SMU013

LGB BANK: COMPOSITE MICROSERVICES IN A LARGE GLOBAL BANK (A)

Jane Blanston, a newly hired Enterprise Architect at LGB Bank, had developed over her career a specialisation in Service-Oriented Architecture (SOA). Prior to joining LGB in January 2019, she had recent successes in a smaller regional bank where she had replaced its outdated batch-mode file transfer style of integration with a real-time SOA style of integration. This provided the bank with a more flexible architecture and an abstraction layer which enabled the bank to make large scale changes in the backend (e.g., a core banking system replacement) with minimal impact to the bank's front-end banking channels (e.g., internet banking). Furthermore, with an SOA in place, the bank had the ability to rapidly assemble new solutions by reusing existing services.

Blanston was headhunted by LGB and offered a job based on her expertise in SOA and reputation as a problem solver. LGB specifically wanted her to review its current microservices-based architecture (a subset of SOA) and to propose an improved architecture which emphasized service reuse, like what she had implemented in the past. Even though LGB was a larger bank with larger integration problems, Blanston decided to take up the challenge. She accepted the job offer to join LGB's technology department based in Singapore.

LGB Bank

LGB Bank was a large global bank headquartered in New York with a large regional presence in Asia. Over the years, LGB had been aggressively expanding their business and had grown into a leading global bank that provided both commercial banking and bancassurance services.

Retail banking services were delivered to retail customers who were members of the general public. This customer segment was also known as consumer banking or personal banking. The bank had branches located in major cities across the world. Some of the services offered by the retail banking division were checking and savings accounts, fixed deposit accounts, home mortgages, auto financing, credit cards, investment products, and foreign currency remittance services.

Corporate banking services were delivered to Small and Medium-sized Enterprises (SME)'s, regional corporations, large multi-national corporations, and institutional banking customers like pension funds and hedge funds. Some of the services offered by the corporate banking division were working capital financing, project financing, commercial real estate mortgage, payments and cash management products, treasury and investment products, and trade finance.

This case was written by Assistant Professor Alan Megargel and Assistant Professor Christopher M. Poskitt at the Singapore Management University. The case was prepared solely to provide material for class discussion. The author does not intend to illustrate either effective or ineffective handling of a managerial situation. The author may have disguised certain names and other identifying information to protect confidentiality.

LGB had been aggressive with their technology investments and was considered a leader in the adoption of enabling technologies such as Service-Oriented Architecture (SOA), Microservices-Based Architecture (MSA), Business Process Management (BPM), and Robotic Process Automation (RPA). LGB would "incubate" their enabling technologies in Asia, before rolling out the solutions globally. LGB had a strong governance process in place, which enforced enterprise-wide standards. System designs (good or bad) churned out in Asia had global impact.

INSCO: A Third-Party Insurance Company

INSCO was a large global insurance company with branch offices in Asia. INSCO offered a full range of retail insurance products including health insurance, home insurance, vehicle insurance, and travel insurance. Like many banks, LGB offered bancassurance products as part of a cross-selling strategy to grow the share-of-wallet with its existing customers. LGB had a long-standing partnership with INSCO as a third-party provider of retail insurance products, which LGB could then bundle together with their retail bank products.

Initial Architecture

The partnership between LGB and INSCO preceded the advent of MSA and Open API standards. As such, the initial integration between LGB and INSCO was done through batch-mode file transfer. File transfers ran on a daily scheduled batch such that a request-response interaction between LGB and INSCO took at least one day to complete. As MSA and Open API standards became prevalent, both parties independently uplifted their IT architectures accordingly.

Travel Insurance – Problem Statement

It took three days to issue a travel insurance policy, but most travelers needed insurance immediately.

INSCO provided an API, but all of the processing at LGB (e.g., creating an insurance account for a customer, confirming an insurance plan with INSCO, transferring the insurance premium amount from the customer's deposit account to INSCO's corporate account, etc.) was done in batch mode (i.e., to perform operations at a scheduled time), which took three days. To reduce the delay in issuing travel insurance, the process needed to be automated.

LGB had two options to automate the process:

1. Build or buy a new monolithic application.
2. Assemble Composite Microservices to automate the process.

Seeing a familiar problem, Blanston felt confident that she could automate the process by assembling composite services from LGB's existing catalogue of reusable microservices. Referring to her previous job where she had had successes in rapidly assembling new solutions by reusing existing services, she was able to convince LGB to select option 2 above.

Microservice Attributes

Blanston began to socialise the attributes of an SOA-layered architecture, and the benefits of service reuse. In her previous job, she had defined two distinct layers of services within an SOA. The bottom layer of services called "Atomic" or simple services encapsulate a single entity (e.g., Customer, Product, Account, etc.) and were designed for optimum reuse. Atomic services had exclusive access to data which they "owned". Atomic services never invoked other services. The top layer of services called "Composite" or complex services encapsulated a single process (e.g., originate loan account, evaluate credit worthiness, etc.) and orchestrated or aggregated other Atomic or Composite services in order to automate a process. Composite services typically did not "own" their own data. To distinguish the different categories of services, Blanston pointed out that the names of Atomic services were "things" and the names of Composite services were "actions".

Blanston realised that MSA was a subset of SOA, and that microservices were simply traditional SOA-based services with additional attributes, namely microserves were a) independently deployable (e.g., through containerisation technology), b) independently scalable such that multiple instances of the same service could handle increases in loading, and c) could be developed using any programming language. Blanston listed relevant microservice attributes for her LGB colleagues to assess (refer to **Exhibit 1**).

LGB had started developing and deploying Atomic microservices before Blanston had joined the team; however, they had not developed or conceived any composite microservices. Before Blanston joined, LGB had intended to build or buy monolithic (i.e., all in one) applications to implement process automation.

LGB Bank Architecture Principles

LGB had two different interaction designs for its microservices. Some microservices were designed to interact via Advance Message Queuing Protocol (AMQP), an asynchronous message-based standard. Some microservices were designed to interact via Hypertext Transfer Protocol (HTTP), a synchronous invocation-based standard. The role of composite microservices was to interact with multiple atomic microservices to orchestrate a process. Blanston introduced two new architecture principles to guide the interaction design depending on whether or not the user or process required an immediate response, as follows:

Principle P1: Invocation-based Interaction

If a process or a user needed an immediate response, then the interaction with all of the relevant underlying microservices would need to use an invocation-based, synchronous interaction. The rationale for this principle was to make it straightforward to receive a response and to avoid the perception of slow performance. The implication was that the process/UI and its underlying microservices could be tightly coupled.

Principle P2: Message-based Interaction

If a process or a user did not need an immediate response, then the interaction with all of the relevant underlying microservices could use a message-based, asynchronous interaction. The rationale for this principle was that message-based interaction could enable loose coupling. The implication was that it added complexity to the process due to the message broker.

LGB Bank Existing Atomic Microservices

LGB had existing atomic microservices in place (refer to **Exhibits 2, 3, and 4**). For each microservice, the description depicted the operations (e.g., "Create", "Get", "Update") and whether or not the microservice published an event out to other microservices or subscribes to events which were published from other microservices. And "event" represented a state change in the data owned by the microservice, e.g., in the case of the Deposit Account microservice, an event was published whenever a deposit account was created or updated. Other microservices which needed to know about that status change in a deposit account would subscribe to that event, e.g., "Transaction Journal", "Real-time Offer", or "Anti-Money Laundering".

Atomic Microservices (API's) Hosted by INSCO

As MSA and Open API standards became prevalent, INSCO independently uplifted their IT architecture accordingly, concurrently with the architecture uplift activities at LGB. In doing so, INSCO developed the capability to expose its business services to external third parties, via a set of atomic microservices (API's) (refer to **Exhibit 5**). This enabled external third-party resellers of insurance products, such as LGB, to have access to INSCO's insurance plans, calculate insurance premiums, prepare insurance proposals, and initiate new insurance policies.

INSCO documented their API's so that third-party insurance reseller companies could easily integrate to and include INSCO's APIs into their end-to-end insurance policy origination processes. Since INSCO had a prior business relationship with LGB, they approached LGB to pilot such a process, and Blanston was assigned to be the lead architect for LGB.

Description of Insurance Documents

Insurance products are traditionally document-centric. Before architecting the end-to-end automated process, Blanston first needed discover and understand the various insurance related documents. The three main documents involved (refer to **Exhibit 6**) included the Insurance Plan which is marketed by the insurance company, the Insurance Proposal which is offered by the insurance company to the customer, and the Insurance Policy which is purchased by the customer.

Travel Insurance Issuing Process

With an understanding of the insurance documents provided by INSCO, Blanston then pieced together the improved business process needed for LGB to issue travel insurance to their customers, including automated steps involving the invocation of INSCO API's. The process required human input via a customer-facing user interface, involving a sequence of five (5) human-triggered process steps (refer to **Exhibits 7, 8 and 9**).

Step 1: Get Insurance Plans

In step 1, the LGB customer selected "Travel" as the type of insurance. This triggered a request-reply invocation, over the internet, to INSCO's API called "Get Insurance Plans". The INSCO API returned a list of all insurance plans related to travel, including marketing level details, and payment options for each insurance plan. Since the user expected an immediate response after selecting the insurance type, Architecture Principle P1 (invocation-based interaction) was used (refer to **Exhibits 7**).

Step 2: Calculate Insurance Premium

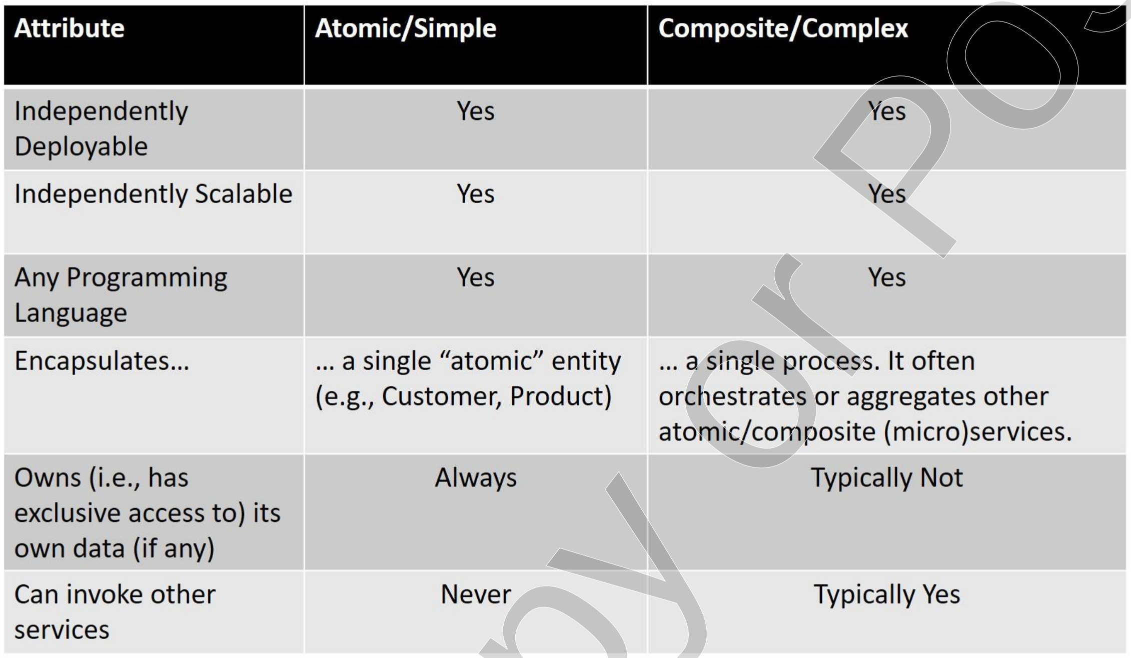
In step 2, the LGB customer selected a travel insurance plan, and selected payment options related to that plan. This triggered a request-reply invocation, over the internet, to INSCO's API called "Calculate Insurance Premium". The INSCO API returned an insurance premium ($ amount). Since the user expected an immediate response after selecting an insurance plan and payment option, Architecture Principle P1 (invocation-based interaction) was used (refer to **Exhibits 8**).

Step 3: Get Payment Modes

In step 3, after receiving the insurance premium amount, the LGB customer needed to select a payment mode. The payments modes available to the customer were retrieved by invoking three LGB internal atomic microservices: "Deposit Account", "Card Account", and "Loyalty Points" (refer to **Exhibits 2 and 4**). Rather than invoking these three microservices individually from the user interface, Blanston decided to implement a composite microservice called "Get Payment Modes" which could aggregate the available payment modes into one response. The microservices would take the Customer ID as an input and return a list of available deposit and card accounts, and the loyalty points balance for that customer (refer to **Exhibit 9**). Blanston started to draw the interactions between the "Get Payment Modes" composite microservice (refer to **Exhibit 10**) and the three atomic microservices: "Deposit Account", "Card Account", and "Loyalty Points" and considered the following:

1. Which architecture principle (P1 or P2) would she employ for each interaction?
2. What would be the information returned from each microservice? What name/label would she include in her diagram for each return value?
3. What would be the sequence of invocations from the composite microservice to each of the atomic microservices?
4. Could any of the atomic microservices be invoked concurrently?

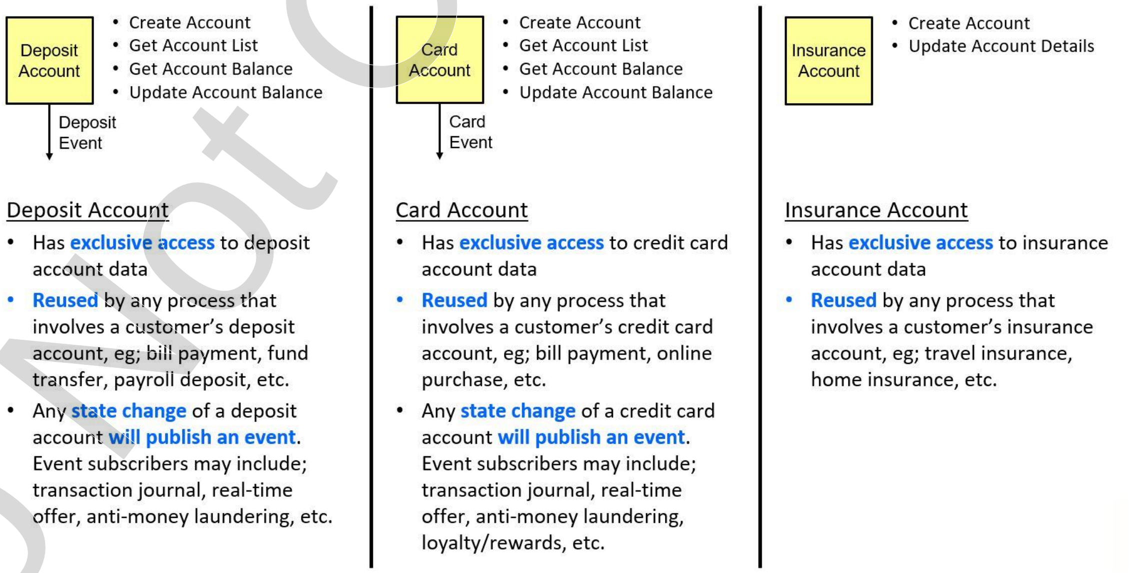
EXHIBIT 1: MICROSERVICE ATTRIBUTES



Attribute	Atomic/Simple	Composite/Complex
Independently Deployable	Yes	Yes
Independently Scalable	Yes	Yes
Any Programming Language	Yes	Yes
Encapsulates…	… a single "atomic" entity (e.g., Customer, Product)	… a single process. It often orchestrates or aggregates other atomic/composite (micro)services.
Owns (i.e., has exclusive access to) its own data (if any)	Always	Typically Not
Can invoke other services	Never	Typically Yes

Source: Authors' Concept Diagram

EXHIBIT 2: LGB BANK EXISTING ATOMIC MICROSERVICES



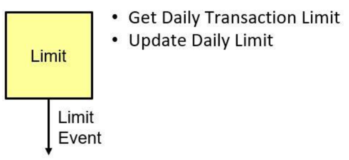
Deposit Account
- Has **exclusive access** to deposit account data
- **Reused** by any process that involves a customer's deposit account, eg; bill payment, fund transfer, payroll deposit, etc.
- Any **state change** of a deposit account **will publish an event**. Event subscribers may include; transaction journal, real-time offer, anti-money laundering, etc.

Card Account
- Has **exclusive access** to credit card account data
- **Reused** by any process that involves a customer's credit card account, eg; bill payment, online purchase, etc.
- Any **state change** of a credit card account **will publish an event**. Event subscribers may include; transaction journal, real-time offer, anti-money laundering, loyalty/rewards, etc.

Insurance Account
- Has **exclusive access** to insurance account data
- **Reused** by any process that involves a customer's insurance account, eg; travel insurance, home insurance, etc.

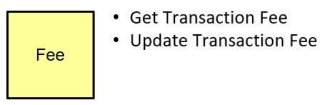
Source: Authors' Concept Diagram

EXHIBIT 3: LGB BANK EXISTING ATOMIC MICROSERVICES (CONT.)



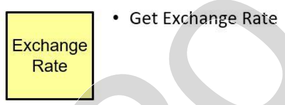
Limit

- Has **exclusive access** to transaction limits data
- **Reused** by any process that involves a customer's transaction limit, eg; bill payment, fund transfer, etc.
- Any **state change** of a transaction limit **will publish an event**. Event subscribers may include; overdraft offer, fraud detection, etc.



Fee

- Has **exclusive access** to transaction fee data
- **Reused** by any process that involves a financial transaction, eg; bill payment, fund transfer, teller deposit, ATM withdrawal, etc.

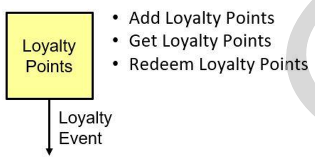


Exchange Rate

- Has **exclusive access** to foreign exchange rate data
- **Reused** by any process that involves a foreign exchange, eg; fund transfer, online purchase, cross-border payment, etc.

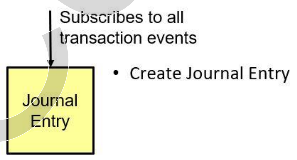
Source: Authors' Concept Diagram

EXHIBIT 4: LGB BANK EXISTING ATOMIC MICROSERVICES (CONT.)



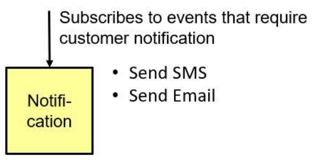
Loyalty Points

- Has **exclusive access** to loyalty points data
- **Reused** by any process that involves a customer's loyalty points, eg; bill payment, online purchase, etc.
- Any **state change** of loyalty points **will publish an event**. Event subscribers may include; product performance analysis, etc.



Journal Entry

- Has **exclusive access** to the bank's transaction journal
- **Subscribes to** all transaction **events**, eg; payment events, deposit events, card events, etc.
- Transaction journal entries are then posted into the bank's general ledger during overnight batch processing.

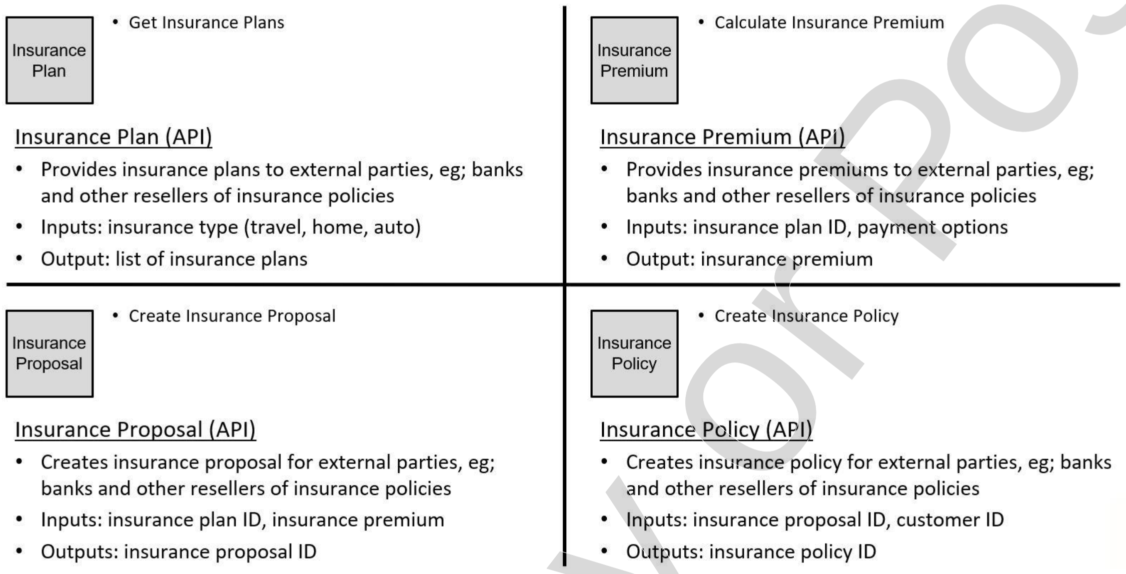


Notification

- Has **exclusive access** to SMS and Email services, and notification log
- **Subscribes to events** that require customer notification, eg; payment events, insurance policy events, add beneficiary events, etc.

Source: Authors' Concept Diagram

EXHIBIT 5: ATOMIC MICROSERVICES (API'S) HOSTED BY INSCO



- Get Insurance Plans

Insurance Plan (API)
- Provides insurance plans to external parties, eg; banks and other resellers of insurance policies
- Inputs: insurance type (travel, home, auto)
- Output: list of insurance plans

- Calculate Insurance Premium

Insurance Premium (API)
- Provides insurance premiums to external parties, eg; banks and other resellers of insurance policies
- Inputs: insurance plan ID, payment options
- Output: insurance premium

- Create Insurance Proposal

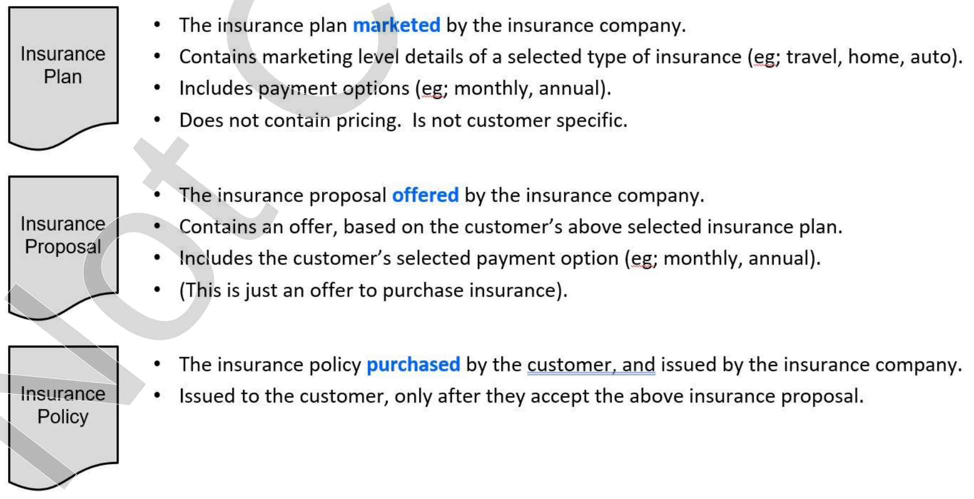
Insurance Proposal (API)
- Creates insurance proposal for external parties, eg; banks and other resellers of insurance policies
- Inputs: insurance plan ID, insurance premium
- Outputs: insurance proposal ID

- Create Insurance Policy

Insurance Policy (API)
- Creates insurance policy for external parties, eg; banks and other resellers of insurance policies
- Inputs: insurance proposal ID, customer ID
- Outputs: insurance policy ID

Source: Authors' Concept Diagram

EXHIBIT 6: DESCRIPTION OF INSURANCE DOCUMENTS



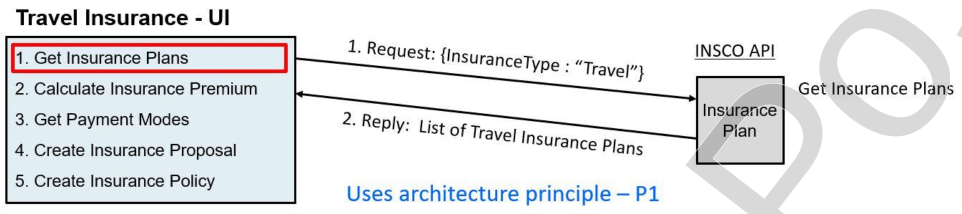
- The insurance plan **marketed** by the insurance company.
- Contains marketing level details of a selected type of insurance (eg; travel, home, auto).
- Includes payment options (eg; monthly, annual).
- Does not contain pricing. Is not customer specific.

- The insurance proposal **offered** by the insurance company.
- Contains an offer, based on the customer's above selected insurance plan.
- Includes the customer's selected payment option (eg; monthly, annual).
- (This is just an offer to purchase insurance).

- The insurance policy **purchased** by the customer, and issued by the insurance company.
- Issued to the customer, only after they accept the above insurance proposal.

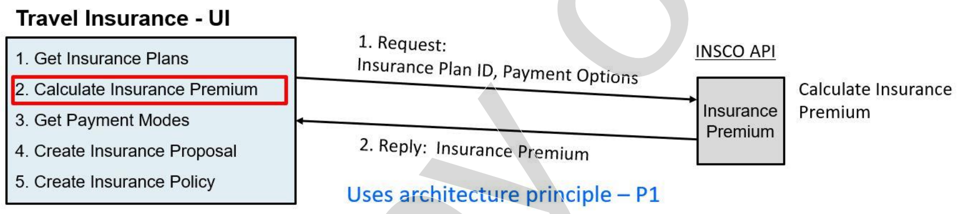
Source: Authors' Concept Diagram

EXHIBIT 7: GET INSURANCE PLANS


Travel Insurance - UI
1. Get Insurance Plans
2. Calculate Insurance Premium
3. Get Payment Modes
4. Create Insurance Proposal
5. Create Insurance Policy
1. Request: {InsuranceType : "Travel"}
2. Reply: List of Travel Insurance Plans
INSCO API
Insurance Plan
Get Insurance Plans
Uses architecture principle – P1

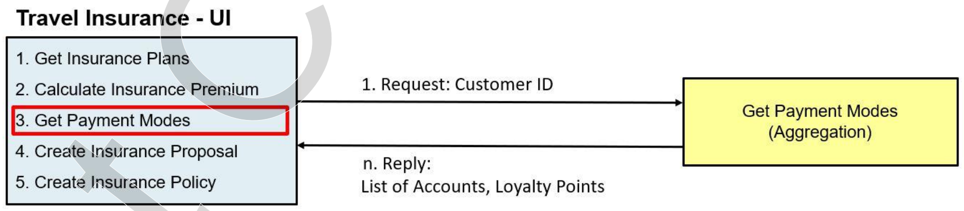
Source: Authors' Concept Diagram

EXHIBIT 8: CALCULATE INSURANCE PREMIUM


Travel Insurance - UI
1. Get Insurance Plans
2. Calculate Insurance Premium
3. Get Payment Modes
4. Create Insurance Proposal
5. Create Insurance Policy
1. Request:
Insurance Plan ID, Payment Options
2. Reply: Insurance Premium
INSCO API
Insurance Premium
Calculate Insurance Premium
Uses architecture principle – P1

Source: Authors' Concept Diagram

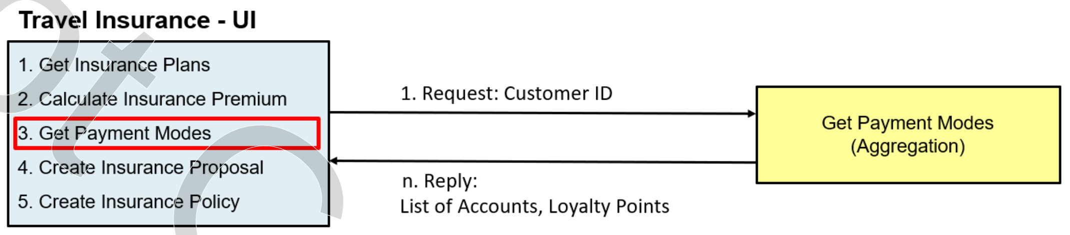
EXHIBIT 9: GET PAYMENT MODES



Source: Authors' Concept Diagram

EXHIBIT 10: GET PAYMENT MODE INTERACTIONS

The white space below is used to draw the interactions between the composite microservice "Get Payment Modes" and the three atomic microservices: "Deposit Account", "Card Account", and "Loyalty Points".



Source: Authors' own