WORK LOG 3

Shandong University

March 15, 2020

高德琛

Contents

1 引言

、

《Work Log 3》收集了个人在本此实验当中的工作记录。通过本次工作日志，得以整理罗列所学原理之间的逻辑，精炼所调研内容。使之条理清晰、重点分明；从而让软件开发原理、所调研学得的知识素材，可以在小组讨论当中发挥出更大的价值。

章节 2当中首先介绍 CASE 相关定义，以及个人对于 CASE 工具的调研，不同类别工具对于 *μVloger* 项目的适应性。

接下来是个人对于敏捷模型中的开发方法以及传统软件开发方法对比调研。章节 3当中介绍了敏捷模型的相关原理 =。章节 4介绍了传统软件开发方法，并对比和总结二者的不同之处。其中最为关键的部分在于，软件开发方法如何灵活运用于我们的项目，这一问题在章节 4.3当中进行了探讨。

2 CASE

2.1 定义

计算机负责软件工程 CASE（Computer-Aided Software Engineering）是一组工具和方法的集合。是辅助软件开发的任何计算机技术：

1. 在软件开发和维护中，提供计算机辅助支持
2. 在软件开发和维护中，引入工程化方法

狭义上 CASE 指代一类特殊的软件工具，用于辅助开发、分析、测试、维护另一计算机程序和文档。而广义可以指代除了 OS 以外所有软件工具的总称。

2.2 CASE 工作台

1. 工作台是一组工具，支持分析、设计或测试等特定的软件开发阶段。
2. 工作台将一组工具组装（通过共享文件、数据结构和数据仓库等实现集成），并使这组工具可以协同工作
3. 分为开放式工作台与封闭式工作台

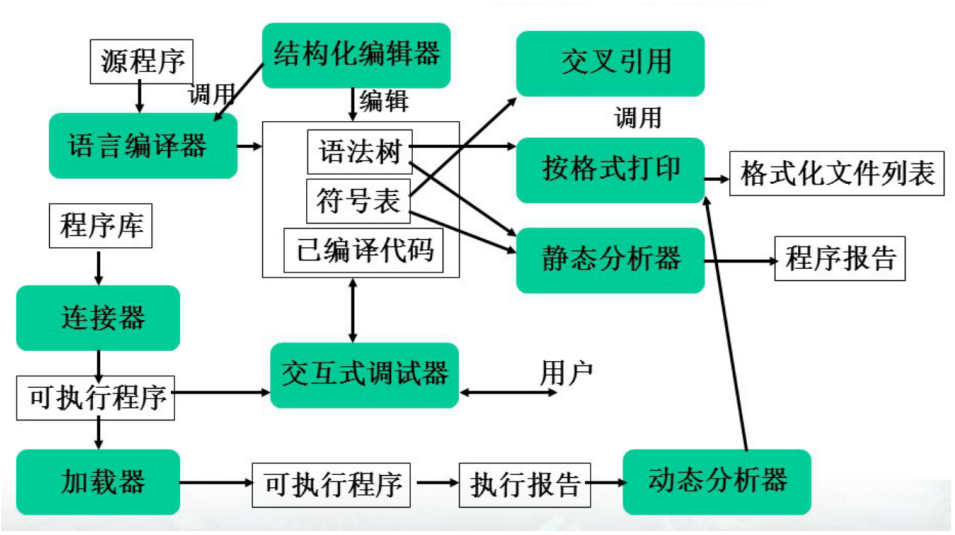

图 **1:** 程序设计工作台

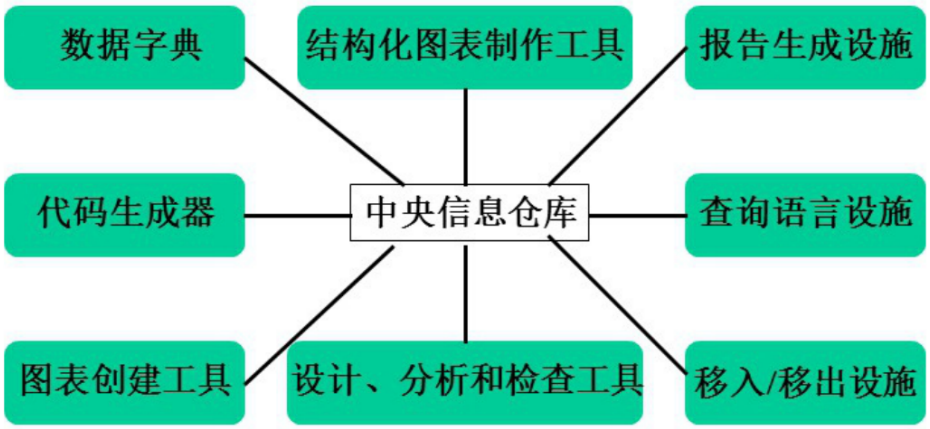

图 **2:** 设计和分析工作台

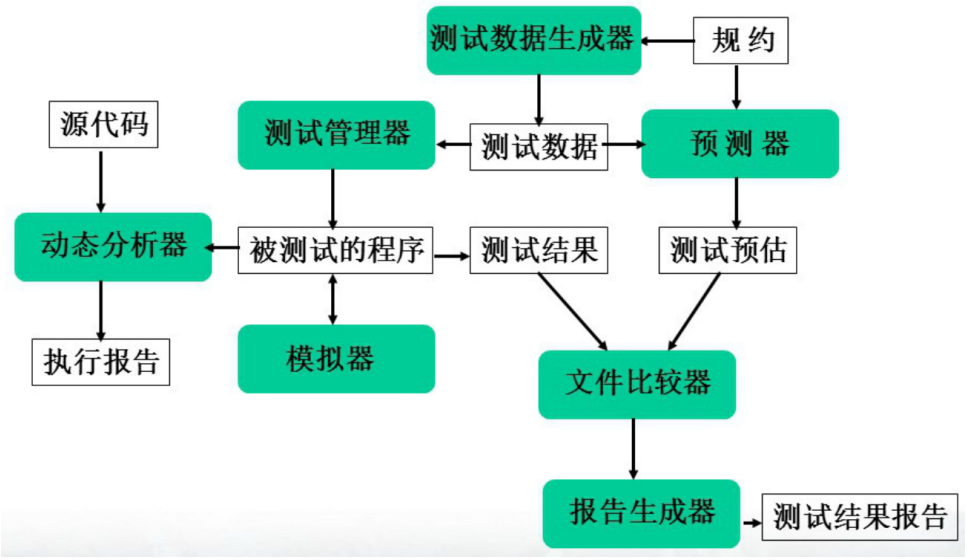

图 **3:** 测试工作台

2.3 CASE 团队选择

2.3.1 作图工具

在课设文档编写过程当中，我们队员深刻理解了“一图胜千言”这句话。严谨专业的图注往往可以大大节约读者的阅览成本。数据、流程与结构等复杂概念应尽可能采用图形式描述。

对比 ProcessOn（图 4）以及 Microsoft Office Visio（图 5）可知，Visio 作为

Office 组件，更为专业和全面。而 ProcessOn 作为 Web 客户端无需安装，浏览器打开即可作图，而且可以通过网络共享编辑。

我认为团队可以采用 ProcessOn 负责简单图示的制作，例如简单的函数流程图、概念的展示；而 Visio 用于设计复杂的图示，例如深度学习的模型设计图、系统整体的架构图。此外，Visio 不仅仅是作图工具，还是流程建模工具。

此外，考虑到键鼠操作的成本过高，而且部分原理图像更侧重于展示原理而不是图像的美关。此时可以采用 Graphviz 这类基于结构化语言的可视化工具。

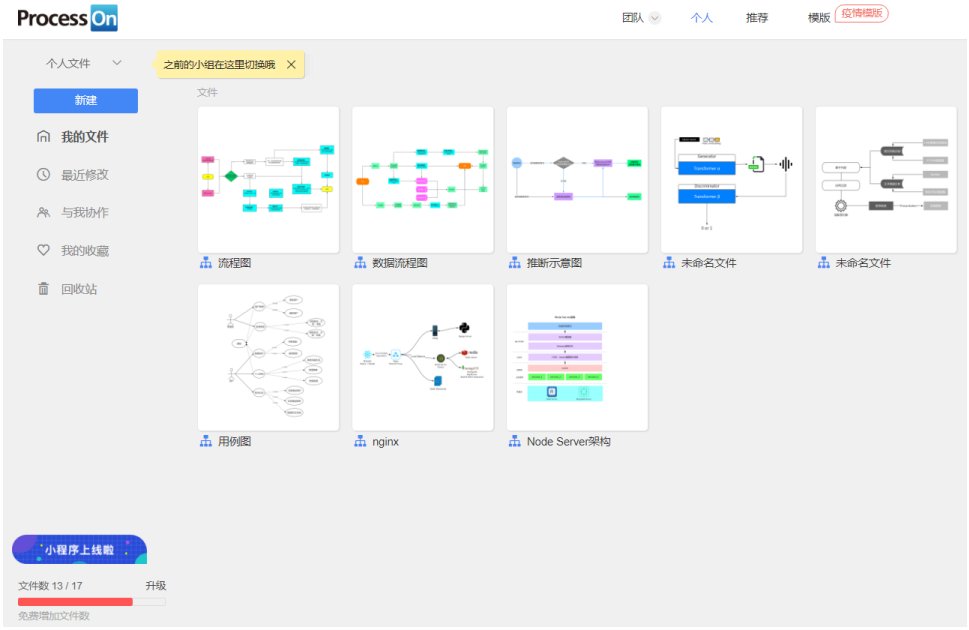

图 **4:** ProcessOn Web

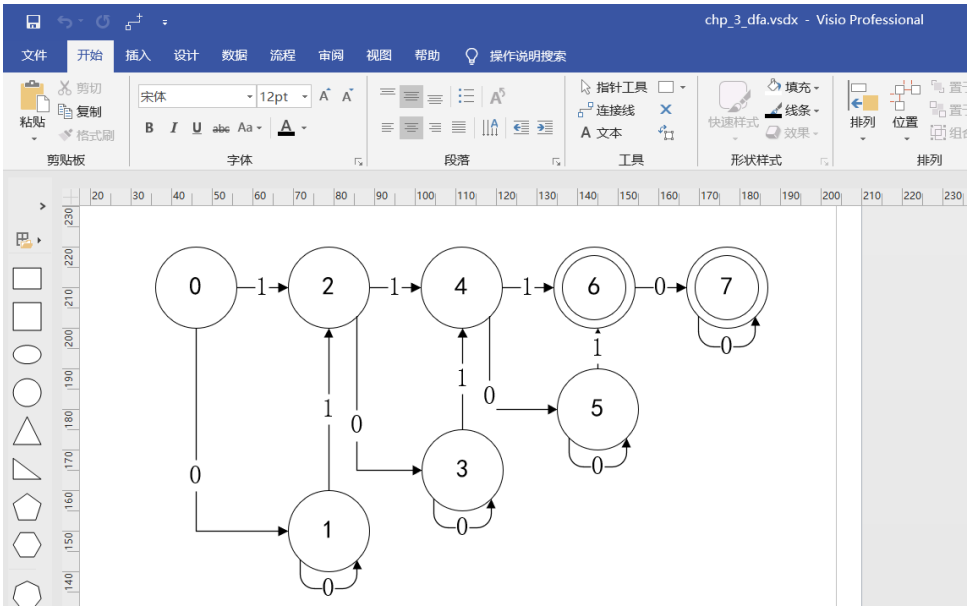

图 **5:** Microsoft Office Visio

2.3.2 项目管理工具

《工作日志 1》当中已经讨论主流的项目管理工具以及工具的选用。这些工具用于项目计划，成本和工作量估计，项目调度和资源规划。经理人必须严格遵守项目执行与软件项目管理的每提及一步。项目管理工具可以帮助存储和整个组织共享项目信息的实时性。我们团队选用的是 Teambition（图 6），Teambition 支持线上组织企业和项目管理。

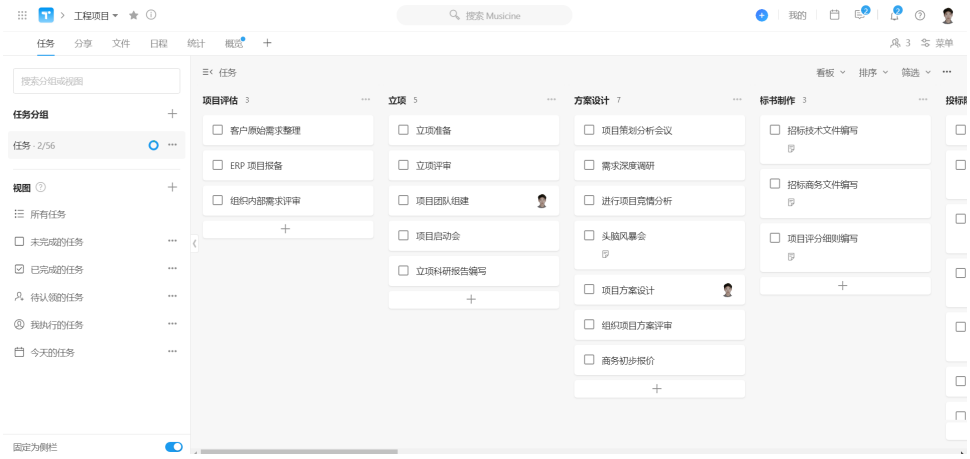

图 **6:** Teambition

2.3.3 配置管理工具

此类工具用于管理软件实例，进行版本控制和软件发布。我们选择了最主流的 Git 工具，使用官网 GitHub 进行托管（图 7）。Git 允许我们团队进行共享代码上的共享合作，同时可以按时序控制软件版本、源码变更。

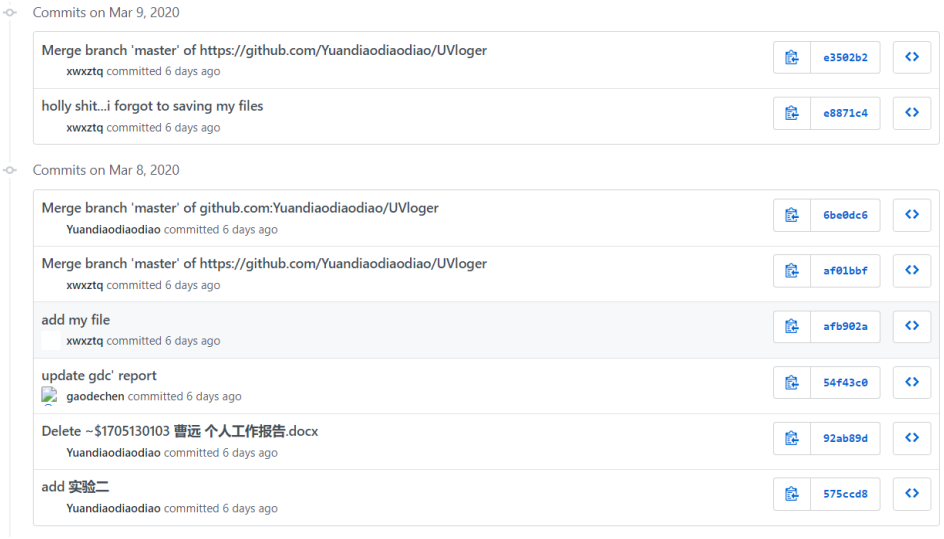

图 **7:** GitHub Commit History

2.3.4 文档工具

文档生成工具，用于生成面向用户以及开发者文档。例如面向用户的参考手册，培训手册，安装手册；面向开发者的 API 文档。

对于面向开发者的文档，我们选用了自文档化工具 Doxygen，用于生成 API 文档，提高团队内源码开发交流的规范性、效率。对于面向用户的文档、面向需求方的文档，重点在于文档格式、文字的严谨和专业，我们采用共享文档以及 LaTeX 结合的方式编辑和管理文档（图 8）。

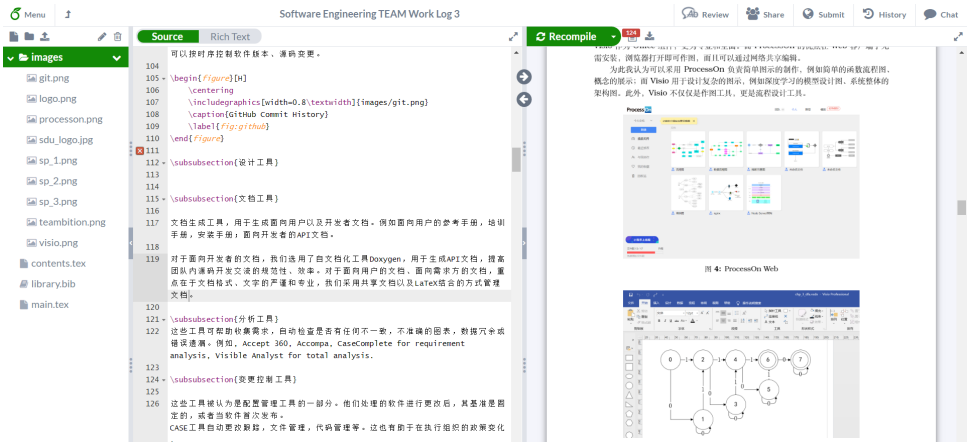

图 **8:** Overleaf LaTeX Online

2.3.5 原型开发工具

原型 CASE 工具具有图形库，可以创建独立于硬件的用户界面设计。这些工具可以帮助我们根据现有的信息来建立快速原型。我们团队选用了 Axure（图 9）作为原型工具，并且参照设计原理、美学原理旨在设计出符合 Vloger 理念的前端原型。

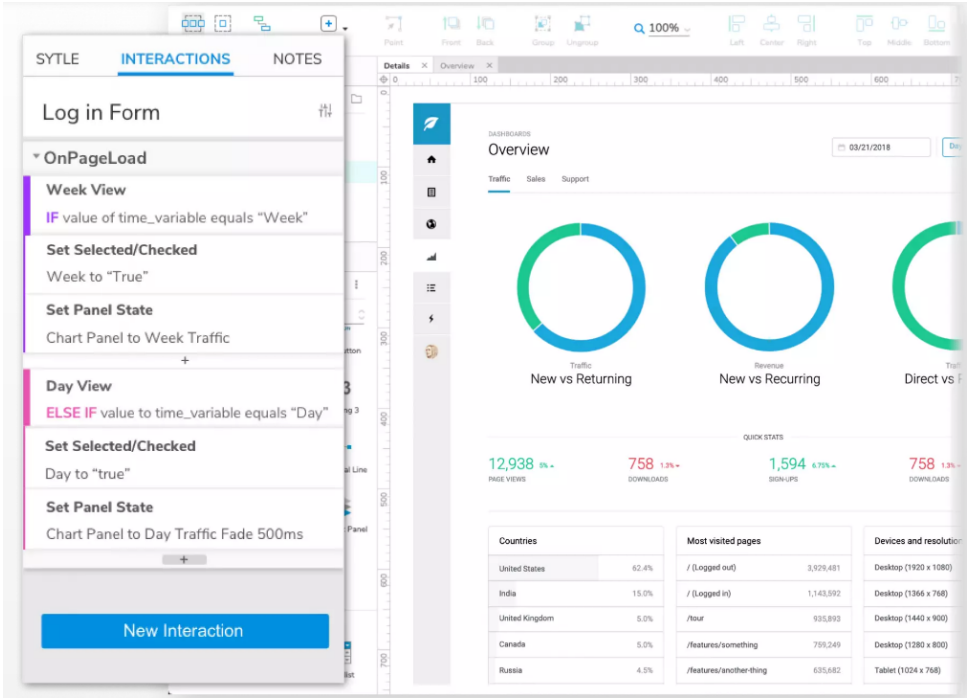

图 **9:** Axure

2.3.6 编程工具

由于我们团队选择了前后端分离的开发方式，前后端的解耦使得开发不必统一编程工具。且快速开发的方法让我们追求高效、便捷、灵活的工具。

前端优秀的编辑工具 Visual Studio Code（图 10），支持 Vim、Emacs 等编辑器模式，集成终端，社区极其活跃，可扩展性强，并且非常轻便灵活适合高效的前端快速开发。

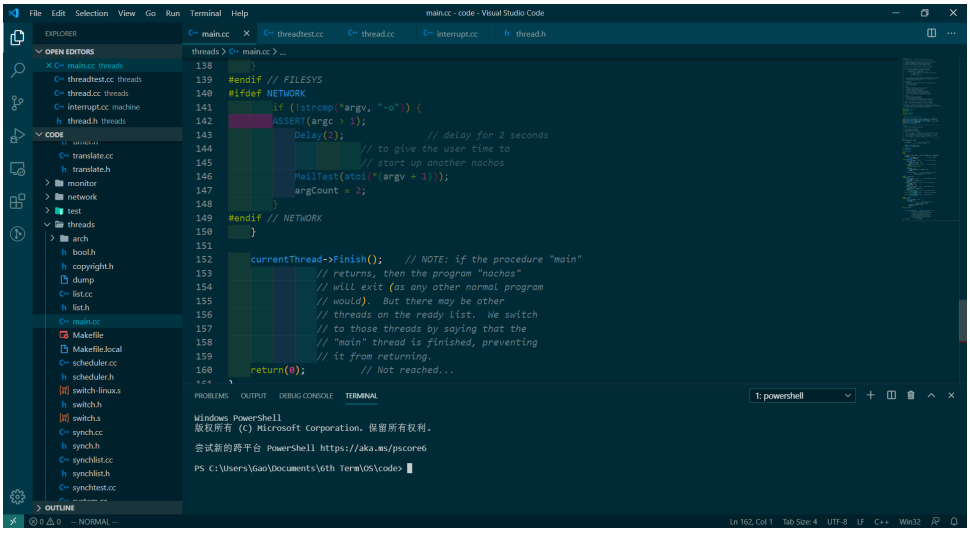

图 **10:** Visual Studio Code

后端的需求相比之下较为复杂，由于队员需要密切合作并且开发流程涉及复杂的调试过程，为此我们选用了 Visual Studio（图 11），同样也是目前最为强大和完善的 IDE，可以帮助开发者完整的进行构建、开发、发布软件业务。

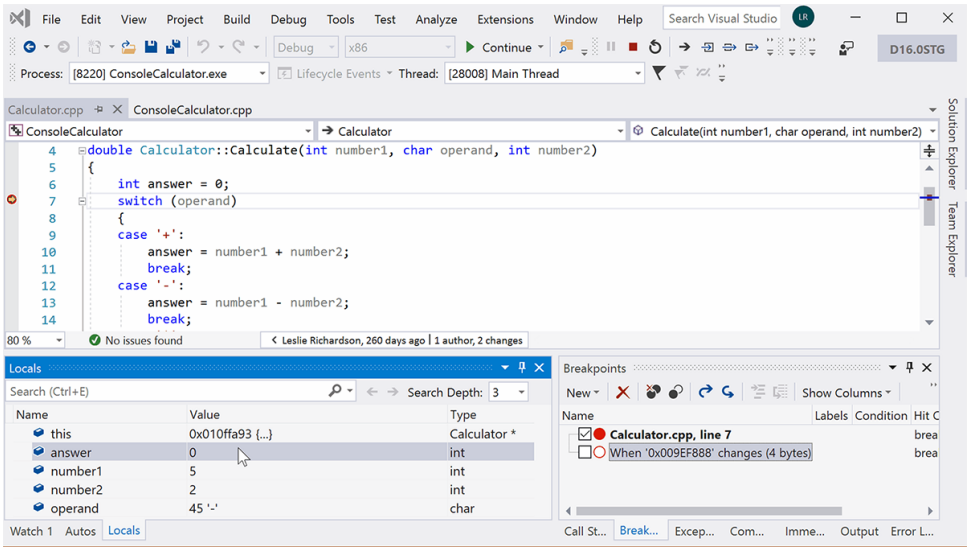

图 **11:** Visual Studio

2.3.7 质量保证工具

质量保证的软件组织监控工程过程和方法，通过开发软件产品，以确保质量的一致性按组织的标准。QA 工具包括配置和变更控制工具和软件测试工具。例如，SoapTest, AppsWatch, JMeter。

其中，JMeter（图 12）用于对服务器、网络或对象模拟负载，测试系统强度、分析整体性能。JMeter 能够对应用程序做功能/回归测试，此外支持断言以及正则表达式断言。JMeter 社区的完善是我倾向该工具的主要原因；此外测试当中引入的断言以及正则表达式，极大的提高了测试的灵活性，简化了测试操作。

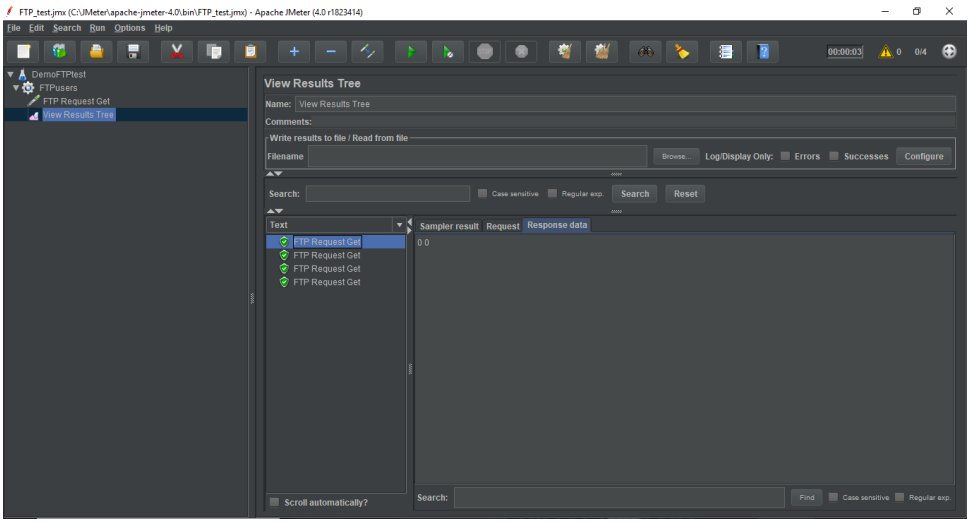

图 **12:** JMeter

3 敏捷开发调研

3.1 敏捷开发宣言

We are uncovering better ways of developing software by doing it and helping others do it.

Through this work we have come to value:

Individuals and interactions over processes and tools

Working software **over** comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, **while there is value in the items on the right**, we value the items on the left more.

3.2 敏捷开发背后的原则

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals.

Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development.

The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

Simplicity–the art of maximizing the amount of work not done–is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

3.3 敏捷开发的概念

1. 敏捷开发是一种价值观与原则，指导我们更加高效的开发。
2. 敏捷开发以用户需求为核心，采用迭代 (时间周期)、增量 (循序渐进，功能模块) 的方式开发软件，目的在于快速覆盖、响应市场需求。
3. 大项目划分为小项目，分别完成，独立运行，如微服务的开发过程，就是将系统独立进行开发。

3.4 方法论-Scrum

敏捷开发的方法有很多：ASD、AUP、DSDM、XP、FDD、Kanban、RAD、Scrum。目前国内最流行的应属 Scrum。

Scrum 的英文意思是橄榄球运动的一个专业术语，表示“争球”的动作；把一个开发流程的名字取名为 Scrum，寓意开发团队在开发一个项目时，大家像打橄榄球一样迅速、富有战斗激情、人人你争我抢地完成它。而 Scrum 就是这样的一个开发流程，运用该流程旨在使得团队高效运作。

Scrum 开发流程的三大角色：

1. 产品负责人 (Product Owner)：主要负责确定产品的功能和达到要求的标准，指定软件的发布日期和交付的内容，同时有权力接受或拒绝开发团队的工作成果。
2. 流程管理员 (Scrum Master)：主要负责整个 Scrum 流程在项目中的顺利实施和进行，以及清除挡在客户和开发工作之间的沟通障碍，使得客户可以直接驱动开发。
3. 开发团队 (Scrum Team)：主要负责软件产品在 Scrum 规定流程下进行开发工作，人数控制在 5 10 人左右，每个成员可能负责不同的技术方面，但要求每成

员必须要有很强的自我管理能力，同时具有一定的表达能力；成员可以采用任何工作方式，只要能达到 Sprint 的目标。

4 传统与敏捷对比与理解

从本质来讲，传统软件开发方法是一个软件开发架构，其开发过程是通过一系列阶段顺序展开的。通常，这一方法不能很好地表达和描述用户的需求，而且在项目整个开发周期的所有阶段都有需要不断完善的文档。

软件行业飞快发展，软件技术不断创新，客户期望迅速变化，考虑到需要克服传统开发方法的缺点，敏捷开发在近十年来兴起，以其灵活性，易操作性得到软件行业的广泛关注。敏捷方法通过使用迭代、早期测试和客户协作来处理不稳定的需求，在项目的整个开发周期中不断改进，从而使得敏捷开发方法能够尽快提供业务价值。也因此，在过去几年中，敏捷软件开发已经成为一种极具前景的复杂性方法，并提出了各种敏捷方法，其中包括被广泛应用的极限编程（XP）。

4.1 生命周期的对比

4.1.1 传统模式的生命周期

传统模式阶段划分分明，项目需求的细节要求在开发之前给定，并且要求用户需求明确，只有在正确的需求下才能得到正确的下一步结果。与此相对应的，每一阶段没有得到相应的文档，是无法进行下一阶段工作的。开发过程中客户不会参与。这也往往会导致最终开发软件与顾客理想软件有差距。瀑布模式是传统方法最典型的代表。

在实际的软件开发过程中，软件的需求往往是变化的，而瀑布开发模型很难适应这种变化。针对瀑布模型的这一不足，又出现了螺旋模型和统一过程开发模型，但仍然无法很好地适应需求的快速变化。

4.1.2 敏捷方法的生命周期

不同于传统模式，敏捷开发以用户的需求进化为核心，采用迭代、循序渐进的方法进行软件开发。所有的迭代（不论长度）都有相同的模式，这也是敏捷开发规律的一部分。在敏捷开发中，软件项目在构建初期被切分成多个子项目（得到大概的需求和架构模型），各个子项目的成果都经过测试，具备可视、可集成和可运行使用的特征。由此，敏捷开发的生命周期由多个迭代过程完成，在迭代的每次过程中都要重复分析、设计、编码等。

敏捷方法允许变化可以随时随地发生。极限编程（XP）是较为典型、最为完善的敏捷开发方法。XP 作为一种近螺旋式的敏捷开发方法，将复杂的开发过程分解为一个个相对比较简单的小周期，通过与客户积极的沟通、反馈以及其它一系列的技术方法，这一过程使得开发人员和客户都可以非常清楚开发的进度、变化、待解决的问题和潜在的阻力等，并根据实际情况及时调整开发过程。XP 的生命周期如图 3 所示，它首先创造一个候选架构，然后通过一个关于整个系统运作方式的简单描述来指导全部开发过程，而不需要提前进行详细架构设计。

4.1.3 比较讨论

传统方法在开发初期和客户沟通，获取尽可能明确详尽的需求，其开发软件的过程往往是客户与开发团队的利益博弈的过程，所以在开发过程中顾客的参与度不高，主要强调计划、过程和文档等。而敏捷方法对于需求不明确的复杂项目，要求客户和开发团队一起开发能够在较短时间和较低预算内成功完成，主要强调团队、客户合作和拥抱变化等。

	敏捷开发	传统开发
用户需求	迭代获取	编码之前获得详细的需求
修改成本	低	高
开发方向	易改变	确定的
测试	每次迭代	编码阶段完成后
客户参与	高	低
对开发人员素质要求	个人技能和基本业务技能	无特殊要求
适合的项目规模	小型、中型	大型

4.2 范围管理对比

范围用以限制和控制项目中包含的工作。良好定义和良好管理的范围对于项目成本效益和软件开发时间表非常重要。项目范围主要涉及到两方面内容：

1. 产品范围界定：产品范围的特征和功能包含在产品或服务中。

2. 工作范围界定：项目工作的完成为的是能交付一个有特殊的特征和功能的产品。

项目范围管理包括的程序，要求能确保该项目所覆盖的整体工作要求和单项工作要求，从而促使项目工作成功地完成。针对软件开发过程中不明确的需求、不可用的资源，不断变化的环境和不灵活性等众多可能问题，项目中需要进行范围管理，如果不仔细管理，可能导致项目失败。范围管理主要包括以下五个过程：

1. 启动阶段：督促项目管理组织开始着手项目下一阶段的工作。
2. 范围规划报告：写出一份书面报告，作为未来项目决策基础。
3. 范围界定：把主要的项目工作细目分解成更小、更易管理操作的单元。
4. 范围核实：正式认可这个项目范围。
5. 范围变化控制：对项目范围的变化进行控制。

4.2.1 传统方法的范围管理

在传统软件开发方法中，范围被定义为软件项目的完整需求规范。它包含项目开始时的详细需求，在开发过程的后期阶段需要分析使用这些需求。然而，耗费开发人员很多时间和精力完成的相关文档并不支持在开发过程的后期阶段可能发生的变化。这些不可控的变化往往会导致范围蠕变（在产品或项目开发期间，需求驱动发生变化，带来一些开始没有计划的产品特点，对产品质量或软件开发时间表产生影响），而处理范围蠕变需要使用不同的工具和技术，这可能使得项目逾期并且超出预算。

传统方法创建工作分解结构（WBS），用以把项目可交付成果和项目工作分解成较小的、更易于管理的组成部分的过程。当范围发生变化时，传统方法需要审查整个工作分解结构，很难基于特定变化做出良好快速的适应，这将不可避免地影响项目的成本、资源、质量和时间表。因此，为避免项目失败，传统方法需要非常仔细地定义和管理其范围。

4.2.2 敏捷方法的范围管理

敏捷方法拥抱软件开发过程的后期阶段的变化，即接受项目范围的波动。因此，敏捷方法的项目范围常常需要满足高级别的要求。敏捷方法的范围采用迭代并接受渐进的变化，由负责接受或拒绝在每个迭代期间完成的功能的客户来验证和控制。

管理范围蠕变是敏捷方法范围管理的一个非常重要的方面。在软件开发过程中会不断地产生一些变化，其中可能存在着影响整个项目的变化，敏捷方法中的范围管理技术会管理这些不可控的改变，这在一定程度上保证了软件项目在开发过程中的稳定性。与传统方法不同，在敏捷方法中没有创建 WBS。

4.2.3 对比讨论

范围定义了软件开发过程的边界。范围管理是关乎敏捷方法和传统方法成功完成整个过程的一个重要因素。

敏捷方法中的范围管理允许变化并且可以减少不必要的变化，在范围上遵循迭代计划，需要满足高级别的要求。具有上述特征使得敏捷方法的范围管理保证了软件的时间表，并且软件产品可以在预算内以良好的质量交付。

传统方法中的范围管理中不可控的变化导致范围蠕变，往往使项目超出预算并且破坏软件开发时间表。同时，在传统方法中需要创建 WBS，且管理的范围需要以全面的文档的形式进行详细定义。

4.3 方法、理念及其适应性

敏捷方法体现着以“人”为核心的理念。

章节 3.1当中值得强调的是，作者并没有否认 right value，而是侧重于 left value。也就是说，尽管比其文档而言，敏捷开发看重实际代码，但是：文档一样重要。敏捷开发的转移对传统工程中繁重部分的注意力，而致力于确实推进软件项目的部分。

敏捷宣言原则与传统软件开发方法很明显的不同在于，就是让我深刻感受到了工程项目中的“人”的元素。软件开发是“人与人的合作”，软件开发的目的也是满足“人的需求”（章节 3.2中倡导“欣然接受需求变更”）。一切都是为了活生生的人服务，所以我们在敏捷开发当中促进开发者的有效沟通交流、尽可能地满足人的需求。

我们可以认为敏捷方法是综合多种传统方法优点整理出来的一种开发方法，是一个新的思路，但这并不意味着不一定是所有软件开发的最优选择。当然，敏捷方法也存在一些问题，如敏捷方法中通常使用代码替代文档，在很多实际情况下大大降低了系统的可读性。这就需要我们能够随机应变，采用更务实的思想和方法。

软件开发过程没有最完美的方法，只有最合适的方法。即便是 Scrum、XP 等成熟方法往往在实际应用当中也没有固定的规则。对于我们项目而言，小团队为高效完成项目、降低交流成本，侧重实际编码工作，为此我认为敏捷开发的生命周期及范围管理无疑更为适合。我们欣然接受需求变更，同时采用敏捷开发的方法论灵活处理变更、调整开发策略。