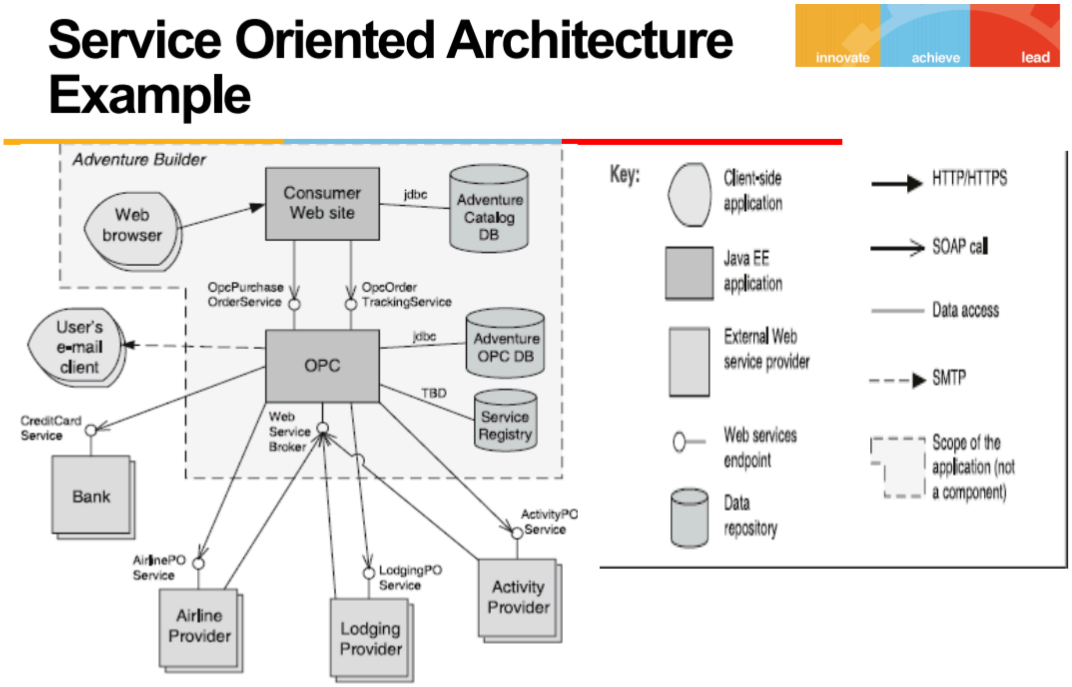
The Road Ahead: SOA, Cloud, and CAP Theorem

1. Service-Oriented Architecture (SOA) Pattern

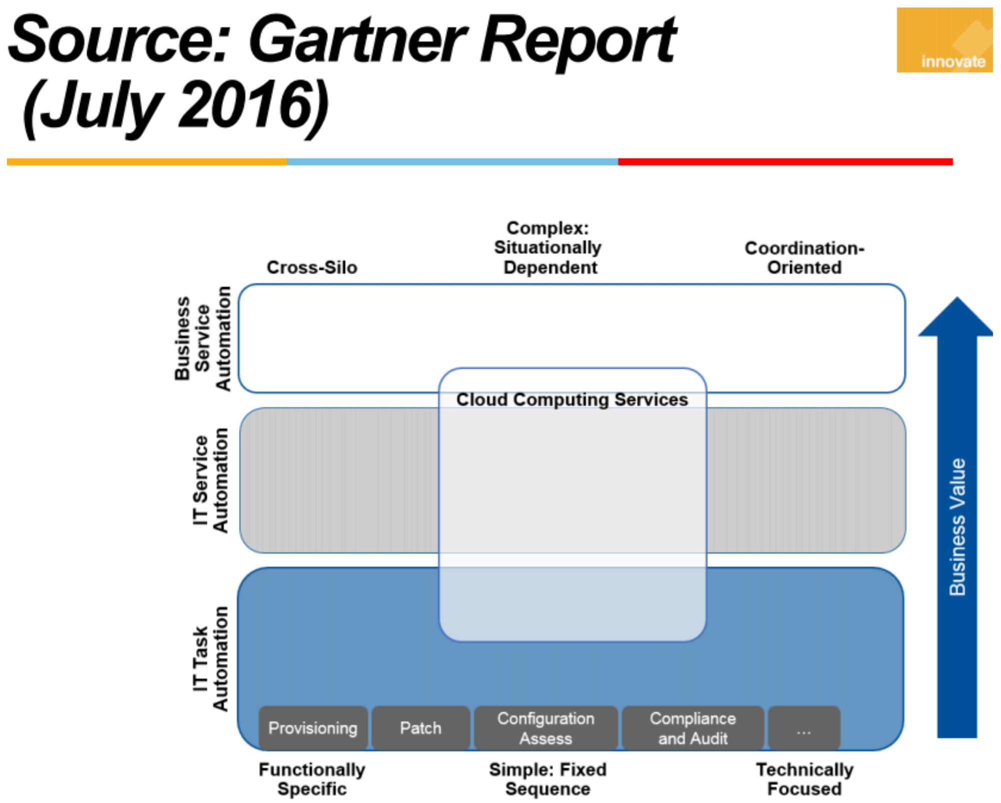
- **Context**:
  - Services are offered by providers and consumed by users over a network. Consumers need to understand and use these services without knowing their implementation details.
- **Problem**:
  - Supporting interoperability across distributed components running on different platforms, written in diverse languages, and offered by various organizations.
- **Solution**:
  - SOA comprises distributed components that provide or consume services.
  - **Example**: Multiple web services interacting via APIs to perform tasks like authentication, payment processing, and notifications.
- **Elements**:
  - **Service Providers**: Offer services through defined interfaces.
  - **Service Consumers**: Invoke services either directly or through intermediaries.
  - **ESB (Enterprise Service Bus)**: Routes and transforms messages between providers and consumers.
  - **Registry of Services**: Helps consumers discover available services at runtime.
  - **Orchestration Server**: Manages interactions between services based on workflows.
- **Connectors**:
  - **SOAP Connector**: For synchronous communication using the SOAP protocol.
  - **REST Connector**: Uses HTTP operations for request/reply interactions.
  - **Asynchronous Messaging Connector**: Facilitates asynchronous communication via messaging systems.
- **Weaknesses**:
  - SOA-based systems can be complex, with middleware introducing performance overhead and potential bottlenecks.
- **Diagram**:



Service Oriented Architecture Example

2. Cloud Computing Strategies

- **Key Challenges**:
  - Many organizations fail in cloud implementation due to the lack of a cloud strategy linked to business outcomes.
  - Uncertainty about starting cloud projects can hinder timely business opportunities.
- **Recommendations**:
  - Identify the cloud services to offer or procure.
  - Document internal processes impacted by cloud services.
  - Map applications and workloads to cloud services.
- **Diagram**:



Source: Gartner Report (July 2016)

3. CAP Theorem

- **Definition**:
  - The CAP Theorem explains that a distributed system cannot simultaneously provide all three guarantees:
    - **Consistency**: All nodes should have the same data at the same time.
    - **Availability**: System remains operational despite node failures.
    - **Partition-Tolerance**: Continues operation even with network partitions.
- **CAP Theorem Scenarios**:
  - **AP Database (Not Consistent)**:
    - **Scenario**: Data is read from one node, while data is being written to another node.
    - **Examples**: CouchDB, Cassandra, ScyllaDB.
  - **CP Database (Not Available)**:
    - **Scenario**: Data in one partition is locked for consistency, but node is unavailable.
    - **Examples**: MongoDB, Redis.
  - **CA Database (Not Partition-Tolerant)**:
    - **Scenario**: All nodes show the same data, but cannot handle network partitions (theoretical).