Ztest: A C++ Unit Testing Framework

High-level Language Programming Project

郑辰阳　叶穗华　吴泓庆　齐彦松　王瑞箐

未来技术学院
数据科学与大数据技术

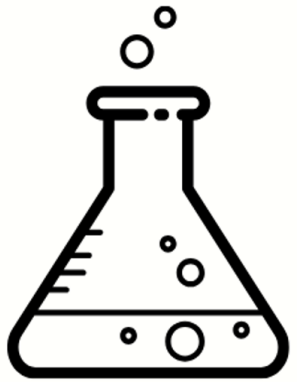
2025 年 6 月 6 日

背景介绍

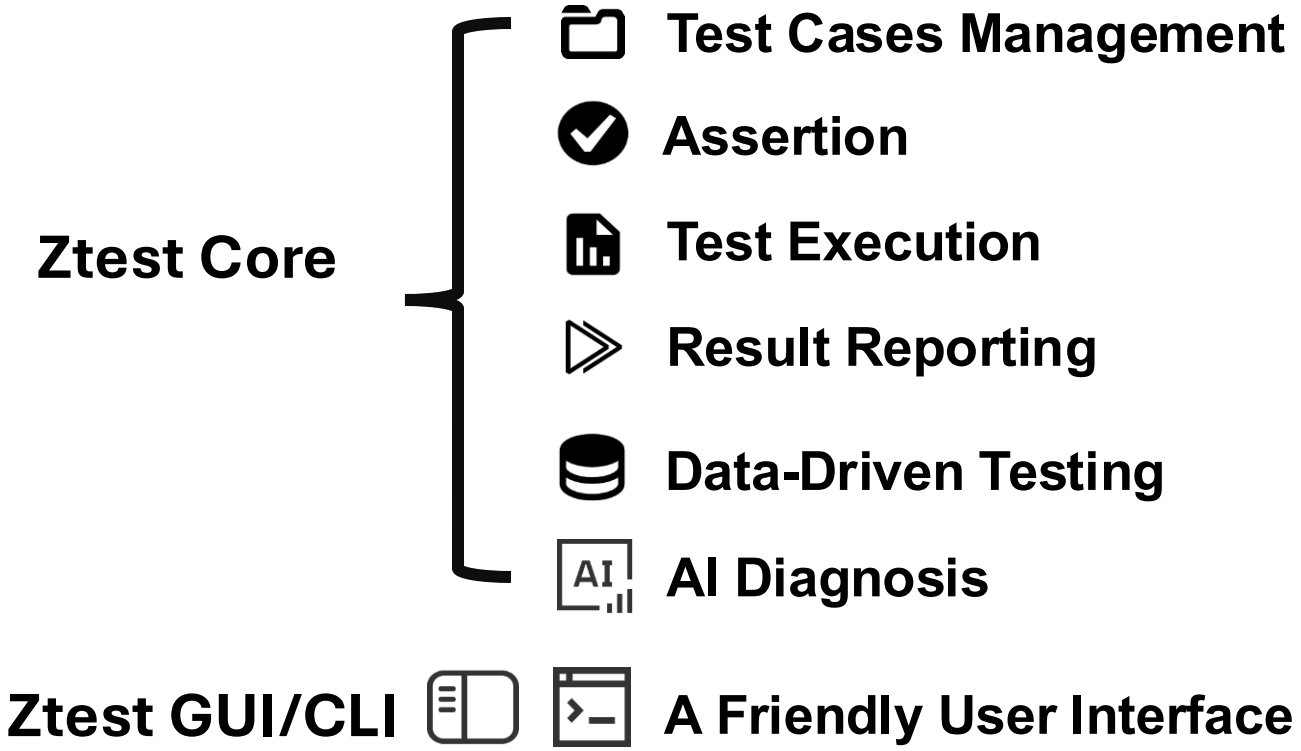
单元测试（英语：Unit Testing）又称为模块测试，是针对程序模块（软件设计的最小单位）来进行正确性检验的测试工作。程序单元是应用的最小可测试部件。在过程化编程中，一个单元就是单个程序、函数、过程等；对于面向对象编程，最小单元就是方法，包括基类（超类）、抽象类、或者派生类（子类）中的方法。

表 1: 主流测试框架对比

框架	GUI 支持	并发测试	报告系统	扩展性	数据驱动
Google Test(C++)	无	有限	基础	中等	不支持
JUnit (Java)	Eclipse 插件	支持	HTML/XML	高	支持
PyTest (Python)	第三方工具	优秀	丰富	优秀	支持
Catch2 (C++)	无	一般	简洁	中等	不支持
Ztest* (C++)	精美	优秀	丰富	高	支持



Ztest Function



TEST Management

Two types of basic testing

Safe Test
Concurrent and thread-safe test

Unsafe Test
serial execution or thread-unsafe

Two types of special testing

Benchmark Test
Multiple iterations to evaluate performance

Parameterized Test
Data-driven testing with parameters

Test Suite

Hooks

- BeforeAll
- AfterEach
- AfterAll

Tests

N * Safe test

M* Unsafe test

两种定义方式

```
  auto test1 = TestFactory::createTest("Addition", ZType::z_safe, "", add, 2,
2)                  .setExpectedOutput(4)
                    .build();
```

链式定义

```
ZTEST_F(BasicMath, FailedAdditionTest)
{ EXPECT_EQ(6, add(2, 3));
  return ZState::z_success;
}
```

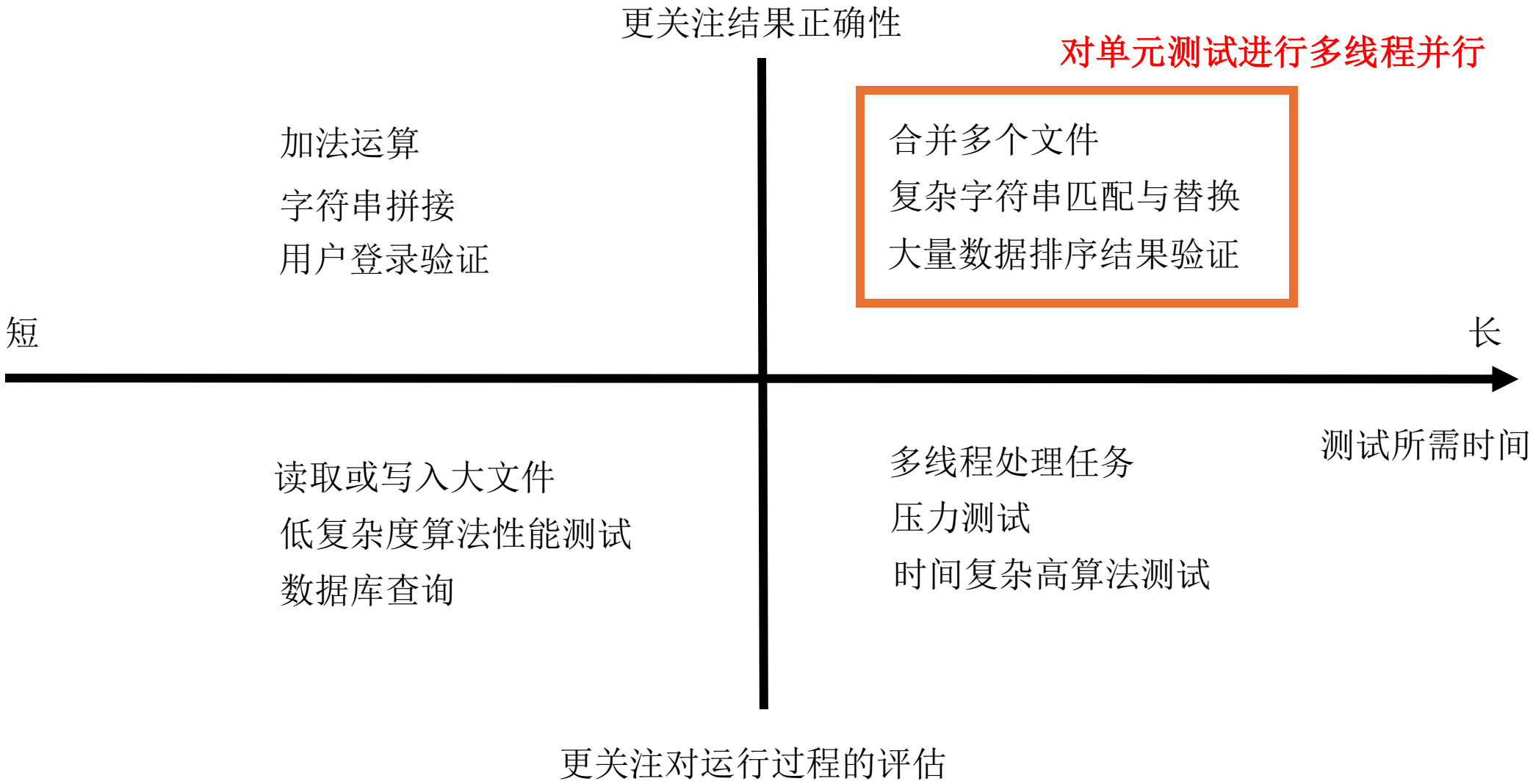
宏定义

Assertion

- **EXPECT_EQ**: Verifies whether two values are equal.
- **ASSERT_TRUE**: Verifies whether a condition is true.
- **EXPECT_NEAR**: Verifies whether two floating-point values are close enough.

```
// If the assertion fails, an exception is thrown
EXPECT_EQ(5, add(2, 3));
ASSERT_TRUE(6 == add(2, 3));
EXPECT_NEAR(5.0, add(2.0, 3.0), 0.1);
```

任务分类

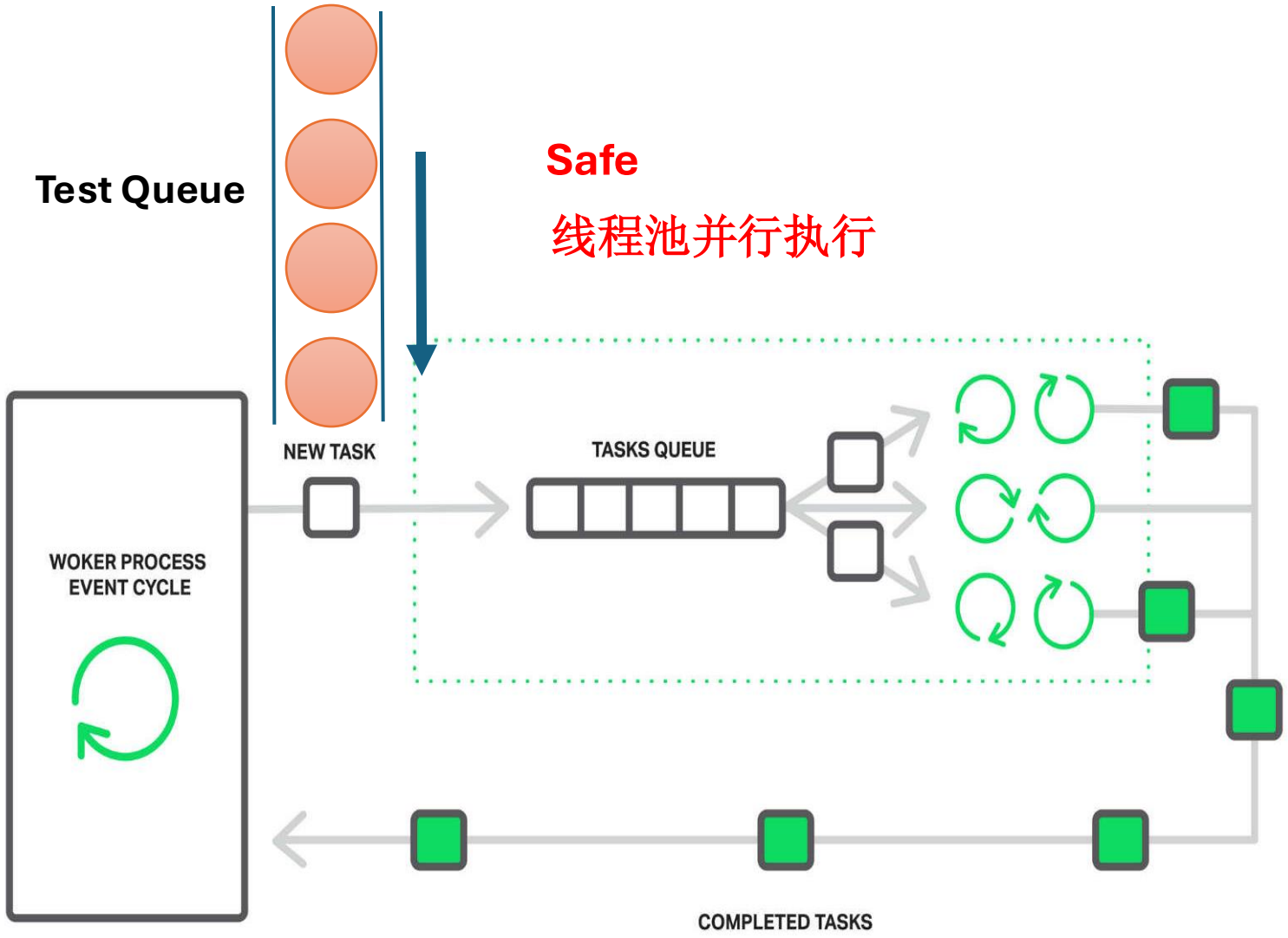

更关注结果正确性
对单元测试进行多线程并行
加法运算
字符串拼接
用户登录验证
合并多个文件
复杂字符串匹配与替换
大量数据排序结果验证
短
长
测试所需时间
读取或写入大文件
低复杂度算法性能测试
数据库查询
多线程处理任务
压力测试
时间复杂高算法测试
更关注对运行过程的评估

TEST Actuator

Test Queue

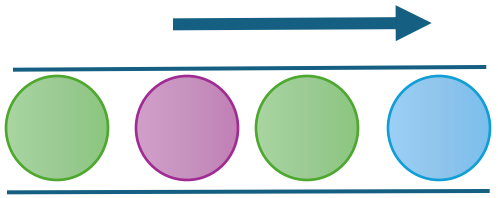
Safe

线程池并行执行



UnSafe,参数化和benchmark

顺序执行



Test Queue

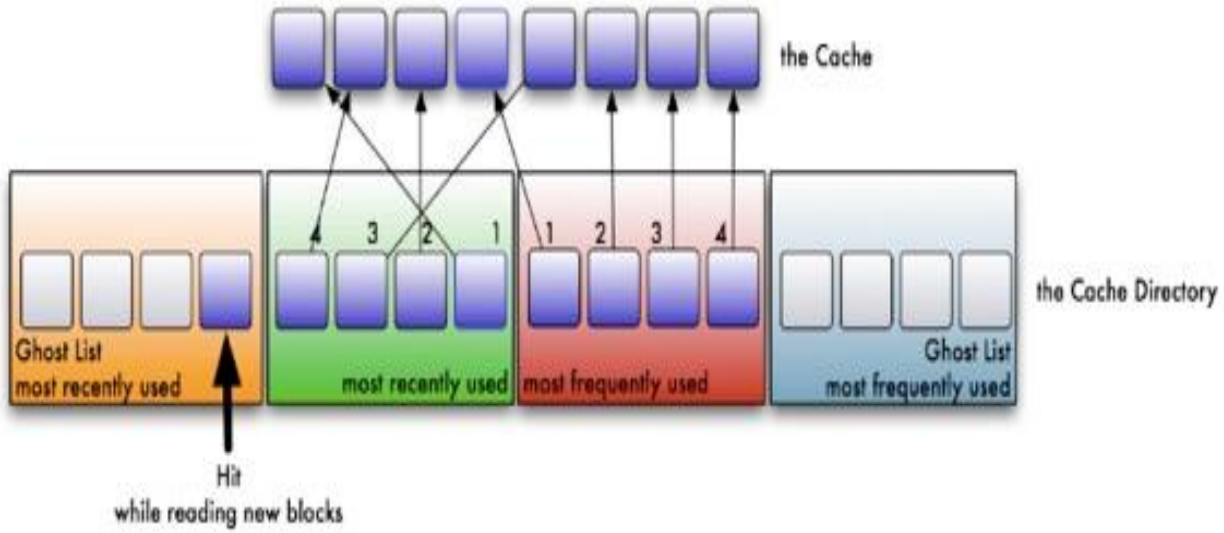
Data-Driven Testing

Data.csv

Storage

cache ↓

Memory



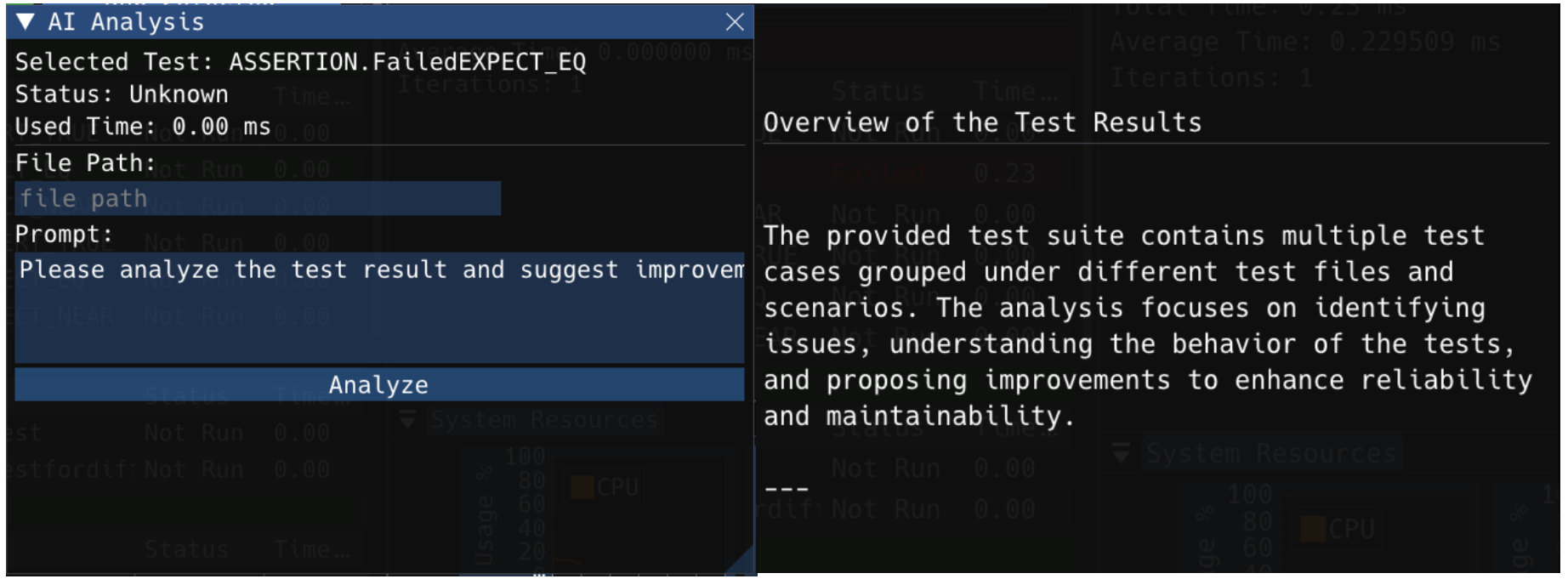

```
ZTestDataManager<vector<int>, int> sum_test_data = {
    {{1, 2}, 3}, {{-1, 1}, 0}, {{10, 20}, 30}};
ZTEST_P(ArithmeticSuite, SumTest, sum_test_data) {
  auto &&[inputs, expected] = _data.current();
  int actual = inputs[0] + inputs[1];
  EXPECT_EQ_FOREACH(expected, actual);
  return ZState::z_success;
}
```

```
ZTEST_P_CSV(MathTests, AdditionTests, "data.csv") {
  auto inputs = getInput();
  auto expected = getOutput();
  double actual = std::get<double>(inputs[0]) + std::get<double>(inputs[1]);
  EXPECT_EQ(actual, std::get<double>(expected));
  return ZState::z_success;
}
```

AI Diagnosis

AI分析报告

1. **根本原因**: 测试失败源于逻辑错误或精度问题，如加法计算偏差、断言条件不满足及浮点数比较精度不足。
2. **修复建议**: 修正加法函数逻辑，使用更宽松的浮点数比较容差，确保断言条件正确。
3. **高风险测试用例**: ASSERTION.FailedEXPECT_EQ、ASSERTION.FailedEXPECT_NEAR、ASSERTION.FailedASSERT_TRUE。
4. **测试覆盖率评估**: 覆盖率较高，但需关注浮点运算和逻辑分支覆盖。
5. **稳定性改进建议**: 增强浮点数比较容差，优化异常处理，增加边界值测试。



Ztest Core Architecture

Registry for Testing

ZTestRegistry

Run

Test

Register

ZTest Suite

N * Safe test

M* Unsafe test

Safe Test

Unsafe Test

Benchmark

Parameterized

Assertion

Construct

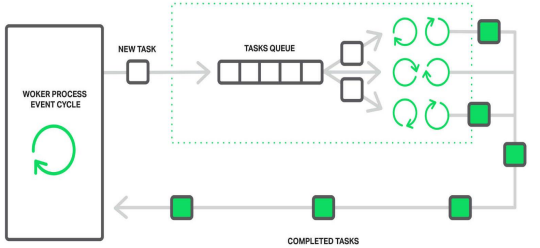
EXPECT_EQ

ASSERT_TRUE

Test Actuator

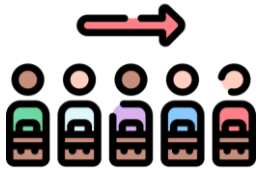
Strategy 1: Parallel Testing

Safe test



Strategy 2: Serial Testing

others



Result Manager

Report

Diagnosis

ZTestResultManager

AI

Export

Report

HTML

JSON

XML

Integrate

CI/CD

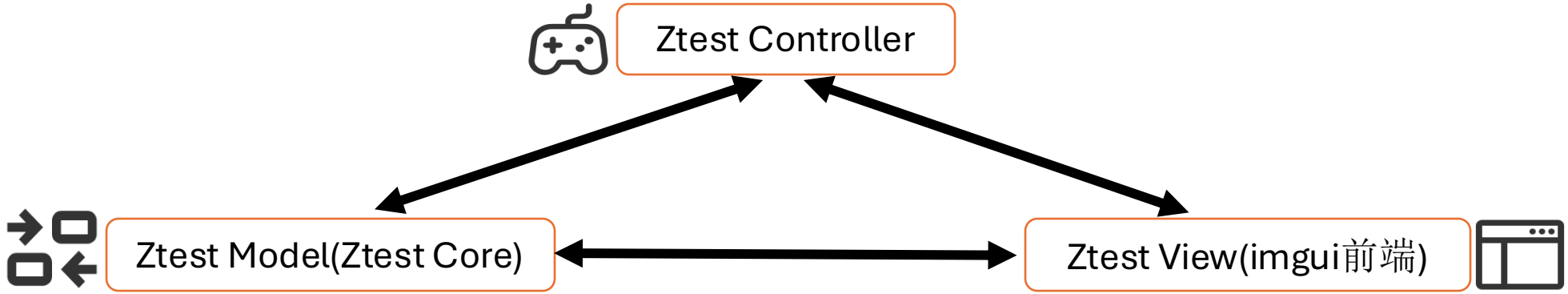
Data-Driven

ZDataRegistry
(Cache with LRU)

Cache

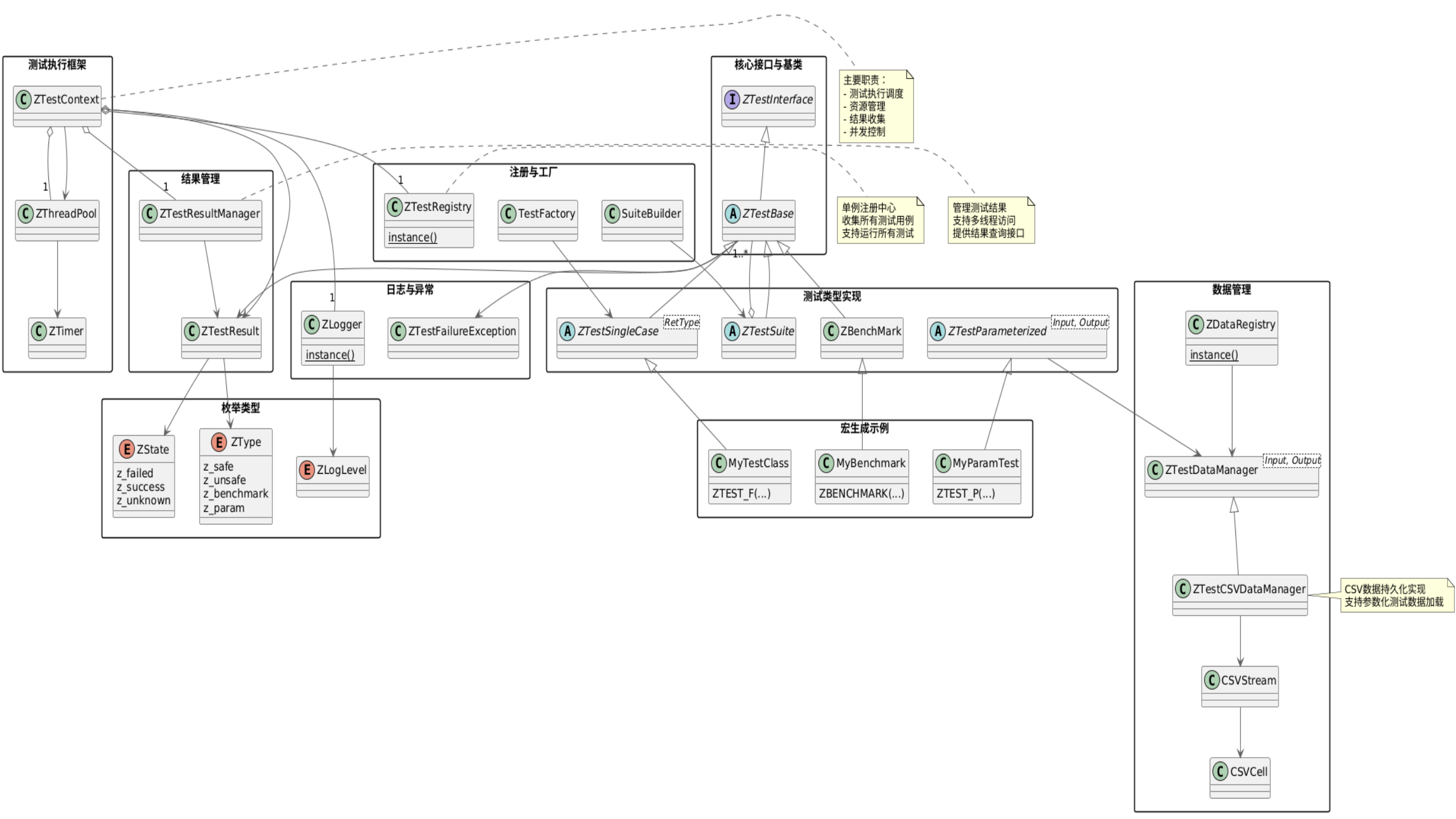
External Data(*.csv)

GUI Architecture



CLI

```
Usage: executor_name [OPTIONS]
Options:
--help Show help
--run-all Run all tests
--list-tests List all tests
--no-gui Run in headless mode
```


测试执行框架
ZTestContext
ZThreadPool
ZTimer
结果管理
ZTestResultManager
ZTestResult
注册与工厂
ZTestRegistry
instance()
TestFactory
SuiteBuilder
核心接口与基类
ZTestInterface
ZTestBase
主要职责：
- 测试执行调度
- 资源管理
- 结果收集
- 并发控制
单例注册中心
收集所有测试用例
支持运行所有测试
管理测试结果
支持多线程访问
提供结果查询接口
日志与异常
ZLogger
instance()
ZTestFailureException
测试类型实现
ZTestSingleCase
RetType
ZTestSuite
ZBenchMark
ZTestParameterized
Input, Output
枚举类型
ZState
z_failed
z_success
z_unknown
ZType
z_safe
z_unsafe
z_benchmark
z_param
ZLogLevel
宏生成示例
MyTestClass
ZTEST_F(...)
MyBenchmark
ZBENCHMARK(...)
MyParamTest
ZTEST_P(...)
数据管理
ZDataRegistry
instance()
ZTestDataManager
Input, Output
ZTestCSVDataManager
CSV数据持久化实现
支持参数化测试数据加载
CSVStream
CSVCell

Responsibilities

- Zheng Chenyang: Architectural design, main code writing for ztest core and GUI, report writing, and presentation
- Ye Suohua: GUI improvement, partial test logic improvement
- Wu Hongqing: GUI improvement
- Qi Yansong: GUI improvement
- Wang Ruizhen: Attempt to migrate to Windows

THANKS