SCA 和 SDO 简介：如何构建业务应用程序

Jeff Mischkinsky
Director, Oracle Fusion Middleware and Web Services Standards

OSOA Collaboration | 2007 年 5 月 30 日

提纲

- SOA——业务推进器
- SOA 远景——模型和特征
- SCA/SDO——概览
- SCA——细节和示例
- 总结

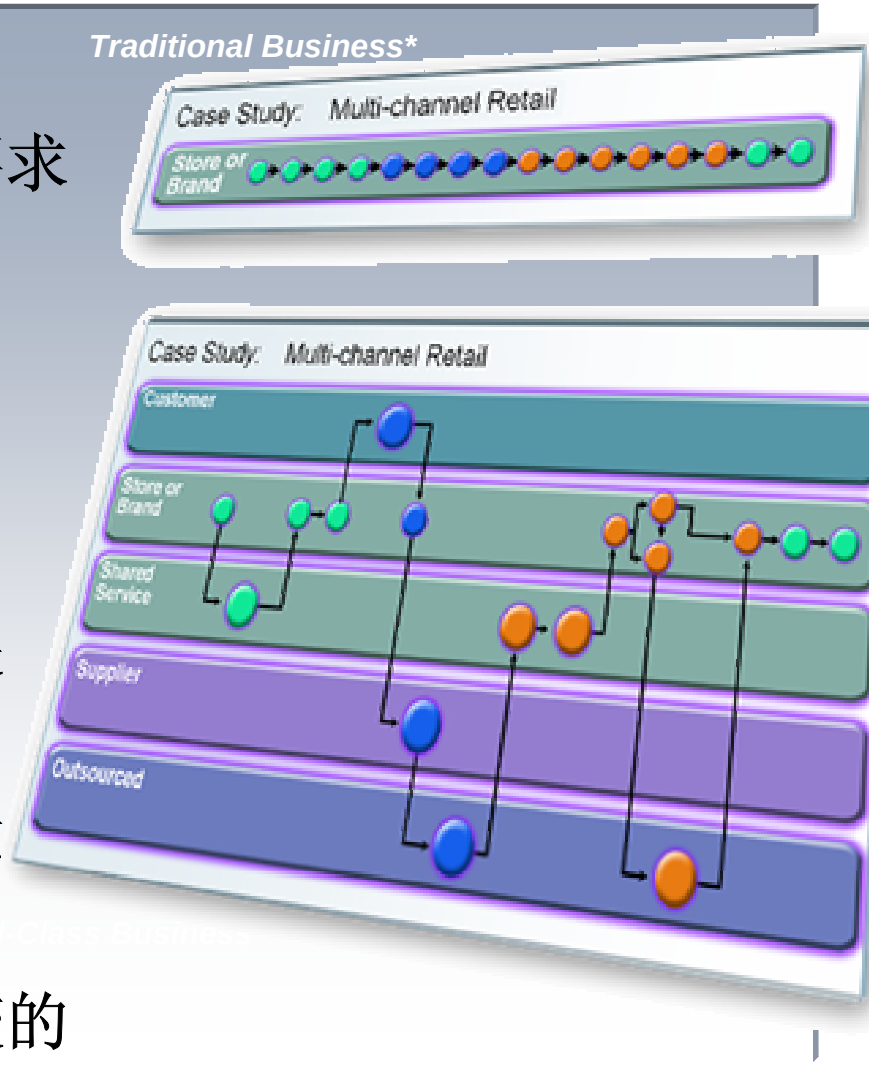
SOA——业务推进器

OSOA Collaboration | 2007 年 5 月 30 日

业务推进器

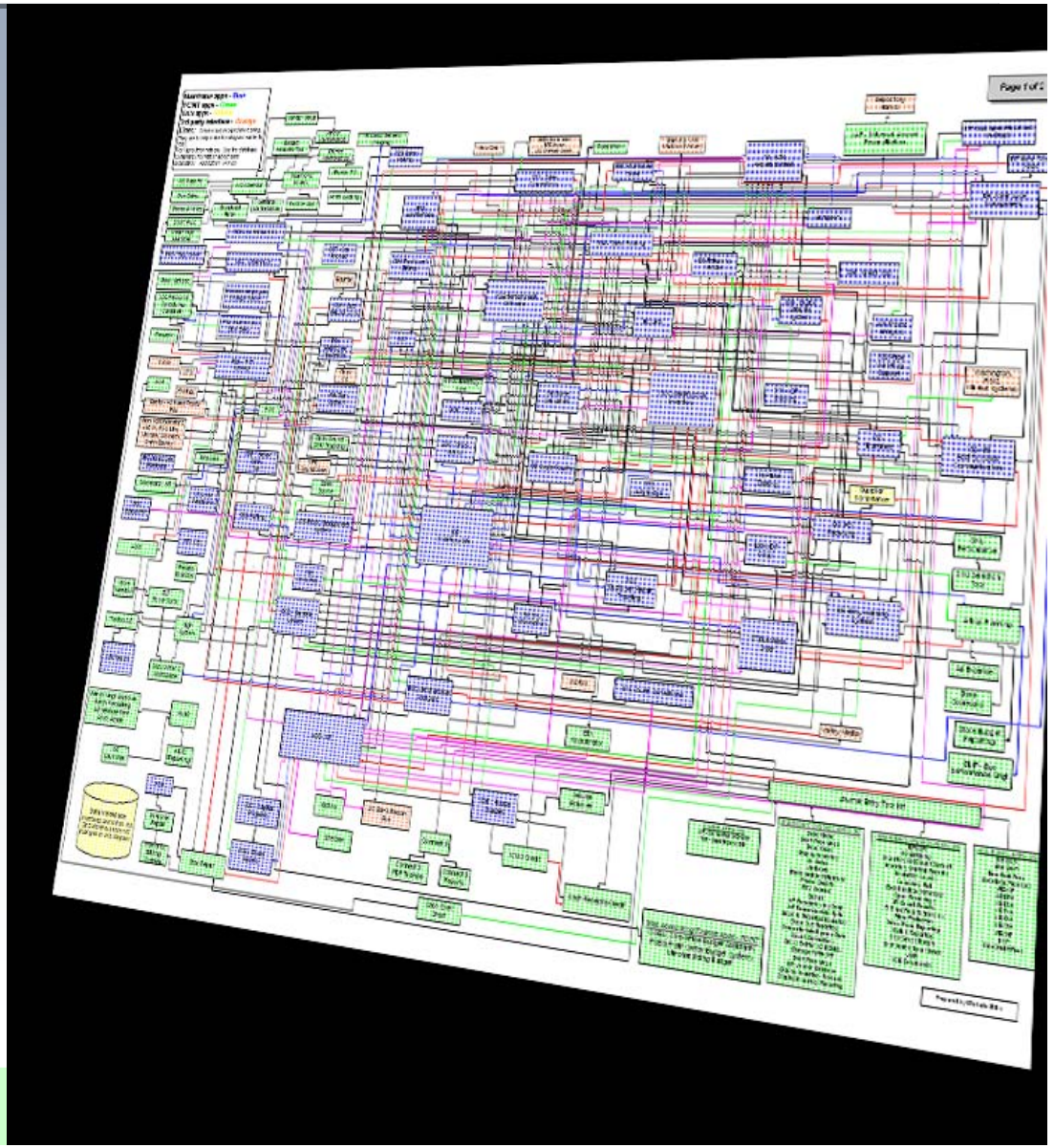
灵活的业务需要灵活的 IT

- 从经济学的角度来看：全球化要求更大的灵活性
- 全球供应链集成
- 业务流程：
  每日变更与每年变更
- 通过获得灵活性促进业务增长是 CEO 议程的头顶大事
- 可复用的资产可以削减成本多达 20%
- 灵活性至关重要，成为随需应变的企业

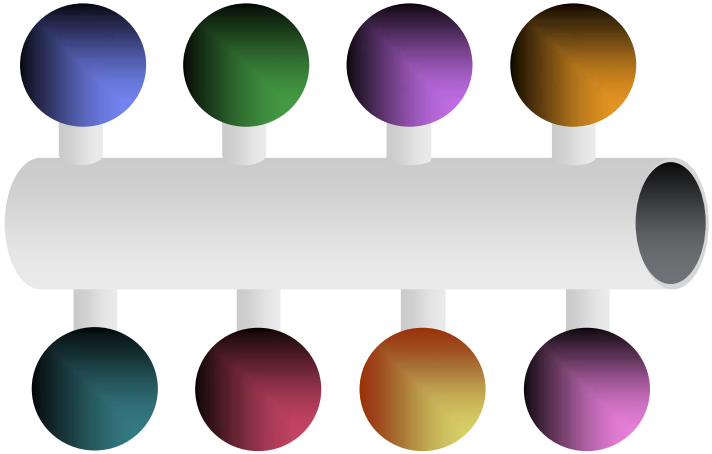


当今我们碰到的问题

- 复杂性
- 僵硬、脆弱的架构
- 无法促进业务增长



我们想要得到什么



+

- 与业务级别的语义之间具有定义良好的接口
- 标准化的通信协议
- 灵活地重新组合服务，以增强软件灵活性

面向服务的架构 **(SOA)** 是获得灵活性和降低复杂性的关键技术之一

SOA 远景：
模型和特征

OSOA Collaboration | 2007 年 5 月 30 日

SOA 的特征

- 面向业务职能的服务模型
  - 服务是粗粒度的和面向网络的
- 特征
  - 灵活性
    - 自治的服务是高度可复用的
      不使用预测模式
  - 效率
    - 高级别的抽象
  - 理解
    - 经过良好理解的系统架构和行为

SOA 编程模型 (1)

- SOA 编程模型源自*服务*的基本概念：
  - 服务是一个封装了软件功能的抽象体
  - *开发人员*构建服务、使用服务和开发聚合服务的解决方案
  - 将服务*组合*为集成*解决方案*是一项关键活动

SOA 编程模型 (2)

- 核心元素：
  - *服务组装 (Service Assembly)*

    与技术和语言无关的服务复合表示法
  - *服务构件 (Service Components)*

    与技术和语言无关的、可复合的服务实现表示法
  - *服务数据对象 (Service Data Objects)*

    与技术和语言无关的服务数据项表示法

SCA/SDO：概览

OSOA Collaboration | 2007 年 5 月 30 日

服务构件架构 (SCA)：一种简化的 SOA 编程/部署模型

- 一种用于制作构件，将构件组装为应用程序，再部署到各种运行环境中的模型
  - 构件可以*按照 SOA 规则用新的或既有的代码*制成
  - *供应商无关*——受到整个行业的支持
  - *语言无关*——使用任何语言编写的构件
  - *技术无关*——可以使用任何通信协议和基础设施连接构件

什么是 SCA？

- 一种 *可执行的* 模型，用于将服务构件组装成业务解决方案

- 面向服务实现的简化的 *构件编程模型：*
  - 业务服务可以采用各种技术来实现
    例如，EJB、Java POJO、BPEL 进程、COBOL、C++、PHP ……

SCA：不是什么

- 不建模单独的*工作流*
  - 使用 BPEL 或其他工作流语言
- 不是 *Web service*
  - SCA 可以使用 Web service，但也可以构建不带任何 Web service 内容的解决方案
- 不局限于特定的运行时环境
  - 分布式、异构性、大、小
- 不强制使用特定的编程语言和技术
  - 目标是欢迎使用各种语言和技术

SCA 的关键获益

- *松散耦合：*无需知道其他构件的实现方式，即可实现构件集成
- *灵活性：*构件可以很容易地被其他构件所替代
- *服务* 可以很容易地被调用，既可以是同步调用，也可是异步调用
- 解决方案的*复合*：前面已介绍
- *效率：*易于集成构件，以形成复合应用程序
- *异构性：*
  - 多种实现语言 / 不同的框架
  - 多种通信机制
- 基础设施服务的*声明式* 应用程序
  - WSDL 是契约语言
- 为所有开发人员、集成人员和应用程序部署人员带来*简化* 的开发体验

SCA——高级别的视图

- *统一的声明式模型 (Unified declarative model)* 用于描述服务组装
  - 依赖性分析和配置
  - 用于基础设施服务的声明式策略
    - 安全，事务，消息传递
- *业务级别的模型 (Business-level model)* 用于实现服务
  - 具有服务接口的服务构件
  - 没有任何技术性 API，如 JDBC™、JCA™、JMS™……
- *绑定模型 (Binding model)* 用于多种访问方法和基础设施服务
  - WSDL、SOAP over HTTP、JMS™/messaging、Java™ RMI/IIOP……
- *交互模型 (Interaction Model)* 用于连接和断开服务
  - 同步、异步和会话式服务关系

服务数据对象 (SDO)：简化的 SOA 数据处理

- 用于访问业务数据的简化编程模型
  - 统一的数据格式模型
  - 统一的数据访问模型
- 与数据形式无关（不管是源数据还是目标数据）
  - 以各种格式存储的数据
  - RDBMS
  - XML 格式
  - 非结构化的数据
- 为 SOA 设计的交互方式
  - 离散的、乐观更新的策略
    - 读取、操纵、更新

SCA：
细节和示例

OSOA Collaboration | 2007 年 5 月 30 日

SCA 元素

- *组装* 模型
  - 如何定义复合应用程序的结构
- *客户端和实现* 规范
  - 如何以特定的语言编写业务服务
  - Java、C++、BPEL、PHP……
- *绑定* 规范
  - 如何使用访问方法
  - Web service、JMS、RMI-IIOP、REST……
- *策略框架*
  - 如何将基础设施服务添加到解决方案
  - 安全、事务、可靠消息传递……

基本的组装元素

- *构件*
  - *配置好的实现* 的实例
  - *提供* 和 *使用服务*
  - 设置实现的 *属性*
- *复合体*
  - *组合构件* 的集合
  - *连线* 引用和服务
  - 选择 *绑定*、端点
  - 应用 *策略*

SCA 绑定

- 针对特定情况：
  - 访问方法 / 协议 / 传输
  - 序列化
  - 框架
- 应用到服务和引用
- 通常在部署期间添加
- 当前定义的绑定：
  - Web service 绑定
  - JMS 绑定
  - JCA 绑定
  - EJB (RMI-IIOP) 绑定

SCA 客户端和实现规范

- 指定如何构建服务构件和服务客户端
- 针对特定的语言或框架，或者针对特定于语言或框架的 API
- 可扩展的
- 当前已定义的 SCA 客户端和实现规范：
  - BPEL
  - Java
  - Spring Framework
  - EJB
  - JAX-WS
  - C++
  - (PHP)

SCA 策略和基础设施能力

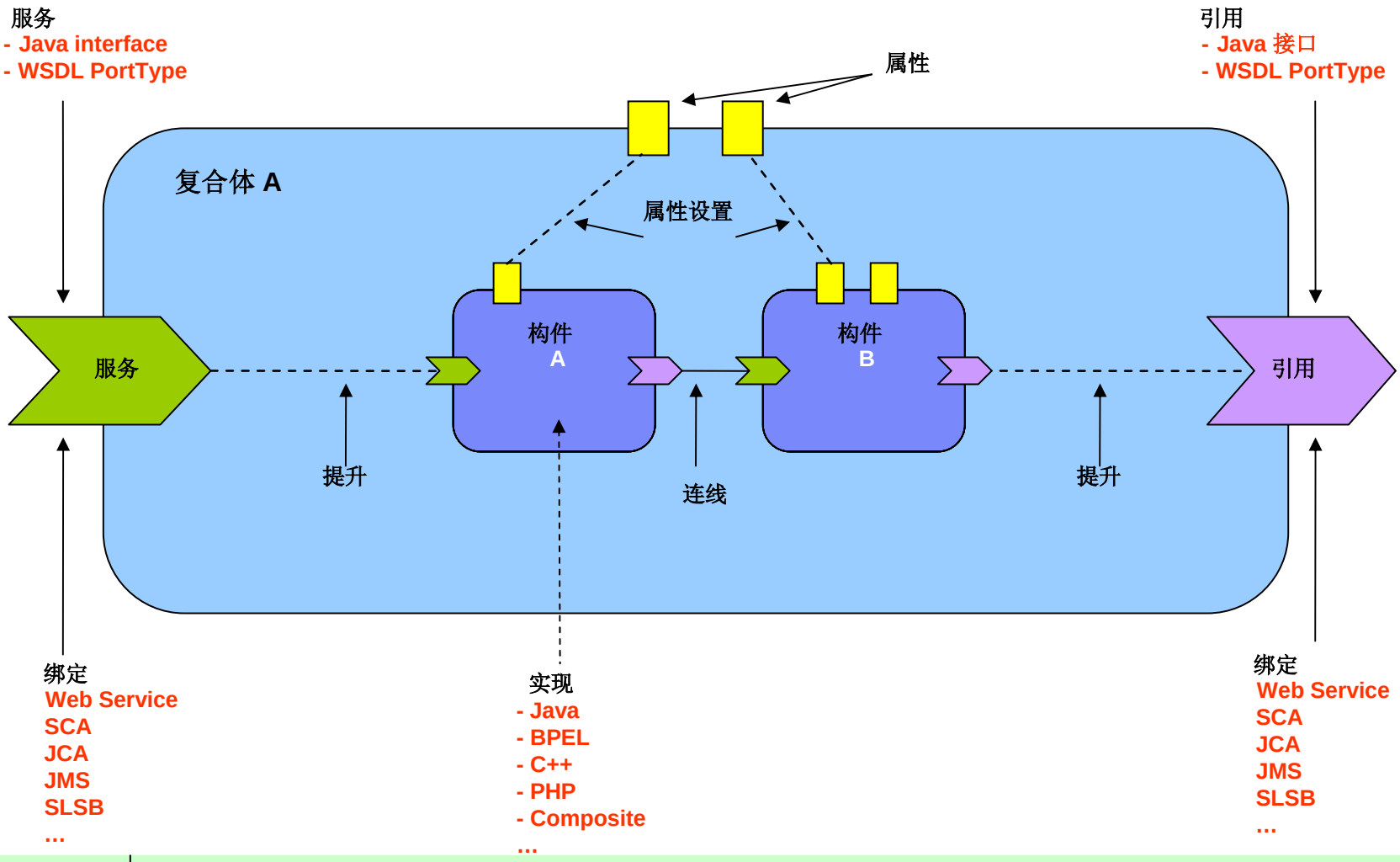
- *基础设施 (Infrastructure)* 有许多可配置的能力
  - 安全：身份验证和授权
  - 安全： 隐私、加密、不可抵赖
  - 事务、可靠消息传递，等等
  - 跨多个关注点域的复杂配置集合

- SCA 利用*声明式模型*抽象掉复杂性
  - 不受实现代码的影响
  - 通过声明式策略意图简化了使用
  - 易于应用和修改
  - 在策略集中保存了复杂的细节

SCA 策略框架

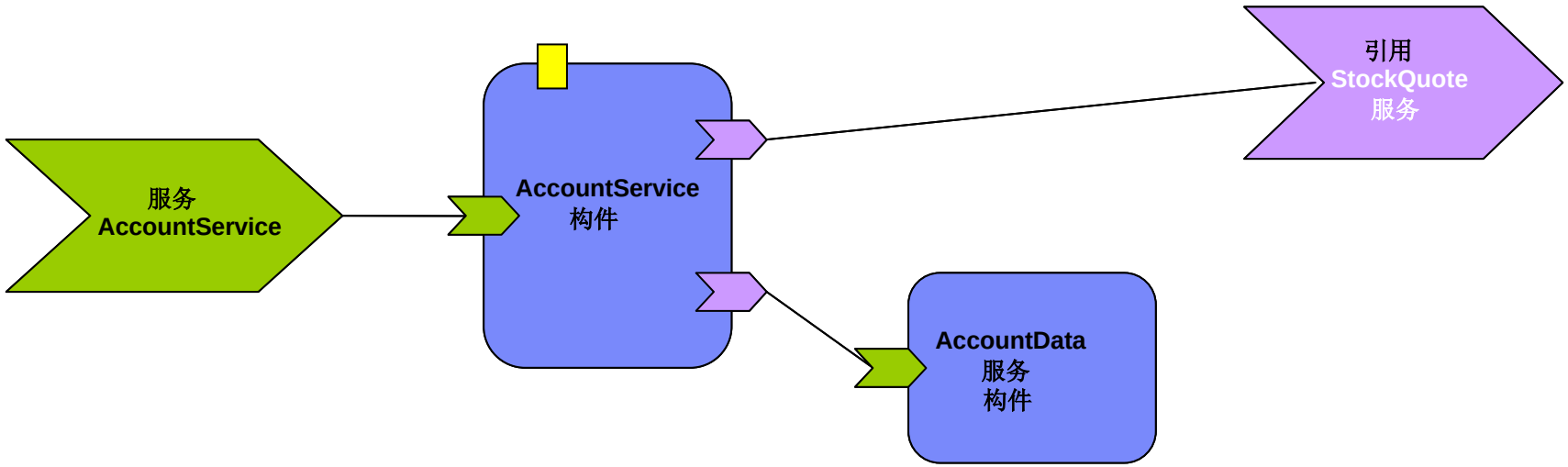
- 框架由以下部分组成：
  - SCA 策略*意图*
    - 每个 SCA 策略意图都表示一个单独的抽象 QoS 意图
    - 例如，可靠消息传递
  - SCA *策略集*
    - 表示一组具体的策略，用于实现某个抽象的 QoS 意图
  - WS-Policy
    - 用于策略集中具体策略的语法
    - 其他可能……

SCA 复合体

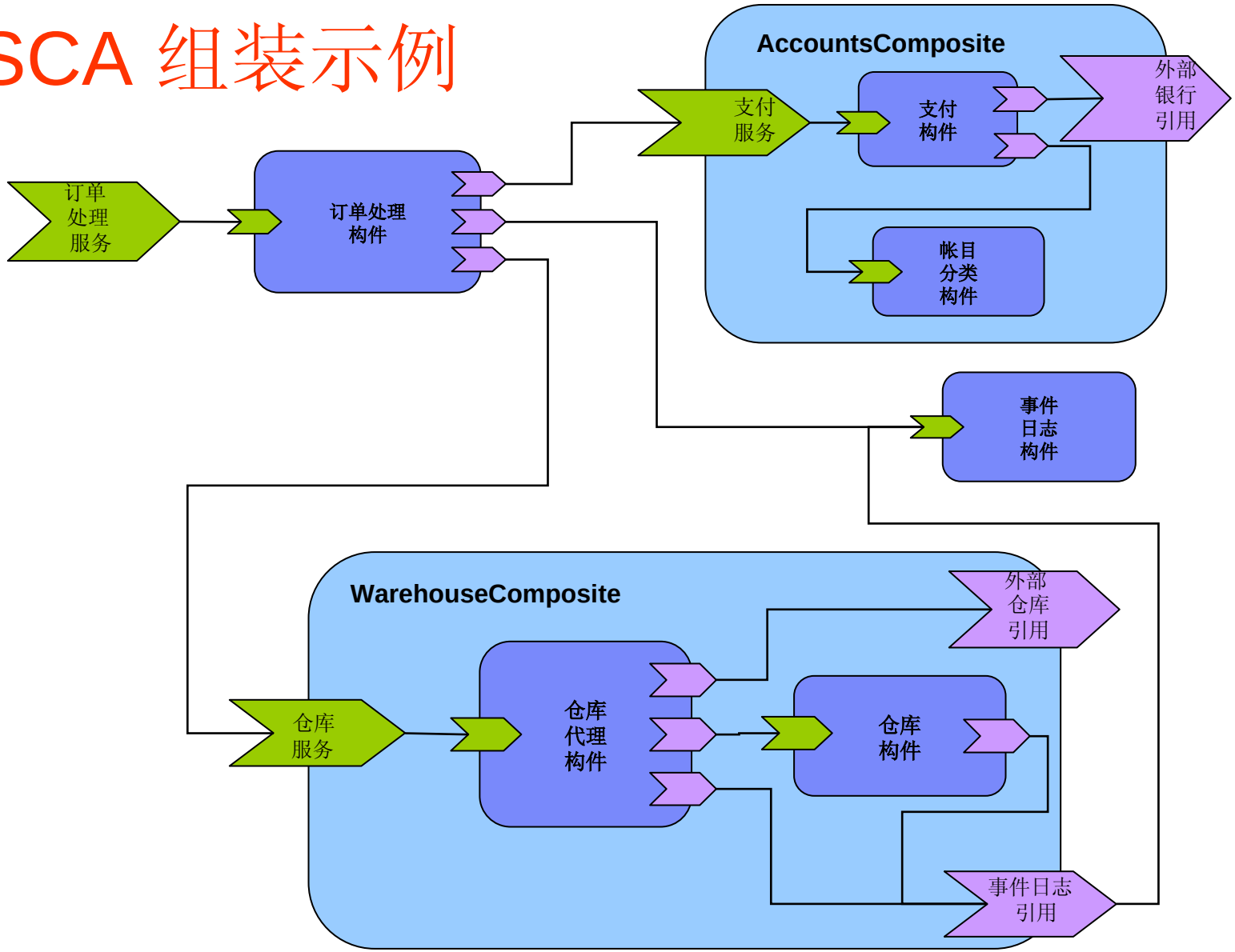

服务
- Java interface
- WSDL PortType
引用
- Java 接口
- WSDL PortType
属性
复合体 A
属性设置
构件
A
构件
B
服务
引用
提升
连线
提升
绑定
Web Service
SCA
JCA
JMS
SLSB
...
实现
- Java
- BPEL
- C++
- PHP
- Composite
...
绑定
Web Service
SCA
JCA
JMS
SLSB
...

复合体示例

bigbank.accoun tcomposite



SCA 组装示例


AccountsComposite
支付
服务
支付
构件
外部
银行
引用
订单
处理
服务
订单处理
构件
帐目
分类
构件
事件
日志
构件
WarehouseComposite
外部
仓库
引用
仓库
服务
仓库
代理
构件
仓库
构件
事件日志
引用

总结

总结

- 基本的 SOA 价值观
  - 低成本的集成，获得更大的灵活性
- 使用 SOA 构建的 SCA 模型系统
  - SDO 提供了理想的数据操纵层
- SCA 是 SOA 的利器
  - 也许是最重要的利器

谢谢！

请提问！