大连理工大学
Dalian University Of Technology

系统分析与设计概述

大连理工大学　　软件学院

马瑞新

目录

1.1 系统的概念与特性

系统是一组为实现某些结果相互联系、相互作用的部件的集合体。

(一)系统具有整体性

整体性是系统的最基本特性，也是观察和分析系统最基本的思想和方法。

例一：“木桶理论”认为，木桶的盛水量取决于最短的那一块木板的长度。

解读：把木桶看做一个整体，每块木板就是部分，木桶理论反应了系统部分与整体的关系，部分影响整体。

例二：一个和尚挑水吃，两个和尚抬水吃，三个和尚没水吃。

解读：三个人共同构成了一个系统，每个人都是其组成要素之一，如果三人之间的协作关系处理的不好，反而办不好事。

系统是一个整体，它不是各个部分的简单相加，系统的整体功能是各部分在孤立状态下所没有的。一般来说，系统的整体功能大于组成系统的各部分的功能之和。

例一：有研究表明，人的双眼视觉功能大大超过两只单眼视觉功能简单相加的总和。

例二：一支强悍的球队，队员之间配合的很默契。

例三：“三个臭皮匠顶个诸葛亮”
三个皮匠构成一个整体，
整体功能大于单个个体功能之和。

（二）系统具有目的性

任何系统都具有某种目的，都要实现一定的功能，这也是区别不同系统的标志。

例一：人们将发条、游丝、齿轮、表壳等零件组装成钟表，其目的是？

满足人们对计时功能的需要。

例二：经营管理系统通过优化和配置企业的人力资源和物力资源，其目的是？

实现企业利润的最大化，成本和能源使用的最小化。

系统的目的一般通过更具体的目标来实现，系统的多个目标有时不完全一致，甚至互相矛盾，这就需要协调，寻求平衡或折中的办法，从而收到整体最佳效果。

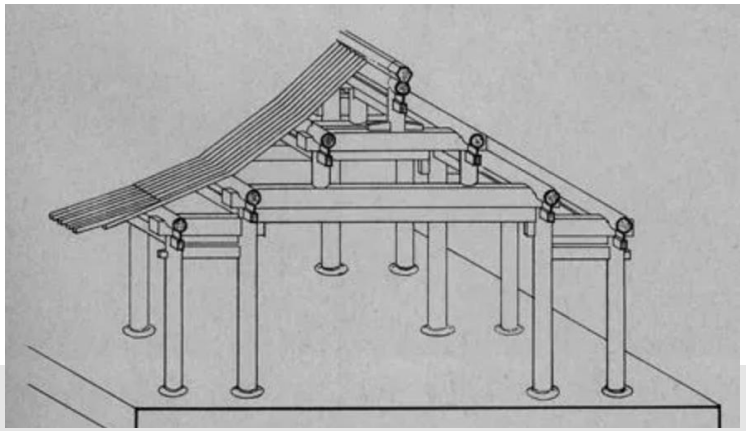
例如：汽车车速过快或过慢都会格外耗油，两者之间要有个最适当的速度，即所谓巡航速度（又叫经济速度）最省油。

（三）系统具有相关性

系统内各部分之间存在着相互依赖和相互制约的特定关系，某一个部分的变化会影响其他部分实现功能。

例一：建筑物通过结构来承受重力，古代建筑多采用“梁柱结构”当柱与柱之间的梁的跨度增大时，则梁的厚度要相应地加大，否则就不足以承受设计所需的重力，梁的跨度与梁的厚度之间的关系就反映了这一系统内部要素与要素之间的相关性。

例二：做一次家务劳动——和面、包水饺。取干面粉若干，自来水适量，将水逐量加入到干面粉中，直到面团的软硬适中。初次操作的同学，可能会出现面粉多了加水，水多了加面粉的现象，以至于和的面团越来越大。



（四）系统具有环境适应性

任何系统都存在于一定的环境中，必然要与外界进行物质、能量和信息的交换，外界环境的变化会相应地引起系统功能和内部组成的变化。系统具有适应环境的变化，保持原有功能的特性。

例：鹿、兔跑的快可以逃避猛兽的追捕，刺猬浑身是刺可以让野兽无从下口。

（五）系统具有层次性

系统无论大小，都可以分解为一系列的子系统，并存在一定的层次结构。系统各个层次具有独自的功能，他们通过相互联系、相互作用共同完成系统的功能。

自行车系统分解

子系统	组成零部件
制动子系统	刹车柄、车闸、连接线等
承重子系统	鞍座、车架、车轮等
行驶子系统	前后车轮组成
附设子系统	挡泥板、车后架、车铃等

自行车及其部件构成

1.2 系统分析与设计方法

修建茅屋需要分析设计吗？

修建大厦需要分析设计吗？

卓越 VS 普通

举重：成果差距约 3:1

篮球：成果差距约 50:1

软件开发：　　成果差距约 260：1

软件危机

引起灾难性后果的实例非常多，如在美国国防部统计的17个主要的软件项目中，100％延时，计划28个月的项目花费了48个月才完成，要求4年完成的项目用了7年还没有提交。

缺乏管理软件过程的能力，主要体现在：

延时

超出成本

新的技术和工具的好处难以体现

结论：很少有这样的领域，在最佳实践与一般实践之间有如此巨大的鸿沟！

掌握系统的特性可以帮助我们全面、深入地认识事物，掌握事物的发展规律，从而提高分析处理问题的能力和工作效率。系统的特性归纳了我们分析、解决问题应遵循的基本原则，构成了系统的基本思想。

你发现了什么?

1　3　7　8

5　9

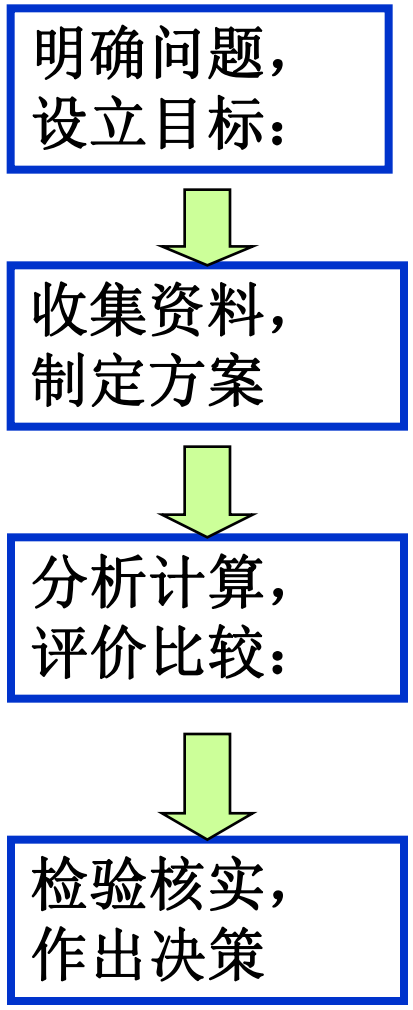
2　4　6

- 系统分析（System Analysis）是对一种业务问题域的学习活动，能够在系统解决方案中为提升系统性能和明确业务需求提供良好的建议。（理解问题域）

- 系统设计（System Design）是对系统分析中已确定的业务需求的说明或者构建一种相关技术的解决方案。（求可行解）

- 系统分析与设计的过程本质是一种认知活动，把设想中的结构落在一个抽象问题域上，对从各种用户那里得到的不同信息进行加工；作出一个逻辑的、一致的规格说明，以产生一个成功的系统。

系统分析的步骤



明确要研究问题的性质和范围，提出所要达到的目标，明确约束条件。

收集相关资料，制定解决问题的各种备选方案，预计可能产生的各种结果。

对资料和数据做必要的计算，进行各子系统的分析，再进行系统的整体分析，将各种方案进行评价对比，选择最佳方案。

如果对制定的方案不满意，还可按上述程序反复进行，直到获的满意为止。

案例分析

田忌赛马

	方　　案	W:T
1	W上：T上，W中：T中，W下：T下	3：0
2	W上：T中，W中：T下，W下：T上	2：1
3	W上：T下，W中：T上，W下：T中	1：2
4	W上：T上，W中：T下，W下：T中	2：1
5	W上：T中，W中：T上，W下：T下	2：1
6	W上：T下，W中：T中，W下：T上	2：1

案例分析：

丁谓修复皇宫

系统分析首先要着眼于系统整体，要先分析整体，再分析部分；先看全局，后看局部；先看全过程，再看某一个阶段；先看长远，再看当前。

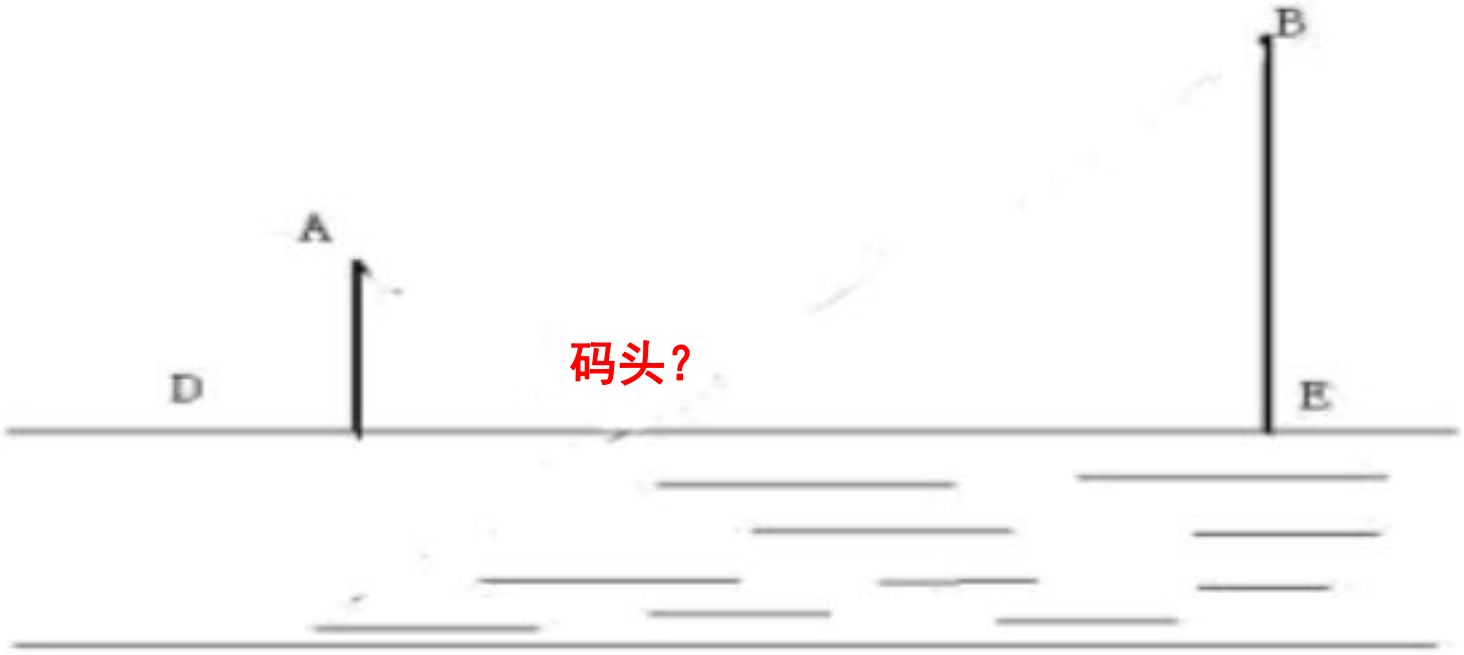


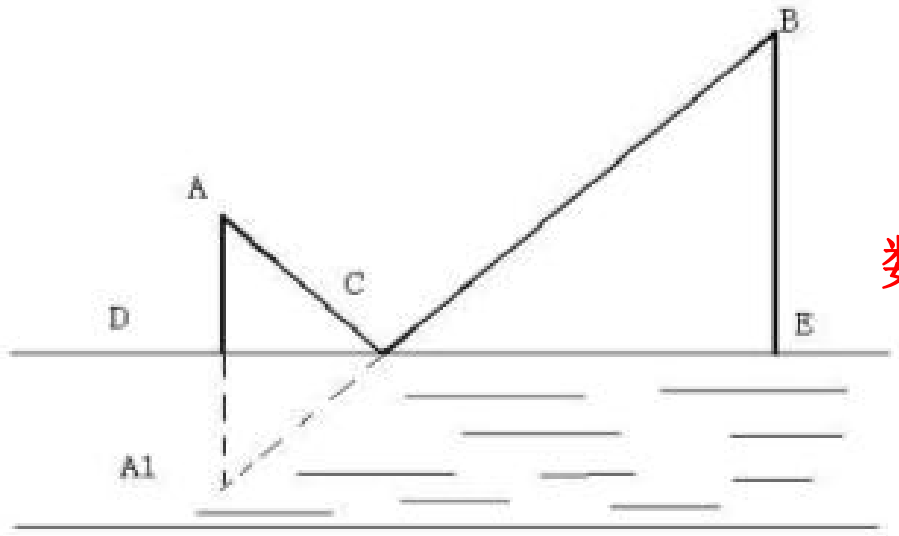
丁谓所采用的方案为什么能一举三得？

分析丁谓修复皇宫

阐明问题 确定目标	分析研究 制定方案	评价比较 确定最佳方案
北宋真宗年间，皇城失火，宫殿烧毁，大臣丁谓主持了皇宫修复工程。	方案一：运走皇宫中的瓦砾灰土→从城外取土烧砖→ 从汴水经陆路运输建筑材料 方案二：皇宫前大街挖沟取土烧砖形成水渠→将汴水引入水渠运输建筑材料→完工后将皇宫中的瓦砾灰土回填沟渠修复大街。	丁谓把取土烧砖、运输、回填三者看成一个整体中的相关部分，加以协调处理，得出方案二为最佳方案

【案例分析】在江边一侧有A、B两个工厂，它们到江边的距离分别是2km和3km，设两工厂沿江方向的距离是3.5km，现在要在江边修建一个码头，使得两厂的产品能够顺利过江，问码头应建在什么位置，才能使运输路线最短？





数学模型为： Smin=AC+BC

过A点作关于直线DE的对称点A1，连接A1B与DE相交于C，这一点既为所求的码头的地点。

根据相似三角形原理，求得 DC=1.4km，码头建在与A厂到江边垂直距离位置相距1.4km处，运输路线最短。

一、明确问题，设立目标

施工项目：教学楼装修

施工内容：教学楼A、B、C栋，需要水电、木工、油漆三个施工过程，每个施工过程需要3周时间

目标：组织装修，使装修工期和资源利用最为合理。

二、收集资料，制定方案

方案一：按照施工要求，从A到B到C，依次进行施工。

方案二：按照施工要求，A、B、C栋同时进行施工。

方案三：按照施工要求，A、B、C栋进行流水施工。

楼	装修过程	工期	装修施工进度（周）								
			3	6	9	12	15	18	21	24	27
A	水电	3									
	木工	3									
	油漆	3									
B	水电	3									
	木工	3									
	油漆	3									
C	水电	3									
	木工	3									
	油漆	3									

三、分析计算，评价比较

	施工特点	工期	施工队	设备、材料	现场管理
方案一	依此施工	27周	1	投入少	简单
方案二	平行施工	9周	3	投入多	复杂
方案三	流水施工	15周	1	较多	较简单

四、检验核实，确定方案

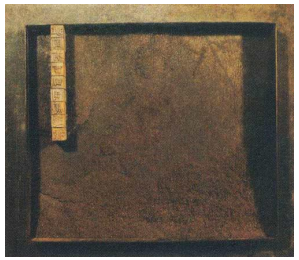
根据分析，确定某方案。

1.3 软件研发趋势

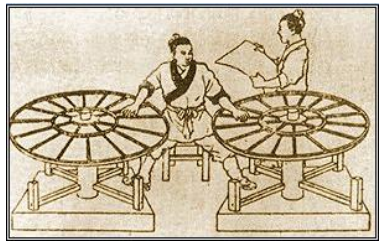
信息技术的发展历程及意义

发展历程	发明与应用的信息技术	提高的信息能力方面的意义	发生年代
第1次	语言的产生和使用	是从猿进化到人类的重要标志，较远距离的传递。	20万年以前
第2次	文字的创造与使用	信息的存储、传递的能力超越时空限制	公元前3500年
第3次	造纸术和印刷术的发明和应用	信息量大存储、及时交流、广泛传播，扩大了信息交流的范围	造纸术105年 印刷术1040年
第4次	电报、电话、广播、电视的发明和普及应用	提高传递的效率，进一步突破时空限制	电话1860年 电报1837年
第5次	计算机和网络的普及应用	处理、传递速度和普及应用程度惊人变化。将人类推进到了数字化的信息时代。	计算机1943年

甲骨文



活字印刷术使用的泥模字块



活字印刷转轮排字盘

信息技术的发展历程及意义

不是我不明白

是這世界變化快

当摩托罗拉还沉当摩托罗拉还沉醉在V8088的时候，不知道诺基亚已迎头赶上。

当诺基亚还注重低端机市场时，乔布斯的苹果已经潜入。

当苹果成为街机的时候，华为已经傲视天下。

当中国移动沾沾自喜为中国最大的通讯商时，浑然不觉微信客户已突破8个亿。

当中国银行业赚的盆满钵满高歌猛进时，阿里巴巴已经推出网络虚拟信用卡。

当很多人还在想租个门面房开个小生意时，光棍节一天中国互联网上创造天价成交额。

软件工程诞生于1968年

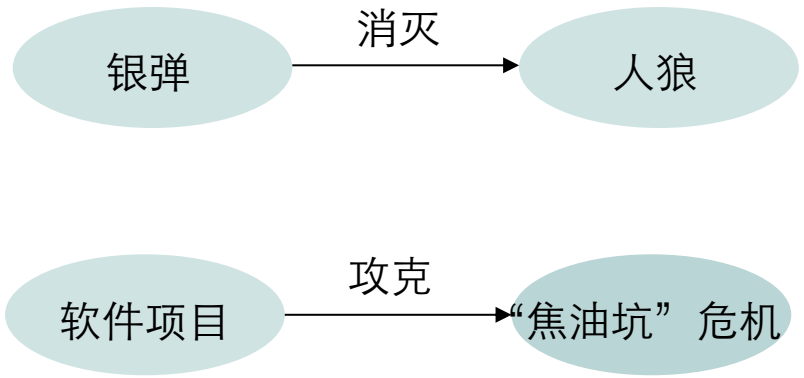
- 1968年秋， NATO（北约）的科技委员会召集了近50名一流的编程人员、计算机科学家和工业界巨头，**讨论和制定摆脱“软件危机”的对策**，第一次提出了软件工程（software engineering）。
- 软件工程研究和应用如何以系统性的、规范化的、可定量的**过程化方法**去开发和维护软件，以及如何把经过时间考验而证明正确的**管理技术**和当前能够得到的最好的**技术方法**结合起来的学科。

System/360的开发过程被视为计算机发展史上最大的一次豪赌。为了研发System/360这台大型机，IBM决定征召6万多名新员工，创建5座新工厂，而当时出货的时间不断的顺延。

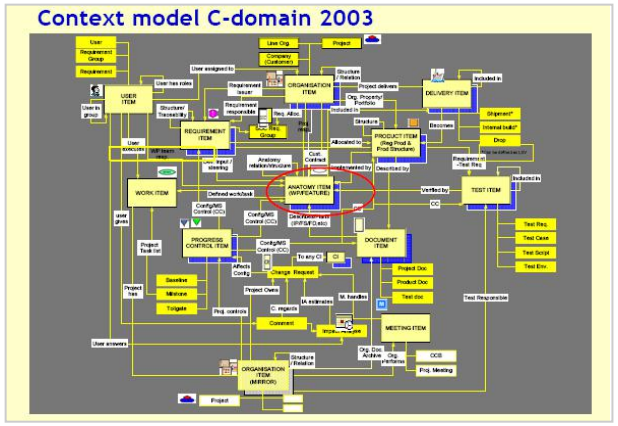
软件没有银弹：软件工程中的根本和次要问题

- Frederick P. Brooks在《人月神话》中指出：
- 现代软件系统中无法规避的内在特性：复杂性、一致性、可变性和不可见性。其中复杂性和可变性是根本问题

图：1685年德国线刻版画人狼故事

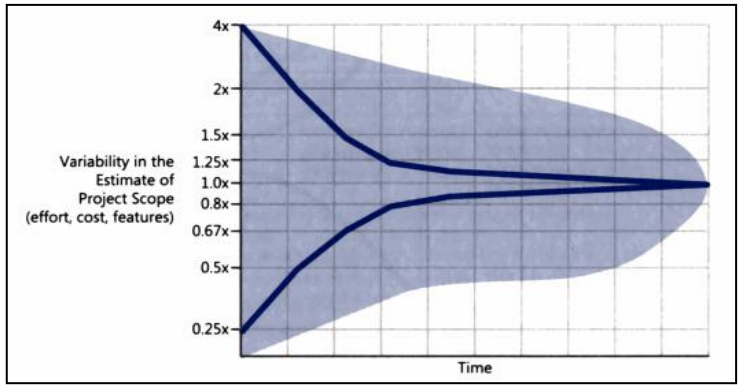


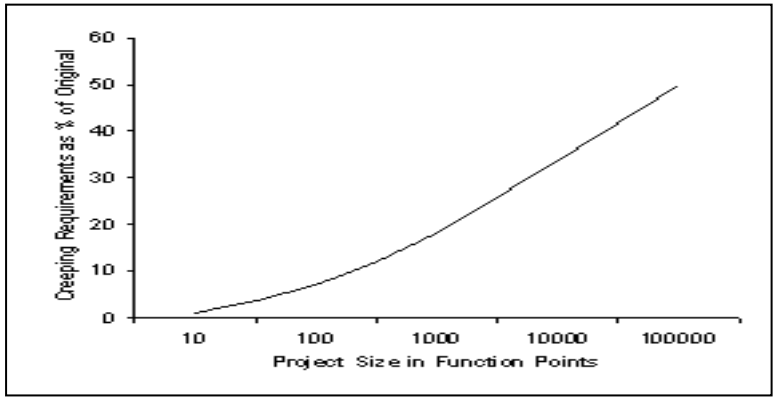
复杂性



- 规模上，软件实体可能比任何由人类创造的其他实体更复杂
- 软件拥有大量的状态（gcc运行的瞬间有千万个状态值），这使得构思、描述和测试非常困难
- 复杂性导致技术实现上的困难，从而又引发管理上的问题，它使得全面理解问题变得困难，引起大量的学习和理解上的负担，使开发慢慢演变成灾难

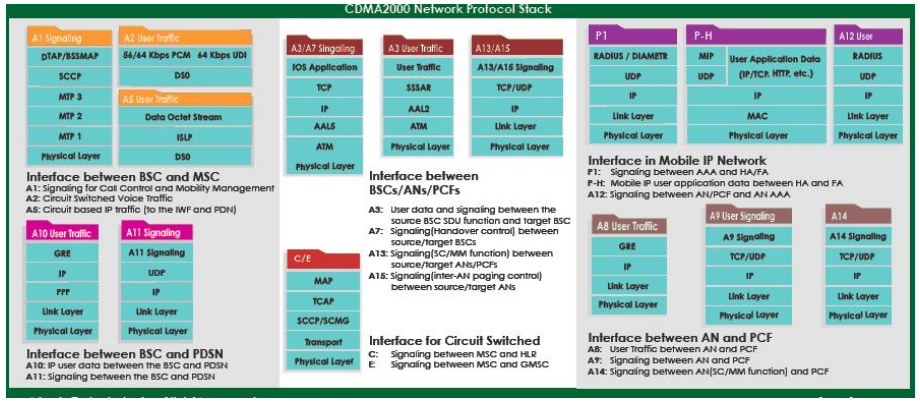
可变性





- 软件是纯粹思维活动的产物，可以无限扩展，并且，软件的修改更容易。
- 客户对系统功能的要求随时可能变化，而软件修改更容易，因此往往软件被更改。
- 软件的设计和实现随着技术、开发语言、方法、人的经验习惯等随时可能变化。

一致性

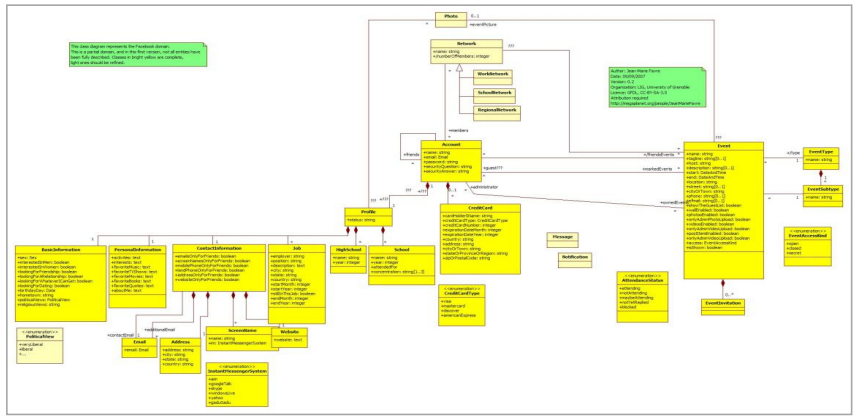


- 不同人有不同的经验和惯例，而软件工程师必须掌握由不同人设计的系统，其中很多系统复杂度是随心所欲、毫无规则可言的。
- 新开发的软件，必须遵循各种接口，软件开发的目标就是兼容性，很多复杂性来自保持与其他接口的一致性，这是非常困难的。

例如：

- 需求分析人员列出的规格是用户想要的么？
- 研发人员的实现和系统工程师的想法一致么？
- 高层软件组和底层软件组对接口理解一致么？

不可见性



- 软件是不可见的，无法可视化，难以给人全面、直观的感受，不同于建筑业、机械业等。
- 当前的软件表达方式（数据流图、流程控制图等）和建模方法，都无法详尽展现软件，无法让不同的人理解一致，这严重阻碍了交流，从而限制了设计和实现过程。

传统软件开发 VS 现代软件开发

传统软件开发

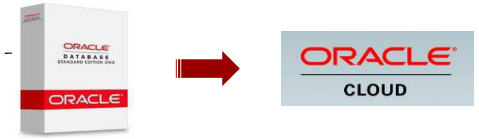
- 把软件开发类比为传统工业，规范后可重复“生产”
  - 计划、预测
  - 预定义过程和分阶段控制
  - 一次性交付，变更成本大
  - 严格过程监控
  - 过程决定质量

现代软件开发

- 认为软件开发是复杂的有机生态系统，不可重复
  - 探索、适应、演进
  - 在变化中基于经验的过程
  - 小批量快速交付，拥抱变化
  - 严格交付验收
  - 高效的人和协作是核心

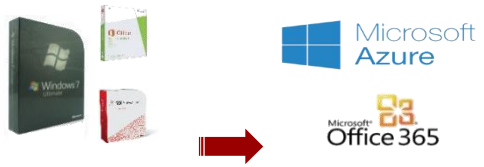
传统IT软件厂商在向云服务转型

未来十年：
80%企业应用迁移到SaaS云
100% Dev/Test运行在云上
绝大部分企业数据存储在云



用5年时间重写几乎所有软件，提供云服务

……我们的战略是为移动优先、云优先的世界构建一流的最好的平台和生产力业务。



用混合云服务为用户提供良好的统一体验

云软件是商业模式的变革，是“供应商+客户/运营商”到“开发+运营”的转变。

云计算时代软件的变化

- **云软件的商业模式发生了变化……**
  - 云软件由卖license到卖service (hosting)方式；
  - 软件的部署，由过去分散小规模，转为集中大规模；
  - 客户更期望更快、更新、更好的特性和服务；
  - 云服务商业竞争的法宝是“快”和体验，按周发布特性；
- **云软件的系统架构发生了变化……**
  - 软件架构模式的转变，由过去Silo、分层模型，转为SOA服务化、网状模型、微服务；
  - 由单租户、集中式，转为多租户、服务化、分布式，系统用户由百、千到现在上亿用户；因为要大规模、因为要快速创新，要求软件的子系统之间、服务与服务之间进一步解耦；
  - 软件子系统不再关注功能的全面/完备，而转为关注精、专业、以及接口的标准化，即一个子系统/服务只做一件事，这样所有系统组成一个大平台的时候，就类似乐高积木；
- **云软件的开发模式发生了变化……**
  - 全功能团队 / FullStack工程师；小团队自己规划、自己决策、自己交付；
  - 每个“人”都是围绕业务目标，进行自我激发的；而不是依靠流程来驱动的；
  - 云服务的核新竞争力“快”和“体验”，这只能依靠小团队、个人的自我激发才能实现。而过去大团队、强调组织/规划/流程，忽视个人，恰恰是障碍；

多语言混合编程成为常态，新语言不断涌现

JavaScript，Python、Ruby，Go已经在开源社区广泛使用

编程语言	1~10	1~100	1~1000	1~5000
JavaScript	7	54	385	1605
CSS	2	8	41	174
Ruby	1	9	153	786
Python		5	64	420
Unknown		5	30	138
C++		4	22	108
PHP		3	38	248
Shell		3	19	89
Objective-C		2	89	495
C		2	31	185
Go		2	13	61
Java		1	32	255

在云计算、大数据等开源项目中，多语言混合编程成为常态

企业自身软件交付需应对来自市场、协作、开放与安全的多重挑战

交付频率高，研发周期短

- 市场变化快，产品盈利窗口窄
- 按需发布，一天交付多次，快速试错，快速反馈
- 持续快速创新，快速将idea转变为产品

跨地域协作多，研发平台复杂

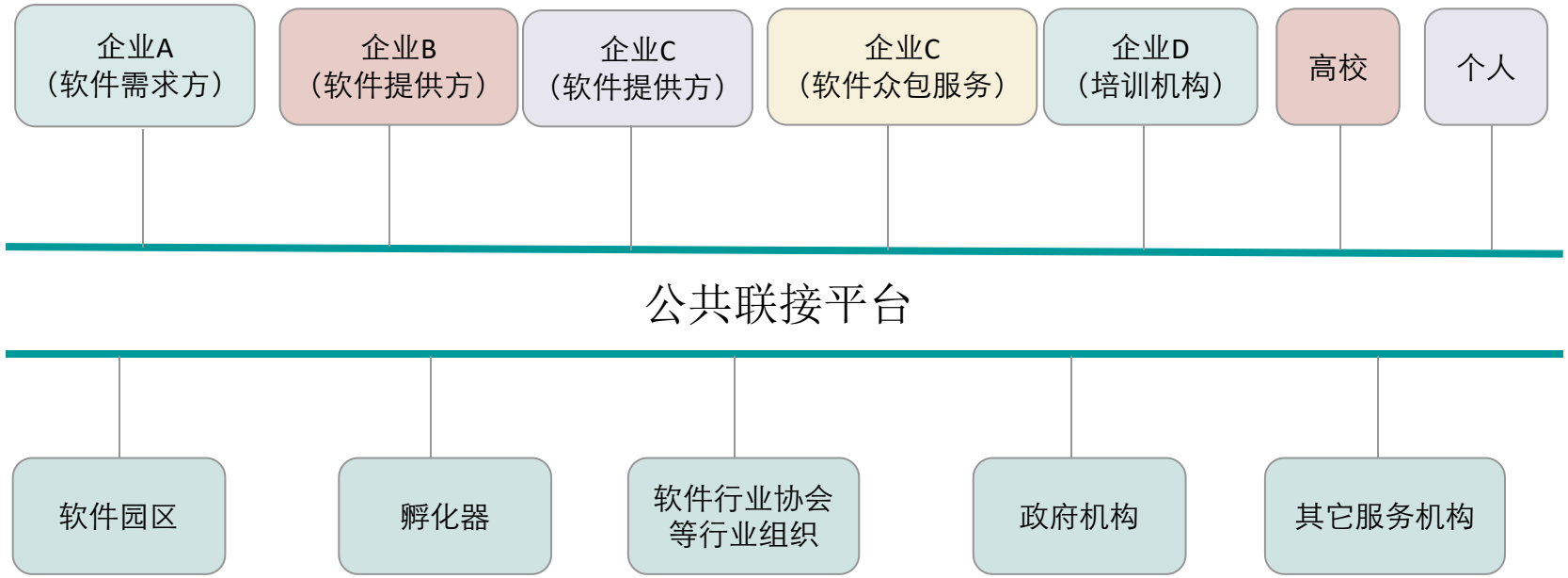
- 国际化、跨地域团队沟通协作多
- 新技术、新语言学习曲线长
- 工具部署和维护低效、复杂

开放与安全要求高

- 平台开放，能与其它平台进行对接或数据交互
- 数据在传输与存储上要求安全并且可靠

软件企业的竞争已经从单一产品竞争演进到生态联盟竞争

- 软件需求方与软件提供方、软件提供方之间更好地协作，以共同把握需求、进度、质量以及风险。
- 软件园区、孵化器向软服务转型，整合在园企业优势，并为企业提供研发架构、质量、工具等方面的服务与经验。
- 高校、培训机构等与产业需求密切衔接，培养符合产业需求并具有工程化实战能力的人才。



1.4 DevOps研发模式

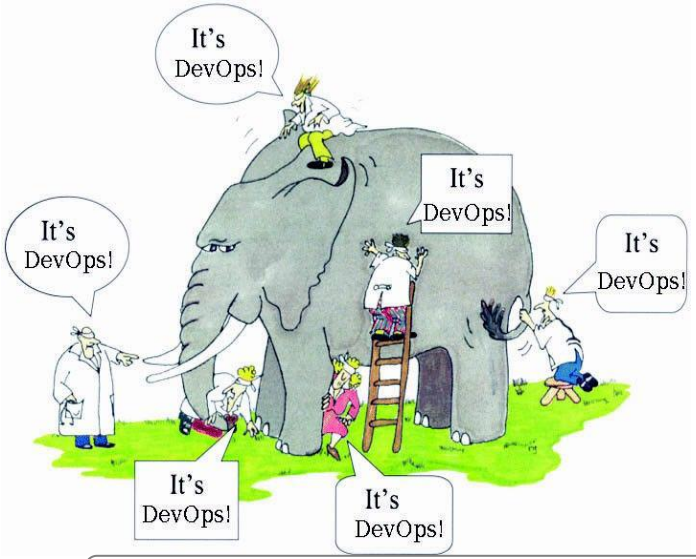
DevOps起源

源于Google、Amazon、Facebook等企业实践，2008年Patrick Debois在“Agile 2008 conference”首次提出Devops术语，由Flickr展示的开创性的“一天10次部署”，基础设施即代码“(Mark Burgess 和Luke Kanies)，“敏捷基础设施”（Andrew Shafer)，“敏捷系统管理”(Patrick DeBois)，Amazon的“平台即服务”，这些相辅相成，让DevOps在2012-2013成为IT业界潮流。

Dev和Ops存在“混乱之墙”—Andrew Shafer

- **不同的世界观：**运维人员要求稳定可靠，认为变更充满风险，开发人员则被鼓励频繁发布新代码，认为运维部门对流程的坚持，阻碍了开发的速度。
- **它在我的机器上没有问题！**常常听说开发人员这么说，而运维团队的确遇到了麻烦-->开发和运维之间的脚本、配置、过程和环境存在差别。
- **沟通壁垒：**开发和运维团队通常处于公司组织的不同部门，通常有不同管理者，通常是不信任的关系，并且通常工作在不同的地点。

DevOps的不同视角



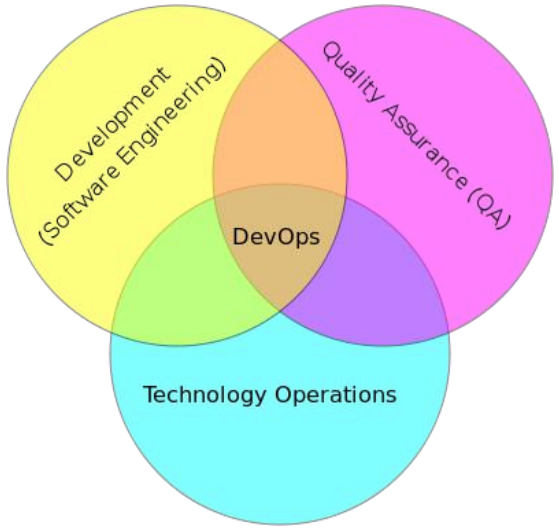
研发工程师：DevOps是一组技术和工具(Jenkins/Puppet..)

咨询师：DevOps是一种新的软件开发模式

HR：DevOps是一个组织和职位

大师：DevOps是一种文化和理念

企业高管：DevOps是一种全新的商业模式和管理体系



“DevOps是软件开发、运维和质量保证三个部门之间的沟通、协作和集成所采用的流程、方法和体系的一个集合，它是人们为了及时生产软件产品或服务，以满足某个业务目标，对开发与运维之间相互依存关系的一种新的理解。———WikiPedia

DevOps兴起的驱动因素

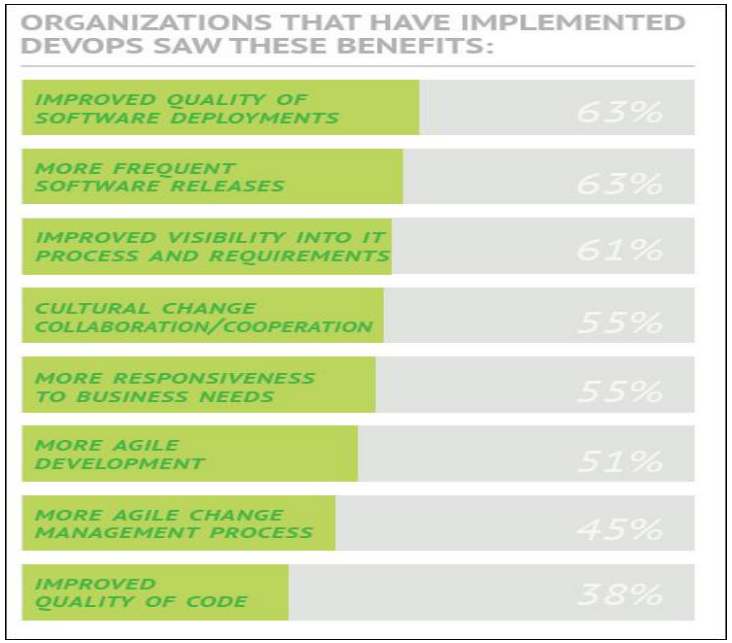
业务负责人要求加快产品交付的诉求

大规模使用敏捷软件开发过程与方法

虚拟化和云计算基础设施日益普遍

数据中心自动化技术和配置管理工具的普及

DevOps收益



【实施DevOps的收益调查（排序）】：

- 提升软件部署质量（占63%）
- 更频繁的软件发布（63%）
- 提升IT过程和需求的可视化程度（61%）
- 改变合作协同文化（55%）
- 响应更多的业务需求（55%）
- 开发更敏捷（51%）
- 管理过程更敏捷（45%）
- 提升代码质量（38%）

《 State of DevOps Report》—2013年，90个国家4000人调查报告

截止2014年，采用DevOps的IT组织比例为66%， IT组织的效率要高出5到7倍，变更多出14倍，变更故障率下降50%。

定义DevOps

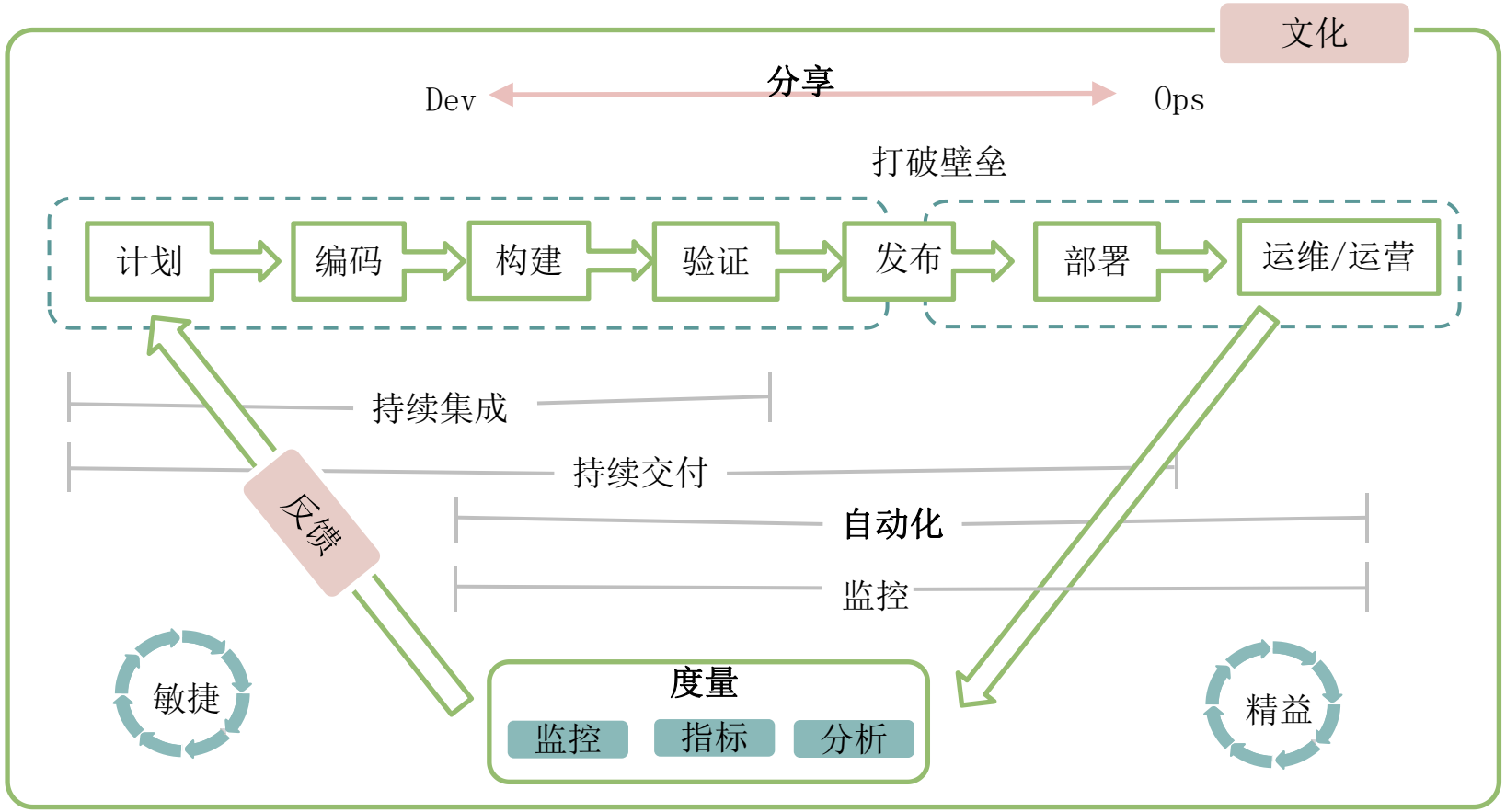
DevOps是一套实践方法，在保证高质量的前提下缩短系统变更从提交到部署至生产环境的时间。

- 部署对系统的变更时，质量很重要：保证方法有上生产环境前跑通自动化测试用例、先让一小部分用户对生产环境的变更进行测试、对新部署的代码密切监控。
- 交付机制是高质量的：交付机制应具备高度的可靠性和可重复性。
- 两个时间很重要：一是开发人员提交新开发的代码的时间；另一个是代码部署到生产环境的时间。
- 目标导向：不拘泥于实践的形式或者是否使用工具，目标是减少从提交代码到部署到生产环境之间的时间。
- 不局限于用户测试和部署实践：在需求阶段就包含运维人员的视角，是非常重要的。

DevOps的五个要素

- 文化：建立一体化的全功能团队，打破开发(Dev)与技术运营(Ops)隔阂
- 自动化：自动化一切可以自动化的
- 精益：以精益的方式小步快跑，持续改善
- 度量：建立有效的监控与度量手段快速获得反馈，推动产品和团队的持续改进
- 分享：不同职能、不同产品之间分享经验

DevOps生命周期过程



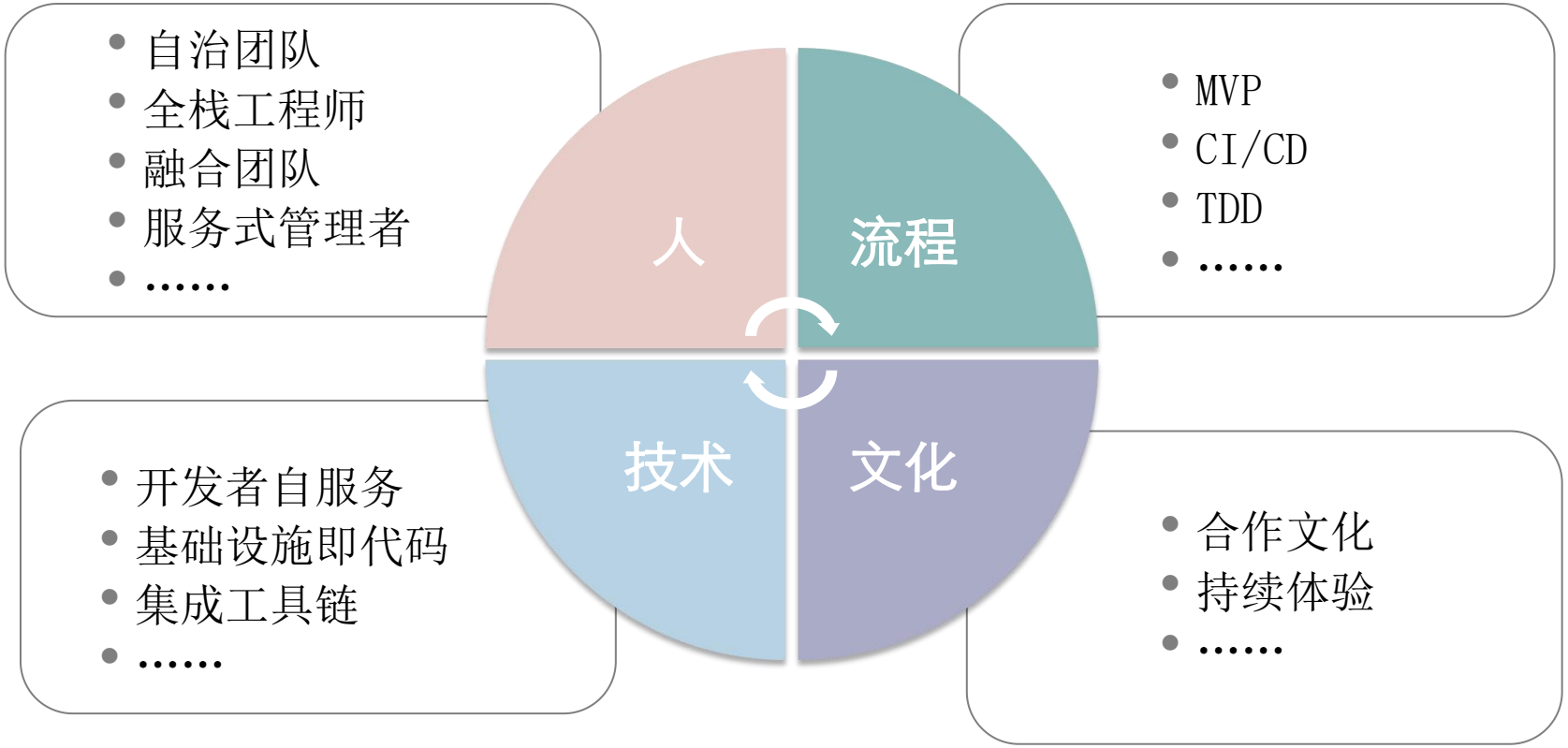
DevOps与敏捷

- DevOps是敏捷理念从开发领域向运维领域的延伸
  - 计划阶段：运维人员为开发人员提供需求，并制定发布计划
  - 编码/构建/验证阶段：Scrum、极限编程和精益生产，持续集成、自动化测试等
  - 部署阶段：开发团队负责部署、监控部署过程，以及部署后的运行

DevOps转型障碍

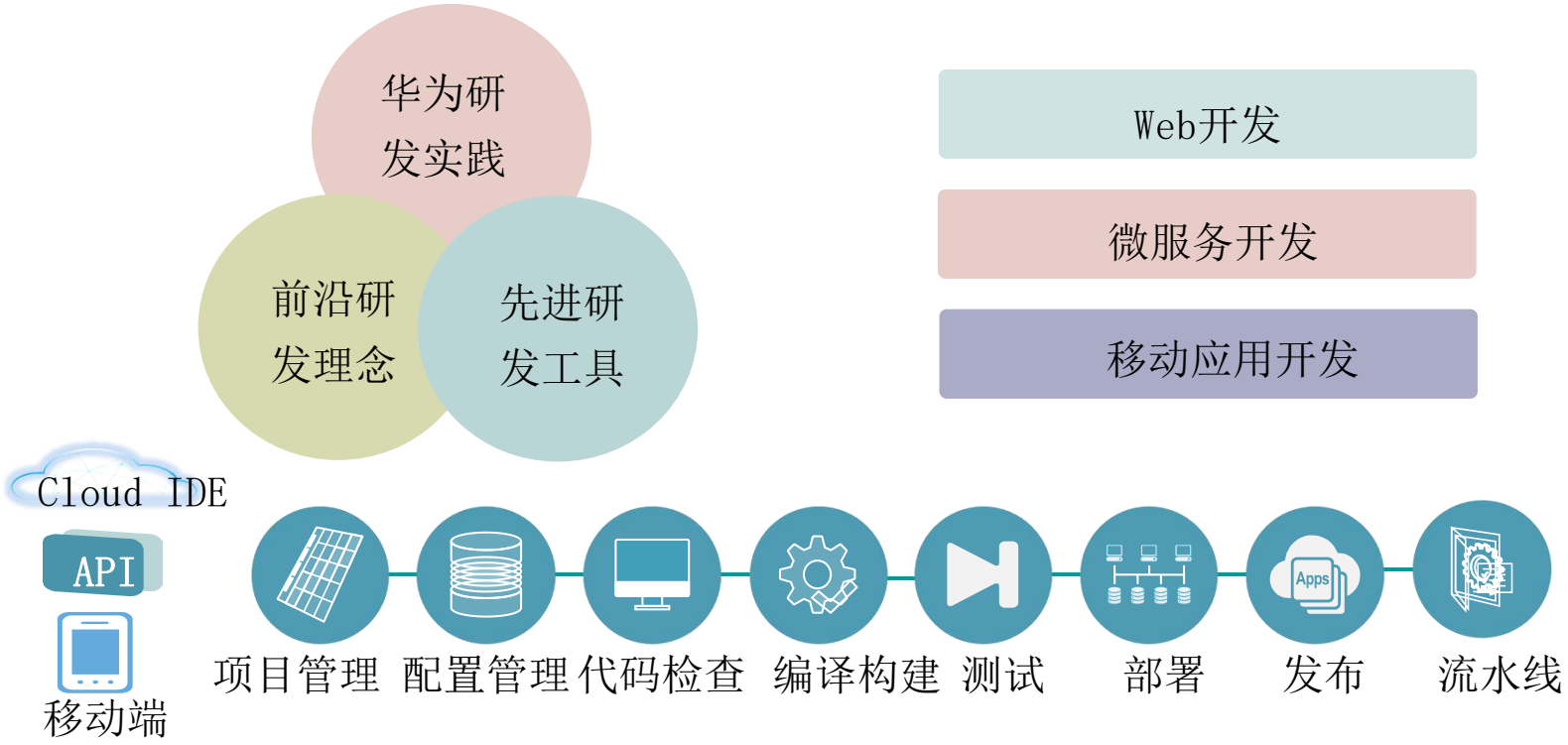
- **文化及组织类型**
  - 权衡缩短面市时间所带来的收益与某些事情出错的风险，例如在监管领域运作的组织（金融等）
  - 对组织中其他部分的影响
- **部门类型**
  - 激励方式：对开发人员和运维人员的激励
- **筒仓思维方式**
  - 组织中的不同部门的思维
- **工具支持**
  - 工具需要专业技能以及经验
  - 确保所有的开发团队遵守一套过程
- **人员问题**
  - 将运维人员的任务转移给开发人员，成本可能会面临增加，要通过自动化等进行缩减
  - 为开发人员增加更多的任务可能会导致开发人员短缺

DevOps关键模式与实践



华为云（DevCloud ）：一站式云上DevOps平台

华为云（DevCloud ）是集华为研发实践、前沿研发理念、先进研发工具为一体的研发云平台；面向开发者提供研发工具服务，让软件开发简单高效。



华为云（DevCloud ）核心特性

DevOps研发模式

项目管理	配置管理	代码检查	编译构建	测试	部署	发布	流水线
• 敏捷迭代开发 • 多项目管理 • 看板跟踪 • 社交化协作 • 多维度报表 • 文档管理 • Wiki • 追溯能力	• 基于GIT跨地域协同开发 • 在线代码阅读修改 • 在线提交代码 • 分支管理 • 代码加密传输 • 基于代码的统计分析 • 基于角色的权限控制 • 关联需求与缺陷	• 缺陷快速定位和修复 • 主流编程语言 • 多种检查规则套餐 • 自定义检查规则集 • 缺陷批量处理 • 多维度报表	• Maven等主流构建标准 • 简化配置，简单易用 • 多种编程语言 • 多种工具插件 • 多语言并行构建	• 需求-用例-缺陷 • 用例管理 • 缺陷管理 • 测试设计 • 测试验收 • 缺陷统计 • 测试排行	• 云应用引擎CAE • 容器部署CCE • 集群管理 • 监控	• Maven仓库 • 软件包高速下载 • 软件包高速分发 • 软件包自动归档	• 可视化按需制定自动化流程 • 利用云端能力并行执行 • 实施监控流水线状态

谢谢