

adapter 适配器 => 类接口转换

③ 设计模式 => 常见问题解决方案

④ 设计原则: {单一职责

LSP (父类子类, 契约, 前置后置条件)

~~单一职责~~

开闭原则

依赖倒置 (高层依赖抽象)

接口隔离 (ISP)}

C9: programming

① 构建可维护性

② 防御性编程 => 断言 / 异常处理

③ 编码规范:

格式: 头部 / 导入 / Main / 可读

定义 => 简短, 一致

命名 => 自文档

语句: 文件 with close

多使用列表推导

④ 代码审查, 代码检查, 单元测试, 集成测试

⑥ CI/CD:

CI (持续集成): 开发合并主干, 自动化测试

CD (持续交付): 随时可交付 (自动化)

(DevOps 快速反馈)

C6：单元测试

① 目标：(发现代码各种错误)

② 任务：~~[illegible]~~ 测

单元，集成，功能，性能，验收，安装

③ 模块接口，局部数据结构，边界条件，各独立路径，出错处理

快速执行，可维护，可验证(自动化，有结果输出)，[illegible]

④ 适用于，需求(行业标准)

⑧ 逻辑覆盖：

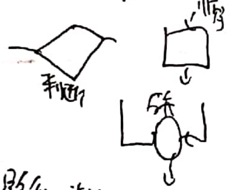
判定 

条件：条件覆盖

判定条件：判定条件覆盖

条件组合：各种条件组合

路径：控制流图所有路径



路径＝判定＋1

⑤：

方法：静态，动态

黑盒，白盒

Mock：不易获取模块

工具→替换模块

⑥ 测试用例：

值

边界值

前提

问题

优先级

名称

正常流程

非正常用例

⑦ 等价类：

尽可能少，但可覆盖

不相交

每个类一个代表

不同用例

有效，无效

边值：接近边值

健壮性：略超边值

故障例：80%→20%

⑨ 优化：

(1) 保证正确，可靠，健壮性

(2) 全局[illegible]

[illegible]

先优化[illegible]，算法

时空效率

C11：系统测试

① 手动，自动化测试

② 回归：验证修改变更不影响功能 → 上一轮迭代的全部测试

③ 测试文档：计划，用例；缺陷报告，测试报告 → 严重 → 排序；优先级 → 修复全部

④ 功能测试：(用户角度)(黑盒) 功能验证(界面，数据，操作，逻辑等)

(适合性测试，[illegible]，兼容性测试，互操作测试，内容测试)

⑤ 安全性：漏洞，防攻击

功能：做该做的

安全：不做不该做的

渗透测试

功能验证

模拟攻击

⑥ 性能测试：

响应时间(t)(2-5-8)

吞吐量(请求/t)

利用率

⑦ 负载：满足性能指标最大负载

压力：崩溃负载

⑧ 过程：[illegible] — 方案 — 执行 — 分析

1% - 2%～ —

C12：部署交付

① 交付工作：

部署到客户电脑

培训

验收

② 部署：保障上线运行

交付运行，投入使用

③ 持续交付

提交

⇓

内测 hook，合并入主干，自动化测试

(单元测试)

⇓

构建：运行

⇓

二轮：(单元，集成，[illegible])

(自动化)

⇓

部署

SE
错题

12/25/2019

清華大學
TSINGHUA UNIVERSITY

1. 软件工程方法是 ____技术上解决方法____

2. 大多数软件不易修改，除非设计时考虑了变化 (√)

3. 软件能够通过复制生成 (X)

4. 使所有变量名简短以便代码紧凑 (X)

5. 代码要优先考虑其时间效率的减少 (X) (空间瓶颈)

6. 优化不能破坏正确性 (√)

7. 代码审查前要编译通过 (√)

8. ____桩模块____ 在单元测试中代替被测模块子模块

9. 单元测试包括
- 全局数据结构 (X) (局部)
- 出错处理 (√)
- 用户界面 (X)

10. 等价类划分把输入划分为尽可能少的等价类 (X) ([illegible])
- 等价类划分唯一 (X)
- 同一等价类任一输入，输出相同 (√)
- 测试用例唯一 (X)

11. 软件构造活动包括 ____代码编写/审查____ ____测试、调试____ ____优化____

12. 瀑布模型是不适应变化的 (X)

13. 开发一个支持3D打印操作的系统 ____增量模型____

14. 计划阶段需要较多参加，因为 ____需求问题____

15. 迭代/增量模型 ____来自所有环节的目的模型____

扫描全能王 创建

16. 软件配置管理目的 控制版本

17. 面向创新产品的需求获取 竞品分析

18. B/S优势 更易部署和升级维护

19. Restful API 前后端分离

20. 大型数据：Redis
不同结构：mongo
安全事务：sql

21. 黑盒测试依据 需求规格说明 及(用例)

22. 基线目的 保证配置项完整正确