

# 基于云原生Serverless和消息服务同步全球分布业务数据实践

马腾/资深云解决方案架构师  
微软（中国）有限公司



# 精彩继续！ 更多一线大厂前沿技术案例

上海站



时间：2023年4月21-22日  
地点：上海·明捷万丽酒店

扫码查看大会详情>>



广州站



时间：2023年5月26-27日  
地点：广州·粤海喜来登酒店

扫码查看大会详情>>





# 议程

- 解决方案背景
- 多种方案架构对比
- 解决方案技术挑战
- 技术架构设计
- 经验总结

# 解决方案背景

- 客户是中国Top3的手机厂商和互联网公司，海外业务全球分布（美洲、亚洲、欧洲、非洲等）
- 依照各区域的数据合规性要求，每个业务系统数据需要在各个洲独立存储，主要是结构化数据。
- 业务场景需要应用及数据集成，将海外业务数据与中国数据中心数据同步
- 不同的业务场景需要实现不同的实时性要求，主要是准实时（1分钟内）和每天或每月定期同步。

# 国内售后CRM – 海外售后CRM双向定时同步

业务场景：订单，定价，促销策略等业务数据在国内CRM与海外CRM之间异步定时复制



# 技术方案对比

- 云原生Serverless+消息服务
- 云原生数据ETL工具
- 云端VM/容器服务
- 数据库自带的数据同步功能

# 技术解决方案架构对比

	数据实时性	成本	可靠性	安全性	性能/弹性	运维（精力）
云原生Serverless (FaaS) +消息服务	实时或准实时	低	高	高	强	低
云原生数据ETL工具	以定时大批量数据同步为主， 可以实现准实时	中	高	高	强	低
云端VM或云原生容器服务	通过自行实现数据同步逻辑 可实现实时或准实时	高	中 (需使用其他组件共同保障)	中 (需使用其他组件共同保障)	中	高
数据库自带的数据同步功能	实时或准实时， 使用场景受制于数据库部署环境	低	低	高	低	中




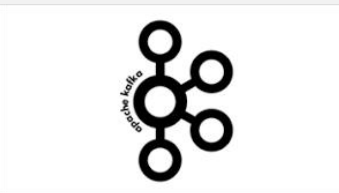


# 技术挑战

- Serverless计算资源如何保证可靠扩展性和可靠性？
- Serverless冷启动问题：如何实现低延迟的计算实例启动？
- Serverless函数如何监控，诊断，链路追踪？
- 消息服务如何保障“高价值消息”不丢失不重复，先进先出？
- 消息队列异常消息如何处理？
- 消息队列如何实现并发处理机制？



# Azure上多种消息服务如何选择？

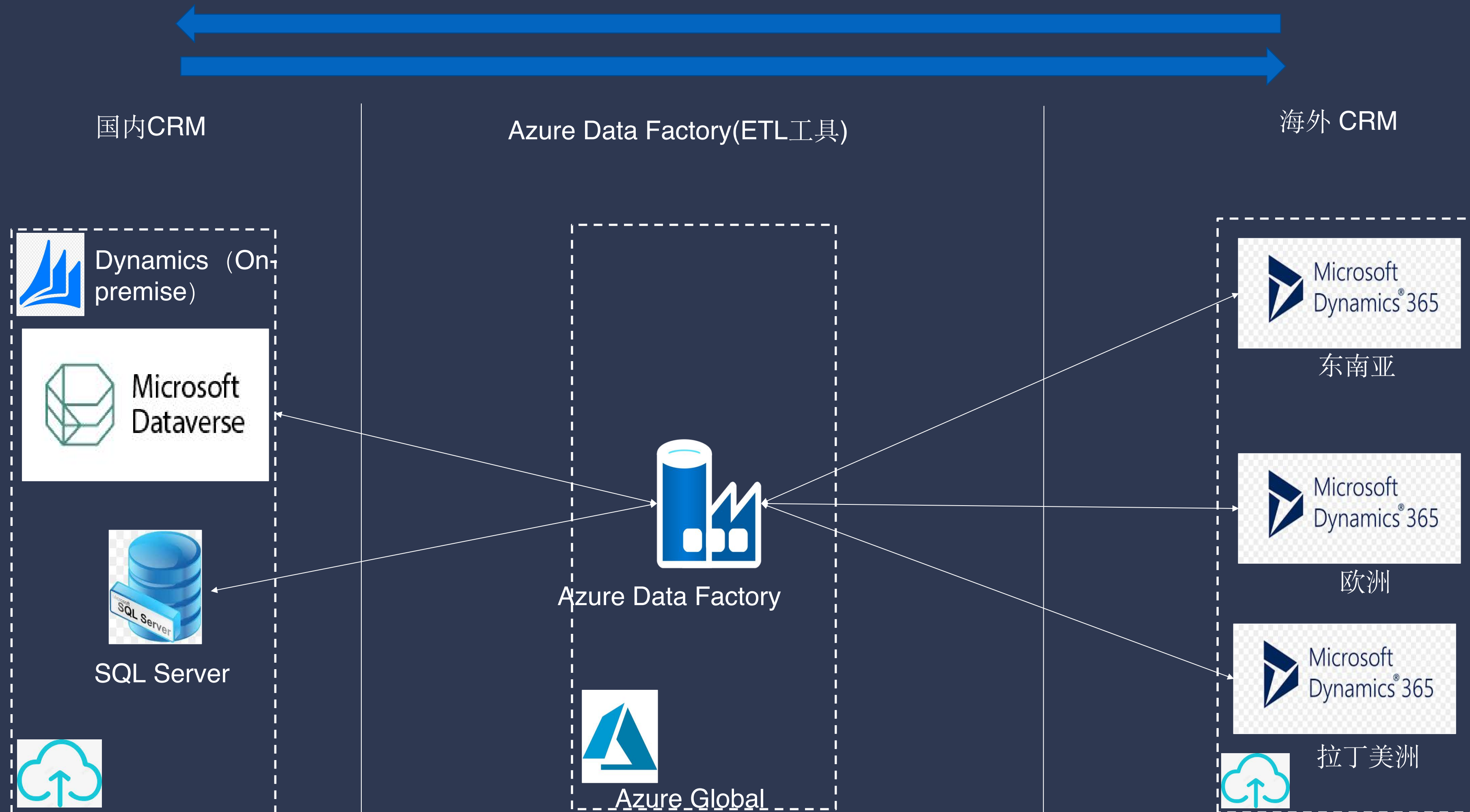
消息服务	特性	使用场景
 Azure Service Bus	支持多种消息传递模式，包括队列、主题和订阅。可靠性、安全性和可伸缩性高。提供定时消息传递、消息回溯、基于身份验证和授权的访问控制等高级功能。	适用于“ <b>高价值业务消息</b> ”传递场景。例如异步通信、任务分发、应用程序集成等。
 Azure Event Hub	大规模、高吞吐量	用于传输大量的事件数据，例如传感器数据、日志数据、跟踪数据等。
 Azure IoT Hub	专门为物联网应用程序设计的消息传递服务。它支持多种协议，例如 AMQP、MQTT 和 HTTP，以及各种设备和平台	连接和管理大规模 IoT 设备，并可将设备数据发送到云中进行分析和处理。
 HD Insight-Kafka	分布式流数据平台，提供了高吞吐量、低延迟、高可用性、可扩展性和容错性等特性	大规模流数据处理的场景，例如实时数据分析、数据流管道等

# 方案架构设计

- 数据同步基于消息队列加Serverless架构实现准实时数据同步方案
- 通过消息队列实现异步解耦
- 基于Serverless FaaS架构，实现高可靠、高可扩展、低成本技术方案

# 国内售后CRM –海外售后CRM双向定时同步

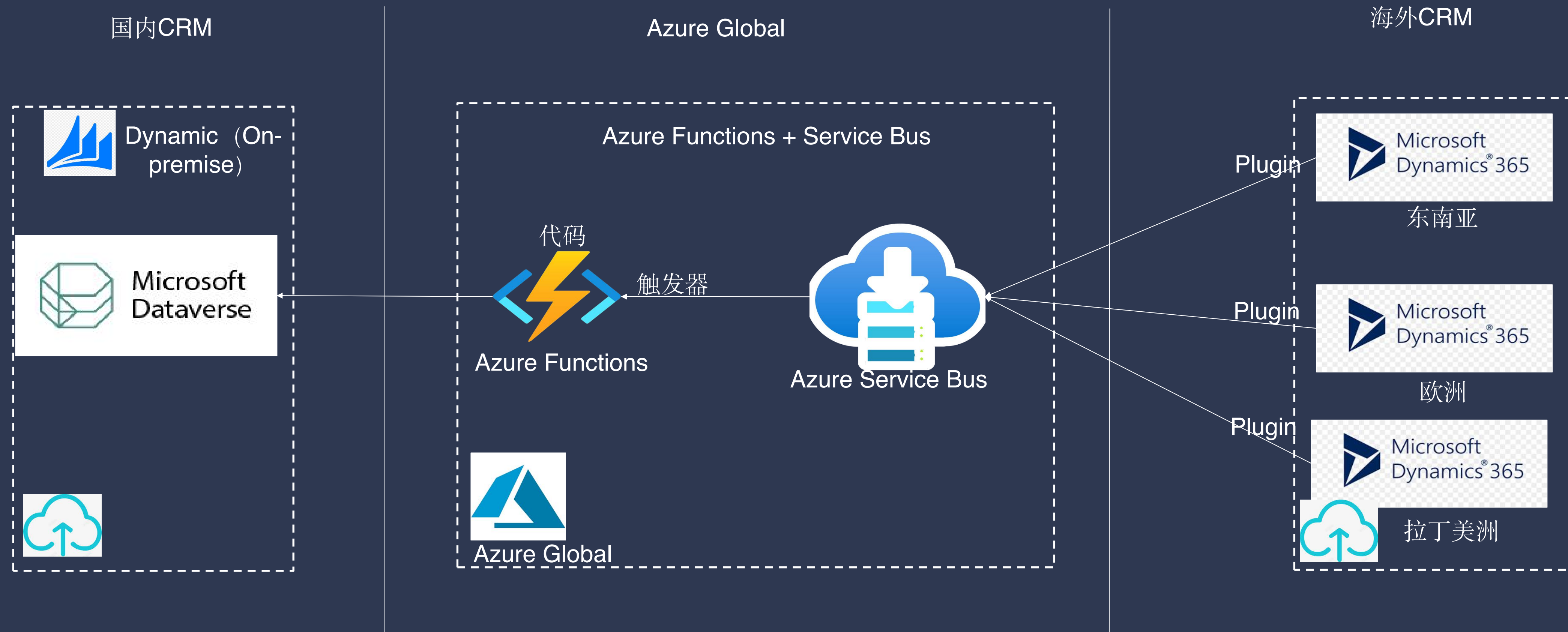
业务场景：公用业务数据，订单，价格，促销策略等业务数据“定时批量”在国内CRM与海外CRM之间异步定时复制





# 国内服务CRM –海外服务CRM双向数据实时同步

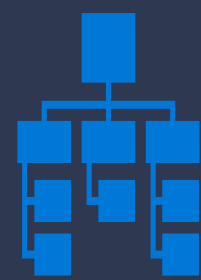
业务场景：订单数据由海外CRM同步到国内CRM（实时或准实时同步，要求1分钟内）



# Function as a Service

使用无服务器代码处理事件

- 让编写云应用变得简单
- 根据客户需求扩展功能
- 使用 C#、Node.js、F#、Python、Java 等语言开发函数
- 跨服务轻松安排事件驱动型任务
- 将函数公开为 HTTP API 端点



无需基础架构管理



根据工作负载自动缩放



不浪费资源，只需按实际使用量付费

# Functions—基于事件驱动

专注于代码，而不是管道

事件



对计时器、HTTP 或 Azure 服务的事件做出反应，并不断推出更多事件源

编码



在 C#、F#、Node.js、Java 等环境中创作函数

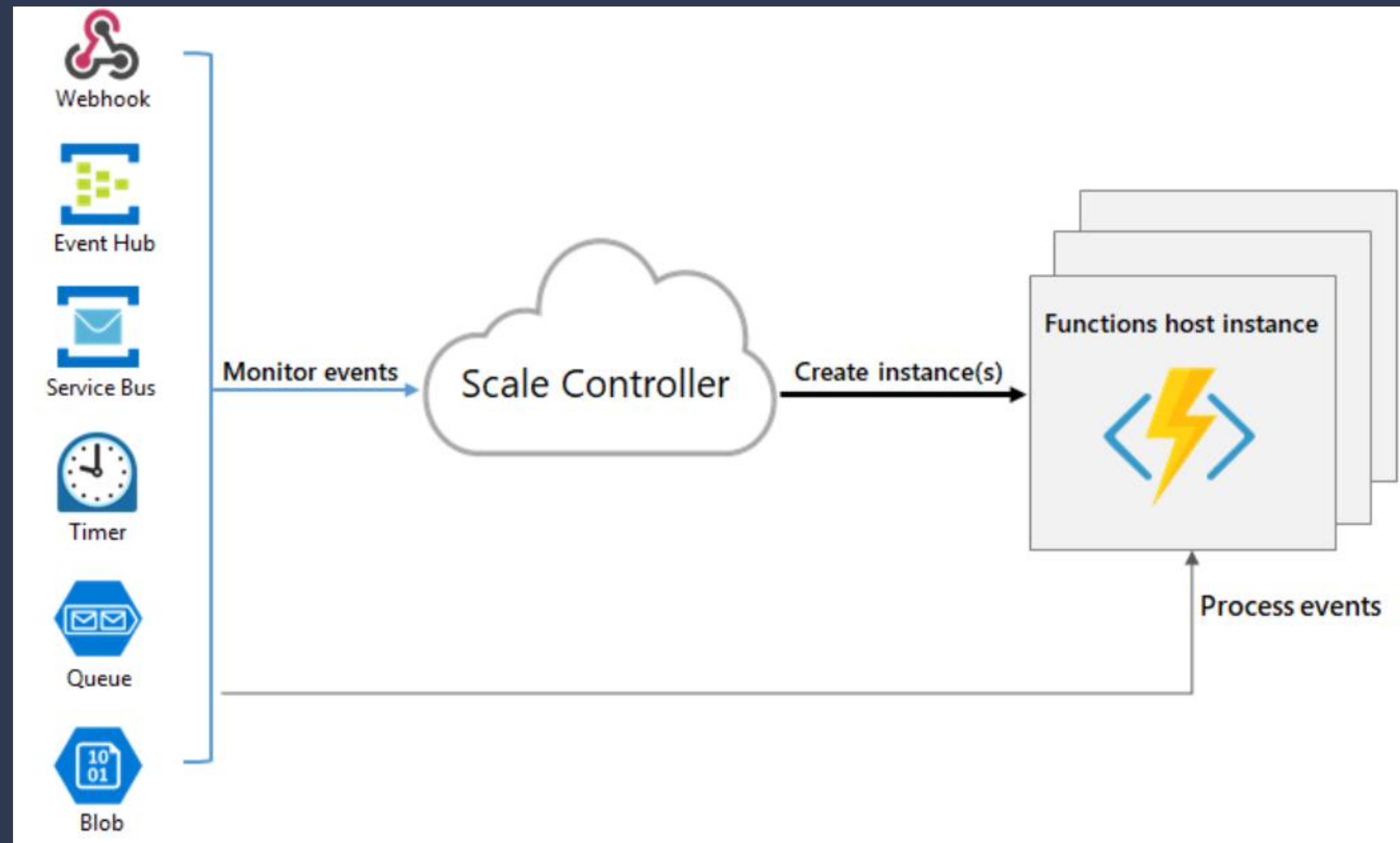
输出



将结果发送到下游服务，下游服务也不断增加

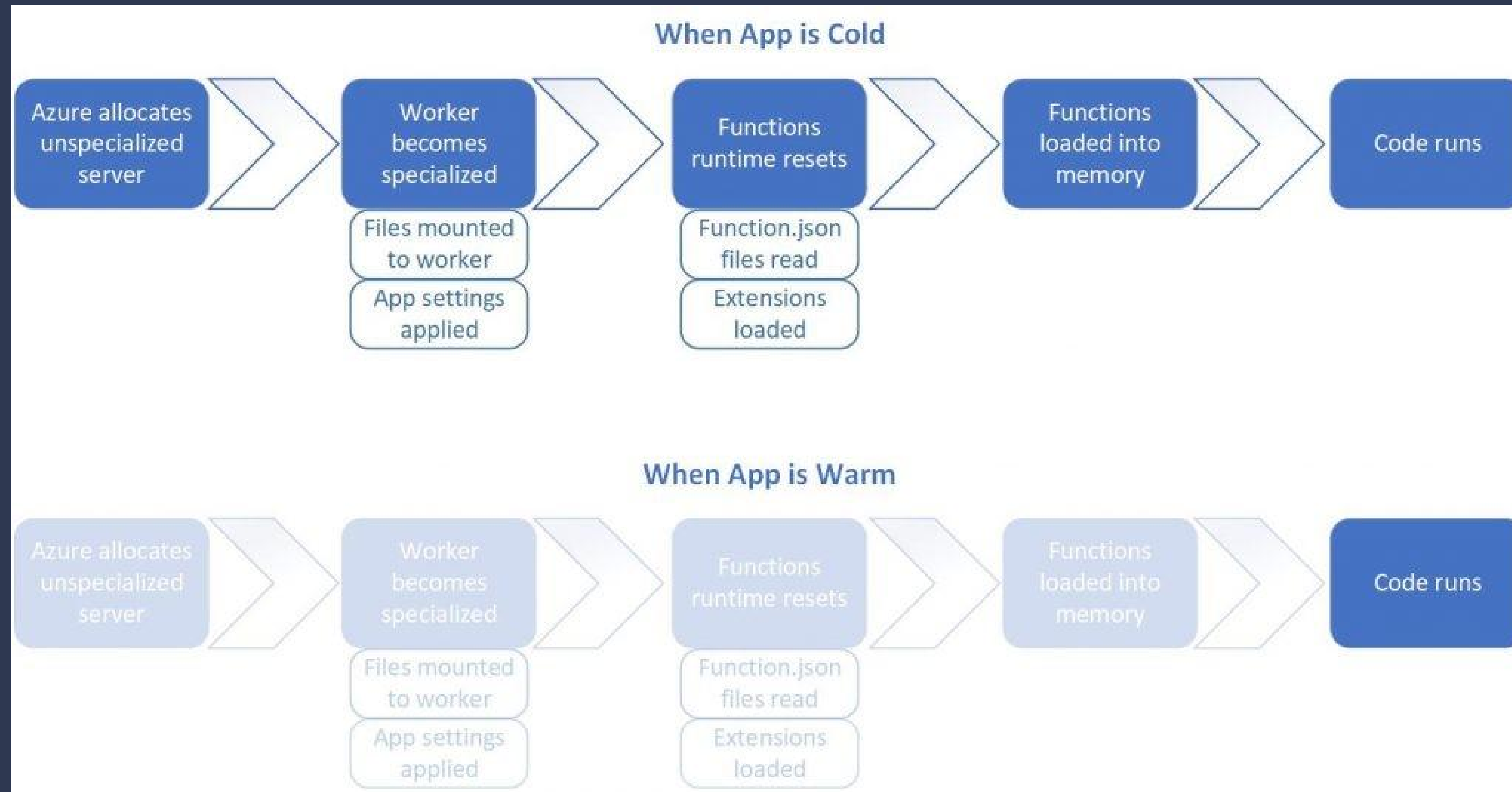


# Functions的运行时（Runtime）缩放



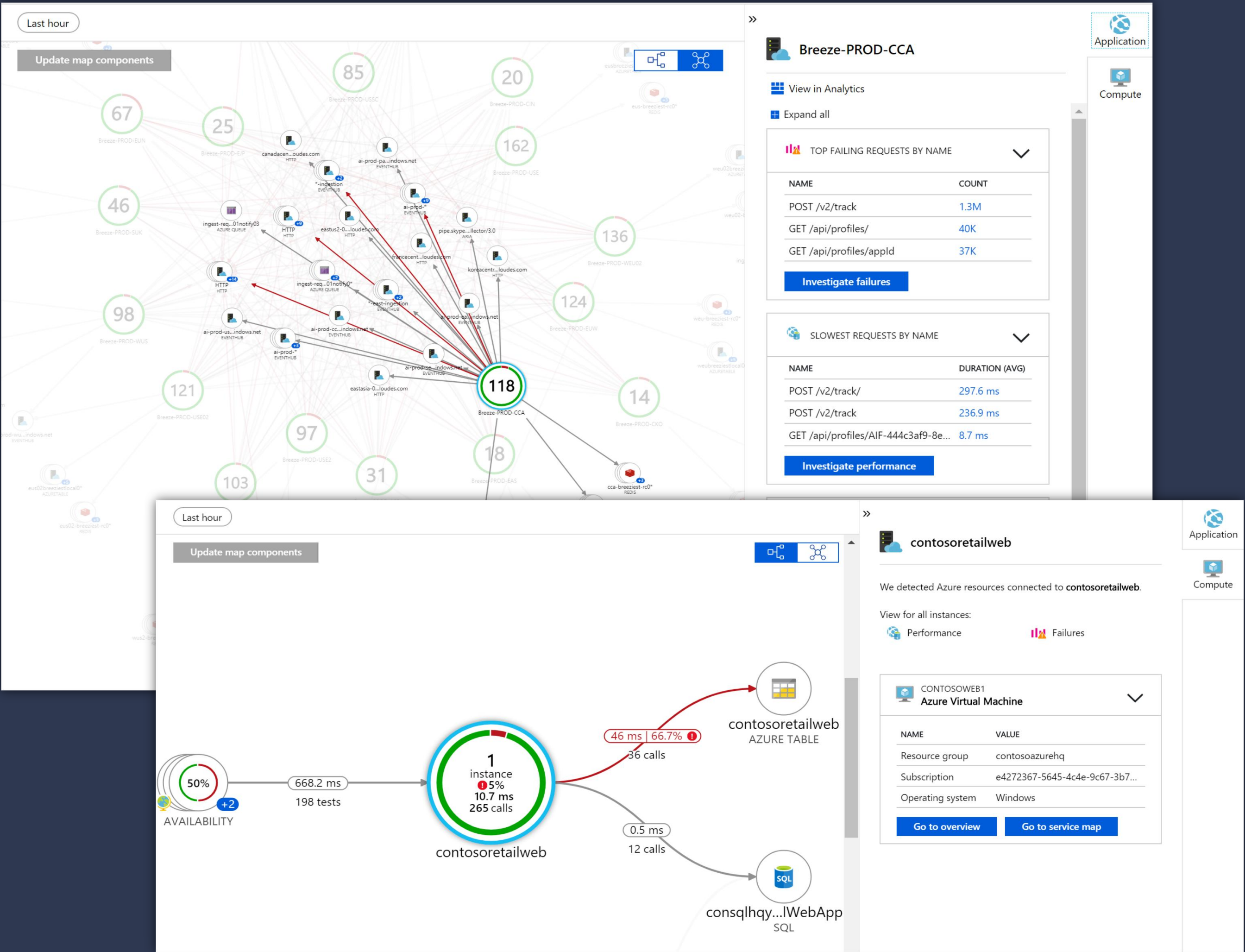
# Functions冷启动问题解决方案

公有云提供多种SKU，使用高层级或专属计划时，确保至少有一个函数实例始终运行，可以没有冷启动。



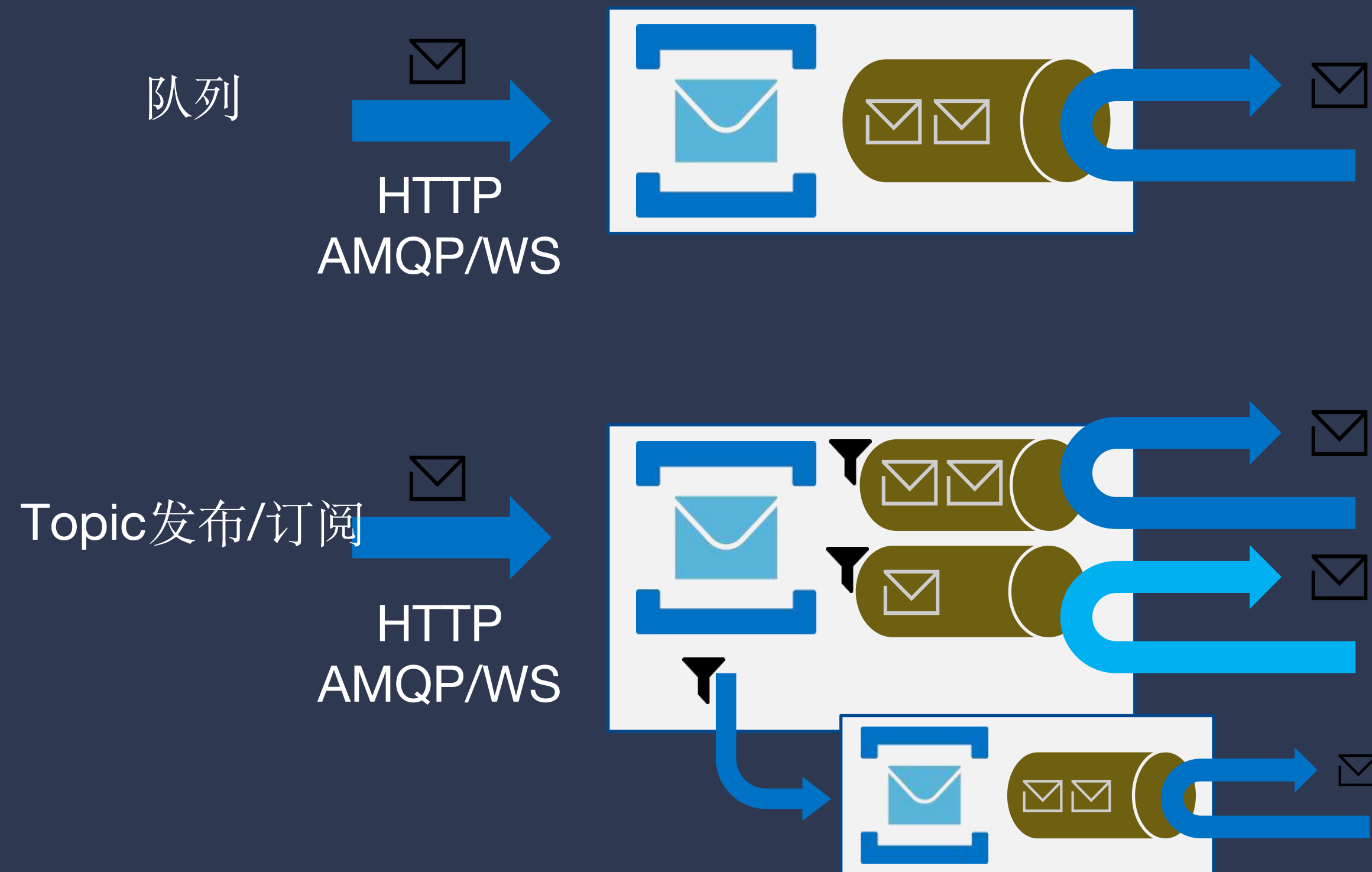
# 使用Function原生集成的Application Insights对应用进行诊断监控

- 通过Application Map和端到端交易详细信息进行跟踪  
使用实时指标流实时可视化事件和指标





# 云原生消息服务Service Bus高级功能



## 架构目的:

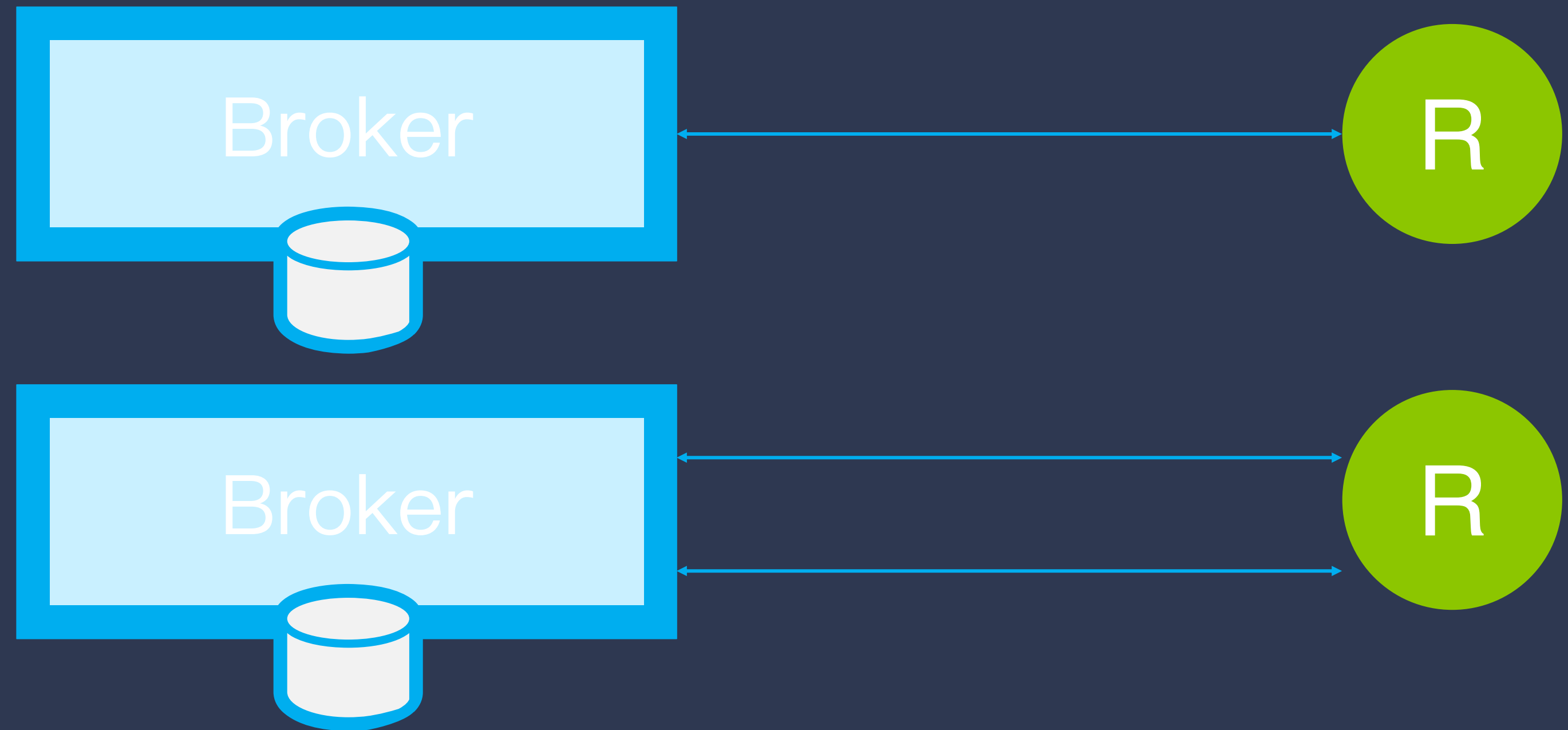
- 消息服务
- 应用解耦
- 负载均衡 (削峰填谷)
- 发布订阅

## 高级特性:

- 会话支持
- 事务性支持
- 自动转发
- 死信队列 (DLQ)
- 定时发送
- 消息延时
- 重复消息检测
- 消息过滤
- 出现空闲队列时自动删除
- 安全控制 (SAS token, RBAC, managed identity)
- 跨区federation
- 多可用区支持
- 异地灾备恢复
- 兼容行业标准 (AMQP, [JMS](#))
- 多语言SDK支持: Java/JavaScripts/Python/.NET

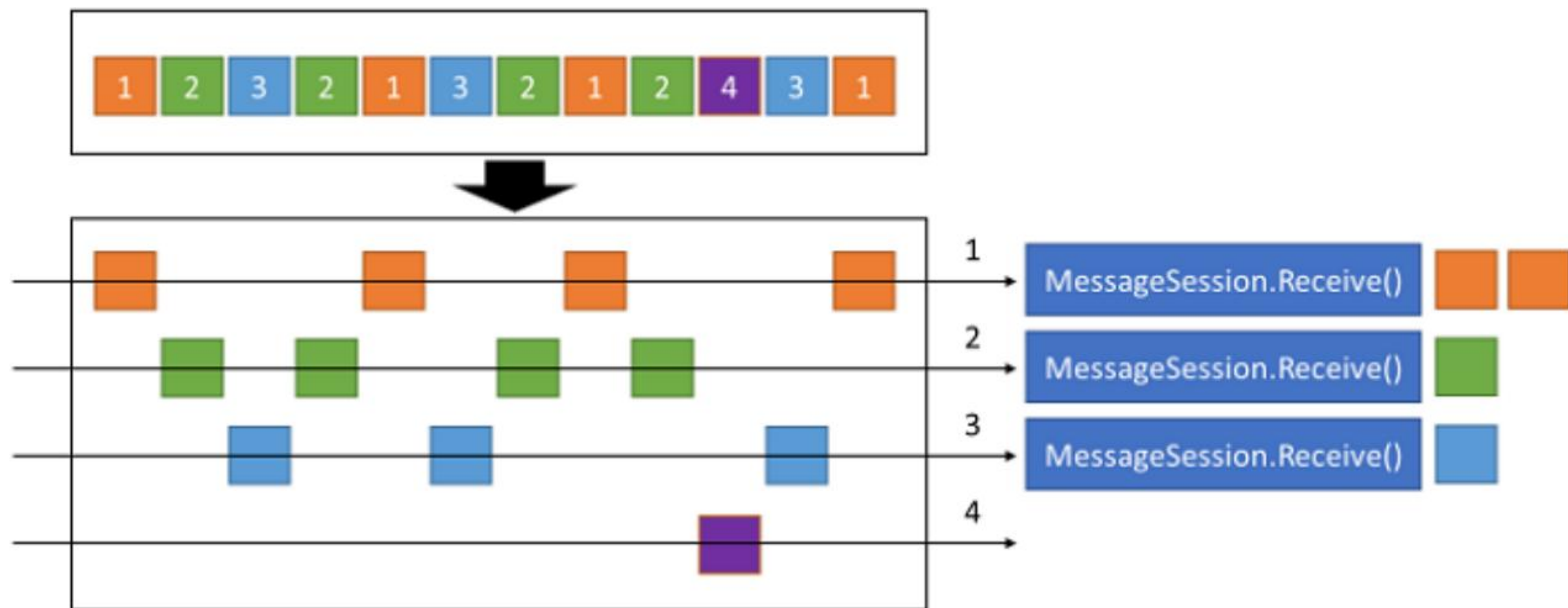
# 确保消息不丢失,不重复

- Peek Lock方式确保消息不丢失:消息在消费端没有返回确认前,并且在锁定时间内消息不会删除,锁定超时后,被其他客户端拉去。
- 重复检测窗口机制确保消息在设定时间内不重复



# 消息先进先出 (FIFO) 特性

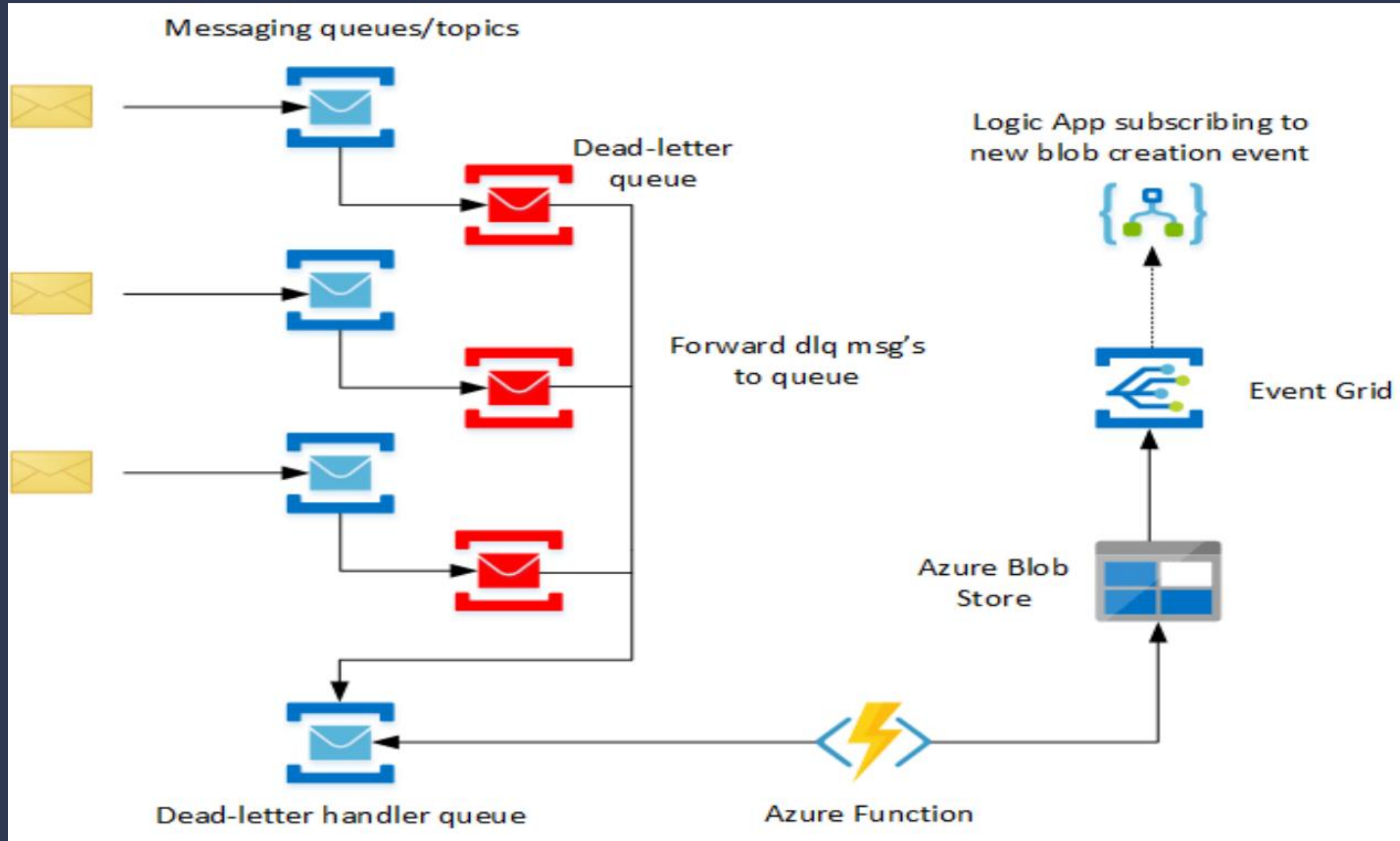
需要先进先出的一组消息，设置相同的Session ID





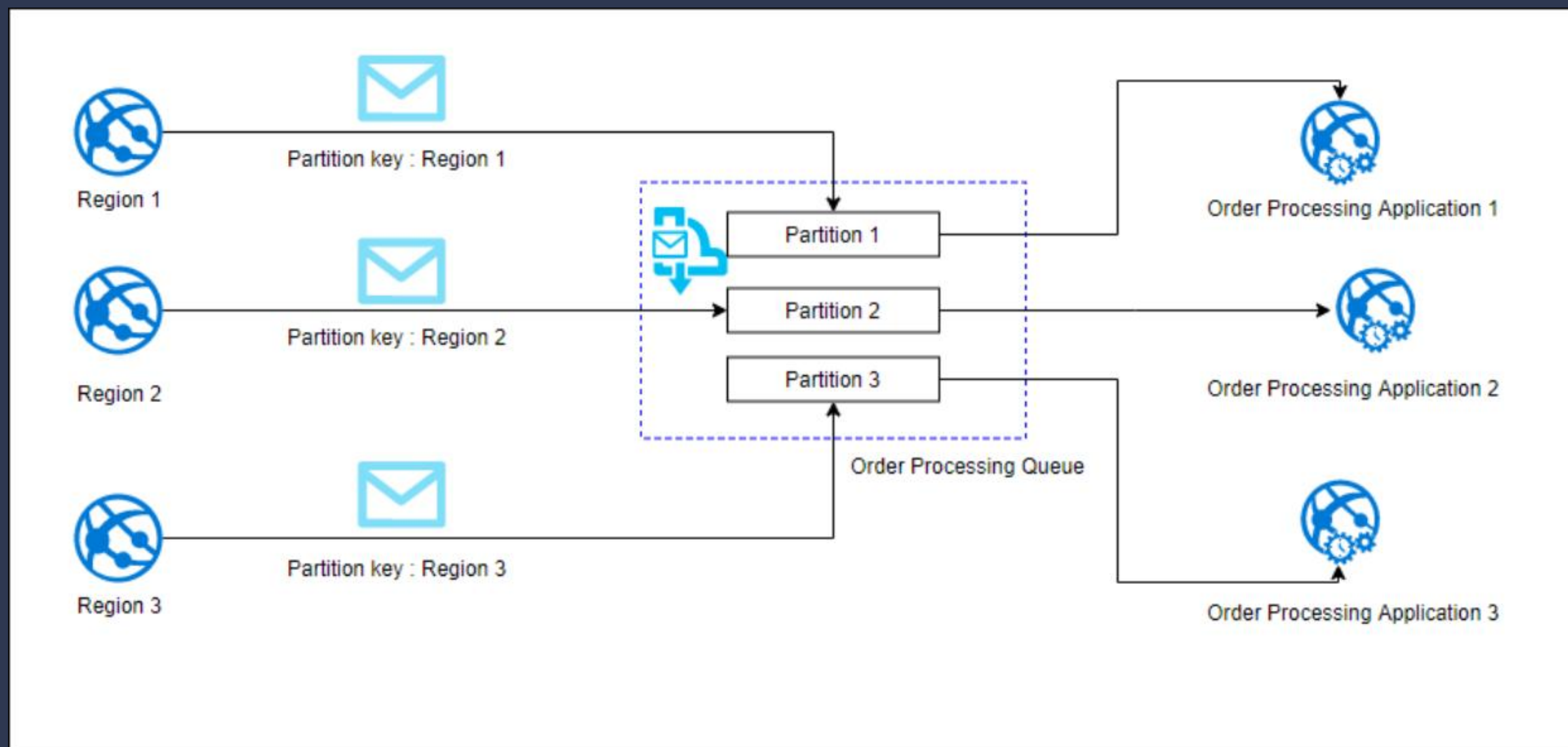
# 通过死信(Dead Letter)队列处理异常消息

死信队列支持事务性操作



# 消息并发处理机制

- 多种性能服务层级
- Service Bus Partitioning
- 多客户端负载均衡
- 批量处理访问
- 消息预提取



# 经验总结

- Serverless架构的优点：低成本、高可靠、高可扩展性
- 云原生消息服务的优点：异步解耦、高并发、可靠性高
- 在设计全球数据同步方案时，需要考虑业务场景的不同实时性要求，合理技术架构
- 采用云原生 Serverless 和消息服务解决方案，可快速构建可靠的跨数据中心全球务数据同步解决方案，具有高扩展性和高可靠性，能够满足业务需要
- 在设计方案时，需要考虑到业务数据安全性和合规性，选择能够满足合规要求的云服务提供商
- 在实施方案时，需要充分考虑监控，链路跟踪，异常数据处理机制，确保数据同步过程的可靠性



想一想，我该如何把这些  
技术应用在工作实践中？

---

THANKS