Software Architecture Introduction

软件体系结构概述

梁　鹏

武汉大学　计算机学院

liangp@whu.edu.cn

Why software architecture?

Grady Booch ✓
@Grady_Booch

The best formal and unambiguous specification of a software-intensive system is its executable code.

A specification that is, however, insufficient for visualizing, reasoning about, understanding, and communicating about that system.

6:53 AM · Nov 24, 2019 from Kaanapali, HI · Twitter for iPhone

Why architecture design?

The Computational Limits of Deep Learning

Neil C. Thompson[1*], Kristjan Greenewald[2], Keeheon Lee[3], Gabriel F. Manso[4]

[1]MIT Computer Science and A.I. Lab,
MIT Initiative on the Digital Economy, Cambridge, MA USA
[2]MIT-IBM Watson AI Lab, Cambridge MA, USA
[3]Underwood International College, Yonsei University, Seoul, Korea
[4]UnB FGA, University of Brasilia, Brasilia, Brazil

*To whom correspondence should be addressed; E-mail: neil_t@mit.edu.

To understand why **deep learning** is so **computationally expensive**, we analyze its statistical and computational scaling in theory. We show deep learning is not computationally expensive by accident, **but by design**.

深度学习**计算成本高昂**并非偶然，而是从**设计之时**就注定了。

软件工程能力、高质量产品

总 裁 办 电 子 邮 件

电邮讲话【2019】001号　　签发人：任正非

全面提升软件工程能力与实践，打造可信的高质量产品

致全体员工的一封信

我今天写信，是要和大家沟通公司如何全面提升软件工程能力和实践。二十年前的IPD变革，重构了我们的研发模式，实现了从依赖个人、偶然性推出成功产品，到制度化、持续地推出高质量产品的转变。至今为止，我们的产品和解决方案已经在170多个国家安全稳定运行，并因此积累和赢得了全球数万客户的信任。今天，我们又处在一个新的起点，全面云化、智能化、软件定义一切等发展趋势，对ICT基础设施产品的可信提出了前所未有的要求。可信将成为客户愿买、敢买和政府接受、信任华为的基本条件。可信不仅仅是产品外在表现的高质量结果，更是产品内在实现的高质量过程，是结果和过程的双重可验证的高质量。而只有全面提升软件工程能力和实践，才有可能打造出可信的高质量产品。

17次谈到架构

- 我们要优化并遵循公司各种编程规范，遵从架构与设计原则，熟练使用各种编程库和API，编写出简洁、规范、可读性强、健壮安全的代码

- 我们要深刻理解架构的核心要素，基于可信导向来进行架构与设计。在确保可信的前提下，要在性能、功能、扩展性等方面做好权衡；慎重地定义我们的模块与接口，真正做到高内聚与低耦合；

17次谈到架构

- 我们要遵循权限和攻击面最小化等安全设计原则，科学设计模块之间的隔离与接口，提升安全性；低阶架构与设计要遵循高阶的架构与设计原则，在充分理解原有架构与设计的情况下，持续优化；

17次谈到架构

- 我们要重构腐化的架构及不符合软件工程规范和质量要求的历史代码。我们知道，再好的架构，其生命力也是有限的。随着时间的推移、环境的变化以及新技术、新功能特性的引入，架构也会腐化。面对腐化了的架构，要毫不犹豫地去重构它。同时主动以可信设计原则为导向，去重构不符合软件工程规范和质量要求的历史代码，提升软件架构的生命力。

17次谈到架构

- 为此我们将建立一支更高水平的Committer角色群体，负责软件架构的看护、代码的审核和提交，整体保障合入代码的高质量。我们要变革考核机制，要让架构设计好。

你认为什么是软件体系结构？

- 框架？
- 结构？
- 概要设计？
- 构件和连接件？
- 你的理解？…

软件体系结构相关的应用问题?

- 同时满足安全性和性能，什么架构模式最好?
- 需求改变了，软件体系结构如何修改?
- 软件体系结构变化后，代码如何修改?
- 某个构件接口改变会影响到系统哪些部分?
- 某个架构师离职了，她/他做的设计如何理解?
- 没有软件体系结构文档，如何恢复设计?
- 在系统尚未实现的情况下，怎么判断某个软件体系结构设计是合理的?
- 如何避免潜在的软件体系结构设计风险?
- ...

软件体系结构设计的实例

- LOFAR System
- 苏宁易购

软件设计

- 软件过程：分析、**设计**、编码、测试、维护
- **抽象设计** vs. 详细设计
- 面向对象分析与设计(UML)
- 系统体系结构 vs. **软件体系结构**
- 什么是**软件体系结构设计**

大纲

- 软件体系结构发展史
- 什么是软件体系结构
- 软件体系结构的重要性
- 软件体系结构的常见术语

软件生命周期

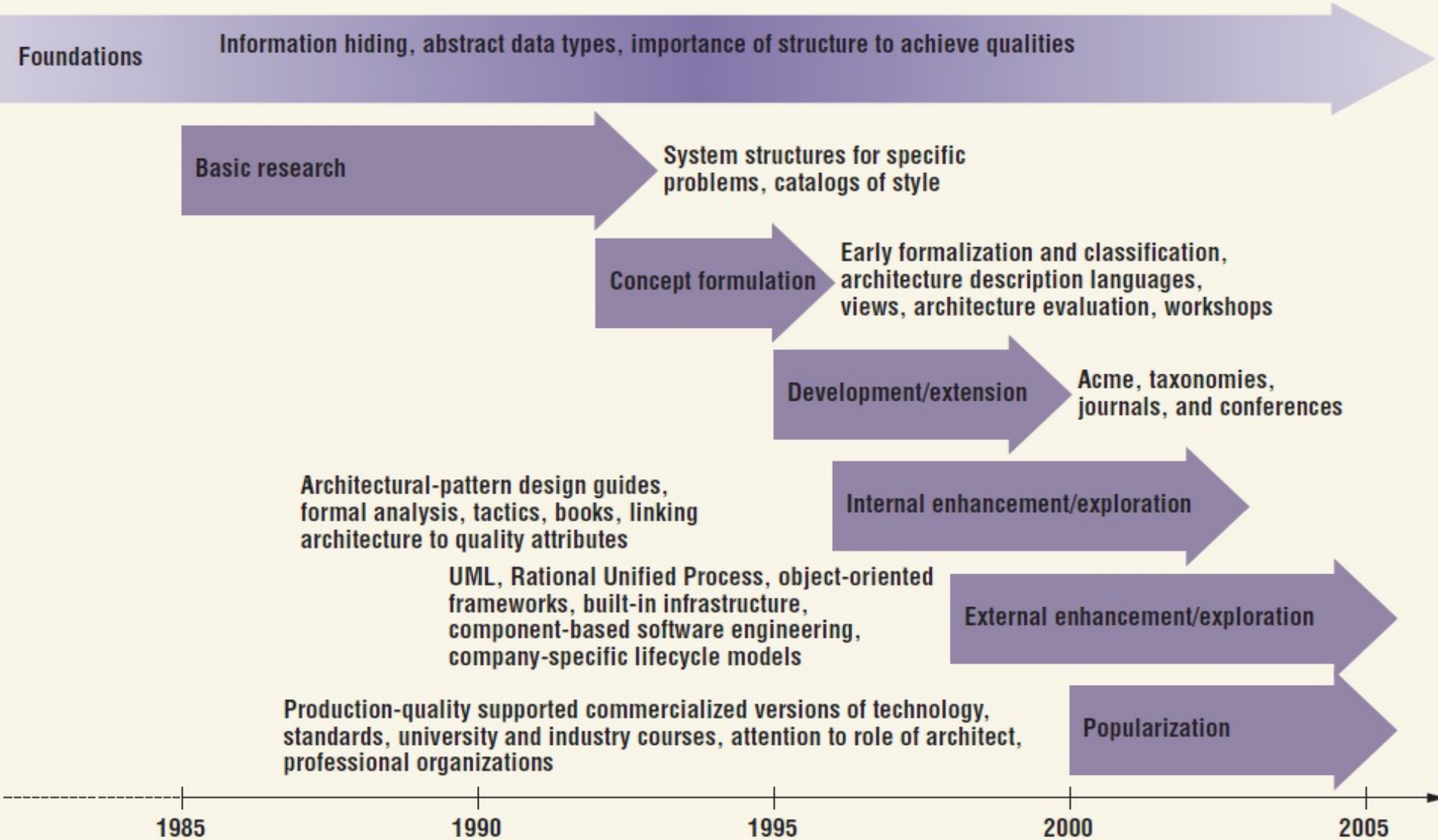
- 软件生命周期的主要活动阶段
  - （1）可行性研究和计划制定。
  - （2）需求分析
  - （3）软件设计
  - （4）软件实现
  - （5）软件测试
  - （6）运行和维护

- 软件体系结构属于软件设计的早期阶段，主要目的是为了确立软件的总体蓝图，并分析软件的质量属性。

软件体系结构的兴起

Antecedents of Software Architecture
Integrated product lines
2000
Component-based systems
Software architecture
1990
Packages
Pipes and filters
Software development environments
Inheritance
Abstract data types
objects
1980
Programming-in-the-large
Information hiding
1970
NATO SE conference
Separate compilation
1960
Subroutines
1950
Programming-in-the-world
Programming-in-the-large
Programming-in-the-small
Programming-any-which-way
Institute for Software Research, International

软件体系结构的发展阶段

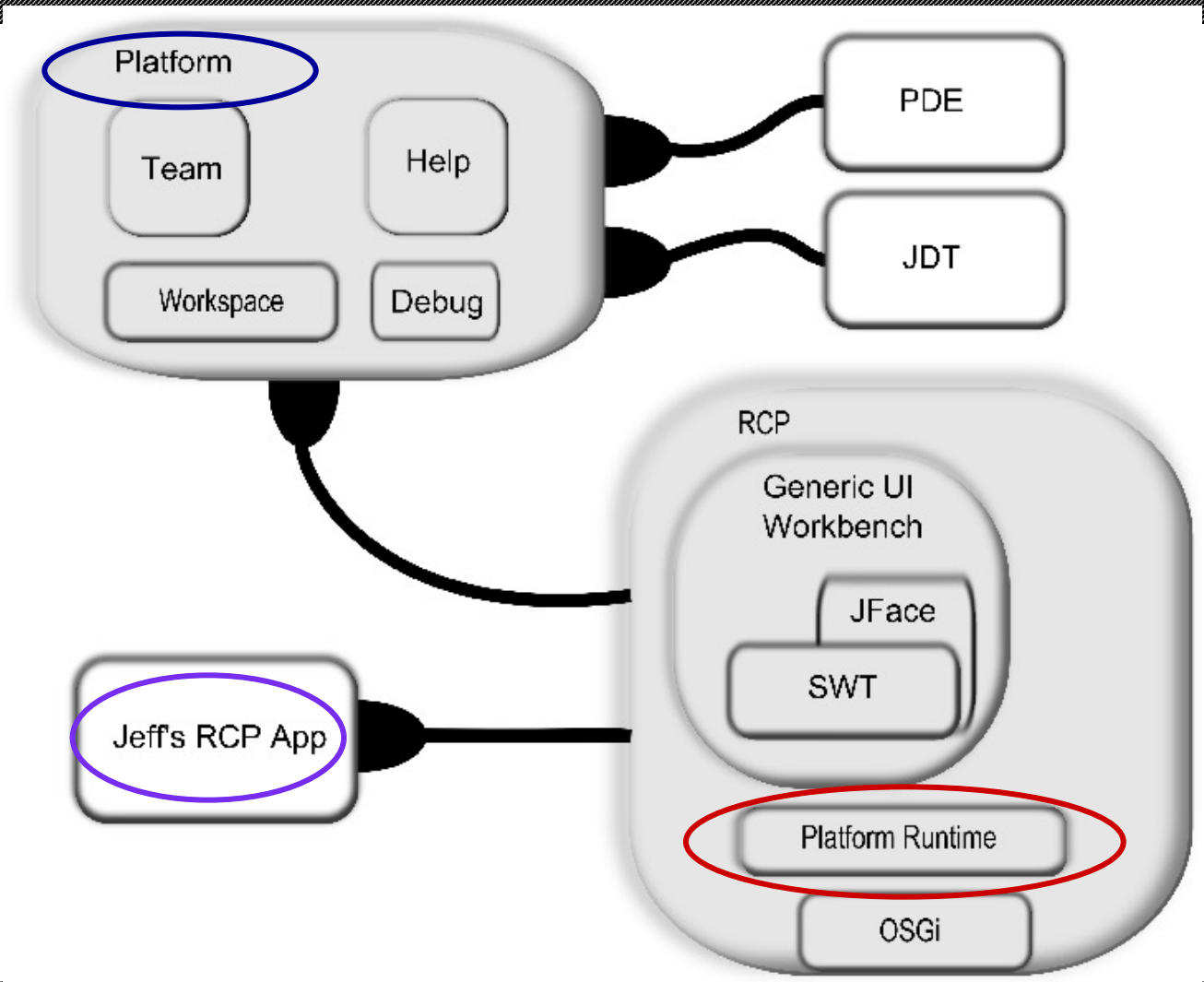


Mary Shaw：The Golden Age of Software Architecture

大纲

- 软件体系结构的发展史
- 什么是软件体系结构
- 软件体系结构的重要性
- 软件体系结构的常见术语

Eclipse 3.0 Architecture



Earth Observing System

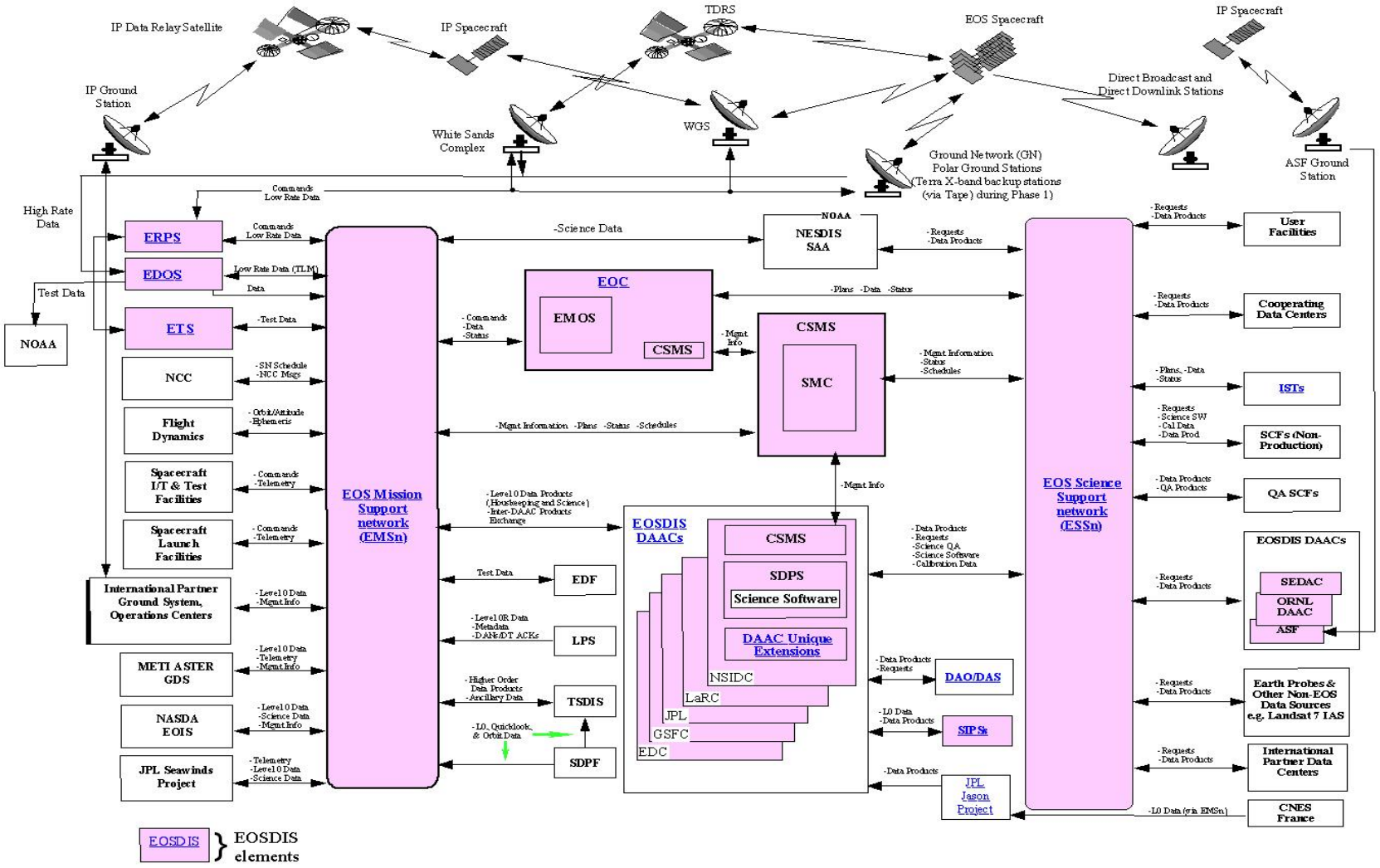
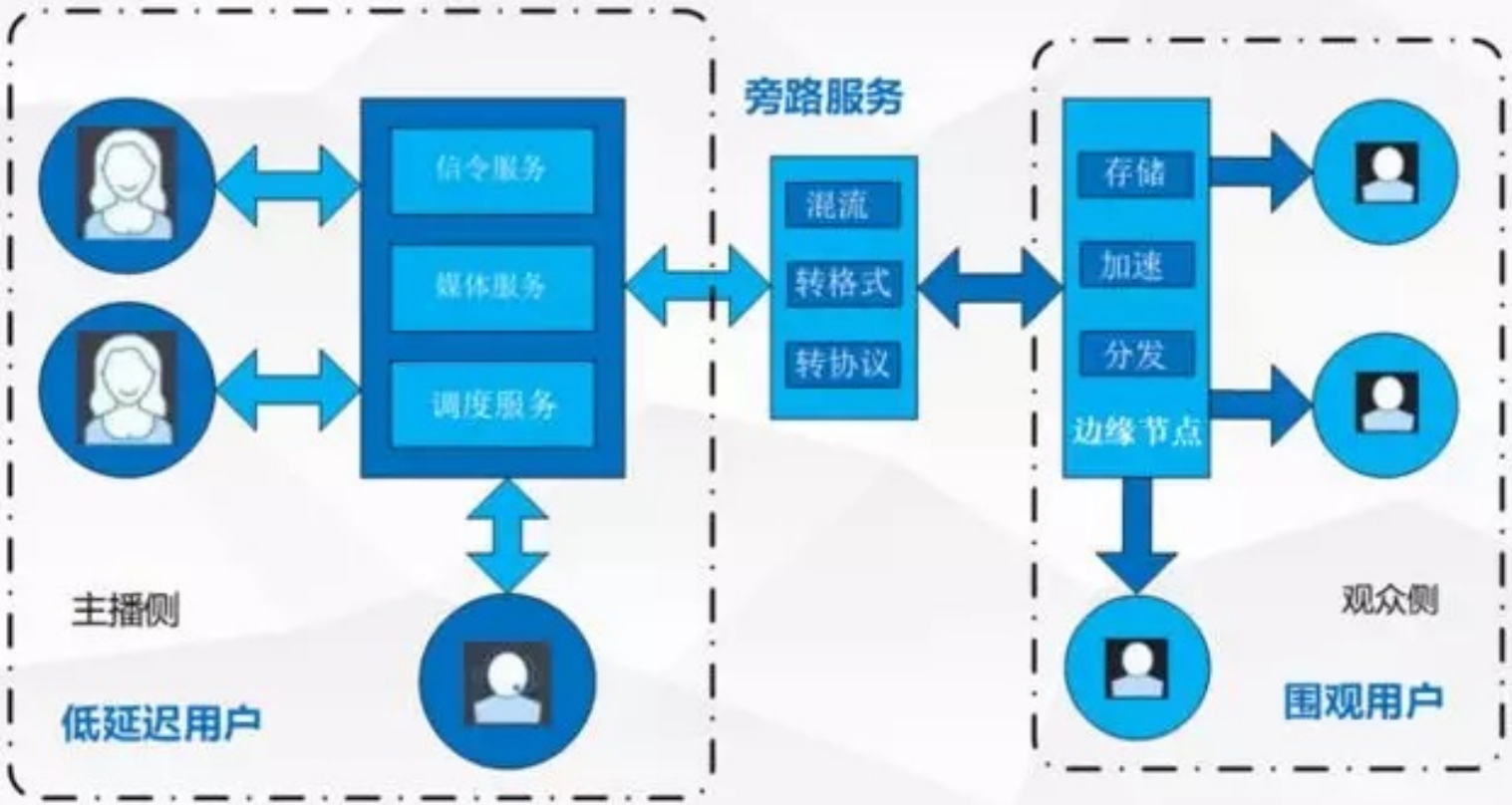


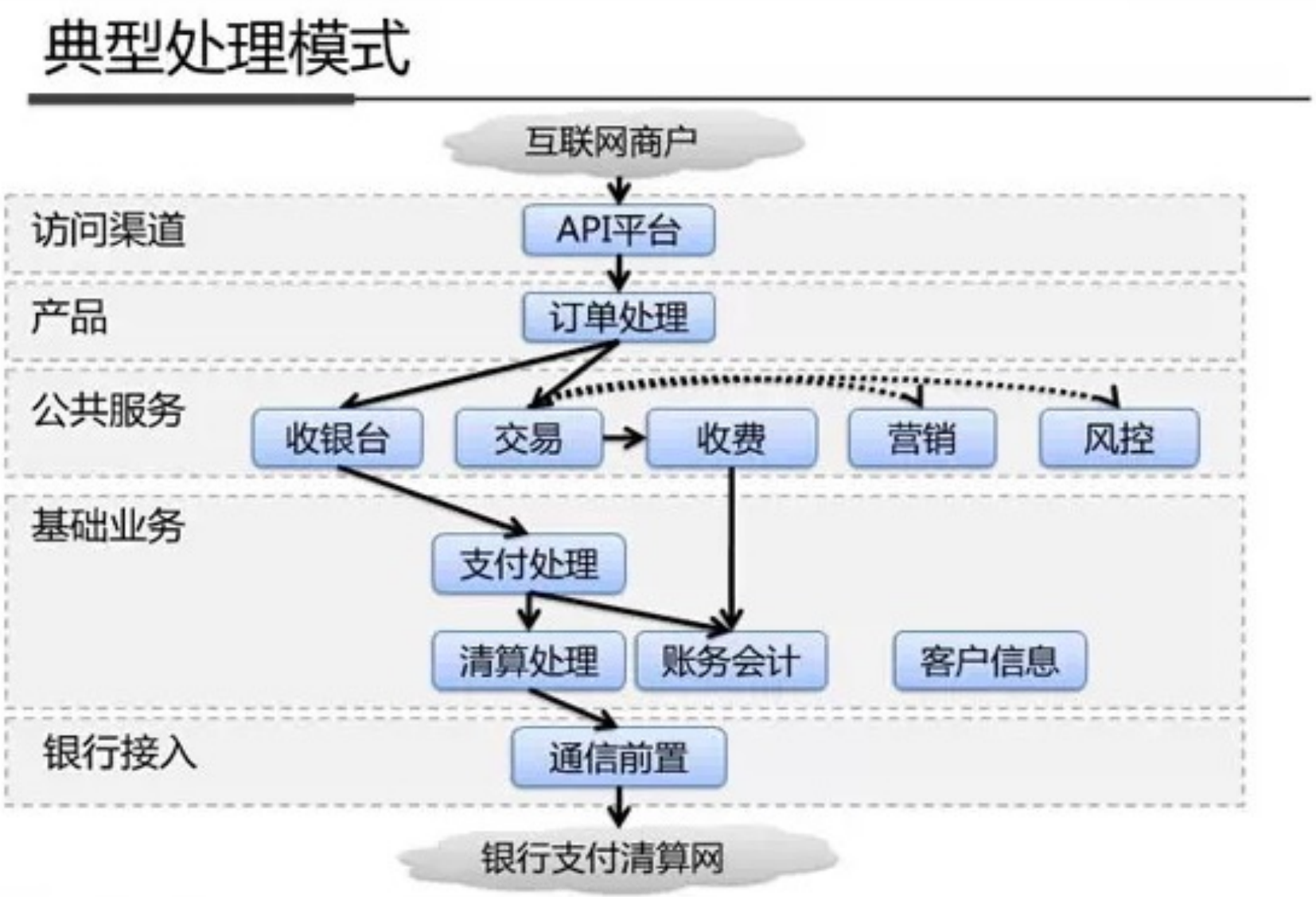
Figure 3-1. EOS Ground System High-Level Architecture

直播系统

互动直播的实时架构



支付宝钱包系统处理架构

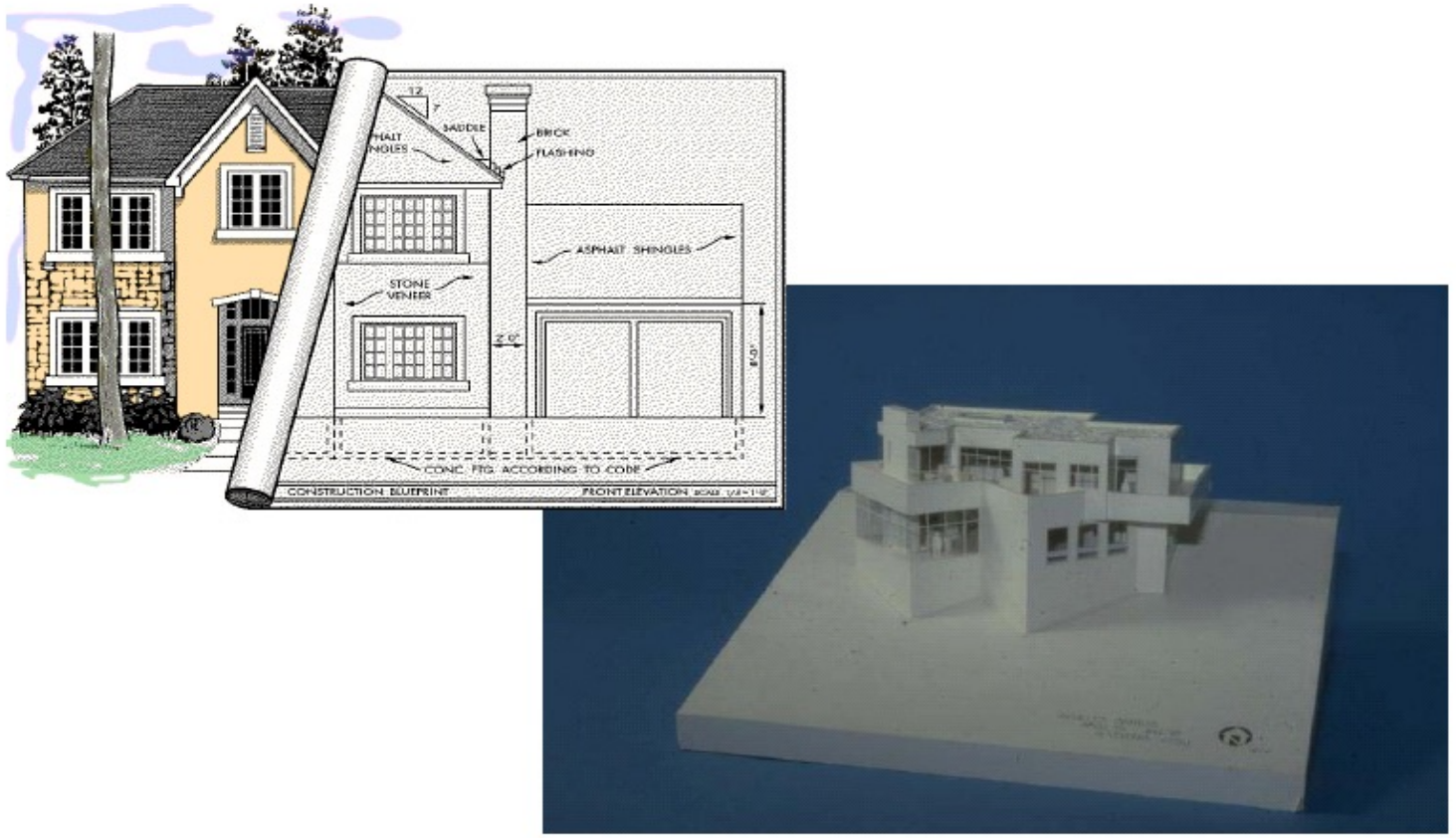


体系结构设计与规模的关系

一个人搭建
需要
- 简单建模
- 简单的过程
- 简单工具

一个团队高效适时地建造，
需要
- 建模
- 良好的过程定义
- 良好的工具

建筑的“体系结构”设计


SADDLE
BRICK
FLASHING
ASPHALT SHINGLES
STONE VENEER
CONC. FTG. ACCORDING TO CODE
CONSTRUCTION BLUEPRINT
FRONT ELEVATION

建筑的“体系结构”设计

- 建筑中包含多个层面的技术
  - 外观、挖掘、地基、结构、墙、地板、电梯、电气、空调、水、卫生
  - 其它专门为居住者提供服务的设施
- 架构师需要把所有的层次结合起来:
  - □ 使客户理解
  - □ 在建造的过程中为施工者提供指导
- 体系结构提供一个蓝图
- 体系结构不包括每个部分的细节

软件体系结构的别名

- 软件体系结构（Software Architecture）的同义词
  - 软件架构
  - 软件构架（不建议使用）
  - 简称：SA

软件体系结构的定义

- CMU-SEI在其网站上公开征集SA的定义,已有500多种不同的定义

软件体系结构的定义

- Bass，Clements 和 Kazman （1997）
  - SA是软件系统的结构(structure or structures),包含软件元素、软件元素外部可见的属性以及这些软件元素之间的关系
  - 外部可见的属性：包括提供的服务、性能特征、错误处理、共享资源的使用等。

对定义的理解

- **体系结构定义了一组软件元素**
  - 体系结构包含了系统由哪些元素构成，以及元素之间如何关联的信息
  - 体系结构只关注元素的外部特征（如接口），而不关注其内部实现。

- **软件系统可能有多个结构**
  - 从源代码组织的角度
  - 从运行时系统结构的角度
  - 从部署的角度

基于决策的软件体系结构

Software Architecture =

design (components & connectors)

\+ design decisions

Software Architecture Decisions

Grady Booch 
@Grady_Booch

Software architecture is all about decisions. Models are largely just the scaffolding we use to visualize, reason about, and document those decisions. Code is the medium whereby we make those decisions manifest.

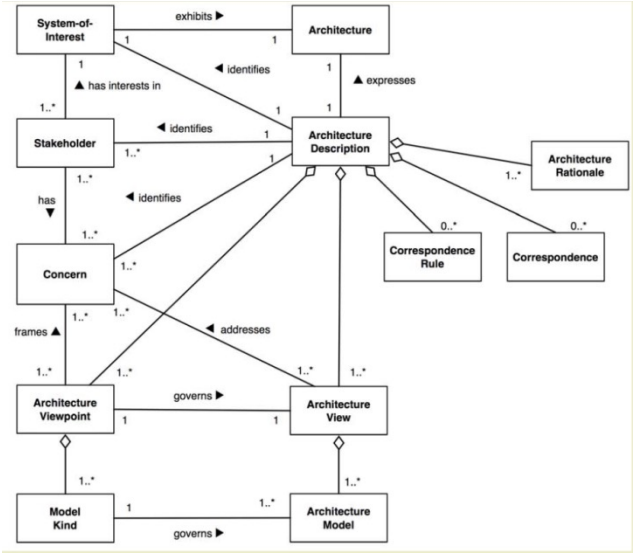
Every enduring system has all three of these, in varying quantity.

2:15 AM · Aug 1, 2020 · Twitter for iPad

ISO的SA定义

- ISO/IEC 42010: 2011
  - SA是软件系统的基本组织结构，包含构件、构件之间、构件与环境之间的关系,以及相关的设计与演化原则

ISO标准概念模型



哪些设计属于软件体系结构

- 需要耗费较大成本来更改的设计
- 一般是质量需求（以可用性为例）

课堂问答

- 识别软件体系结构设计案例中
  - System-of-Interest
  - Stakeholders
  - Concerns
  - Architecture Design Decisions (Rationale)
  - Architecture Description

实例

- **快速发展涉及很多问题。**我们是一家创业公司，业务发展非常快，可能准备不是很充分，比如说监控、日志、告警、框架、消息、数据库，很多基础设施还在建设之中。在这个过程中出现一些问题是在所难免的，对系统的要求不是不能挂、不能出问题，而是出了问题要第一时间能恢复。这是整个系统架构的前提。

实例

- 促销模式变化多端，可能每个月都会有变化，通常的面向接口编程和加上工厂方法或者依赖管理框架Spring也很难做到真正的解耦，虽然这样做已经符合开闭原则。我们通过观察者模式很好的解决了这个问题。让前端的页面模版和JAVA应用程序之间真正的解耦。

实例

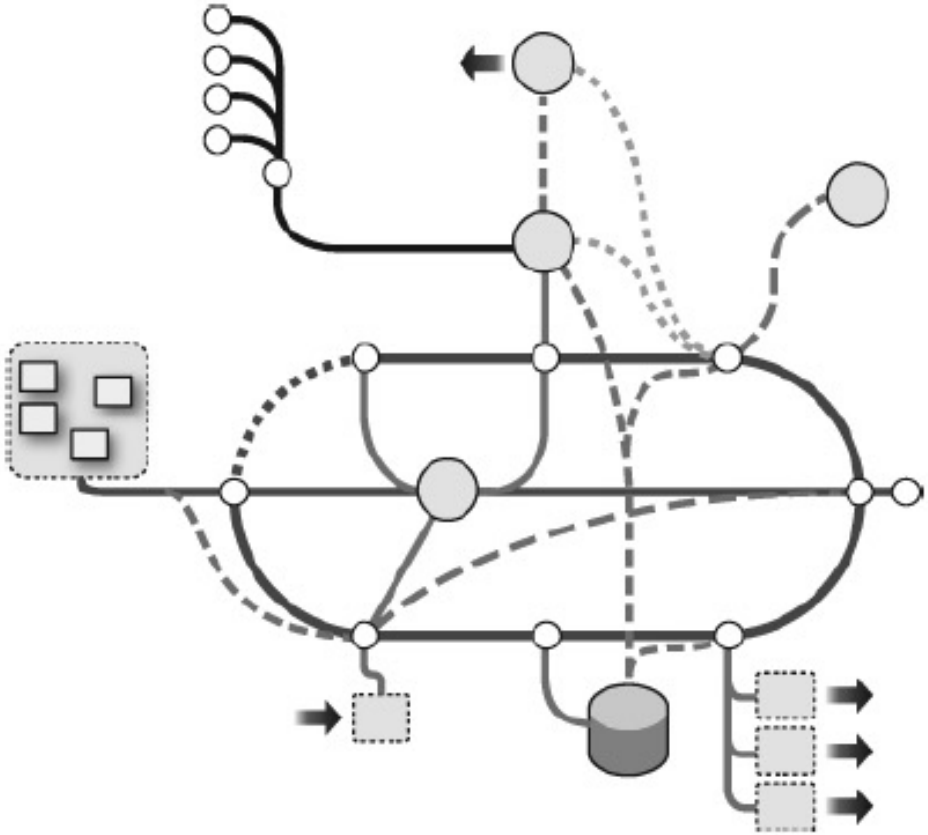
- 就订单系统而言主要有以下四个内容。第一是消息广播补偿，第二是主流程补偿，第三是灾备，第四是随机故障测试系统。

大纲

- 软件体系结构的发展史
- 什么是软件体系结构
- 软件体系结构的重要性
- 软件体系结构的常见术语
- 软件体系结构的经典案例

“混乱设计”和“良好设计”

- “混乱大都市”

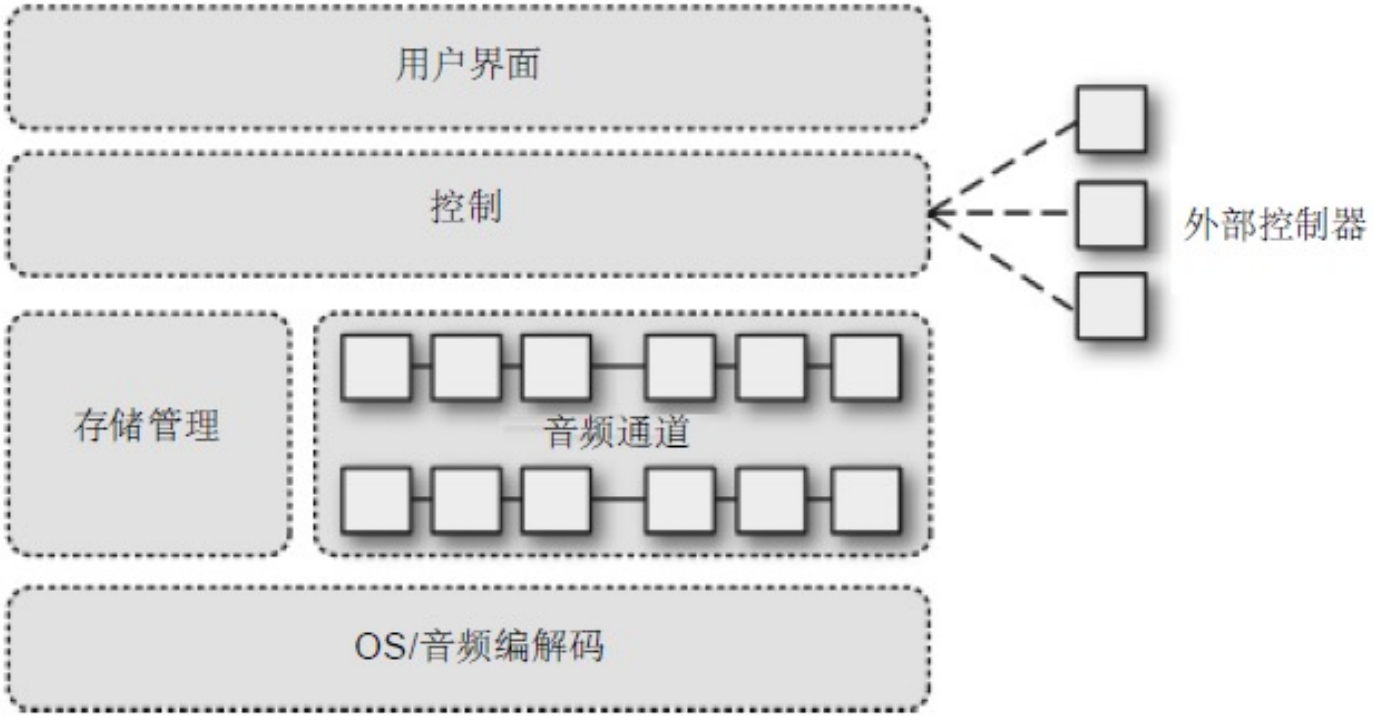


“混乱设计”带来的问题

- 无法预测
  - 系统行为
  - 开发成本
  - 发布周期
- 低质量
  - 难以修改
  - 难以维护
  - 修改导致新问题
- 难以重用
  - 难以模块化重用
  - 重用产品而非设计
- 难以变化
  - 适应需求变化
  - 花费更多时间

“良好设计”

- “设计之城”



软件体系结构的作用

（1）涉众（Stakeholder）进行沟通的手段

- □最终的架构设计的需要多方沟通和折衷
- □架构沟通的是关键的设计决定
- 不同的视图定位于不同涉众的关注点

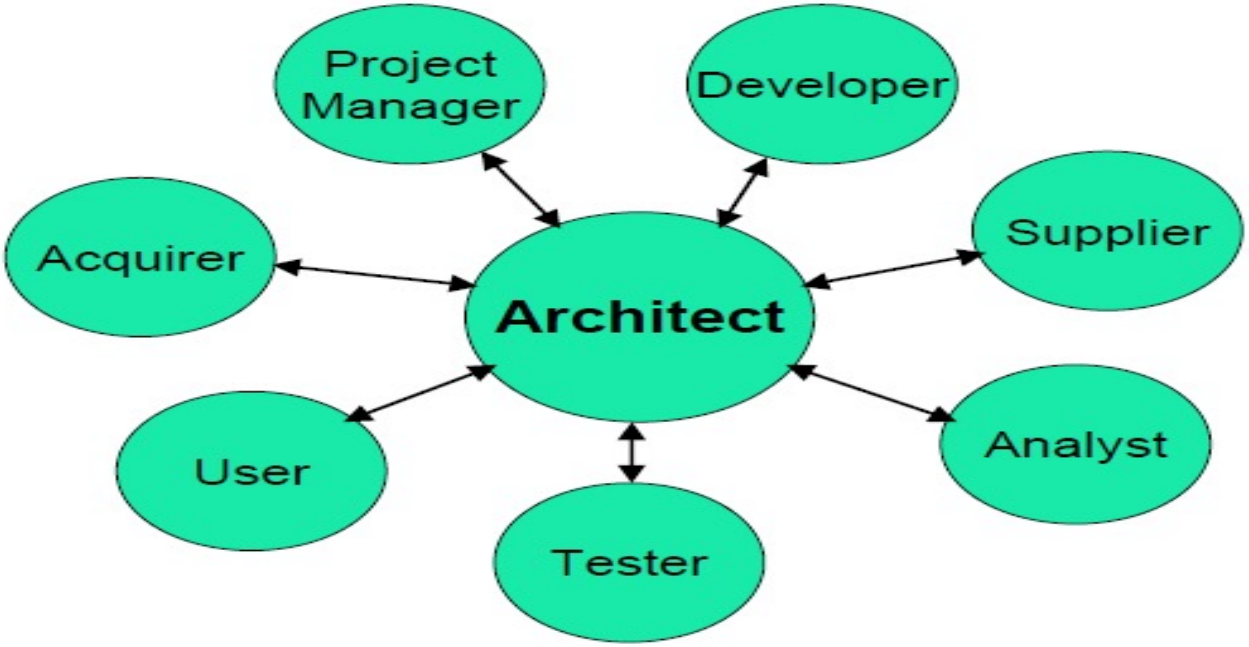
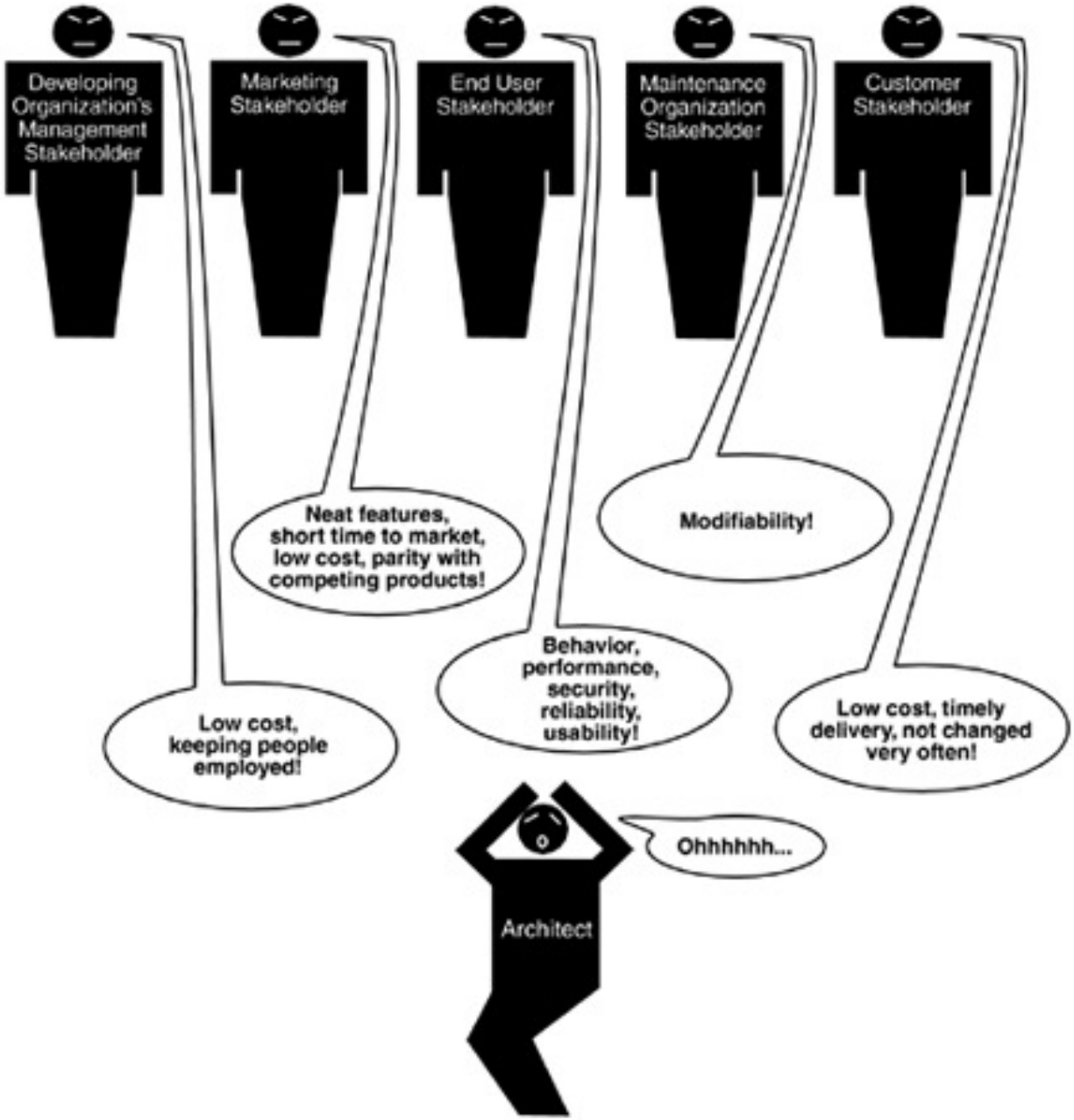


Figure 1.2. Influence of stakeholders on the architect



软件体系结构的作用

(2) 体系结构是早期设计决策的体现

- 软件体系结构明确了对系统实现的约束条件
  - ✓结构方面
  - ✓平台、能力、资源、环境等约束

- 软件体系结构决定了开发和维护组织的组织结构
  - ✓功能的分解，可以作为分工结构图（work breakdown structure）的基础依据
  - ✓进而决定计划、预算、团体交流、配置管理、文件管理、测试计划集成、团队组织等

体系结构的作用

(2) 体系结构是早期设计决策的体现

- 软件体系结构制约着系统的质量属性
- 通过评估软件体系结构可预测软件的质量

- Performance
- Availability
- Usability
- Security

End User's view

- Time To Market
- Cost and Benefits
- Projected life time
- Targeted Market
- Integration with Legacy System
- Roll back Schedule

Business Community view

- Maintainability
- Portability
- Reusability
- Testability

Developer's view

软件体系结构的作用

(2) 体系结构是早期设计决策的体现

- 软件体系结构使软件的变更管理更简单
  - ✓80%的软件成本产生在软件第一次成型之后
  - ✓三类变更： local, nonlocal, and architectural.
- 软件体系结构有助于进行原型设计
- 软件体系结构有助于更精确的进度和成功评估

软件体系结构的作用

（3）软件体系结构可以支持复用

- 产品线 (product line)
- 组件（库）
- 软件框架

（4）软件体系结构是需求和代码之间的桥梁，为开发提供了建设的蓝图，也是测试、维护和升级的依据。

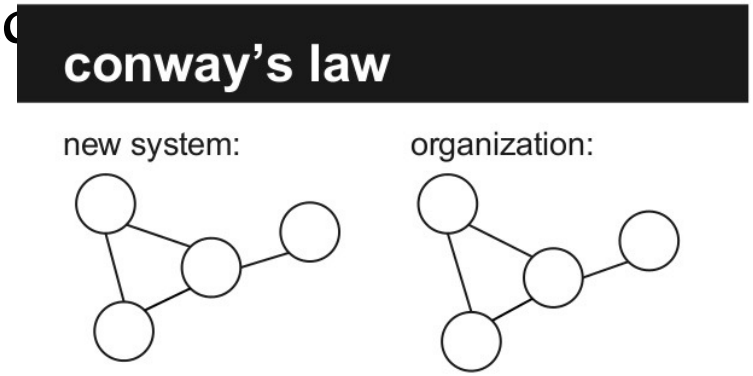
软件体系结构的作用

- Conway's law (1967)

Organizations which design systems are constrained

to produce designs which are copies of the

communico organi



大纲

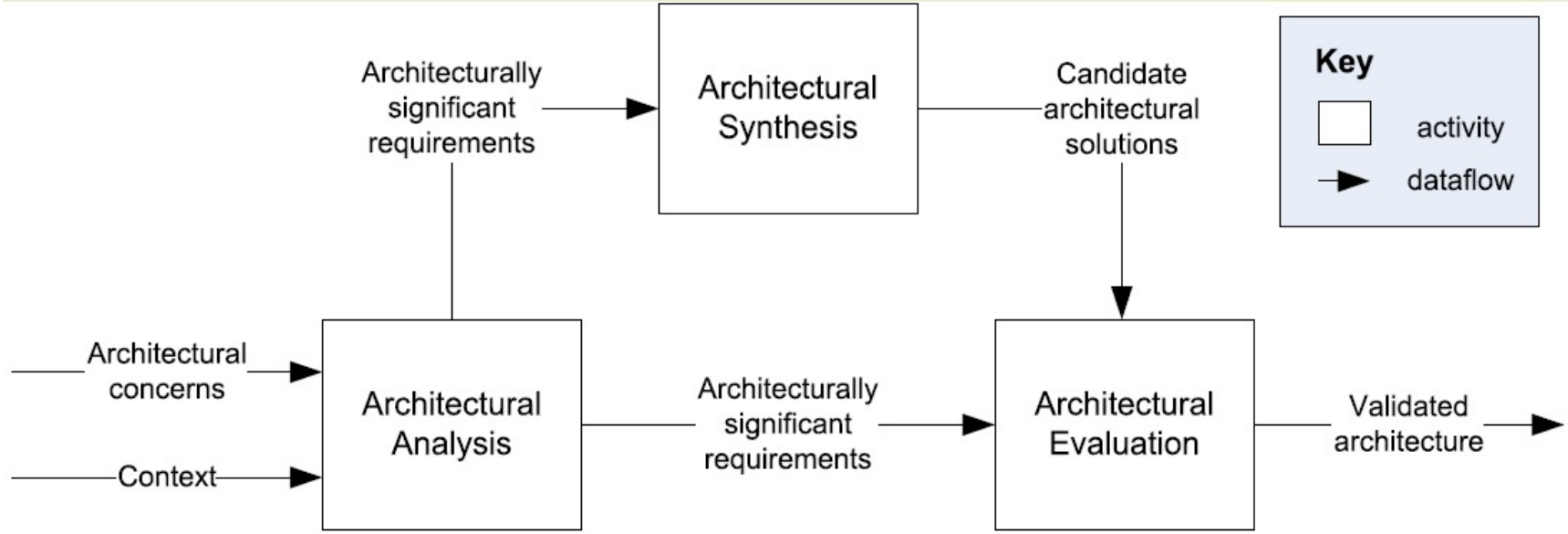
- 软件体系结构的发展史
- 什么是软件体系结构
- 软件体系结构的重要性
- 软件体系结构的术语简介

软件体系结构的术语

- 软件体系结构架构过程
- 软件体系结构描述语言
- 软件体系结构的建模
- 软件体系结构的分析
- 软件体系结构风格和模式
- 基于软件体系结构的软件开发方法
- 软件产品线体系结构

软件体系结构架构过程

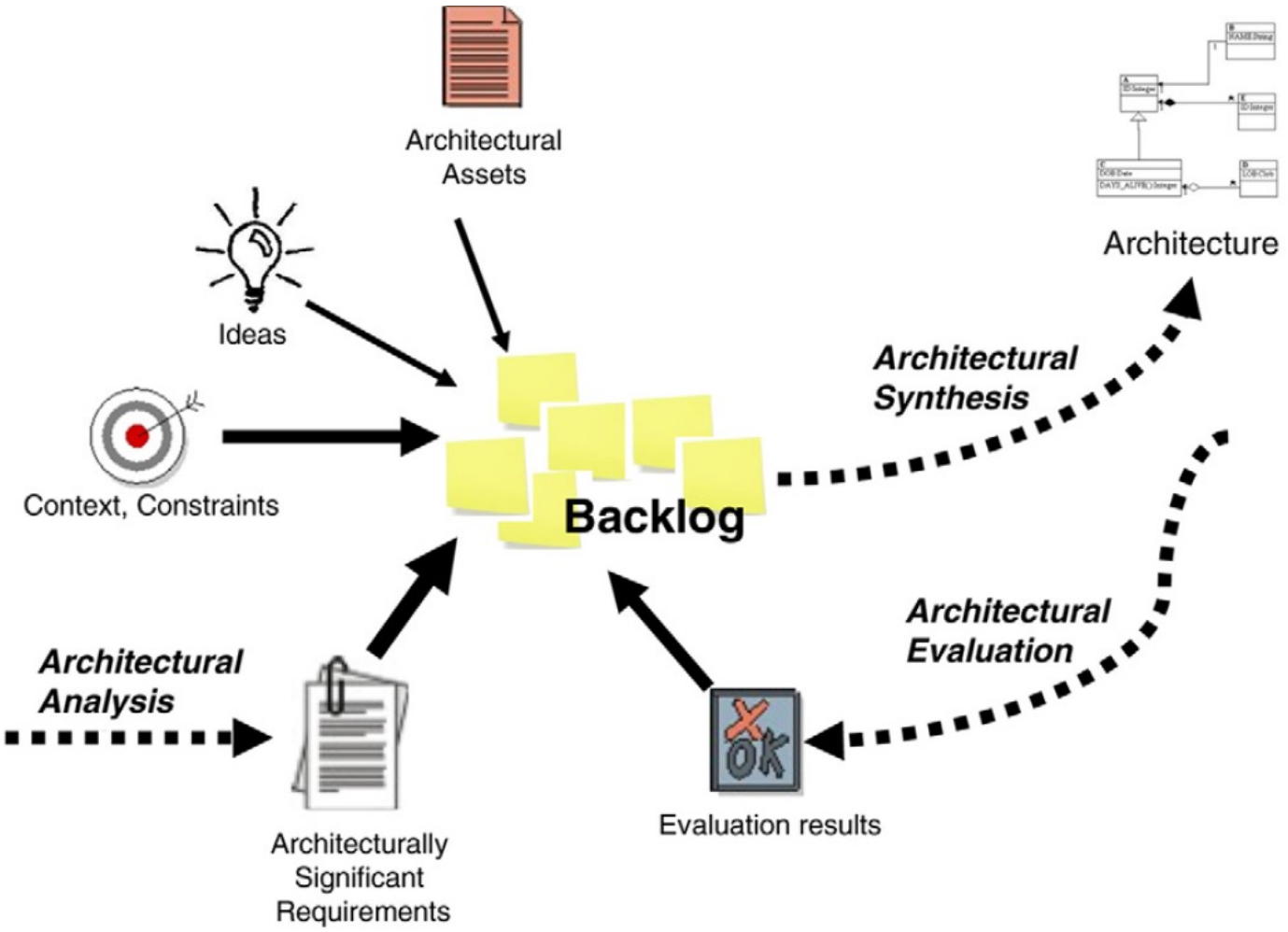
软件体系结构综合



软件体系结构分析

软件体系结构评价

软件体系结构架构过程

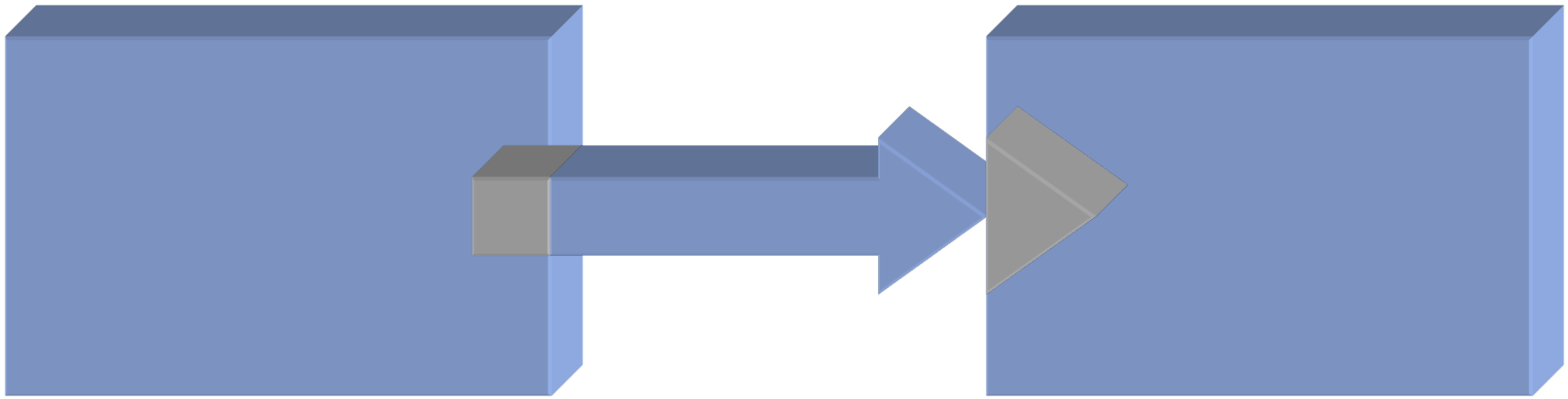

Architectural
Assets
Ideas
Context, Constraints
Backlog
Architecture
Architectural
Synthesis
Architectural
Evaluation
Architectural
Analysis
Architecturally
Significant
Requirements
Evaluation results

软件体系结构描述语言

- ADL (Architecture Description Language)
  - 早期主要使用**非形式化**的、用方框和连线描述体系结构的方法，不能清楚地表述设计的含义
  - 对体系结构进行描述和规范的方法；对体系结构的理论认识的**形式化描述**
  - 形式化理论的重点:通过形式化语言，对结构设计给出准确的描述
  - ADL为软件体系结构特征既提供了概念性框架，也提供了描述的具体语法，同时提供解析、显示、编译、分析或者仿真体系结构描述的工具

ADL中的体系结构元素

- ADL通常需要描述如下元素
  - 组件components
  - 连接件connectors
  - 配置configurations
  - 约束constraints



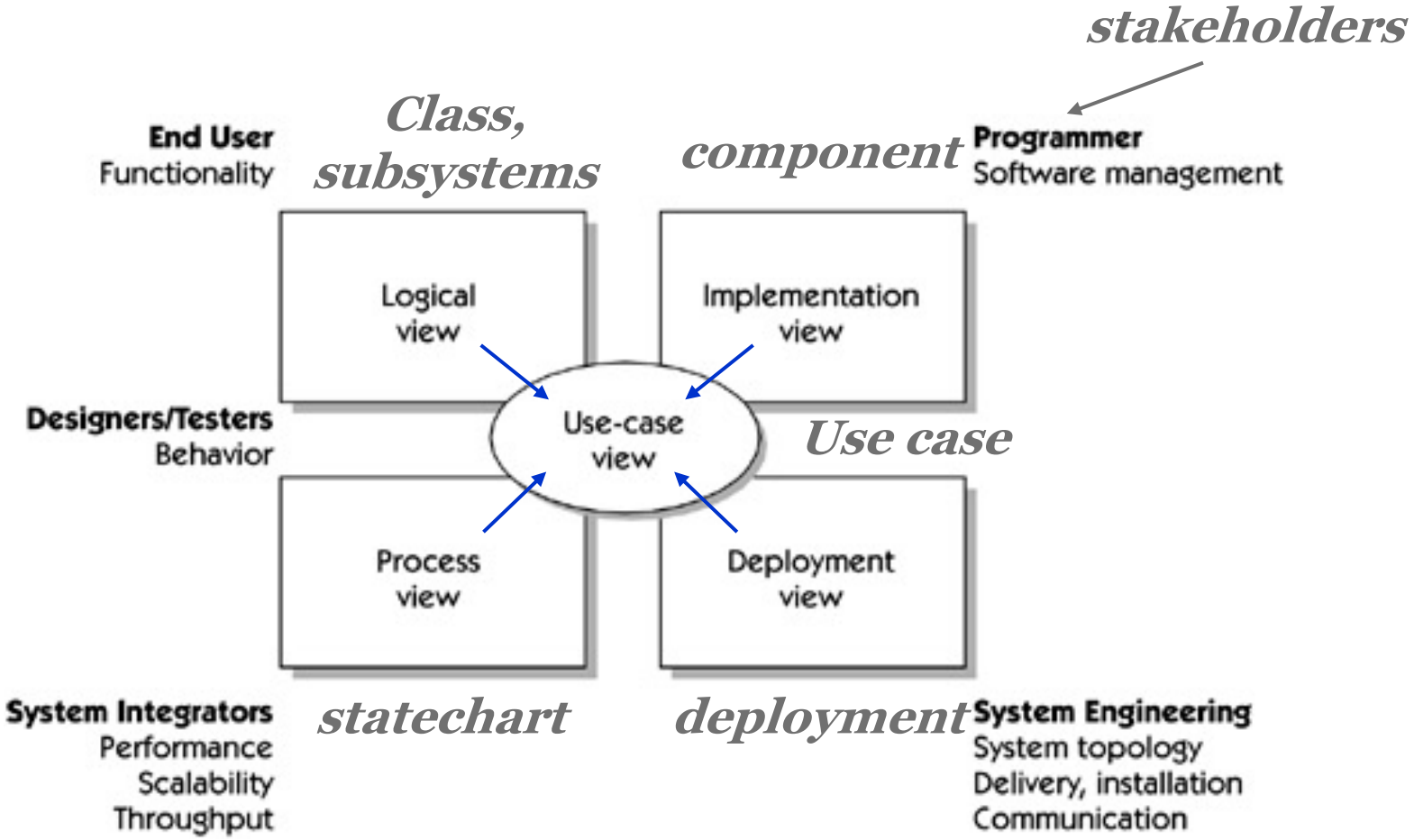
常见的ADL

- Leading candidates
  - ACME (CMU/USC)
  - Rapide (Stanford)
  - Wright (CMU)
  - Unicon (CMU)
- Secondary candidates
  - Aesop (CMU)
  - MetaH (Honeywell)
  - C2 SADL (UCI)
  - SADL (SRI)
- Others
  - Lileanna
  - UML
  - Modechart

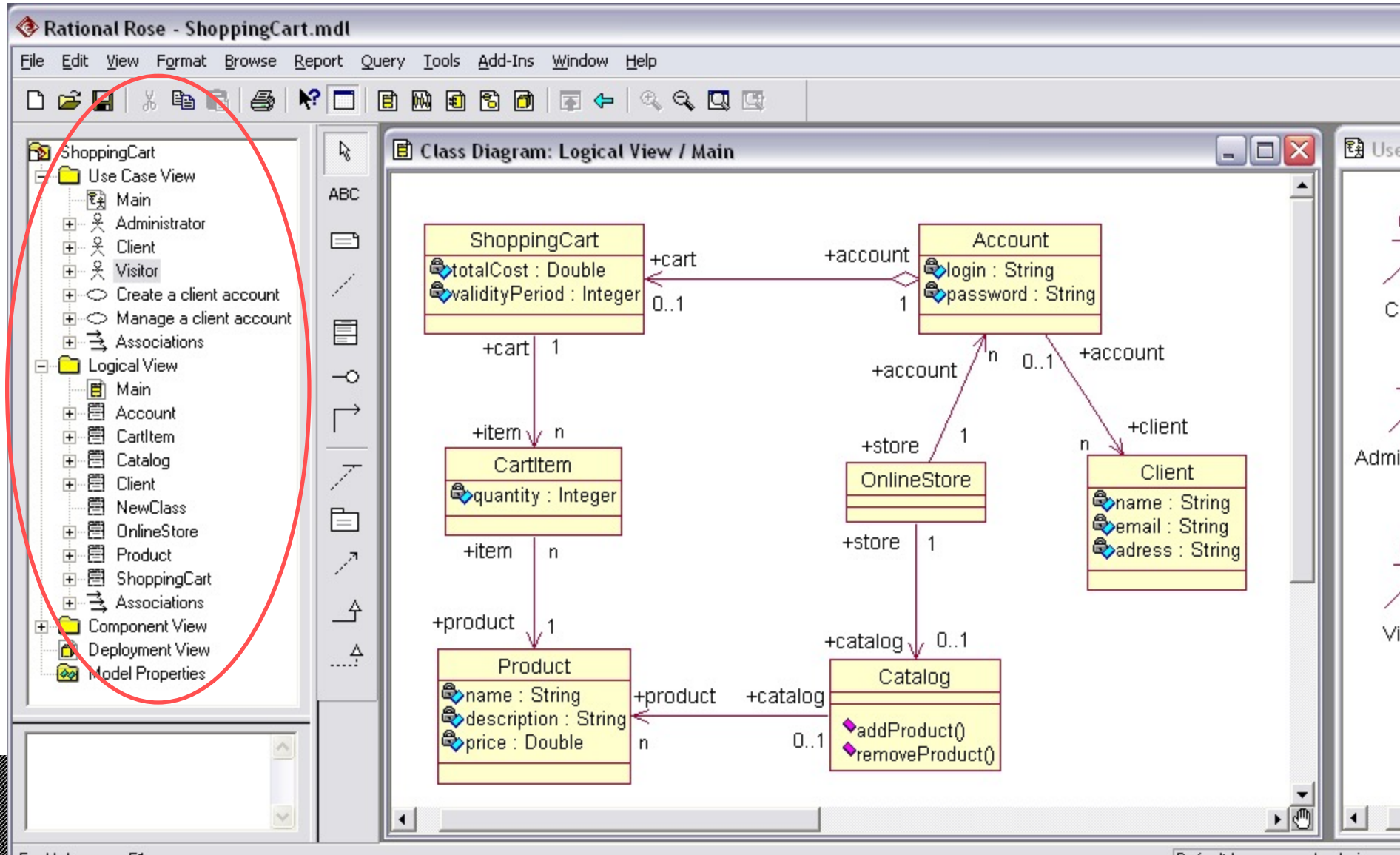
软件体系结构建模

- 按照一定的描述方法，用体系结构描述语言对体系结构进行说明的结果则称为体系结构的模型，将描述体系结构的过程称为体系结构建模
-
  - Kruchten提出的“4+1”体系结构视图。
  - Booch从UML的角度给出了一种由设计视图、过程视图、实现视图和部署视图，再加上一个用例视图构成的体系结构描述模型。
  - Rational从资产重用的角度提出了体系结构描述的规格说明框架。

4+1体系结构视图


stakeholders
Class,
subsystems
component
End User
Functionality
Programmer
Software management
Logical
view
Implementation
view
Designers/Testers
Behavior
Use-case
view
Use case
Process
view
Deployment
view
System Integrators
Performance
Scalability
Throughput
statechart
deployment
System Engineering
System topology
Delivery, installation
Communication

4+1体系结构视图工具(Rationale Rose)



软件体系结构分析方法

- 体系结构分析
  - 通过分析SA设计所产生的模型,预测系统的质量属性并界定潜在的风险

(1) 基于形式化方法、数学模型的分析方法

- ✓基于进程代数、CHAM(chemical abstraction machine)、有穷状态自动机、LTS(labeled transition systems)等分析SA模型中是否包含死锁
- ✓基于排队论模型分析SA模型的性能
- ✓基于马尔科夫模型分析系统的有效性

软件体系结构分析方法

(2) 基于调查问卷、场景分析、检查表的分析

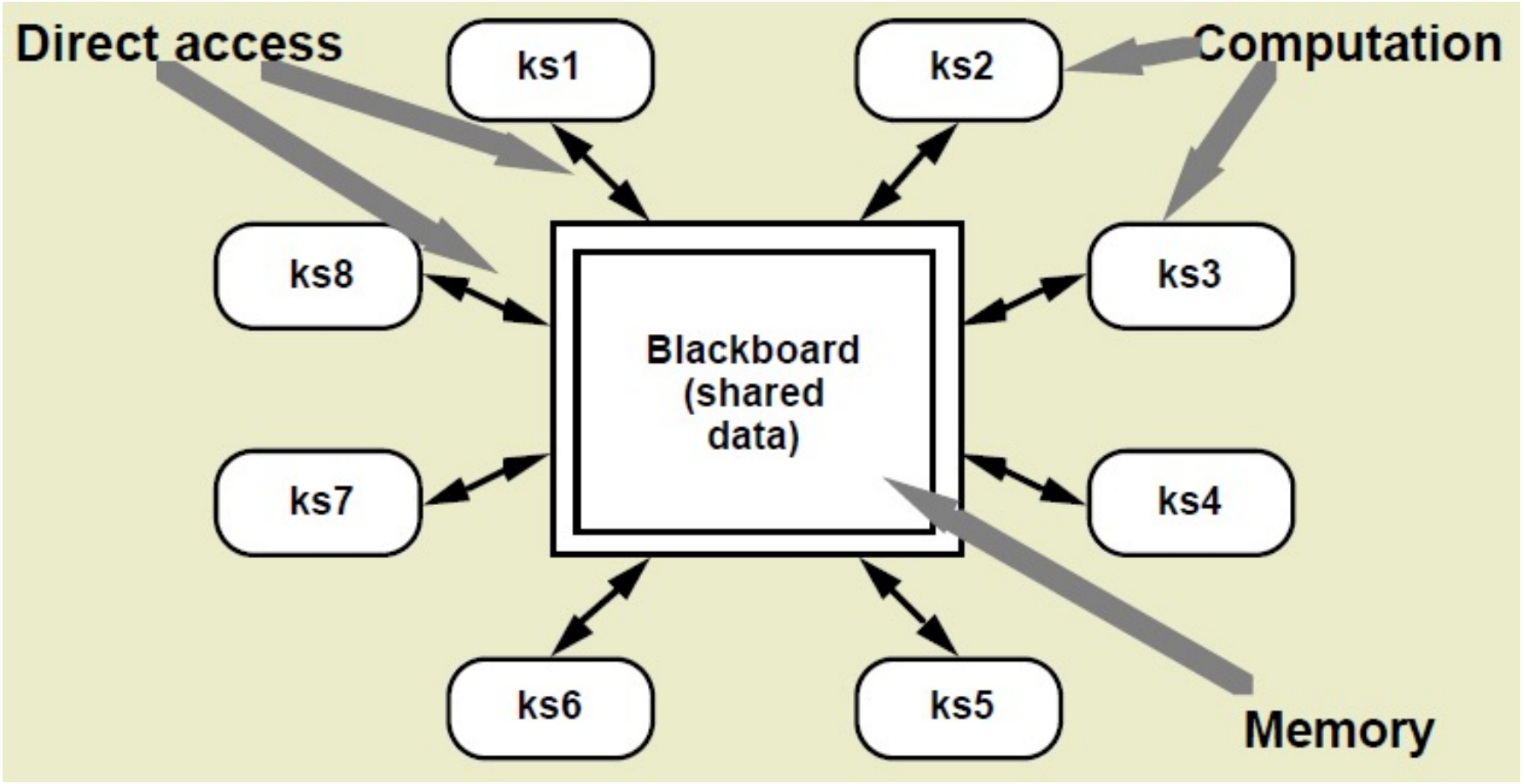
- 侧重得出关于SA可维护性、可演化性、可复用性等难以量化的质量属性
- 强调SA的各类涉众stakeholder)的参与,往往是手工完成
- 典型的有
  - ✓基于场景的体系结构分析方法SAAM及其3个扩展方法(针对复杂场景的扩展SAAMCS、针对可复用性的扩展ESAAMI和SAAMER)
  - ✓体系结构权衡分析方法ATAM
  - ✓基于场景的体系结构再工程方法SBAR
  - ✓体系结构层次软件可维护性预测方法ALPSM
  - ✓SA评估模型SAEM

软件体系结构风格（软件体系结构模式）

- 描述某一特定应用领域中系统组织方式的惯用模式,作为“可复用的组织模式和习语”为设计人员的交流提供了公共的术语空间,促进了设计复用与代码复用.
- 对设计模式的扩展,描述了软件系统基本的结构化组织方案,可以作为具体SA的模板

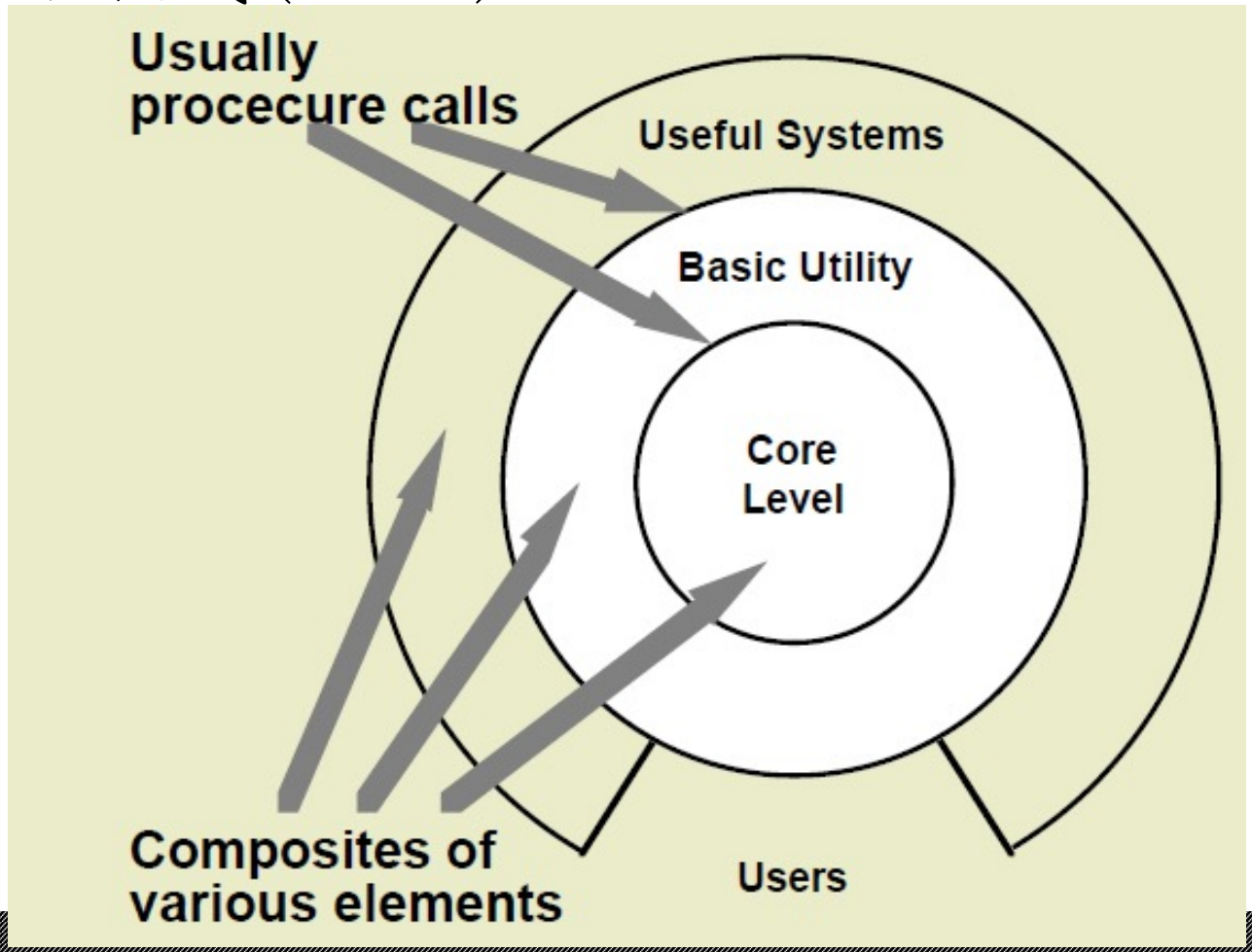
软件体系结构模式实例

- 黑板模式(call-center)



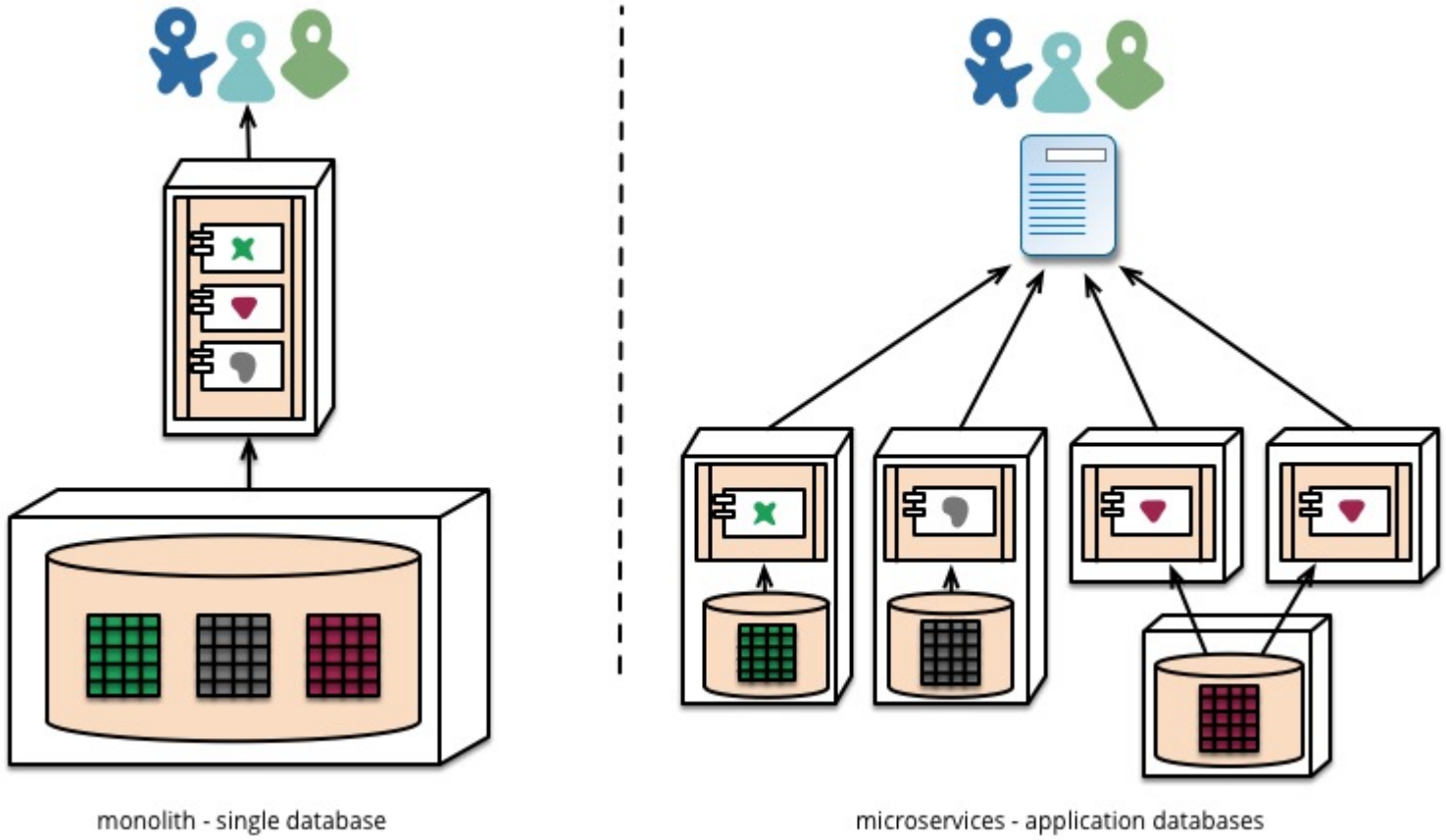
软件体系结构模式实例

- 分层模式 (Linux)



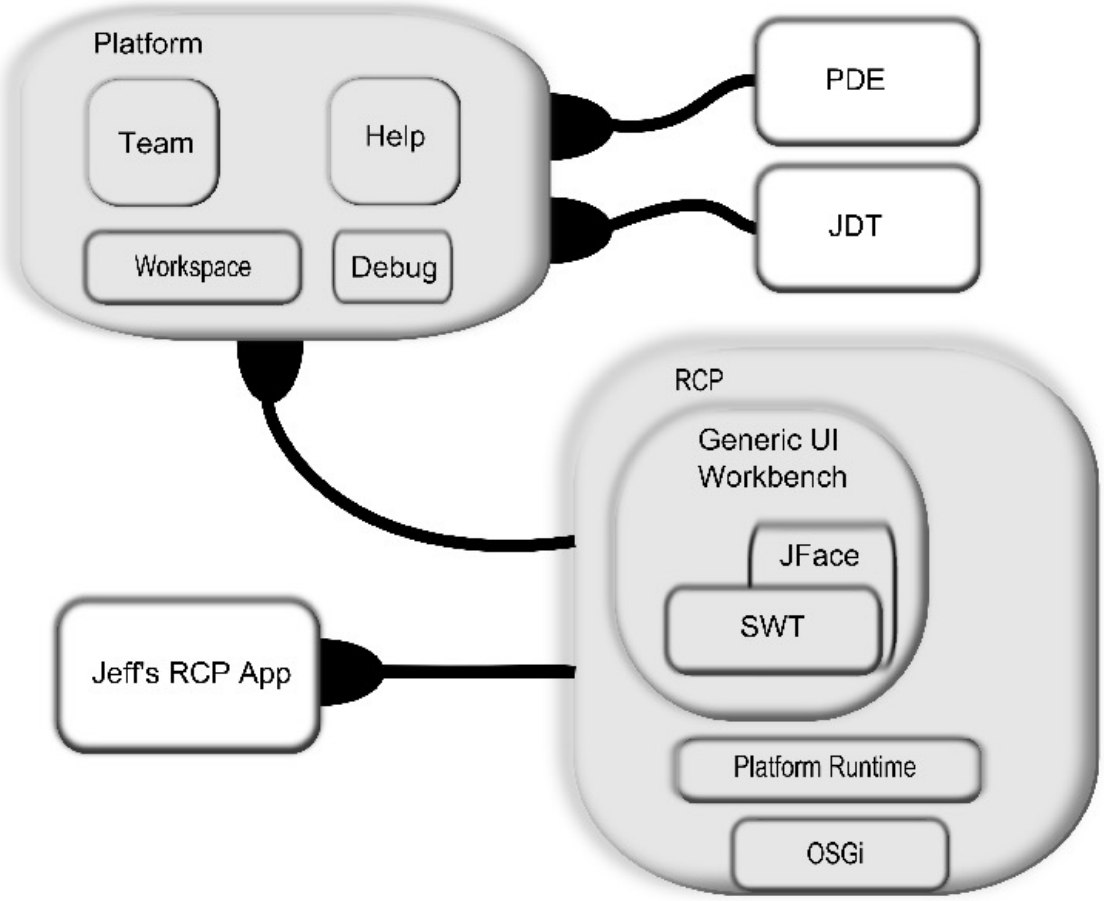
软件体系结构模式实例

- 微服务架构-Amazon AWS



软件体系结构模式实例

- 插件式架构-Eclipse



基于体系结构的软件开发方法

- 在将体系结构引入软件开发之后，应用系统的构造过程变为“问题定义—>软件需求—>软件体系结构—>软件详细设计—>软件实现”
- RUP：用例驱动，以体系结构为中心，迭代式开发过程
  - 以体系结构为中心
    - ✓早期把全局结构基本稳定
    - ✓避免重大风险
    - ✓经验丰富的高层次人员完成架构设计

产品线体系结构

- 领域工程和应用工程
- 需求重用与可变性建模
- 特定领域软件体系结构
- 嵌入式设备、操作系统

产品线体系结构

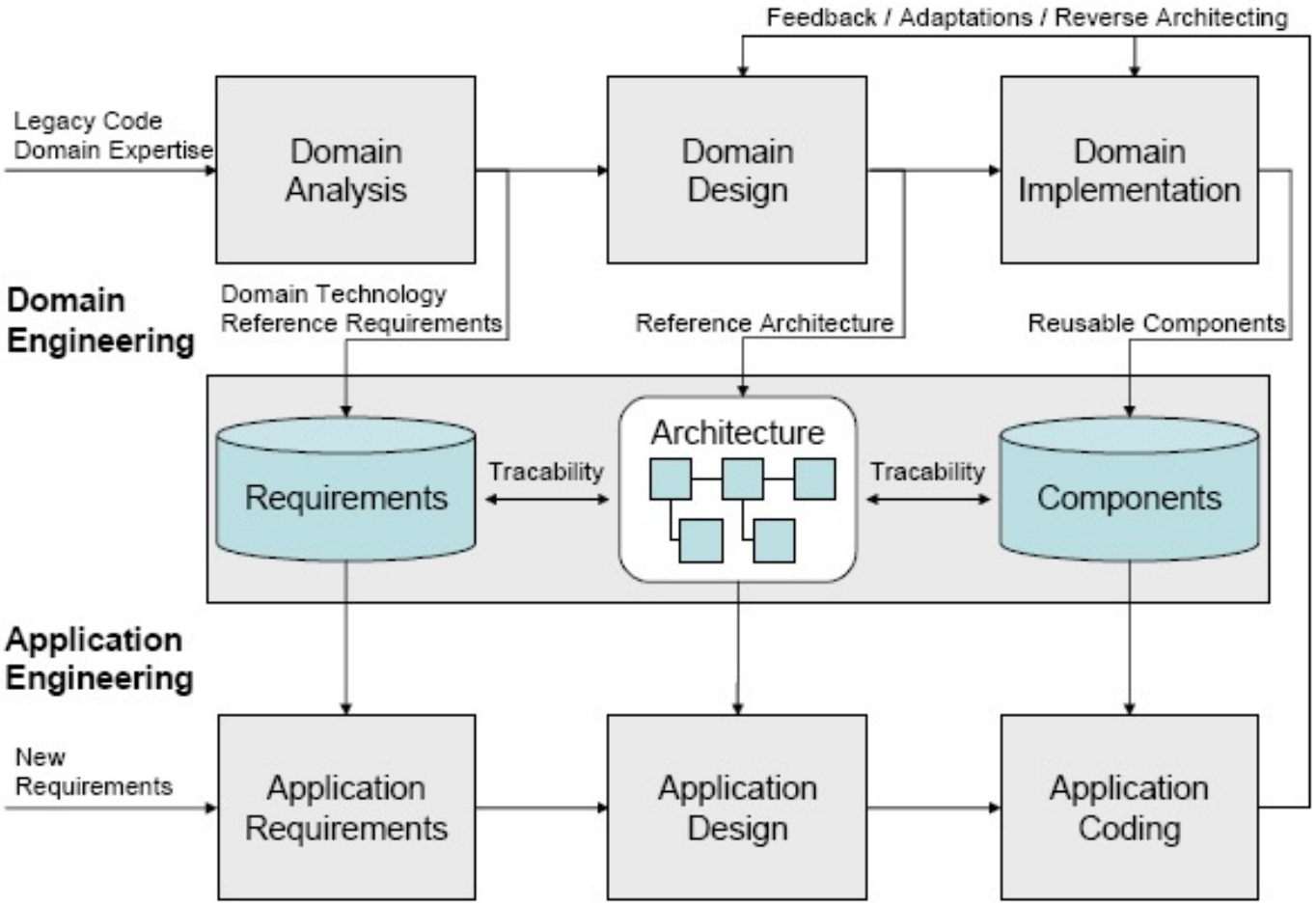


Figure 1. The ESAPS reference process according to van der Linden (2002).

总结

- 什么是软件体系结构
- 软件体系结构的组成
- 软件体系结构决策
- 软件体系结构活动和过程
- 软件体系结构模式