S90.09$^{Q\&As}$

SOA Design & Architecture Lab

QUESTION 1

Upon reviewing these requirements it becomes evident to you that the Orchestration compound pattern will need to be applied. However, there are additional requirements that need to be fulfilled. To build this service composition architecture, which patterns that is not associated with the Orchestration compound pattern need to also be applied? (Be sure to choose only those patterns that relate directly to the requirements described above. Patterns associated with the Orchestration compound pattern include both the required or core patterns that are part of the basic compound pattern and the optional patterns that can extend the basic compound pattern.)

A. Atomic Service Transaction

B. Compensating Service Transaction

C. Data Format Transformation

D. Data Model Transformation

E. Event-Driven Messaging

F. Intermediate Routing

G. Policy Centralization

H. Process Centralization

I. Protocol Bridging

J. Redundant Implementation

K. Reliable Messaging

L. Service Data Replication

M. State Repository

Correct Answer: CL

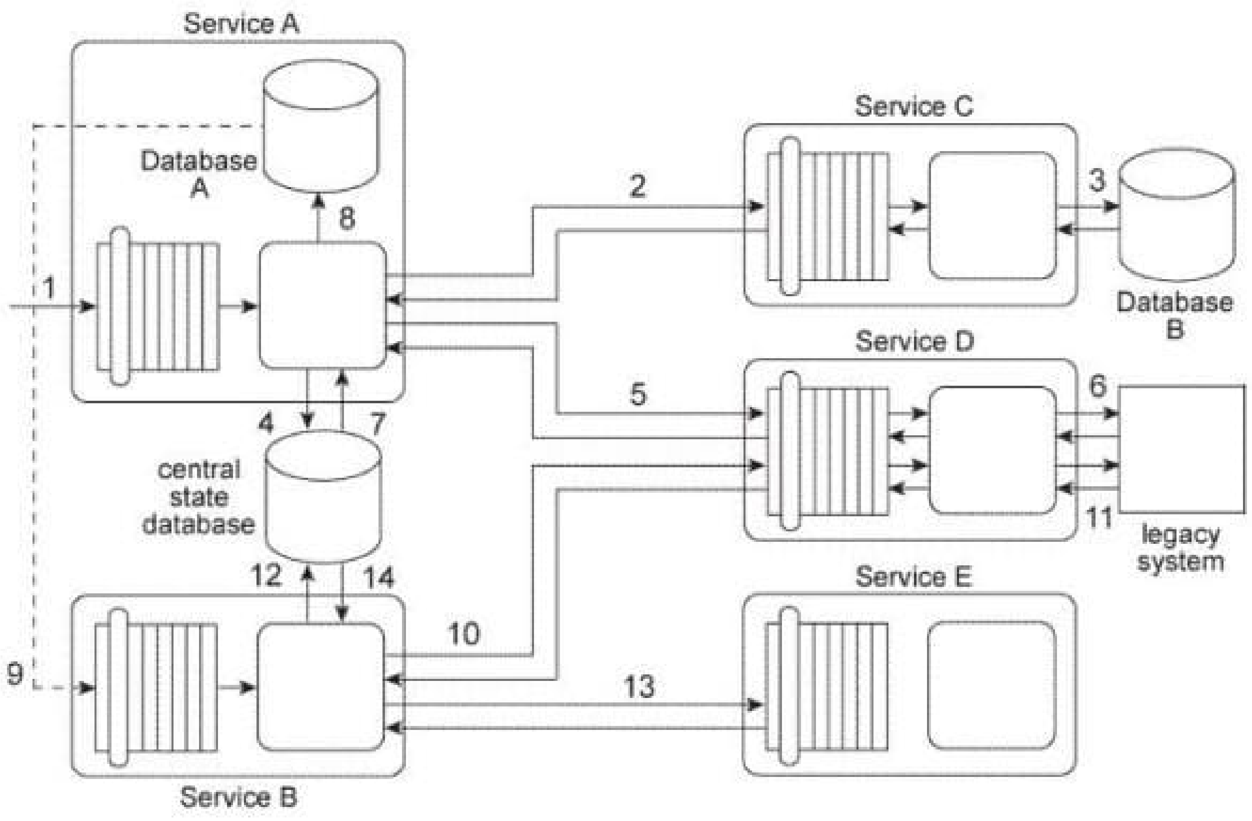
QUESTION 2

Service A is an orchestrated task service that is invoked by a separate composition initiator (1) and then sends a request message to Service C (2). Service C queries Database B to retrieve a large data record

(3) and provides this data in a response message that is sent back to Service A. Service A temporarily stores this data in a central state database (4) and then sends a request message to Service D (5), which accesses a legacy system API to retrieve a data value (6). Service D then sends this data value in a response message back to Service A. The data in the state database is subsequently retrieved by Service A (7) and merged with the newly received data value. This combined data is written to Database A (8), which triggers an event that results in the invocation of Service B (9).

Service B is an orchestrated task service that sends a request message to Service D (10). which accesses a legacy system API to retrieve a data value (11) and then sends this data value in a response message back to Service B. Service B temporarily stores this data in a central state database (12) and then sends a request message to Service E (13), which performs a runtime calculation and then responds with the calculated data value back to Service B. The data in the state database is then retrieved by Service B (14) and merged with the calculated data value. Service B then uses the merged data to complete its business task.

The following specific problems and requirements exist:



Upon reviewing these requirements it becomes evident to you that the Enterprise Service Bus compound

pattern will need to be applied. However, there are additional requirements that need to be fulfilled. To build this service composition architecture, which patterns that is not associated with the Enterprise Service Bus compound pattern need to also be applied? (Be sure to choose only those patterns that relate directly to the requirements described above. Patterns associated with the Enterprise Service Bus compound pattern include both the required or core patterns that are part of the basic compound pattern and the optional patterns that can extend the basic compound pattern.)

A. Atomic Service Transaction

B. Compensating Service Transaction

C. Data Format Transformation

D. Data Model Transformation

E. Event-Driven Messaging

F. Intermediate Routing

G. Policy Centralization

H. Process Centralization

I. Protocol Bridging

J. Redundant Implementation

K. Reliable Messaging

L. Service Data Replication

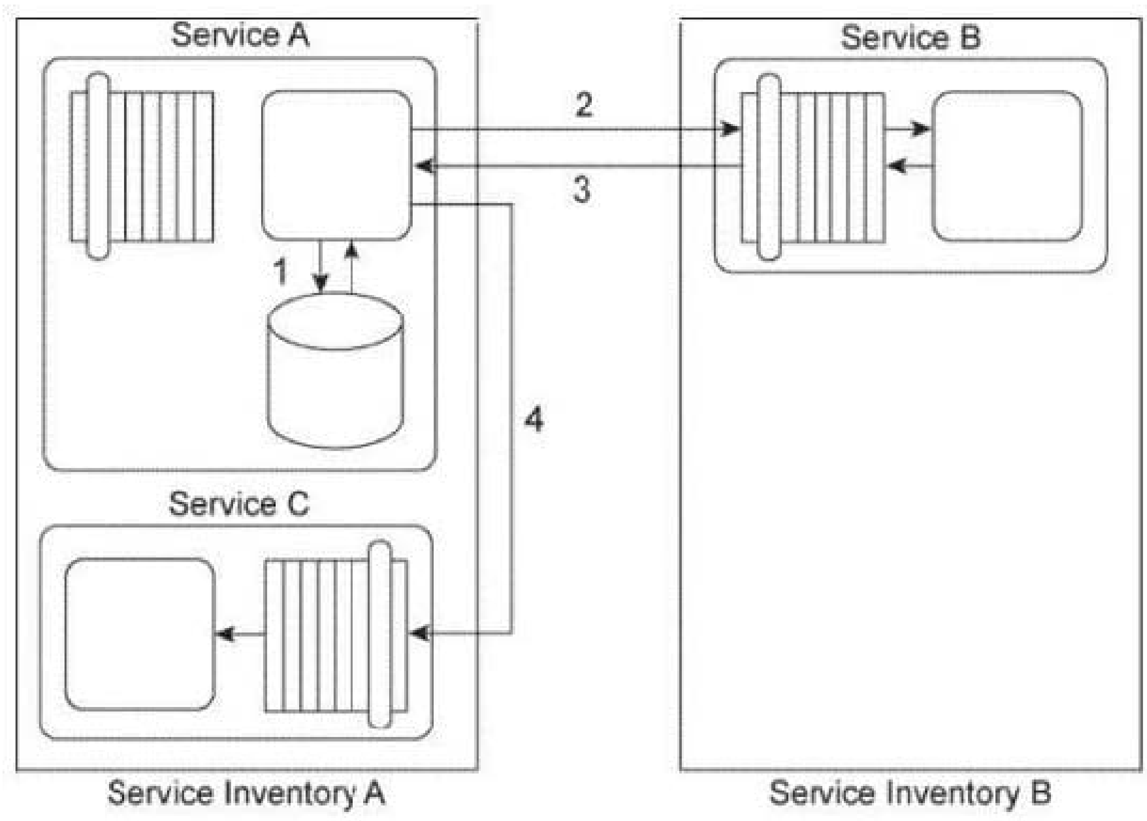
M. State Repository

Correct Answer: HLM

QUESTION 3

Service A is a task service that sends Service B a message (2) requesting that Service B return data back to Service A in a response message (3). Depending on the response received. Service A may be required to send a message to Service C (4) for which it requires no response. Before it contacts Service B, Service A must first retrieve a list of code values from its own database (1) and then place this data into its own memory. If it turns out that it must send a message to Service C, then Service A must combine the data it receives from Service B with the data from the code value list in order to create the message it sends to Service C. If Service A is not required to invoke Service C, it can complete its task by discarding the code values.

Service A and Service C reside in Service Inventory A. Service B resides in Service Inventory B.



You are told that the services in Service Inventory A are all SOAP-based Web services designed to exchange SOAP 1.1 messages and the services in Service Inventory B are SOAP-based Web services designed to exchange SOAP 1.2 messages. Therefore, Service A and Service B cannot currently communicate. Furthermore, you are told that Service B needs to access a shared database in order to retrieve the data required by Service A. The response time of the database can sometimes be lengthy, which would cause Service A to consume too much resources while it is waiting and keeping the code values in memory. How can this service composition architecture be changed to avoid these problems?

A. The Protocol Bridging pattern can be applied by establishing an intermediate processing layer between Service A and Service B that can convert SOAP 1.1 messages to SOAP 1.2 messages and vice versa. The Service Data Replication pattern can be applied to Service B so that it is given a dedicated database with its own copy of the data it needs to access. The Service Normalization pattern can then be applied to ensure that the data within the replicated database is normalized with the shared database it is receiving replicated data from.

B. The Protocol Bridging pattern can be applied by establishing an intermediate processing layer between Service A and Service B that can convert SOAP 1.1 messages to SOAP 1.2 messages and vice versa. The Service Statelessness principle can be applied with the help of the State Repository pattern so that Service A can write the code value data to a state database while it is waiting for Service B to respond.

C. The Protocol Bridging pattern can be applied by establishing an intermediate processing layer between Service A and Service B that can convert SOAP 1.1 messages to SOAP 1.2 messages and vice versa. The Intermediate Routing pattern can be applied to dynamically determine whether Service A should send a message to Service C. The Service Autonomy principle can be applied to Service A to further increase its behavioral predictability by reducing the amount of memory it is required to consume.

D. None of the above.

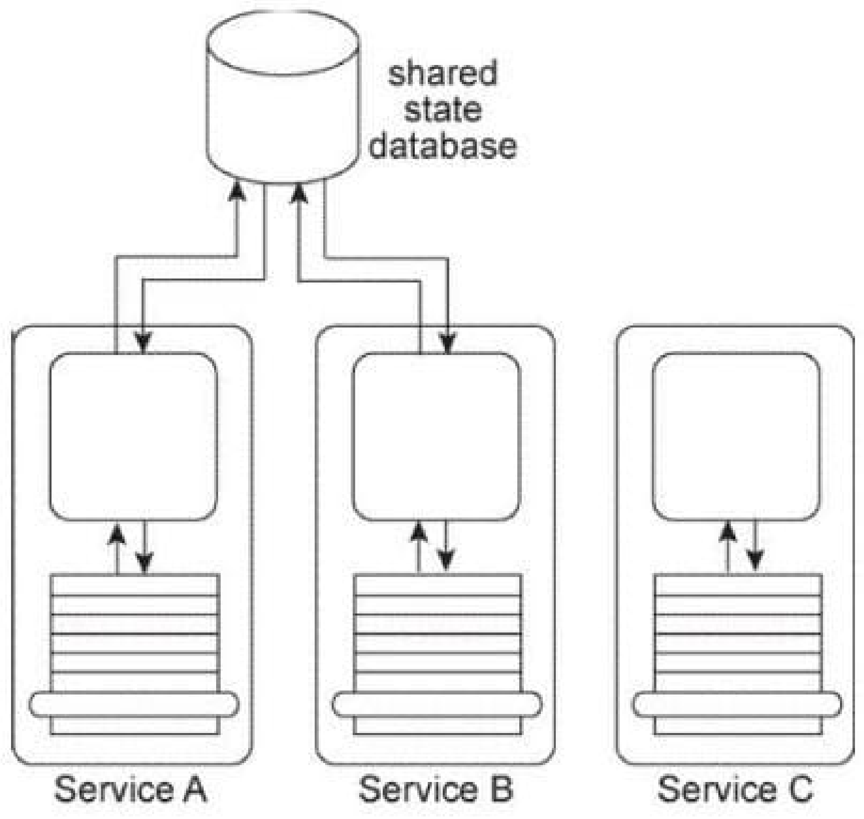
Correct Answer: B

QUESTION 4

Services A, B, and C are non-agnostic task services. Service A and Service B use the same shared state database to defer their state data at runtime.

An assessment of these three services reveals that each contains some agnostic logic, but because it is bundled together with the non-agnostic logic, the agnostic logic cannot be made available for reuse.

The assessment also determines that because Service A and Service B and the shared state database are each located in physically separate environments, the remote communication required for Service A and Service B to interact with the shared state database is causing an unreasonable decrease in runtime performance.



You are asked to redesign this architecture in order to increase the opportunity for agnostic service logic to be reused and in order to decrease the runtime processing demands so that performance can be improved. What steps can be taken to achieve these goals?

A. The Enterprise Service Bus pattern can be applied to establish an environment whereby the Process Abstraction and Process Centralization patterns are naturally applied, resulting in a clean separation of non-agnostic task services from newly designed agnostic services that are further shaped into reusable services by the application of the Service Reusability principle.

B. The Process Centralization pattern can be applied, resulting in a redesign effort where agnostic logic is removed from the three task services so that they only encapsulate non- agnostic logic. The agnostic logic is then moved to one or more new agnostic services that are shaped into reusable services by the application of the Service Reusability principle. The Process Abstraction pattern is then applied to the redesigned task services Service A and Service B, so that their logic is physically centralized, turning them into orchestrated task services.

C. The Process Abstraction pattern can be applied, resulting in a redesign effort where agnostic logic is removed from the three task services so that they only encapsulate non- agnostic logic. The agnostic logic is then moved to one or more new agnostic services that are shaped into reusable services by the application of the Service Reusability principle. The Orchestration pattern can be further applied to establish an environment whereby the Process Centralization pattern is naturally applied to Services A and B and the State Repository pattern in naturally applied to further help avoid remote communication by providing a local and centralized state database that can be shared by both services.

D. None of the above.

Correct Answer: C

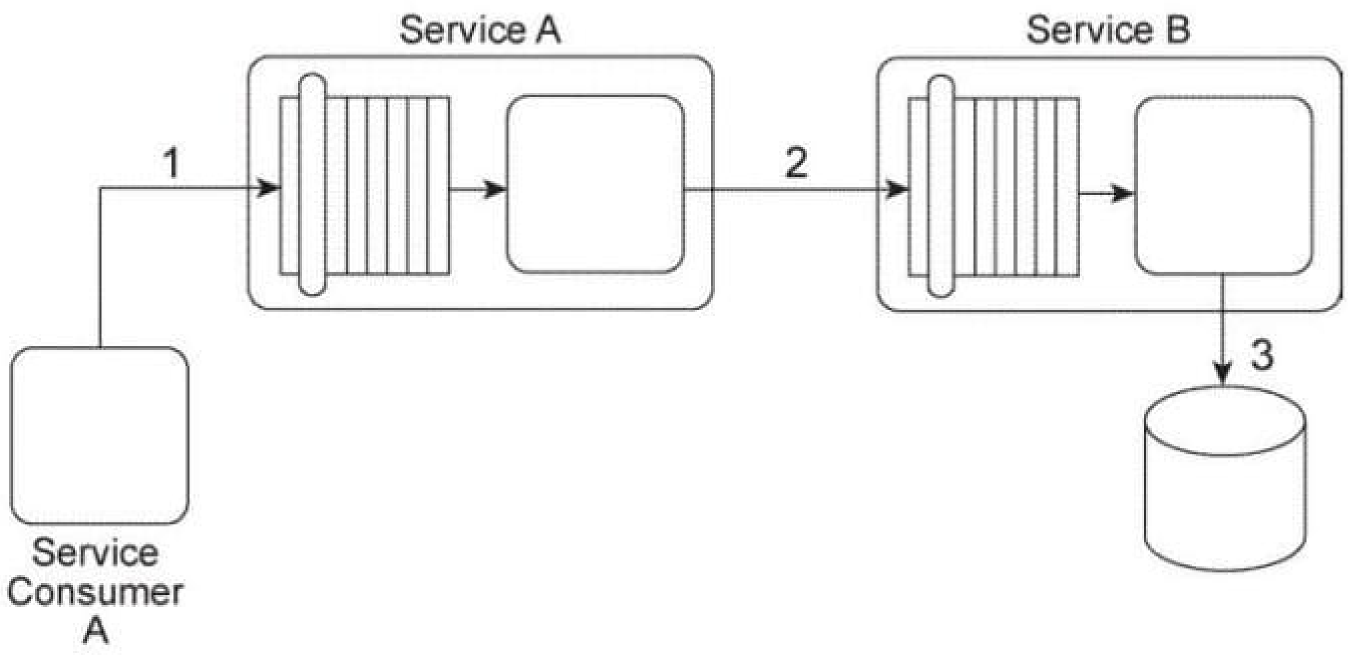
QUESTION 5

Service A is an entity service with a functional context dedicated to invoice-related processing. Service B is a utility service that provides generic data access to a database.

In this service composition architecture, Service Consumer A sends a SOAP message containing an invoice XML document to Service A(1). Service A then sends the invoice XML document to Service B (2), which then writes the invoice document to a database.

The data model used by Service Consumer A to represent the invoice document is based on XML Schema

A. The service contract of Service A is designed to accept invoice documents based on XML Schema B. The service contract for Service B is designed to accept invoice documents based on XML Schema A. The database to which Service B needs to write the invoice record only accepts entire business documents in Comma Separated Value (CSV) format.



Due to the incompatibility of XML schemas used by the services, the sending of the invoice document from Service Consumer A through to Service B cannot be accomplished using the services as they currently exist. Assuming that the Contract Centralization and Logic Centralization patterns are being applied, what steps can be taken to enable the sending of the invoice document from Service Consumer A to the database without adding logic that will increase the runtime performance of the service composition?

A. The Data Model Transformation pattern can be applied so that the invoice document sent by Service Consumer A is transformed into an invoice document that is compliant with the XML Schema B used by Service A . The Data Model Transformation pattern can be applied again to ensure that the invoice document sent by Service A is compliant with XML Schema A used by Service B.

B. The service composition can be redesigned so that Service Consumer A sends the invoice document directly to Service B. Because Service Consumer A and Service B use XML Schema A, the need for transformation logic is avoided. This naturally applies the Service Loose Coupling principle because Service Consumer A is not required to send the invoice document in a format that is compliant with the database used by Service B.

C. The Standardized Service Contract principle can be applied to the service contract of Service A so that it is redesigned to use XML Schema A. This would make it capable of receiving the invoice document from Service Consumer A and sending the invoice document to Service B without the need to further apply the Data Model Transformation pattern.

D. None of the above.

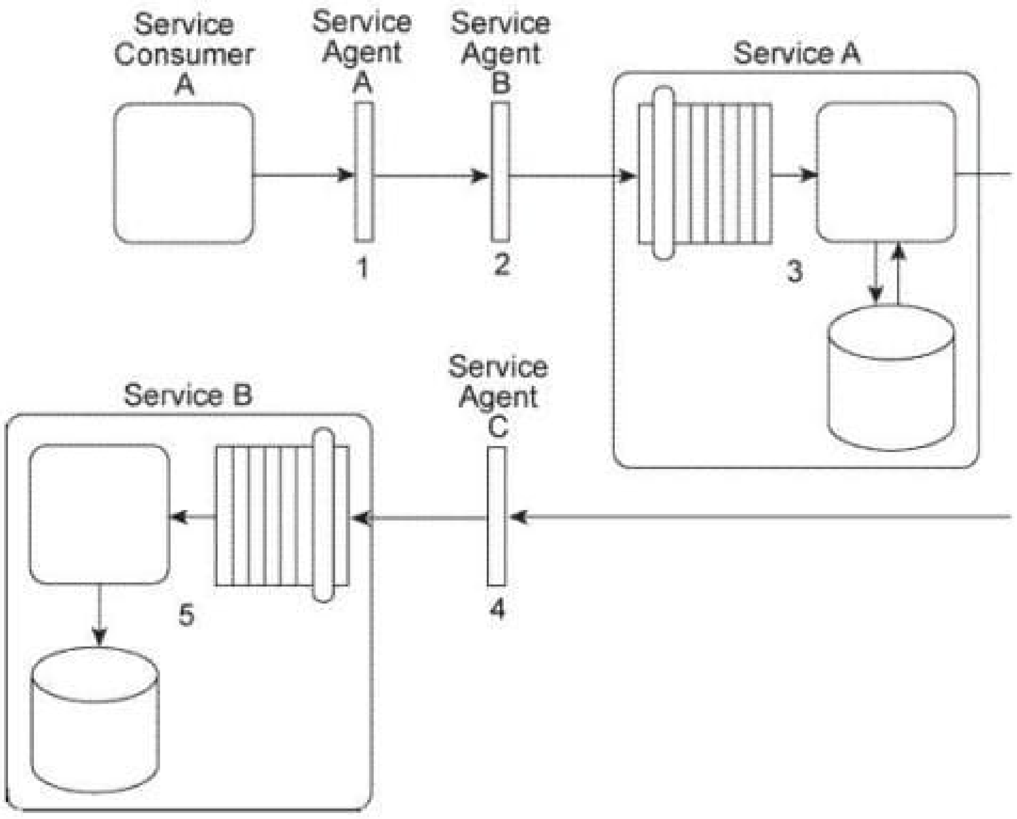
Correct Answer: C

QUESTION 6

Service Consumer A sends a message to Service A. Before the message arrives with Service A, it is intercepted by Service Agent A (1). which checks the message for compliance to Policy A that is required by Service A. If the message fails compliance, Service Agent A will not allow it to proceed and will instead write the message contents to a log. If the message does comply to the policy, it continues to be transmitted toward Service A, but before it arrives it is intercepted by Service Agent B (2), which validates the security credentials in the message header. If the security credential validation fails, the message is rejected and a runtime exception is raised. If the security credentials are validated, the message is sent to Service A.

Upon receiving the message, Service A retrieves a data value from a database and populates the message header with this data value (3) prior to forwarding the message to Service B. Before the message arrives at Service B. it is intercepted by Service Agent C (4) which checks the message for compliance with two policies: Policy B and Policy C. Policy B is identical to Policy A that was checked by Service Agent

A. To check for compliance to Policy C. Service Agent C uses the data value added by Service A. If the message complies with both of the policies, it is forwarded to Service B (5), which stores the message contents in its own database.



You are told that Policy B and Policy C have changed. Also, in order to carry out the compliance check of Policy C, Service Agent C will now require a new data value from the Service B database. How can this service composition architecture be changed to fulfill these new requirements?

A. The Policy Centralization pattern can be applied so that only one service agent is used to enforce Policy A and Policy

B. Service A is redesigned to first query Service B for the value required by Service Agent C to check the compliance of the updated Policy C. If the compliance check is successful, the message is sent to Service B .

B. The Policy Centralization pattern can be applied so that only one service agent is used to enforce Policy A and Policy B. Service Consumer A is redesigned to first query Service B for the value required by Service Agent C. This way, Service Consumer A can include this value in the message header prior to sending the message to Service A .

C. The Policy Centralization pattern can be applied so that only one service agent is used to enforce Policy A and Policy B. The policy enforcement logic for Policy C is removed from Service Agent C and instead embedded within the logic of Service B . This way, Service B can itself retrieve the value required to check compliance with Policy C. If the message received is not in compliance, Service B will reject it.

D. None of the above.

Correct Answer: D

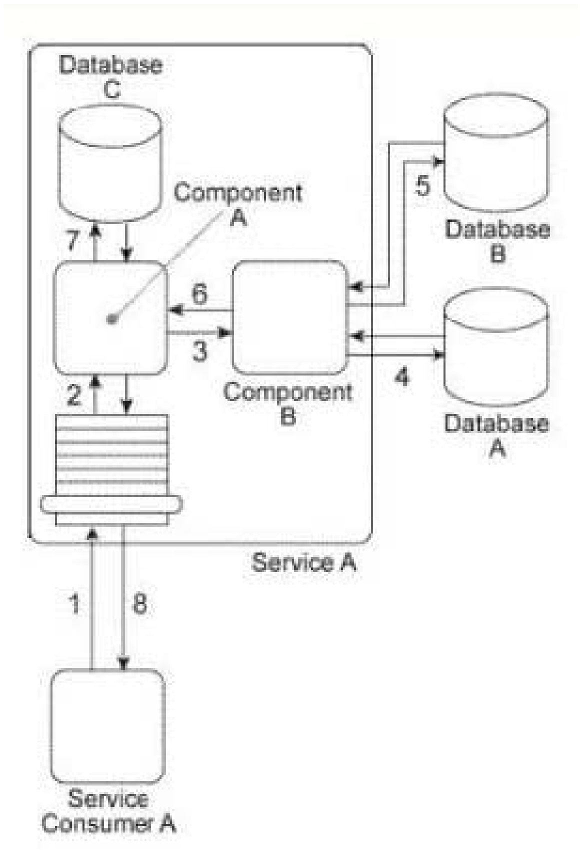
QUESTION 7

Service Consumer A sends Service A a message containing a business document (1). The business document is received by Component A, which keeps the business document in memory and forwards a copy to Component B (3). Component B first writes portions of the business document to Database A (4).

Component B writes the entire business document to Database B and then uses some of the data values from the business document as query parameters to retrieve new data from Database B (5).

Next, Component B returns the new data back to Component A (6), which merges it together with the original business document it has been keeping in memory and then writes the combined data to Database C (7). The Service A service capability invoked by Service Consumer A requires a synchronous request-response data exchange. Therefore, based on the outcome of the last database update, Service A returns a message with a success or failure code back to Service Consumer A (8).

Databases A and B are shared and Database C is dedicated to the Service A service architecture.



There are several problems with this architecture: The business document that Component A is required to keep in memory (while it waits for Component B to complete its processing) can be very large. Especially when Service A is concurrently invoked by multiple service consumers, the amount of runtime resources it uses to keep this data in memory can decrease the overall performance of all service instances. Additionally, because Database A is a shared database that sometimes takes a long time to respond to Component B, Service A can take a long time to respond back to Service Consumer A . Currently, Service Consumer A will wait for a response for up to 30 seconds after which it will assume the request to Service A has failed and any subsequent response messages from Service A will be rejected. What steps can be taken to solve these problems?

A. The Service Statelessness principle can be applied together with the State Repository pattern in order to extend Database C so that it also becomes a state database allowing Component A to temporarily defer the business document data while it waits for a response from Component B. The Service Autonomy principle is applied together with the Legacy Wrapper pattern to isolate Database A so that it is encapsulated by a separate wrapper utility service. The Compensating Service Transaction pattern is applied so that if the response time of Service A exceeds 30 seconds, a notification is sent to a human administrator to raise awareness of the fact that the eventual response of Service A will be rejected by Service Consumer A.

B. The Service Statelessness principle can be applied together with the State Repository pattern in order to establish a state database that Component A can defer the business document data to while it waits for a response from Component B. The Service Autonomy principle can be applied together with the Service Data Replication pattern to establish a dedicated replicated database for Component B to access instead of the shared Database

C. The Asynchronous Queuing pattern can be applied to establish a messaging queue between Service Consumer A and Service A so that Service Consumer A does not need to remain stateful while it waits for a response from Service A .

D. The Service Statelessness principle can be applied together with the State Repository pattern in order to establish a state database that Component A can defer the business document data to while it waits for a response from Component B. The Service Autonomy principle can be applied together with Service Abstraction principle, the Legacy

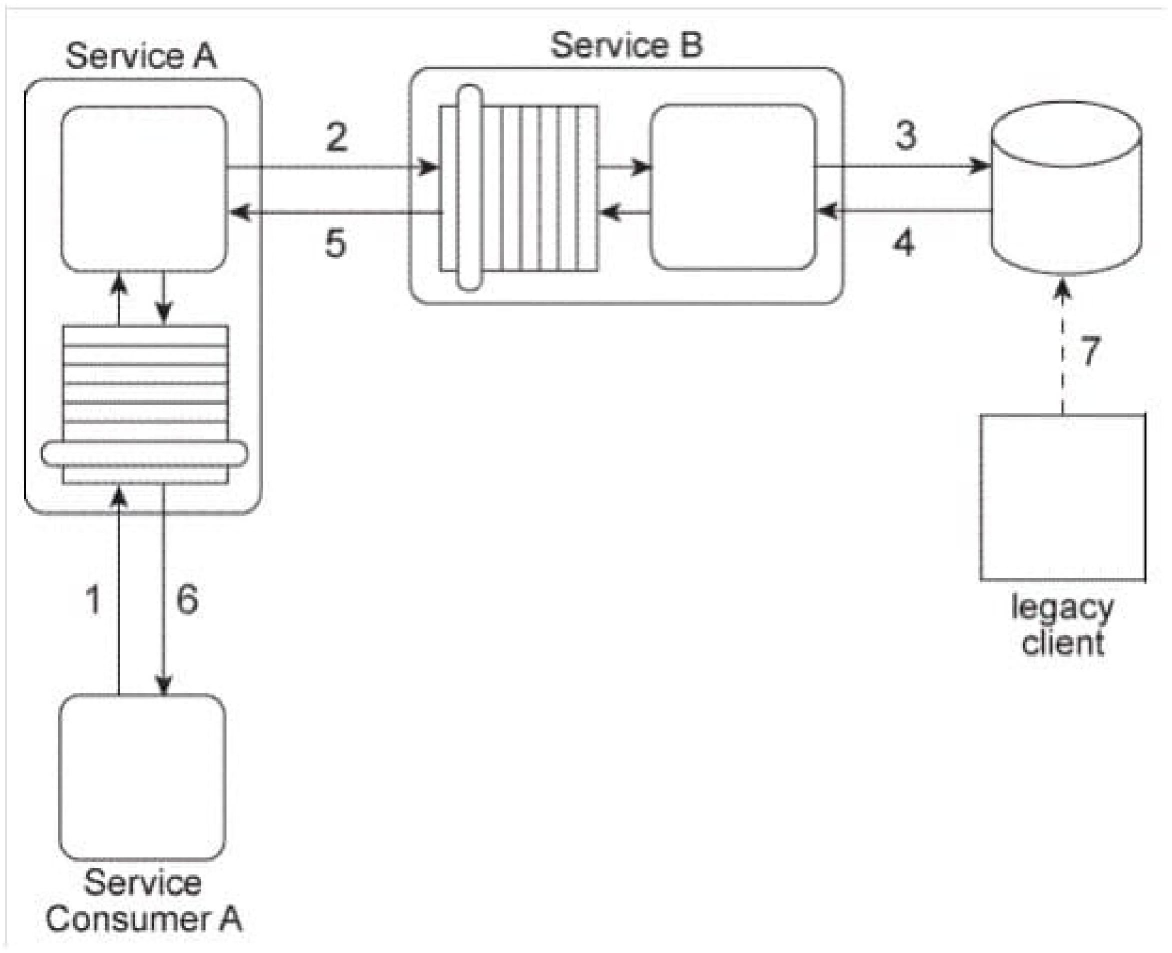
Wrapper pattern, and the Service Facade pattern in order to isolate Database A so that it is encapsulated by a separate wrapper utility service and to hide the Database A implementation from Service A and to position a Facade component between Component B and the new wrapper service. This Facade component will be responsible for compensating the unpredictable behavior of Database A.

E. None of the above.

Correct Answer: B

QUESTION 8

Service A has become increasingly difficult to maintain. Its core service logic has become bloated and convoluted because it has been updated numerous times during which additional functionality was added to interact with the database and the legacy system and to support interaction with Service Consumers A and B (via the two service contracts) as well as interaction directly with Service Consumer C.



What steps can be taken to solve these problems and to prevent them from happening again in the future?

A. The Service Facade pattern can be applied to position a Facade component between the core service logic and the implementation resources (the database and the legacy system) and to also position a Facade component between the two service contracts and Service Consumers A and

B. The Official Endpoint pattern can be applied to limit access to Service A to one of its two published service contracts. The Service Loose Coupling principle can be applied so that Service Consumer C does not negatively couple itself directly to the core service logic of Service A . B. The Service Facade pattern can be applied to position a Facade component between the core service logic and the implementation resources (the database and the legacy system) and to position a faade component between the core service logic and the two service contracts. The Contract Centralization pattern can be applied to limit access to Service A to one of its two published service contracts. The Service Abstraction principle can be applied to hide the implementation details of Service A from service consumers.

C. The Service Faade pattern can be applied to position a Facade component between the core service logic and the two service contracts. The Contract Centralization pattern can be applied to limit access to Service A to one of its two published service contracts. The Service Loose Coupling principle can be applied so that Service Consumer C does not negatively couple itself directly to the core service logic of Service A .

D. None of the above.

Correct Answer: B

QUESTION 9

You are an architect with a project team building services for Service Inventory A . You are told that no SLAs for Service B and Service C are available. You cannot determine how available these services will be, but it has been confirmed that both of these services support atomic transactions and the issuance of positive and negative acknowledgements. However, you also find out that the services in Service Inventory B use different data models than the services in Service Inventory A. Furthermore, recent testing results have shown that the performance of Service D is steady and reliable. However, Service D uses a different transport protocol than the services in Service Inventory A. The response time of Service A is not a primary concern, but Service Consumer A does need to be able to issue request messages to Service A 24 hours a day without disruption. What steps can be taken to fulfill these requirements?

A. The Event-Driven Messaging pattern is applied so that a subscriber-publisher relationship is established between Service Consumer A and Service A . This gives Service A the flexibility to provide its response to Service Consumer A whenever it is able to collect the three data values without having to require that Service Consumer A remain stateful. The Asynchronous Queuing pattern is applied so that a central messaging queue is positioned between Service A and Service B and between Service A and Service C . The Data Model Transformation and Protocol Bridging patterns are applied to enable communication between Service A and Service B and between Service A and Service C . The Service Autonomy principle is further applied to Service A in order to improve its overall runtime behavioral predictability.

B. The Reliable Messaging pattern is applied so that a system of acknowledgements is established between Service Consumer A and Service A . This gives Service A the flexibility to provide Service Consumer A with acknowledgements that indicate that the processing steps that are occurring between Service A and Service B, Service C, and Service D are progressing. The Asynchronous Queuing pattern is applied so that a central messaging queue is positioned between Service A and Service B and between Service A and Service C and between Service A and Service D . The Redundant Implementation pattern is applied so that a copy of Service D is brought in-Upon reviewing these requirements it becomes D with a standardized service contract that is in compliance with the design standards used in Service Inventory A.

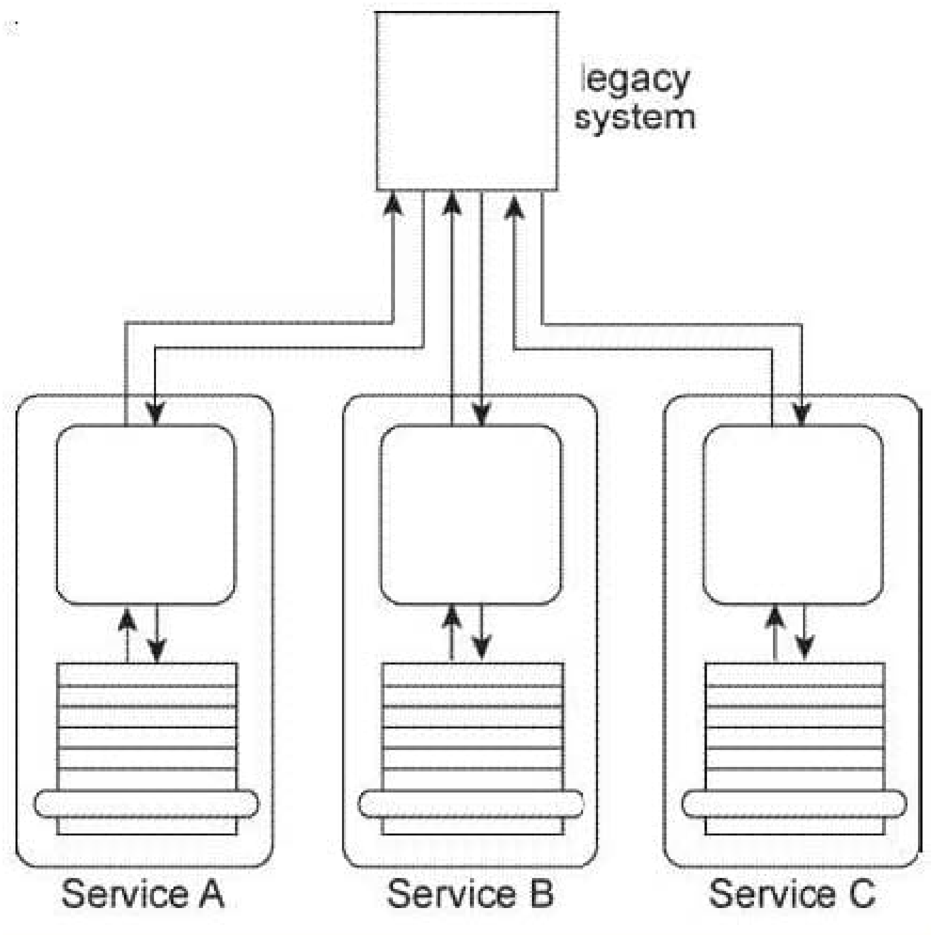
C. The Asynchronous Queuing pattern is applied so that a central messaging queue is positioned between Service A and Service B and between Service A and Service C and between Service A and Service D and so that a separate messaging queue is positioned between Service A and Service Consumer A. The Data Model Transformation pattern is applied to enable communication between Service A and Service B and between Service A and Service C . The Protocol Bridging pattern is applied to enable communication between Service A and Service D .

D. None of the above.

Correct Answer: C

QUESTION 10

Service A. Service B. and Service C are each designed to access the same shared legacy system. The service contracts for Service A, Service B, and Service C are standardized and decoupled from the underlying service logic. Service A and Service B are agnostic services that are frequently reused by different service compositions. Service C is a non- agnostic task service that requires access to the legacy system in order to retrieve business rules required for the service to make runtime decisions that determine its service composition logic. The legacy system uses a proprietary file format that Services A, B, and C need to convert to and from.



Service A is an agnostic utility service that is used by other services to gain access to the legacy system. Services B and C were not designed to access the legacy system via Service A because the Service A service contract was derived from the legacy system API and is therefore not standardized and exhibits negative contract-to-implementation coupling. You are told that additional services need to be created, all of which need access to the legacy system. You are also told that the legacy system may be replaced in the near future. What steps can be taken to ensure that the replacement of the legacy system has a minimal impact on Services B and C and any future services that are designed to rely upon it?

A. The Service Abstraction, Service Reusability, and Service Autonomy principles need to be applied in order to support the application of the Official Endpoint pattern to Service A . This would position Service A as the official utility service through which the legacy system can be accessed. Service B will need to be redesigned to access Service A instead of accessing the legacy

system directly. Due to the dependency on business rules embedded within the legacy system the

option of applying the Rules Centralization pattern is not available. Service C will therefore need to continue accessing the legacy system directly.

B. The Standardized Service Contract and Service Loose Coupling principles can be applied in order to establish a standardized service contract for Service A that will eliminate its negative contract coupling. Service B will need to be redesigned to access Service A instead of accessing the legacy system directly. Due to the dependency on business rules embedded within the legacy system the option of applying the Rules Centralization pattern is not available. Service C will therefore need to continue accessing the legacy system directly.

C. The Legacy Wrapper pattern can be applied together with the Standardized Service Contract principle in order to establish a standardized service contract for Service A that will eliminate its negative contract coupling. The Official Endpoint pattern can then be applied to position Service A as the official utility service through which the legacy system can be accessed. Services B and C will need to be redesigned to access Service A instead of accessing the legacy system directly.

D. None of the above.

Correct Answer: C