Lecture 1: Introduction to SOA

Lian Yu

The School of Software and Microelectronics

Peking University

No.24 Jin yuan RD, Beijing 102600

Agenda

- SOA架构的十大技术理论体系
- A case study 1: automation of reimbursement process
  - What problems are we expected to solve?
  - What solution might we have?
  - What have we learnt and What are we going to learn?
- A case study 2. Process improvement & optimization
- SOA principles (OASIS: Organization for the Advancement of Structured Information Standards)
  - What is a service?
  - SOA Concepts Example
  - Web services approach to a service-oriented architecture
  - What is SOA? And Why?
  - SOA principles
  - SOA Reference Model

SOA架构的几大技术理论体系

- 当前国内要发展SOA主要有三方面工作：
  - 方法、工具和环境。
  - 方法是工程技术，由基础理论来指导提出的。
  - 所以一门科学必需要包括：认知科学(哲理)、工程技术和方法、最后是理论。
- 架构的演化过程
  - SOA是从面向对象、构件架构等逐步发展完善，且相互依托、相互补充、又各自适应不同范围，因此在讨论SOA理论时，要了解它是如何演化过程来，继承了哪些理论体系，其适应度如何。

**** 2010-4-13　转载自：赛迪网

结构编程方法

- 40年前国际上发生了“软件危机”

- 正在此时，一位荷兰的物理家 E. W. Dijkstra 提出了一种“结构程序设计方法”，他认为：人的智力是有限的，采用数学或物理学的思维方法，用枚举、抽象、归纳、类比等思维方式简化问题。

- 所谓“结构程序设计方法”，就是基于面向对象设计方法的早期蓝本，侧重於解决程序**正确性**的编程的方法，以此为基础建立了软件工程这门学科，建立了编程的基础理论体系，也是第一个技术与基础理论体系。

“面向对象”的可重用理论

- 由面向对象发展到面向构件，由面向构件再发展到面向服务，因此它们的认知观和基础理论都是息息相关的。
  - 解决大型软件的**开发效率和质量**除了要解决编程的正确性外，还必需解决开发周期长、复用性差、成本高、文档多以及难以适应系统**演化**等问题，这些问题十多年来仍旧困惑着这门学科，“软件危机”仍未解决。
  - 人们的知识是从一个定理、一个原理逐步积累起来的，社会是依靠知识的不断积累发展的。然而编制软件每次却都是从零开始，这是造成“软件危机”的根本原因。由此提出了编程工作是否也可以重用以前成功的经验和程序呢?整整经过十多年的探索，到七十年代才获得成功。
- 这套方法和理论在产品开发和科研领域方面用得很多，称它为第二个技术与基础理论体系。

面向构件和架构 (1/2)

- 鉴于面向对象的缺陷，三位面向对象的奠基人联合起来，创建了UML统一建模语言。UML为软件开发和SOA的产生起到奠基和里程碑的作用。
  - UML主要理论成果是：统一面向对象的基本概念，并引进了许多新的概念，认为软件开发的过程实质上是从抽象的模型逐步细化，过渡到具体的实现，其中间的每个阶段都是实现了某一抽象模型，UML为此提供了建立模型的工具。
  - 用直觉的图形来建立模型，从此软件专家就有了自己的工具，正如音乐家有了五线谱工具那样。为适应软件的多变性，提供了演化的概念。
- 实际上此建模理论是第三个技术与基础理论体系，它为演化到构件和架构概念奠定基础理论模型。

面向构件和架构 (2/2)

- 由于工程上的实施缺乏开发规范，在技术上要求开发人员的素质较高，很少见到真正运用UML的方法于实际的工程开发应用软件中，最大的问题是被开发出来的软件难以演化，而软件要能适应变化是客观存在的。
- 为此发展出单纯重用的“构件和架构”技术及其理论体系。在1998年日本京都召开的“基于构件的软件开发(CBSD)”国际专题学术会议上，一致认为软件开发技术离不开构件和体系结构。软件体系结构现简称“架构”。
  - 在此之前的软件架构都采用层次结构的架构，直到分布式系统提出了用户端/服务器模式后，才产生对架构的研究，出现了构件和架构，也就是**第四个技术与基础理论体系**。
  - 卡内基·梅隆大学为软件的架构和框架建立了扎实的基础理论，软件体系结构是软件系统的高级抽象，体现了软件设计思想。反映了系统开发中最早的决策，明确了系统有哪几部分组成，它们之间是如何交互的;进一步影响到资源的配置、团队的组织以及产品的质量。系统的成败也在于体系结构。

三层体系结构分布式系统 (1/2)

- 三层体系结构是由二层结构的胖终端中的应用构件独立出来组成了应用层。为解决分布式系统中的各种潜在复杂性，提出了中间件技术及其理论，称为第五个技术与基础理论体系。
- 2006年IBM公司提出SOA后，很快被广泛接受，其原因可从客观需求上和技术成熟度上三方面来叙述

三层体系结构分布式系统 (2/2)

- 其一，客观上需要，随着网络普及化，用户越来越迫切需要将现有多个应用系统集成，以能实现更强的信息处理功能。如电子商务的供应链、智能交通、电子政务、数字地球等已是本世纪发展的热点。Gartner预计，到2008年基于构件产品将占领70%的市场份额。
- 其二，面向对象和构件架构的基础理论和技术已趋向成熟，发展到统一建模语言，提供建模工具。中间件集群理论已趋向成熟，并提出了中间件Inter Bus技术。
- 其三，浏览器技术普及，已成为行业标准，奠定了SOA的基础理论和技术规范，由此已是水到渠成，使SOA茁壮成长。

SOA在实现中的组成部分

- SOA的体系结构仍旧是三层或N层结构，但对异构平台各层之间的联系，不是用CORBA、J2EE 或 .NET 的方式，而且用WBDL和SOAP来实现，它们的概念简单统一。
- 目前都是采用嵌入ESB企业服务总线的平台来实现，ESB是一个中间件群，确保系统实现了服务接口、各种中间件以及松耦合的三个方面功能，因此称它为SOA技术与基础理论体系组成部分。
- 另外，普遍采用BPEL(业务过程执行语言)来描述用户需求，由BPM(业务过程管理平台)来解释执行，构成了SOA技术与基础理论体系组成部分。

SOA的主要优点

- 1. 利用现有的资产。
  - 方法是将这些现有的资产包装成提供企业功能的服务。组织可以继续从现有的资源中获取价值，而不必重新从头开始构建。
- 2. 更易于集成和管理复杂性。
  - 将基础设施和实现发生的改变所带来的影响降到最低限度。因为复杂性是隔离的，当更多的企业一起协作提供价值链时，这会变得更加重要。
- 3. 更快地整合现实。
  - 通过利用现有的构件和服务，可以减少完成软件开发生命周期所需的时间。这使得可以快速地开发新的业务服务，并允许组织迅速地对改变做出响应和缩短开发时间。
- 4. 减少成本和增加重用。
  - 通过以松散耦合的方式公开业务服务，企业可以根据业务要求更轻松地使用和组合服务。
- 5. SOA业务流程是由一系列业务服务组成的，可以更轻松地创建、修改和管理它来满足不同时期的需要。

建立软件开发方法和规范

- 构件构架理论体系的应用是适用于构件技术创立的，当发展到面向服务的体系结构时，必需加以修改和扩充，现在称为**模型驱动**MDD的需求工程建模理论，可以称它为SOA技术与基础理论体系组成部分。
- 另一个构件的**领域工程**将要扩充成SOA的参考结构，这是SOA技术与基础理论体系组成部分。SOA的**门户**将要反映SOA所有功能的表现层界面，为此如何将最新的WEB2.0与SOA给合，这是SOA技术与基础理论体系组成部分。
- **模型驱动、领域工程及门户等**上述三方面是SOA在实际应用时必需要建立的理论和技术。

SOA的发展状况

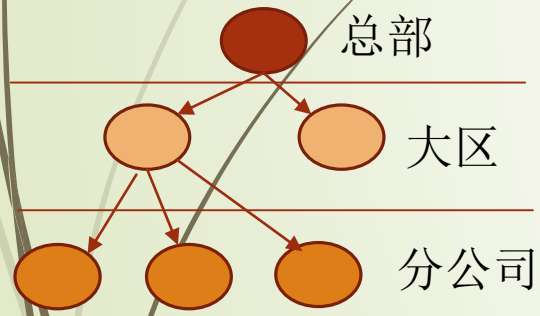
- 2006年，IBM公开宣布SOA计划，2009年年底，BEA公司、甲骨文公司、惠普等所有名牌公司都在中国发布了关于SOA的消息。
- 由于SOA模型统一，因此都是把本公司的中间件产品向SOA靠拢，提供开发和运行SOA系统的相应工具和环境，以争取市场的份额。
- 北京市市科委将为SOA核心平台研发提供资金，由软件行业促进中心统一管理，促进北京市IP行业发展。
- 随着SOA理论的发展，各种与SOA有关的规范和标准将不断出现，如SOAP、WSDL、ESB、BEPL语言等，它们的出现象征着SOA将逐步走向成熟。我们更应注意着各家公司所开发的工具和环境产品，有助于SOA的大力推广应用

A Case Study

- Automating Reimbursement Process
  - Define the Requirements (High level)
  - Describe the Reimbursement Process
  - Design the Architecture

SL公司的“差旅费报销制度”

- 差旅费用发生前，事前要有申请，有借款的情况下，需填借款申请单。员工出差完毕后，需在3个工作日内填写完差旅费报销单。对于超标准，超预算的情况，需走相关审批流程。
  - 费用会计在审核报销单时，对于超标超申请额度的情形，若报销金额<申请金额+100，可以直接进行核定金额。
- 费用的归属按照受益原则进行划分。
  - 总部到大区和分公司检查工作或开展其他总部工作事宜的，费用全部计入总部；大区到分公司检查或开展大区的其他工作，费用计入大区；分公司因工作需要人员到大区或总部学习的，费用计入分公司；大区人员因工作需要到总部学习的费用计入大区。总部各个中心统一组织、安排分公司人员到总部集中培训的，费用计入总部对口中心的费用；筹建、增援人员的安排原则上都应该是根据当地的需求或征得当地负责人同意后进行组织，费用计入被筹建和增援的公司。
  - 在提交报销时，费用只能按照报销人所属成本中心进行归属，此类情况后期进行还原即可。



◎ Business Rules

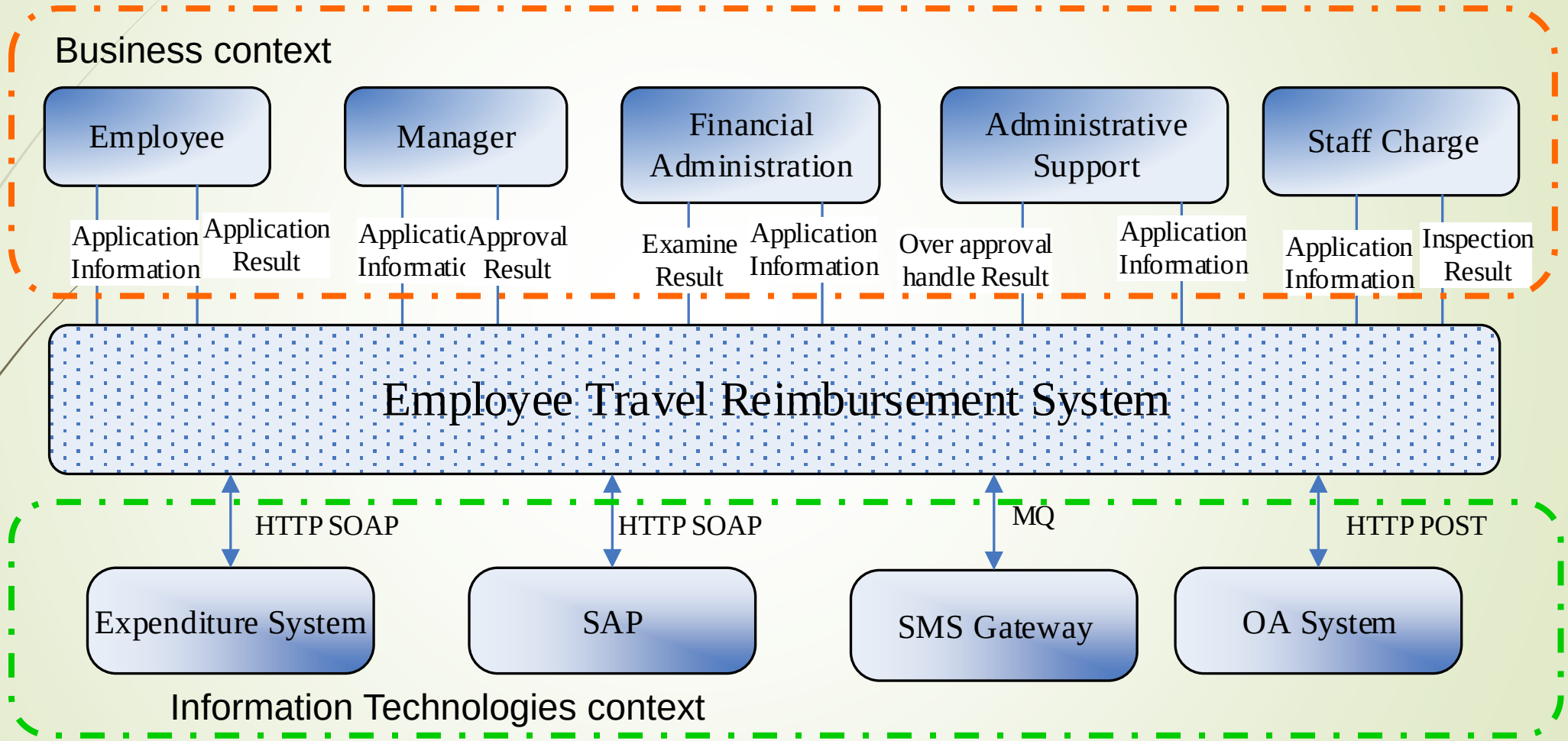
SL公司的差旅费报销业务需求

- 员工出差结束后，要报销差旅费用，需使用浏览器填写报销申请单，启动报销流程
- 部门负责人、财务服务部、员工费用部等相关负责人登录后在代（待）办事项中查看等待审批的流程，根据业务规则进行处理
- 在审批过程中，审批通过或者拒绝均要发送邮件通知员工；如果最终审批通过，需要在SAP系统记账，并调用资金系统向员工支付报销费用，支付完成后，发送短信通知员工。
- SL公司已有系统是
  - OA系统、短信网关、资金系统、SAP系统

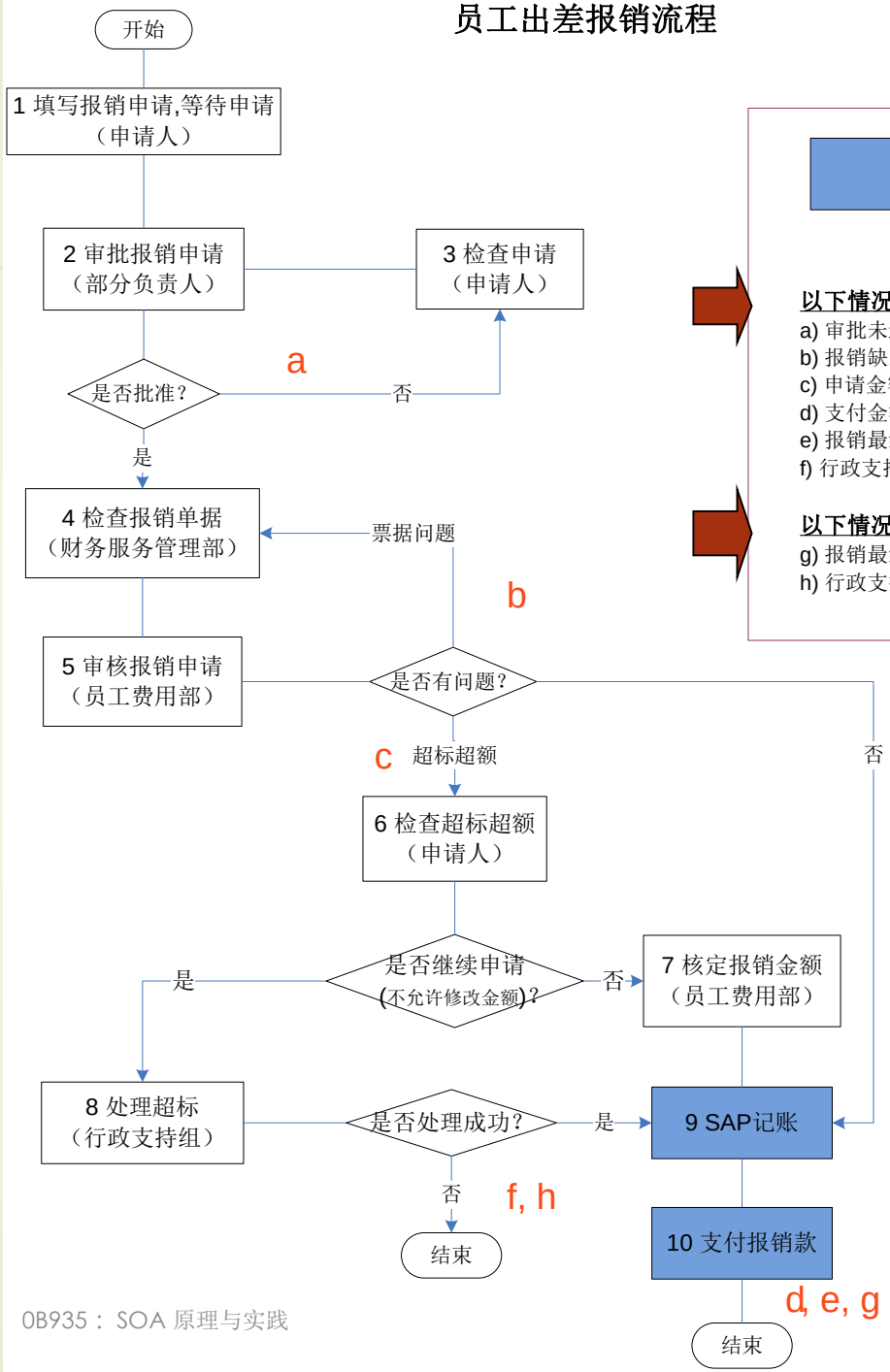
Employee Travel Reimbursement System

Vision	▫ Assist the company in handling travel reimbursement of employees ***effectively***. ▫ Standardize the reimbursement process for the company. ▫ Enhance work ***efficiency and accuracy*** on reimbursing.
Scope	▫ Employee applies for a travel reimbursement. ▫ Department Manager verifies the request. ▫ Department of Financial Admin examines the request. ▫ Department of Staff Charge inspects the request. ▫ Department of Administration Support handles over approval problems. ▫ Email and SMS (short Message Services) notification. ▫ SAP keeps accounts. ▫ Expenditure Payment in Finance System

Business/IT Context Diagram



员工出差报销流程



表示调用服务

以下情况需要发送邮件通知员工

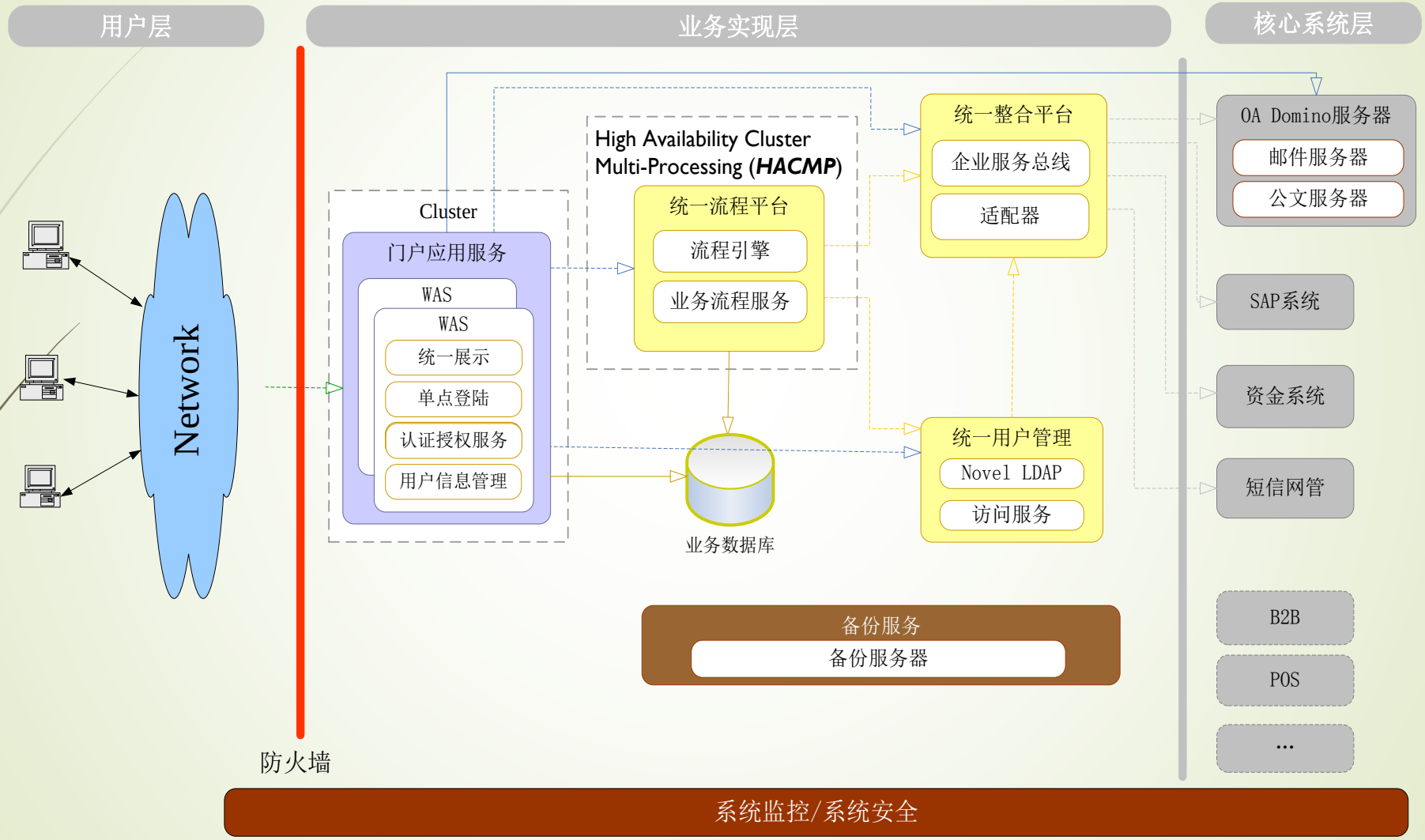
a) 审批未通过，打回给员工
b) 报销缺乏票据
c) 申请金额出现超标超额
d) 支付金额通过
e) 报销最终审批通过,流程成功结束
f) 行政支持组未能接受员工的超标申请，流程中止

以下情况需要发送短信通知员工

g) 报销最终审批通过,流程成功结束
h) 行政支持组未能接受员工的超标申请，流程中止

注：具体流程参考《员工出差费用报销流程》

逻辑架构


用户层
业务实现层
核心系统层
Network
Cluster
门户应用服务
WAS
WAS
统一展示
单点登陆
认证授权服务
用户信息管理
High Availability Cluster
Multi-Processing (HACMP)
统一流程平台
流程引擎
业务流程服务
统一整合平台
企业服务总线
适配器
业务数据库
统一用户管理
Novel LDAP
访问服务
备份服务
备份服务器
OA Domino服务器
邮件服务器
公文服务器
SAP系统
资金系统
短信网管
B2B
POS
...
防火墙
系统监控/系统安全

Architectural Decisions

- ESB Services Interface Type

-- Web Services / JMS / MQ => Web Services

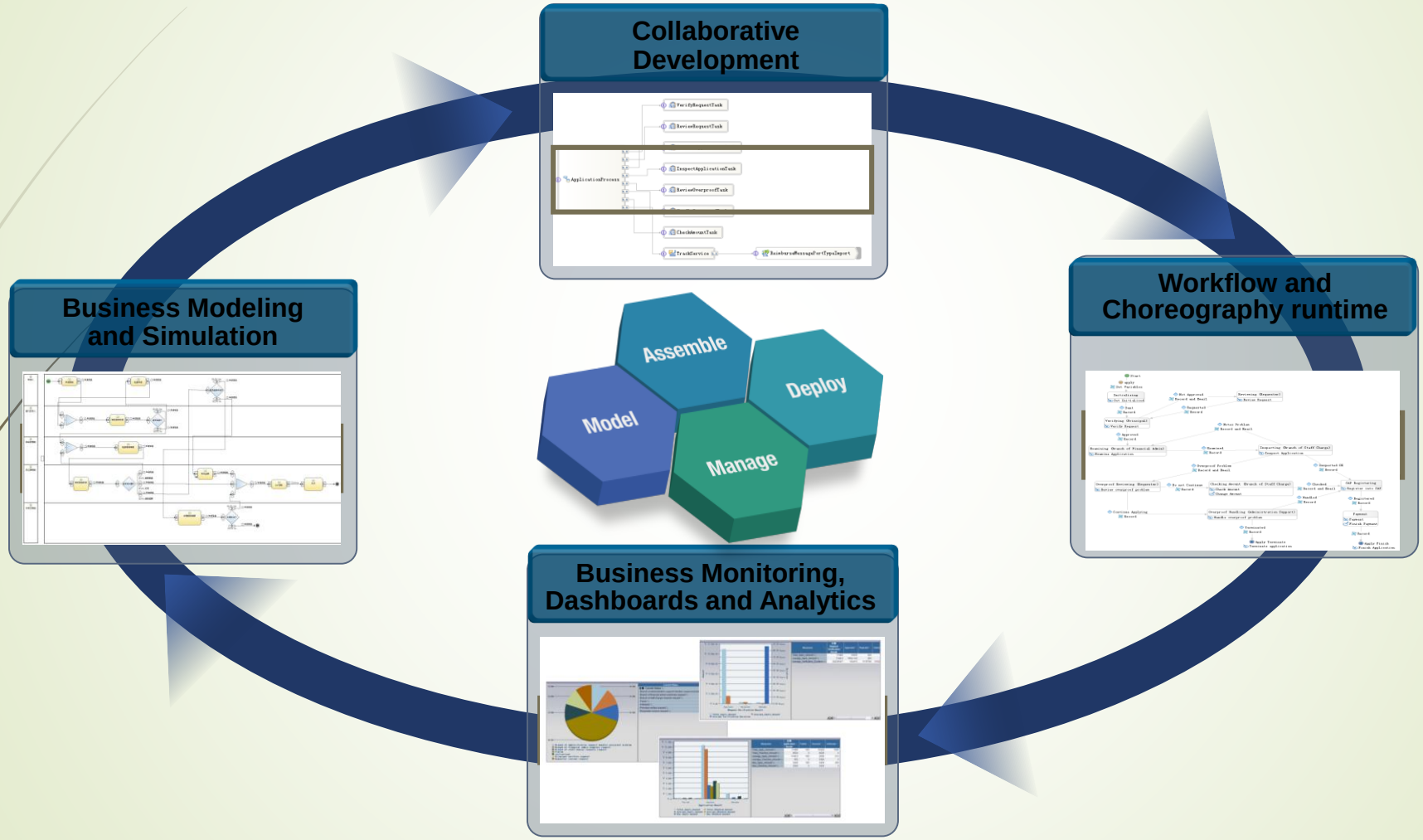
- User Registry

-- LDAP / Database => LDAP

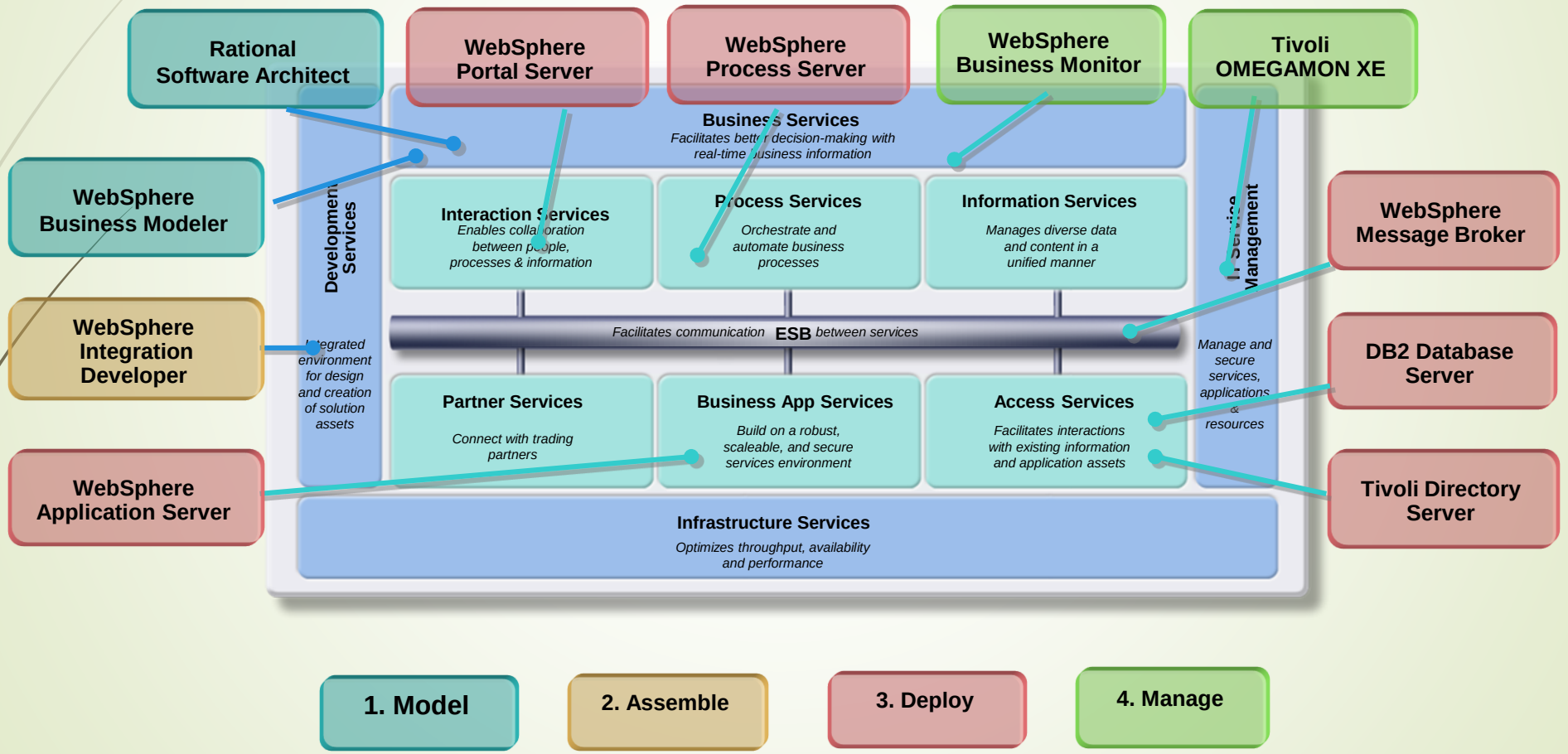
- Process Implementation

-- BPEL / Business State Machine (BSM) => BSM

Business Process Management Lifecycle

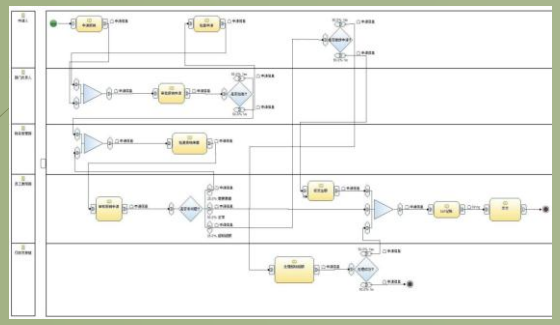


Service Types and SOA Tools



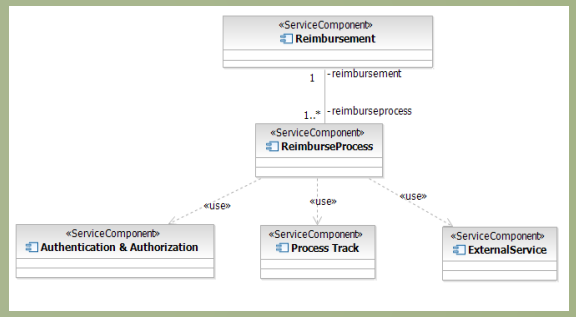
Business Process Model to Service Model

WebSphere Business Modeler



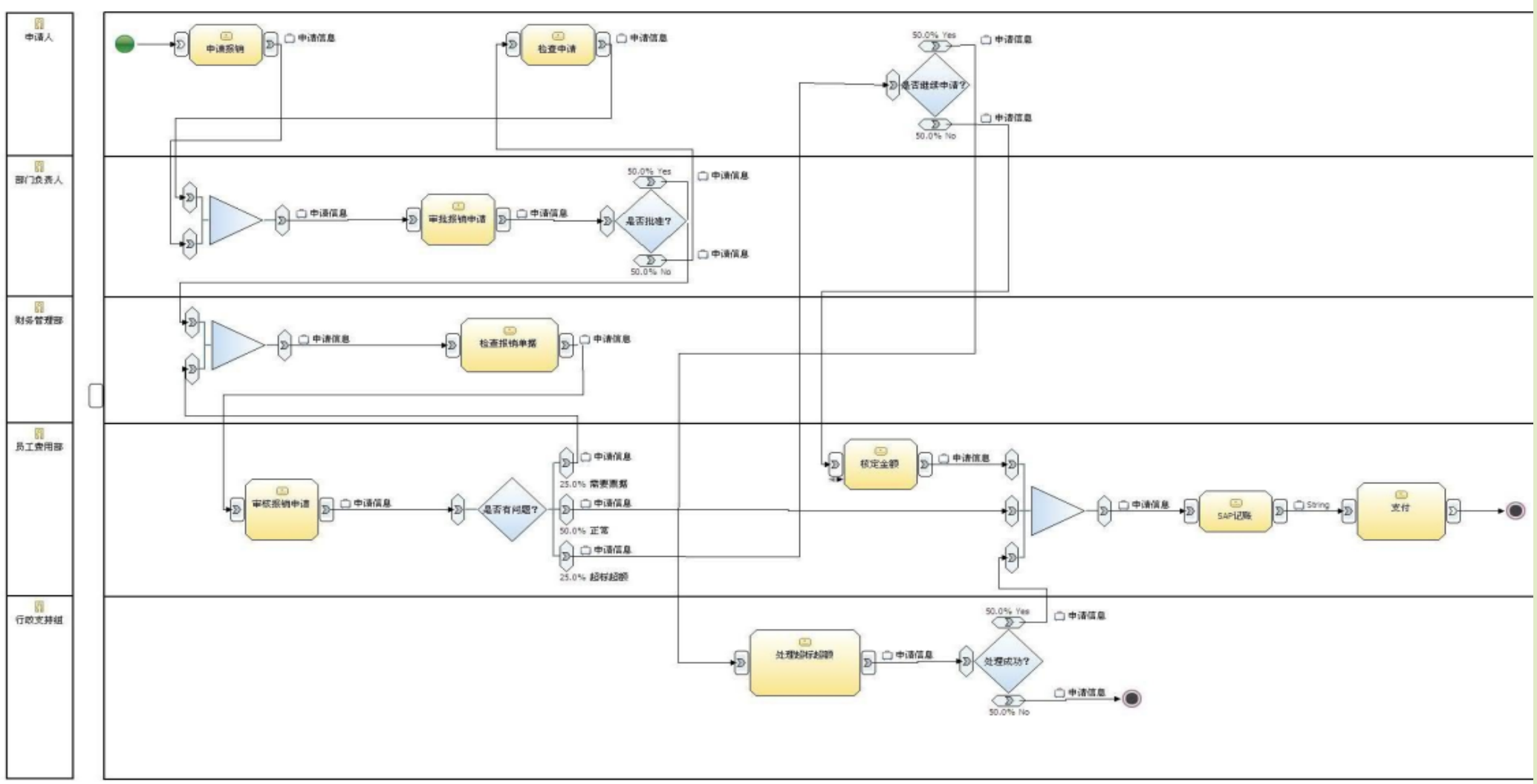
Business Analysis model

Rational Software Architect

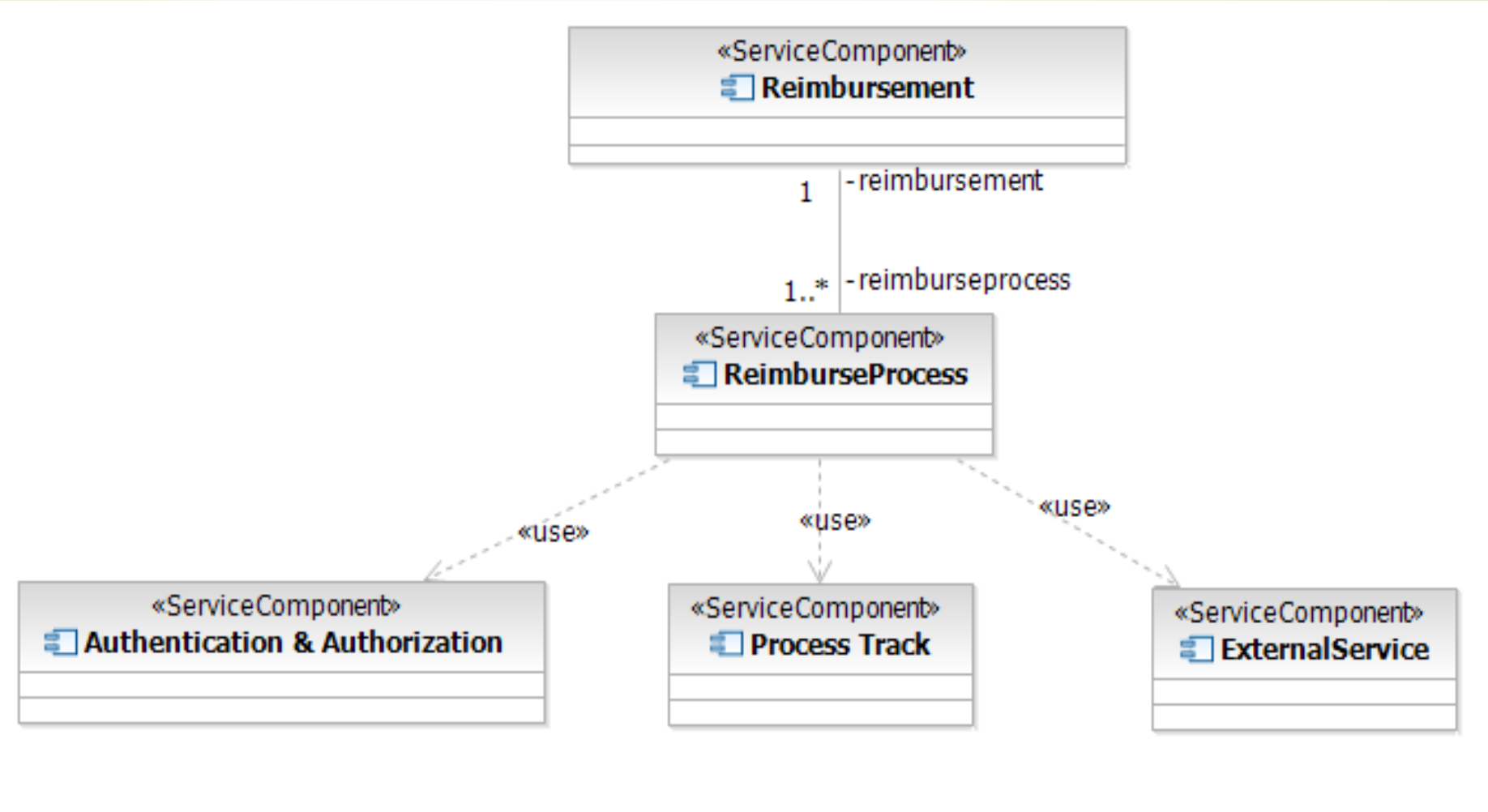


Design Solution Architecture

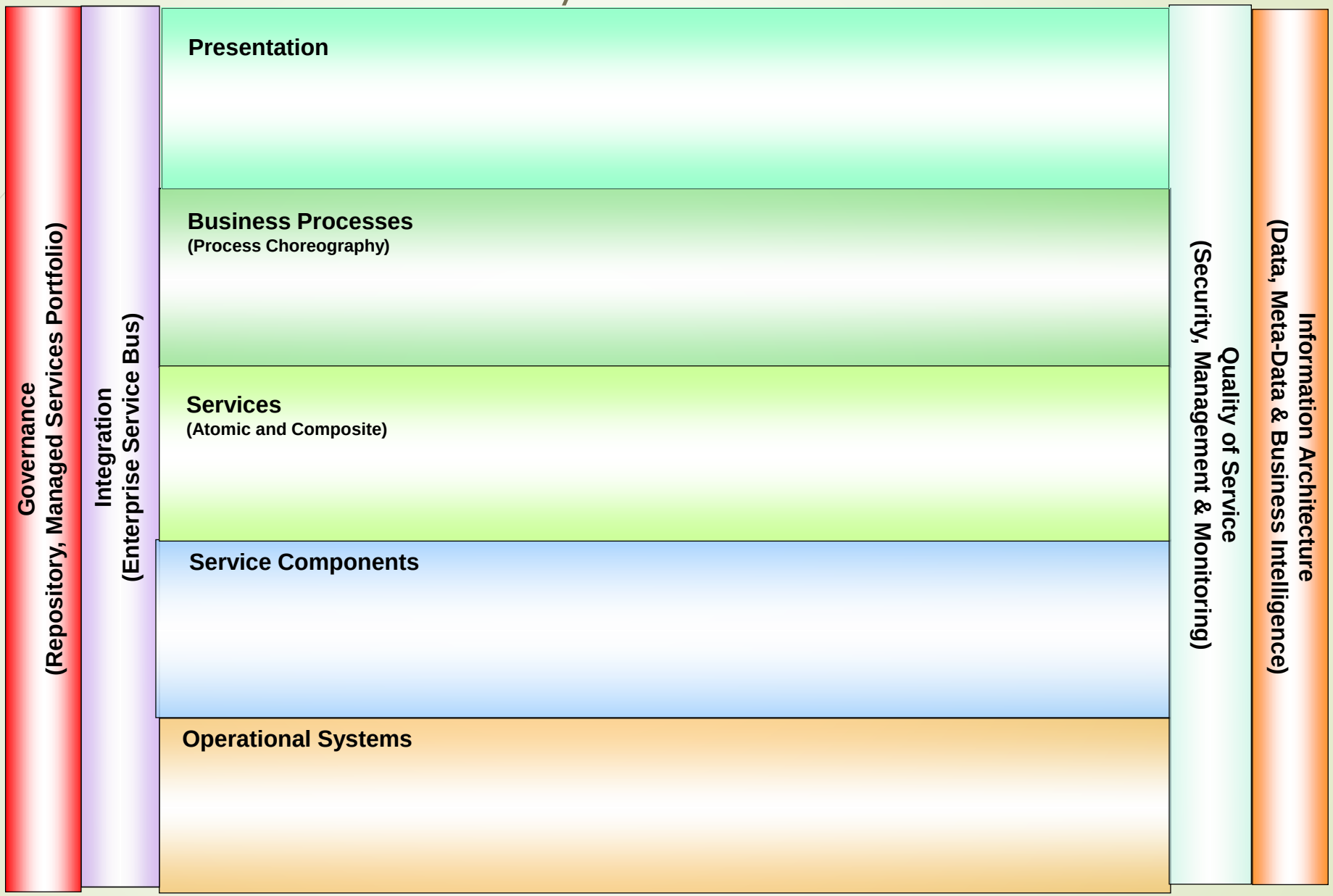
Process Model



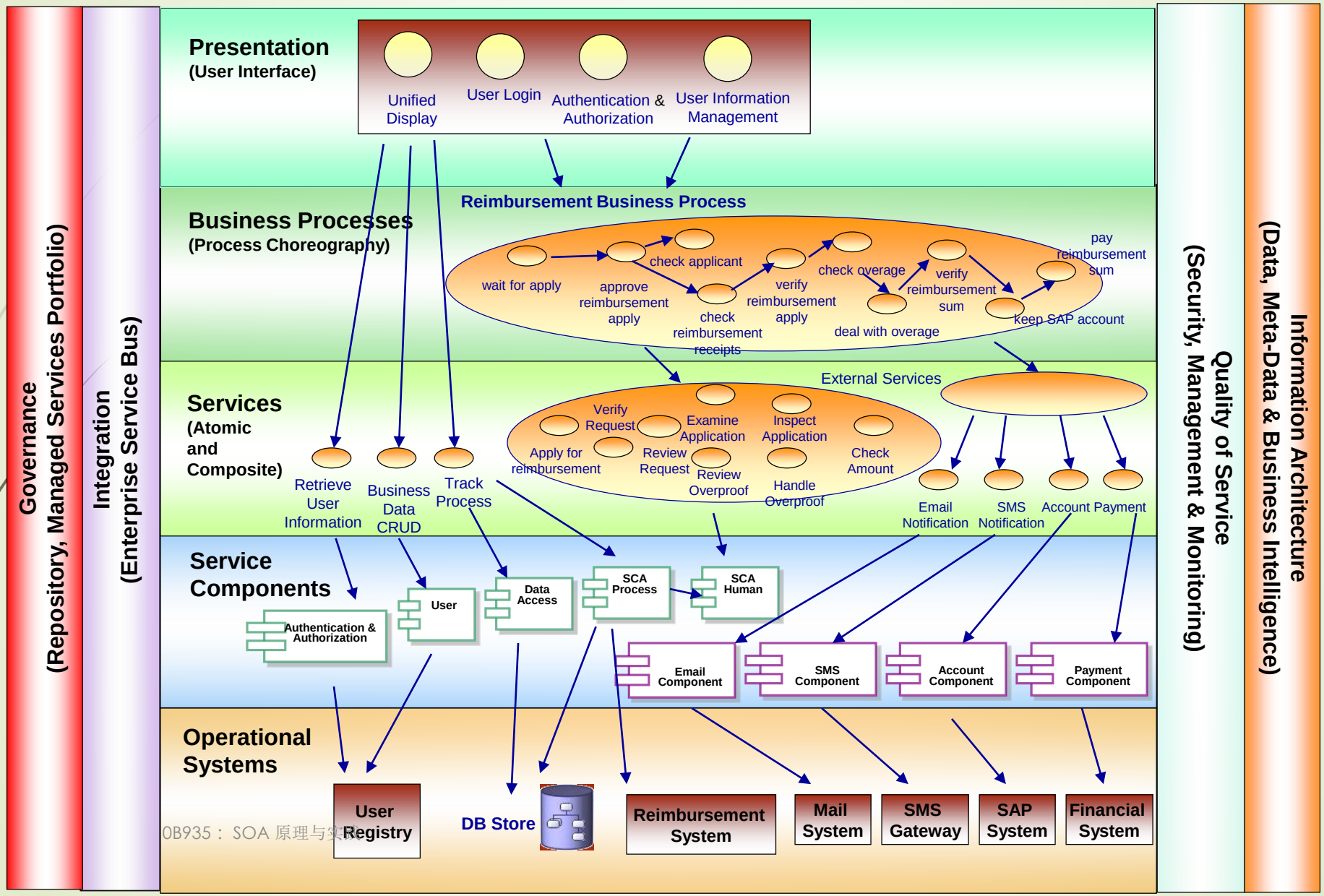
Service Component Model



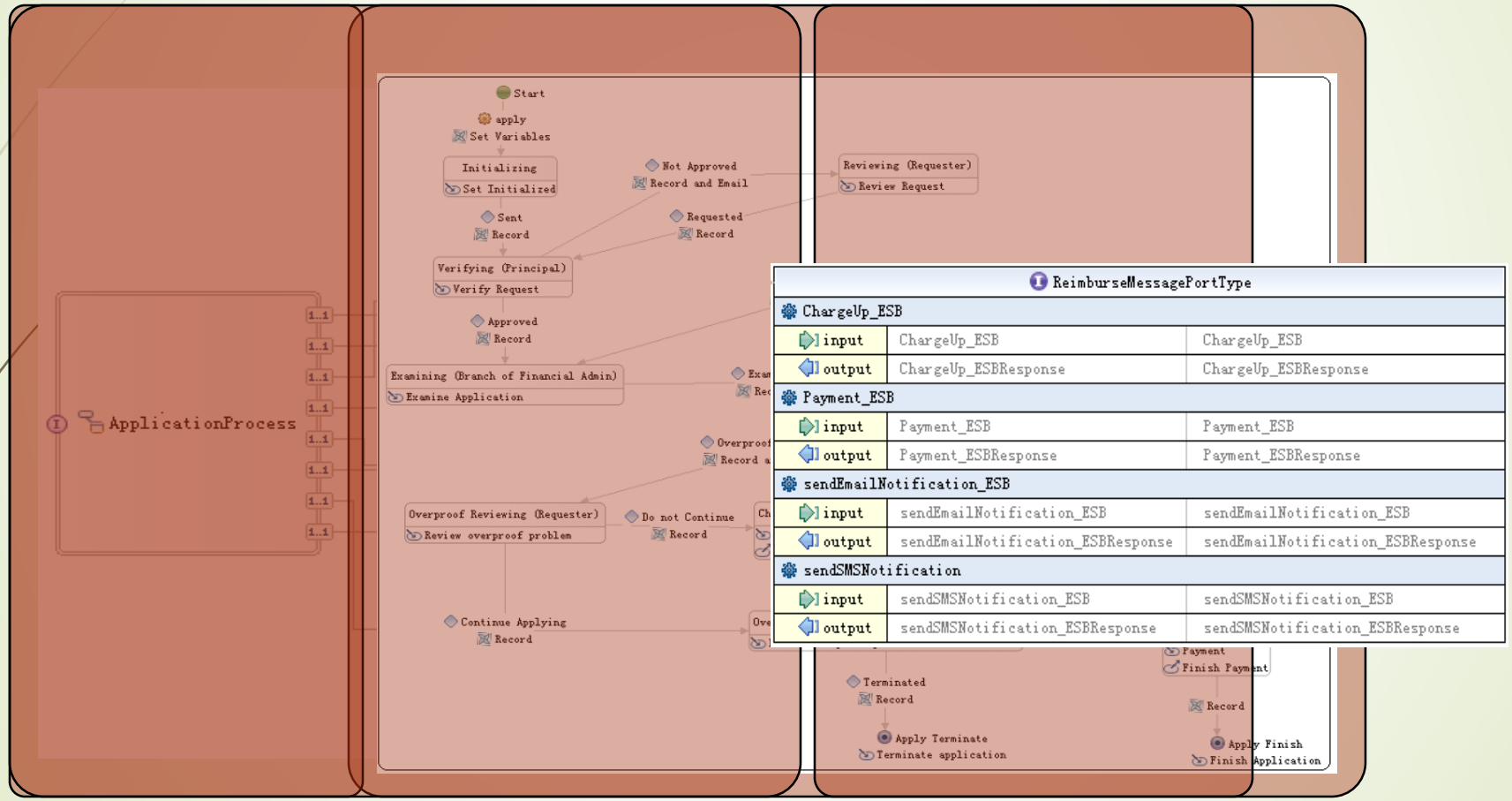
SOA Solution Stack Layer



Solution Stack Layer – Allocating Components to Layers



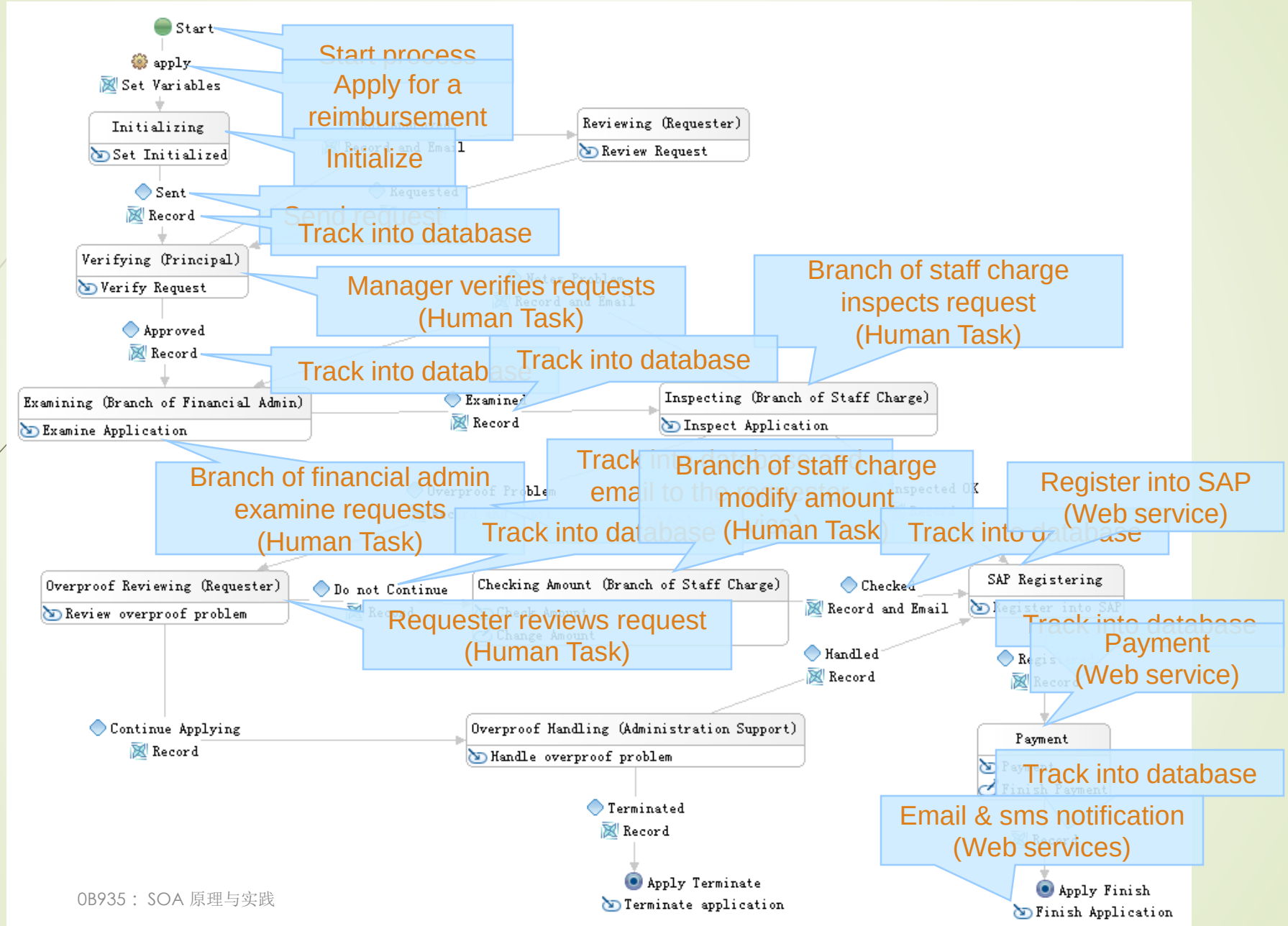
Component Implementation



Business State Machine (BSM)

BSM provides support for business items:

- have a life cycle
- have states that are driven by events
- typically have loop-based logic
- a way of developing, debugging, and monitoring


Start
apply
Set Variables
Initializing
Set Initialized
Sent
Record
Verifying (Principal)
Verify Request
Approved
Record
Examining (Branch of Financial Admin)
Examine Application
Examined
Record
Inspecting (Branch of Staff Charge)
Inspect Application
Reviewing (Requester)
Review Request
Overproof Reviewing (Requester)
Review overproof problem
Do not Continue
Checking Amount (Branch of Staff Charge)
Checked
Record and Email
SAP Registering
Handled
Record
Continue Applying
Record
Overproof Handling (Administration Support)
Handle overproof problem
Terminated
Record
Apply Terminate
Terminate application
Payment
Apply Finish
Finish Application
Start process
Apply for a reimbursement
Initialize
Track into database
Manager verifies requests (Human Task)
Branch of staff charge inspects request (Human Task)
Track into database
Branch of financial admin examine requests (Human Task)
Branch of staff charge modify amount (Human Task)
Register into SAP (Web service)
Requester reviews request (Human Task)
Payment (Web service)
Track into database
Email & sms notification (Web services)

Message Broker

Integrate services, coordinate resources

Transform different message formats and transport protocols,

Route messages

Message set

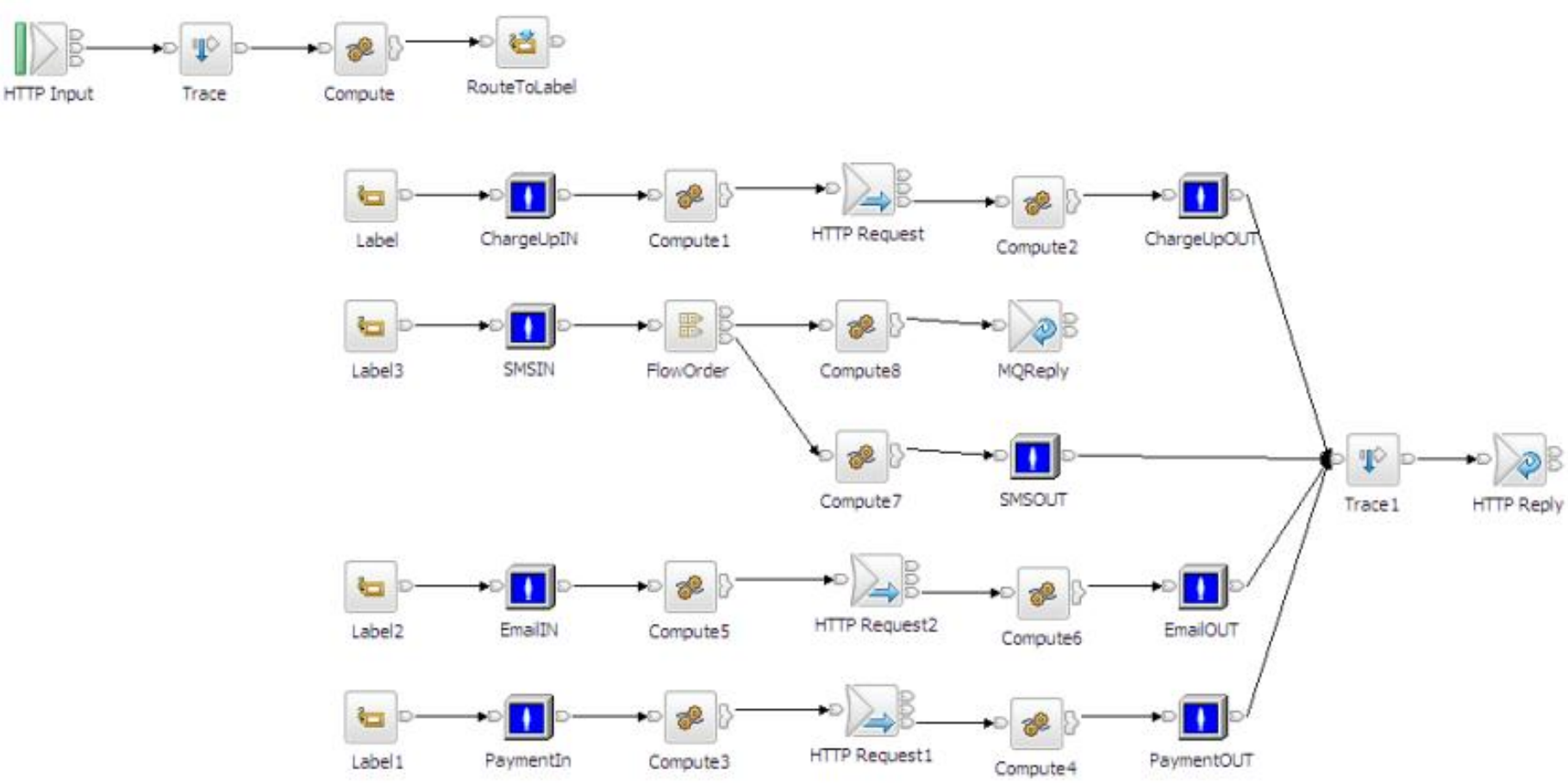
message formats and relationship definitions

Message flow

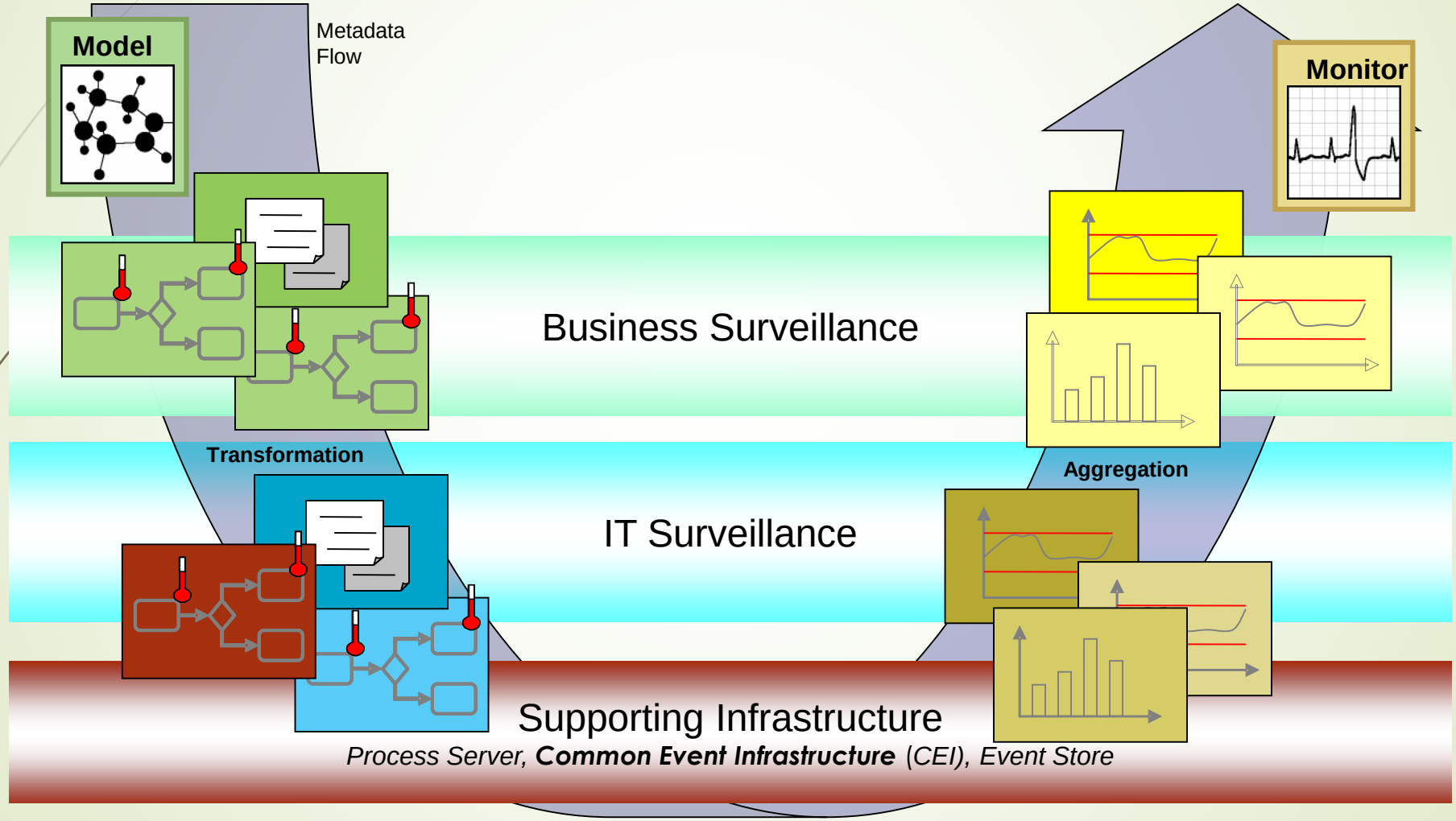
format and protocol transformation

message routing and common services

Message Flow



Monitor Conceptual View



Monitor

Business surveillance

- Instances
- KPIs
- Dimensions
- …….

WebSphere Business Monitor

DB2 Universal Enterprise Server
WebSphere Process Server
Dashboard Server

IT surveillance

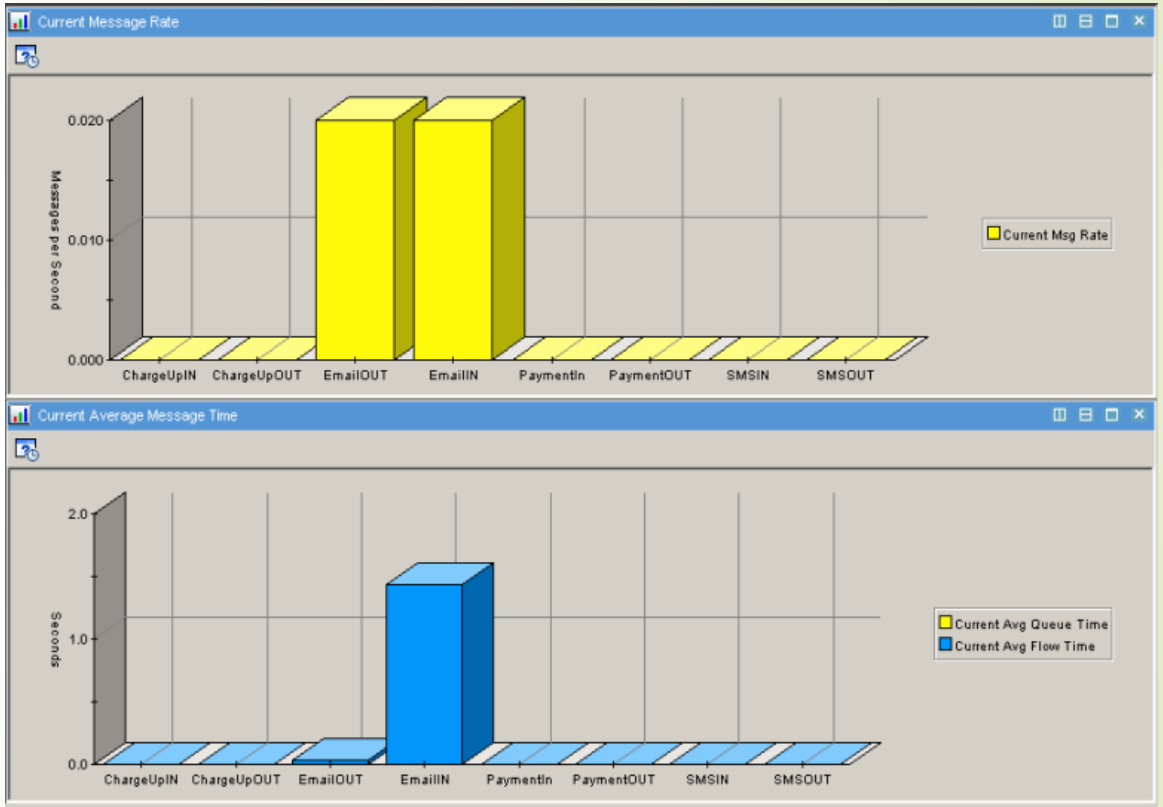
- Message flow
- Web Services
- ……

Tivoli OmegaMon XE for Messaging

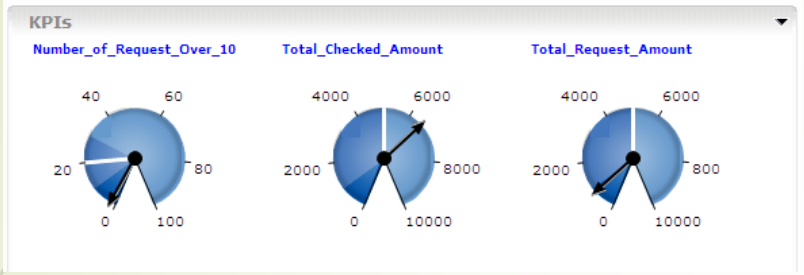
Tivoli Enterprise Portal Server / Client
Tivoli Enterprise Management Server
Monitoring agents

Service Monitoring

- Email Service
- SMS Service
- Changeup Service
- Payment Service



Business Dashboard Show



Number_of_Request_over_1000rm

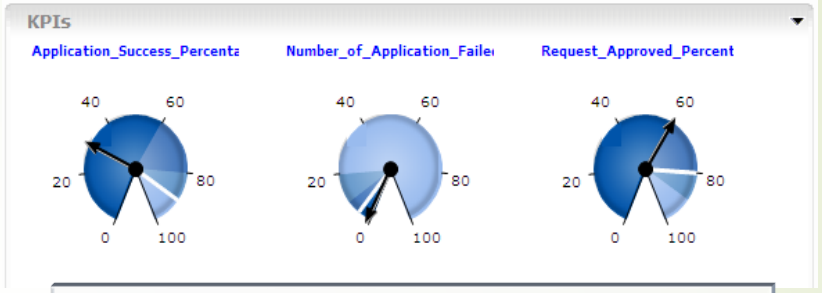
范围名称	起始值	结束值
Good	0	< 10
Acceptable	10	< 30
Poor	30	< 100

Total_Checked_Amount

范围名称	起始值	结束值
Low	0	< 1000
Middle	1000	< 5000
High	5000	< 10000

Total_Request_Amount

范围名称	起始值	结束值
Low	0	< 1000
Middle	1000	< 5000
High	5000	< 10000



Application_Success_Percentage

范围名称	起始值	结束值
Poor	0	< 60
Acceptable	60	< 80
Good	80	< 90
Excellent	90	< 100

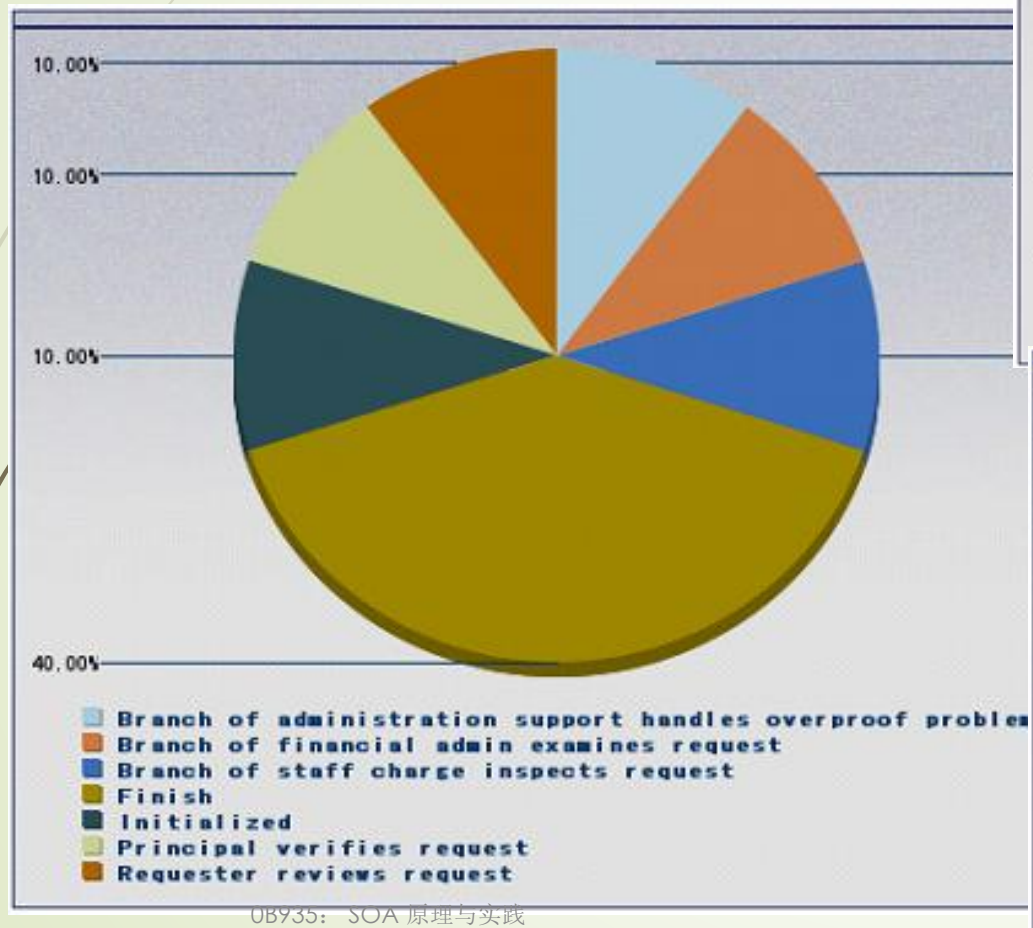
Number_of_Application_Failed

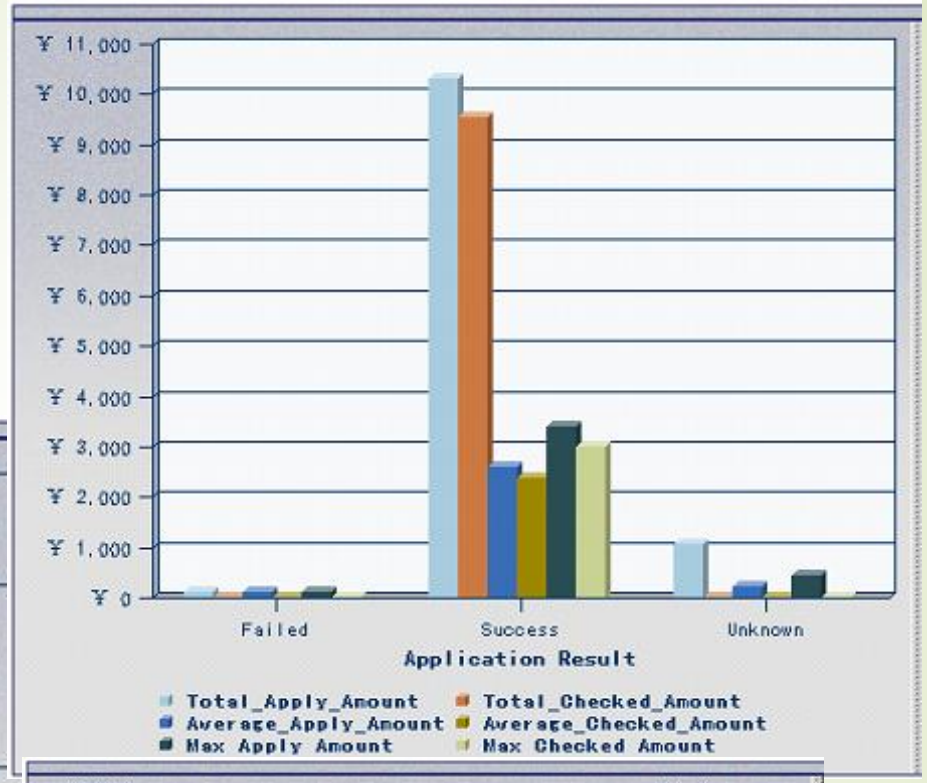
范围名称	起始值	结束值
Excellent	0	< 5
Good	5	< 10
Acceptable	10	< 20
Poor	20	< 100

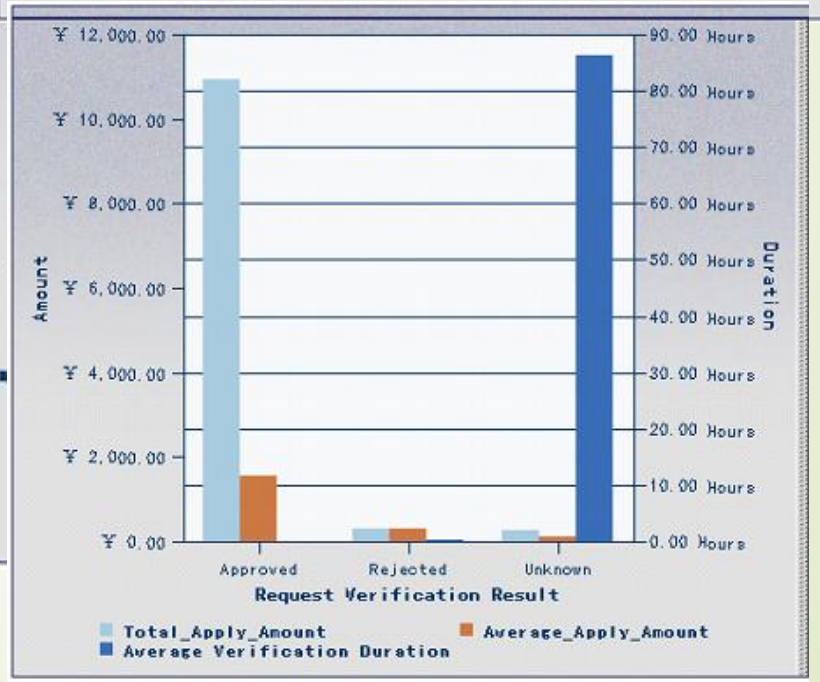
Request_Approved_Percentage

范围名称	起始值	结束值
Poor	0	< 60
Acceptable	60	< 80
Good	80	< 90
Excellent	90	< 100

Business Dashboard Show







What have we learnt and What are we going to learn? (1/2)

- What have we learnt?
  - Requirements Engineering
    - for requirements analysis, modeling and validation.
  - Architecture Design
    - Understand layered style, tied style and bus style
  - Integration/Assemble
    - Web Services, BEPL, Portal, Database
  - Testing
  - Deployment
  - Documentation

What have we learnt and What are we going to learn? (2/2)

- What are we going to learn?
  - Service Oriented Architecture
    - Concepts
    - Reference architecture and reference model
  - SOA entry points
  - SOA development life cycle
    - model
    - Assemble
    - Deployment
    - Monitor
  - OASIS SOA Model, and OASIS SOA Reference
  - SOA relevant tools (IBM, open source)

A Case Study (2)

- 京能热电公司的SOA业务流程分析及优化
  - Introduction to SWOT & Balanced Scorecard
  - 京能热电公司组织图
  - 管理功能模块图
  - 组件化业务模型
  - 燃煤检验子流程 (As-Is and To-Be)

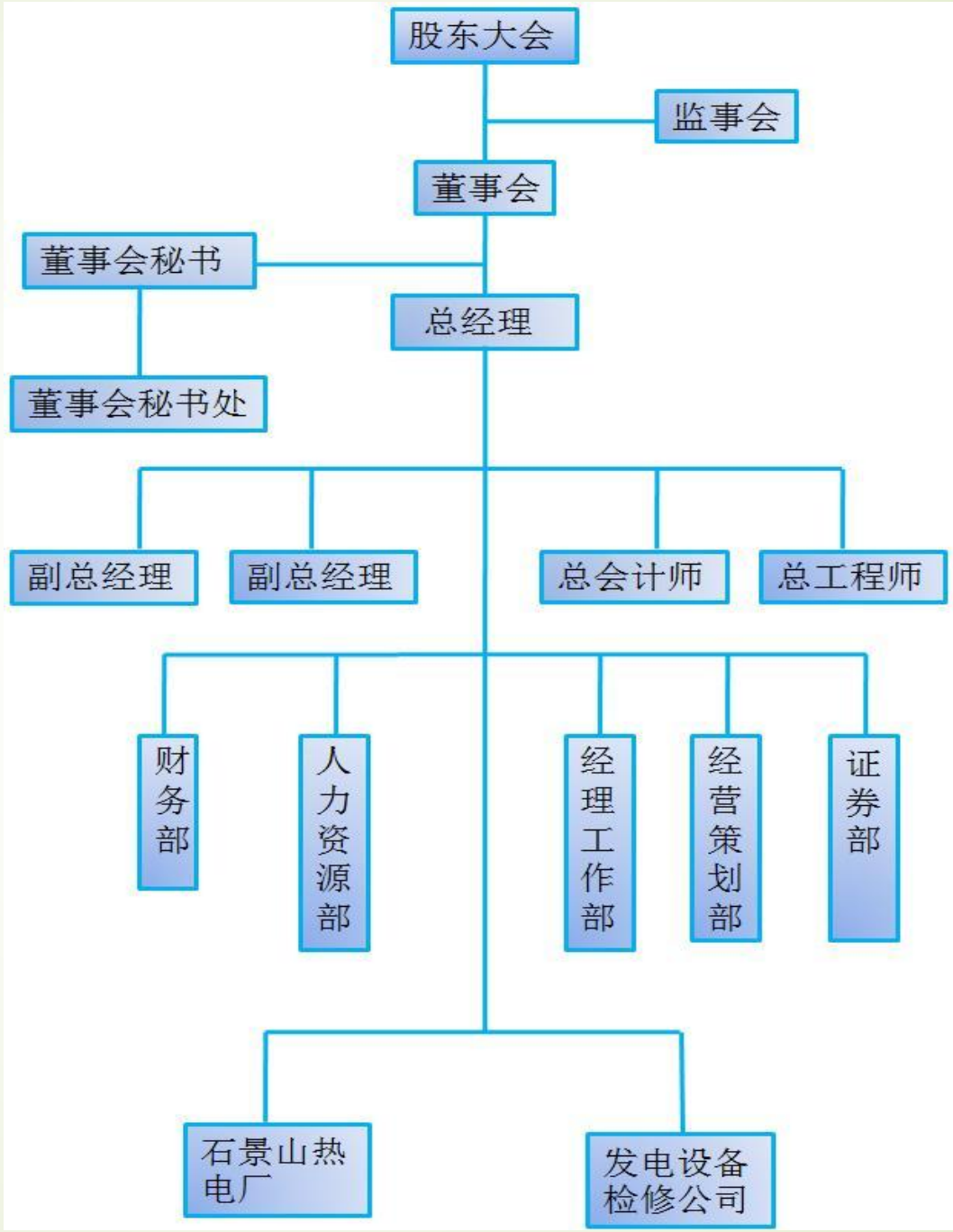
京能热电简介

北京地区第一家现代化大型股份制发电供热企业

发电量占北京地区发电量的四分之一

供热能力占北京市集中供热面积的五分之一

京能热电公司组织图


股东大会
监事会
董事会
董事会秘书
总经理
董事会秘书处
副总经理
副总经理
总会计师
总工程师
财务部
人力资源部
经理工作部
经营策划部
证券部
石景山热电厂
发电设备检修公司

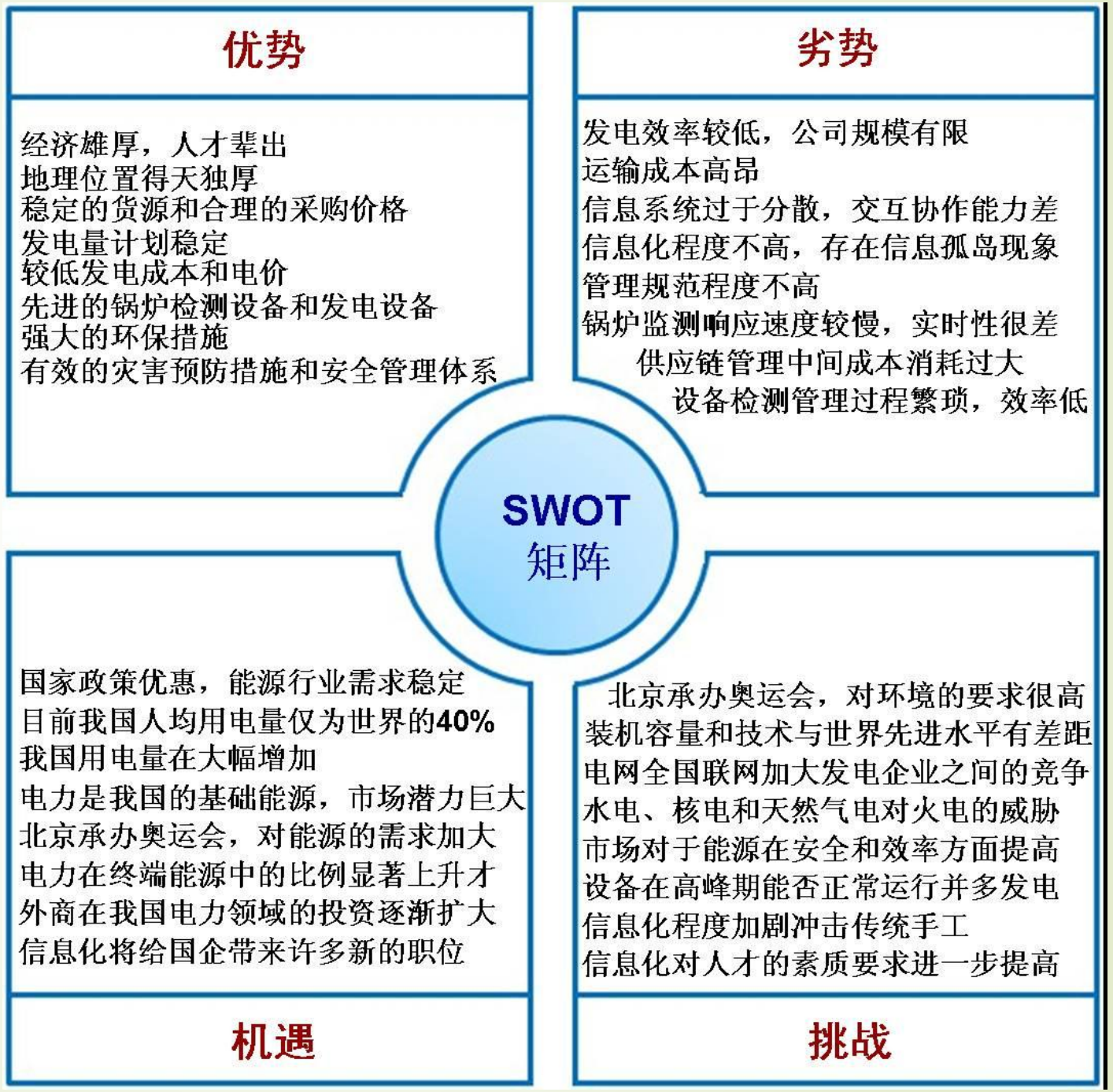
SWOT analysis (1/2)

- **SWOT analysis** (alternately **SLOT analysis**) is a strategic planning method used to evaluate the **S**trengths, **W**eaknesses/Limitations, **O**pportunities, and **T**hreats involved in a project or in a business venture.
  - **S**trengths: characteristics of the business, or project team that give it an advantage over others
  - **W**eaknesses (or Limitations): are characteristics that place the team at a disadvantage relative to others
  - **O**pportunities: *external* chances to improve performance (e.g. make greater profits) in the environment
  - **T**hreats: *external* elements in the environment that could cause trouble for the business or project

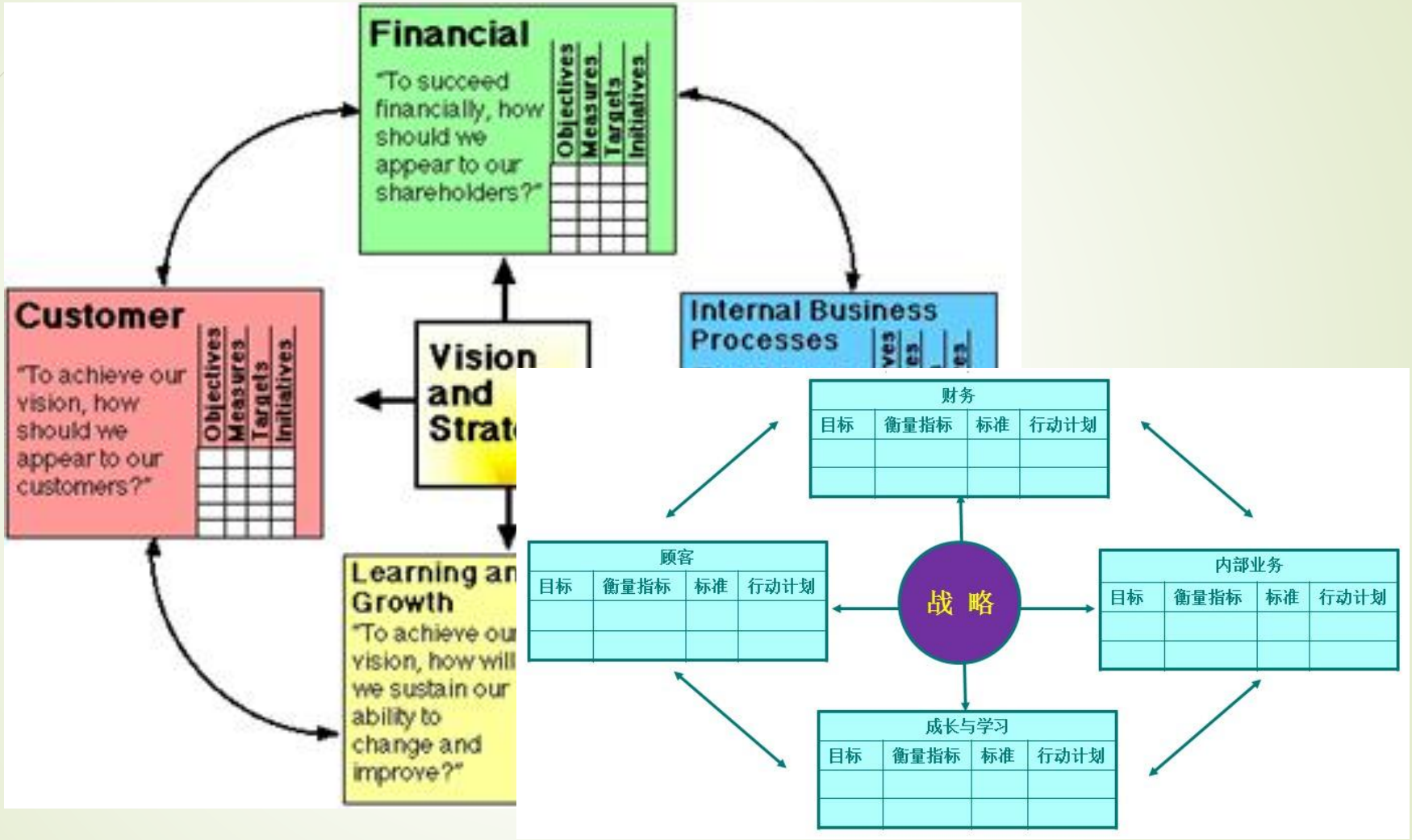
SWOT analysis (2/2)

- SWOT analysis is a method for **analysing a business, its resources, and its environment.**

- SWOT is commonly used as part of str :
  - Internal strengths
  - Internal weaknesses
  - Opportunities in the external environm
  - Threats in the external environment
- SWOT can help management in a bu
  - What the business does better than th
  - What competitors do better than the business
  - Whether the business is making the most of the opportunities available
  - How a business should respond to changes in its external environment


优势
经济雄厚，人才辈出
地理位置得天独厚
稳定的货源和合理的采购价格
发电量计划稳定
较低发电成本和电价
先进的锅炉检测设备和发电设备
强大的环保措施
有效的灾害预防措施和安全管理体系
劣势
发电效率较低，公司规模有限
运输成本高昂
信息系统过于分散，交互协作能力差
信息化程度不高，存在信息孤岛现象
管理规范程度不高
锅炉监测响应速度较慢，实时性很差
供应链管理中间成本消耗过大
设备检测管理过程繁琐，效率低
SWOT
矩阵
国家政策优惠，能源行业需求稳定
目前我国人均用电量仅为世界的40%
我国用电量在大幅增加
电力是我国的基础能源，市场潜力巨大
北京承办奥运会，对能源的需求加大
电力在终端能源中的比例显著上升才
外商在我国电力领域的投资逐渐扩大
信息化将给国企带来许多新的职位
机遇
北京承办奥运会，对环境的要求很高
装机容量和技术与世界先进水平有差距
电网全国联网加大发电企业之间的竞争
水电、核电和天然气电对火电的威胁
市场对于能源在安全和效率方面提高
设备在高峰期能否正常运行并多发电
信息化程度加剧冲击传统手工
信息化对人才的素质要求进一步提高
挑战

Balanced scorecard (1/2)

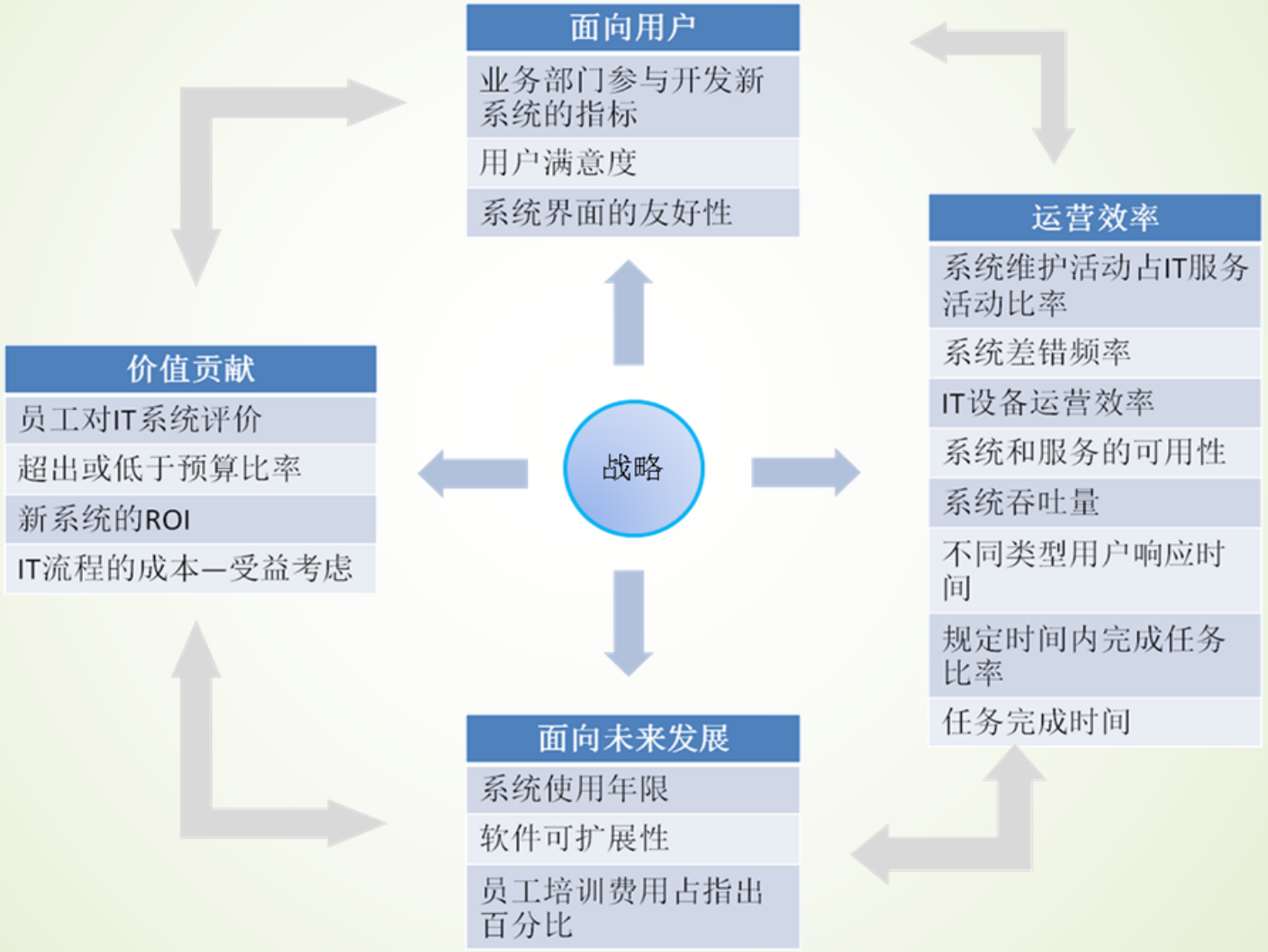


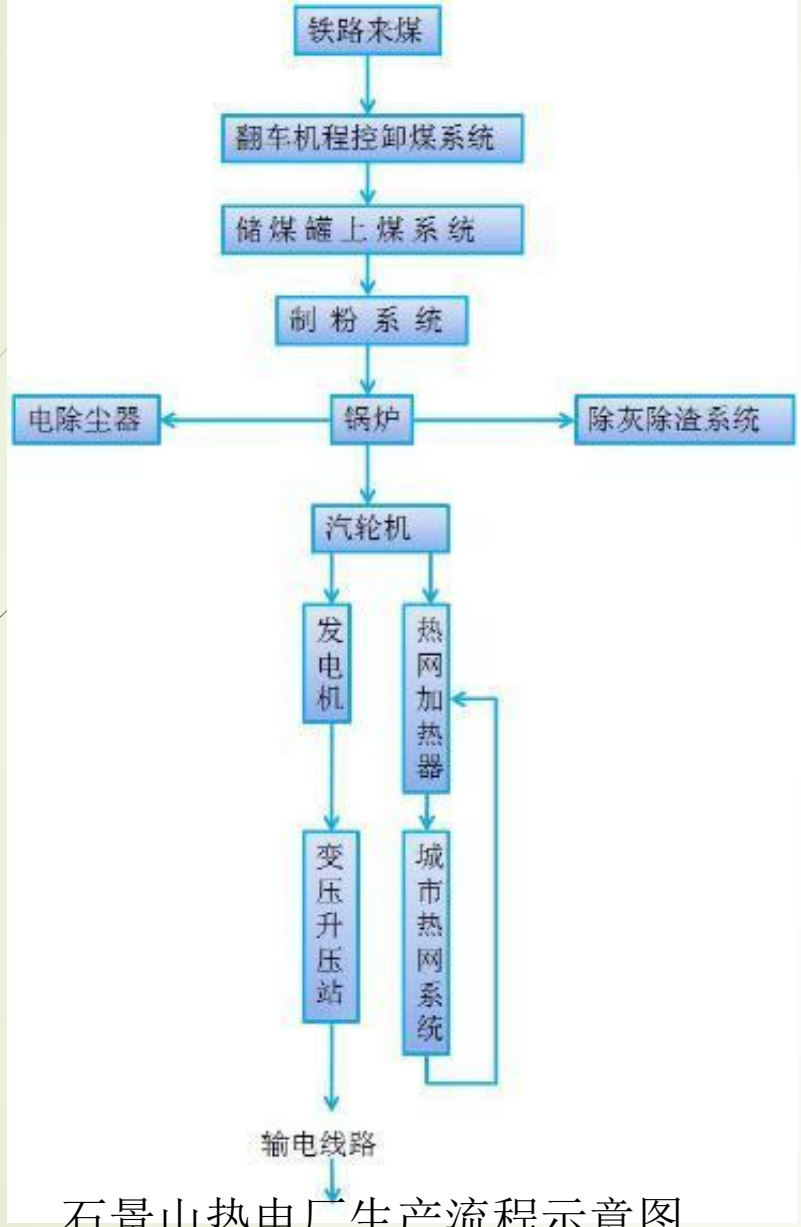
Balanced scorecard (2/2)

The Balanced Scorecard goes beyond standard financial measures to include the following additional perspectives: the customer perspective, the internal process perspective, and the learning and growth perspective.

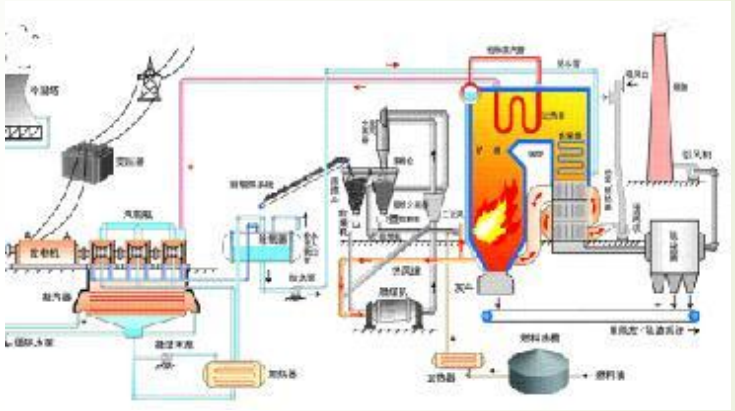
- **Financial perspective** - includes measures such as operating income, return on capital employed, and economic value added.
- **Customer perspective** - includes measures such as customer satisfaction, customer retention, and market share in target segments.
- **Business process perspective** - includes measures such as cost, throughput, and quality. These are for business processes such as procurement, production, and order fulfillment.
- **Learning & growth perspective** - includes measures such as employee satisfaction, employee retention, skill sets, etc.

IT平衡计分卡


面向用户
业务部门参与开发新系统的指标
用户满意度
系统界面的友好性
运营效率
系统维护活动占IT服务活动比率
系统差错频率
IT设备运营效率
系统和服务的可用性
系统吞吐量
不同类型用户响应时间
规定时间内完成任务比率
任务完成时间
面向未来发展
系统使用年限
软件可扩展性
员工培训费用占指出百分比
价值贡献
员工对IT系统评价
超出或低于预算比率
新系统的ROI
IT流程的成本—受益考虑
战略

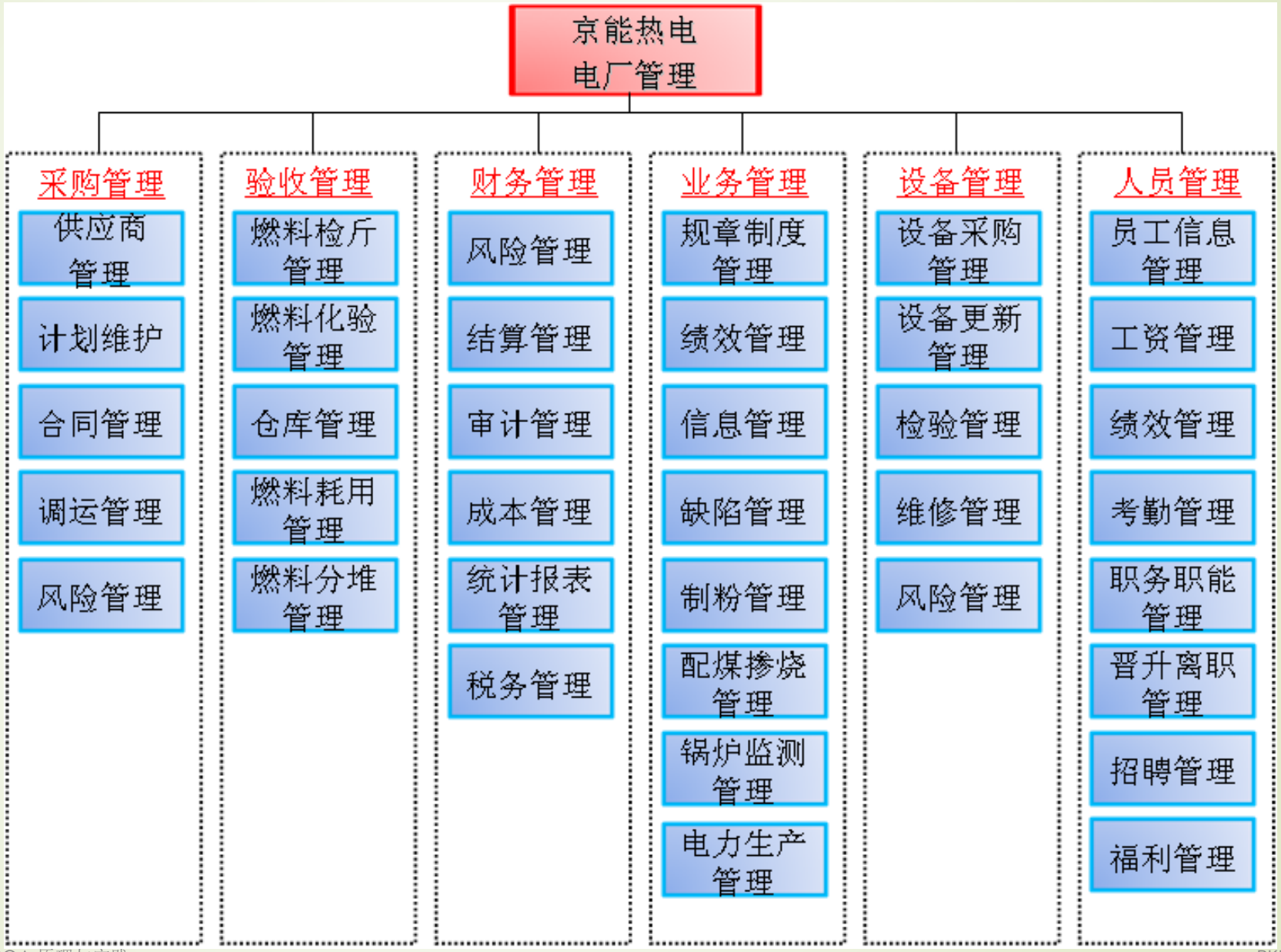


石景山热电厂生产流程示意图

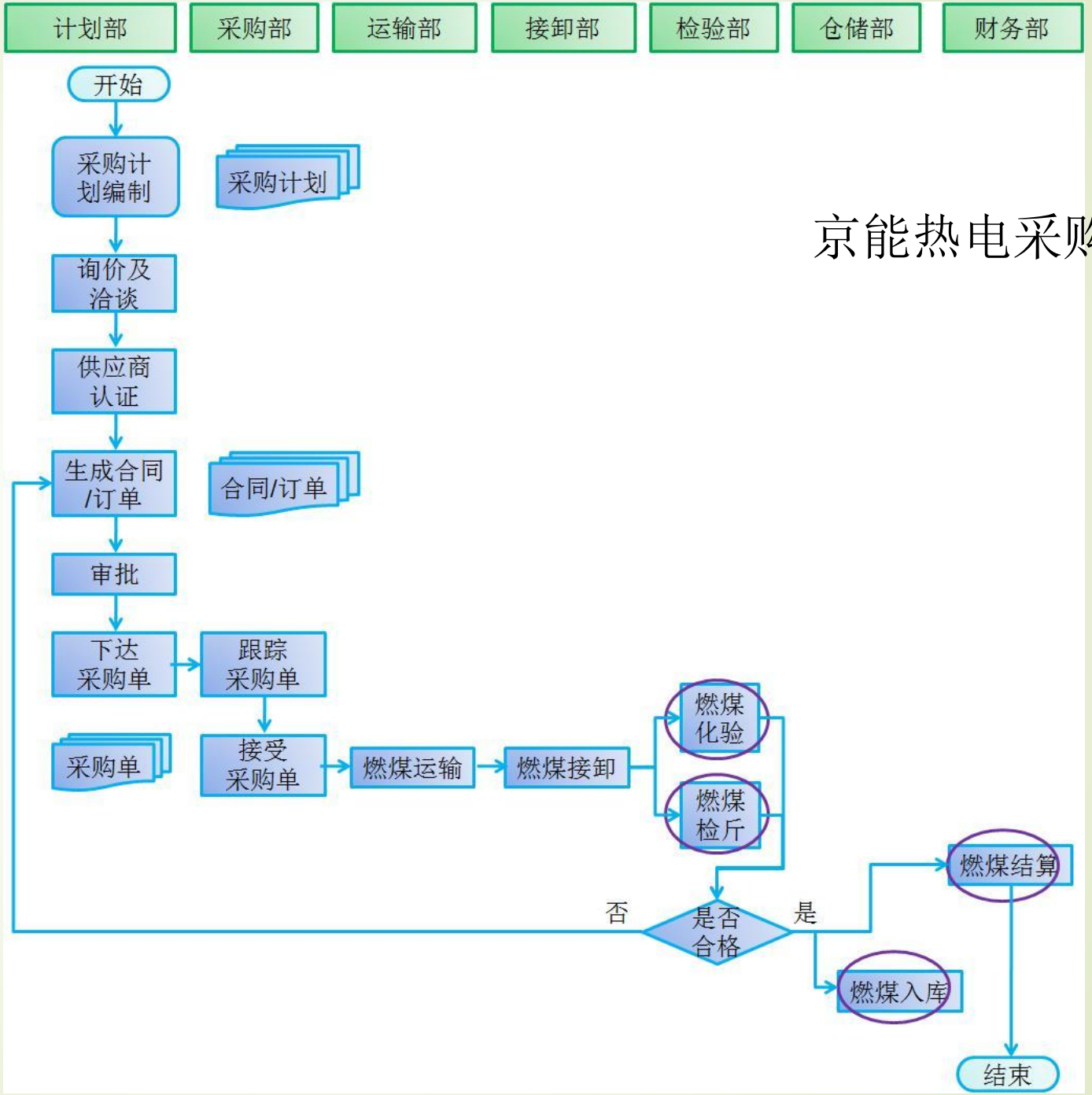


生产流程简图

京能热电电厂管理功能模块图



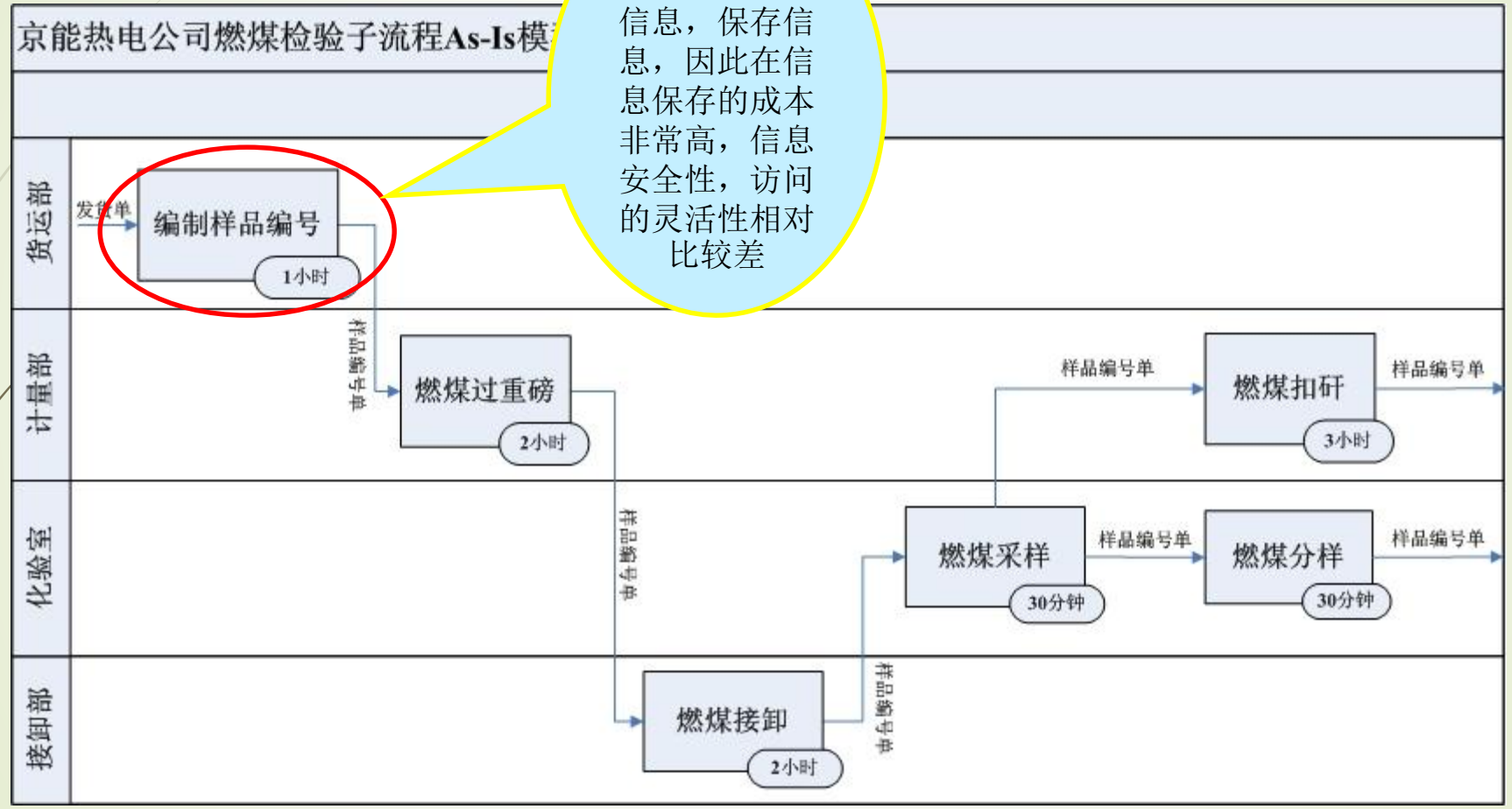
京能热电采购流程


计划部
采购部
运输部
接卸部
检验部
仓储部
财务部
开始
采购计划编制
采购计划
询价及洽谈
供应商认证
生成合同/订单
合同/订单
审批
下达采购单
跟踪采购单
采购单
接受采购单
燃煤运输
燃煤接卸
燃煤化验
燃煤检斤
否
是否合格
是
燃煤结算
燃煤入库
结束

京能热电组件化业务模型（Component Business Model）

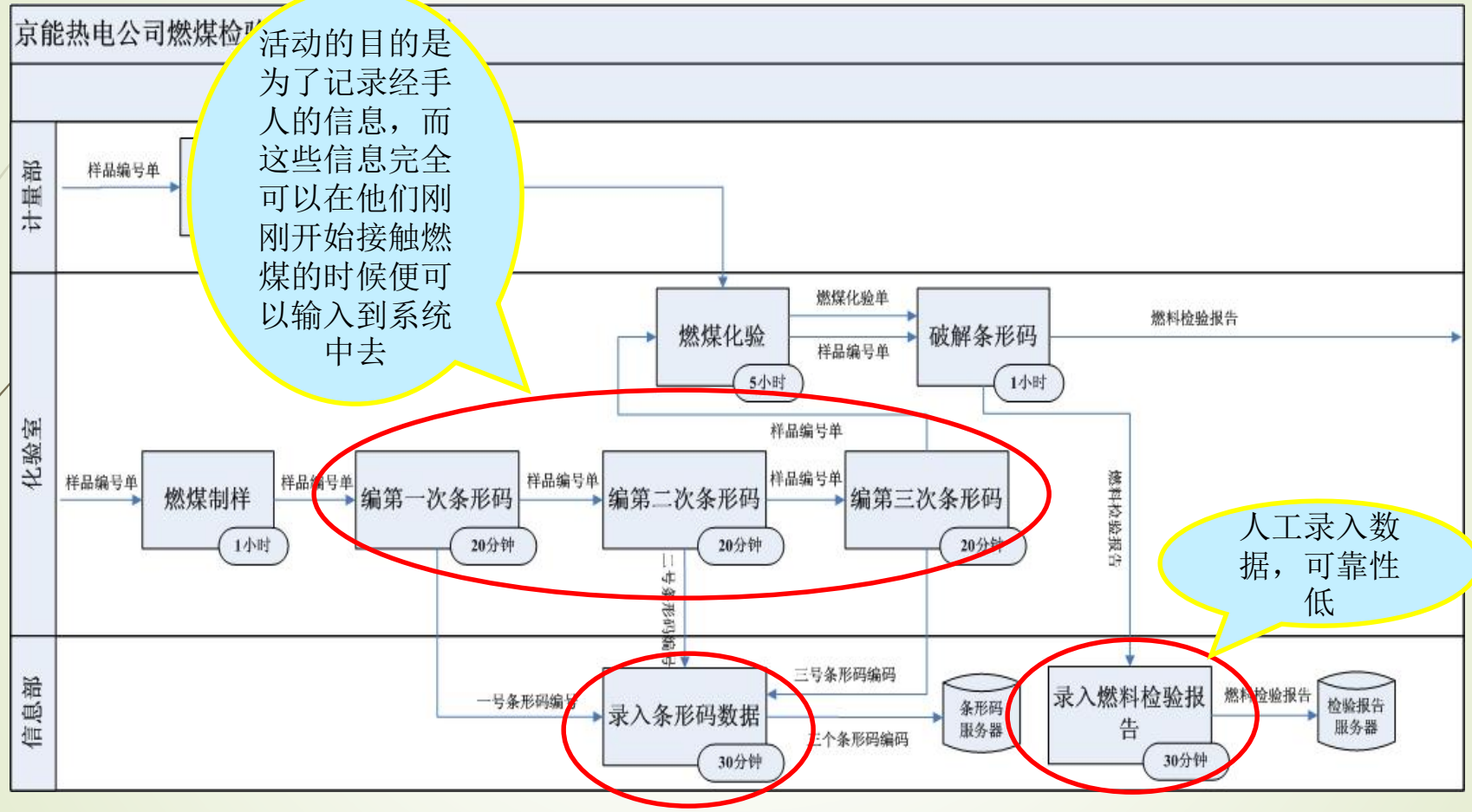
	供应链	燃料管理	财务管理	业务管理	设备管理	人力资源管理
Direct	供应链计划 供应商关系 燃料计划	燃料品牌战略 仓库设计	财务预测 财务计划	业务战略 业务计划 IT战略	采购策略 维护策略 校验计划	组织策略 薪酬计划 培训计划
Control	供应商管理 物流管理 合同管理 风险管理	燃料检验管理 仓库管理 燃料耗用管理	风险管理 审计管理 税务管理 成本管理	规章制度 绩效评估 信息管理 缺陷管理	检验管理 维修管理 设备采购管理	员工信息管理 绩效评估 工资管理 保险管理
Execute	燃煤采购 燃煤运输 风险分析	燃煤称重 煤质化验 燃煤入库 燃煤出仓 燃煤分堆	结　算 核　算 统计报表	制粉 配煤掺烧 入炉煤监测 锅炉优化燃烧 电力生产 电话呼叫	定期校验 设备维修 设备更新 设备购买	招聘 晋升 退休 员工培训 后勤运作

燃煤检验流程As-Is模型



纸质形式在业务流程中传递信息，保存信息，因此在信息保存的成本非常高，信息安全性，访问的灵活性相对比较差

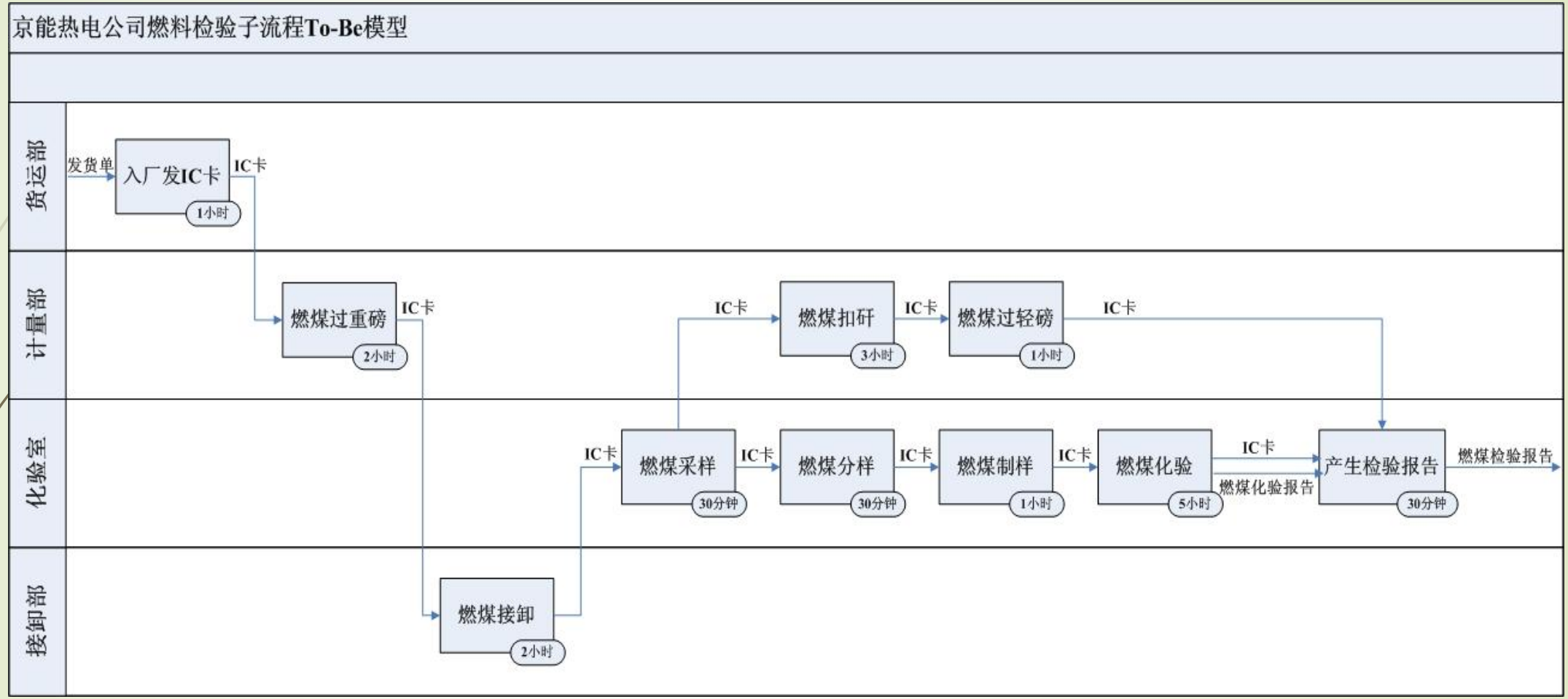
燃煤检验流程As-Is模型



燃煤检验As-Is缺陷分析

- 采用大量的纸质形式在业务流程中传递信息，保存信息，因此在信息保存的成本非常高，信息安全性，访问的灵活性相对比较差；
- 人工录入数据，可靠性低；
- 在燃煤进行制样之后有三个贴条形码的活动，这些活动的目的是为了记录经手人的信息，而这些信息完全可以在他们刚刚开始接触燃煤的时候便可以输入到系统中去；
- 在上述流程中燃煤化验的任务是在过轻磅和贴第三次条形码都完成之后才开始的，然而燃煤化验任务完全没有必要在等过轻磅之后再开始，因而可以与过轻磅一起并行发生，从而缩短整个流程的时间

燃煤检验流程To-Be模型



燃煤检验To-Be模型改进点

- 采用无纸化方式来传递数据，用IC卡代替样品编号单；
- 发IC卡时同时将供货人信息存入IC中；
- 燃煤计量等信息改为直接从计量仪器读入到IC卡中（增加相应的如写和连接设备）；
- 去除三次贴条形码活动；
- 去除数据录入人员

关键成功因素、关键绩效指标

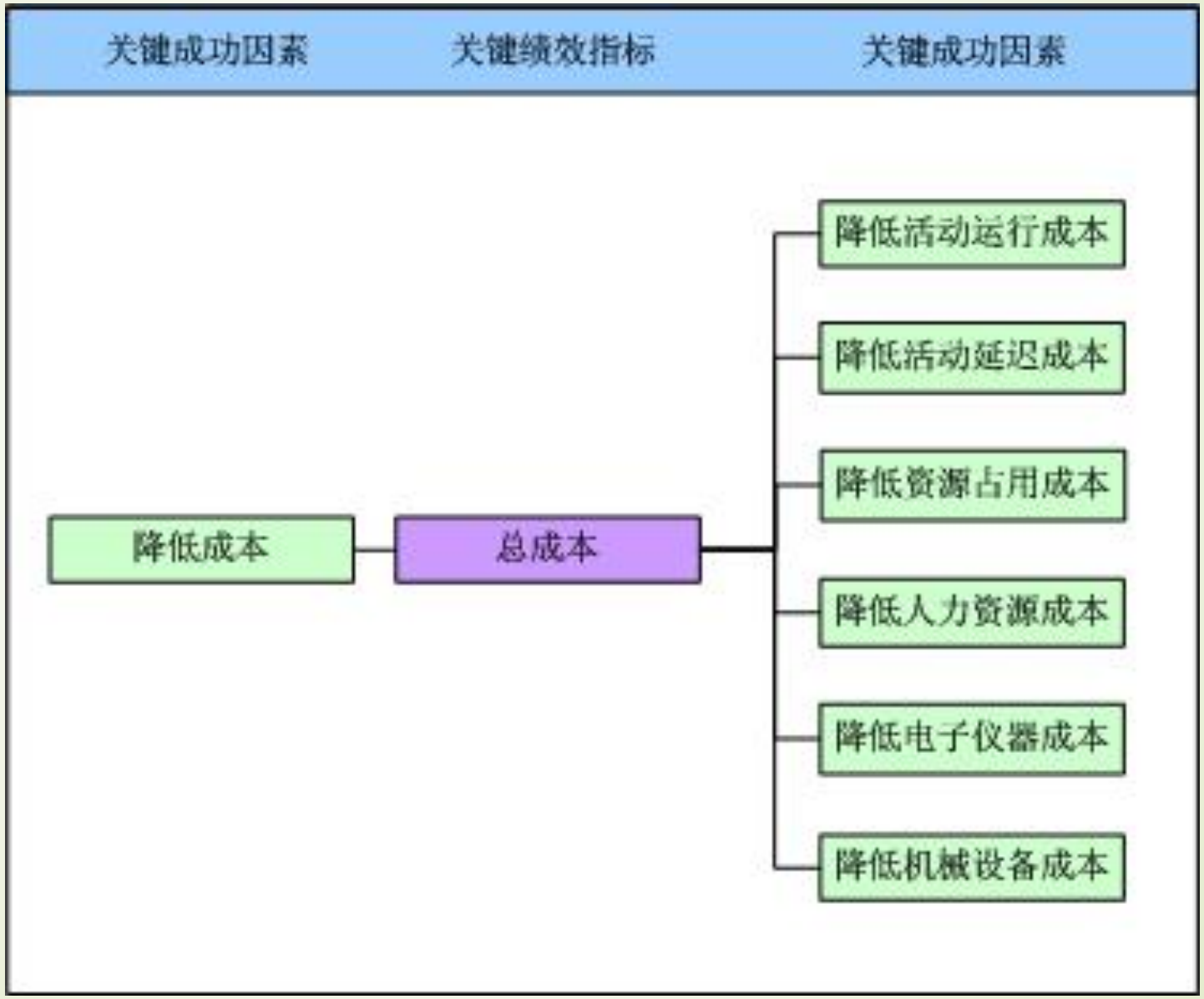
关键成功因素

- 企业实现战略目标的关键领域
- 反映了企业所期望达到的目标
- 将企业的战略目标转化为明确的行动内容

关键绩效指标

- 用来评估目标达成的量化指标
- 用来回答“如何评估成功？”

降低成本与KPI的因果关系



燃料检验子流程成本方面KPI

关键成功因素	关键绩效指标	主要负责部门
降低活动运行成本	录入条形码数据运行成本	信息部
	破解条形码运行成本	化验室
	编制样品编号运行成本	货运部
	编条形码运行成本	化验室
	录入燃料检验报告运行成本	信息部
降低活动延迟成本	录入条形码数据延迟成本	信息部
	破解条形码延迟成本	化验室
	编制样品编号延迟成本	货运部
降低资源占用成本	燃煤接卸资源占用成本	接卸部
	燃煤采样资源占用成本	化验室
	录入条形码数据资源占用成本	信息部
	破解条形码资源占用成本	化验室
	编制样品编号资源占用成本	货运部
	编条形码资源占用成本	化验室
	录入燃料检验报告资源占用成本	信息部
	燃煤制样资源占用成本	化验室
	燃煤分样资源占用成本	化验室
	燃煤过重磅资源占用成本	计量部
	燃煤扣矸资源占用成本	计量部

燃料检验子流程As-Is和To-Be模型对比分析表

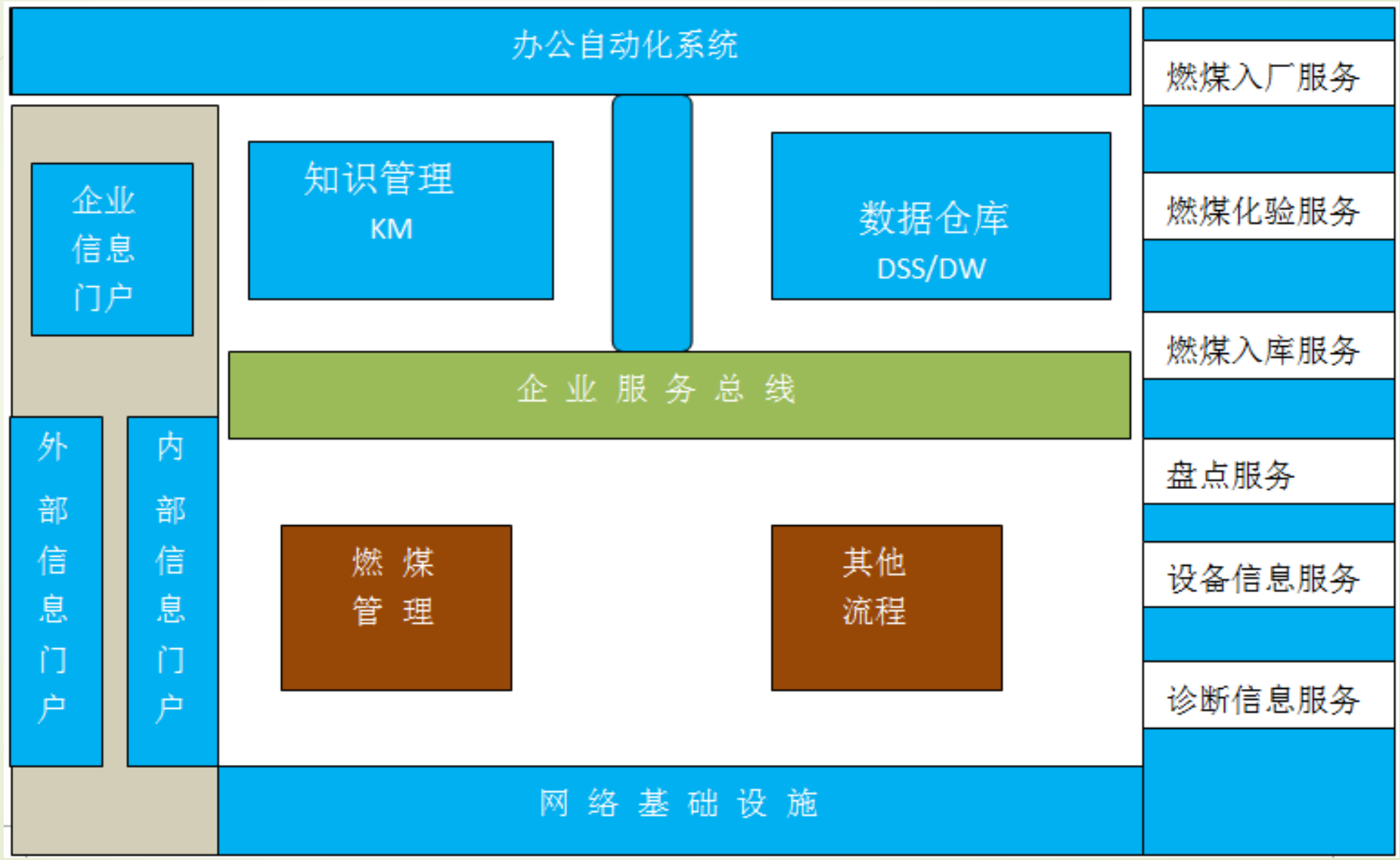
燃料检验子流程	As-Is模型	To-Be模型
平均运行成本	7,060.00 RMB	7,100.00 RMB
平均延迟成本	4,649.07 RMB	3,048.14 RMB
平均资源占用成本	22,884.95 RMB	14,858.68 RMB
平均成本	34,594.02 RMB	24,006.82 RMB
平均工作时间	21小时	19小时
平均资源占用时间	4天2小时10分	3天15小时
平均延迟时间	3天4小时58分53秒	15小时28分53秒
平均耗费时间	3天21小时58分53秒	1天4小时28分53秒
人力资源数量	27人	25人
人力资源成本	112000 RMB	103500 RMB
电子仪器数量	15部	15部
电子仪器成本	24800 RMB	24800 RMB
机械设备数量	7台	7台
机械设备成本	240000 RMB	240000 RMB
总体成本	411394.02 RMB	392306.82 RMB

服务总览

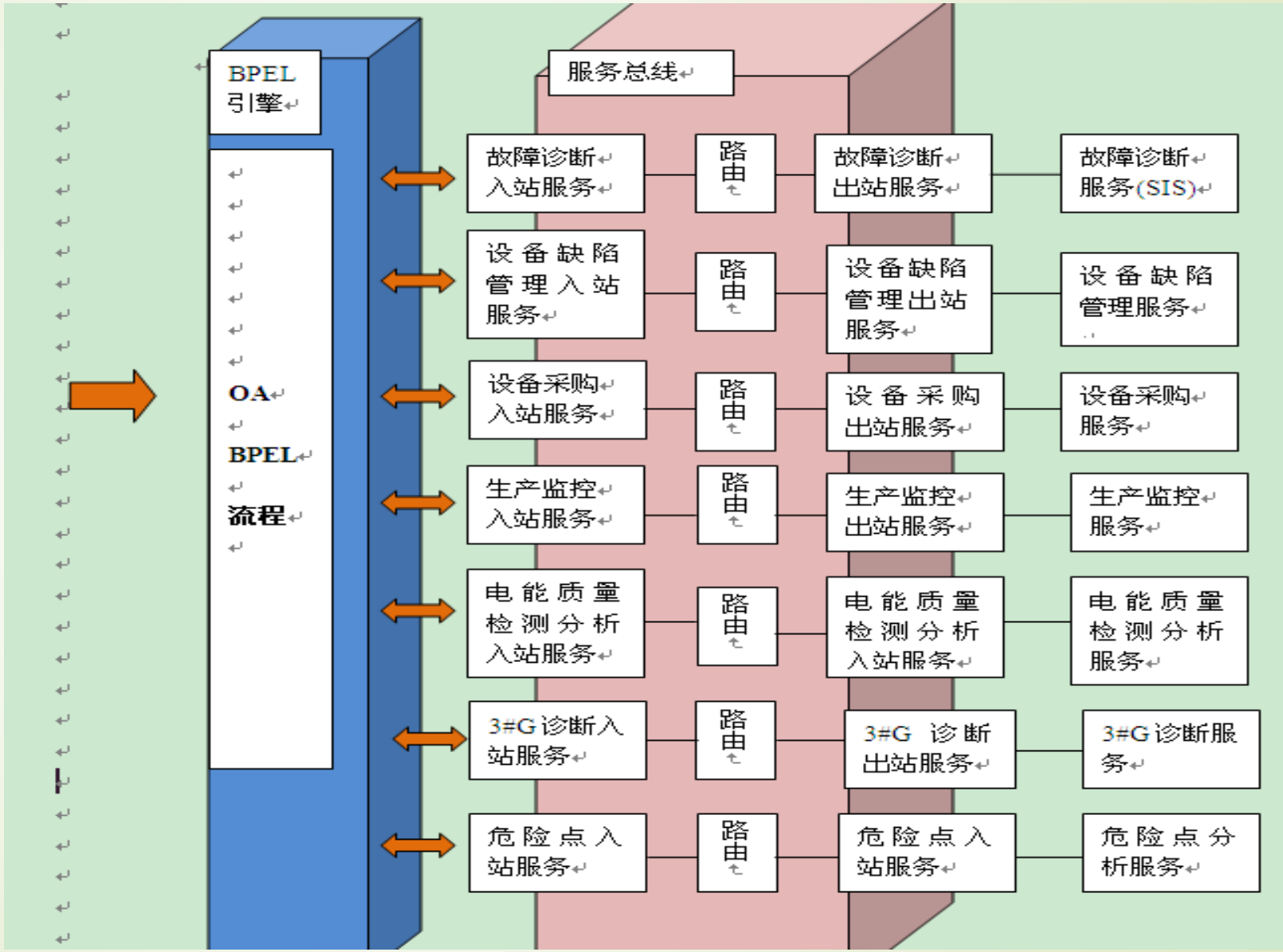
- 在完成功能分解后，以表格的方式就每个功能分析其特征，看能否继续分解。
- 任意的可继续分解的节点，将在流程分解的章节中逐一描述其构成关系和流转。
- 任一节点均在功能定义的章节中逐一描述。
- 通过对每个功能点的进一步分析，明确其是否能够被系统支持、优先级等要素。

功能分类:	相关功能:	服务描述:	是否原子功能:
A1.1燃煤检验管理	A1.1.1编制样品编号	对新到的燃煤样品进行编号处理	是
	A1.1.2过重磅	燃煤样品通过重磅称重	是
	A1.1.3燃煤采样	采样燃煤,为检验之用	是
	A1.1.4煤厂扣矸	将燃煤样品进行扣矸操作,去除杂质	是
	A1.1.5燃煤过轻磅	燃煤样品通过轻磅称重	是
	A1.1.6获得燃煤分样信息	测试获得燃煤分样信息	是
	A1.1.7获得燃煤制样信息	测试获得燃煤制样信息	是
	A1.1.8燃煤化验	化验燃煤样品	是
	A1.1.9检验报告生成	生成最后的检验报告并输出	是

信息化愿景概况


办公自动化系统
企业信息门户
知识管理
KM
数据仓库
DSS/DW
企 业 服 务 总 线
外部信息门户
内部信息门户
燃 煤 管 理
其他流程
网 络 基 础 设 施
燃煤入厂服务
燃煤化验服务
燃煤入库服务
盘点服务
设备信息服务
诊断信息服务

ESB架构设计


BPEL
引擎
OA
BPEL
流程
服务总线
故障诊断
入站服务
路由
故障诊断
出站服务
故障诊断
服务(SIS)
设备缺陷
管理入站
服务
路由
设备缺陷
管理出站
服务
设备缺陷
管理服务
设备采购
入站服务
路由
设备采购
出站服务
设备采购
服务
生产监控
入站服务
路由
生产监控
出站服务
生产监控
服务
电能质量
检测分析
入站服务
路由
电能质量
检测分析
入站服务
电能质量
检测分析
服务
3#G诊断入
站服务
路由
3#G 诊断
出站服务
3#G诊断服
务
危险点入
站服务
路由
危险点入
站服务
危险点分
析服务

燃煤检验管理对应Portal页面

燃煤化验流程

编制样品编号 | 燃煤采样 | 燃煤过轻磅 | 获得燃煤制样信息 | 检验报告生成
过重磅 | 煤 扣叶 | 获得燃煤分样信息 | 燃煤化验

燃煤采样

编号# 111燃煤采样进行中
- 编号# 112燃煤采样进行中
- 编号# 113燃煤采样进行完毕
- 编号# 114燃煤采样进行完毕
- 编号# 115燃煤采样进行完毕

燃煤过重磅

编号# 221燃煤过重磅
- 编号# 222燃煤过重磅
- 编号# 223燃煤过重磅完毕
- 编号# 224燃煤过重磅完毕
- 编号# 225燃煤过重磅完毕

编制样品编号

- **燃煤样品编号# 331**
- 燃煤样品编号# 332
- 燃煤样品编号# 333
- 燃煤样品编号# 334
- 燃煤样品编号# 335
- 燃煤样品编号# 336
- 燃煤样品编号# 337
- 燃煤样品编号# 338

输入样品编号 Go

获得燃煤分样信息

- **编号# 221燃煤分样信息**
- 编号# 222燃煤分样信息
- 编号# 223燃煤分样信息
- 编号# 224燃煤分样信息
- 编号# 225燃煤分样信息

获得燃煤制样信息

- **编号# 221燃煤制样信息**
- 编号# 222燃煤制样信息
- 编号# 223燃煤制样信息
- 编号# 224燃煤制样信息
- 编号# 225燃煤制样信息
- 编号# 226燃煤制样信息

燃煤过轻磅

- **编号# 331燃煤过轻磅**
- 编号# 332燃煤过轻磅
- 编号# 333燃煤过轻磅
- 编号# 334燃煤过轻磅
- 编号# 335燃煤过轻磅

燃煤化验

- 编号# 551号燃煤化验结果
- 编号# 552号燃煤化验结果
- 编号# 553号燃煤化验结果
- 编号# 554号燃煤化验结果
- 编号# 555号燃煤化验结果
- 编号# 556号燃煤化验结果
- 编号# 557号燃煤化验结果
- 编号# 558号燃煤化验结果

输入燃煤编号 Go

检验报告生成

- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告
- 编号# 551号燃煤检验报告

输入检验报告编号 Go

Agenda

- A case study: automation of reimbursement process
  - What problems are we expected to solve?
  - What solution might we have?
  - What have we learnt and What are we going to learn?
- SOA principles (OASIS: Organization for the Advancement of Structured Information Standards)
  - What is a service?
  - SOA Concepts Example
  - What is SOA? And Why?
  - SOA Reference Model
  - Web services approach to a service-oriented architecture
  - SOA principles

What is Service?

- OASIS Reference Model defines as follows:

  - A mechanism to enable access to one or more *capabilities*, where the access is provided using a *prescribed interface* and is exercised consistent with *constraints and policies* as specified by the *service description*.

Principal Concepts of Services

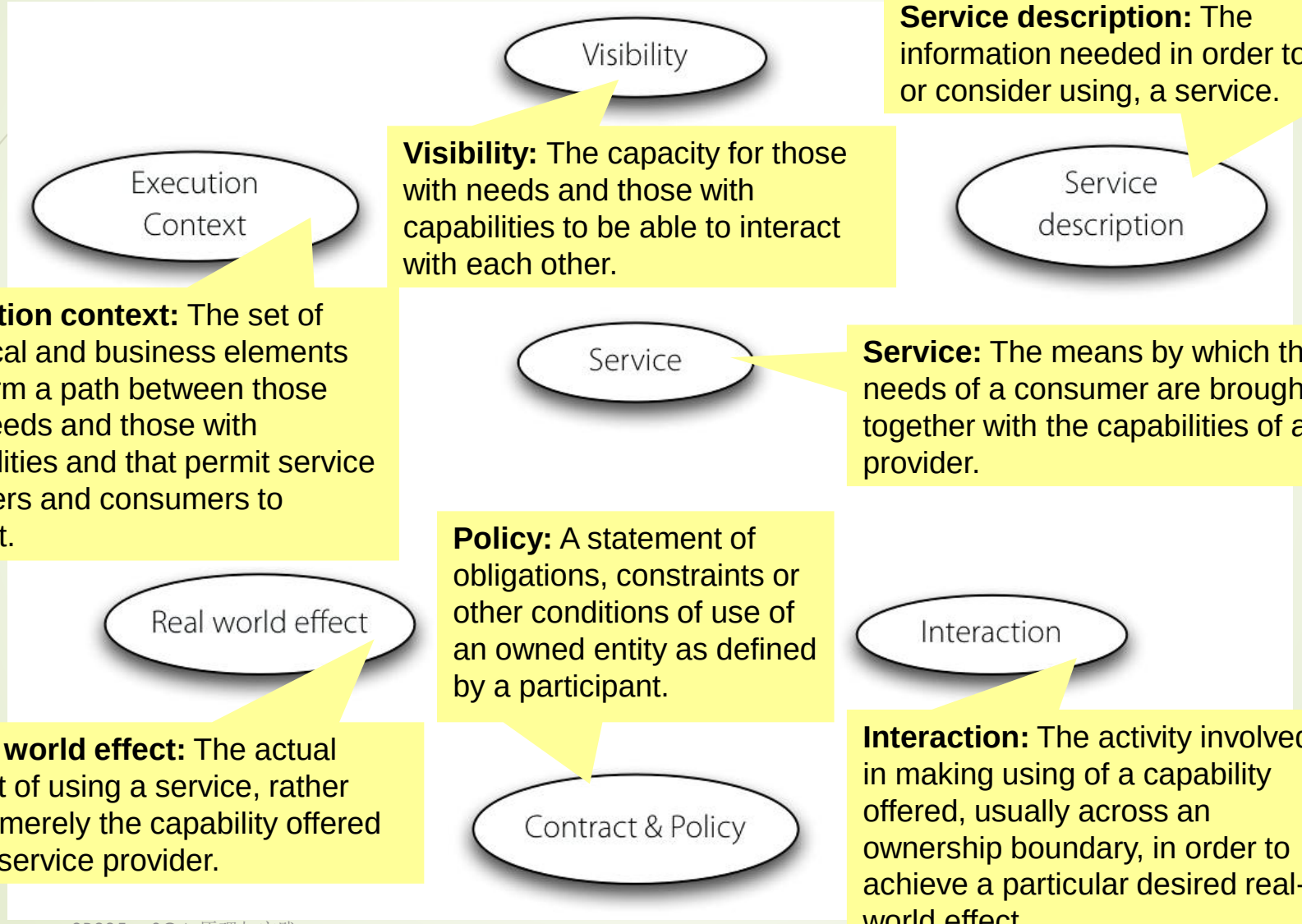
- Visibility
- Interaction
- Real World Effect

Visibility, Interaction, and Real World Effect address the **dynamic** aspects of services (interactions with services).

- Service Description
- Execution Context
- Contract & Policy

Service Description, Execution Context, Contract & Policy address **static** aspects.

Principal concepts



Service description: The information needed in order to use, or consider using, a service.

Visibility: The capacity for those with needs and those with capabilities to be able to interact with each other.

Execution context: The set of technical and business elements that form a path between those with needs and those with capabilities and that permit service providers and consumers to interact.

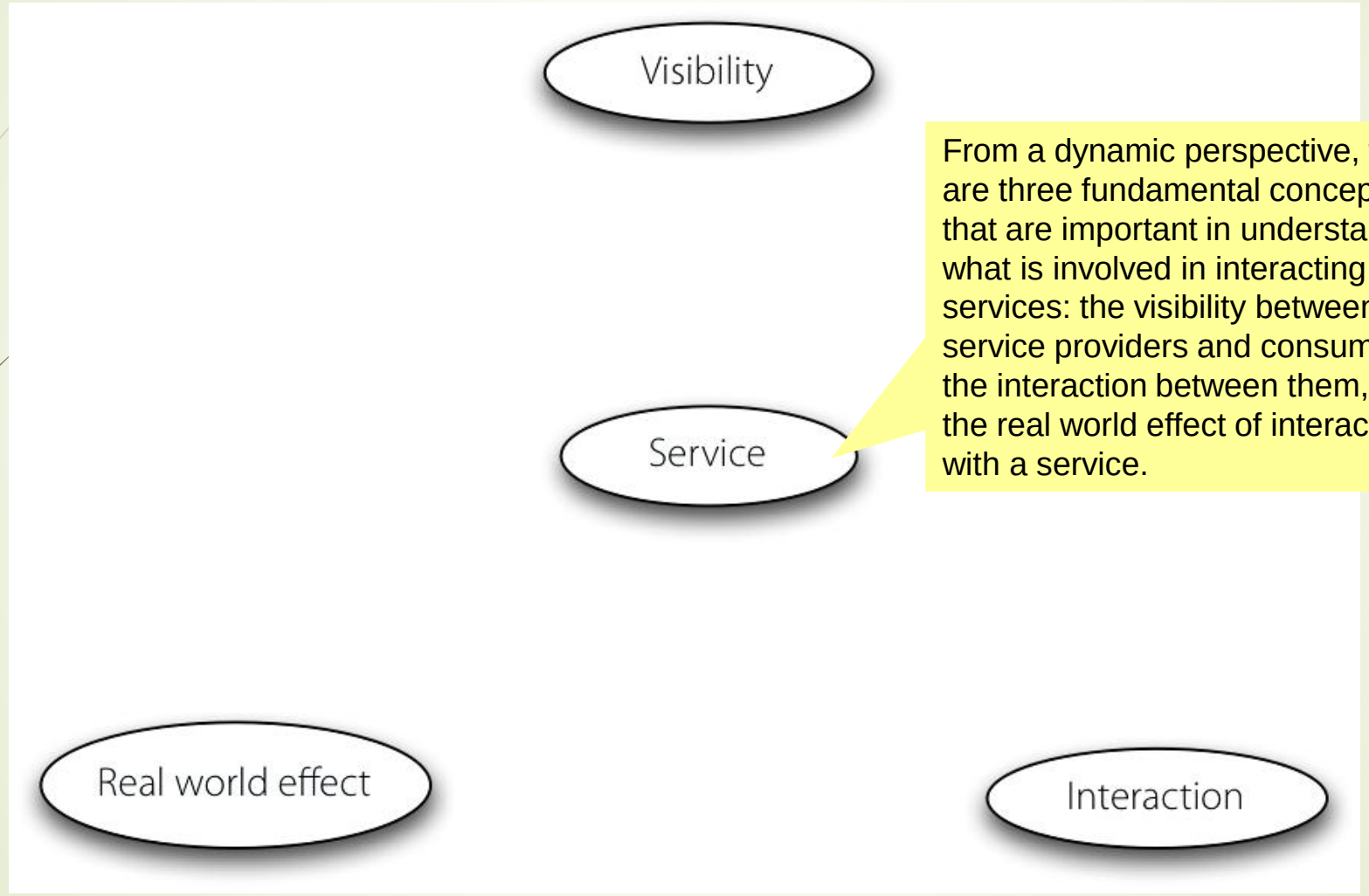
Service: The means by which the needs of a consumer are brought together with the capabilities of a provider.

Policy: A statement of obligations, constraints or other conditions of use of an owned entity as defined by a participant.

Real world effect: The actual result of using a service, rather than merely the capability offered by a service provider.

Interaction: The activity involved in making using of a capability offered, usually across an ownership boundary, in order to achieve a particular desired real-world effect.

Dynamics of services



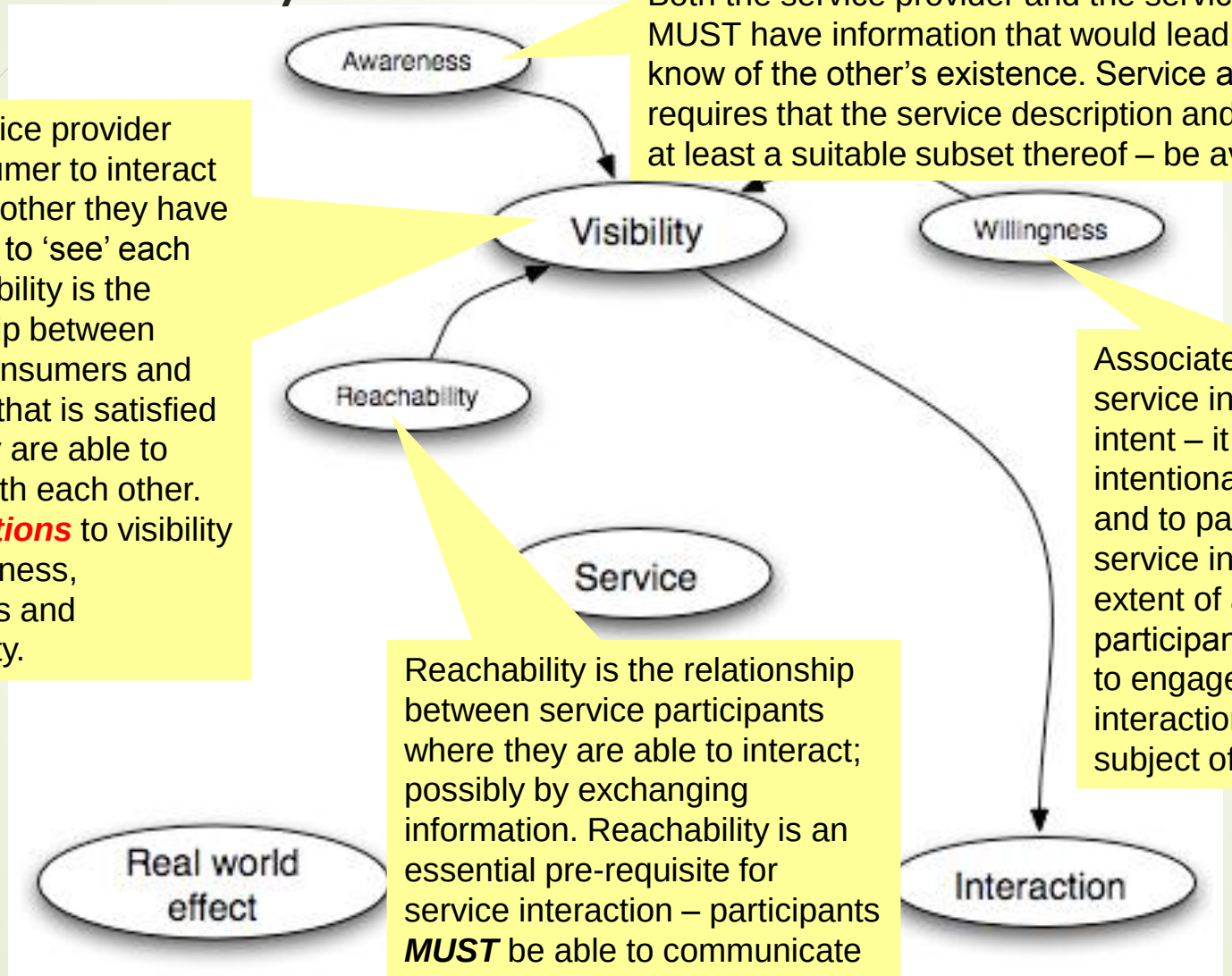
From a dynamic perspective, there are three fundamental concepts that are important in understanding what is involved in interacting with services: the visibility between service providers and consumers, the interaction between them, and the real world effect of interacting with a service.

Visibility

- The capacity for those with needs and those with capabilities to be able to interact with each other.

- This is typically done by providing descriptions for such aspects as
  - functions and technical requirements,
  - related constraints and policies, and
  - mechanisms for access or response.

Visibility



Both the service provider and the service consumer MUST have information that would lead them to know of the other's existence. Service awareness requires that the service description and policy – or at least a suitable subset thereof – be available.

For a service provider and consumer to interact with each other they have to be able to 'see' each other. Visibility is the relationship between service consumers and providers that is satisfied when they are able to interact with each other. ***Preconditions*** to visibility are awareness, willingness and reachability.

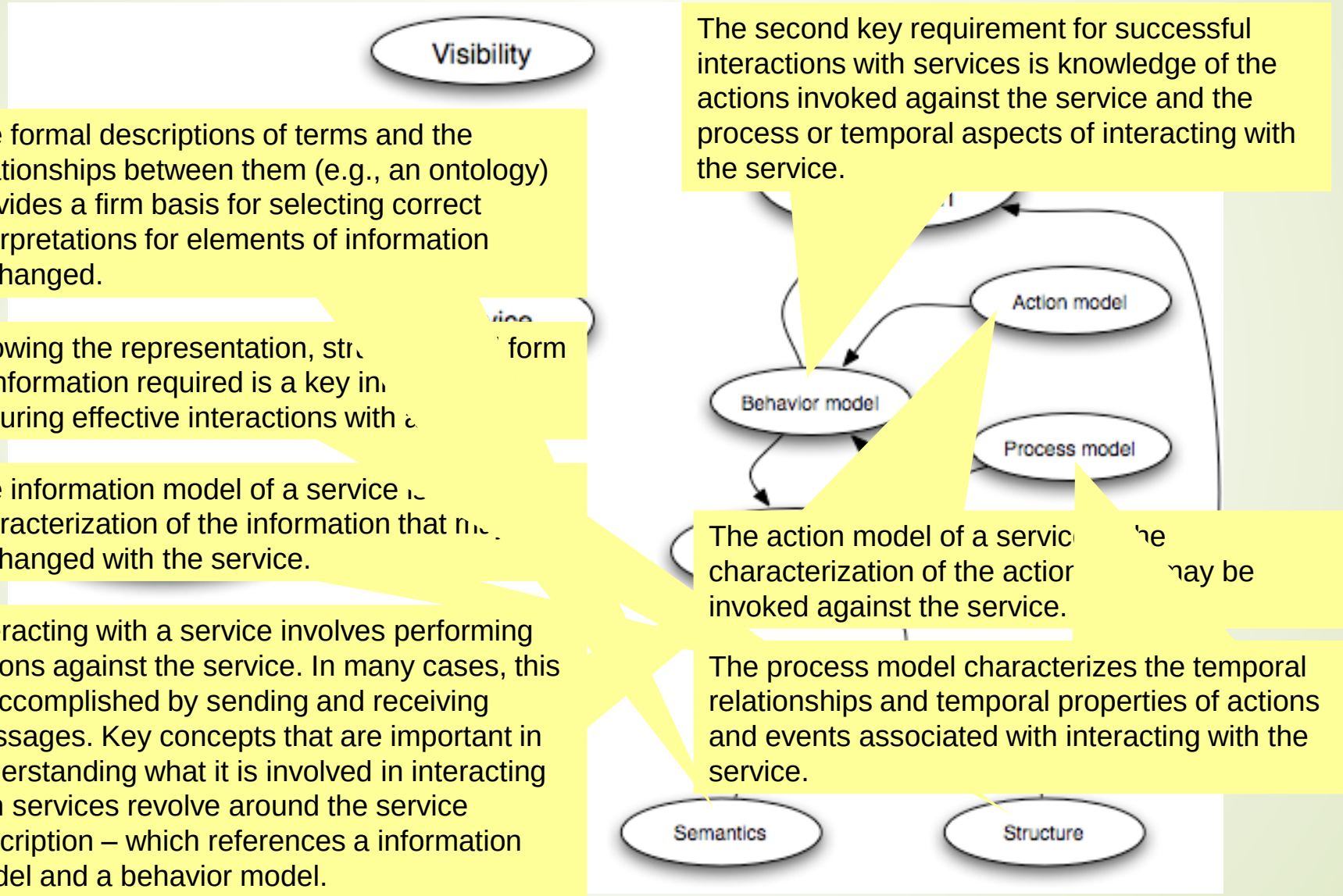
Associated with all service interactions is intent – it is an intentional act to initiate and to participate in a service interaction. The extent of a service participant's willingness to engage in service interactions may be the subject of policies.

Reachability is the relationship between service participants where they are able to interact; possibly by exchanging information. Reachability is an essential pre-requisite for service interaction – participants ***MUST*** be able to communicate with each other.

Interaction

- Refers to the interaction between service providers and consumers.
- Typically mediated by the exchange of messages, an interaction proceeds through a series of information exchanges and invoked actions.
- The result of an interaction is a real world effect.

Interacting with services



The second key requirement for successful interactions with services is knowledge of the actions invoked against the service and the process or temporal aspects of interacting with the service.

The formal descriptions of terms and the relationships between them (e.g., an ontology) provides a firm basis for selecting correct interpretations for elements of information exchanged.

Knowing the representation, str form of information required is a key in ensuring effective interactions with

The information model of a service is characterization of the information that m exchanged with the service.

The action model of a servic he characterization of the actior nay be invoked against the service.

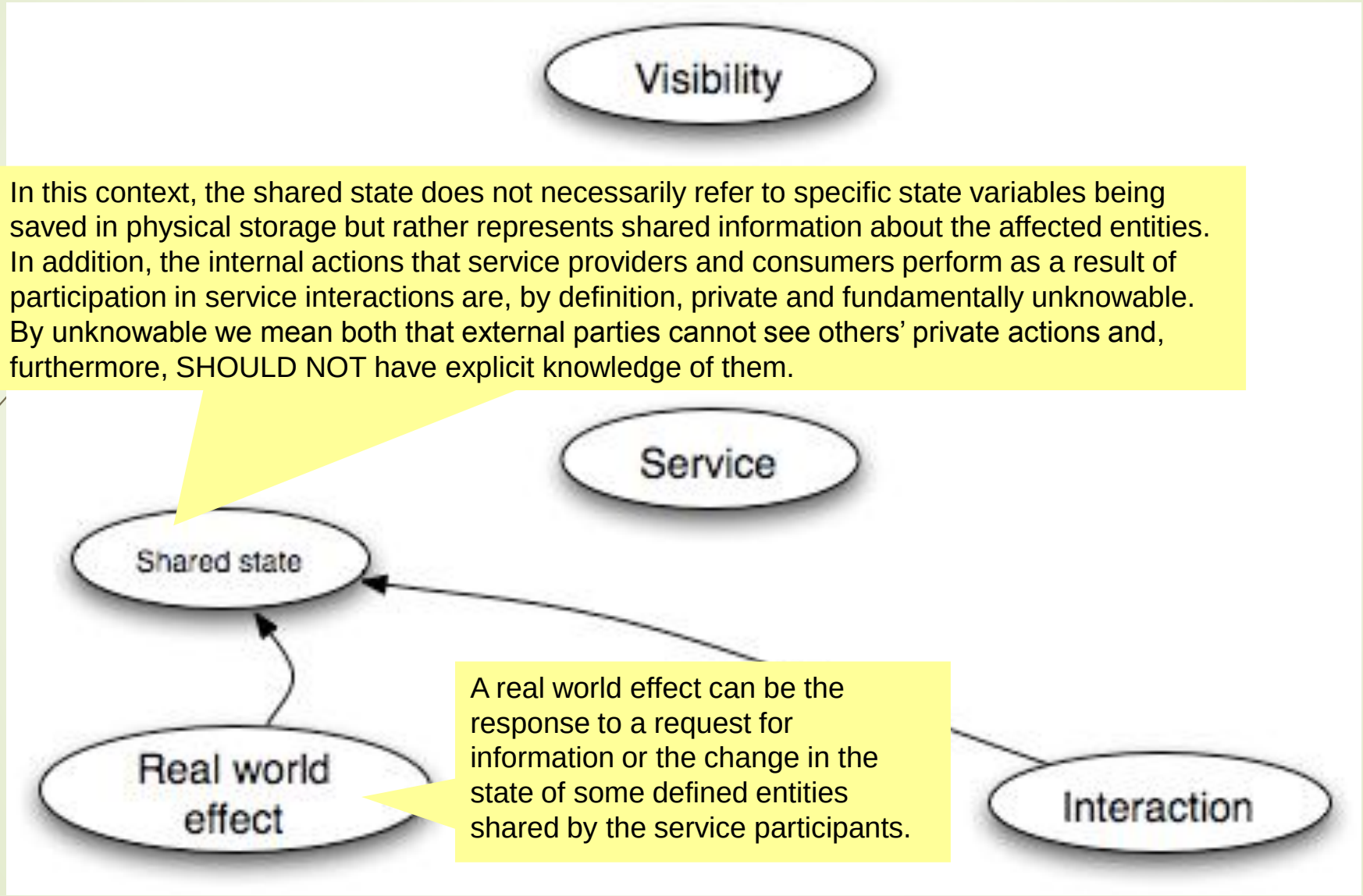
Interacting with a service involves performing actions against the service. In many cases, this is accomplished by sending and receiving messages. Key concepts that are important in understanding what it is involved in interacting with services revolve around the service description – which references a information model and a behavior model.

The process model characterizes the temporal relationships and temporal properties of actions and events associated with interacting with the service.

Real World Effect

- The actual result of using a service.
- This may be the return of information or the change in the state of entities (known or unknown) that are involved in the interaction.

Real world effect



Service Description

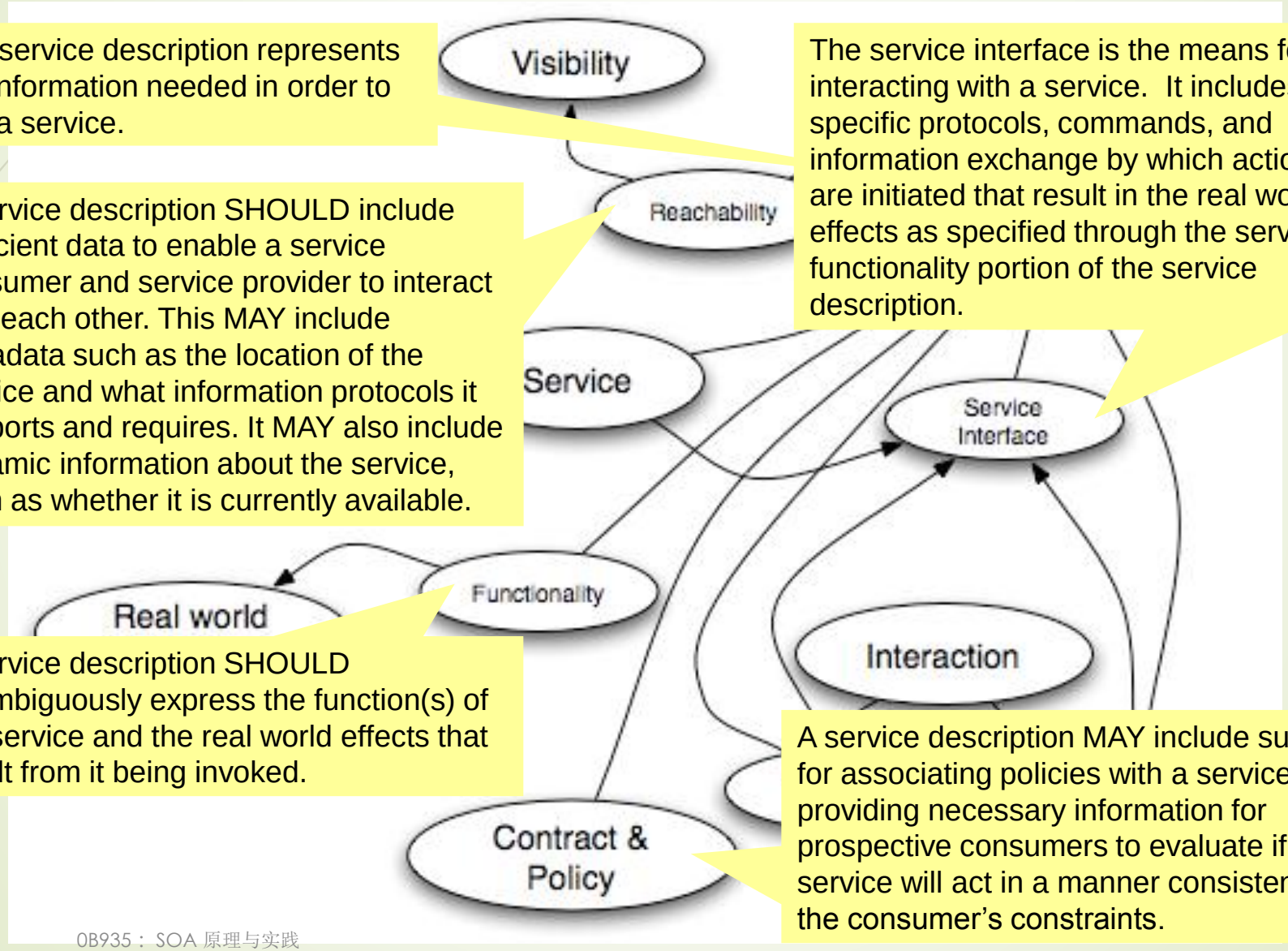
- The information needed in order to use, or consider using a service.
- The purpose of description is to facilitate interaction and visibility,
  - particularly when the participants are in different ownership domains, between participants in service interactions.

Service description

The service description represents the information needed in order to use a service.

A service description SHOULD include sufficient data to enable a service consumer and service provider to interact with each other. This MAY include metadata such as the location of the service and what information protocols it supports and requires. It MAY also include dynamic information about the service, such as whether it is currently available.

A service description SHOULD unambiguously express the function(s) of the service and the real world effects that result from it being invoked.



The service interface is the means for interacting with a service. It includes the specific protocols, commands, and information exchange by which actions are initiated that result in the real world effects as specified through the service functionality portion of the service description.

A service description MAY include support for associating policies with a service and providing necessary information for prospective consumers to evaluate if a service will act in a manner consistent with the consumer's constraints.

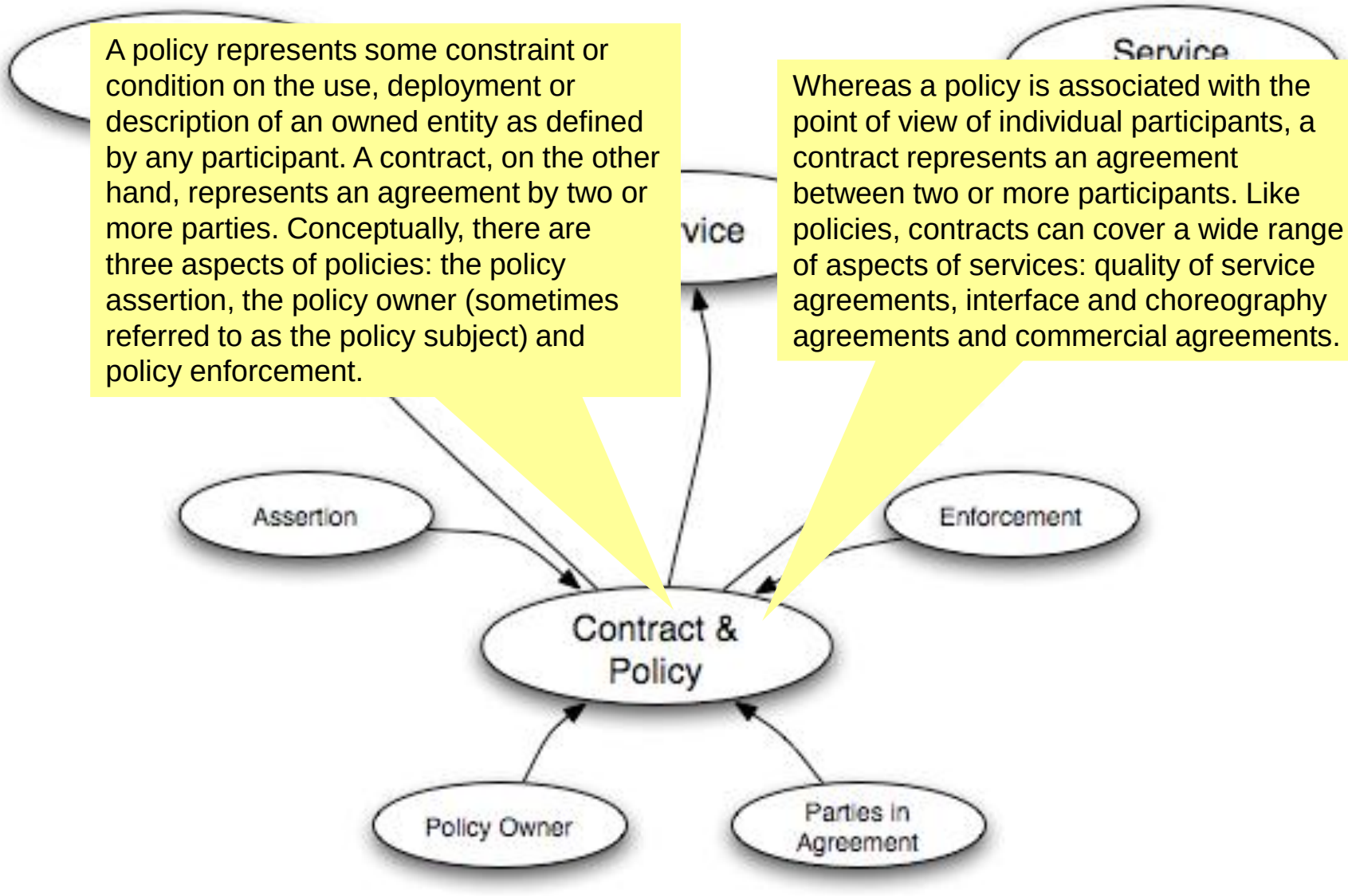
Contract & Policy

- A policy represents some constraint or condition on the use, deployment or description of an owned entity as defined by any participant
- A contract represents an agreement by two or more parties.

Policies and contracts

A policy represents some constraint or condition on the use, deployment or description of an owned entity as defined by any participant. A contract, on the other hand, represents an agreement by two or more parties. Conceptually, there are three aspects of policies: the policy assertion, the policy owner (sometimes referred to as the policy subject) and policy enforcement.

Whereas a policy is associated with the point of view of individual participants, a contract represents an agreement between two or more participants. Like policies, contracts can cover a wide range of aspects of services: quality of service agreements, interface and choreography agreements and commercial agreements.

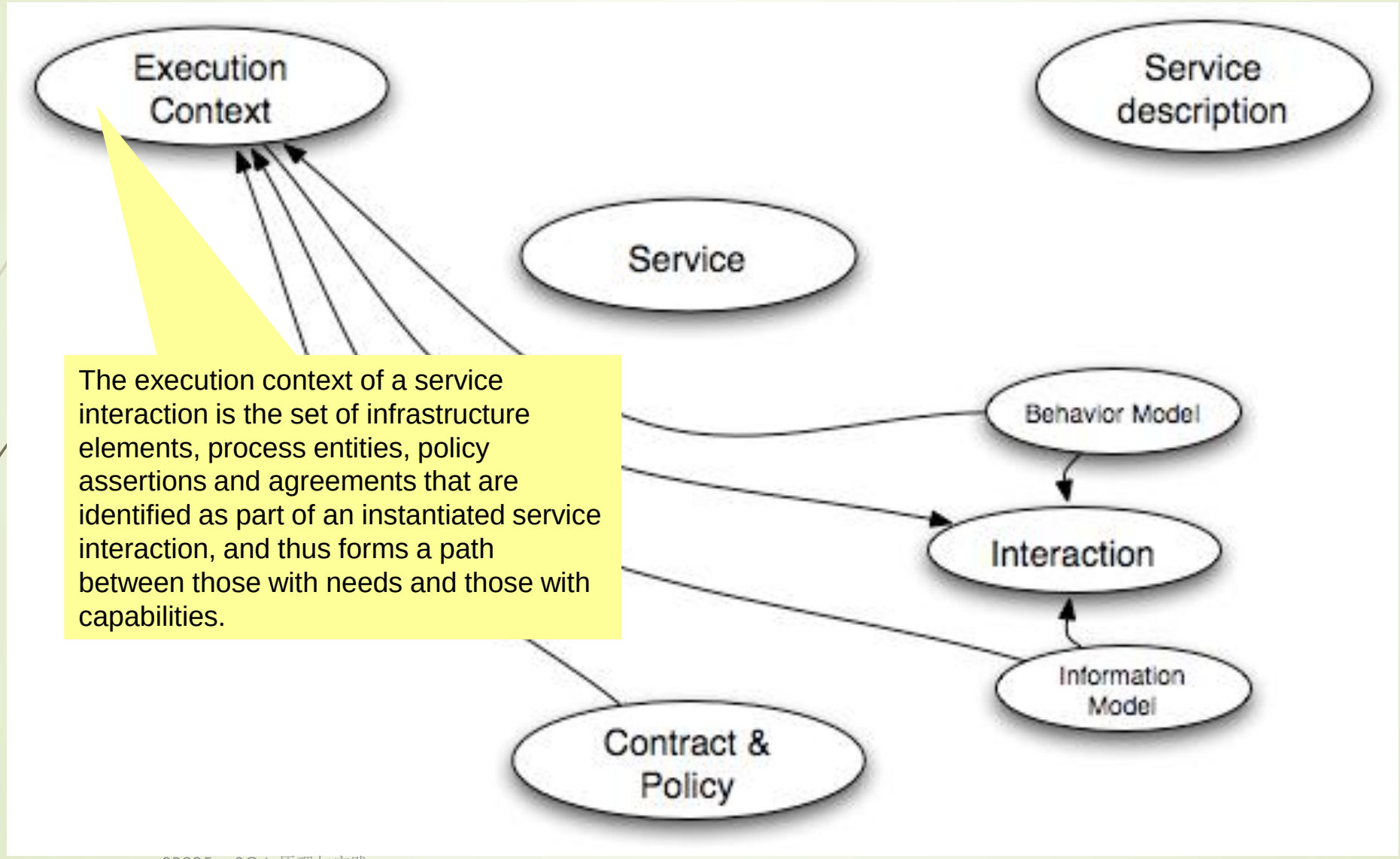


Execution Context

- The set of *technical* and *business* elements
  - that form a path between those with needs and those with capabilities, and
  - that permit service providers and consumers to interact.

- All interactions are grounded in a particular execution context,
  - which permits service providers and consumers to interact and provides *a decision point* for any policies and contracts that may be in force.

Execution context



The execution context of a service interaction is the set of infrastructure elements, process entities, policy assertions and agreements that are identified as part of an instantiated service interaction, and thus forms a path between those with needs and those with capabilities.

SOA Concepts Example (1/5)

- An electric utility has the capacity to generate and distribute electricity *(the underlying capability)*.
  - The wiring from the electric company's distribution grid *(the service)* provides the means to supply electricity to support typical usage for a residential consumer's house *(service functionality)*, and
  - a consumer accesses electricity generated *(the output of invoking the service)* via a wall outlet *(service interface)*.

SOA Concepts Example (2/5)

- In order to use the electricity, a consumer needs to understand
  - what type of plug to use, what is the voltage of the supply, and possible limits to the load;
  - the utility presumes that the customer will only connect devices that are compatible with the voltage provided and load supported; and
  - the consumer in turn assumes that compatible consumer devices can be connected without damage or harm.

service technical assumptions

SOA Concepts Example (3/5)

- A residential or business user will need to open an account with the utility in order to use the supply *(service constraint)* and the utility will meter usage and expects the consumer to pay for use at the rate prescribed *(service policy)*.

- When the consumer and utility agree on constraints and policies *(service contract)*,
  - the consumer can receive electricity using the service as long as the electricity distribution grid and house connection remain intact (e.g., a storm knocking down power lines would disrupt distribution)
  - and the consumer can have payment sent (e.g., a check by mail or electronic funds transfer) to the utility *(reachability)*.

SOA Concepts Example (4/5)

- Another person (for example, a visitor to someone else's house) may use a contracted supply
  - without any relationship with the utility or any requirement to also satisfy the initial service constraint (e.g., ***reachability*** only requires intact electricity distribution)
  - but would nonetheless be expected to be compatible with the service interface.

SOA Concepts Example (5/5)

- In certain situations (for example, excessive demand), a utility may limit supply or institute rolling blackouts *(service policy)*.
- A consumer might lodge a formal complaint if this occurred frequently *(consumer's implied policy)*.
- If the utility required every device to be hardwired to its equipment, the underlying capability would still be there but this would be a very different service and have a very different service interface.

What is SOA? (1/4)

- According to the OASIS SOA-RM (Reference Model) specification:
  - SOA is a paradigm for organizing and utilizing ***distributed capabilities*** that may be under the control of different ownership domains.
  - It provides ***a uniform means*** to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable ***preconditions and expectations***.
- The SOA-RM specification bases its definition of SOA around the concept of "***needs*** and ***capabilities***",
  - where SOA provides a mechanism for matching needs of service consumers with capabilities provided by service providers.

http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf

What is SOA? (2/4)

- SOA is a design for linking business and computational resources (principally organizations, applications and data) on demand
  - to achieve the desired results for service consumers (which can be end users or other services).

http://en.wikipedia.org/wiki/Service-oriented_architecture#SOA_definitions

What is SOA? (3/4)

- In Service-Oriented Architecture, autonomous, loosely-coupled and coarse-grained services with well-defined interfaces provide business functionality and can be discovered and accessed through a supportive infrastructure.
- This allows internal and external system integration as well as the flexible reuse of application logic through the composition of services.

http://blogs.zdnet.com/service-oriented/?p=490

What is SOA? (4/4)

- The modularization of business functions for greater **flexibility and reusability**.
  - Instead of building monolithic applications for each department, an SOA organizes business software in a granular fashion so that
    - common functions can be used interchangeably by different departments internally and by external business partners as well.
  - The more granular the components (the more pieces), the more they can be reused.
- SOA is a way of thinking about **IT assets as service components**.
  - When functions in a large application are made into stand-alone services that can be accessed separately, they are beneficial to several parties.

http://www.techweb.com/encyclopedia/defineterm.jhtml?term=SOA

Why SOA? (1/4)

- SOA can help businesses respond more quickly and cost-effectively to changing market conditions.
- This style of **architecture** promotes reuse at the macro (service) level rather than micro (classes) level.
- It can also simplify interconnection to - and usage of - existing IT (legacy) assets.

Why SOA? (2/4)

- SOA promotes the goal of separating users (consumers) from the service implementations.
- Services can therefore be run on various distributed platforms and be accessed across networks.
- This can also maximize reuse of services.

Why SOA? (3/4)

- SOA is an architectural and design discipline conceived to achieve the goals of
  - increased interoperability (information exchange, reusability, and composability),
  - increased federation (uniting resources & apps while maintaining their individual autonomy & self-governance), and
  - increased business & technology domain alignment.

Why SOA? (4/4)

- A service is a stand-alone unit of functionality which is available only via a formally defined interface.
  - Services can be some kind of "nano-enterprises" which are easy to produce and improve.
  - Also services can be "mega-corporations" which are constructed as coordinated work of sub-ordinate services .
- Services generally adhere to the following principles of service-orientation:
  - *formal contract, loose coupling, abstraction, reusability, autonomy, statelessness, discoverability, and composability*.

SOA principles

- The following **guiding principles** define the ground rules for development, maintenance, and usage of the SOA:
  - Reuse, granularity, modularity, composability, componentization, and interoperability
  - Compliance to standards (both common and industry-specific)
  - Services identification and categorization, provisioning and delivery, and monitoring and tracking

Architectural principles (1/5)

- The following **specific architectural principles** for design and service definition focus on specific themes that influence the intrinsic behaviour of a system and the style of its design.
  - Service encapsulation
  - Service loose coupling
  - Service contract
  - Service abstraction
  - Service reusability
  - Service composability
  - Service autonomy
  - Service optimization
  - Service discoverability

Architectural principles (2/5)

- **Service encapsulation** - Many web-services are *consolidated* to be used under the SOA Architecture.
  - Often such services have not been planned to be under SOA.
- **Service loose coupling** - Services maintain a relationship that minimizes *dependencies* and only requires that they maintain an *awareness* of each other

Architectural principles (3/5)

- **Service contract** - Services adhere to a ***communications agreement***, as defined collectively by one or more service description documents

- **Service abstraction** - Beyond what is described in the service contract, services ***hide logic*** from the outside world

Architectural principles (4/5)

- **Service reusability** - Logic is divided into services with the intention of promoting reuse
- **Service composability** - Collections of services can be ***coordinated and assembled*** to form composite services
- **Service autonomy** – Services have control over the logic they encapsulate

Architectural principles (5/5)

- **Service optimization** – All else equal, high-quality services are generally considered preferable to low-quality ones

- **Service discoverability** – Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms

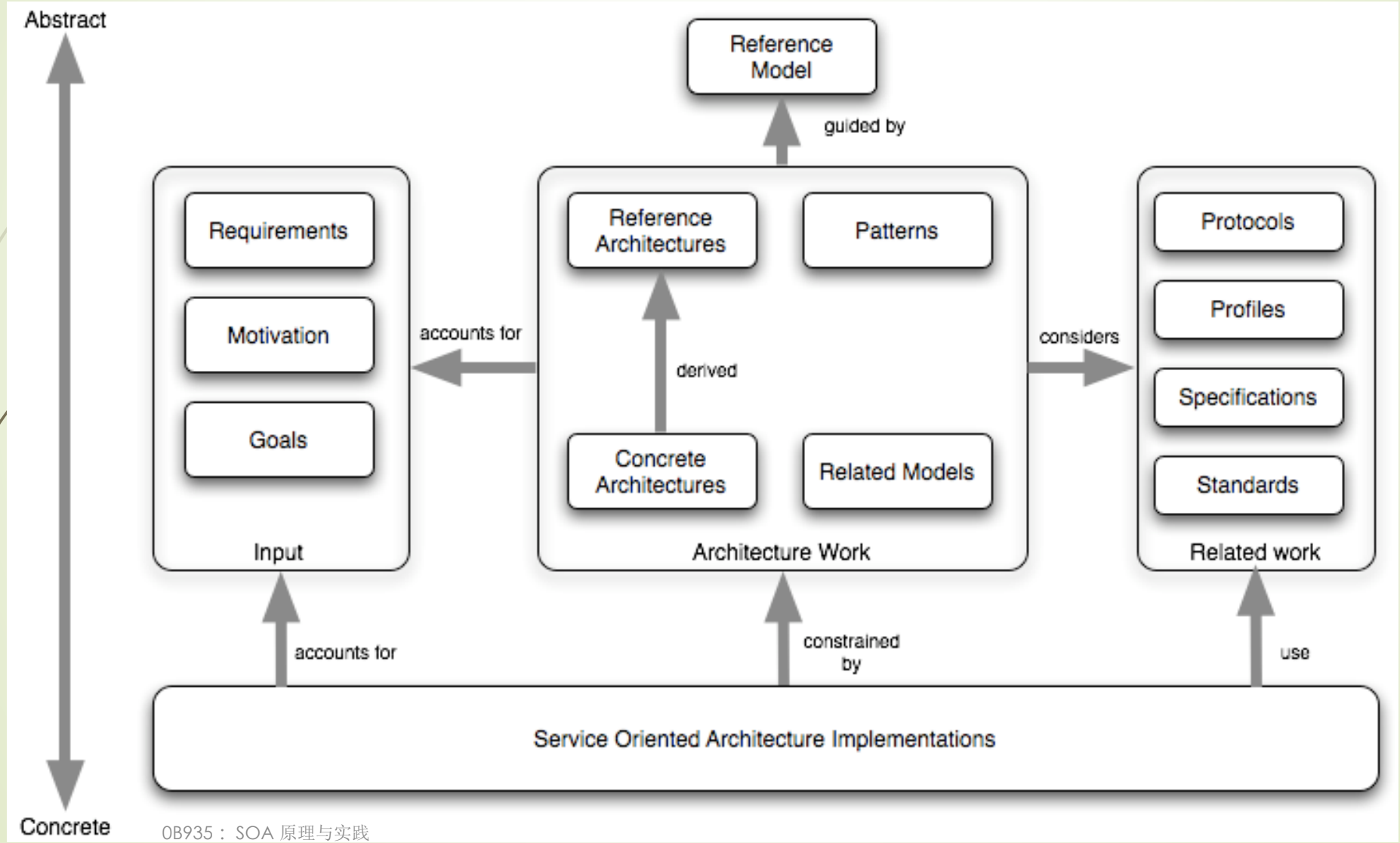
SOA Reference Model - Why?

- SOA itself is used in multiple contexts within the software industry with confusing, differing and even conflicting definitions.
  - If SOA is architecture, as the name implies, how can it be defined and what makes it different from other architectures?
  - How can SOA be described in an architectural manner that is abstract of all implementations?

SOA Reference Model - Definition

- According to the SOA-RM specification,
  - A reference model is an abstract framework for understanding significant relationships among the entities of some environment.
    - It enables the development of specific reference or concrete architectures using consistent standards or specifications supporting that environment.
  - A reference model
    - consists of a minimal set of unifying concepts, axioms and relationships within a particular problem domain, and
    - is independent of specific standards, technologies, implementations, or other concrete details.
  - A reference model for SOA, therefore, is an abstract framework for understanding significant relationships among the entities of SOA.

Scope of the reference model



Reference Model vs. Reference Architecture

- The SOA-RM specification provides a clear distinction between a reference model and a reference architecture, and describes the relationship between them.
  - A reference architecture is an architectural design pattern that indicates how an abstract set of mechanisms and relationships realizes a predetermined set of requirements.
  - One or more reference architectures may be derived from a common reference model, to address different purposes/usages to which the Reference Model may be targeted.

Web services approach to a service-oriented architecture (1/2)

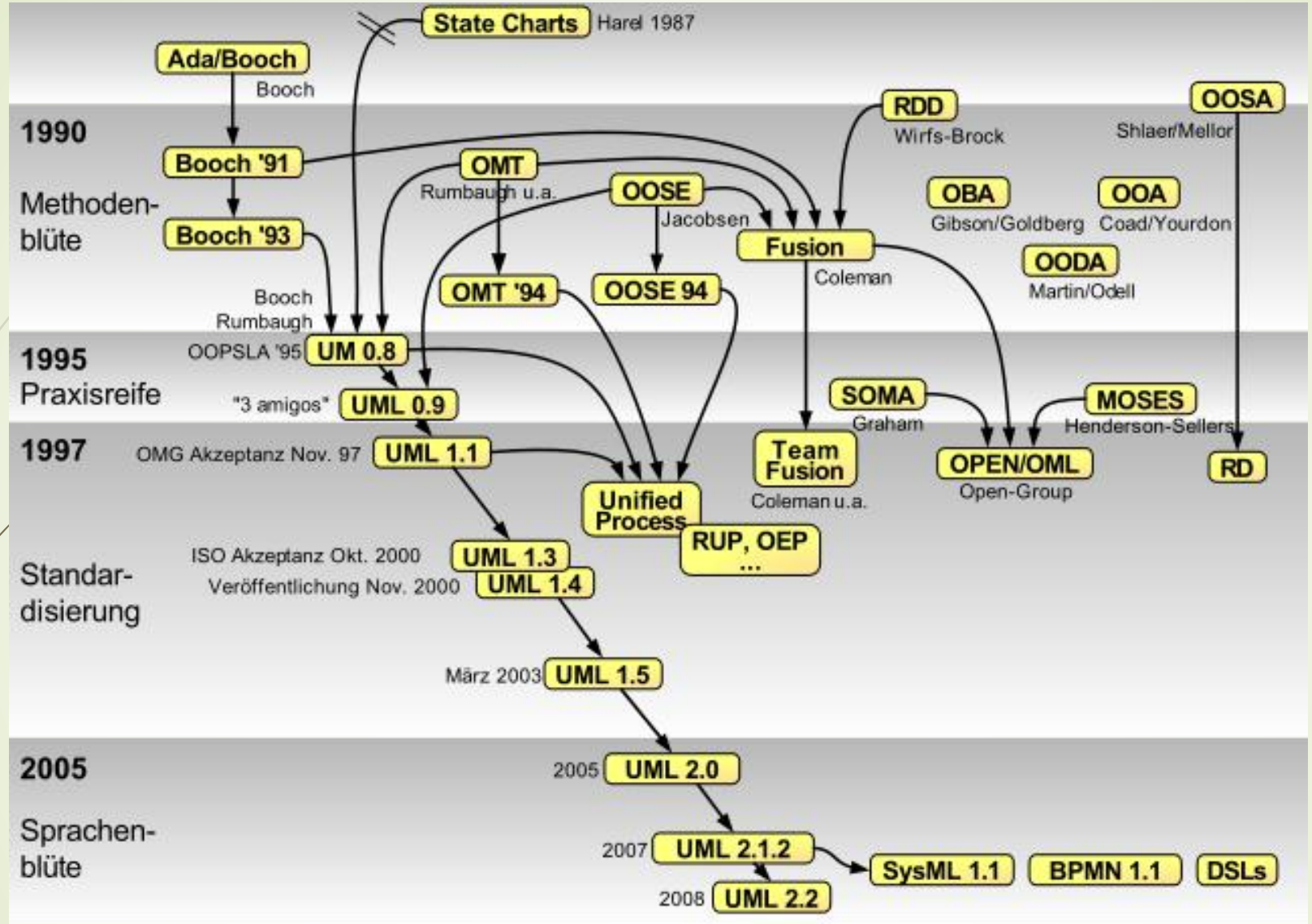
- Web services can be used to implement a service-oriented architecture.
- A major focus of Web services is to make functional building blocks accessible over standard Internet protocols that are independent from platforms and programming languages.
- These services can be new applications or just wrapped around existing legacy systems to make them network-enabled.

Web services approach to a service-oriented architecture (2/2)

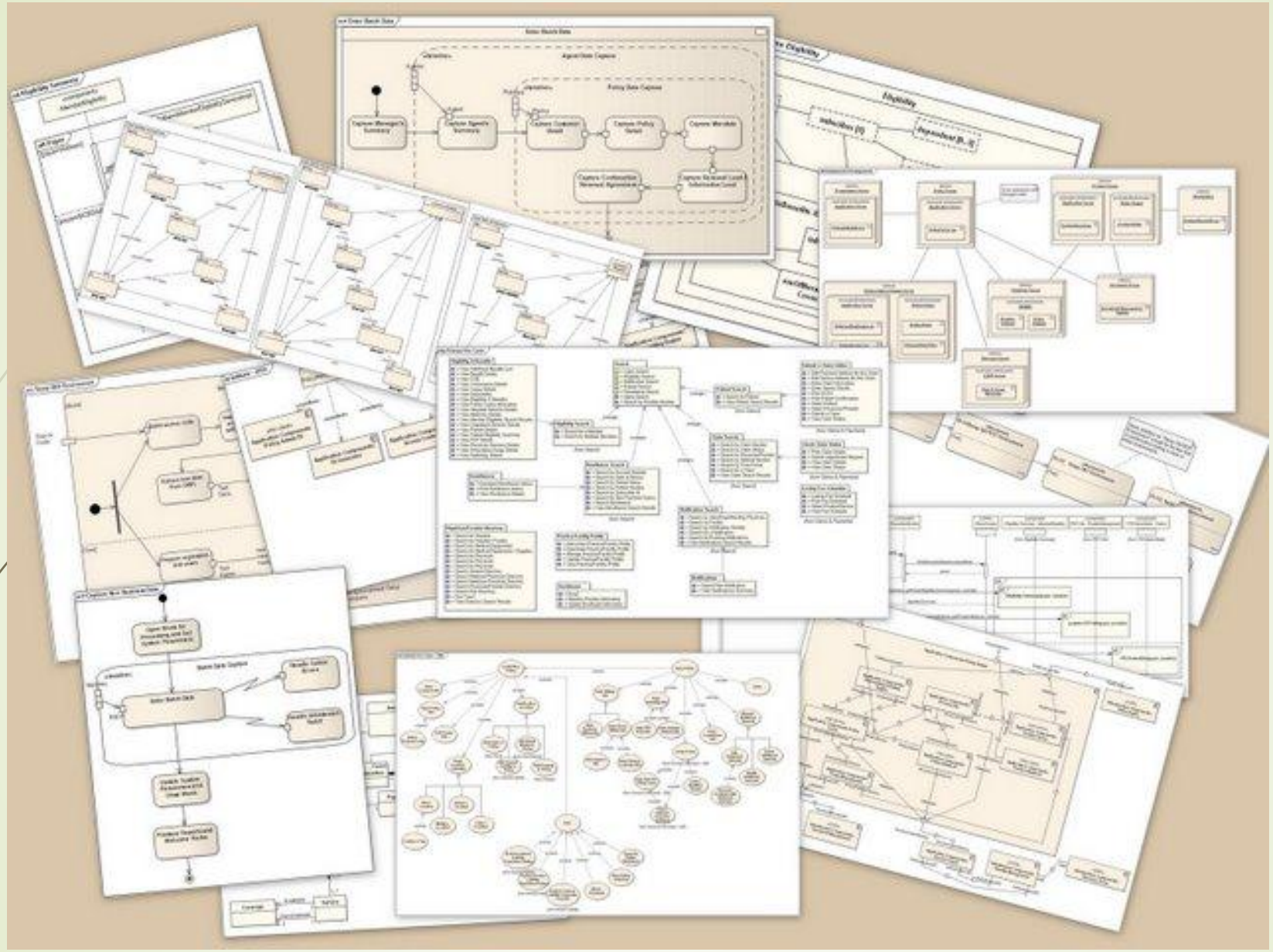
- Each SOA building block can play one or more of three roles:
  - Service provider creates a Web service and possibly publishes its interface and access information to the service registry.
  - Service broker, also known as service registry, is responsible for making the Web service interface and implementation access information available to any potential service requestor.
  - Service requestor or Web service client locates entries in the broker registry using various find operations and then binds to the service provider in order to invoke one of its Web services.

References

- Dion Hinchcliffe Is Web 2.0 The Global SOA?, SOA Web Services Journal, 28 October 2005
- Schroth, Christoph ; Janner, Till; (2007). "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services". . IT Professional 9 (2007), Nr. 3, p. 36-41, IEEE Computer Society Retrieved on 2008-02-23.
- Jason Bloomberg Mashups and SOBAs: Which is the Tail and Which is the Dog?, Zapthink
- What Is Web 2.0. Tim O'Reilly (2005-09-30). Retrieved on 2008-06-10.
- Schroth, Christoph ; Janner, Till; (2007). "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services". . IT Professional 9 (2007), Nr. 3, p. 36-41, IEEE Computer Society Retrieved on 2008-02-23.
- Schroth, Christoph ; Janner, Till; (2007). "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services". . IT Professional 9 (2007), Nr. 3, p. 36-41, IEEE Computer Society Retrieved on 2008-02-23.
- Ruggaber, Rainer; (2007). "Internet of Services—A SAP Research Vision". . IEEE Computer Society Retrieved on 2008-02-23.
- Paul Krill Make way for SOA 2.0, InfoWorld , May 17, 2006
- Yefim Natis & Roy Schulte Advanced SOA for Advanced Enterprise Projects, Gartner, July 13, 2006
- Joe McKendrick Anti-SOA 2.0 petition nears 400, ZDNet.com, June 29, 2006



History of object-oriented methods and notation.



A collage of UML diagrams.