



## 【DevOps云端开发训练营】

### 2.1 – 微软DevOps工具链概述 Team Foundation Server

# DevOps 实施落地的2大法宝

## 提升效率

无论是敏捷，精益或者持续交付，其最终目的都是为了提升效率。

所谓“效率”，就是单位投入的产出量。

## 管理粒度

DevOps从管理角度的优化永远是在通过控制“管理单元”的粒度来完成的。

所谓的“管理单元”可能是团队，需求，任务，测试，交付物等任何研发中的被管理对象

粒度

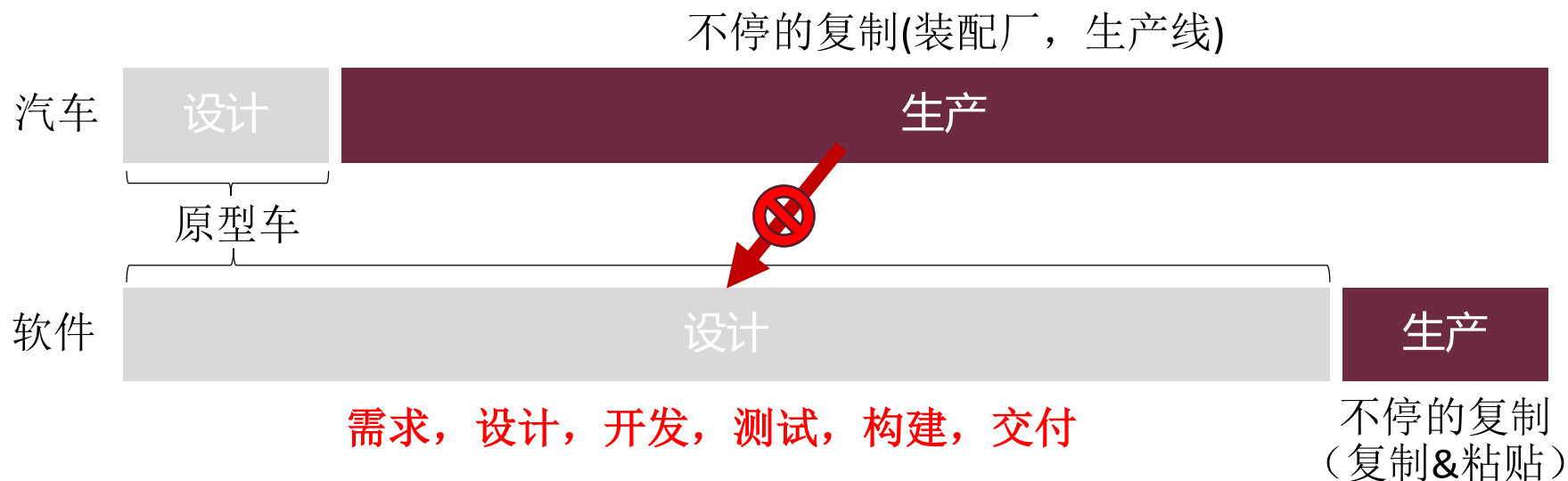
耦合

## 工程解耦

DevOps 从技术角度的优化永远是在通过解除“工程对象”之间的耦合实现的。

所谓“工程对象”可能是系统，工具，代码，模块，服务，平台，云或者任何在研发过程中存在或者交付的“技术对象”。

# 什么是软件的生产制造过程？



➡ 软件实现的整个过程都处于 **设计阶段**

➡ 设计过程是 **创造** 过程

➡ 如何 **计划** 不确定的 **创造** 过程？

**行不通**

你觉得你看到的公路是这样的 ...

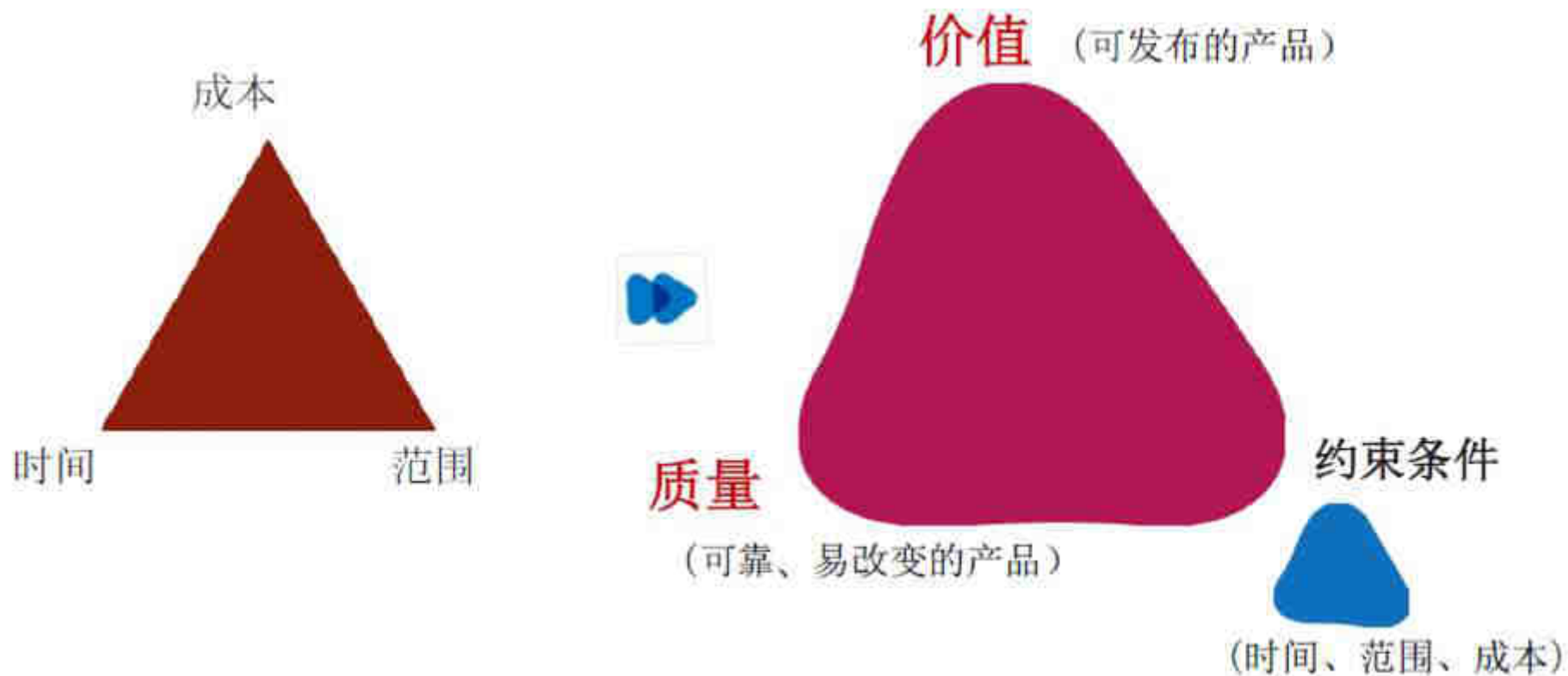


其实 ...





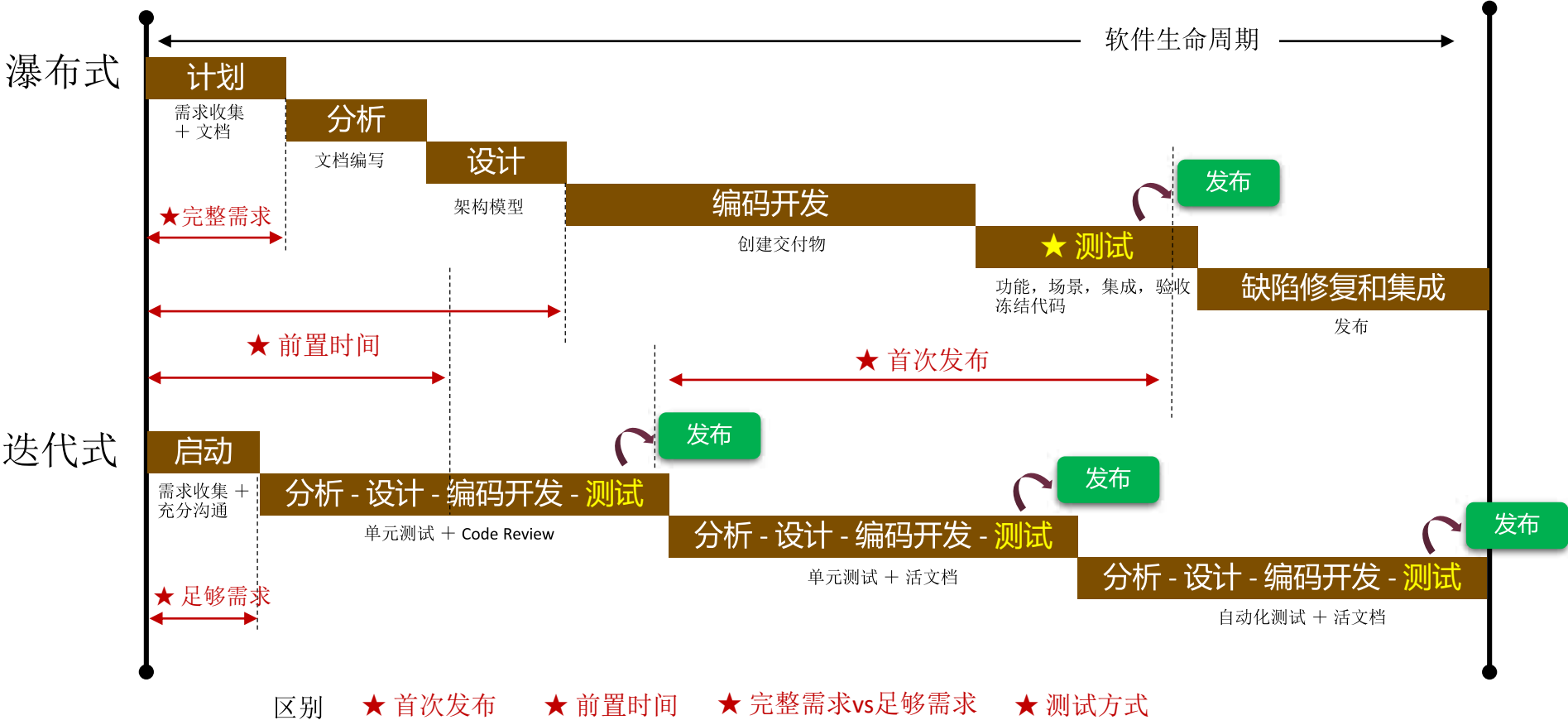
# 敏捷让我们重新定义管理！



传统研发更关注于内向型指标，没有从整体性上考虑问题。  
敏捷要求我们引入用户，DevOps要求我们具备全局观！应从外向型指标评价过程！



# 传统开发 vs. 敏捷开发



# 计划不是用来限制变化的，而是用来适应变化的！

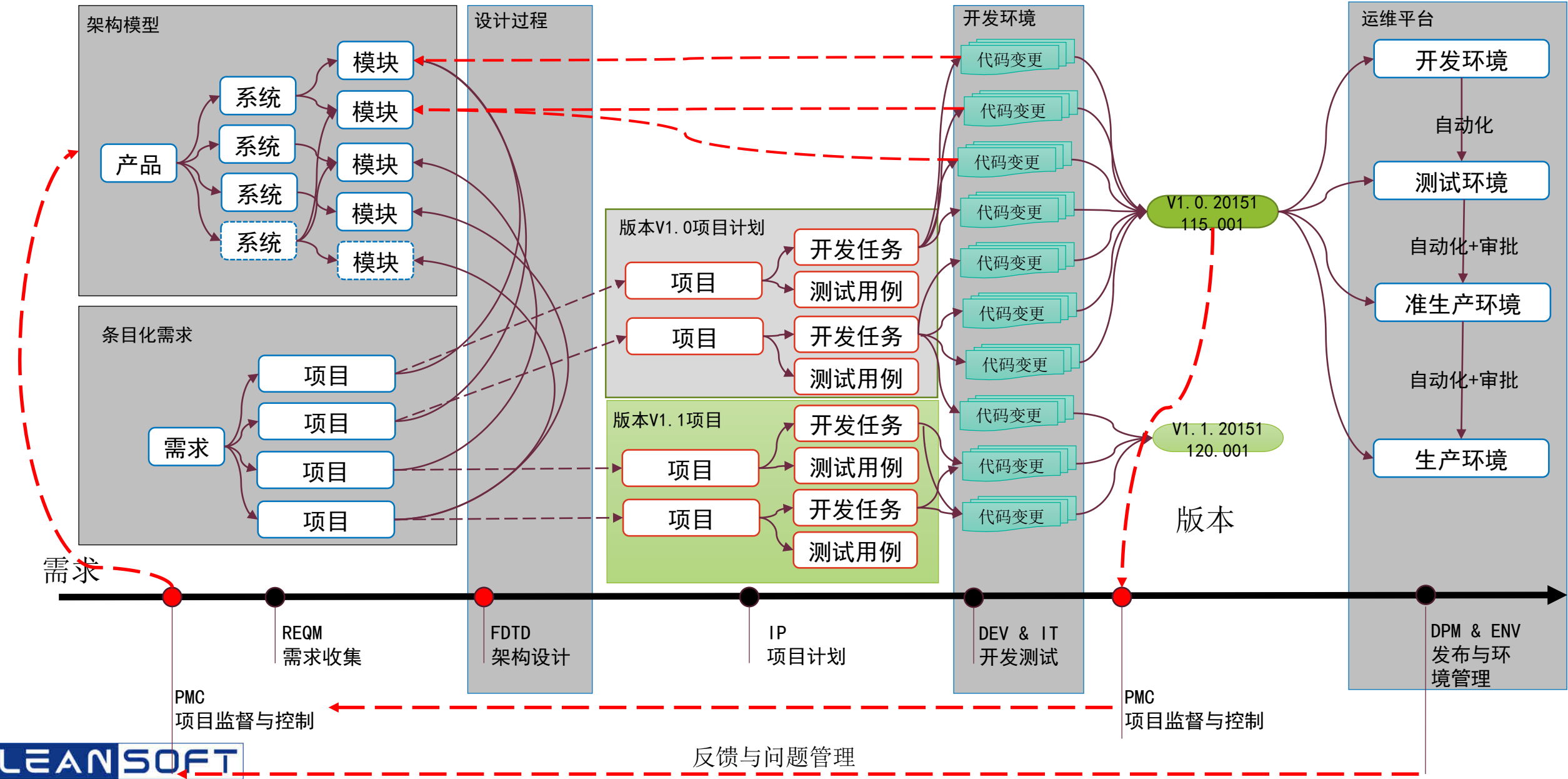
计划本身也是“管理单元”，计划对变化的适应能力来源于计划本身“粒度”的缩小。



软件开发是一个复杂过程。

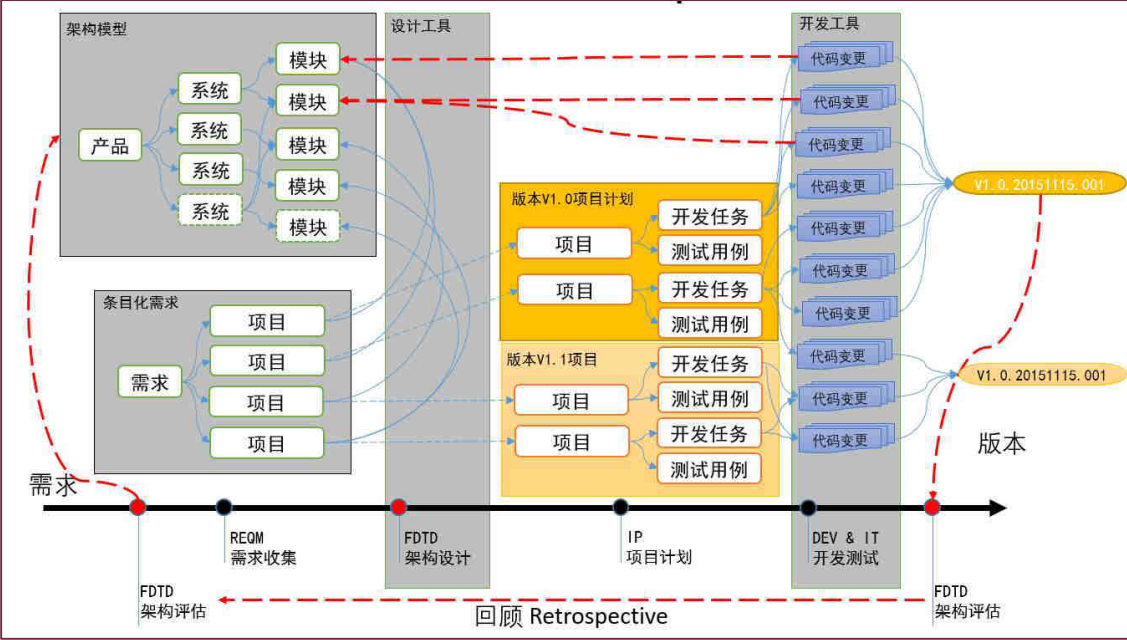
不要试图用复杂方法处理复杂过程，尝试将复杂过程简化成简单过程，再用简单方法处理简单过程。

# 软件研发管理过程全景

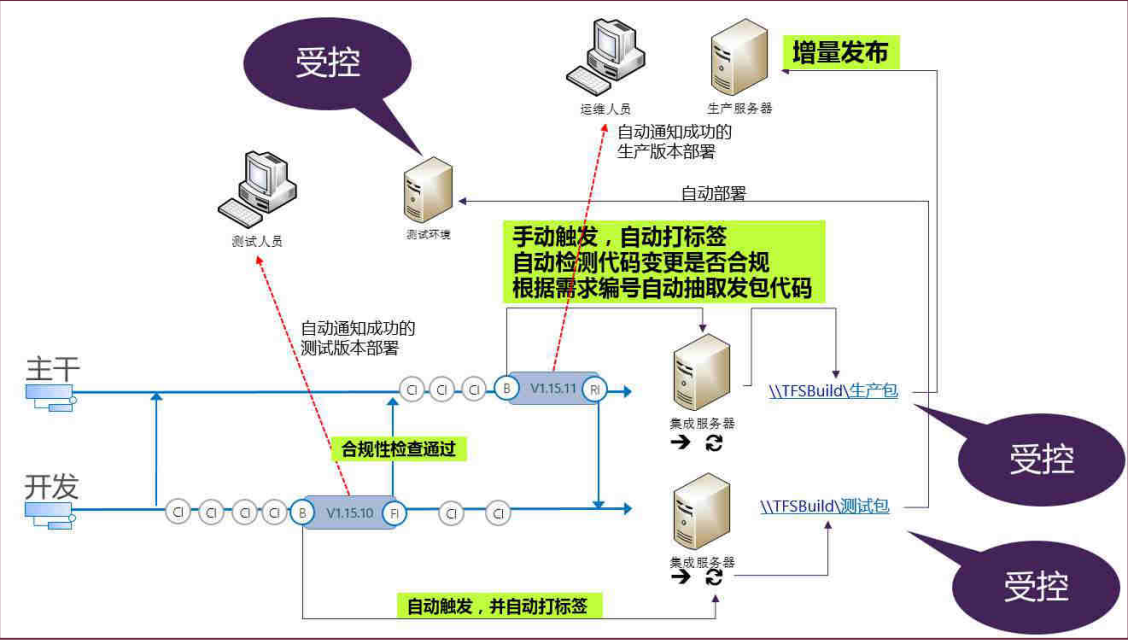


研发过程改进，就是对上图中的点和线建立对应的**管理单元**的过程；并将这些管理单元形成**管理体系**。

# 软件开发过程：管理属性和工程属性



管理属性 - 规划版本



工程属性 - 交付版本

## 规划版本（希望做什么？）

TFS使用工作项提供端到端的需求版本管理能力，每个工作项上都可以设置“迭代路径”字段代表需求所属的“规划版本”；而与这一需求相关的任务/测试用例/缺陷/问题等也都一同归属这一“规划版本”。这样我们就可以用版本号来对开发过程中的所有工件进行查询，分析和报表展现，提供了“规划版本”的管控能力。

## 统一的版本号

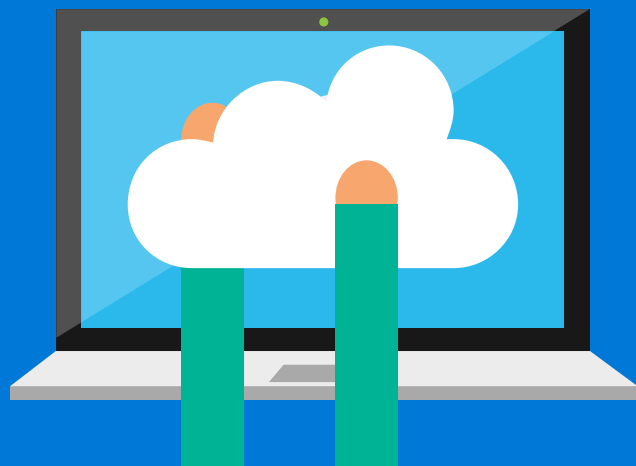
根据开发团队及软件产品的本身的复杂程度，以上“规划版本”和“交付版本”可以使用多级版本号进行标识，如：左图中的规划版本V1.0提供了2级规划版本（大版本.小版本）。而“交付版本”则在“规划版本”基础上继续添加2级，形成【大版本.小版本.编译日期.序列号】的4级版本结构。

## 交付版本（实际做了什么？）

某一规划版本中的任务等工件驱动开发人员完成编码后，开发人员可以将代码变更与“规划版本”进行关联；同时TFS构建服务会自动生成“交付版本”号，并将其所包含的代码变更与之关联，因而形成了从“规划版本”到“交付版本”的跟踪能力。同时，使用TFS构建服务还可以保证交付版本完全受控，确保开发/测试/交付版本的一致性。

管理属性和工程属性的衔接点，就是  
版本管理。

# Agenda



- Visual Studio & DevOps 概述
- DevOps反馈环
- 微软 DevOps 解决方案
- Visual Studio相关功能和优势



DevOps 是应用开发领域的  
新趋势，帮助团队专  
注于用户价值



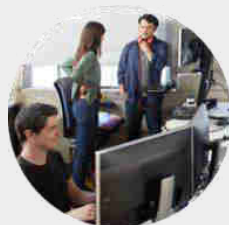
# DevOps 三大核心



## 人员

频繁的协作  
共同的目标  
持续改进

*协同更多人员一起工作*



## 过程

消除浪费  
提高效率  
加速反馈

*加速价值交付*



## 工具

提升生产力  
支撑协作  
协助探索

*为DevOps转型提供支撑*

# DevOps 最佳实践

## 实践

自动化测试  
持续集成  
持续部署  
发布流程管理

加速价值流动

优化组织架构  
企业战略协同

## 实践

企业级敏捷  
持续集成  
持续部署  
发布流程管理

## 实践

行为监测  
数据收集  
在生产中测试  
干系人反馈

基于反馈的  
需求分析过程

生产环境监测

## 实践

在生产中测试  
行为监测  
用户监控  
干系人反馈  
功能开关

## 实践

代码评审/走查  
自动化测试  
持续度量

管理技术负债

## 实践

应用性能监控  
基础设施即代码  
持续交付  
发布流程管理  
配置管理  
自动故障恢复

专注于生产环境

## 实践

应用性能监控  
基础设施即代码  
持续交付  
发布流程管理  
配置管理  
自动故障恢复

更敏捷的基础设施

# 软件工程最新趋势



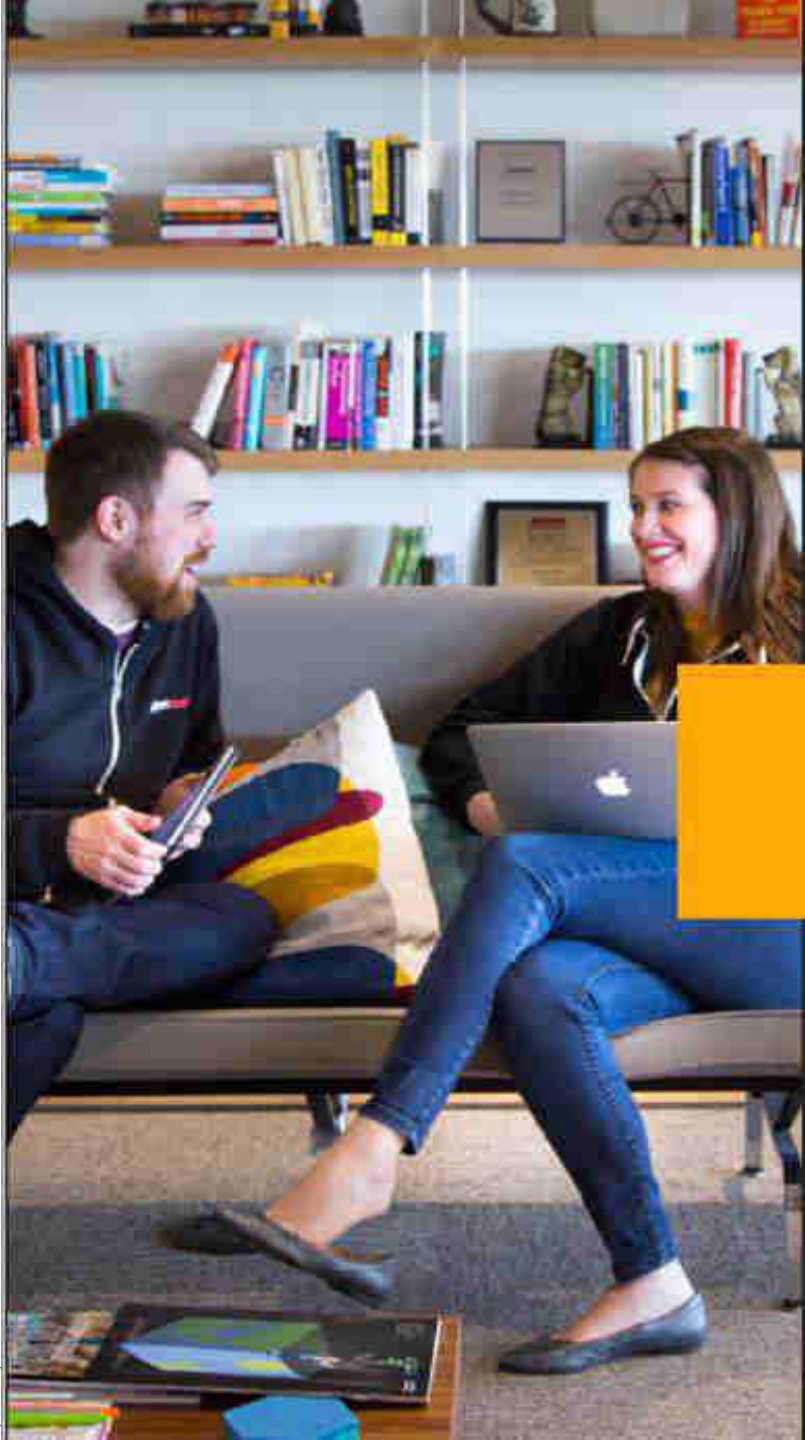
持续交付



持续监控



持续学习



2016

# State of DevOps Report

Presented by:



Sponsored by:



# DevOps 价值

- 速度

- 提升部署速度 200x，加快问题修复速度 24x

- 质量

- 降低变更错误率3x，降低线上问题修复时间 2500x，减少22%的意外问题

- 安全性

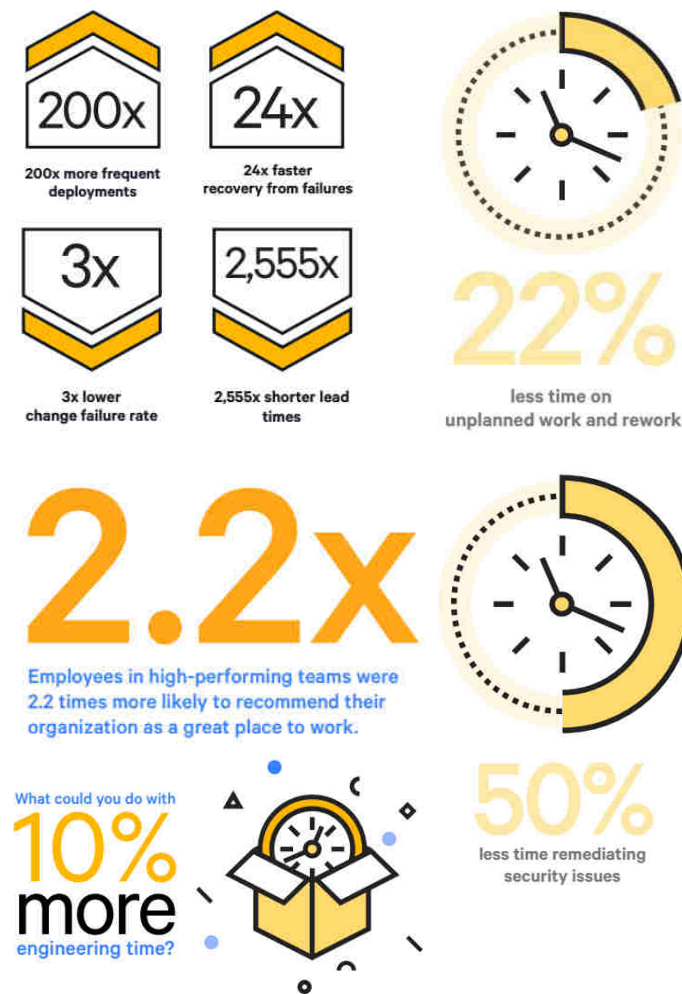
- 减少一半的安全性问题

- 员工满意度

- 更高的工作热情和主动性，更愿意向其他人推荐自己的企业和产品

- ROI投入产出比

- 减少重复性工作，减少宕机时间，让员工有更多的时间投入更有意义的工作，如新功能开发

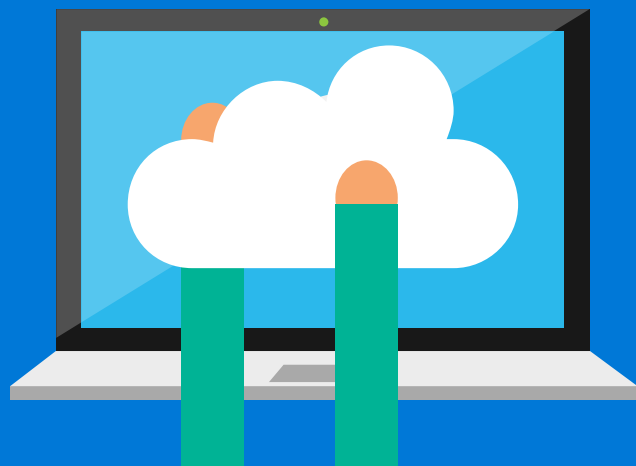


数据来源：“PuppetLab 2016 State of DevOps Report”中汇集了过去5年中超过2,5000名来自全球的技术专业人士的调研结果。

<https://puppet.com/blog/2016-state-of-devops-survey-here>

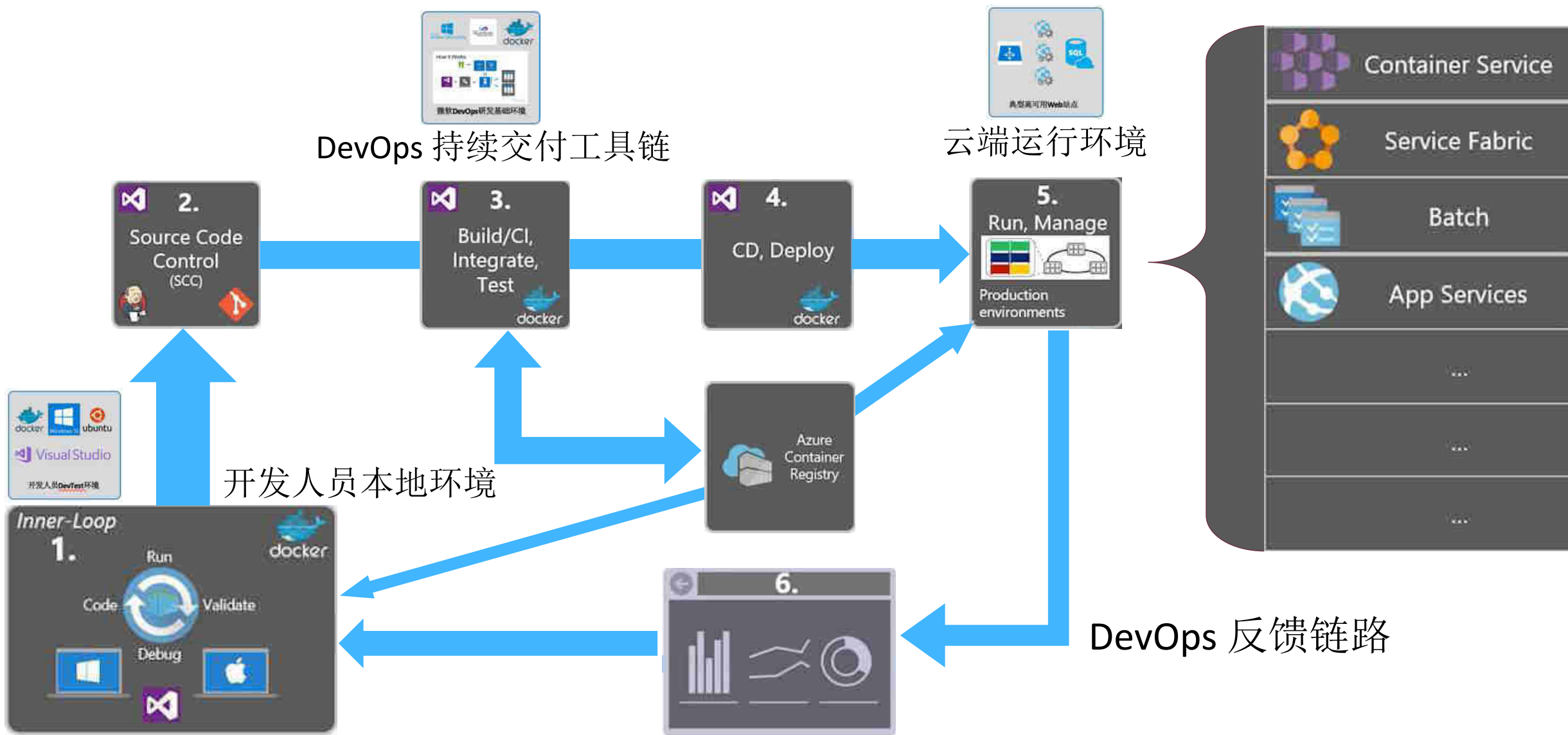


# Agenda

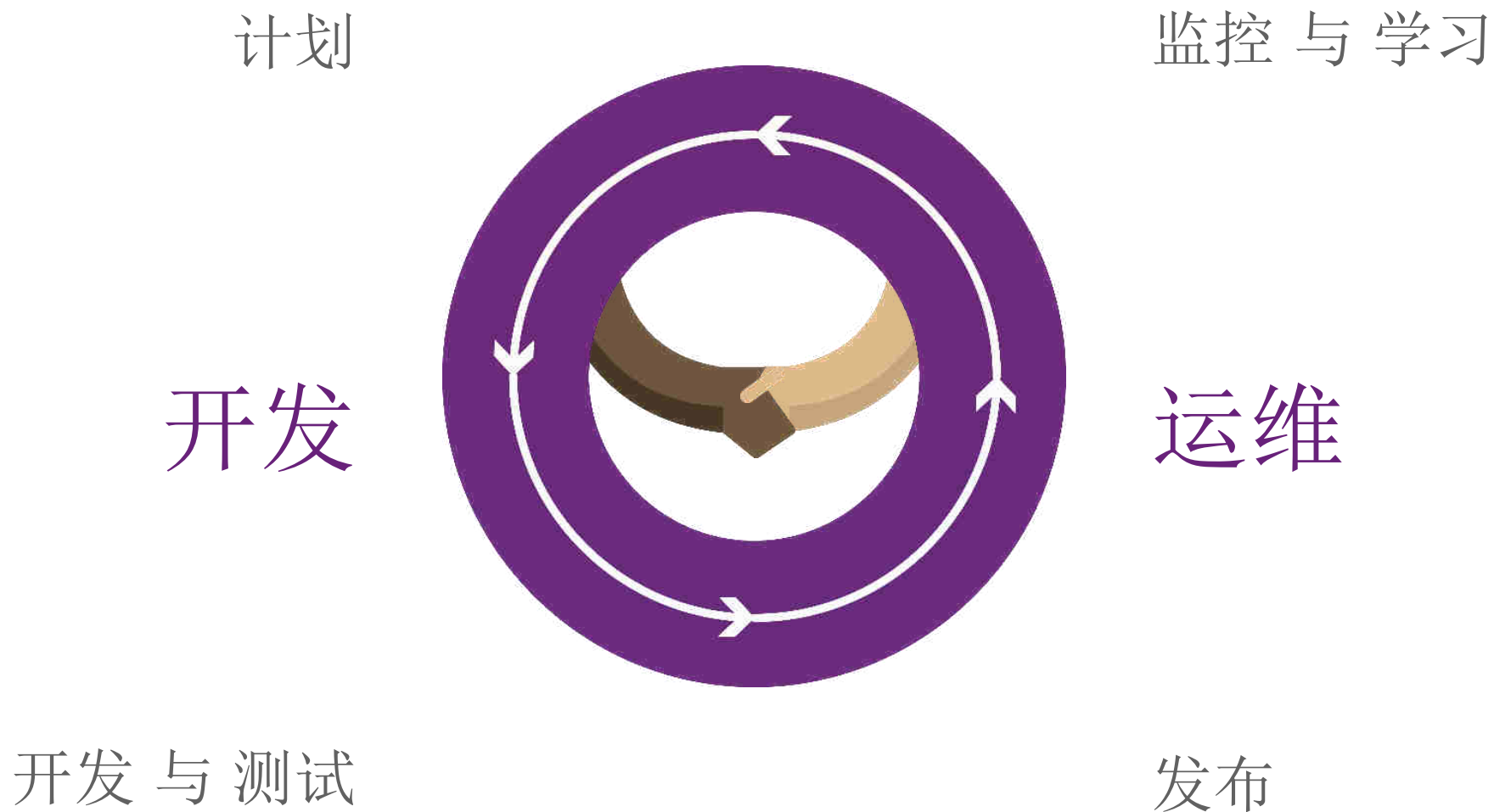


- Visual Studio & DevOps 概述
- DevOps反馈环
- 微软 DevOps 解决方案
- Visual Studio相关功能和优势

# DevOps 反馈环路



# 整合的生命周期管理



# DevOps 驱动力

计划

监控 与 学习

**X** 研发投入无法与最终用户场景的重要性保持一致的优先级

**X** 生产环境性能监控和问题定位困难

开发

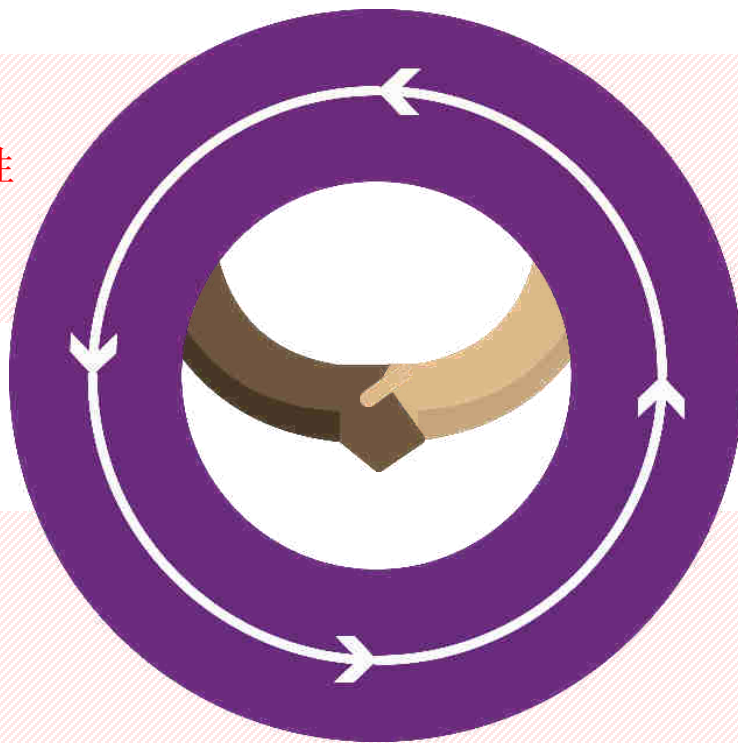
运维

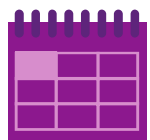
**X** 借助敏捷方法提升开发效率

**X** 运维团队无法适应开发团队的快速发布速度

开发与测试

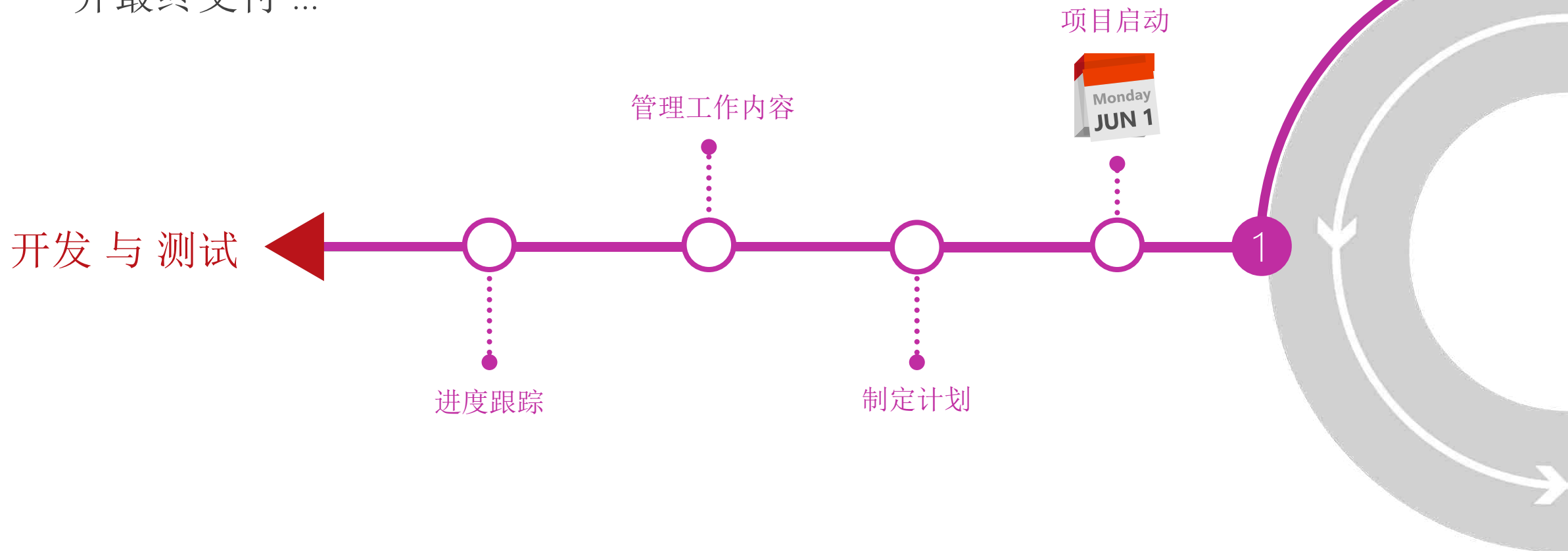
发布





# 计划阶段

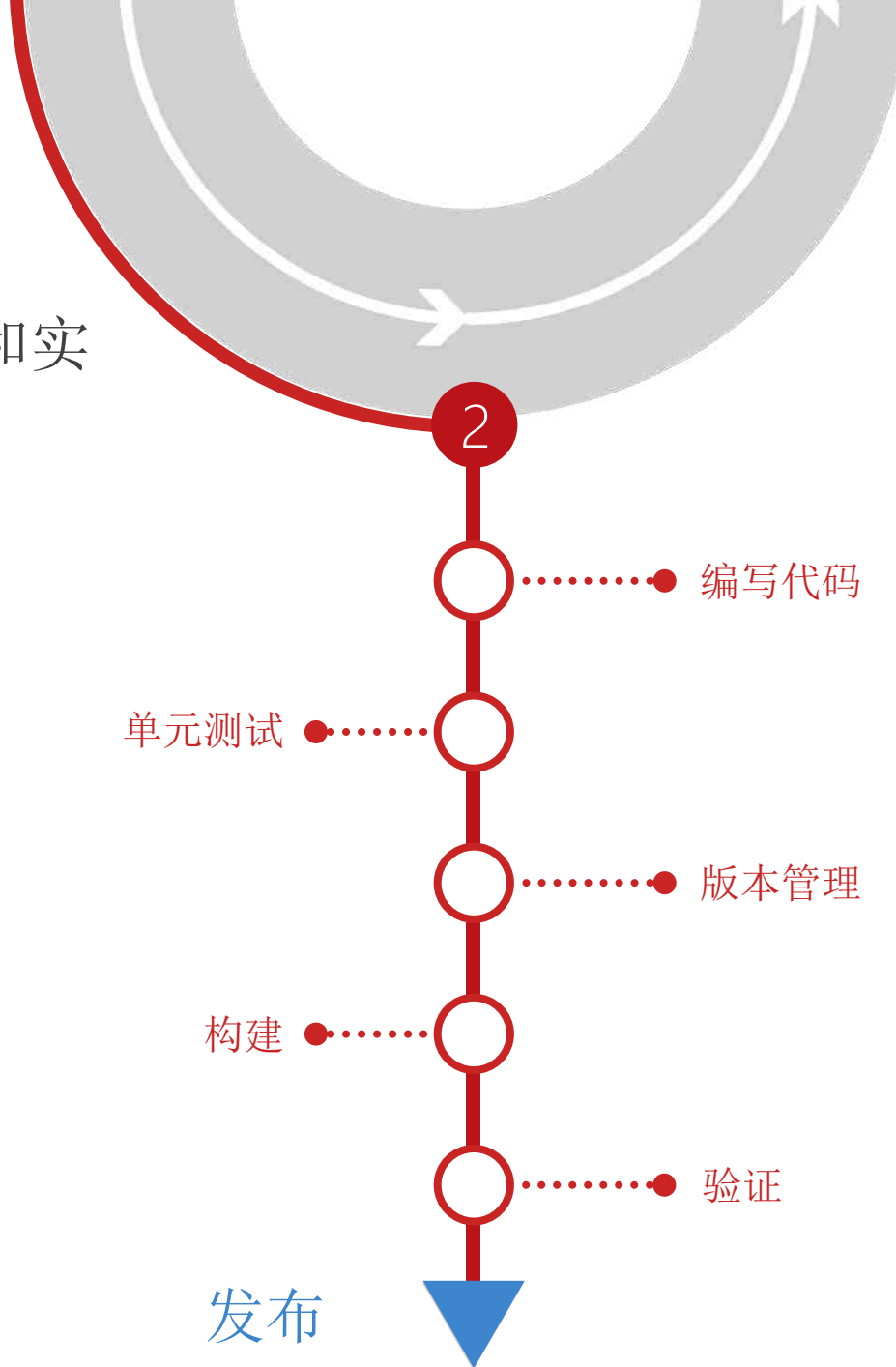
从一个简单的创意开始，制定计划  
并最终交付 ...





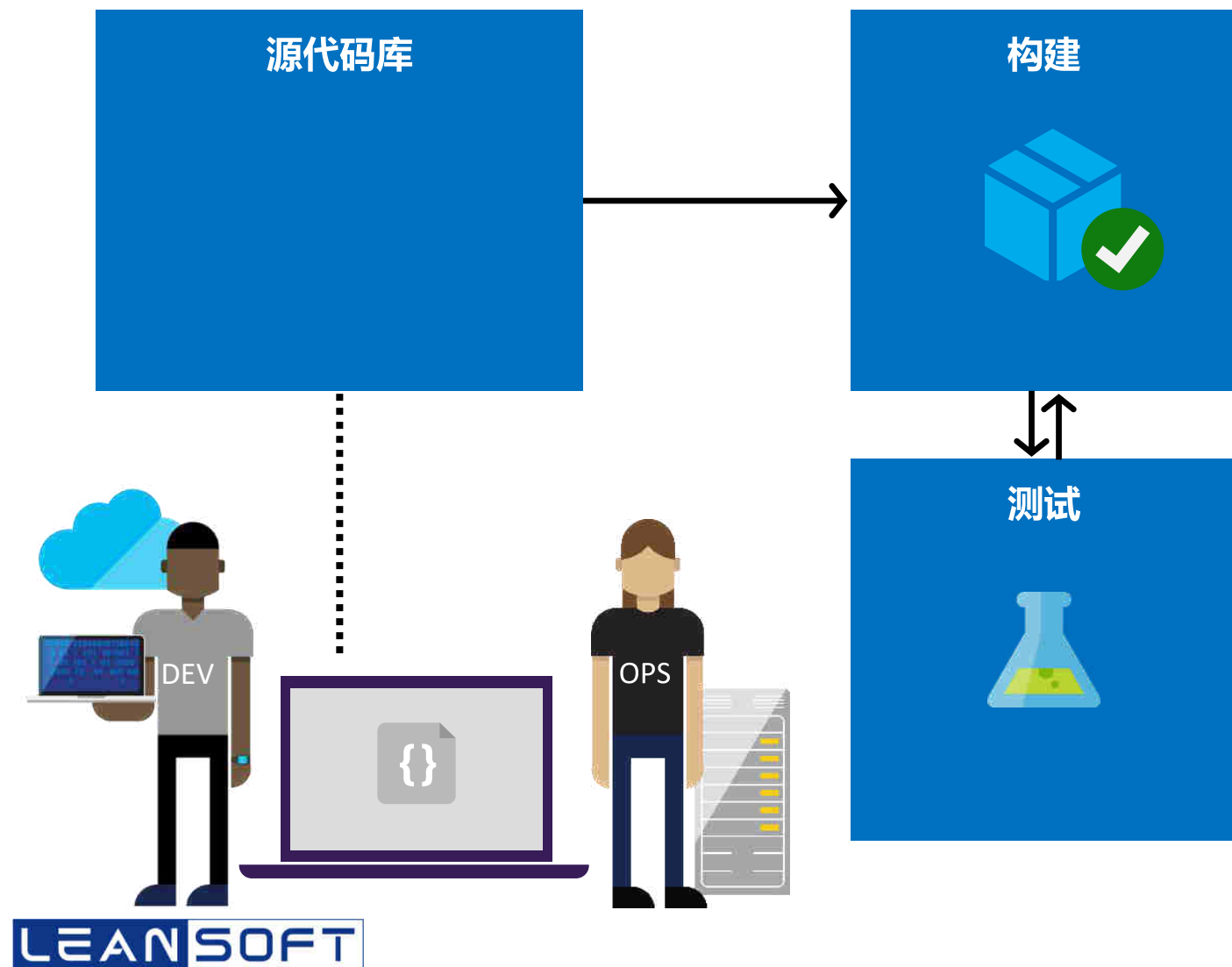
# 开发与测试阶段

迭代开始，开发人员开始编码和实现需求 ...





# 持续集成 (CI) – DevOps 实践



## 问题

- 无法按时交付
- 无法工作的/低质量代码
- 不完整用户场景
- 重复劳动

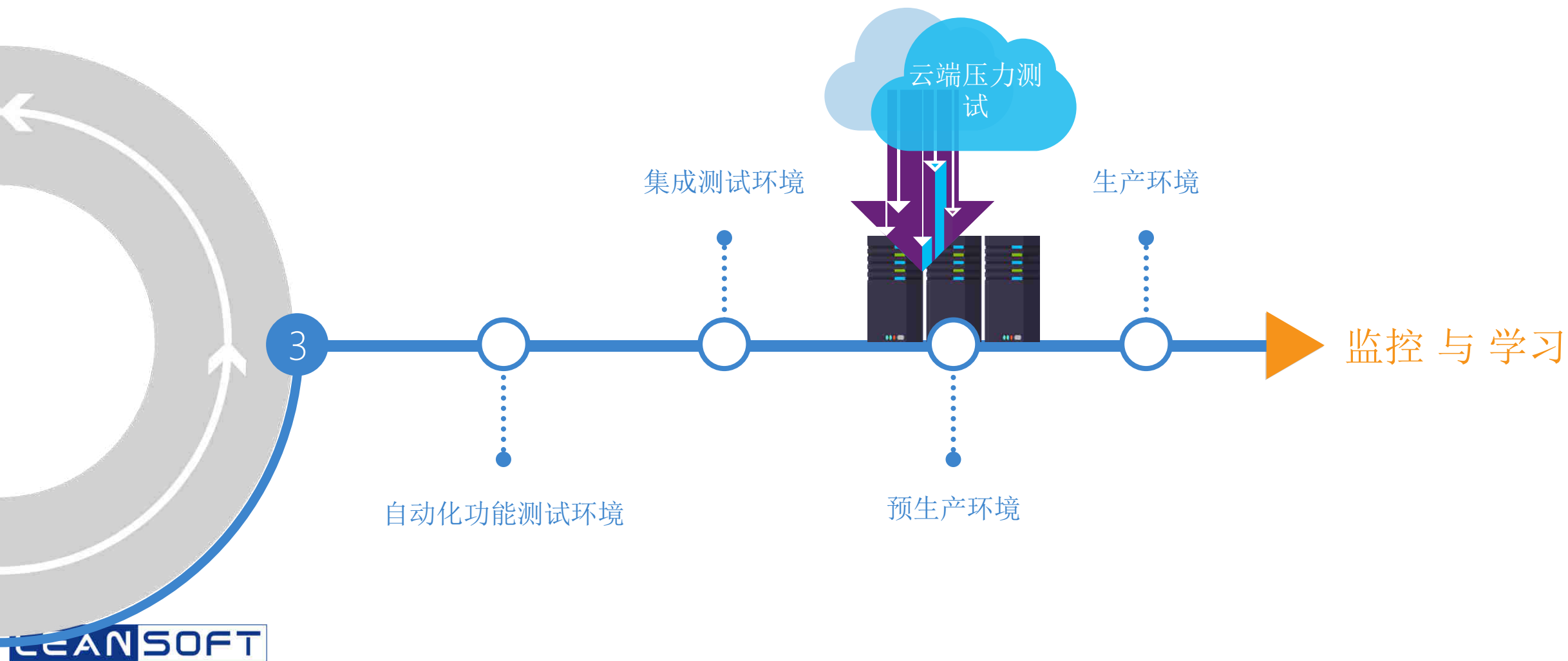
## 价值

- 频繁集成代码
- 提升质量
- 可重复的验证过程



# 发布阶段

通过测试的代码，被部署到测试环境，预生产环境和生产环境，形成完整的交付管道（Release Pipeline）



# 持续部署 (CD) – DevOps 实践

源代码仓库



开发环境



测试环境



生产环境



## 问题

- 缓慢的发布节奏
- 部署成功率低且不可控
- 复杂的部署过程

## 价值

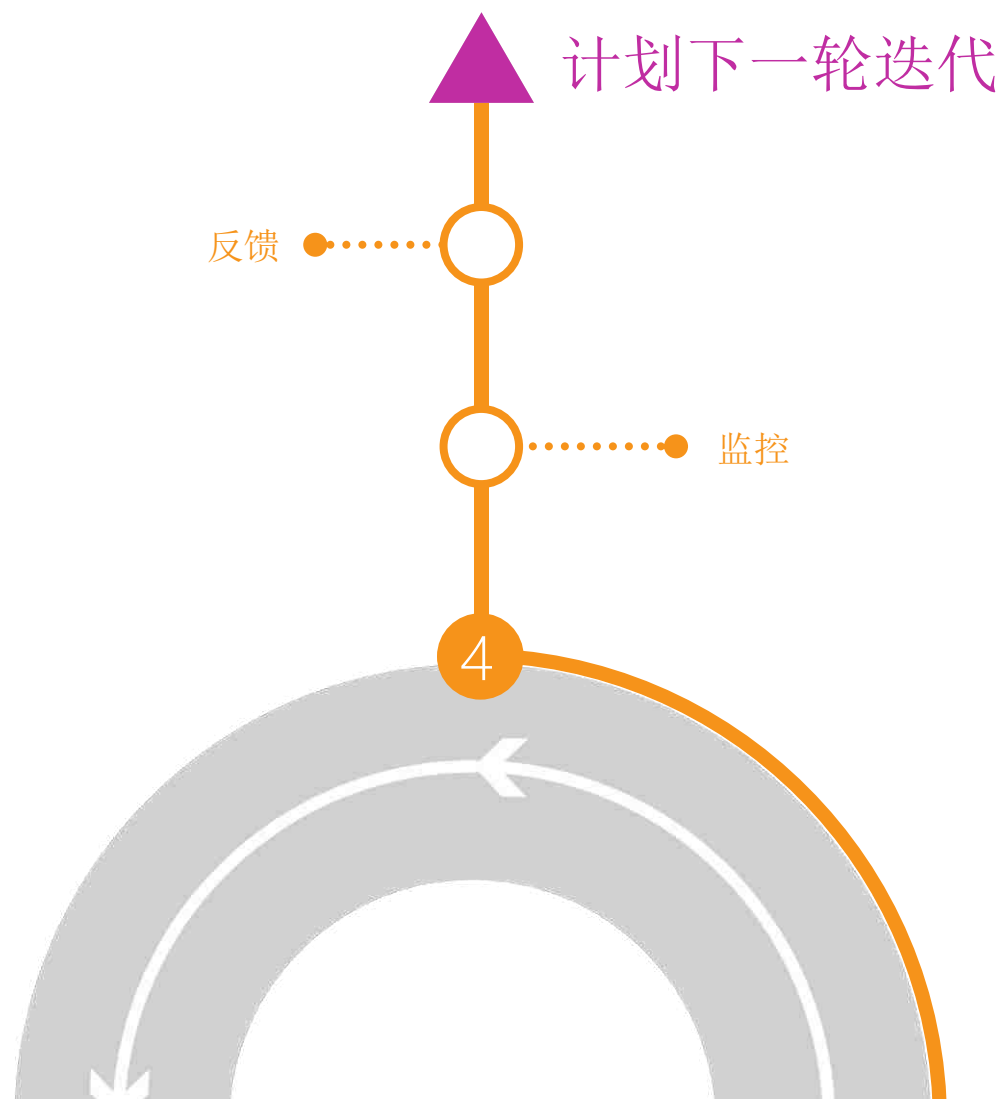
- 一致性
- 快速部署
- 可重复性
- 避免人为错误



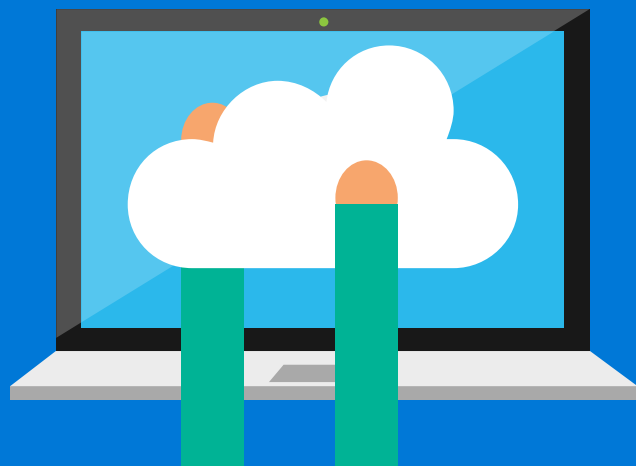


# 监控 + 学习

了解和分析用户行为，据此修复问题并规划下一版本



# Agenda



- Visual Studio & DevOps 概述
- DevOps反馈环
- 微软 DevOps 解决方案
- Visual Studio相关功能和优势

# Visual Studio Team Services



# Team Foundation Server



计划与跟踪

源代码管理

发布包管理

质量管理

跨平台构建

持续部署

发布管理

反馈管理

应用监控与分析

可扩展，可定制&第三方集成



# DevOps 微软生态

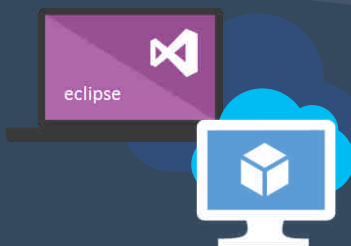
人员 | 过程 | 工具



## 01

Develop

开发环境



团队协作平台

Team Foundation Server

Visual Studio Online

Workstations - On-Premises | Hybrid | Cloud

## 02

Build & Test

持续集成

Team Foundation Server

Visual Studio Online

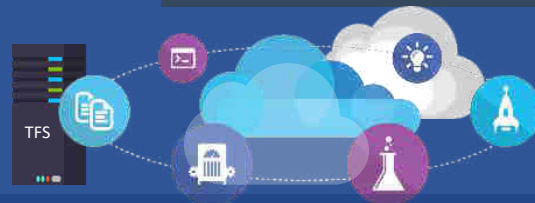
Release Management  
for Visual Studio

Test

Team Foundation Server

Visual Studio Online

Microsoft Test Manager



Services - On-Premises | Hybrid | Cloud

## 03

Deploy

持续部署

Microsoft  
System Center

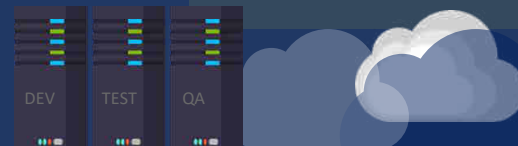
Release Management  
for Visual Studio

Automation  
Service

PowerShell | WAML

Azure  
Resource  
Management

xPlat Command Line



Environments - On-Premises | Hybrid | Cloud

## 04

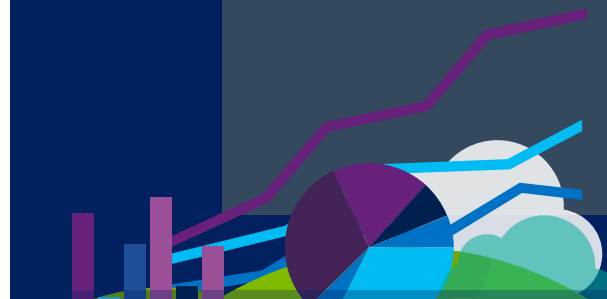
Monitor & Learn

监控

Microsoft  
System Center

Visual Studio Online

Application Insights



Monitoring - On-Premises | Hybrid | Cloud

# DevOps 混合生态

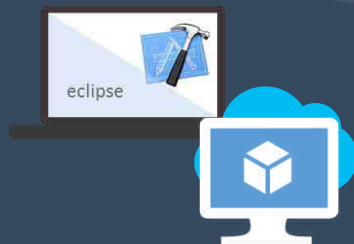
人员 | 过程 | 工具



## 01

Develop

开发环境



Team Collaboration

GitHub  
CodePlex

## 02

Build & Test

持续集成

gradle

GRUNT

Jenkins

Hudson

Test

gradle

GRUNT

## 03

Deploy

配置管理



Release

gradle

GRUNT

Jenkins

Hudson

VAGRANT

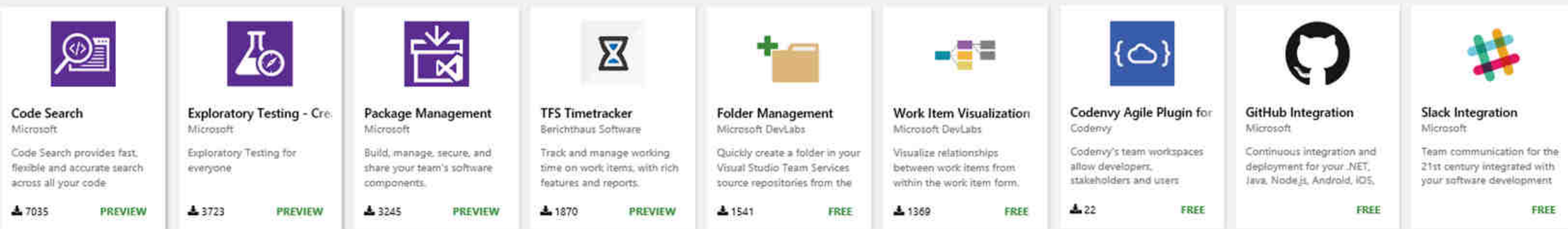
## 04

Monitor & Learn

监控

Nagios

ZABBIX



# Visual Studio插件库

65

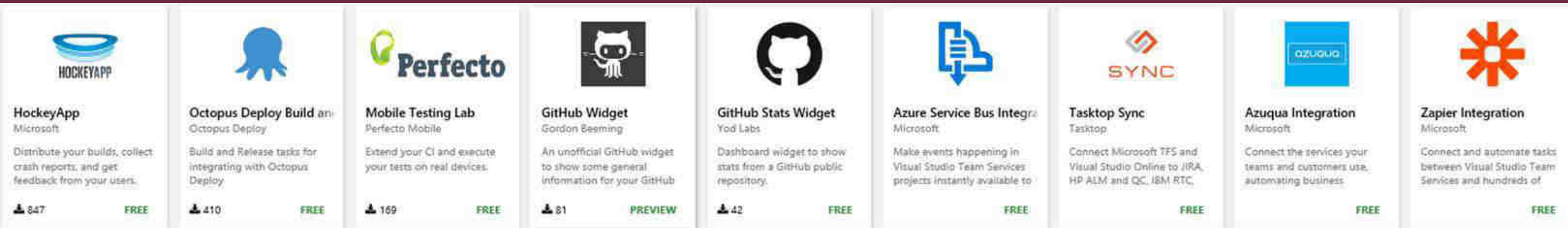
Visual Studio Code  
插件

5,910

Visual Studio  
插件

48

VSTS/TFS  
插件



# 企业级敏捷

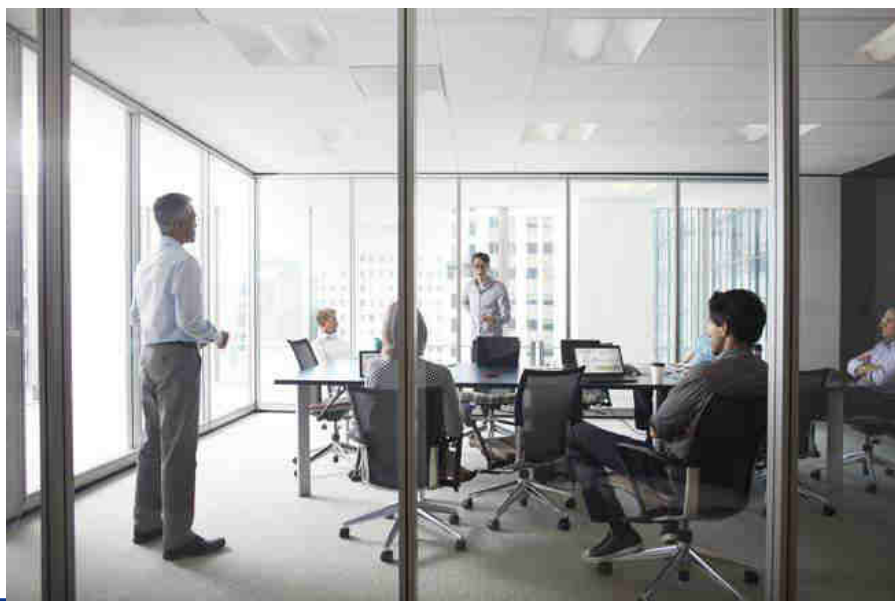
## 协作

跨团队跨项目的持续沟通与协作

轻量级的需求管理，保持团队与项目干系人的统一理解

## 生产力

使用统一的管理平台，管理backlog，迭代计划和日常工作，跟踪和管理进度

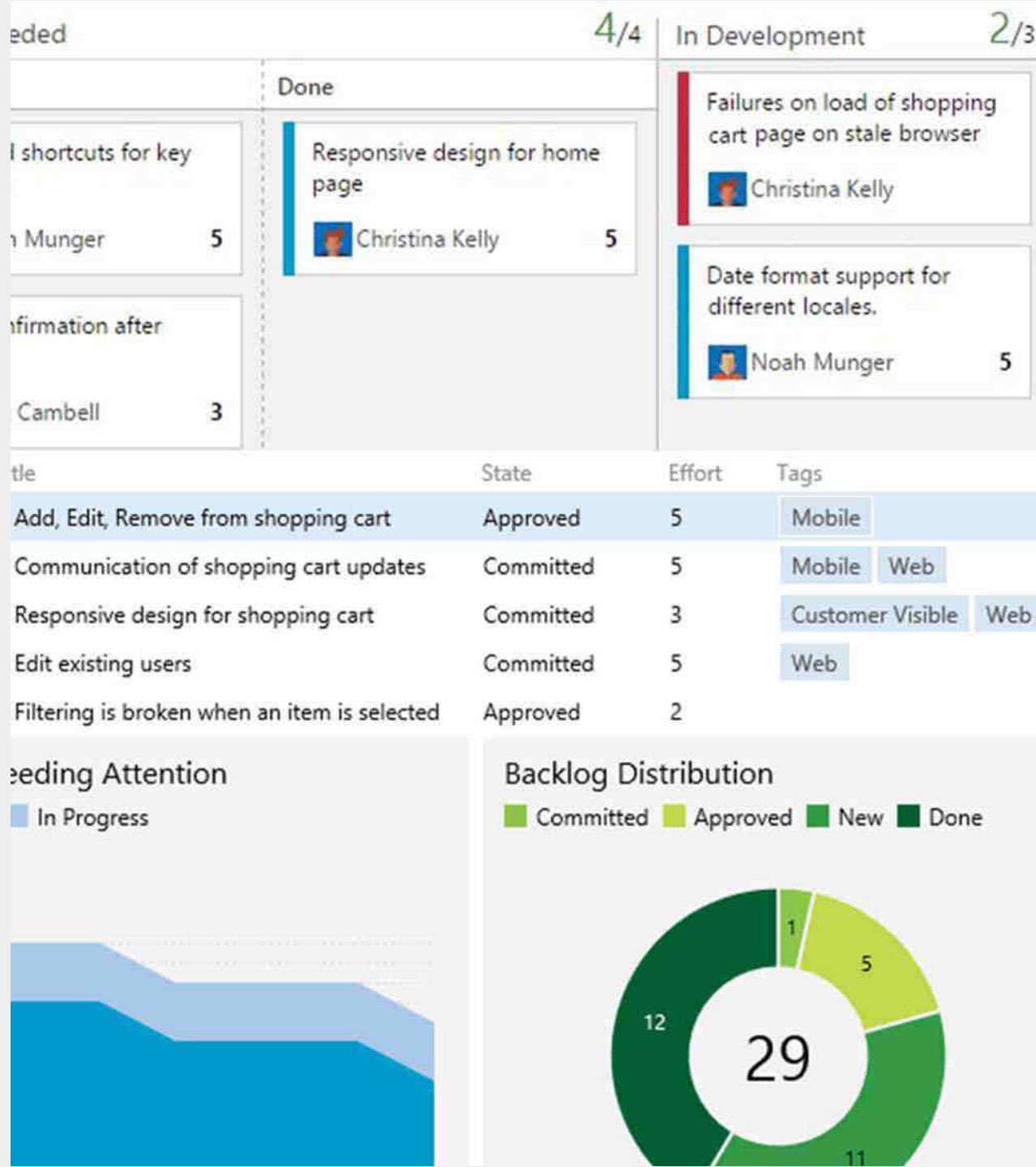


# 敏捷规划工具

## 企业级敏捷需求管理工具

使用企业级的KANBAN工具统一管理所有团队的工作，包括生产环境的反馈。

跟踪所有团队和人员的进度，确保价值持续交付。

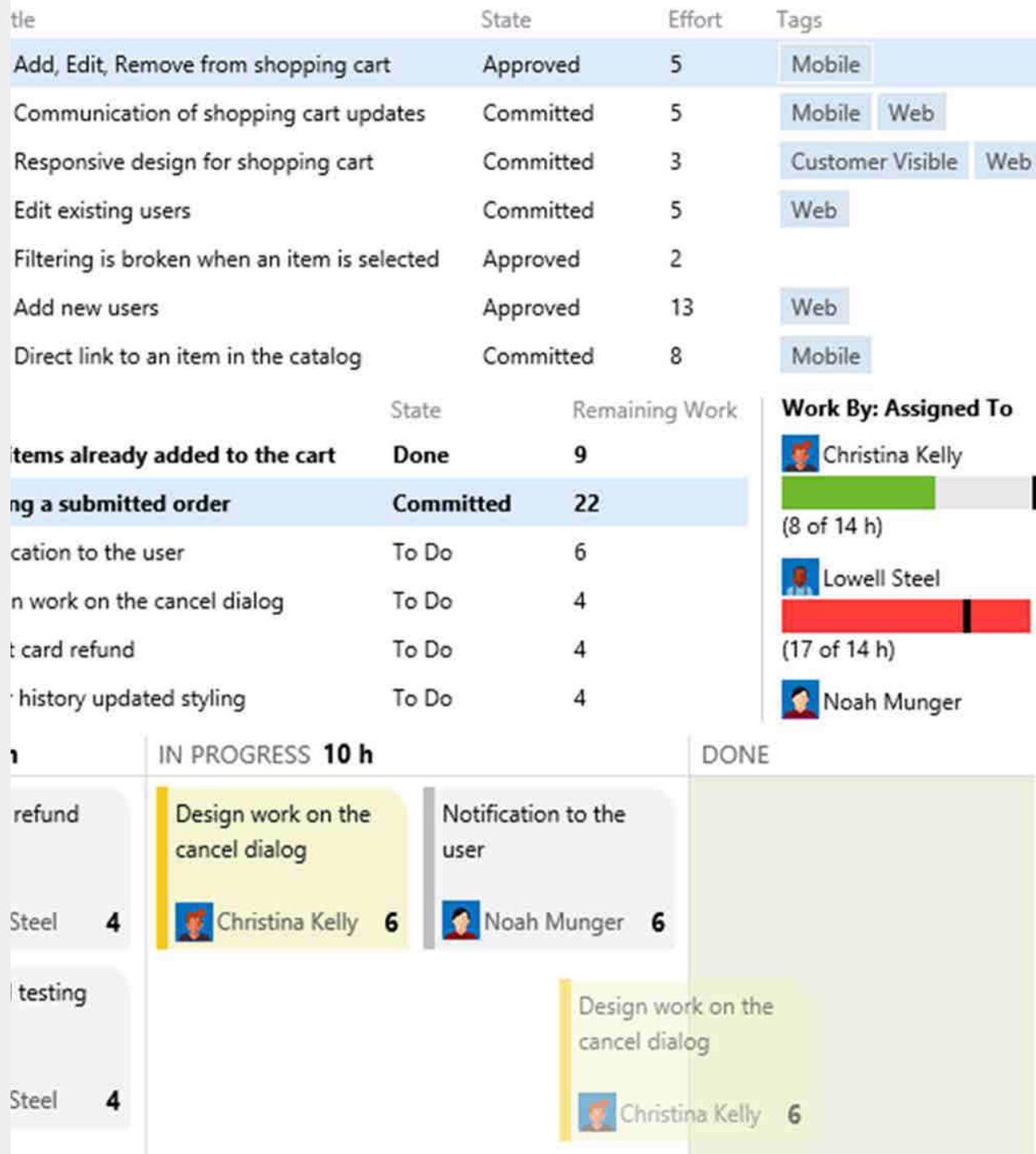




# 敏捷项目管理

## 完全支持Scrum

使用Scrum团队熟悉的用语和模式，包括冲刺，基于故事点的工作量评估，工作分析，监控和内置的燃尽图。



# 电子 Kanban

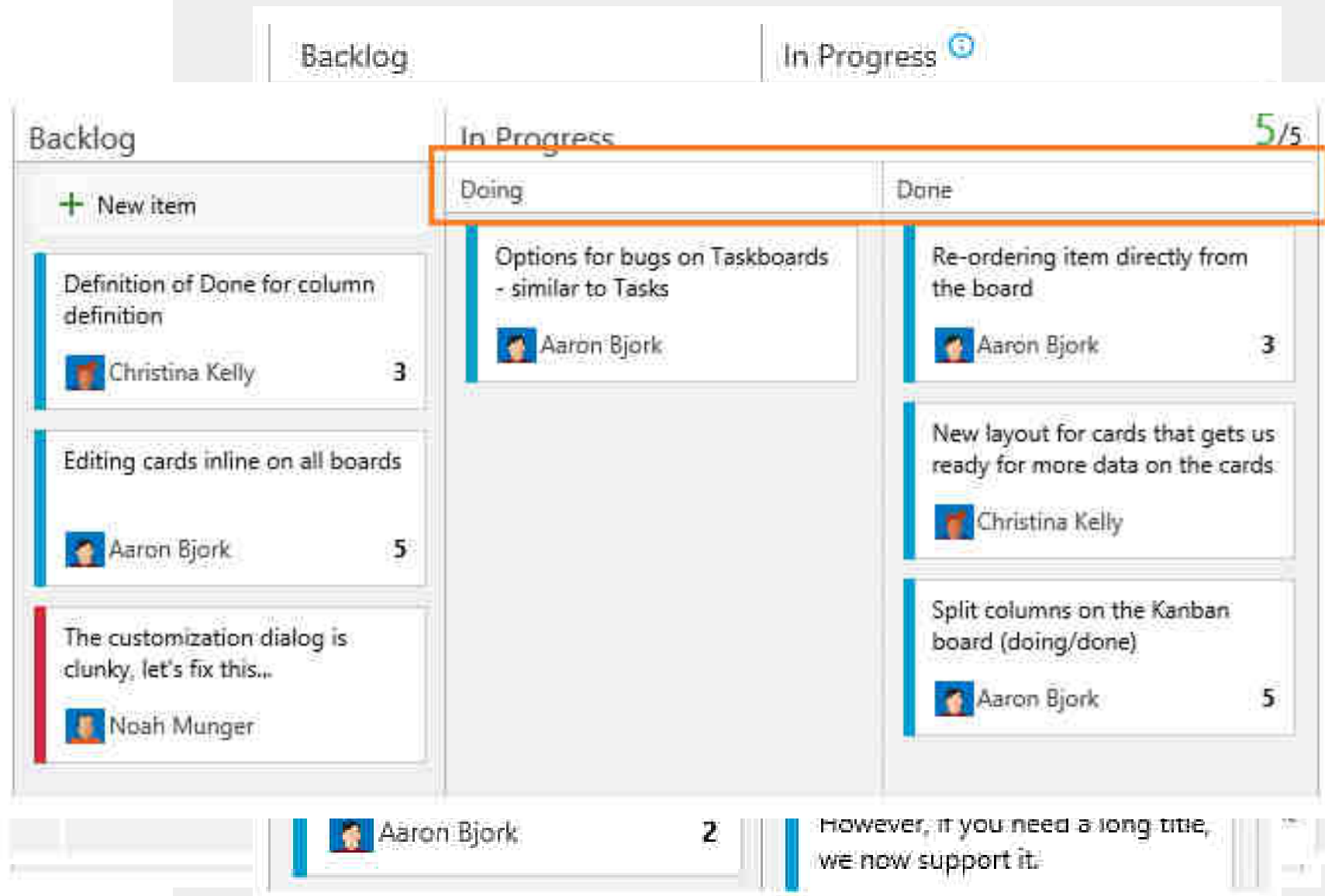
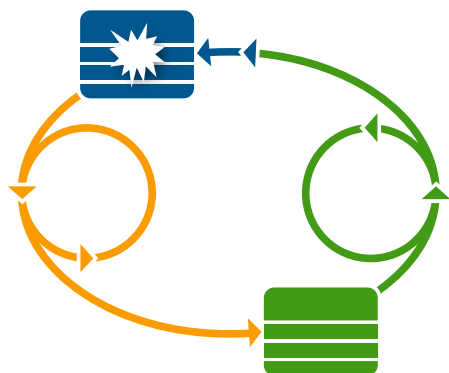
卡片添加 / 编辑

泳道

优先级调整

缓冲区

完成规范

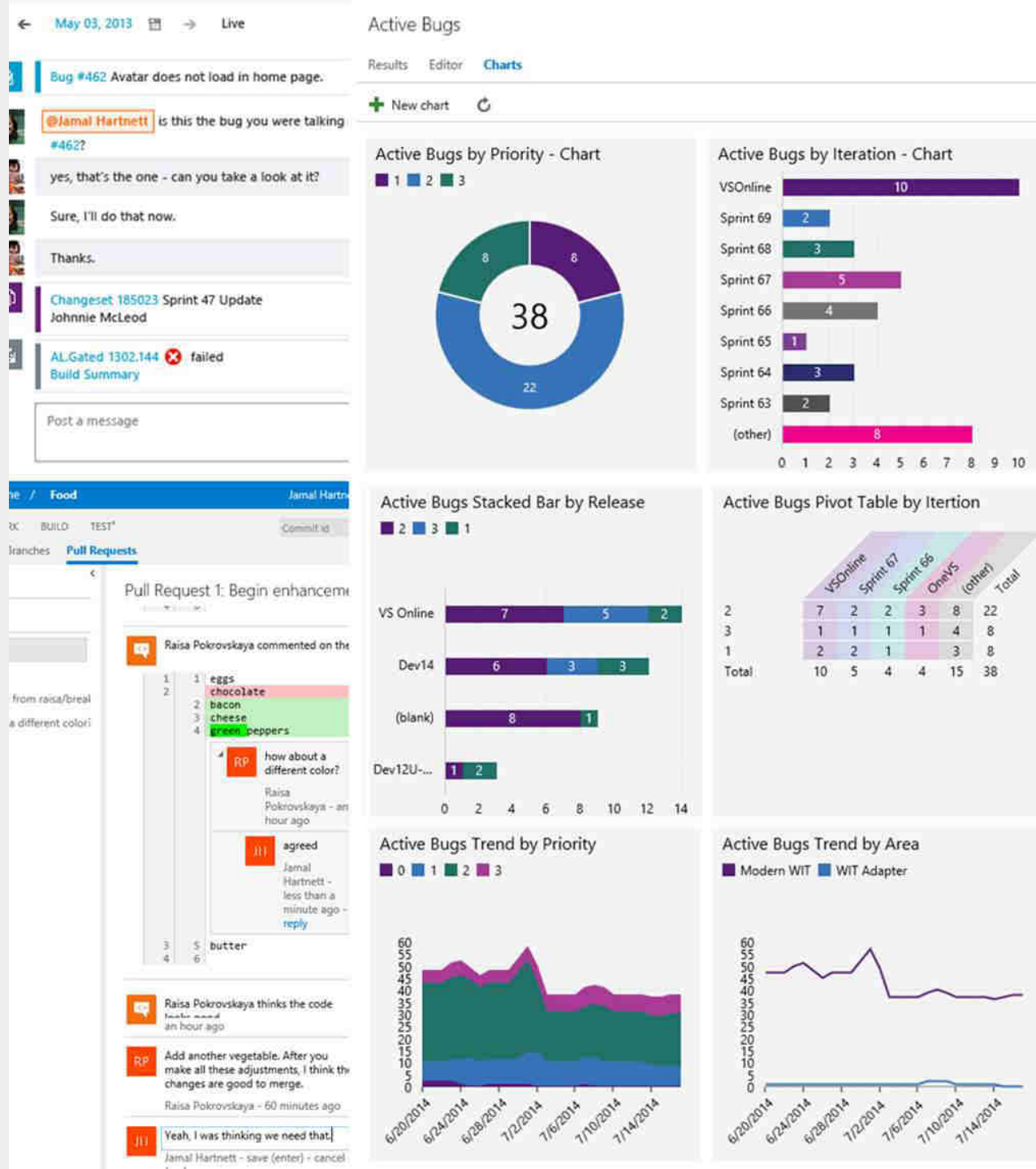


# 仪表盘 即时图表

## 仪表盘

使用灵活定制化的仪表盘为你的团队和干系人提供他们所需要的信息。

集中显示所有相关数据。





# 可视化发布流程

## 持续交付

使用内置的发布管理功能创建持续发布管道，减少不确定性，降低风险。

使用可视化工具创建针对不同环境的部署任务，并连接成一体化的发布管道，监控每一阶段的部署状态，直到生产环境。

Definition: Main | Releases

Environments Artifacts Configuration Triggers General History

Save Release

+ Add environments

**DEV** \*\*\*  
6 / 6 tasks enabled  
2 |

**QA** \*\*\*  
2 / 2 tasks enabled  
2 |

**PROD** \*\*\*  
2 / 2 tasks enabled  
0 |

+ Add tasks

- Azure PowerShell  
Azure PowerShell script: \$(System.DefaultW
- Azure Web App Deployment  
Deploy Website to Azure
- Azure SQL Database Deployment  
Deploy Azure SQL DACPAC: Dacpac
- Docker  
Docker Deployment: namespace/repo\_name
- Visual Studio Test  
Run Tests
- PowerShell  
Powershell: Post Release Steps

**Deploy Website to Azure**

Azure Subscription VSU  
Web App Name DevToolsMa  
Web App Location Central US  
Slot  
Web Deploy Package \$(System.Def  
Set DoNotDelete flag  
Additional Arguments -connectionS

**Control Options**

Enabled ☒  
Continue on error ☐  
Always run ☐

More Information

ADD TASKS

- All
- Build
- Utility
- Test
- Package
- Deploy

- Azure Cloud Service Deployment  
Deploy an Azure Cloud Service Add
- Azure File Copy  
Copy files to Azure blob or VM(s) Add
- Azure PowerShell  
Run a PowerShell script within an Azure environment Add
- Azure Resource Group Deployment  
Deploy, start, stop, delete Azure Resource Groups Add
- Azure SQL Database Deployment  
Deploy Azure SQL DB using DACPAC Add
- Azure Web App Deployment  
Publish a Visual Studio Web project to a Microsoft Azure Web App using Web Deploy Add
- Chef  
Deploy to Chef environments by editing Add

**CONFIGURE - 'DEV' ENVIRONM**

Approvals Queue Va

**Approvers**

Select the users who can app

Pre-deployment approver

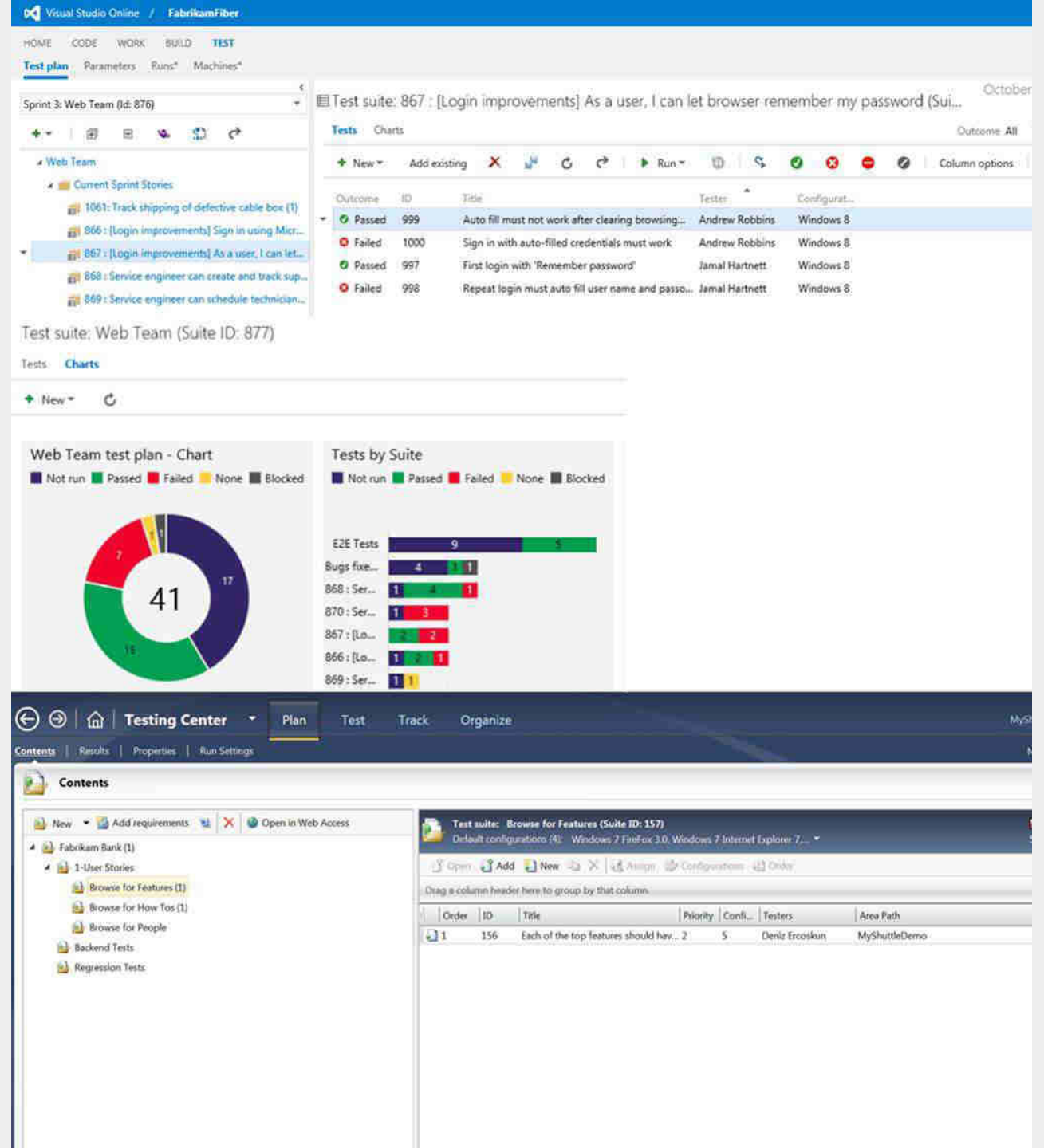
# 测试管理

## 提升质量

提供内置的测试管理工具，让测试团队与开发团队共享同一份需求列表。

基于浏览器的测试计划，测试用例，测试执行工具，让测试人员更便捷的管理和执行测试。

提供微软测试管理器，为测试人员提供专业级测试工具，完成手工/自动化测试用例的编写，录制和执行。



# 发布管理

## 加速应用交付

更便捷的管理多个不同环境的部署自动化，测试自动化和版本升级过程。

提供可视化的部署自动化脚本编写和执行环境，确保自动化脚本的可靠性和标准化，并可以集成自动化功能测试。

The screenshot displays the VSTS interface for the 'FabrikamDev' build and release. At the top, the build status is 'Build Succeeded' for 'Build 20160127.1', which ran for 95 seconds and completed 18.9 hours ago. The 'Summary' tab is active, showing build details such as the definition 'FabrikamDev (edit)', source branch 'refs/heads/master', and source version 'Commit 386238'. It also lists the requested by 'C Reinhart' and the build timeline (Queued, Started, Finished). To the right, the 'Code Coverage' section indicates no data is available, and the 'Associated work items' section shows no items found. The 'Deployments' section shows a successful deployment to 'Dev' with 'Release-4'.

Below the build details, the 'Visual Studio Team Services / Fabrikam' navigation bar is visible, with tabs for HOME, CODE, WORK, BUILD, TEST, and RELEASE\*. The 'RELEASE\*' tab is selected, showing the 'FabrikamDev' release definition. The 'Releases Overview' section displays a list of releases for 'Dev', 'QA', and 'Prod' environments. The 'Dev' environment shows a successful deployment of 'Release-4' 17 hours ago. The 'QA' environment also shows a successful deployment of 'Release-4' 17 hours ago. The 'Prod' environment shows 'No deployments yet'.

# HockeyApp

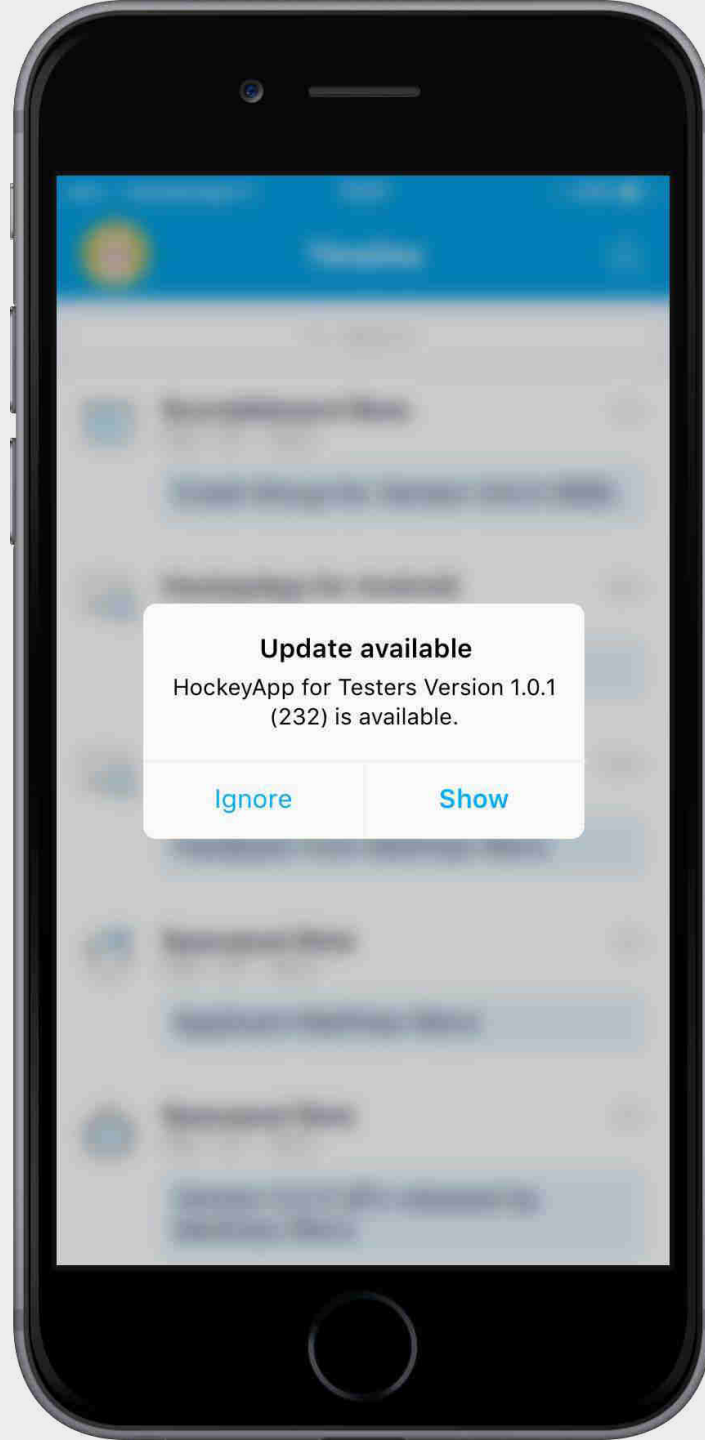
## 移动应用测试平台

### 企业应用商店支持

将移动应用测试版上传至HockeyApp，测试人员即可安装应用至自己的设备并进行测试。

### 简化的部署方式

HockeyApp桌面版自动跟踪应用的所有版本的状态，让发布应用变得无比简单。



# HockeyApp

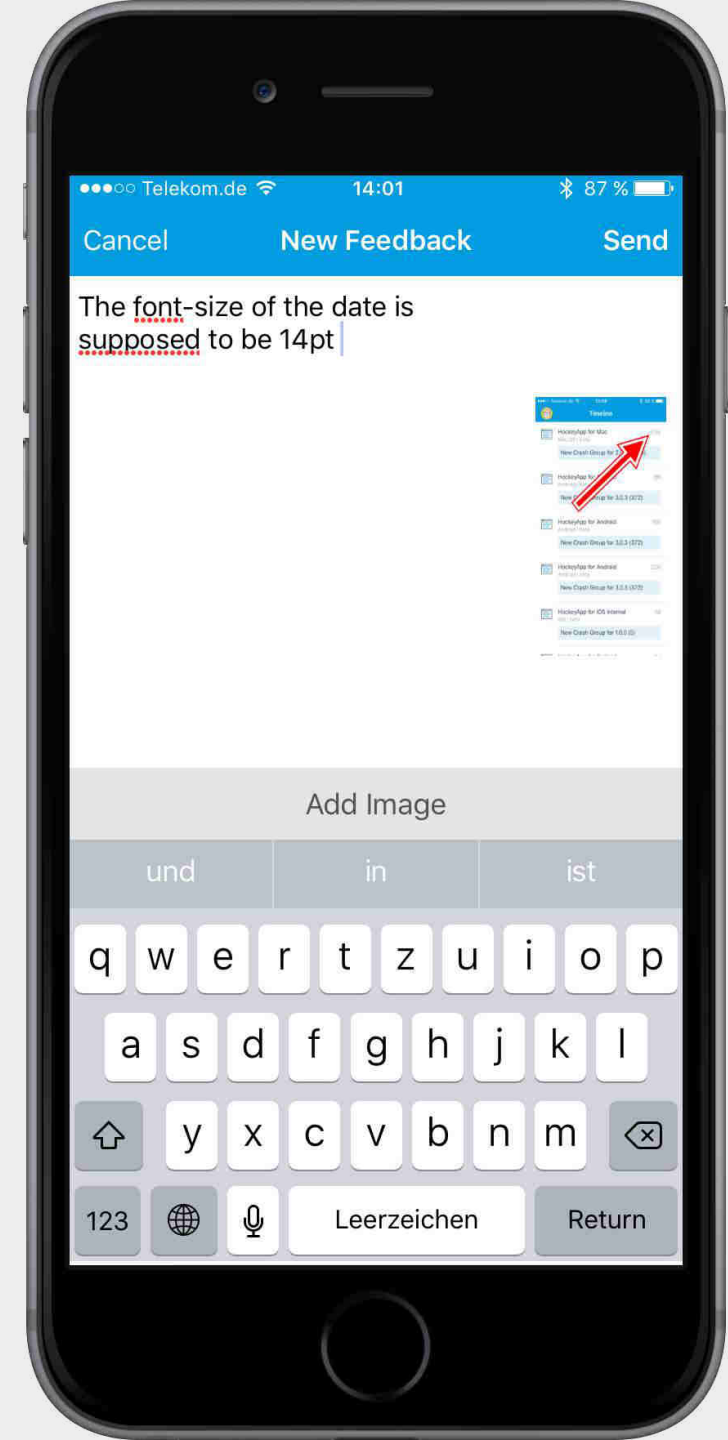
## 反馈管理

### 直接与用户对话

像聊天一样提交反馈，开发团队直接在聊天界面上进行回复，更新修复状态。让用户获得应用内的反馈体验。

### 所有开发阶段的反馈

支持在所有应用版本上提交反馈，包括开发版和预发行版。用户可以直接在应用内提交反馈并获取修复更新。





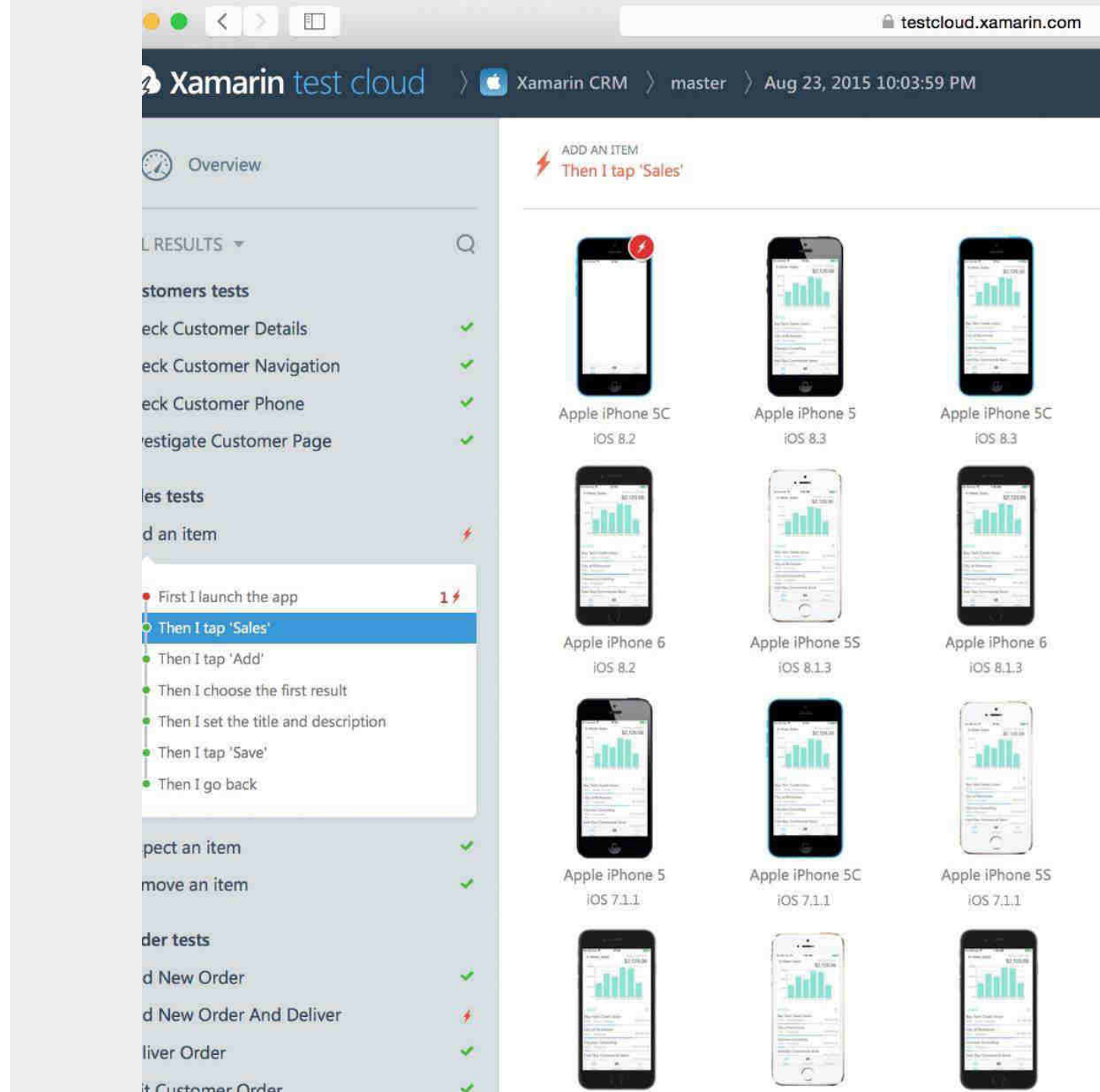
# Xamarin Test Cloud

## 超过2,000台测试设备

提供超过2000台不同型号的移动设备供测试。用户的每一步操作都可以被自动化并提供截屏，记录性能数据和内存使用情况。

## 真机设备，真实质量

- 真机测试提供更好的质量反馈
- 超过2000台设备上运行测试
- 支持使用C#,Ruby或者Cucumber编写自动化测试脚本
- 可与任何持续集成工具集成



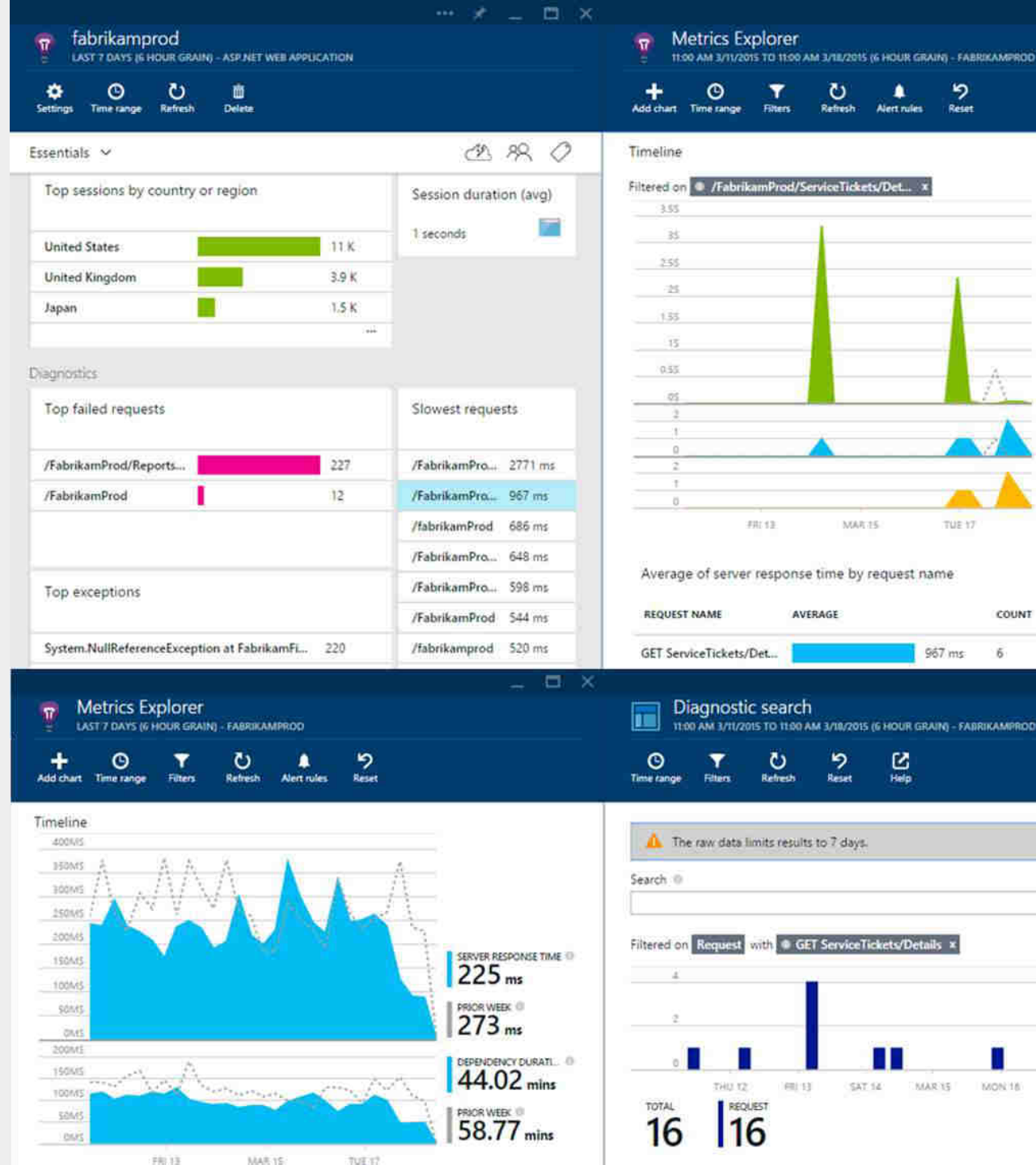
# Application Insights

## 用户行为和应用性能跟踪

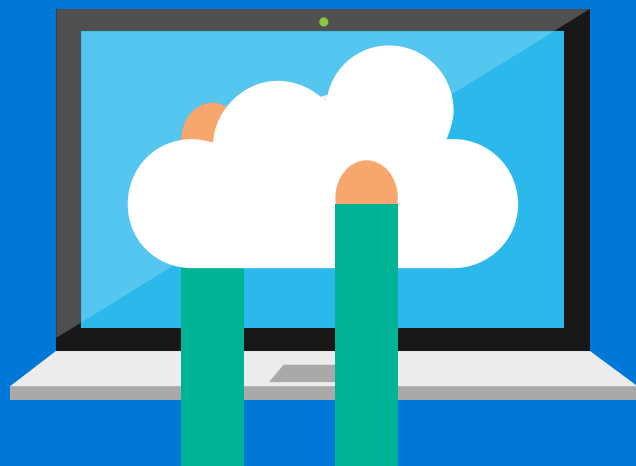
全方位360度的监控

包括可用性，性能和行为数据

为持续改进提供数据支撑



# Agenda



- Visual Studio & DevOps 概述
- DevOps反馈环
- 微软 DevOps 解决方案
- Visual Studio相关功能和优势



# 微软DevOps解决方案

不同规模，端到端，跨平台，企业级，全生命周期管理平台

缩短反馈周期，提升价值交付速度



改进质量和可用性



优化资源利用率，消除浪费



数字化时代的移动应用交付



# Git 支持

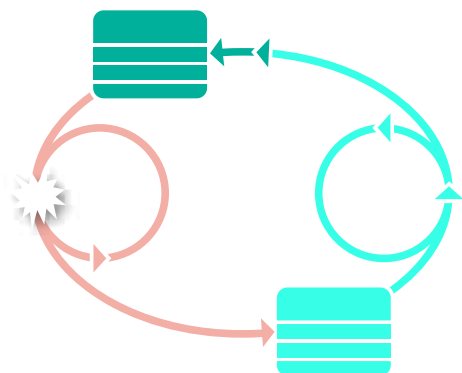
设置

开发

协作

提交注释

冲突解决



fix message

commit 7ff1c324472d814a3c96cc07628e8488 | parent a76430

Team Explorer - Commits

Commits | FabrikamGit

Accept Merge

1 Conflicts (0 Remaining)

Source: ConsoleApplication1\Consol... Target: ConsoleApplication1\Consol...

ConsoleApp... Main(string... ConsoleApp... Main(string...

ain(string[]) args) ain(string[]) args)

☐ riteLine("Greetings, planet."); ☒ riteLine("Hello world!");

100 % 100 %

Result: ConsoleApplication1\ConsoleApplication1\Program.cs

ConsoleApplication1.Program Main(string[] args)

```
ic void Main(string[] args)
{
    Console.WriteLine("Hello world!");
}
```

Team Ex.. - Resolve Conflicts

Resolve Conflicts | FabrikamGit

Commit Merge Undo Merge

Conflicts (1)

Name	Path
C# Program.cs [both mod...	Conso...

Merge

Conflicting content changes have been made. Compare Files

Edited on Remote | Diff | Take Re... Edited on Local | Diff | Keep Local

Resolved

There are no resolved conflicts

Output

14 14 }  
15 15 }  
16 16 }

Jamal Hartnett (Fabrikam) - save (enter) - cancel (esc)

# CodeLens

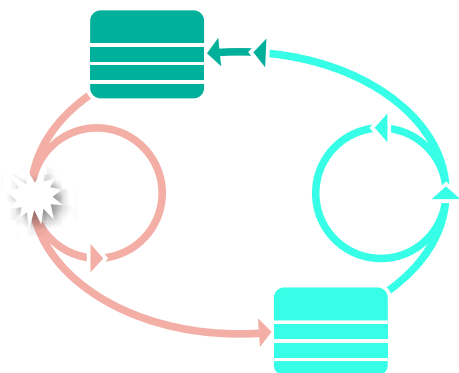
搜索引用

单元测试

代码历史

工作项

用户在线情况



ID	Changeset	Author	Date
63	Fixed the bug in parsing logic		5/10/2013
62	ParseTerm should call Pop in one step		5/10/2013
17	add consolecalculator and unittest project		4/26/2013

```
void ParseTerm()
{
    ParseFactor();
    Token tok;
    do
    {
        tok = PeekToken();
        if (tok.Type == TokenType.EndOfStream)
            break;
        if (tok.Type != TokenType.Operator)
            throw new CalculatorException("Operator expected.");
        var op = new Operator(tok.Value);
    }
}
```

Diff - Parser.cs Changeset 63

ConsoleCalculator.Parser	_scanner	ConsoleCalculator.Parser	ParseTerm()
193		193	
194	var first = Pop();	194	var first = Pop();
195	var second = Pop();	195	var second = Pop();
196	// we should call Pop() here?	196	Push(second % first);
197	Push(Pop() % first);	197	break;
198	break;	198	
199		199	
200		200	
201		201	

# 代码地图

## 可视化你的代码

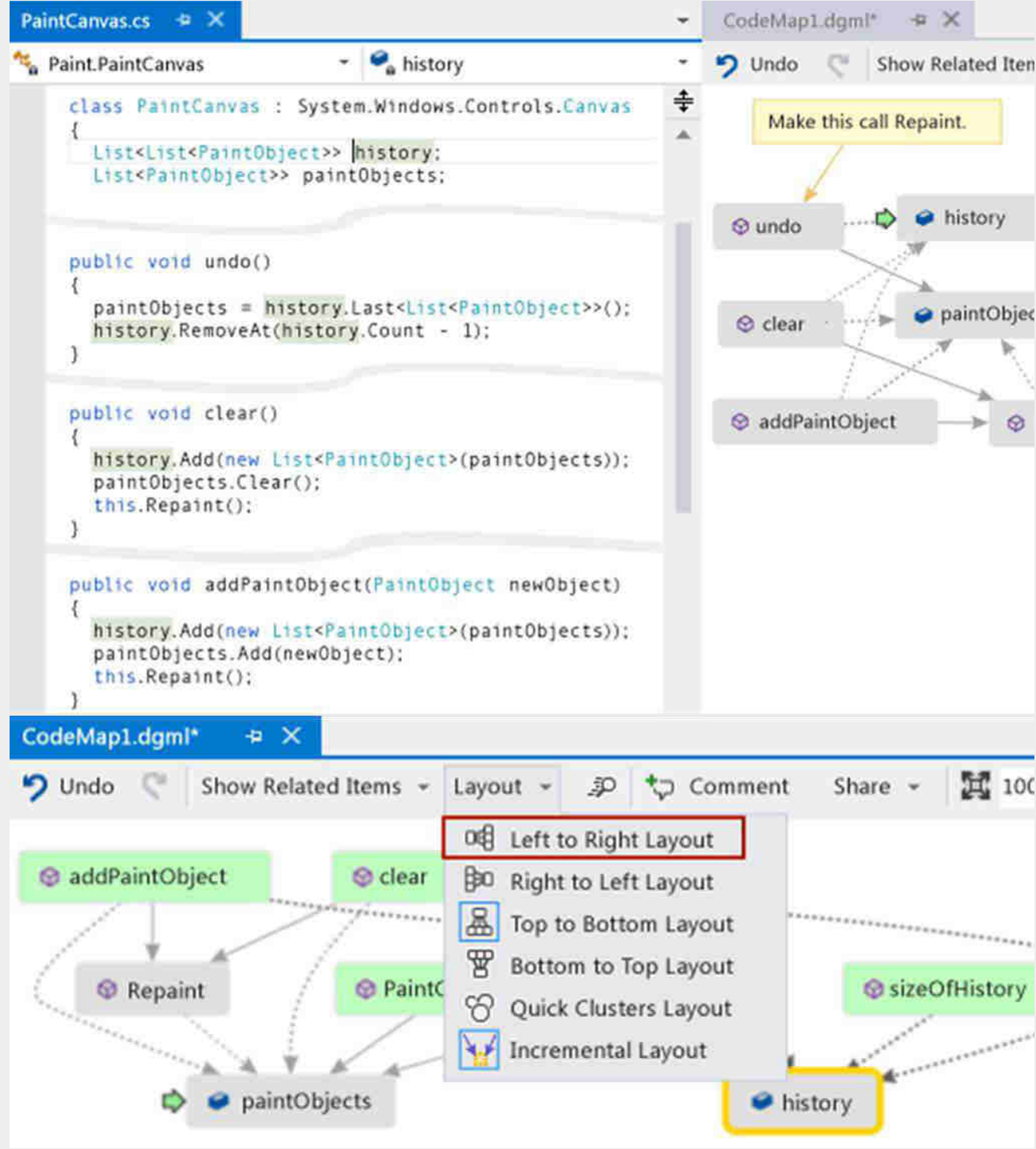
关联关系可视化

与代码并排显示

专注于业务逻辑，而不用逐行阅读复杂的代码

提升阅读已有代码的速度和理解能力

减少由于理解错误而造成了误修改。



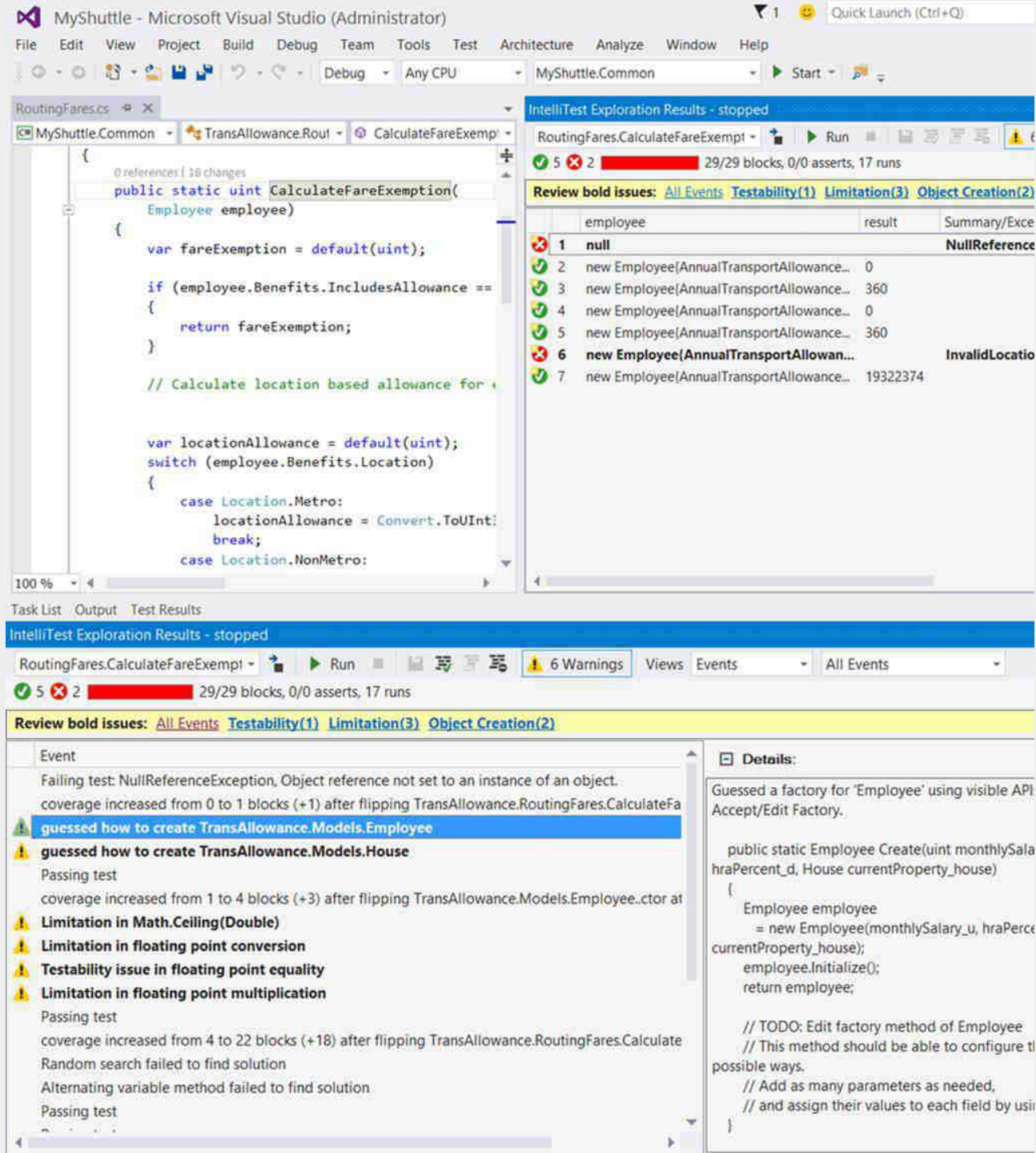


# IntelliTest

## 更容易的单元测试

自动生成单元测试，动态监测代码逻辑路径，并针对各种可能情况生成边界测试。

更好的应对已有代码，让你有更多的时间专注于新功能的开发。



# LEANSOFT 简介

Nobody knows DevOps better than us!

英捷创软科技(北京)有限公司(LEANSOFT)是一家专注于软件工程，敏捷开发和DevOps领域产品开发和服务的解决方案提供商。

公司由15年 软件研发经验,资深ALM/DevOps专家创建并任公司首席架构师，至今已经为超过100家不同类型的客户提供过ALM解决方案的咨询和落地服务。

联系我们

📞 137 1126 4760

✉ info@lean-soft.cn

我们致力于为广大开发人员和企业提供最优化的DevOps/敏捷咨询服务和软件工程解决方案。

请扫描右侧二维码加入中国最大的DevOps社区【DevOpsHub】

社区网站 <https://devopshub.cn>

