

The logo for the Database Technology Conference China (DTCC) 2013. It features the letters 'DTCC' in a bold, orange, sans-serif font. The 'D' and 'T' are connected, and the 'C's are also connected. Below the letters is a thin, curved orange line.

## 2013中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2013

大数据 数据库架构与优化 数据治理与分析

SequeMedia  
盛拓传媒

IT168.com

ITPUB

ChinaUnix

A graphic on the right side of the slide featuring a stack of orange and yellow 3D rectangular blocks. One block is tilted and contains text listing various database technologies. The background is a light yellow with faint, overlapping square outlines.

Database  
BDaaS  
flowingdata  
DB2  
NoSQL MySQL  
Oracle Big Data

# 混合型企业数据访问体系

王翔

软件工程师

# How Do Technology Trends Impact the Human, Business and IT Experiences?

## Top 10 Strategic Technology Trends for 2013

1. Mobile Devices Battles
2. Mobile Applications & HTML5
3. Personal Cloud
4. Internet of Things
5. Hybrid IT & Cloud Computing
6. Strategic Big Data
7. Actionable Analytics
8. Mainstream In-Memory Computing
9. Integrated Ecosystems
10. Enterprise App Stores

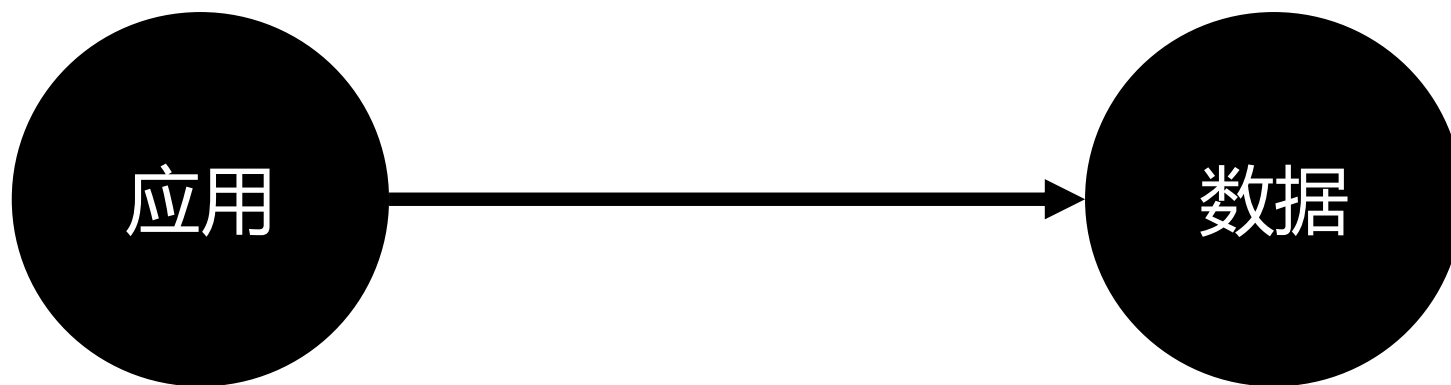
## Trend Impact Assessment



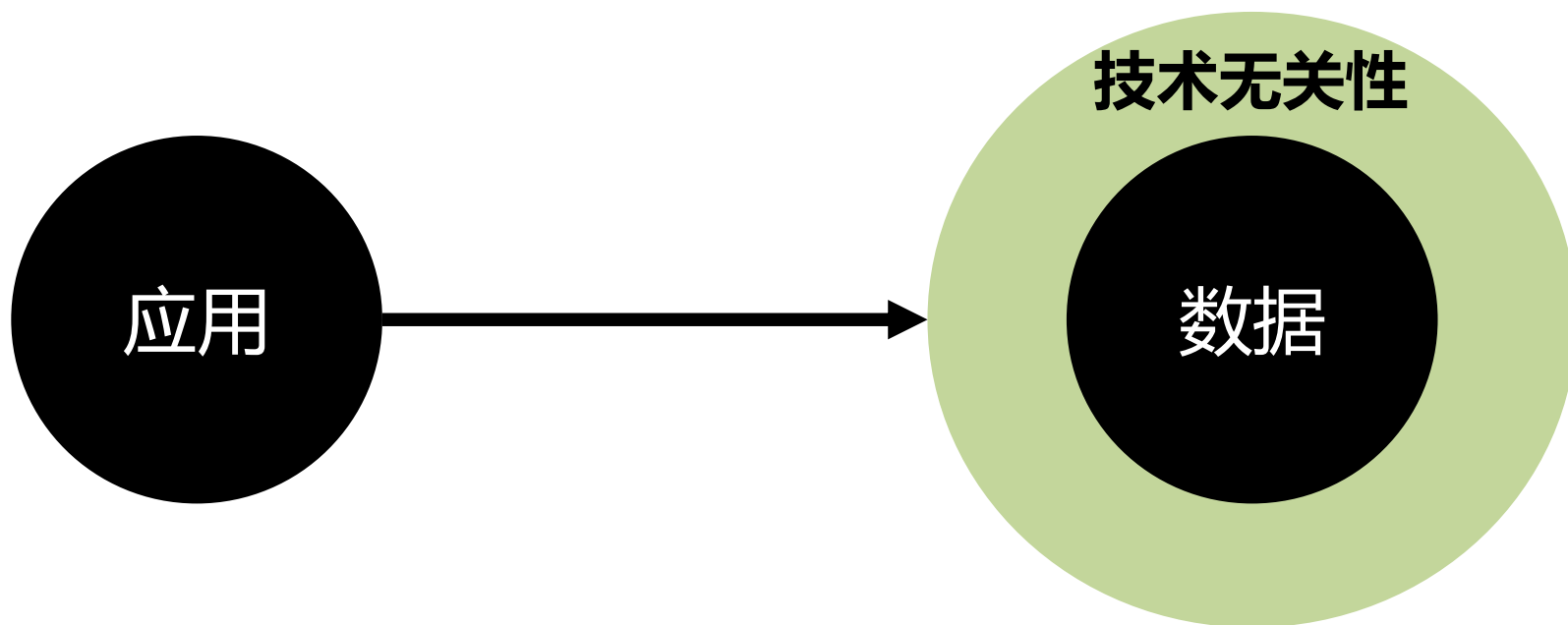
Gartner



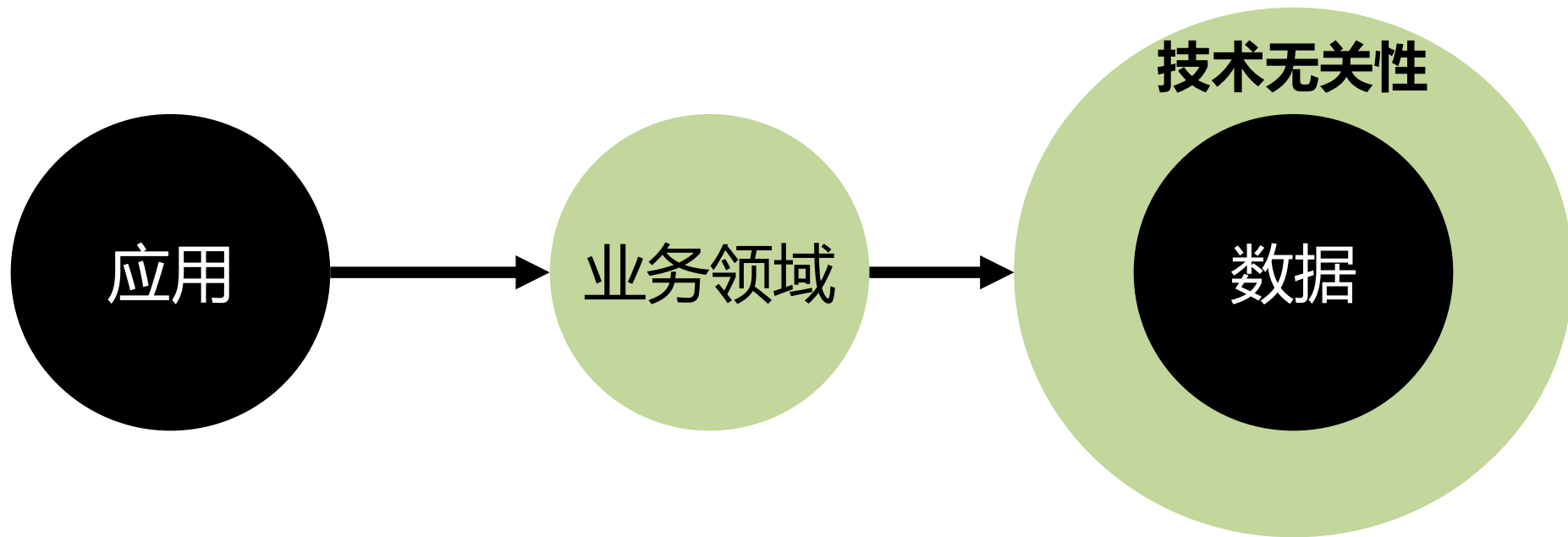
# 数据访问



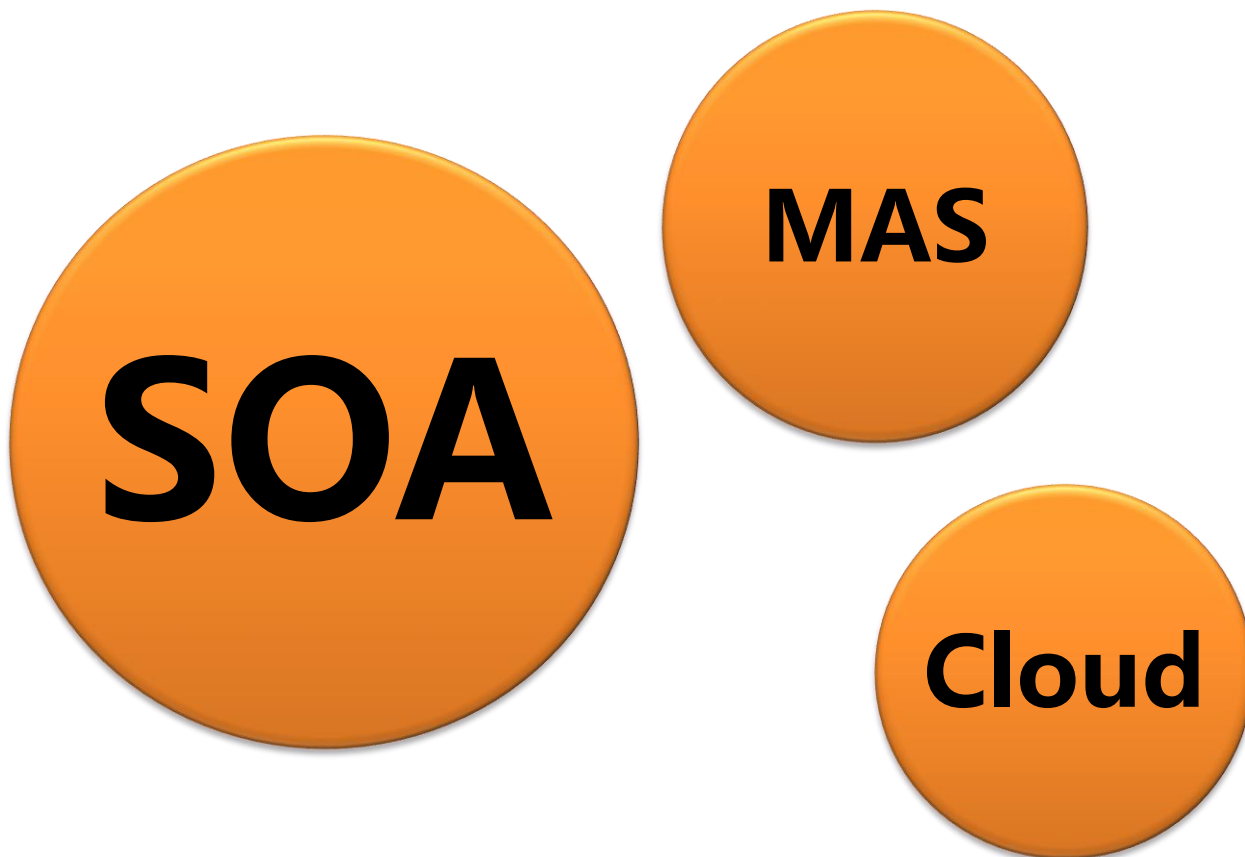
# 数据访问



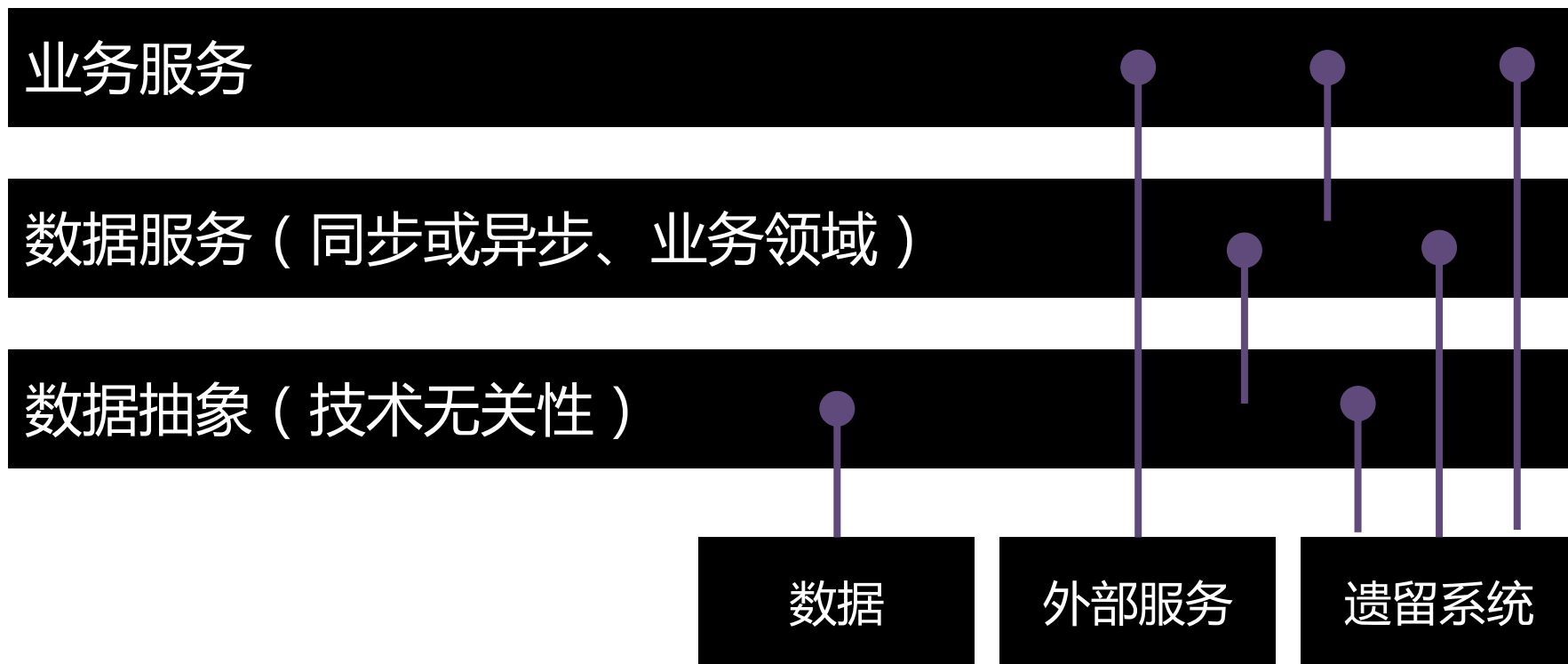
# 数据访问



# 设计理念在改变



# 设计理念在改变-SOA



# 设计理念在改变-Cloud

Cloud

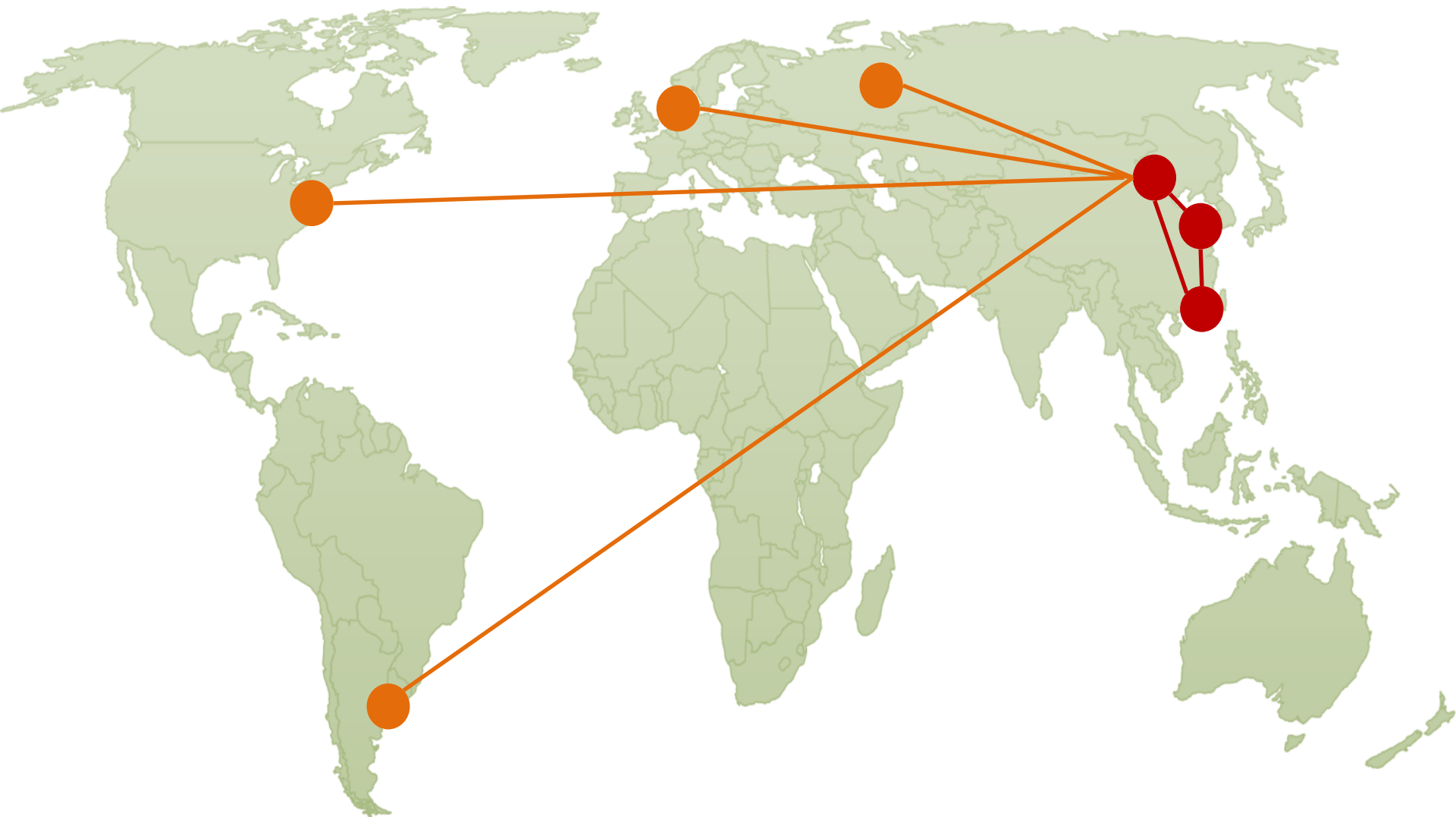


轻量接入端

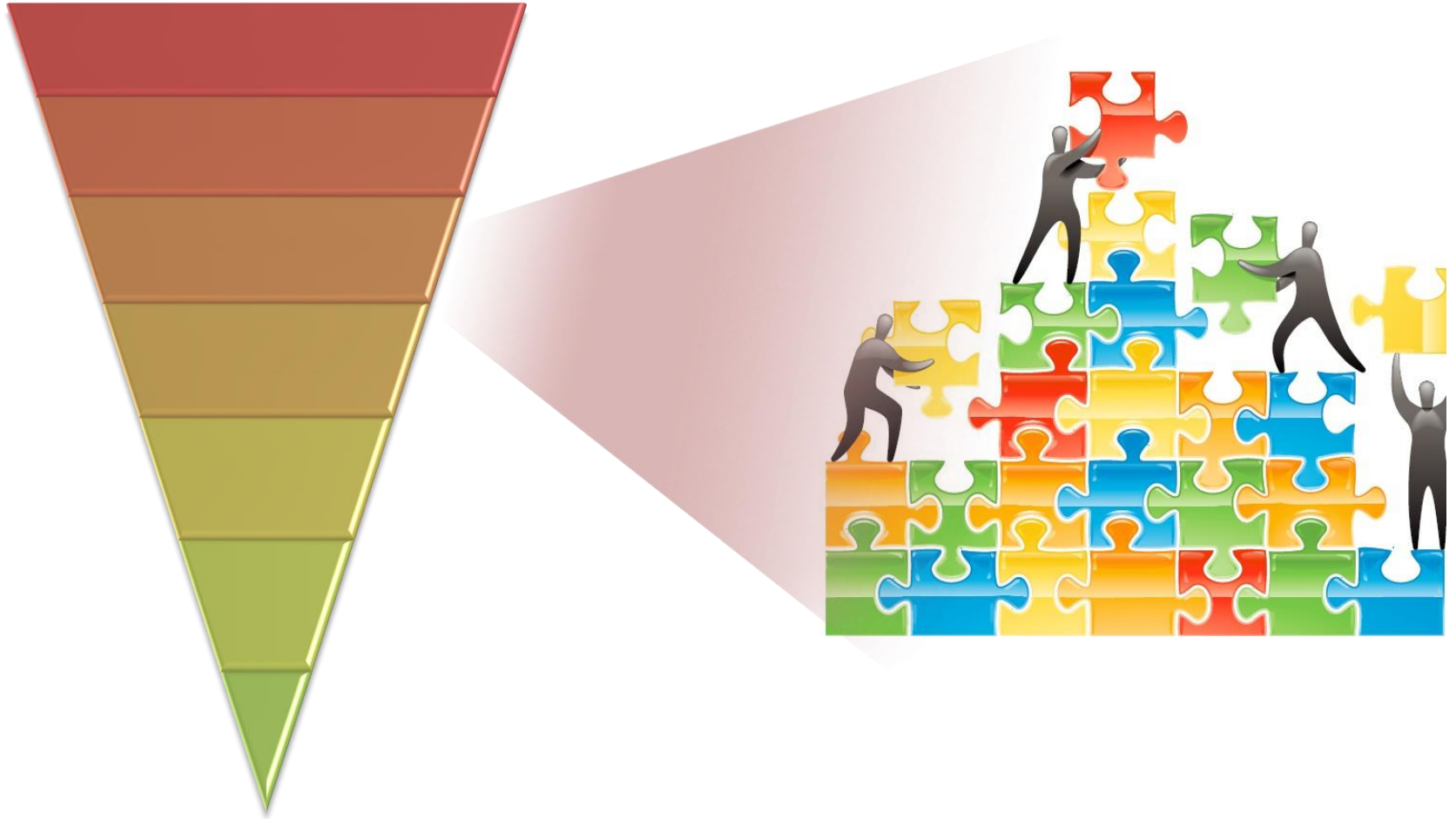
The diagram illustrates a connection between a 'Cloud' component and a '轻量接入端' (Lightweight Interface) component. A vertical line connects the bottom of the 'Cloud' bar to a circular node, which is then connected to the '轻量接入端' box.



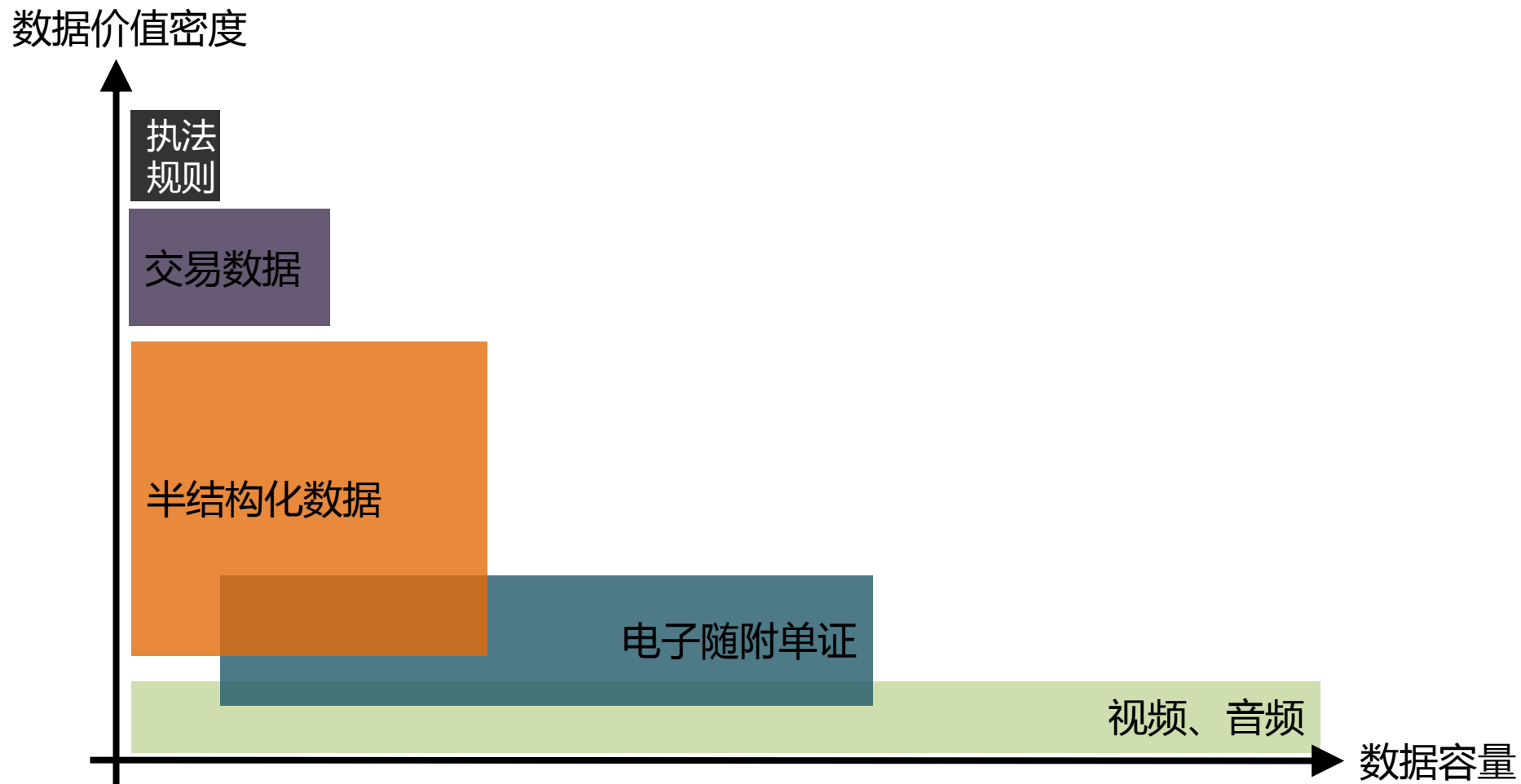
# 问题-异地容灾和高可用数据中心



# 问题-多层次多信任域的跨业务访问



# 问题-不同的数据价值密度和容量



# 问题还有... ..

- 多种开发语言：Cobol、Java、.NET、Dynamics Lang
- 多种数据技术：
  - 关系数据库
  - 并行数据库
  - 列式数据库
  - 分布式文件数据库
  - 嵌入式数据库
  - 半结构化数据和电子商业文档
  - 音频、视频等流媒体数据

# 数据切片概念的泛化

**MapReduce**

**地理和业务水平分区**

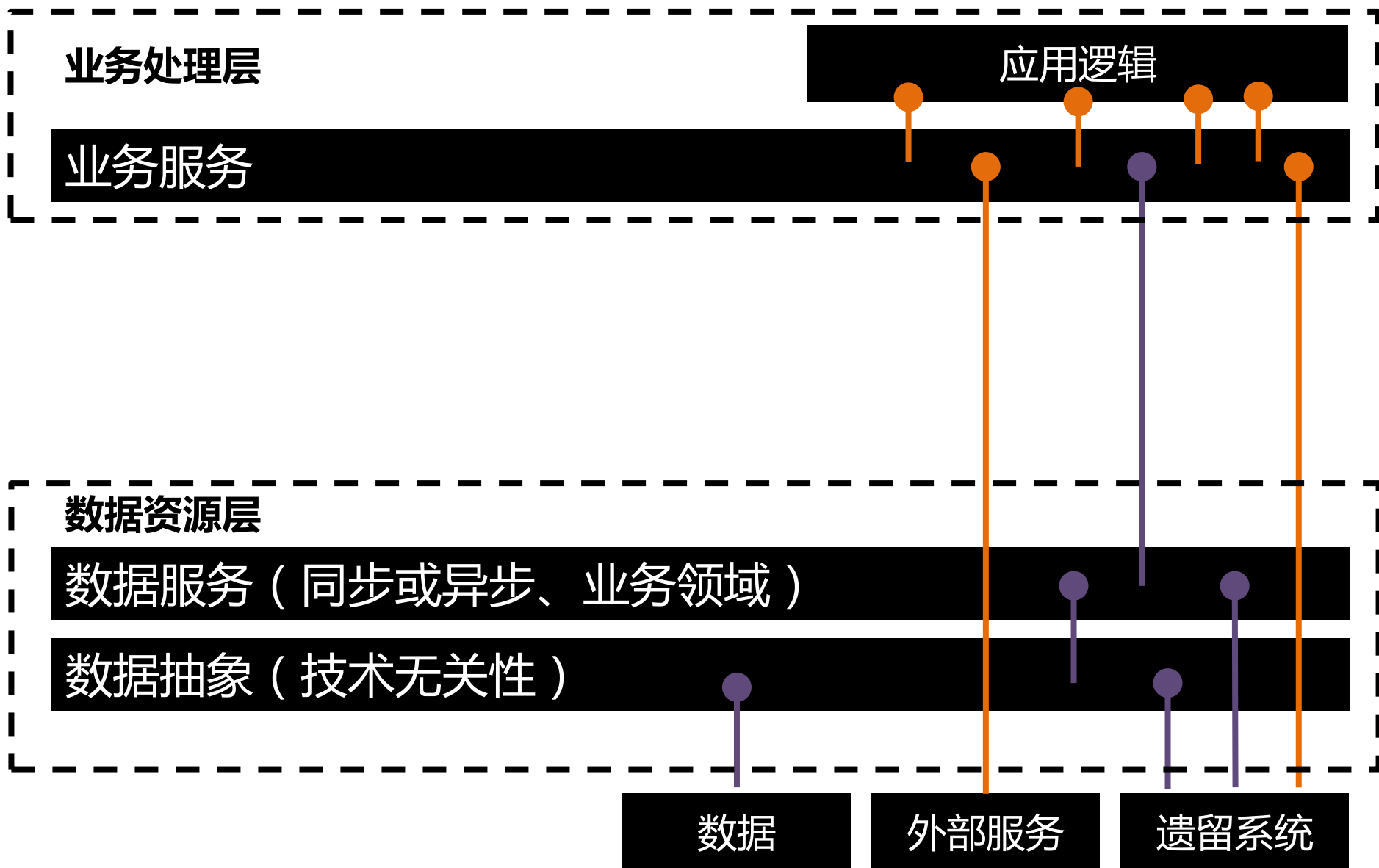
**数据生命周期**



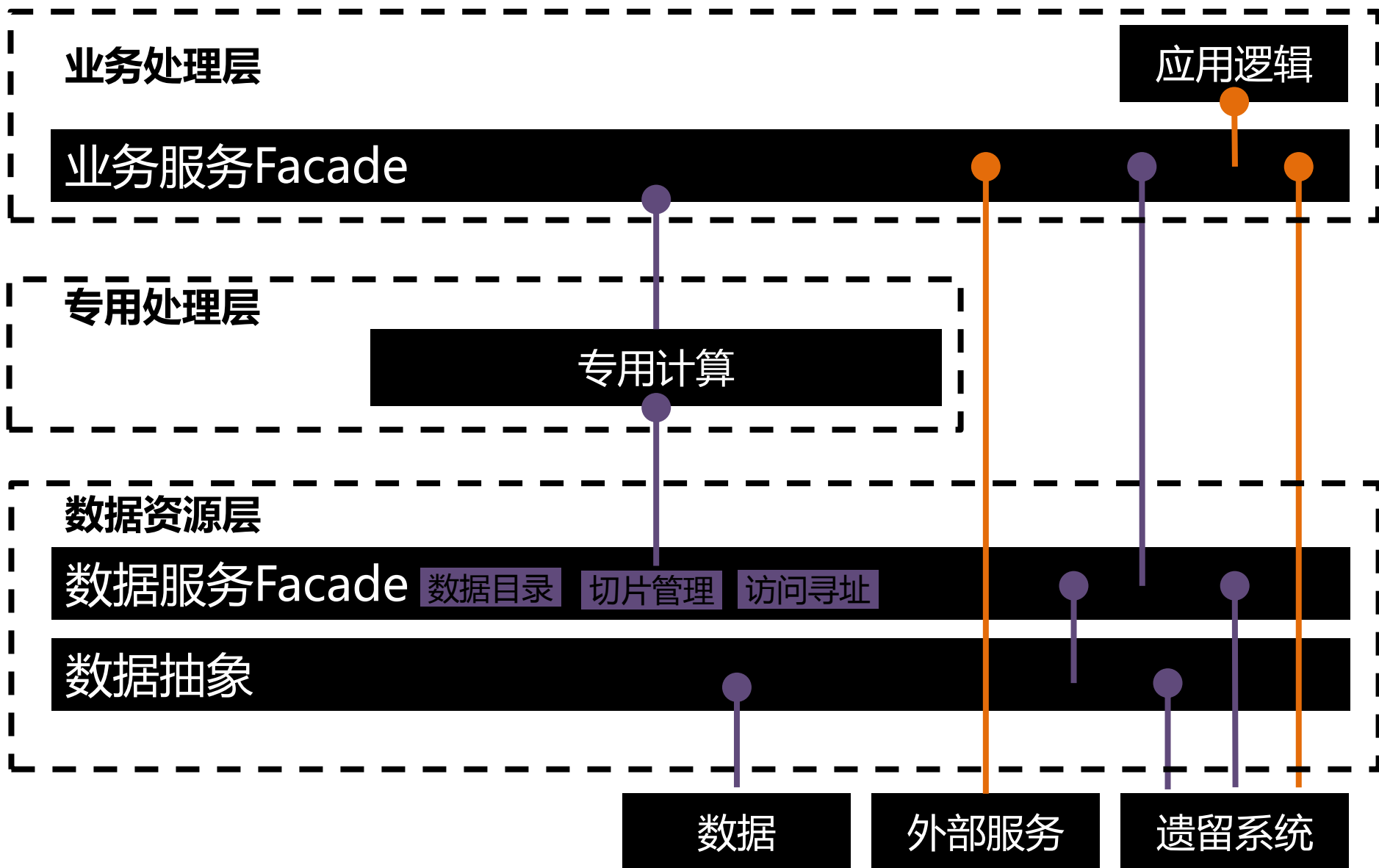
**垂直分区和列分解**

**安全域划分**

# 解决方案 V1



# 解决方案 V2

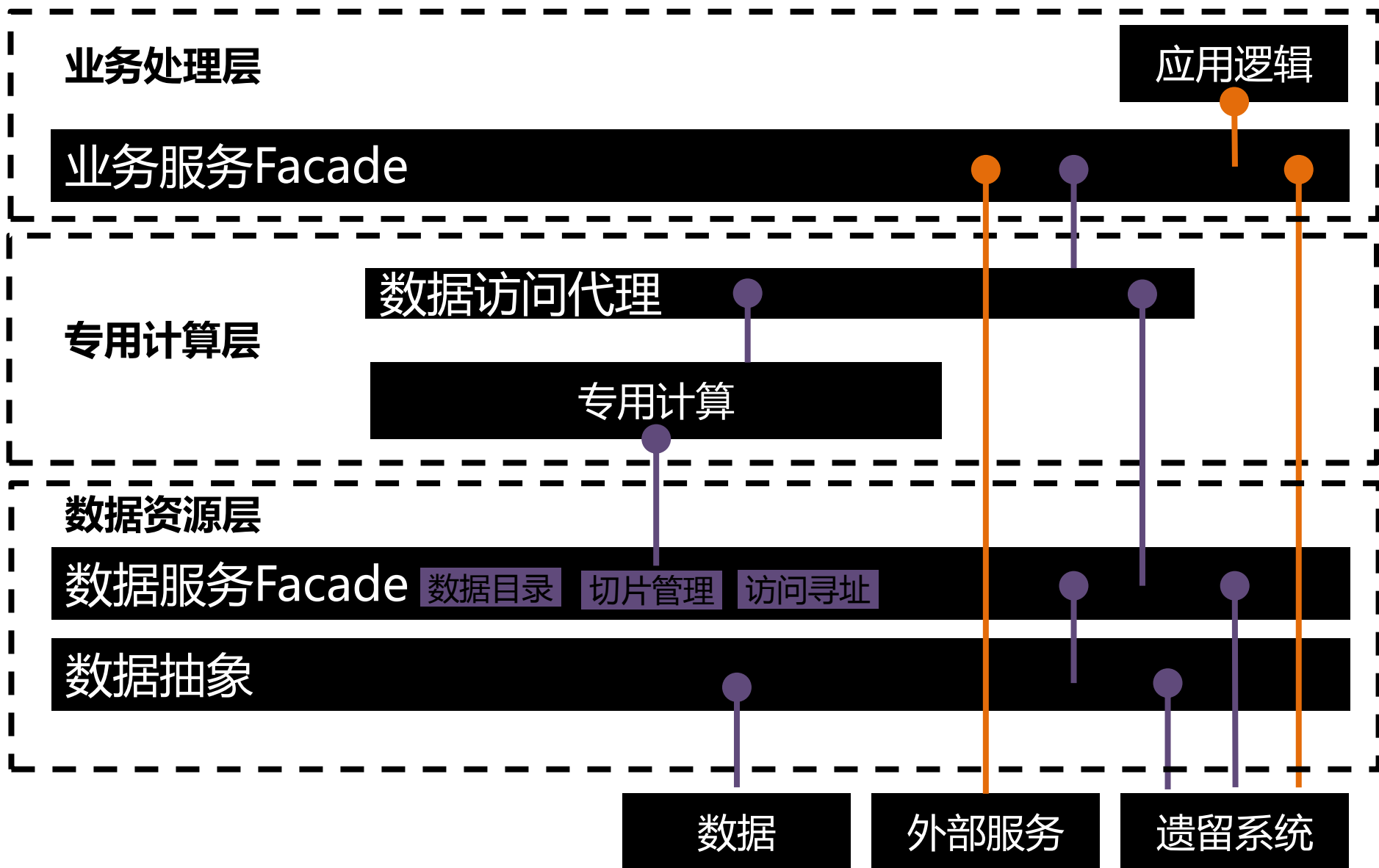








# 解决方案 V3



# 非功能性需求如何解决？



- 如何应对变更
- 如何融合各种非功能性处理要求
- 如何驱动流程

# 非功能性需求如何解决？



```
void 放行处理 ( ) {  
    身份认证 ;  
    检查授权 ;  
    执行放行逻辑 ;  
    错误检查 ;  
    发送处理回执 ;  
}
```



```
void 放行处理 ( ) {  
    身份认证 ;  
    检查授权 ;  
    访问控制 ;  
    执行放行逻辑 ;  
    错误检查 ;  
    推送下一步工作流 ;  
    发送处理回执 ;  
}
```

# 非功能性需求如何解决？

- AOP或SOA?

```
void 放行处理 ( ) {  
    身份认证 ;  
    检查授权 ;  
    访问控制 ;  
    执行放行逻辑 ;  
    错误检查 ;  
    推送下一步工作流 ;  
    发送处理回执 ;  
}
```

```
def aspectLogic(func)  
    def wrap()  
        #预处理逻辑  
        func();  
        #后续处理逻辑  
    return wrap
```

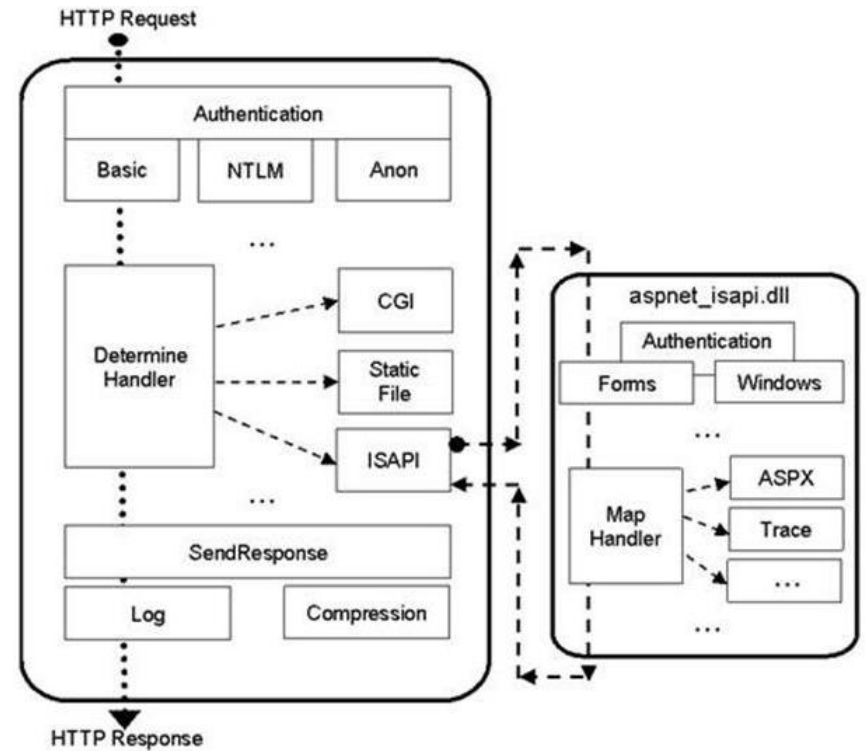
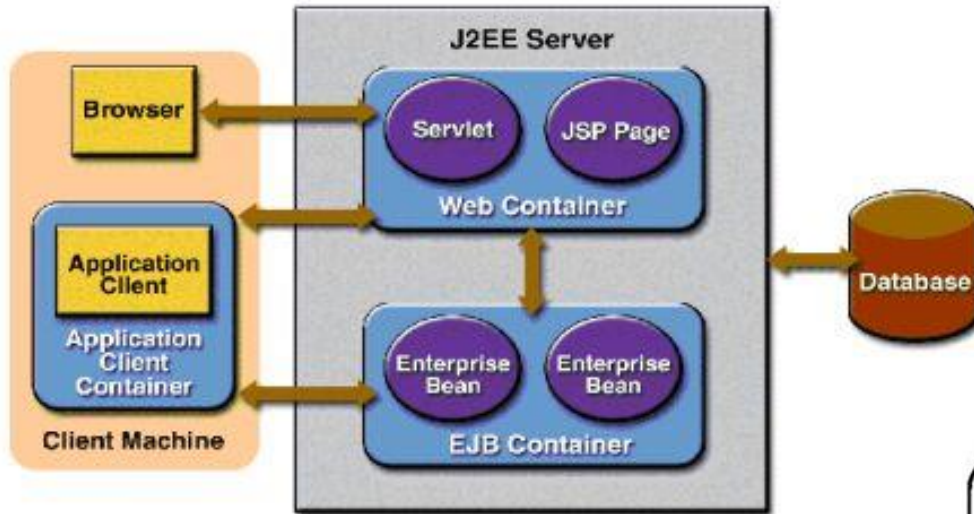
```
@aspectLogic  
def 放行处理():  
    处理放行
```

放行处理()

容器？



# 容器？





# 基于容器的开发

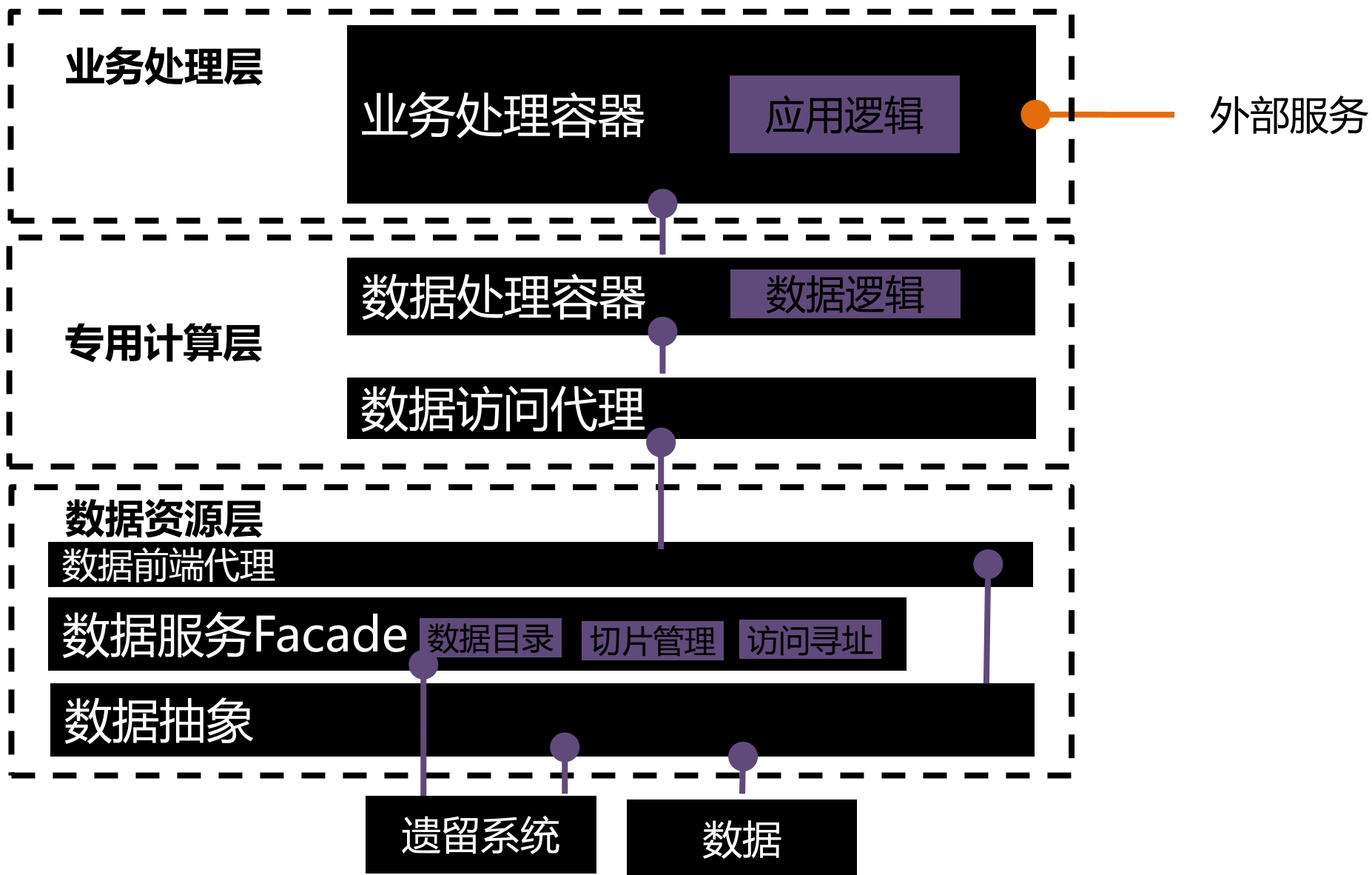


```
void 放行处理 ( ) {  
    身份认证 ;  
    检查授权 ;  
    访问控制 ;  
    执行放行逻辑 ;  
    错误检查 ;  
    推送下一步工作流 ;  
    发送处理回执 ;  
}
```

```
@Secure  
@Notify  
void 放行处理 ( ) {  
    执行放行逻辑 ;  
}
```



# 解决方案 V4





# 充分利用SQL Server 2012

业务处理层

WebLogic Executor

应用逻辑

IIS Executor

应用逻辑

Jboss Executor

应用逻辑

外部服务

业务处理容器服务代理

专用计算层

数据处理容器

数据逻辑

SSDS

SSAS

SSRS

数据资源层



Microsoft®  
SQL Server® 2012



遗留系统

数据

# 为服务器赋予Role



# 基于容器的MAS开发



**[Secure]**

```
public class VehicleCheck<TEntity, TScalar>
{
```

**[AgentList(进港、在港、出港、码头、货场、陆路卡口)]**

```
public void Sniffer(String shipNo, String voyageNo)
{
```

```
    // 获得港口规定路径
```

```
    scalar= db.GetScalar(new KeyValuePair<,>{
        Ship = new Ship(shipNo, voyageNo),
        Decisions = extensiveDecssions
```

```
    });
```

```
}
```

**[AgentList(进港、在港、出港)]**

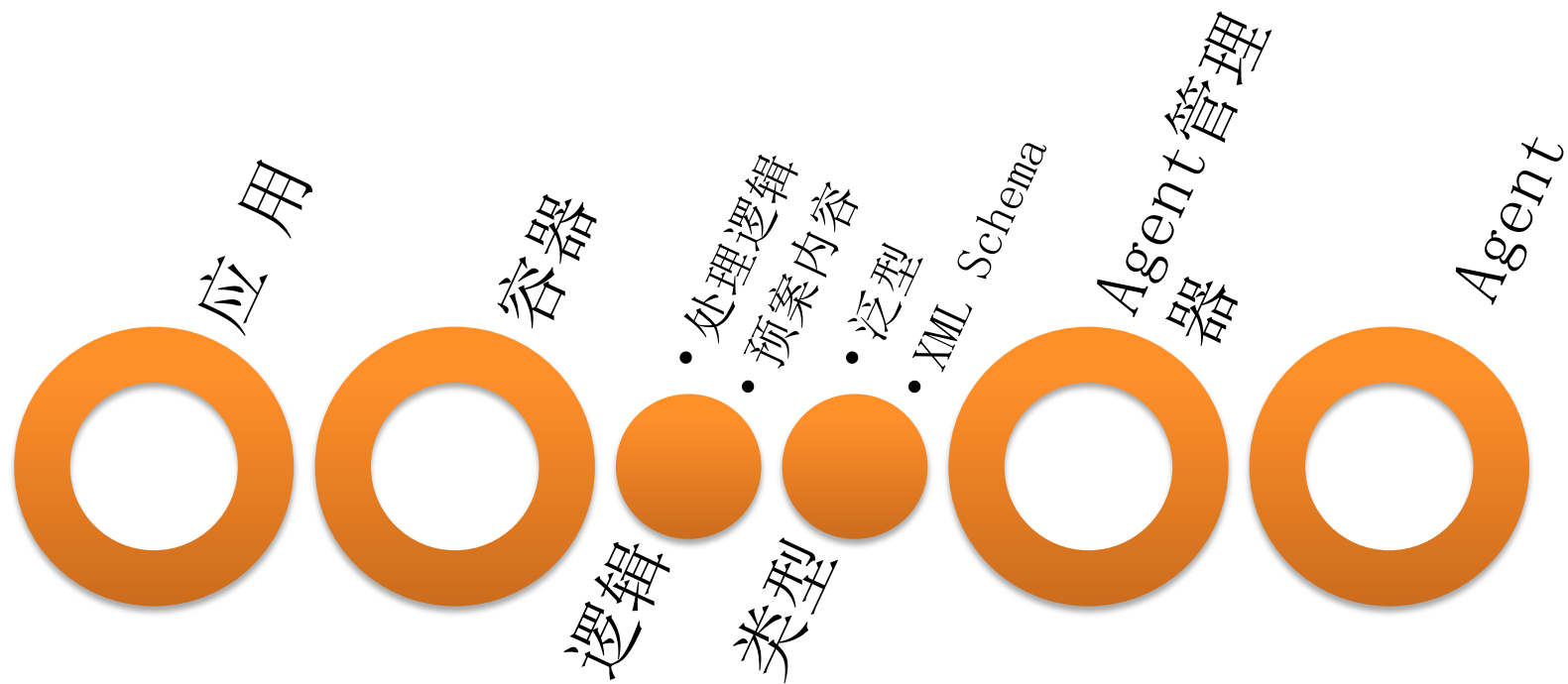
```
public void Alert(TEntity entity)
{
```

```
    // 通过3个Agent逻辑判断船舶在途异常
```

```
}
```

```
}
```

# 执行过程



# 建设步骤



## 阶段0

- 组建平台团队
- 确定目标和原则

## 阶段一

- 开发零散构件块
- 设计容器和代理框架

## 阶段二

- 零散构件块集成
- 搭建平台龙骨
- 提供容器、代理支持

## 阶段三

- 完善容器、代理框架
- 持续集成应用系统

## 阶段四

- 完成平台迁移
- 持续集成应用系统

# SOA

# MAS

# Cloud

## 欢迎莅临

## 2013中国数据库技术大会

Database  
**BDaaS**  
flowingdata  
DB2  
NoSQL MySQL  
Oracle Big Data

