



**LEEDS  
BECKETT  
UNIVERSITY**

**Software Engineering, Technology, and Emerging Practices Research Group (SETEP), School of Built Environment, Engineering & Computing, Leeds Beckett University, Leeds**  
**m.ramachandran@leedsbeckett.ac.uk**



Methods and Design Principles

Process: Business Process Driven Service Development Lifecycle (BPD-SDL)

Reference Architecture

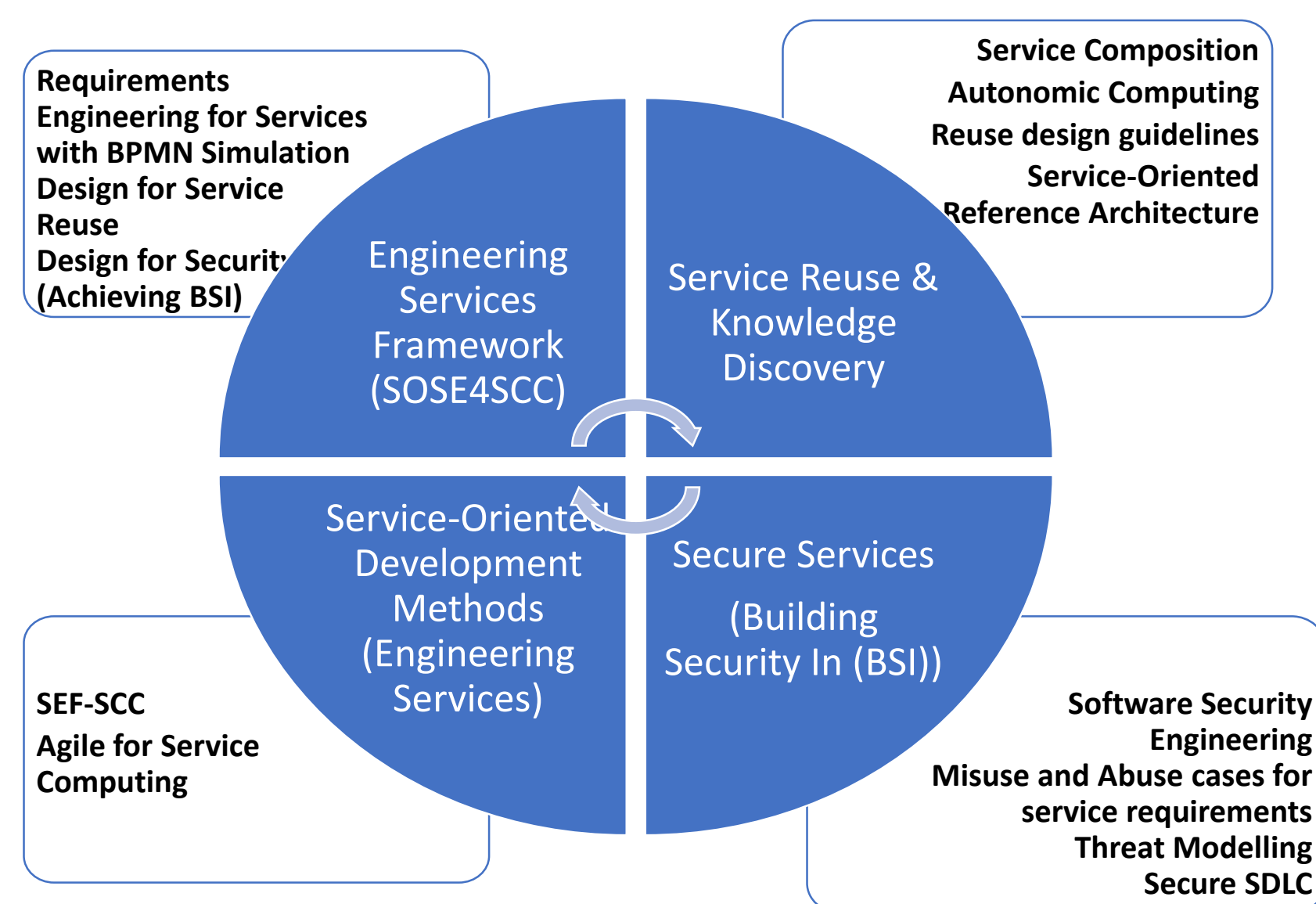
Tools

SE-SCC Services: Software Engineering as a Service (SEaaS): SPMAaaS, SPIaaS, SSREMAaaS, CCAFAaaS, SE for BD, SE for IoT, SE for Cyber-Physical Systems

Adoption Models

Evaluation & Applications

## Four Pillars of Service Computing Principles



**Machine Learning & Knowledge Discovery for Reuse Patterns**

Cloud Reuse SE

Design for Reusable Cloud Services

Design with Cloud Services & Web Services

 <p>Ramachandran, M (2011) Knowledge Engineering for Software Development Life Cycles, IGI Global Publishers</p>	 <p>Ramachandran, M (2012) Software Security Engineering: Design and Applications, Nova Scientific Publisher</p>	 <p>Ramachandran, M (2008) Software Components: Guidelines and Applications, Nova Scientific Publisher</p>
<p><b>Service Development</b></p> <ul style="list-style-type: none"> <li>• Accuracy, Correctness &amp; QoS Design Principles</li> <li>• Service RE with BPMN &amp; Simulation</li> <li>• Service Design with Service Components (SoaML)</li> <li>• Service Development (any platform)</li> <li>• Service Testing &amp; Deployment &amp; Continuous Delivery</li> </ul>	<p><b>Service Security Engineering</b></p> <ul style="list-style-type: none"> <li>• Building Security In (BSI), Resiliency, Fault-Tolerance Design Principles</li> <li>• Service Security RE with Misuse &amp; Abuse Use cases for all identified services</li> <li>• Threat Modelling</li> <li>• Design for Security</li> <li>• Building Security In &amp; Resiliency, Fault-tolerance,</li> <li>• Software security testing</li> </ul>	<p><b>Service Reuse Engineering</b></p> <ul style="list-style-type: none"> <li>• Design for Reuse &amp; Design with Reuse, Composable, Scalable Design Principles</li> <li>• Reuse RE (Commonality &amp; Variability Analysis of secured requirements on selected BPMN &amp; Secured use cases)</li> <li>• Design for reuse approaches</li> <li>• Reuse Development (Implementing composable services)</li> <li>• Testing for reuse, composition &amp; integration</li> </ul>

The diagram illustrates the Business Process Driven Service Development Lifecycle (BPD-SDL) as a continuous cycle of six stages, each represented by a colored wedge in a circle. The stages are: Service Requirements Engineering with BPMN modelling & simulation (orange), Conduct BPMN workflows: Classify Services: Utility, Business, Coordination Services (grey), Interface Identification & specification using WSDL (yellow), Service Design with SoaML: service contract, Service Component Interface design, and SOA (blue), Service cost estimations using service component interfaces & modified COCOMO model (Guha 2013) (green), and Service Implementation: Soap based or RESTful based (orange). A blue arrow indicates the flow from Service Requirements Engineering to the table on the right.

Service Types Business Types	Task-Oriented Services	Entity-Oriented Services	Enterprise Services (B2B)
Utility Services			
Business Services			
Co-ordinations and choreographic services			
Human services			

**Business Process Driven Service Development Lifecycle (BPD-SDL)**

The diagram illustrates a reference architecture for SOA BD, BD Enterprise Service Bus, and BD Proofs... applications & Predicti... The architecture is divided into several layers and components:

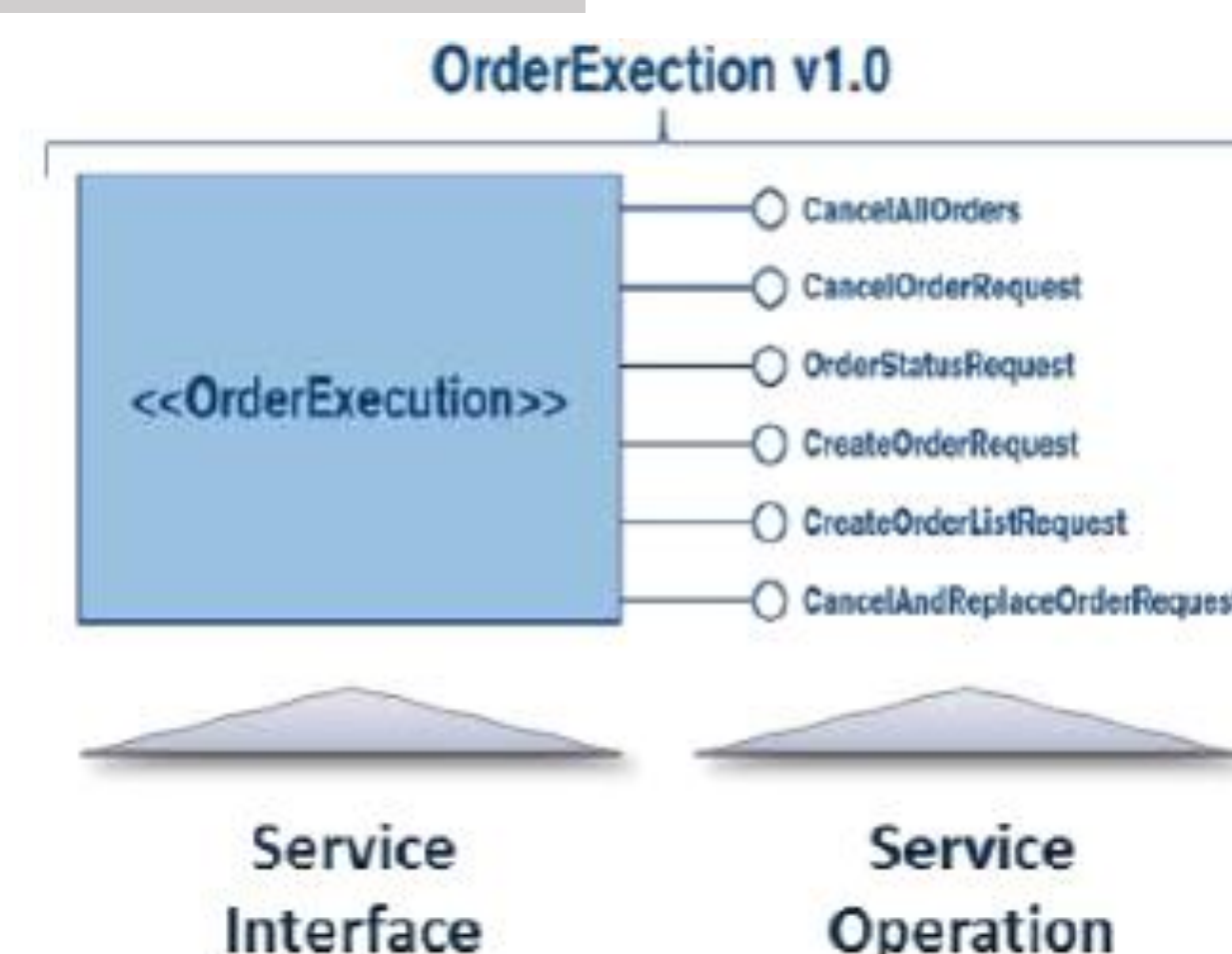
- Input Layer:** Big Data Source & SaaS, Web Service Data, Real-time Data.
- Data Streaming Service:** Receives input from the input layer and feeds into Data Security.
- Data Security:** Feeds into a decision point: Data Valid?
- Decision Point:**
  - If **No**, it leads to **Destroy Data**.
  - If **Yes**, it leads to **SLA: Multi-Cloud & Multi-Server**.
- Service Layer:**
  - SLA: Multi-Cloud & Multi-Server** feeds into **Vitalized Services**.
  - Vitalized Services** feeds into **Service Discovery**.
  - Service Discovery** feeds into **Service ...**.
  - Service ...** feeds into **Service Mediation**.
  - Service Mediation** feeds into **Message Queue management**.
  - Message Queue management** feeds into **Service Interface Connections**.
  - Service Interface Connections** feeds into **Service Integration**.
- Data Processing Layer:**
  - Service Integration** feeds into **Data Extraction**.
  - Data Extraction** feeds into **Data Processing**.
  - Data Processing** feeds into **Data Analysis**.
  - Data Analysis** feeds into **Data Storage & Retrieval**.
- Analytics Layer:**
  - Data Storage & Retrieval** feeds into **Data Analytics**.
  - Data Analytics** feeds into **Predictive Analytics**.
  - Predictive Analytics** feeds into **Visualisation**.
  - Visualisation** leads to **End**.
  - Data Analytics Reuse** feeds into **Predictive Analytics**.

A small line graph is included in the bottom right corner, showing a trend over time with peaks and troughs.

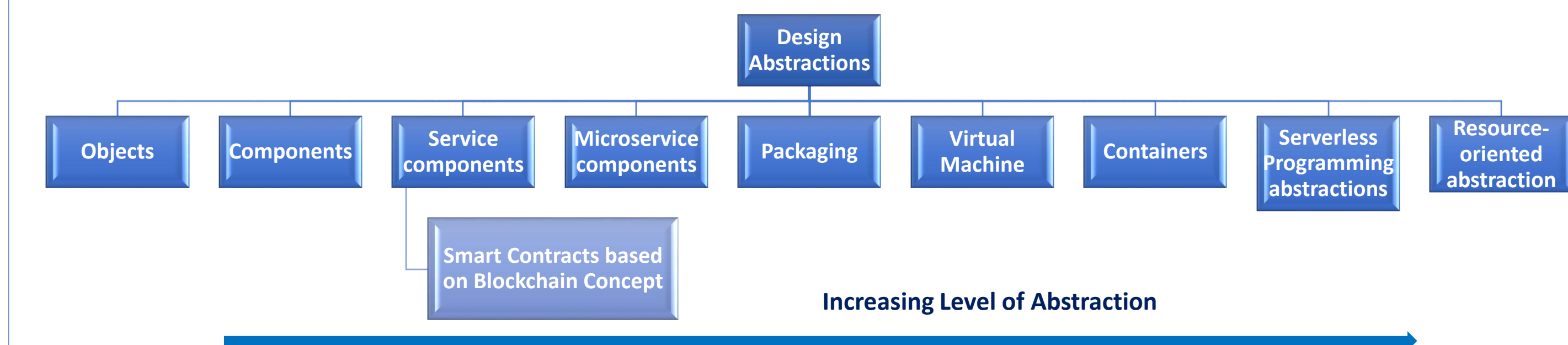
Service is a unit of solution logic to which service-orientation has been applied to meaningful extent.

$$Service\ Point\ (SP) = \sum_{i=1}^n (\pi \times P)$$

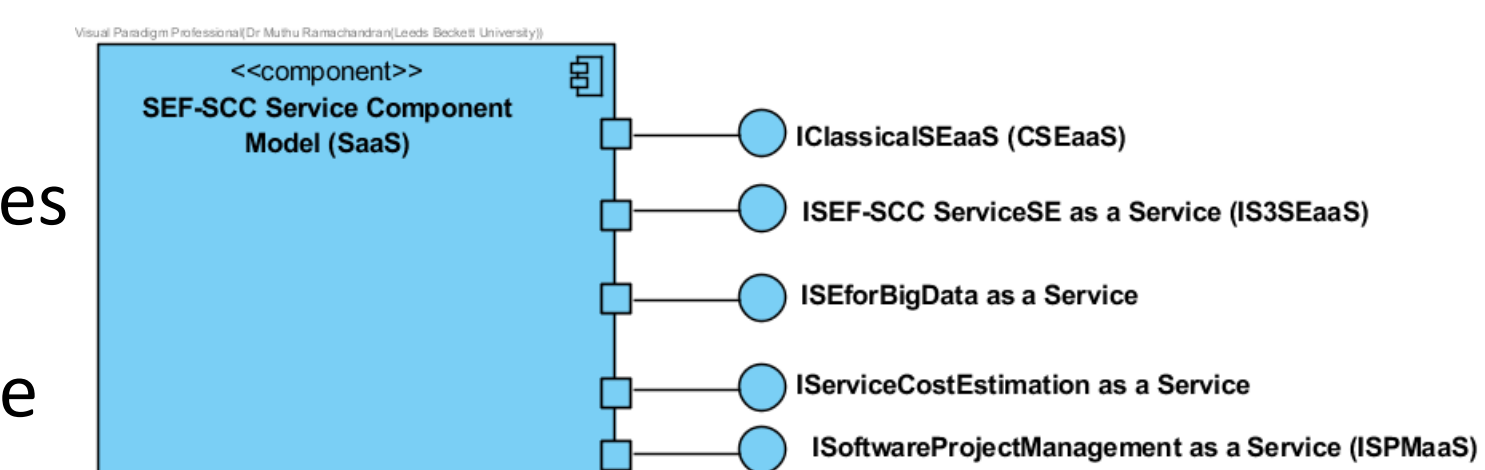
Where,  
 $\pi$  - infrastructure factor with empirical value  
 $P$  - single specific service's estimated size  
 varying with different service types.



## Design Abstraction: Mix & Match



- The best practice principles are:
- use port concept to define all interfaces
- follow best practice guidelines on interface design principles with service components



The diagram illustrates a layered architecture with the following components and relationships:

- Enterprise Service Bus:** A central horizontal layer that facilitates communication and data exchange between the layers above and below it.
- Business Services (Top Layer):**
  - Co-ordination & Orchestration Services:** A light blue box at the top left, connected to the bus.
  - Multi-Cloud Service Composition & Customisation of SEaaS, BPMNaaS:** An orange box at the top right, connected to the bus.
- Infrastructure Services (Bottom Layer):**
  - Core Infrastructure services:** A dark blue box at the bottom left, connected to the bus.
  - Infrastructure utility Services:** A dark blue box at the bottom middle, connected to the bus.
  - Infrastructure co-ordination services:** A dark blue box at the bottom right, connected to the bus.

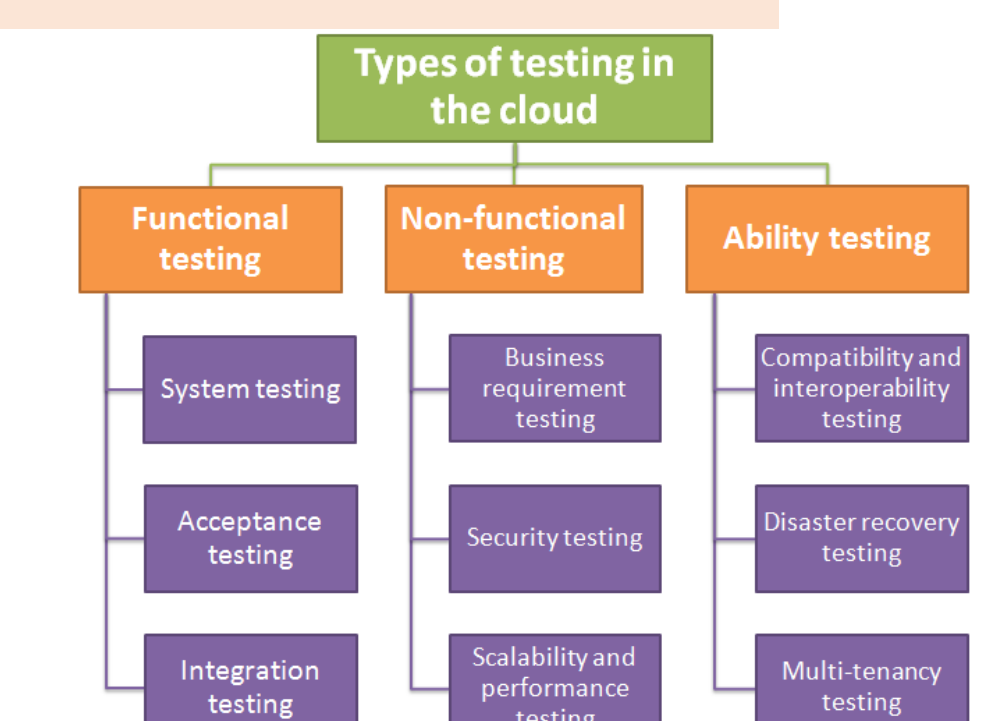
A vertical arrow on the left side points upwards, indicating the flow of data and services from the Infrastructure layer through the Enterprise Service Bus to the Business Services layer.

The diagram illustrates the relationship between Task-oriented services, Enterprise services (ES), and Business Layers. It is organized into three main columns: Task-oriented services, Enterprise services (ES), and Business Layers. The Task-oriented services column lists Utility, Business, and Coordination. The Enterprise services (ES) column lists the same three services. The Business Layers column is divided into three horizontal sections: Top Business Layer (containing Strategic and Core Competence Importance Control, and Strategic Importance in Channels), Middle Business Layer (containing Governmental Incentive Service Bus (E-Gov Service Bus), and a box for Strategic and Core Competence Importance Control), and Bottom Business Layer (containing Local Govt. Importance, Local Govt. Policy Importance, and Local Govt. Policy Importance). Arrows indicate the flow of information and services between these layers and the Enterprise services (ES).

Architect needs to categorise services  
therefore place them in the appropriate  
architecture layers

**Key principles of SEF-SCC service testing:**

- Performance Testing
- Security and Privacy Testing
- Build Security In (BSI) Testing
- Service Requirements Driven Testing
- Service Specification Testing (Automated BPEL)
- SEF-SCC Design Driven Tests
- Design for Service Testing Principles
- Automated Recovery and Safety Driven Tests
- Continuous Monitoring for abnormalities
- Boundary Value Analysis Testing
- Penetration Testing





<http://www.springer.com/978-3-030-33623-3>

Software Engineering in the Era of Cloud Computing

Ramachandran, M.; Mahmood, Z. (Eds.)

2020, XXIV, 354 p. 136 illus., 99 illus. in color.,

Hardcover

ISBN: 978-3-030-33623-3