TFS过程管理规范

技术与质量管理部—程宝君

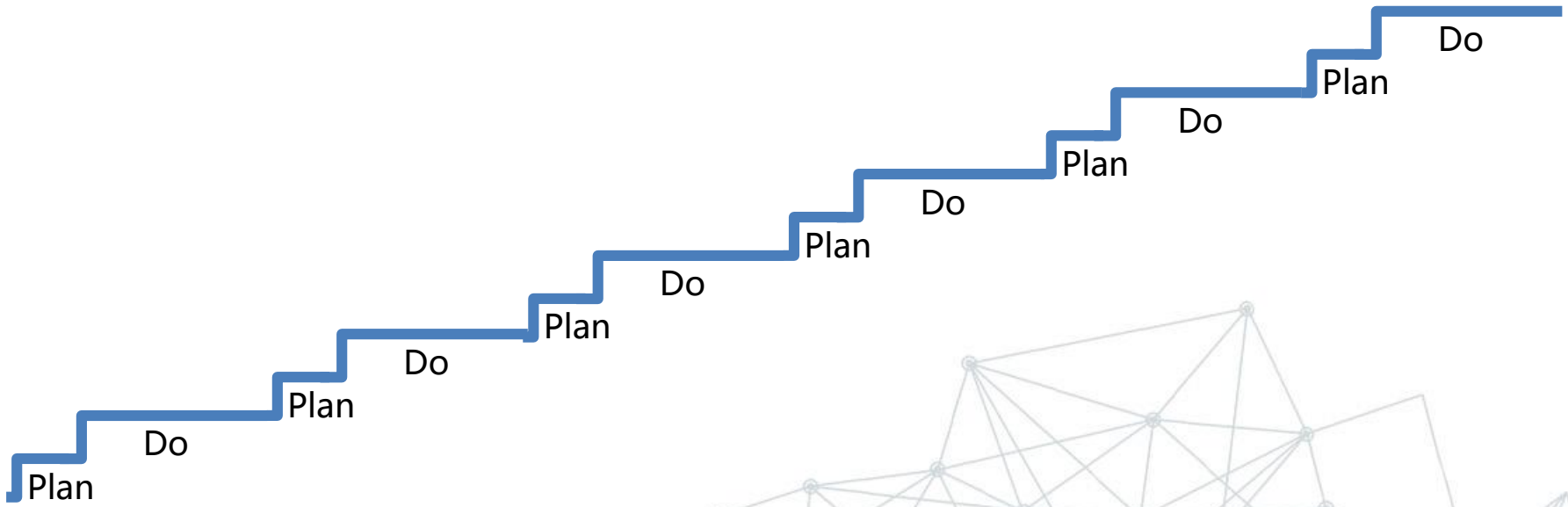
目录 CONTENT

1.Scrum-基本概念

- **Scrum**：是一种迭代式增量软件开发过程，通常用于敏捷软件开发。敏捷开发是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。

传统计划方式：

Scrum计划方式：



1.Scrum-主要角色

- PO（Product Owner）
- ScrumMaster
- Team

1.Scrum-主要角色

- PO（Product Owner）：
  - 主要工作：
    - 负责管理产品待办事项列表（Backlog）
    - 优先级排序
    - 与团队一起估算工作量
    - 对项目负责
  - 建议：
    - 客户项目：由客户代表担任
    - 内部项目：有业务经理担任
    - 不能有ScrumMaster担任

1.Scrum-主要角色

- **ScrumMaster：**
  - **主要工作：**
    - 保证团队遵守Scrum价值、实践和规范
    - 帮助团队采用Scrum模式进行项目流程组织
    - 保护团队不受外界干扰
    - 保证各个角色良好协作
  - **建议：**
    - 不能同时担任PO

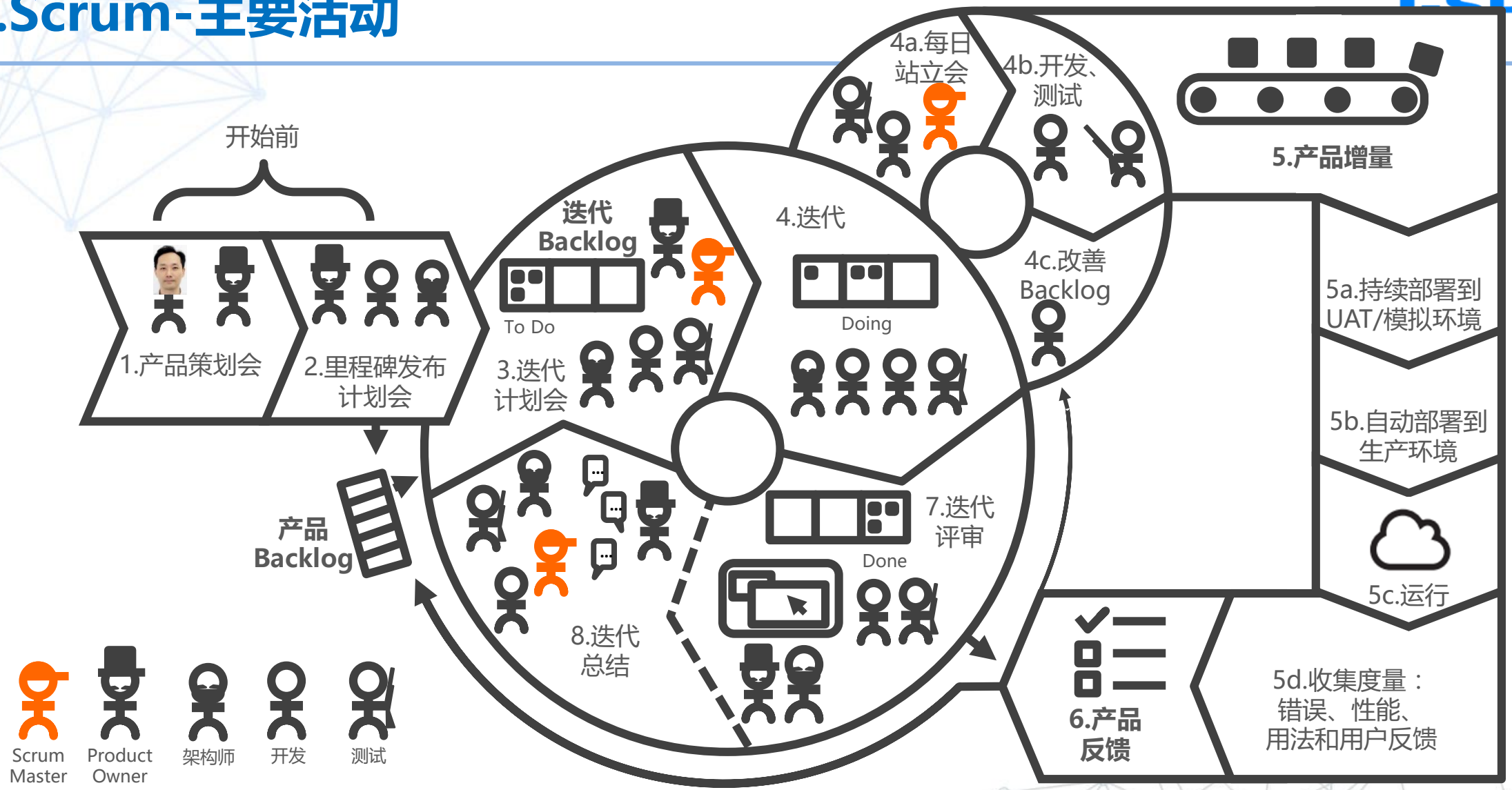
1.Scrum-主要角色

- **团队：**
  - **最佳团队大小：**5-9人
  - **成员：**程序员、测试人员、设计师、数据库管理员、架构师
  - 全职参与
  - 成员更替在迭代之间进行

1.Scrum-主要活动

- **发布计划会**
  - 进行产品规划（必须内容），明确可交付物（产品特性、发布目标、产品待办事项表）
- **迭代计划会**
  - 进行迭代规划，PO向团队介绍产品待办事项表，迭代目标和**迭代合约**（团队成员及分工，对迭代目标的承诺，迭代长度，待办事项工作量估算），产品待办事项表拆成迭代待办事项表
- **迭代**
  - 团队实现迭代目标的时间区间，1-4周，时间长度决定何时结束迭代
- **每日站立会**
  - 站立进行，固定时间，固定地点，3个问题（昨天做了什么，今天计划做什么，障碍），不解决问题
- **迭代评审会**
  - 团队展示完成的功能并收集反馈，邀请所有相关人，包括客户
- **迭代回顾会**
  - 哪些好，哪些不好，哪些可改进，团队成员参加

1.Scrum-主要活动



8、**迭代总结**：检查和调整。Scrum团队讨论什么进展顺利，缺少什么，改善什么

目录
CONTENT

2.整体介绍-基本概念

- **TFS**：TFS (Team Foundation Server)是 Microsoft 应用程序生命周期管理 (ALM) 解决方案的核心协作平台。 不论在本地还是在云中，TFS 均可支持灵活的开发实践、多个 IDE 和平台，并为您提供有效管理整个 IT 生命周期的软件开发项目所需的工具。
- **项目集**：项目的集合，一个项目集可以有多个项目，项目集下的所有项目是共享TFS组的。
- **项目**：是项目集下的一个项目，在一个项目里可以通过各工作项的协作完成研发过程的各项管理工作。

2.整体介绍-基本概念


控制面板 > GSP > GSP6.1
Jon Cheng(程宝君)
控制面板 > GSP > GSP6.1
概述
迭代
区域
安全性
警报
版本控制
服务挂钩
服务
测试
迭代
迭代
选择要用于迭代规划(冲刺(sprint)规划)的迭代。 所选迭代将在你的积压工作 (backlog) 视图中显示为可用于规划的迭代。
新建
新建子级
迭代
开始日期
结束日期
GSP6.1 2014/12/26 2018/12/31 此团队的积压工作 (backlog) 迭代
M0（计划） 2015/5/11 2015/11/29
M1 2015/11/30 2016/7/10
M2 2016/10/8 2016/12/15
M3 2016/12/12 2017/7/31
M4 2017/8/28 2018/4/30
M5 2018/3/19 2018/7/8
Sprint1 2018/3/19 2018/4/1
Sprint2 2018/4/2 2018/4/15
Sprint3 2018/4/16 2018/4/29
、构件等
Restful框架、异常框架、国际化框架、打...
出、分布集中、MQ等
台、集成平台
）、用户权限、消息平台、调度服务、模拟...
，打造一流的云开发平台。
流、移动审批等

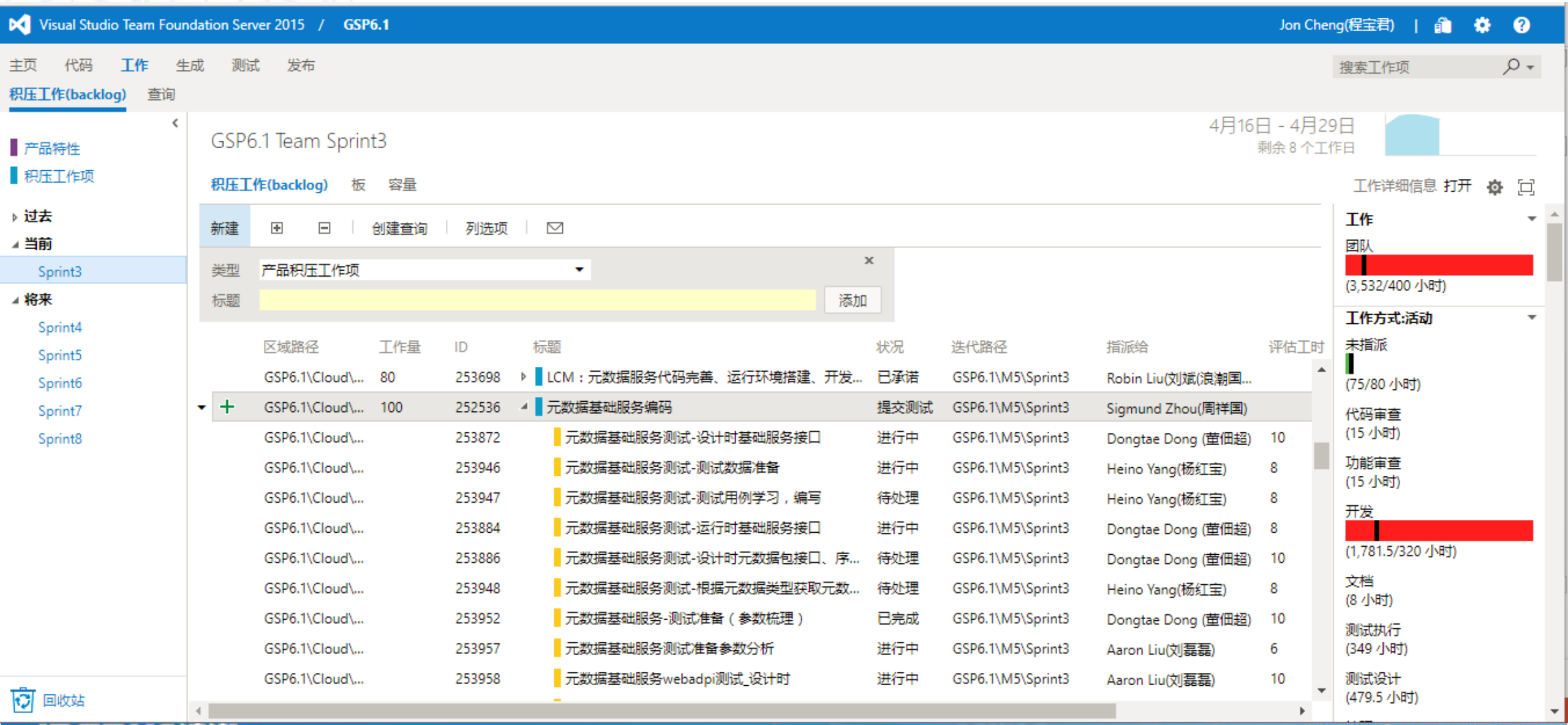
2.整体介绍-基本概念

- **工作项：**软件开发工作的业务载体，就像是表单。
  - Backlog、Task、TestCase、Bug等都是一种类型的工作项；
  - 每种工作项都可以定义自己的字段、界面布局、状态以及状态的流转规则；
  - 工作项之间可以有各种关系：父项、子项、依赖、子项、测试方等
  - 工作项的ID在整个项目集中是唯一的。
- **Scrum与TFS的关系：**
  - Scrum是一套理论及方法，TFS是一个工具或者说框架
  - Scrum可以作为一套模型定义在TFS中，TFS同样也可以支撑并应用其他模型（包括CMMI），也可以在这些模型基础上进行定制。
  - TFS中可以定义一堆项目，每个项目可以使用一个模型

2.整体介绍-主要界面

- **地址**：http://10.24.1.11:8080/tfs/
- **选择团队**：选择团队后主界面各部分都是团队相关的信息。
- **个人信息**：设置个人信息，logout。
- **管理项目**：进行各项设置的入口（团队、区域、迭代、人员、权限等）
- **主页**：通过预制和自定义的小部件查看项目或团队的总体情况。
- **代码**：是一个代码及其他文档文件的配置管理的网页版，主要可以进行查看和下载操作。

2.整体介绍-主要界面


Visual Studio Team Foundation Server 2015 / GSP6.1
Jon Cheng(程宝君)
主页 代码 工作 生成 测试 发布
搜索工作项
积压工作(backlog) 查询
产品特性
积压工作项
过去
当前
Sprint3
将来
Sprint4
Sprint5
Sprint6
Sprint7
Sprint8
回收站
GSP6.1 Team Sprint3
4月16日 - 4月29日
剩余 8 个工作日
积压工作(backlog) 板 容量
工作详细信息 打开
新建 创建查询 列选项
类型 产品积压工作项
标题 添加
区域路径 工作量 ID 标题 状况 迭代路径 指派给 评估工时
GSP6.1\Cloud\... 80 253698 LCM：元数据服务代码完善、运行环境搭建、开发... 已承诺 GSP6.1\M5\Sprint3 Robin Liu(刘斌(浪潮国...
GSP6.1\Cloud\... 100 252536 元数据基础服务编码 提交测试 GSP6.1\M5\Sprint3 Sigmund Zhou(周祥国)
GSP6.1\Cloud\... 253872 元数据基础服务测试-设计时基础服务接口 进行中 GSP6.1\M5\Sprint3 Dongtae Dong (董佃超) 10
GSP6.1\Cloud\... 253946 元数据基础服务测试-测试数据准备 进行中 GSP6.1\M5\Sprint3 Heino Yang(杨红宝) 8
GSP6.1\Cloud\... 253947 元数据基础服务测试-测试用例学习，编写 待处理 GSP6.1\M5\Sprint3 Heino Yang(杨红宝) 8
GSP6.1\Cloud\... 253884 元数据基础服务测试-运行时基础服务接口 进行中 GSP6.1\M5\Sprint3 Dongtae Dong (董佃超) 8
GSP6.1\Cloud\... 253886 元数据基础服务测试-设计时元数据包接口、序... 待处理 GSP6.1\M5\Sprint3 Dongtae Dong (董佃超) 10
GSP6.1\Cloud\... 253948 元数据基础服务测试-根据元数据类型获取元数... 待处理 GSP6.1\M5\Sprint3 Heino Yang(杨红宝) 8
GSP6.1\Cloud\... 253952 元数据基础服务-测试准备（参数梳理） 已完成 GSP6.1\M5\Sprint3 Dongtae Dong (董佃超) 10
GSP6.1\Cloud\... 253957 元数据基础服务测试准备参数分析 进行中 GSP6.1\M5\Sprint3 Aaron Liu(刘磊磊) 6
GSP6.1\Cloud\... 253958 元数据基础服务webadpi测试_设计时 进行中 GSP6.1\M5\Sprint3 Aaron Liu(刘磊磊) 10
工作
团队
(3,532/400 小时)
工作方式:活动
未指派
(75/80 小时)
代码审查
(15 小时)
功能审查
(15 小时)
开发
(1,781.5/320 小时)
文档
(8 小时)
测试执行
(349 小时)
测试设计
(479.5 小时)

2.整体介绍-主要界面

- ~~**生成**：自动构建生成的操作界面~~
- **测试**：测试管理界面，包括创建测试计划、套件、创建用例、测试配置、跟踪执行。
  - Build提交时开发自测检查项也会创建单独测试计划进行测试的跟踪
- ~~**发布**：管理所有的发布定义~~
- **搜索工作项**：是一个非常有用的功能，可以根据ID查找各种工作项，并可以进行过滤。

目录 CONTENT

3.区域路径规范

- 是管理各种工作项的重要属性，是我们统计各模块需求提交率，任务完成情况，测试过程管理，bug修复率的主要抓手。
- 各种工作项都应对区域路径，通常应该至少选到模块级，例如“\Cloud\02运行平台\10运行框架”

产品　关键应用　模块

- **权限**：只有各开发经理可以调整项目的区域路径。
- **调整**：修改区域路径后工作项中的区域也会随之改动。
- **团队**：每个团队可以指定团队拥有的区域。

目录 CONTENT

GSP+ inspur

4.迭代路径相关规范

- **迭代的周期**：迭代，只是一个时间概念。每个迭代应该是固定的周期，迭代不会因为任务没有完成而延长（就像4月份不会因为不开心就加2天）
- 按照敏捷的思想每个迭代末都应该对外发布变更；只发布测试通过的。
- 已经过去的迭代所有工作项都应该是已完成状态，没完成的工作项可以移到后续迭代。
- **计划迭代和实际迭代**：系统中实际起作用的是“实际迭代”

目录 CONTENT

GSP+ inspur

目录 CONTENT

GSP+ inspur

5. Backlog规范

① 基本概念

② 划分原则

③ Backlog主要字段

④ 状态流转规则

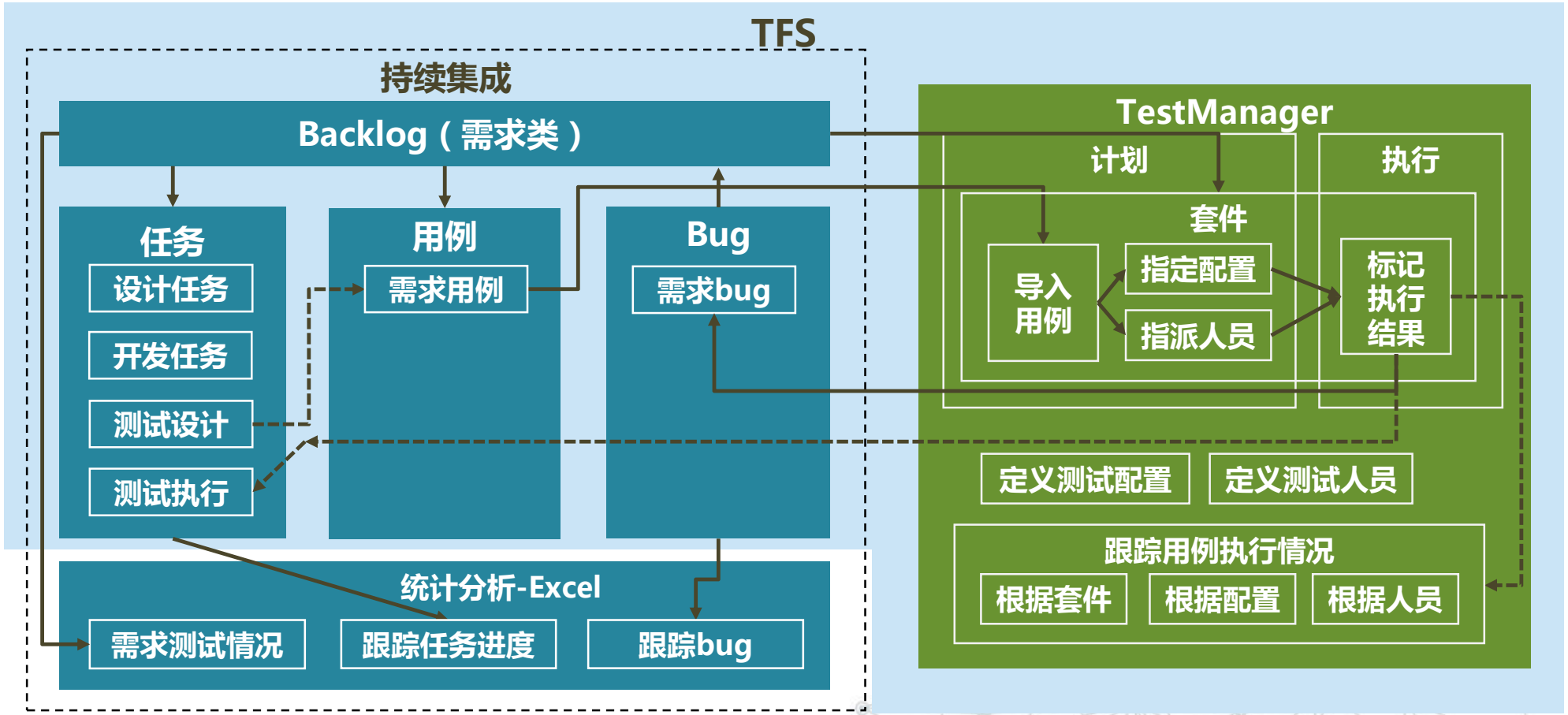
⑤ 其他

5.Backlog规范-基本概念

- 直译为积压未办之事。定义为产品待办列表，英文是Product Backlog，是指产品待办事项的集合，源自于Scrum方法。产品主管（Product Owner）收集来自于各方的需要、期望、诉求等等到产品待办列表中，给定优先级；当迭代计划会议上，团队从产品待办列表中挑选其中事项组成迭代待办列表。常见的待办事项表达形式是用户故事。
- 产品待办列表由所有的功能特性，包括业务功能，非业务功能（技术、架构和工程实践相关），提升点以及缺陷的修复等组成。这些内容也是将来产品版本发布的主要内容。

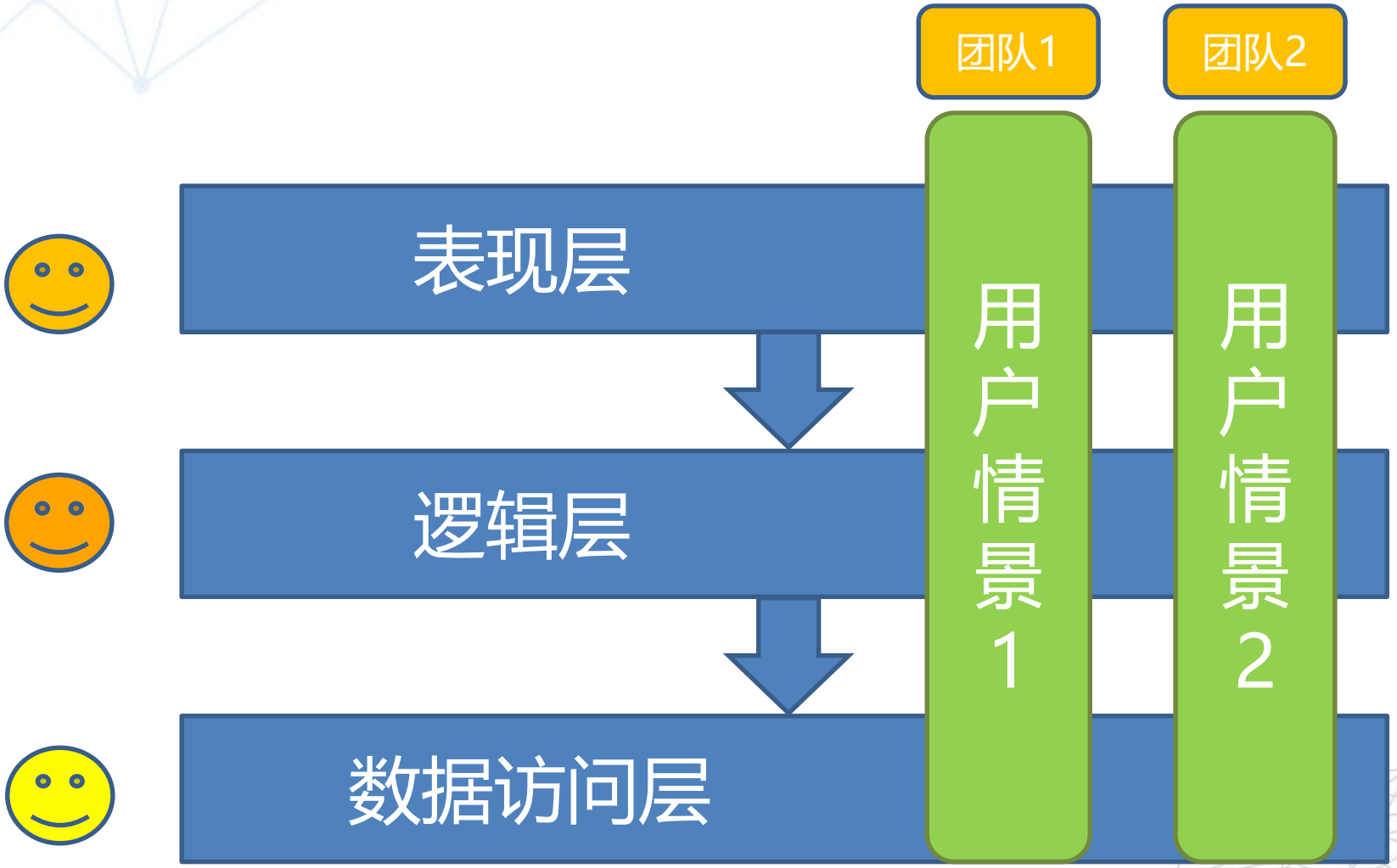
5.Backlog规范-基本概念

- 需求类Backlog是其他所有工作项的来源，是一切软件工作的源头。
- TFS会自动创建关系，也可以自己增加关系



5.Backlog规范-划分规则

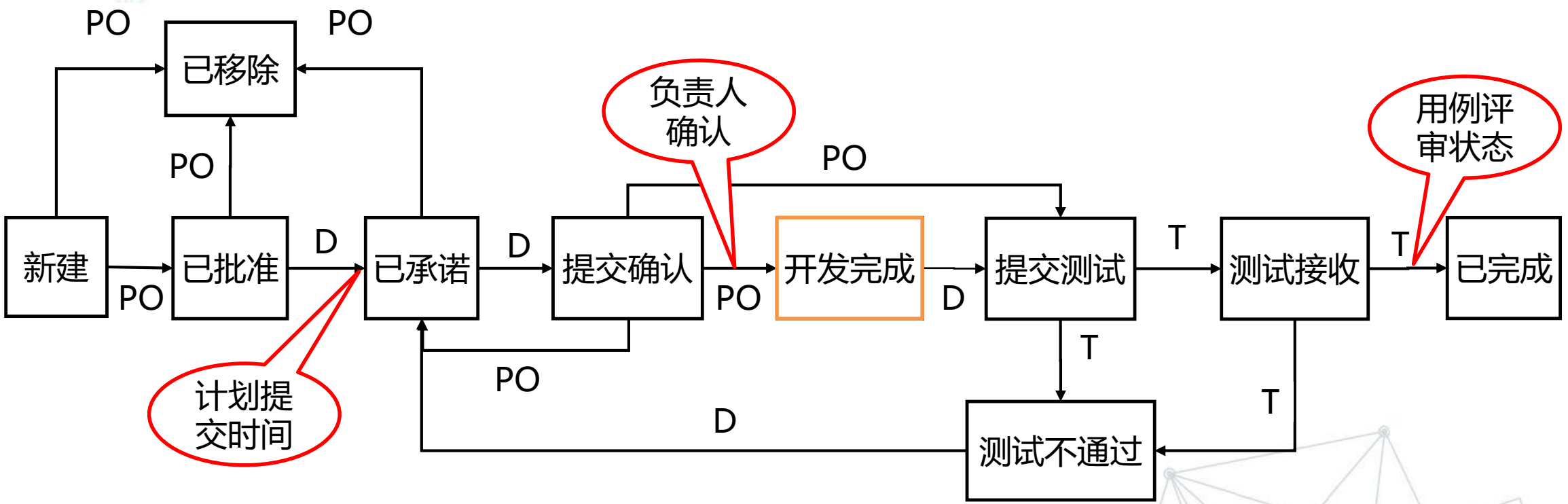
- 需求类Backlog划分应该按照纵向分割的原则：



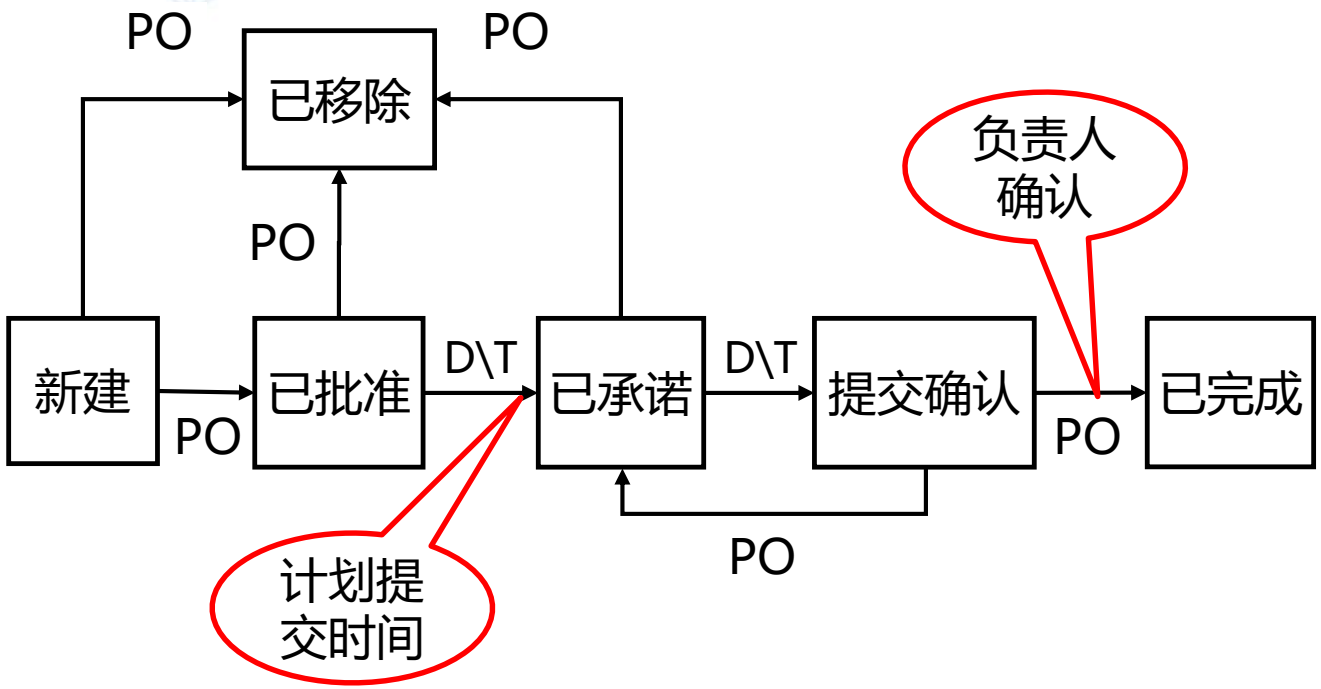
5.Backlog规范-主要字段

字段	说明	规范
标题	简述	• 不只是功能名称，且必须唯一 • 应从需求角度进行描述，不是任务，不能出现“设计”“实现”“开发”字眼。
迭代	开展工作的迭代	• 上一迭代未完成的BackLog只有在迭代会议时可转入下个迭代
状态	状态	• 状态转换规则见“2.2.1”
区域	所属功能模块	• 必须对应一个模块，区域尽量划分细致，要求至少要到模块级别，越细越好
类别	主要有：需求、待办事项等	• “需求”和非需求类状态流转规则不同
说明	详细描述，详细的功能点	• 详细描述，详细的功能点
用户故事	描述用户故事相关的一些场景、业务规则、约束、影响等内容	• 描述需求建议使用用户故事的方式。格式为：作为一位...（角色），希望（通过系统）...（做什么），以...（达到什么目的）
验收条件	条目化地描述大致功能范围以及做到什么程度	• 验收条件由开发负责人和测试人员共同确定 • 通常是冒烟测试点，是测试接收的标准
测试人员	此Backlog的主要测试人员	• 选择测试负责人
用例评审状态	用例是否评审评审通过	• 只能由开发人员修改 • 测试完成前，用例评审状态必须是通过
计划提交时间	计划提交的时间	• 由开发人员在状态更新成“已承诺”的时候填写 • 计划提交时间尽量均匀分布，避免集中提交

5.Backlog规范-状态流转规则（需求类）



5.Backlog规范-状态流转规则（非需求类）



5.Backlog规范-其他

- 及时更新Backlog状态
- 需求的变更（修改、拆解、删除）必须经过PO的确认
- 如果需求只实现了一部分就像提交测试，拆出一条backlog描述实现的部分并提交测试
- Backlog按照有层次的划分，拆成不同backlog。例如：基础功能，完整功能
- 应包含ci、cd的“待办事项”类backlog
- 应包含培训的“待办事项”类backlog
- “方案”“技术验证”等不需要测试的backlog应该是“待办事项”类。
- 需求类Backlog应包含设计、开发、测试设计、测试执行、文档工作量的task
- 未提交的backlog的计划提交时间不能是今天之前的时间
- 历史迭代的backlog必须是已完成

目录 CONTENT

6.任务规范

- **迭代计划会：**
  - 定义团队成员的容量（包括休息日）和主要活动。
  - 每个人计划工时之和=容量*天数
  - 将每个Backlog进行细化，创建可估算的任务并确定负责人
    - 定义评估工时，剩余工时（此时评估工时=剩余工时）
    - 任务的划分以不超过1天的工作量为宜
  - Task可分为：设计、编码、测试设计、测试实现、测试执行
- **迭代中：**
  - 每天9点前通过电子白板更新前一天的实际工时，剩余工时，不要修改评估工时
  - 剩余工时不必须=评估工时-实际工时

6.任务规范

迭代总结会：

- 状态确认：留在本迭代的应该都是“已完成”的。
- 转移迭代：
  - 未开始的直接转移
  - 已开始完成一部分的：把原任务打成完成，实际工时就是已投入的工时，剩余工时为0，留在原迭代
- 新建一个以原任务未完成工时为计划工时和剩余工时的任务，放在打算完成任务的迭代

其他：

- Task应该创建在Backlog之下
- 已完成task实际工时不能为0
- 未完成task剩余工时应大于0
- 所有task必须设置工作量（计划工时、实际工时、剩余工时）
- 历史迭代task必须是已完成状态

目录 CONTENT

目录 CONTENT

7.测试用例规范-主要字段

字段	说明	填写要求
标题	测试用例标题	描述测试场景和测试点
迭代	测试设计对应的迭代	选择相应迭代
区域	所属功能模块	选择相应模块
用例类别	用例的类别	功能（默认）、性能、界面、易用性、文档
用例来源	用例设计思路来源	需求（默认）、设计、业务、原有场景继承、原有TD生成、代码覆盖率分析、客户反馈问题
用例状态	目前用例所处的状态，这个状态是随时变化	概要设计（默认）、评审通过、详细设计、自动化实现
用例级别	用例的优先级别	选择相应级别
适用版本	此测试场景适用的产品版本	Cloud
测试对象	此测试用例测试的需求，主要用于公共测试用例导入时的过滤项	
步骤	执行此测试场景的执行步骤	
关联的自动化	此测试场景对应的自动化测试脚本	如果必须进行人工参与的就不是单元测试，用例也必须标记成不需要自动化。

7.测试用例规范-其他

- 测试用例分为概要设计和详细设计（具体要求见用例规范）
  - 概要设计通过思维导图工具设计和评审，相关开发，开发经理必须回复用例评审邮件。评审通过后作为条目化TestCase导入TFS，包含了详细设计必填的字段才算完成了详细设计
  - 概要设计评审后再做详细设计。
- 测试用例使用TestManager进行管理
- 所有需求类BackLog必须对应测试用例
- 定义各类公共测试用例与私有用例形成完整的测试用例。
- TestCase不只是测试各种操作流程、数据覆盖的带有期望值的描述。
- 测试用例必须利用测试框架进行设计

目录 CONTENT

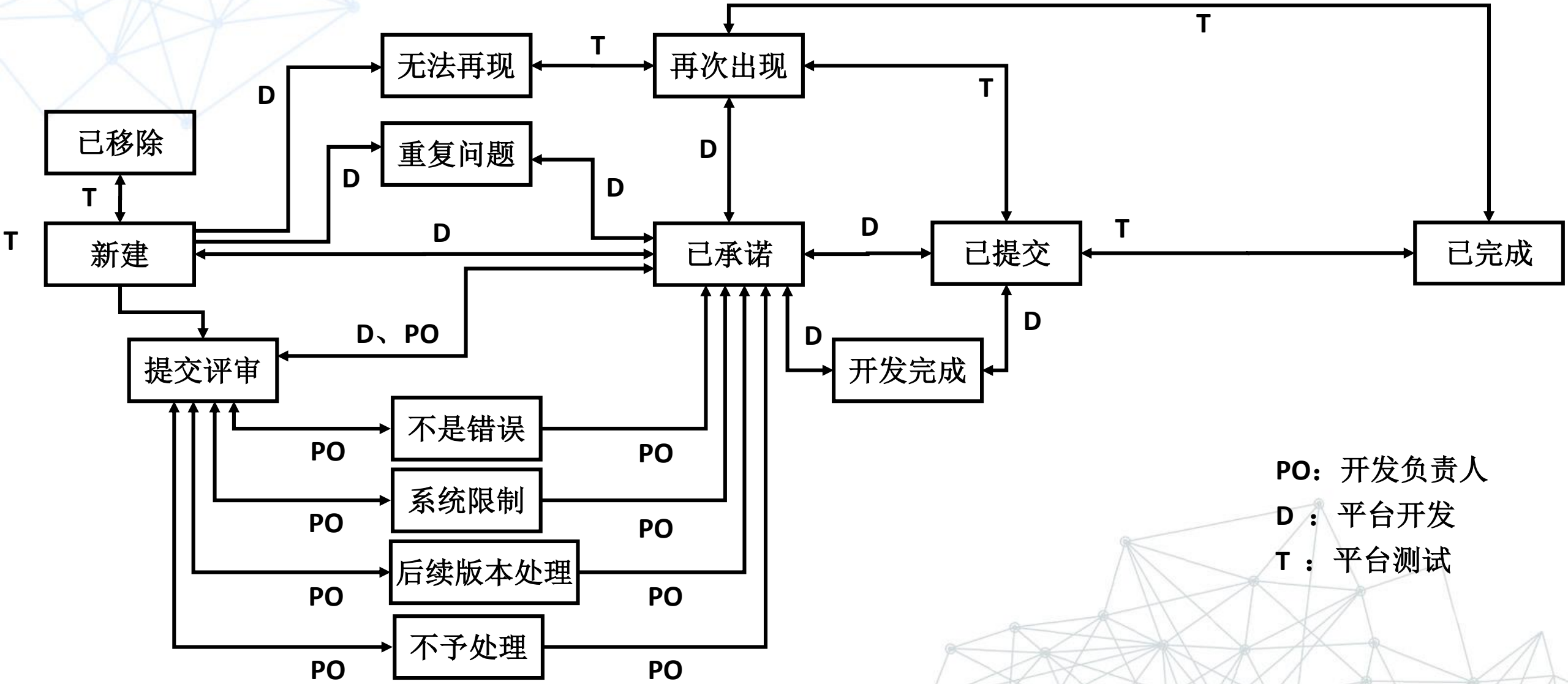
GSP+ inspur

目录 CONTENT

8.Bug规范-主要字段

字段名称	字段说明
创建人	登记Bug的人
创建日期	登记Bug时的系统日期
状态	新建（默认）、已承诺、已提交、已完成、无法再现、系统限制、再次出现、重复问题、不是错误、后续版本处理、不予处理、提交评审
标题	缺陷简单描述
迭代	修复此Bug的迭代
对应的工作产品	需求、设计、测试用例、程序代码、可执行程序（默认）、系统说明书
指派给	负责实现此BackLog的主要开发人员。
缺陷发现阶段	需求分析、设计、单元集成（默认）、系统集成、试销阶段、发版测试、维护阶段
缺陷发现方式	需求评审、设计评审、代码走查、代码结对审查、代码会议评审、自动构建、测试（默认）、演示评审、客户反馈
是否修改引起	否（默认）、是
问题类别	程序错误（默认）、性能、界面、易用性、文档、需求、设计、代码规范、操作问题、环境配置问题、数据问题、测试规范、测试方法
区域	发现问题的区域
严重级别	按照公司严重程度分级标准:1、2、3（默认）、4
状态意见	测试人员对于状态的意见
缺陷来源	需求、设计、编码、集成
解决方法说明	开发人员修改此问题的说明
补丁号	修复此问题所在的补丁号 or Docker??
重现步骤	Bug重现以及与期望值差别的详细说明
系统	发生bug的详细信息
附件	存放与bug相关的附件：截图等

8.Bug规范-状态流转规则-内部登记

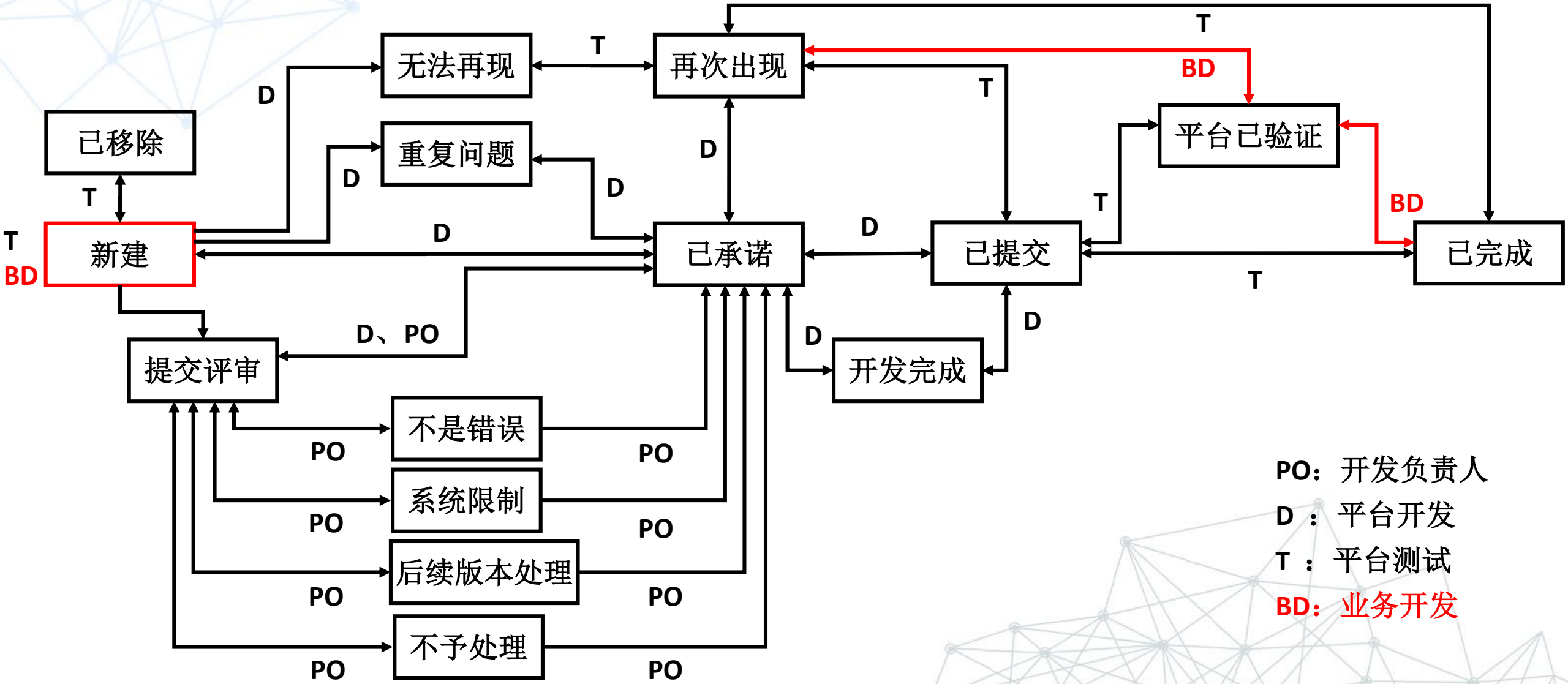


PO：开发负责人

D ：平台开发

T ：平台测试

8.Bug规范-状态流转规则-外部反馈



8.Bug规范-其他

- 在Bug下可以创建相关的设计、开发、测试任务
- 未完成的需要转移，拖到计划修复的迭代
  - 已提交的由测试人员转移
  - 未提交的由开发人员转移

9.其他

- 利用查询进行工作项的跟踪
- 利用excel进行统计分析
- 测试计划

扫描关注
浪潮集团官方微信

谢谢您的仔细聆听

Thank you for watching carefully

关注浪潮，做你企业成功路上的伴侣!