漫谈敏捷测试工具实现

一淘
www.etao.com

测试- 公直

测试 & 测试人员

时间	目的
1946	第一台计算机诞生 “ENIAC（埃尼阿克）”
1947-1956 **Debugging oriented period**	测试 = 调试
1957-1978 **Demonstration oriented period**	表明程序正确
1979-1982 **Destruction oriented period**	发现软件错误
1983-1987 **Evaluation oriented period**	产品评估 & 质量度量
1988–now **Prevention oriented period**	度量 & 缺陷预防

测试的未来

“软件测试的真正价值并不体现在代码中找出多少缺陷，而是发现设计和编程人员解决问题方法上的局限、思路中的狭隘和技能方面的不足。”

-- 托尼.霍尔 1996

对于质量来说，预防问题比发现问题本身更重要。质量更多是开发人员的问题，而不是测试人员的。通过把测试工作融入到开发过程中，我们能降低那些富产Bug的人的出错机会，不仅可以避免了大量最终用户的使用问题，而且还可以极大地降低测试人员报无效Bug的数量。

-- 谷歌如何测试

未来的测试工程师


高灵敏度传感器
可在墙壁后和黑夜中通过雷达、照相机等进行探测并根据图像进行指挥
小型显示器
可通过语音激活显示菜单显示作战信息
供给包
空气过滤系统和饮水管道可在遭遇生化袭击时保护士兵
生化感测器
靠近皮肤，可方便远方的医护人员及时掌握士兵的心跳、脉搏等健康状况
盔甲
采用高科技纳米材料制成，能像变色龙一样跟随周围的环境随时变换颜色
人造肌肉
外部的人造骨头和肌肉部分可提升战士的耐力
智能织品
能够感应和杜绝生化武器带来的伤害

关于软件测试工具

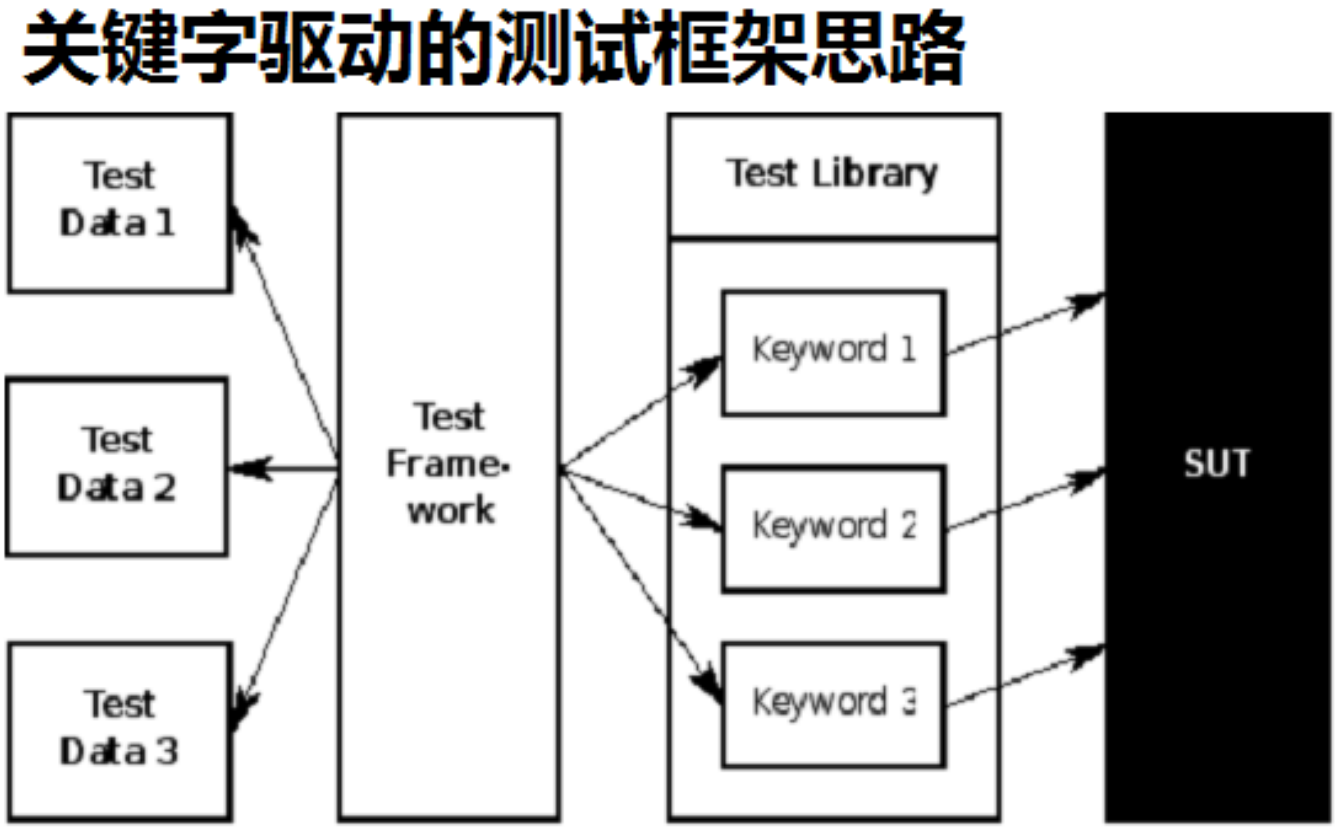
- 测试工具 vs 测试框架
- 已有工具 vs 新工具

Software QA Testing and Test Tool Resources

Last updated: Sunday, 10-Jun-2012 06:00:01 PDT

Information	Application Test Tools	Web Test Tools	Other
• General • General - Tools • Mailing Lists • Publications • Web Sites • QA Job Sites • White Papers • QA Tester Certifications	• Source Test Tools • Functional Test Tools • Performance Test Tools • Java Test Tools • Embedded Test Tools • Database Test Tools	• Link and HTML Test Tools • Security Test Tools • Functional Test Tools • Performance Test Tools • Performance Test Services	• Test Management Tools • Bug Tracking Tools • API Test Tools • Communications Test Tools • Requirements Management Tools • Other Tools • Services
TESTING GLOSSARY · TESTING TYPES · TRAINING COURSES			

常见几种测试框架的设计思想



框架设计 Example


```
from selenium import selenium
from urlparse import urlsplit

class Browser(object):

    def __init__(self, url, browser='*firefox'):
        base, path = self._split_url(url)
        self.selenium = selenium('localhost', 4444, browser, base)
        self.selenium.start()
        self.selenium.window_maximize()
        self.selenium.set_speed(1000)
        self.selenium.open(path)

    def _split_url(self, url):
        tokens = urlsplit(url)
        return '://'.join(tokens[:2]), ''.join(tokens[2:])

    def input_username(self, username):
        self.selenium.type('username_field', username)

    def input_password(self, password):
        self.selenium.type('password_field', password)

    def click_login_button(self):
        self.selenium.click('login_button')
        self.selenium.wait_for_page_to_load(5000)

    def verify_title(self, expected):
        title = self.selenium.get_title()
        if title != expected:
            raise AssertionError("Expected title to be '%s' but it was '%s'"
```

← Test library

↓ Driver script

```
from seleniumlibrary import Browser

browser = Browser('http://localhost:7272/html')
browser.input_username('demo')
browser.input_password('mode')
browser.click_login_button()
try:
    browser.verify_title('Welcome Page')
except AssertionError, err:
    print 'Login test failed:', err
else:
    print 'Login test passed.'
finally:
    browser.close()
```

“敏捷测试”

- 敏捷测试 vs 测试敏捷；
- 关于敏捷的谎言
  - 敏捷注重交流
  - 敏捷是一个概念
  - 大忽悠的工具

测试-层次

单元测试	• 代码 Check In 级别的触发模式
功能测试	• 手动、半手动测试
回归测试	• 自动化功能回归测试，已有sprint功能回归
性能测试	• 基线对比测试居多
冒烟测试	• 产品发布上线，验收测试
流量测试	• 灰度发布中使用

一淘测试实践尝试 - 持续集成 + 自动化测试

- 一种软件开发实践，核心在于提高集成的速度 (瀑布模式 -> 每日构建 –> 持续集成)
- **流程自动化，把软件开发过程的各个流程串通**，快速反馈各个环节的问题；
  1. 代码 Check In
  2. 编译打包
  3. 自动化测试 (静态扫描+ 单元测试 + BVT)
  4. 回归测试 (部署 +自动化功能测试)
  5. 发布 (持续交付)

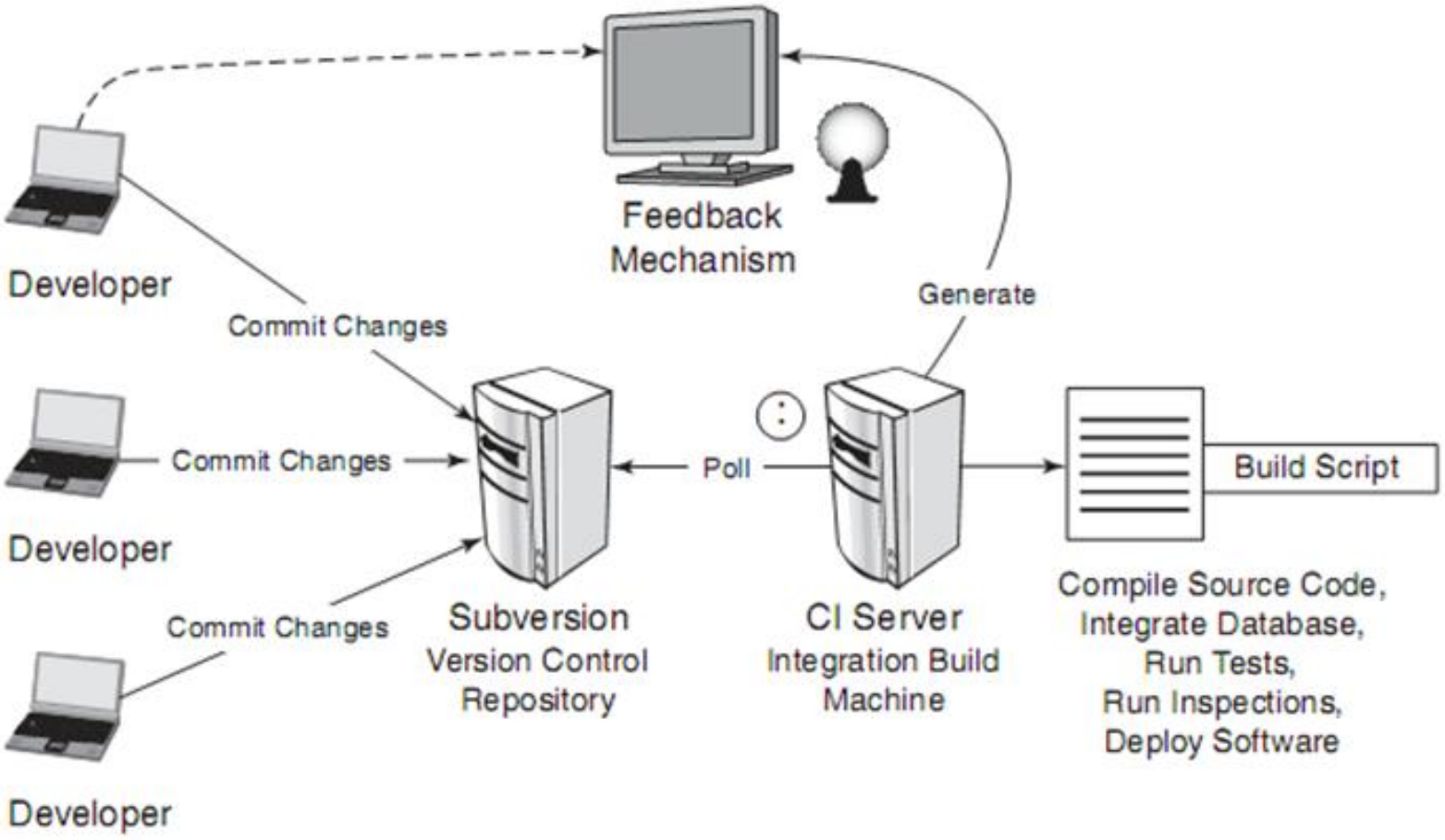
持续集成模式下对自动化测试工具的需求

- 自动化测试运行过程
- 自动化测试运行数据
- 与SVN、 Build系统集成
- 结果分析与通知
- 测试环境的管理

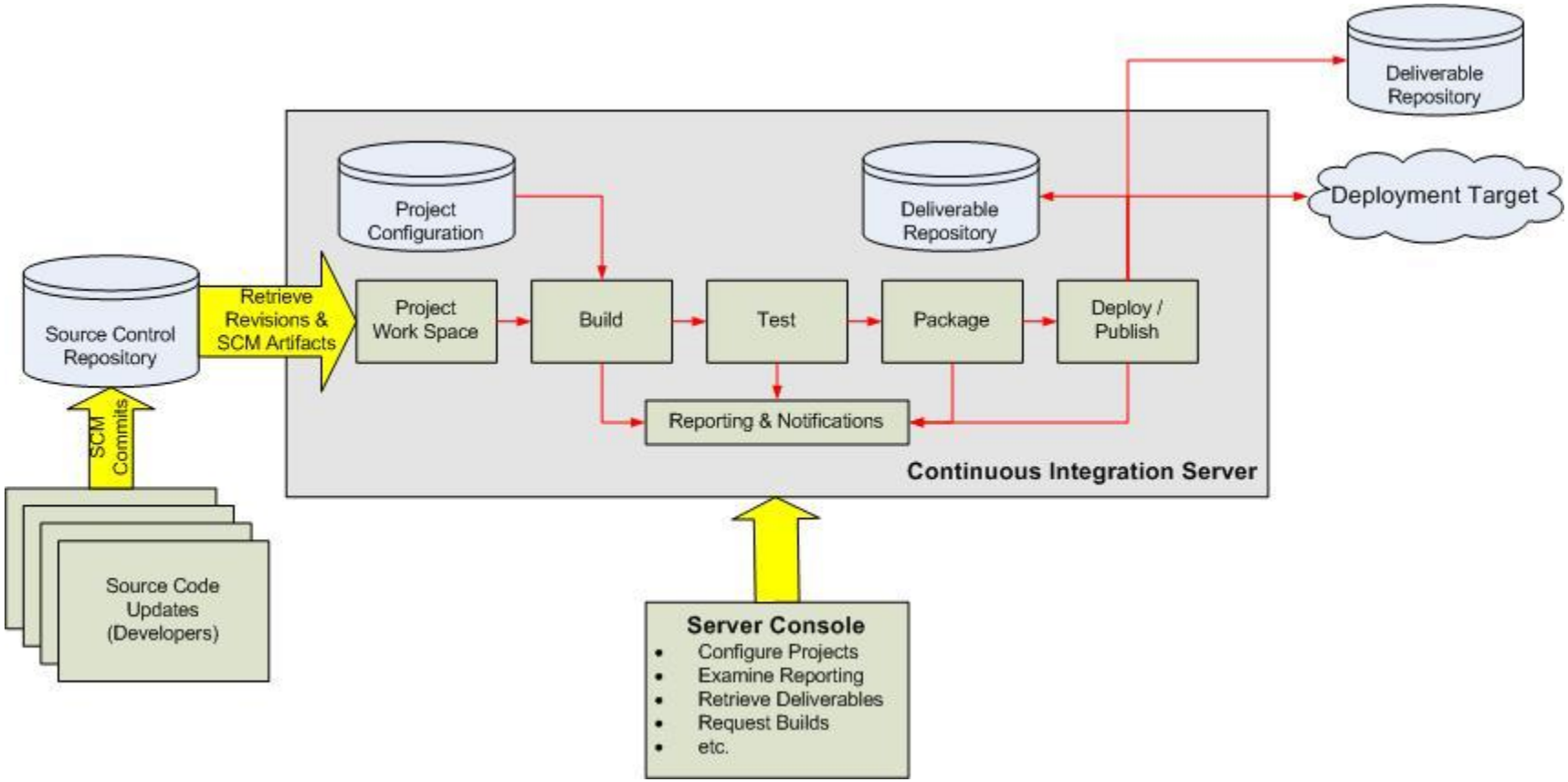
自动化测试调度工具 - TOAST

- **Toast = Toast Open Automation System Test**
- 基本目标
  - 测试运行公开、简单、高效；
  - 测试执行调度工具；
- 为什么不 HUDSON/ Jenkins Plugin
- 主要功能
  - 持续集成测试
  - 分布式测试
  - 报表
  - Open API

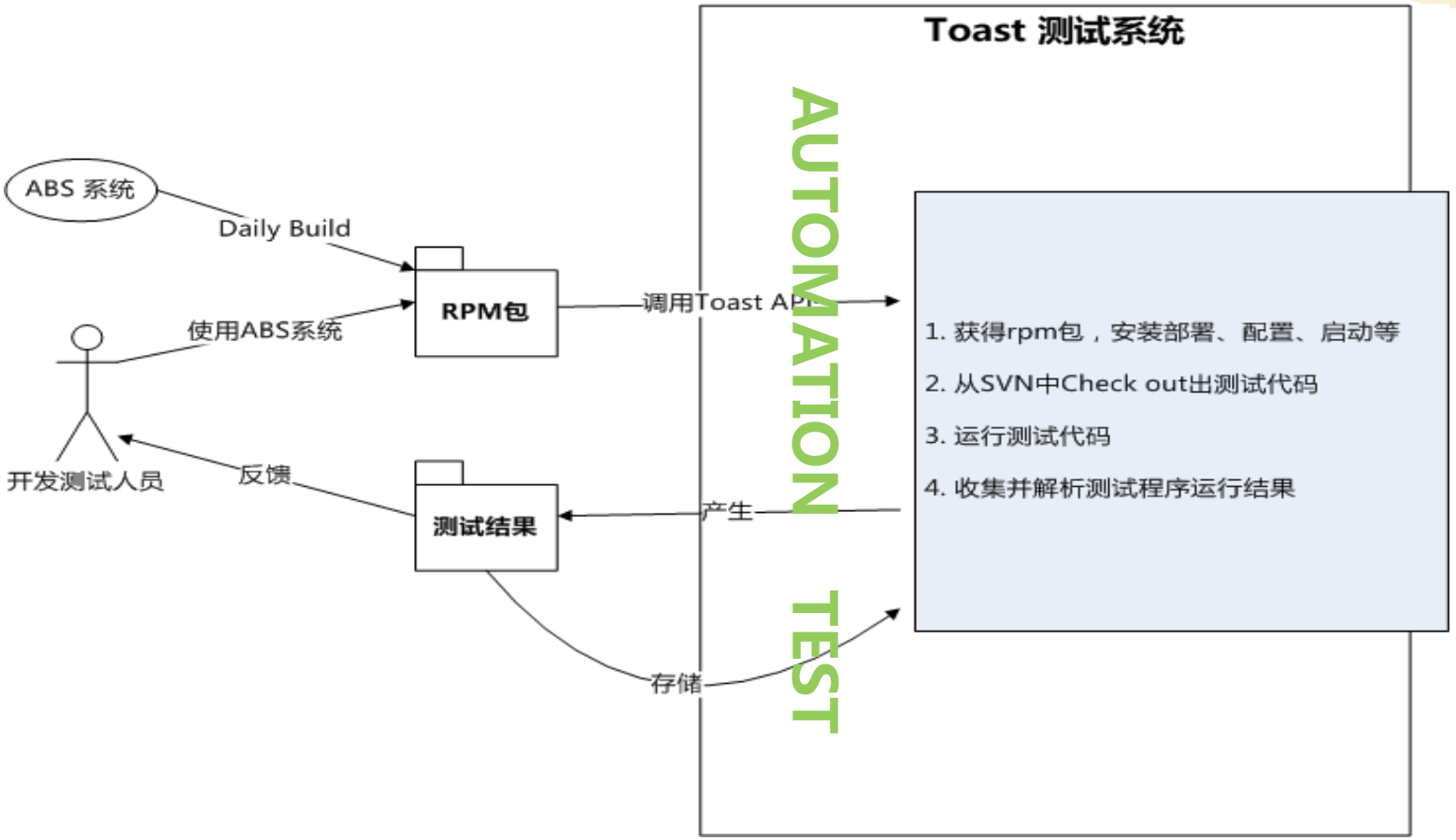
TOAST 之一 Check In 触发单元测试场景



TOAST CI 之一 ABS 触发回归功能测试场景(一)



TOAST CI 之一 ABS 触发回归功能测试场景(一)



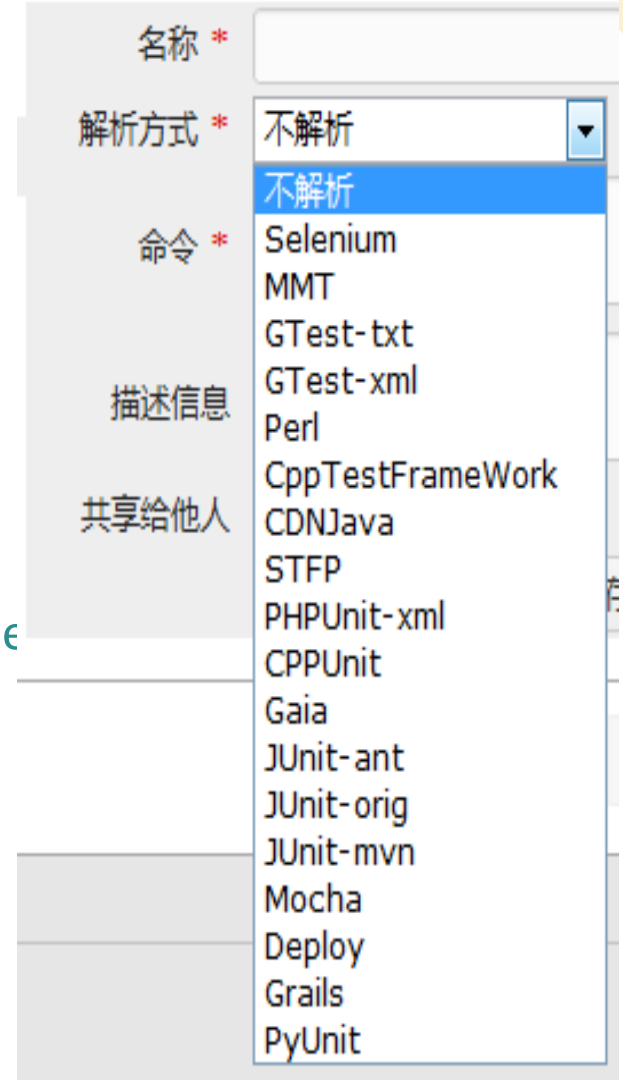
Daily Build → Daily Test

TOAST DEMO

演示时间

支持的自动化框架

- API 级别测试（Xunit 测试框架)
  - Gtest/Cppunit/CpptestFramework
  - JUnit/MRUnit/PHPUnit
- 用户界面级别（黑盒测试系列)
  - MMT/STFP (php script test framework)
  - Selenium/Ruby Watir
  - Custom Framework(Search Perl/CDN Java/Banne



使用指南 -- 自动化任务 – 调度方式

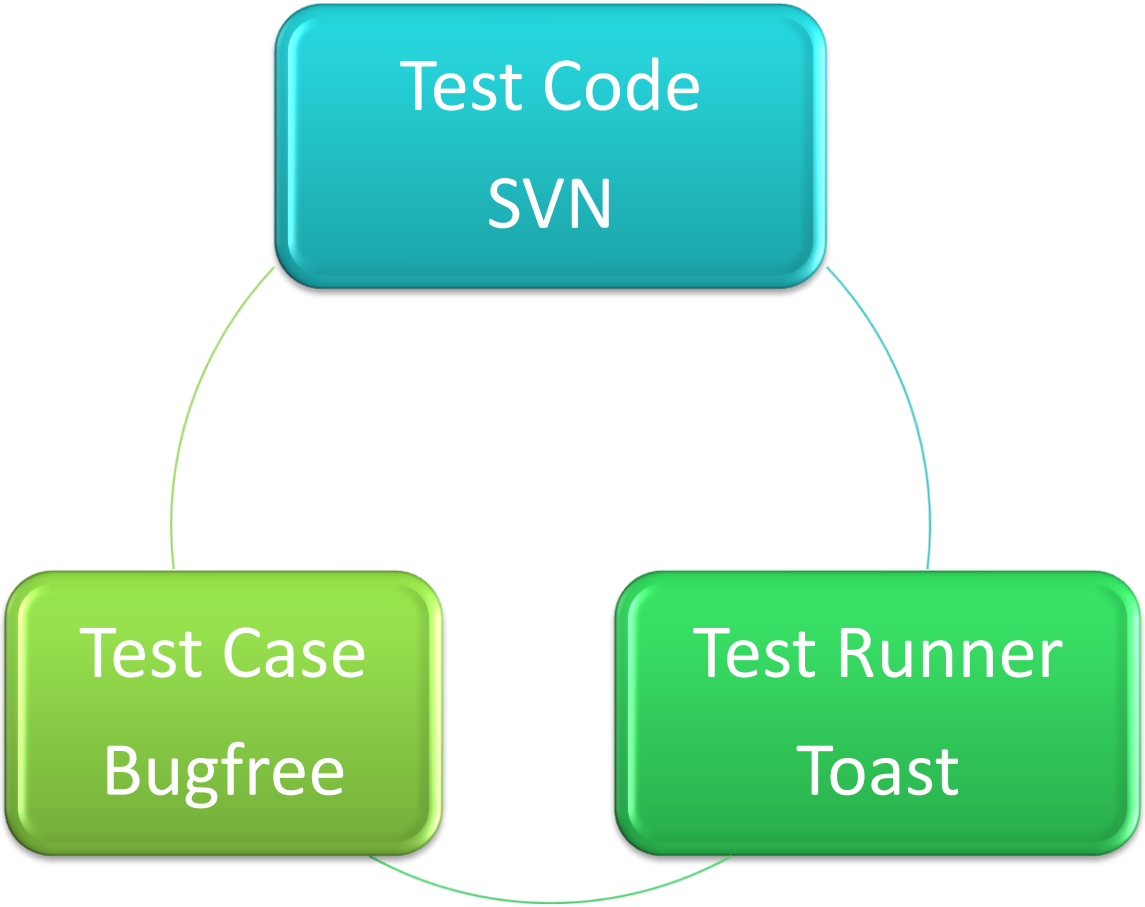
ABS触发 -- Build系统上设置

手动触发 -- 页面上点击“运行”

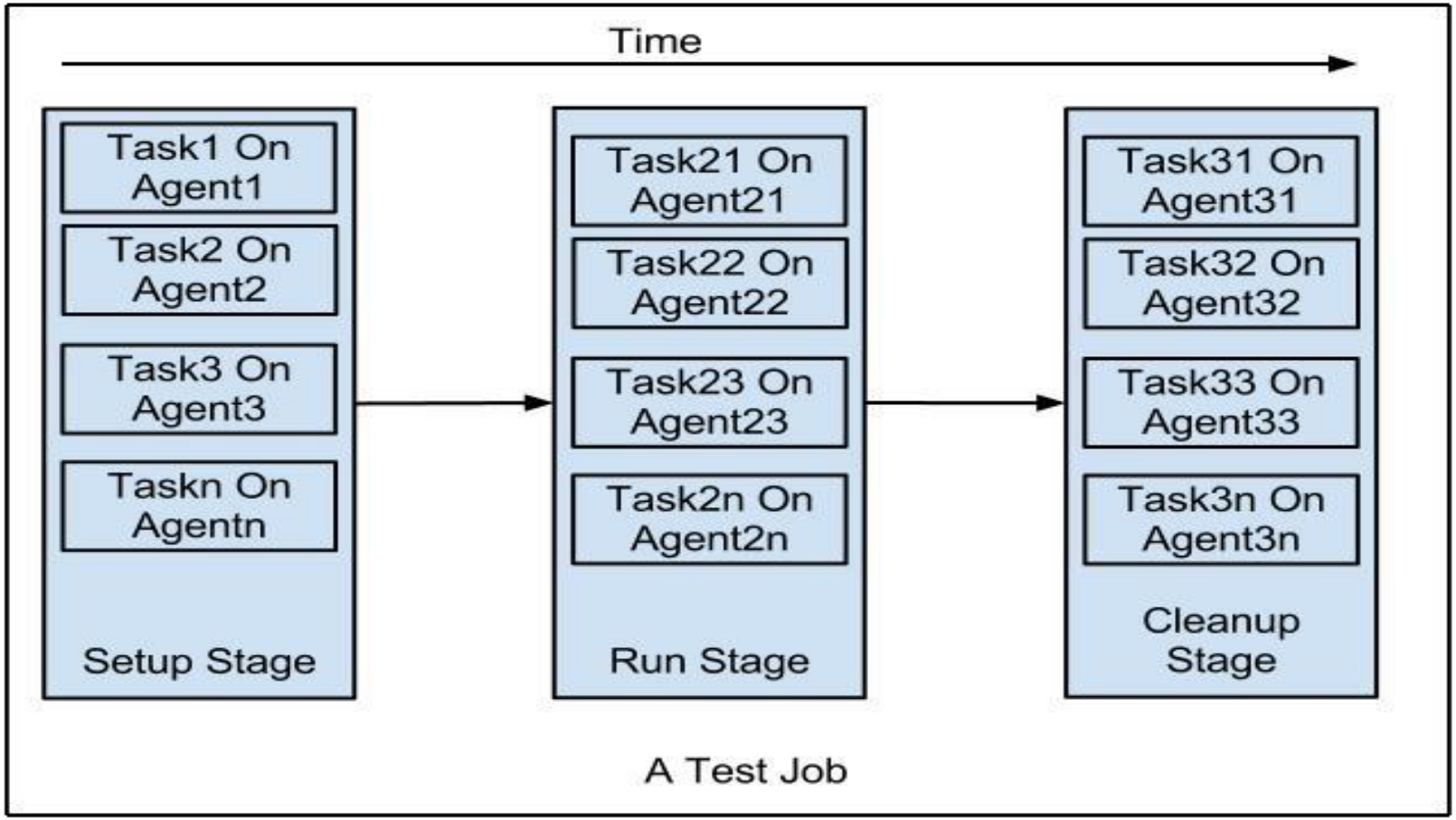
定时触发 -- 定时任务设置

API 触发 -- http curl

代码、用例、运行的关系



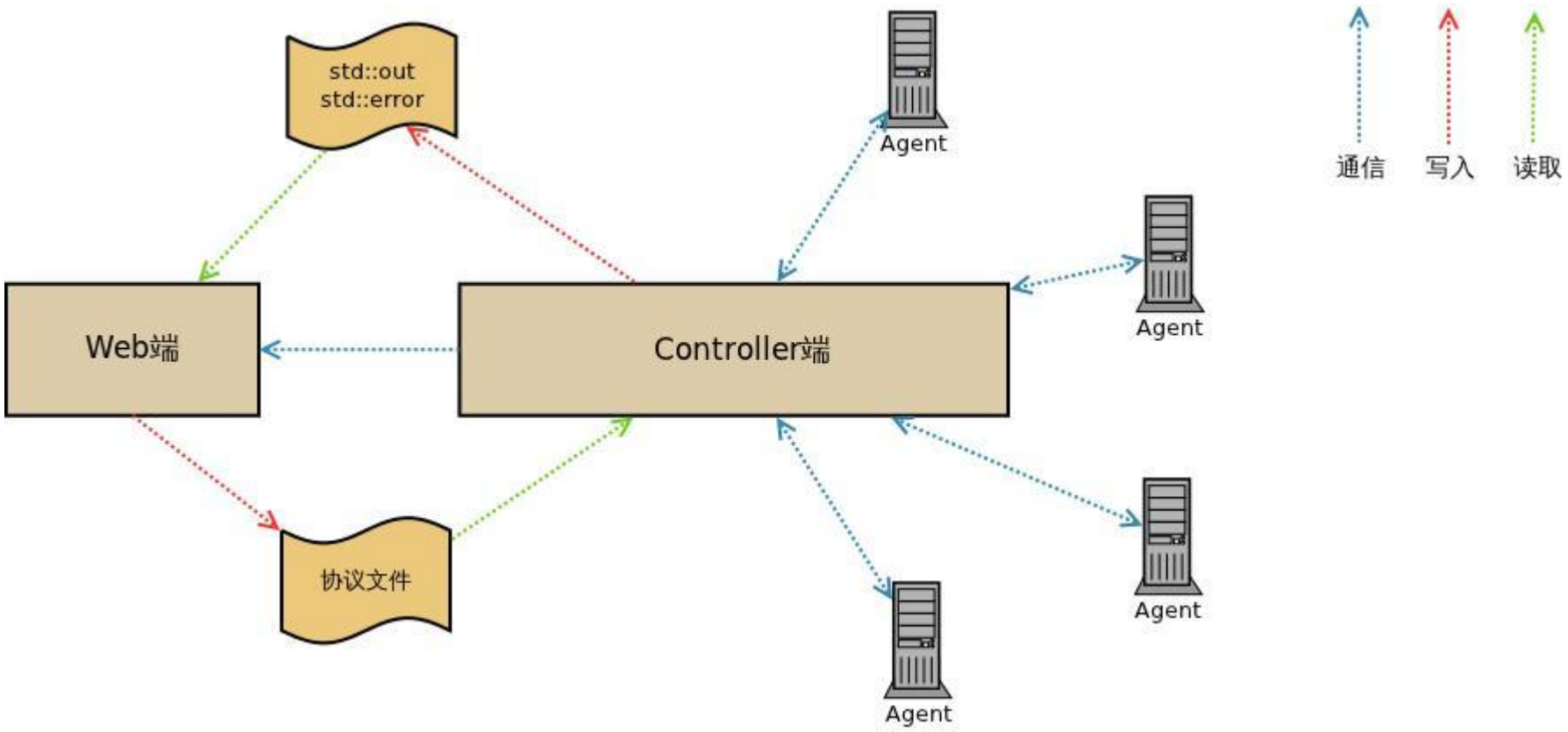
使用指南 -- 自动化任务 – 调度关系



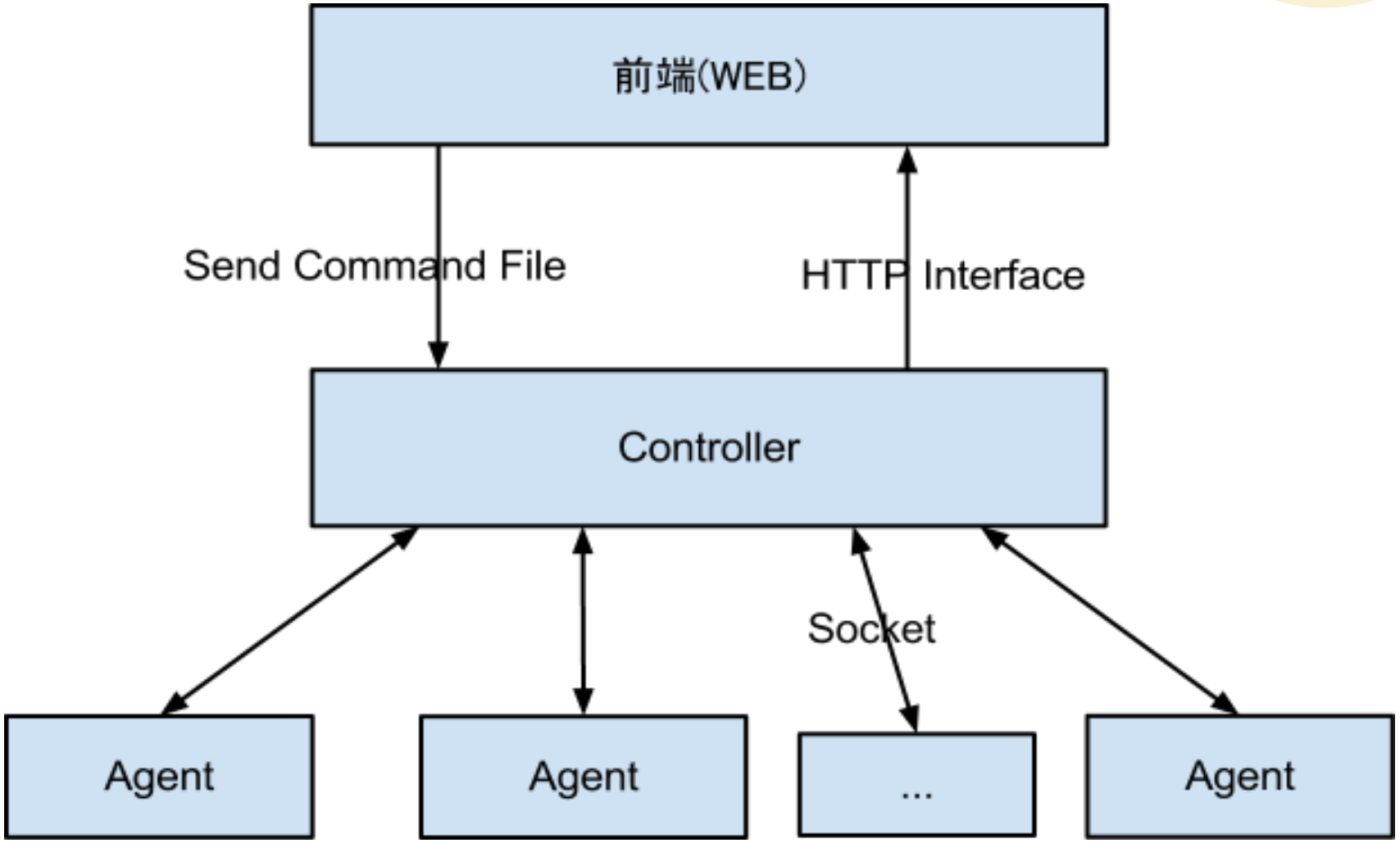
技术内幕 (Inside Toast)

- 前端 Web php
  - YII 框架
  - Highchart ， Jquery，twitter bootstrap
- 后端与前端php web FE通信
  - 本机： 文件请求 → 匿名管道
  - 跨机器：http post → soap web service
- Controller与 Test Agent通信 (基于socket)
  - 同步方式 Syncall
  - 异步方式 Asyncall
- 与 test framework/code 的集成调用
  - Executable binary (python/mmt)
  - C++ test library
- Common 工具集
  - Python 模块封装（ deploy/start services/run test binary/log ）

技术内幕 (Inside Toast) -- 设计概要



技术内幕 (Inside Toast) -- 通信



TOAST 需要改进的地方

- 测试例级别的执行和管理
- 测试环境管理
- 开放的API
- 工具集（单元测试、部署等）

关于测试工具实现的建议

- 一个工具只解决一个特定的问题（工具的基因）
- 注意NIH综合症 (Not Invented Here)
- 不要过分强调技术实现
- 以满足实际需求为工具设计目标
- 开源开放的心态