SOA & Microservices - CCS3341

Module Code / Title:	CCS3341 SOA & Microservices
Assessment Component:	Coursework
Weighting:	60%
Handed out	Sunday 10th August 2025
Due Date:	Monday 1st September 2025 at 1pm Coursework demonstrations will be scheduled during 1st week of September
Learning Outcomes:	ILO1: Describe SOA to structure web-based system ILO2: Explain WS* services ILO3: Apply REST architecture ILO4: Implement microservices in cloud environments.
Expected Deliverables:	• **Design Artifacts:** SOA design doc, WSDL files, UDDI entries, governance policy. • **Source Code:** ○ CatalogService (Java SOAP WAR) ○ OrdersService (Spring Boot or Node.js REST) ○ BPEL process definitions ○ Integration code/config • **Configuration:** sun-jaxws.xml, web.xml, Spring Security/OAuth2, Integration exchanges/queues, BPEL deployment descriptors. • **Test Suites:** SOAP UI project; Postman or curl scripts; BPEL engine console logs; Integration queue status screenshots. • **Reflective Report :** Trade-off analysis • **Viva Slides/Script:** Step-by-step demo plan.

Coursework description and marking scheme

In this coursework you are asked to demonstrate the ability to analyse a problem and plan a development process for its solution. It covers LO1, LO2, LO3 and LO4.

Overview

GlobalBooks Inc. is migrating its legacy monolithic order-processing system to a Service-Oriented Architecture (SOA). Four autonomous services - Catalog, Orders, Payments and Shipping - must be designed, implemented, composed, secured and governed. You will deliver design artifacts, source code, integration configurations, governance policies and a live demonstration (viva) under realistic scenarios.

Scenario

GlobalBooks Inc. has grown into a global e-commerce platform serving millions across North America, Europe and Asia. Its original Java monolith handles catalog lookup, order placement, payment processing and shipment coordination - all within one tightly coupled codebase and database. During peak events (holiday promotions, author signings), the system buckles under load. Even minor updates (e.g., adding a new payment provider) trigger full regression tests and redeployments, risking weeks of downtime.

To solve this, GlobalBooks' CTO has approved a refactoring project:

- **Services:** Catalog, Orders, Payments, Shipping (each with its own data store)
- **Interfaces:** SOAP (legacy partners) and REST (new clients)
- **Registry:** UDDI-based central discovery
- **Integration:** RabbitMQ ESB for asynchronous messaging
- **Orchestration:** BPEL engine for the "PlaceOrder" workflow

- **Security:** WS-Security tokens on SOAP; OAuth2 on REST
- **Governance:** Versioning, SLAs (99.5% uptime; sub-200 ms responses), deprecation schedules

You will assume the roles of architect, developer and integration specialist, culminating in a viva demonstration of each component under real-world load and failure scenarios.

Task No	Task Description and Marking Scheme	Mark
1.	Explain which SOA design principles you applied when decomposing the monolith into independent services.	**10**
2.	Discuss one key benefit and one primary challenge of your approach.	**5**
3.	Provide a WSDL excerpt for the CatalogService (operations, types, binding)	**6**
4.	Draft the UDDI registry entry metadata enabling client discovery.	**4**
5.	Describe in detail how you implemented the CatalogService SOAP endpoint in Java (including sun-jaxws.xml and web.xml snippets).	**10**
6.	Explain how you tested it using SOAP UI (test cases and assertions).	**5**
7.	Design the OrdersService REST API: list endpoints (POST /orders, GET /orders/{id}), sample JSON request & response, and the JSON Schema for order creation.	**10**
8.	Outline the “PlaceOrder” BPEL process: receive, loop for price lookup via CatalogService, invoke OrdersService, reply to client.	**10**
9.	Explain deployment and testing on a BPEL engine (e.g., Apache ODE).	**5**
10.	Explain how you integrated PaymentsService and ShippingService: queue definitions, producers/consumers.	**7**

11.	Describe your error-handling and dead-letter routing strategy	**3**
12.	Detail WS-Security configuration for CatalogService (UsernameToken or X.509).	**4**
13.	Describe OAuth2 setup for OrdersService	**4**
14.	Explain one QoS mechanism you configured for reliable messaging (e.g., persistent messages, publisher confirms).	**2**
15.	Draft the governance policy: versioning strategy (URL & namespace conventions), SLA targets (availability, response time), and deprecation plan (notice period, sunset process).	**10**
16.	Deploy all four services (Catalog, Orders, Payments, Shipping) to a cloud platform	**5**

END OF THE PAPER