Free Questions for S90.08B by vceexamstest

Shared by Skinner on 17-03-2023

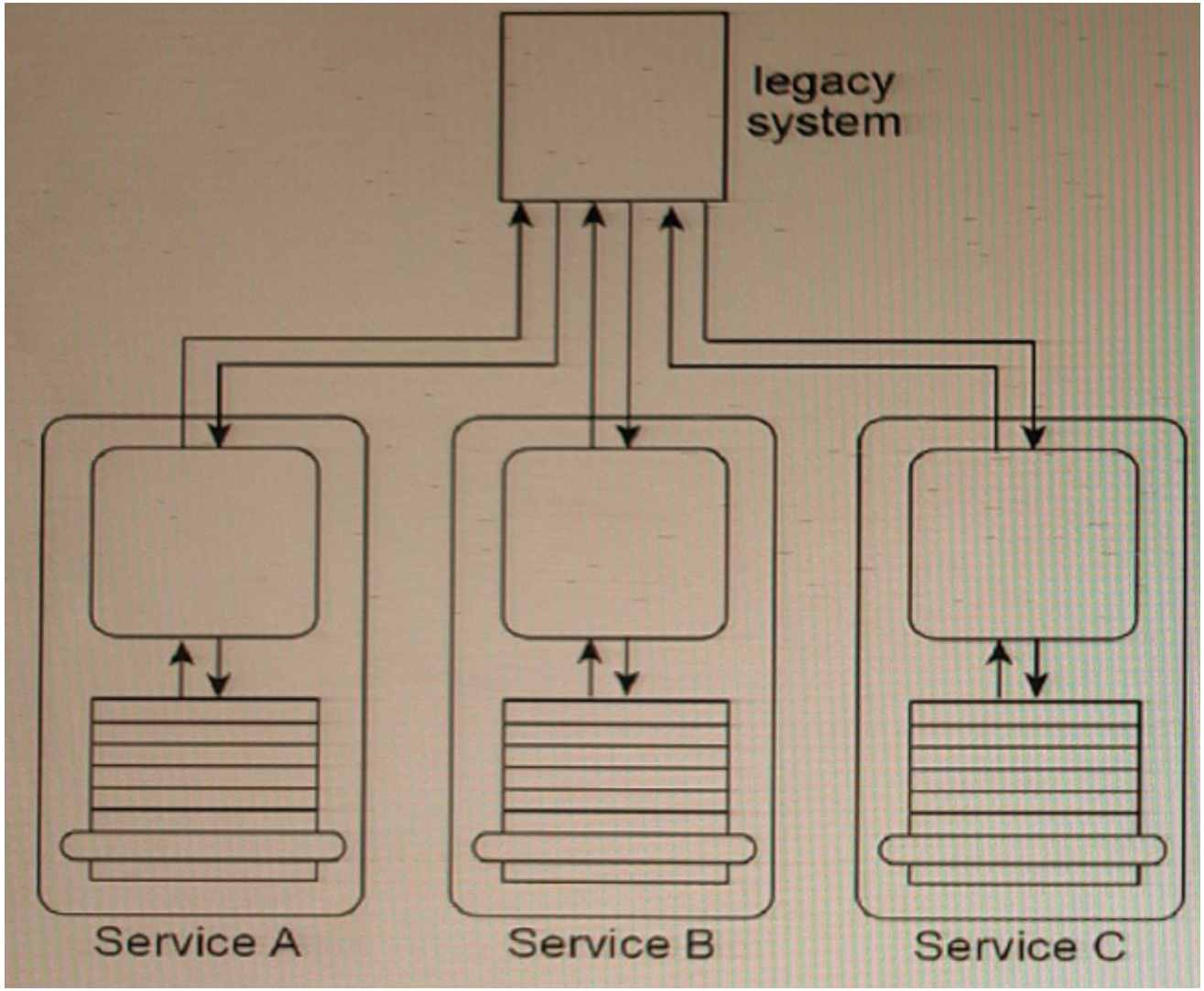
For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

Refer to Exhibit.



Service A, Service B, and Service C are entity services, each designed to access the same shared legacy system. Service A manages order entities, Service B manages invoice entities, and Service C manages customer entities. Service A, Service B, and Service C are REST services and are frequently reused by different service compositions. The legacy system uses a proprietary file format that

Services A, B, and C need to convert to and from.

You are told that compositions involving Service A, Service B, and Service C are unnecessarily complicated due to the fact that order, invoice, and customer entitles are all related to each other. For example, an order has a customer, an invoice has an order, and so on. This results In calls to multiple services to reconstruct a complete order document. You are asked to architect a solution that will simplify the composition logic by minimizing the number of services required to support simple business functions like order management or bill payment. Additionally, you are asked to reduce the amount of redundant data transformation logic that is found in Services A, B, and C.

How will you accomplish these goals?

Options:

A- The Enterprise Service Bus pattern can be applied to introduce an intermediate processing layer between Services A, B, and C and the legacy system. The enterprise service bus can be used to consolidate and execute the necessary transformation logic currently held within the services. The Endpoint Redirection pattern can be applied to re-route calls from one service to another to provide access to related entity data.

B- The Legacy Wrapper pattern can be applied to create a service to expose the legacy system through a standardized service contract. The core logic of the wrapping service would provide all necessary data transformation functionality to convert between inventory-standardized data representations and the proprietary format. The Lightweight Endpoint pattern can be applied to establish lightweight capabilities that can return related entity data directly to service consumers.

C- The Enterprise Service Bus pattern can be applied to introduce an intermediate processing layer between Services A, B, and C and the legacy system. The enterprise service bus can be used to consolidate and execute the transformation logic currently held within the services. The Content Negotiation pattern can be applied to return a content link to related entity data to a service consumer, which allows for simpler and more dynamic composition logic. The service consumer effectively invokes the relevant service through the

returned link to obtain the related entity data.

D- The Legacy Wrapper pattern can be applied to create a service to expose the legacy system through a standardized service contract. The core logic of the wrapping service would provide all necessary data transformation functionality to convert between inventory-standardized data representations and the proprietary format. The Endpoint Redirection pattern can be applied to return a link to related entity data to a service consumer, which allows for simpler and more dynamic composition logic. The service consumer effectively invokes the relevant service through the returned link to obtain the related entity data.

Answer:

B

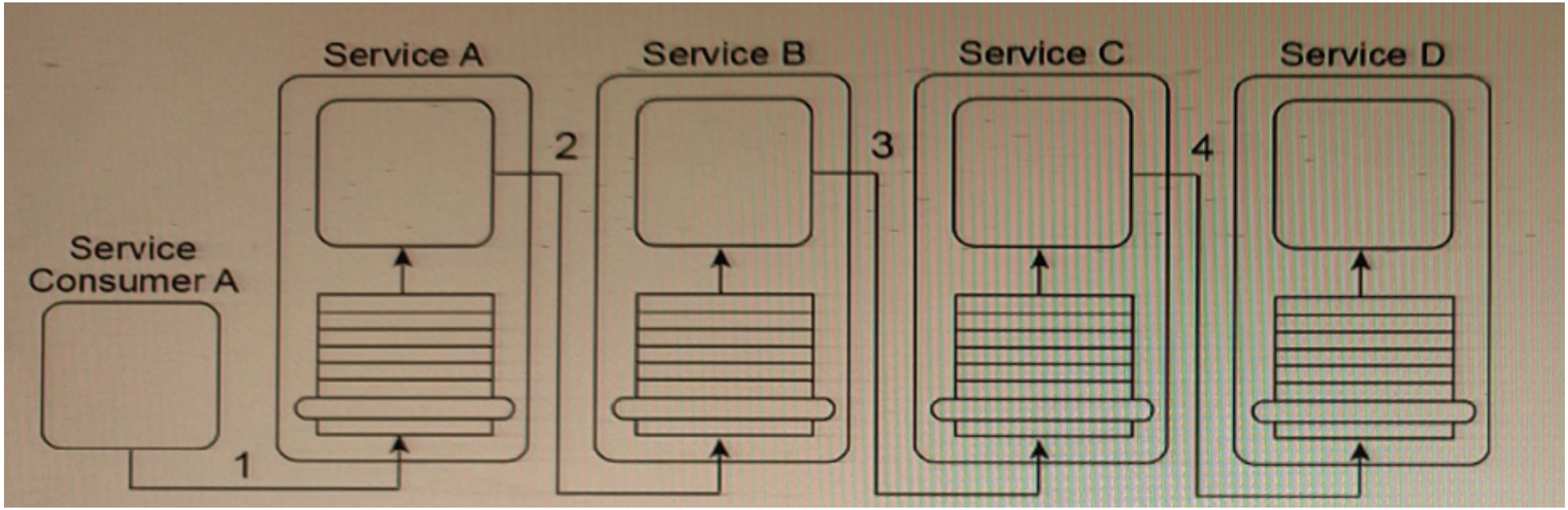
Explanation:

The Lightweight Endpoint pattern can be applied to establish lightweight capabilities that can return related entity data directly to service consumers, simplifying the composition logic by minimizing the number of services required to support simple business functions like order management or bill payment. This approach provides a standardized and simplified interface for the legacy system, reducing the complexity of the integration process with the entity services, and enabling them to focus on their core functionality.

Question 2

Question Type: MultipleChoice

Refer to Exhibit.



Service Consumer A sends a message to Service A (1), which then forwards the message to Service B (2). Service B forwards the message to Service C (3), which finally forwards the message to Service D (4). However, Services A, B and C each contain logic that reads the contents of the message to determine what intermediate processing to perform and which service to forward the message to. As a result, what is shown in the diagram is only one of several possible runtime scenarios.

Currently, this service composition architecture is performing adequately, despite the number of services that can be involved in the transmission of one message. However, you are told that new logic is being added to Service A that will require it to compose one other service to retrieve new data at runtime that Service A will need access to in order to determine where to forward the message to. The involvement of the additional service will make the service composition too large and slow.

What steps can be taken to improve the service composition architecture while still accommodating the new requirements and avoiding an increase in the amount of service composition members?

Options:

A- The Service Instance Routing pattern can be applied to introduce a Routing service to provide a centralized service to contain routing-related business rules. This new Routing service can be accessed by Service A and Service C so they can determine where to forward messages to at runtime. The Service Reusability principle can be further applied to ensure that the logic in all remaining services is designed to be multi-purpose and reusable.

B- The Asynchronous Queuing pattern can be applied together with a Routing service that is invoked by messages read from a messaging queue. This new Routing service can replace Service B and can be accessed by Service A and Service C so they can determine where to forward messages to at runtime. The Service Loose Coupling principle can be further applied to ensure that the new Routing service remains decoupled from other services so that it can perform its routing functions independently from service contract invocation.

C- The Intermediate Routing pattern can be applied together with the Service Agent pattern by removing Service B or Service C from the service composition and replacing it with a service agent capable of intercepting and forwarding the message at runtime based on pre-defined routing logic. The Service Discoverability principle can be further applied to ensure that Service A can be found by any future service consumers.

D- The Intermediate Routing pattern can be applied together with the Service Agent pattern to establish a service agent capable of intercepting and forwarding the message at runtime based on pre-defined routing logic. The Service Composability principle can be further applied to ensure that all services are designed as effective service composition participants.

Answer:

B

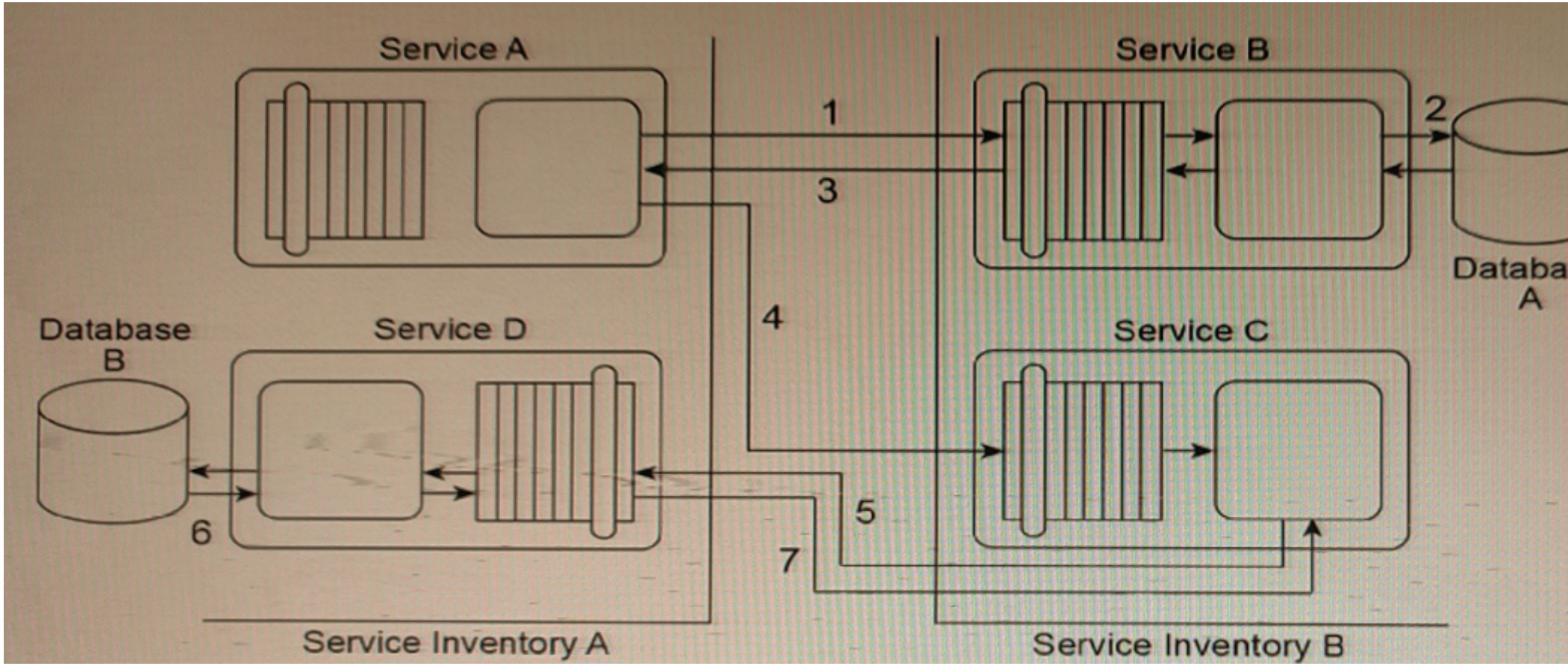
Explanation:

This solution addresses the issue of the service composition becoming too large and slow by introducing a new Routing service that is invoked by messages read from a messaging queue. This allows Service A and Service C to determine where to forward messages to at runtime without the need for additional services in the composition. The Service Loose Coupling principle is applied to ensure that the new Routing service remains decoupled from other services so that it can perform its routing functions independently from service contract invocation.

Question 3

Question Type: MultipleChoice

Refer to Exhibit.



Service A sends a message to Service B (1). After Service B writes the message contents to Database A (2), it issues a response message back to Service A (3). Service A then sends a message to Service C (4). Upon receiving this message, Service C sends a message to Service D (5), which then writes the message contents to Database B (6) and issues a response message back to Service C (7).

Service A and Service D are located in Service Inventory

Options:

A- Service B and Service C are located in Service Inventory B.

You are told that In this service composition architecture, all four services are exchanging invoice-related data in an XML format. However, the services in Service Inventory A are standardized to use a different XML schema for invoice data than the services in Service Inventory B. Also, Database A can only accept data in the Comma Separated Value (CSV) format and therefore cannot accept XML-formatted data. Database B only accepts XML-formatted data. However, it is a legacy database that uses a proprietary XML schema to represent invoice data that is different from the XML schema used by services in Service Inventory A or Service Inventory B. What steps can be taken to enable the planned data exchange between these four services?

A- The Data Model Transformation pattern can be applied so that data model transformation logic is positioned between Service A and Service B, between Service C and Service D, and between the Service D logic and Database B. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between Service A and Service C, and between the Service B logic and Database A.

B- The Protocol Bridging pattern can be applied so that protocol conversion logic is positioned between the Service B logic and Database A. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between Service A and Service B, between Service A and Service C, between Service C and Service D, and between the Service D logic and Database B.

C- The Data Model Transformation pattern can be applied so that data model transformation logic is positioned between Service A and Service B, between Service A and Service C, between Service C and Service D, and between the Service D logic and Database B. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between the Service B logic and Database A.

D- The Protocol Bridging pattern can be applied so that protocol conversion logic is positioned between Service A and Service B, between Service A and Service C, and between Service C and Service D. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between the Service B logic and Database A and between the Service D logic and

Database B.

Answer:

C

Explanation:

This solution addresses the two main challenges in the service composition architecture: the different XML schema used by services in Service Inventory A and Service Inventory B, and the incompatible data formats of the two databases.

By applying the Data Model Transformation pattern, data model transformation logic can be inserted to map the invoice-related data between the different XML schemas used by the services in Service Inventory A and Service Inventory B. This can be done at the appropriate points in the message flow: between Service A and Service B, between Service A and Service C, between Service C and Service D, and between the Service D logic and Database B.

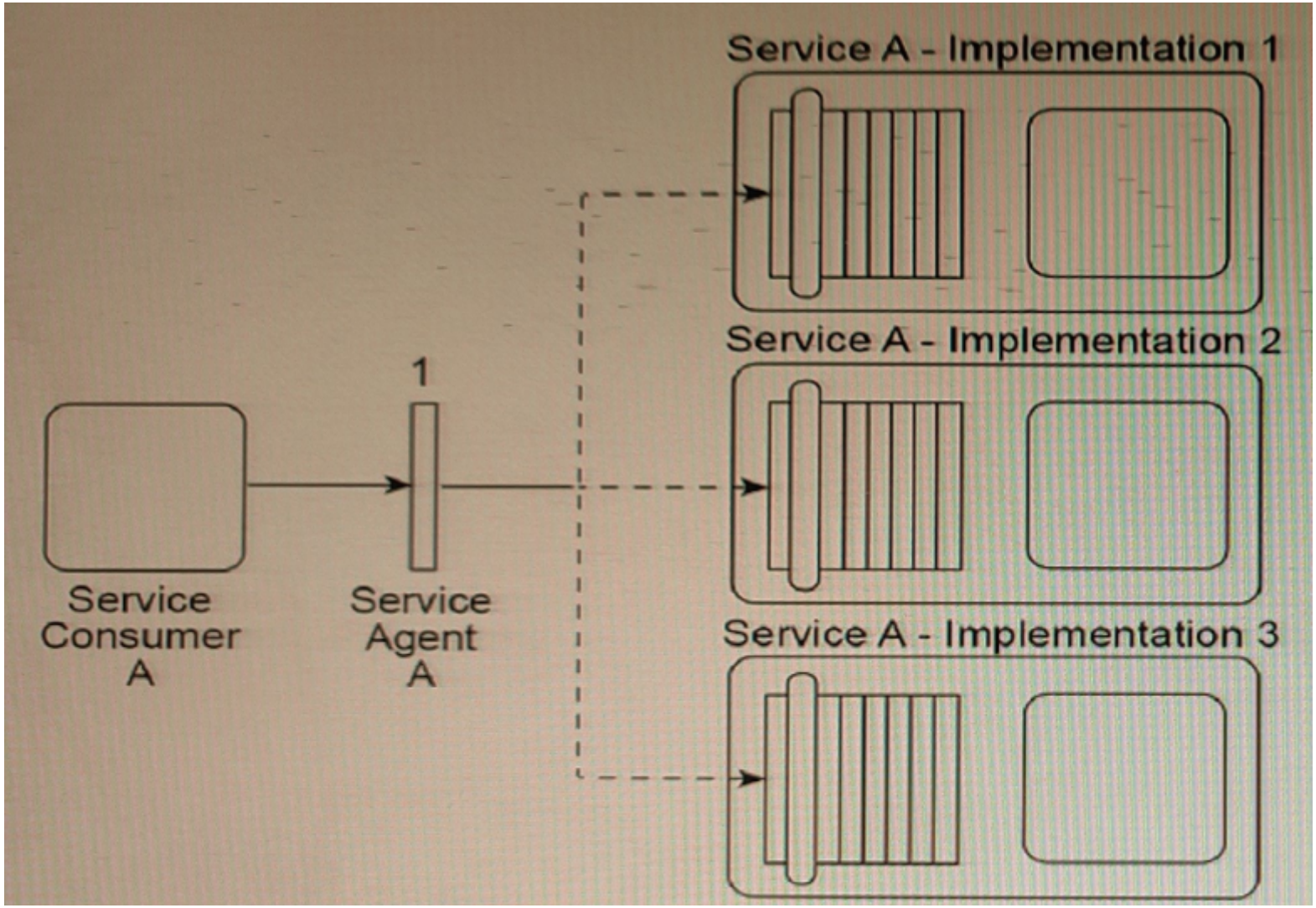
By applying the Data Format Transformation pattern, data format transformation logic can be inserted to convert the XML-formatted data used by the services to the CSV format required by Database A, and to convert the proprietary XML schema used by Database B to the XML schema used by the services. This can be done between the Service B logic and Database A.

The Protocol Bridging pattern is not necessary in this case because all services are already communicating using the same protocol (presumably HTTP or a similar protocol).

Question 4

Question Type: MultipleChoice

Refer to Exhibit.



Service Consumer A sends a message to Service

Options:

A- There are currently three duplicate implementations of Service A (Implementation 1, Implementation 2 and Implementation 3). The message sent by Service Consumer A is intercepted by Service Agent A (1), which determines at runtime which implementation of Service A to forward the message to. All three implementations of Service A reside on the same physical server.
You are told that despite the fact that duplicate implementations of Service A exist, performance is still poor at times. You are also informed that a new service capability will soon need to be added to Service A to introduce functionality that will require access to a shared database being used by many other clients and applications in the IT enterprise. This is expected to add further performance demands on Service A.
How can this service architecture be changed to improve performance in preparation for the addition of the new service capability?

A- The Standardized Service Contract principle can be applied to ensure that the new service capability extends the existing service contract in a manner that is compliant with current design standards. The Redundant Implementation pattern can be applied to establish separate implementations of Service A that include duplicate databases with copies of the data that Service A requires from the shared database.

B- The Service Autonomy principle can be applied to further isolate the individual implementations of Service A by separating them onto different physical servers. When the new service capability is added, the Service Data Replication pattern can be applied to give each implementation of Service A its own copy of the data it requires from the shared database.

C- The Service Loose Coupling principle can be applied together with the Standardized Service Contract principle to ensure that Service Consumer A is not indirectly coupled to the shared database after the new service capability is added to the service contract. The Legacy Wrapper pattern can be applied to establish a new utility service that will provide standardized data access service capabilities for the shared database.

D- The Service Autonomy principle can be applied to further isolate the individual implementations of Service A by separating them onto different physical servers. When the new service capability is added, the State Repository pattern can be applied to give each implementation of Service A its own copy of the data it requires from the shared database.

Answer:

B

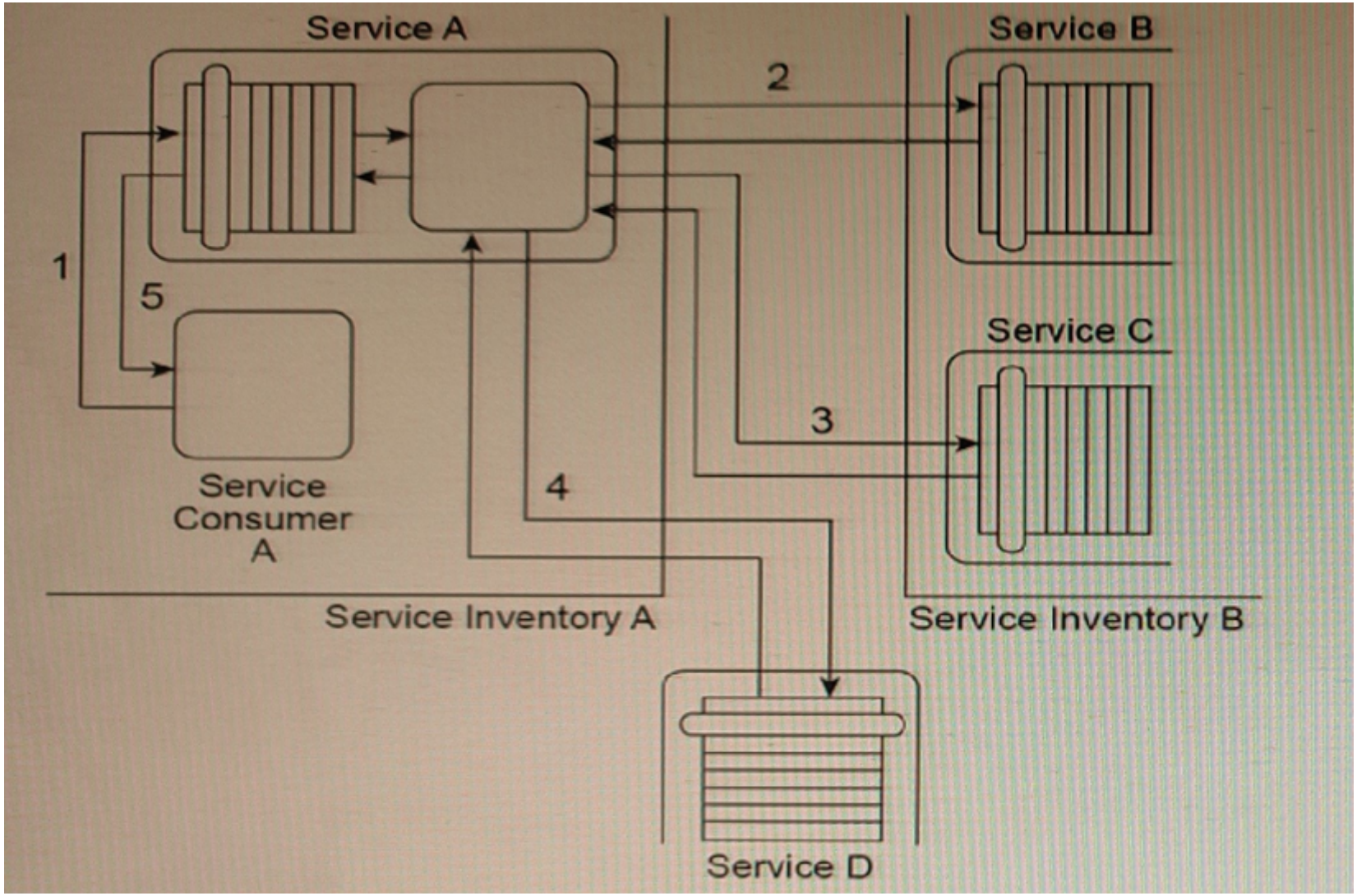
Explanation:

By separating the individual implementations of Service A onto different physical servers, they can be isolated from each other and from other clients and applications in the IT enterprise, which can help improve performance. Additionally, using the Service Data Replication pattern to give each implementation of Service A its own copy of the data it requires from the shared database can help reduce the load on the shared database and improve performance. This can be especially important when a new service capability is added that requires access to the shared database, as it can help ensure that the performance of Service A is not impacted by the additional demands placed on the shared database.

Question 5

Question Type: MultipleChoice

Refer to Exhibit.



Service Consumer A and Service A reside in Service Inventory

Options:

A- Service B and Service C reside in Service Inventory B. Service D is a public service that can be openly accessed via the World Wide Web. The service is also available for purchase so that it can be deployed independently within IT enterprises. Due to the rigorous application of the Service Abstraction principle within Service Inventory B, the only information that is made available about Service B and Service C are the published service contracts. For Service D, the service contract plus a service level agreement (SLA) are made available. The SLA indicates that Service D has a planned outage every night from 11:00pm to midnight.
You are an architect with a project team that is building services for Service Inventory A. You are told that the owners of Service Inventory A and Service Inventory B are not generally cooperative or communicative. Cross-inventory service composition is tolerated, but not directly supported. As a result, no SLAs for Service B and Service C are available and you have no knowledge about how available these services are. Based on the service contracts you can determine that the services in Service Inventory B use different data models and a different transport protocol than the services in Service Inventory A. Furthermore, recent testing results have shown that the performance of Service D is highly unpredictable due to the heavy amount of concurrent access it receives from service consumers from other organizations. You are also told that there is a concern over how long Service Consumer A will need to remain stateful while waiting for a response from Service A.
What steps can be taken to solve these problems?

A- The Event-Driven Messaging pattern can be applied to establish a subscriber-publisher relationship 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 can be applied to position a central messaging queue between Service A and Service B and between Service A and Service C. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B and between Service A and Service C. The Redundant Implementation pattern can be applied so that a copy of Service D is brought in-house and made part of Service Inventory A.

B- The Asynchronous Queuing pattern can be applied to position a central messaging queue between Service A and Service B and between Service A and Service C and so that a separate messaging queue is positioned between Service A and Service Consumer A. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B and between Service A and Service C. The Redundant Implementation pattern can be applied so that a copy of Service D is brought in-house. The Legacy Wrapper pattern can be further applied to wrap Service D with a standardized service contract that is in compliance with the design standards used in Service Inventory A.

C- The Containerization pattern can be applied to establish an environment for Service A to perform its processing autonomously. This gives Service A the flexibility to provide Service Consumer A with response messages consistently. The Asynchronous Queuing pattern can be applied so that a central messaging queue is positioned between Service A and Service B, between Service A and Service C, and between Service A and Service D. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B and between Service A and Service C.

D- The Asynchronous Queuing pattern can be applied to position a message queue between Service A and Service B, between Service A and Service C, and between Service A and Service D. Additionally, a separate messaging queue is positioned between Service A and Service Consumer A. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B, between Service A and Service C, and between Service A and Service D. The Redundant Implementation pattern can be applied so that a copy of Service D is brought in-house. The Legacy Wrapper pattern can be further applied to wrap Service D with a standardized service contract that is in compliance with the design standards used in Service Inventory B.

Answer:

D

Explanation:

The Asynchronous Queuing pattern is applied to position a messaging queue between Service A, Service B, Service C, Service D, and Service Consumer A. This ensures that messages can be passed between these services without having to be in a stateful mode.

The Data Model Transformation and Protocol Bridging patterns are applied to enable communication between Service A and Service B, Service A and Service C, and Service A and Service D, despite their different data models and transport protocols.

The Redundant Implementation pattern is applied to bring a copy of Service D in-house to ensure that it can be accessed locally and reduce the unpredictability of its performance.

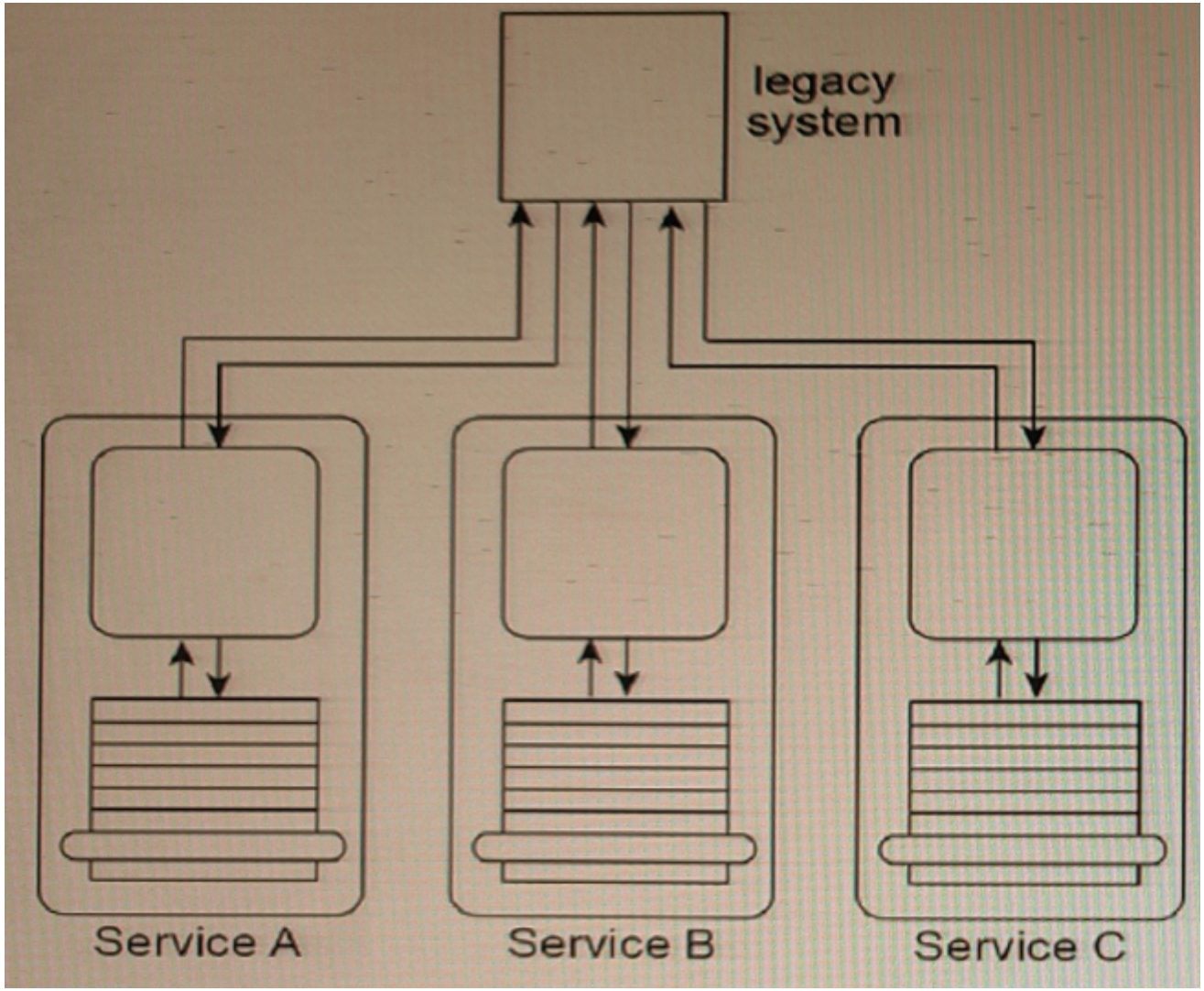
The Legacy Wrapper pattern is applied to wrap Service D with a standardized service contract that complies with the design standards used in Service Inventory B. This is useful for service consumers who want to use Service D but do not want to change their existing applications or service contracts.

Overall, this approach provides a comprehensive solution that addresses the issues with Service A, Service B, Service C, and Service D, while maintaining compliance with the Service Abstraction principle.

Question 6

Question Type: MultipleChoice

Refer to Exhibit.



Service A, Service B, and Service C are entity services, each designed to access the same shared legacy system. Service A manages order entities, Service B manages invoice entities, and Service C manages customer entities. Service A, Service B, and Service C are REST services and are frequently reused by different service compositions. The legacy system uses a proprietary file format that

Services A, B, and C need to convert to and from.

You are told that compositions involving Service A, Service B, and Service C are unnecessarily complicated due to the fact that order, invoice, and customer entitles are all related to each other. For example, an order has a customer, an invoice has an order, and so on. This results In calls to multiple services to reconstruct a complete order document. You are asked to architect a solution that will simplify the composition logic by minimizing the number of services required to support simple business functions like order management or bill payment. Additionally, you are asked to reduce the amount of redundant data transformation logic that is found in Services A, B, and C.

How will you accomplish these goals?

Options:

A- The Enterprise Service Bus pattern can be applied to introduce an intermediate processing layer between Services A, B, and C and the legacy system. The enterprise service bus can be used to consolidate and execute the necessary transformation logic currently held within the services. The Endpoint Redirection pattern can be applied to re-route calls from one service to another to provide access to related entity data.

B- The Legacy Wrapper pattern can be applied to create a service to expose the legacy system through a standardized service contract. The core logic of the wrapping service would provide all necessary data transformation functionality to convert between inventory-standardized data representations and the proprietary format. The Lightweight Endpoint pattern can be applied to establish lightweight capabilities that can return related entity data directly to service consumers.

C- The Enterprise Service Bus pattern can be applied to introduce an intermediate processing layer between Services A, B, and C and the legacy system. The enterprise service bus can be used to consolidate and execute the transformation logic currently held within the services. The Content Negotiation pattern can be applied to return a content link to related entity data to a service consumer, which allows for simpler and more dynamic composition logic. The service consumer effectively invokes the relevant service through the

returned link to obtain the related entity data.

D- The Legacy Wrapper pattern can be applied to create a service to expose the legacy system through a standardized service contract. The core logic of the wrapping service would provide all necessary data transformation functionality to convert between inventory-standardized data representations and the proprietary format. The Endpoint Redirection pattern can be applied to return a link to related entity data to a service consumer, which allows for simpler and more dynamic composition logic. The service consumer effectively invokes the relevant service through the returned link to obtain the related entity data.

Answer:

B

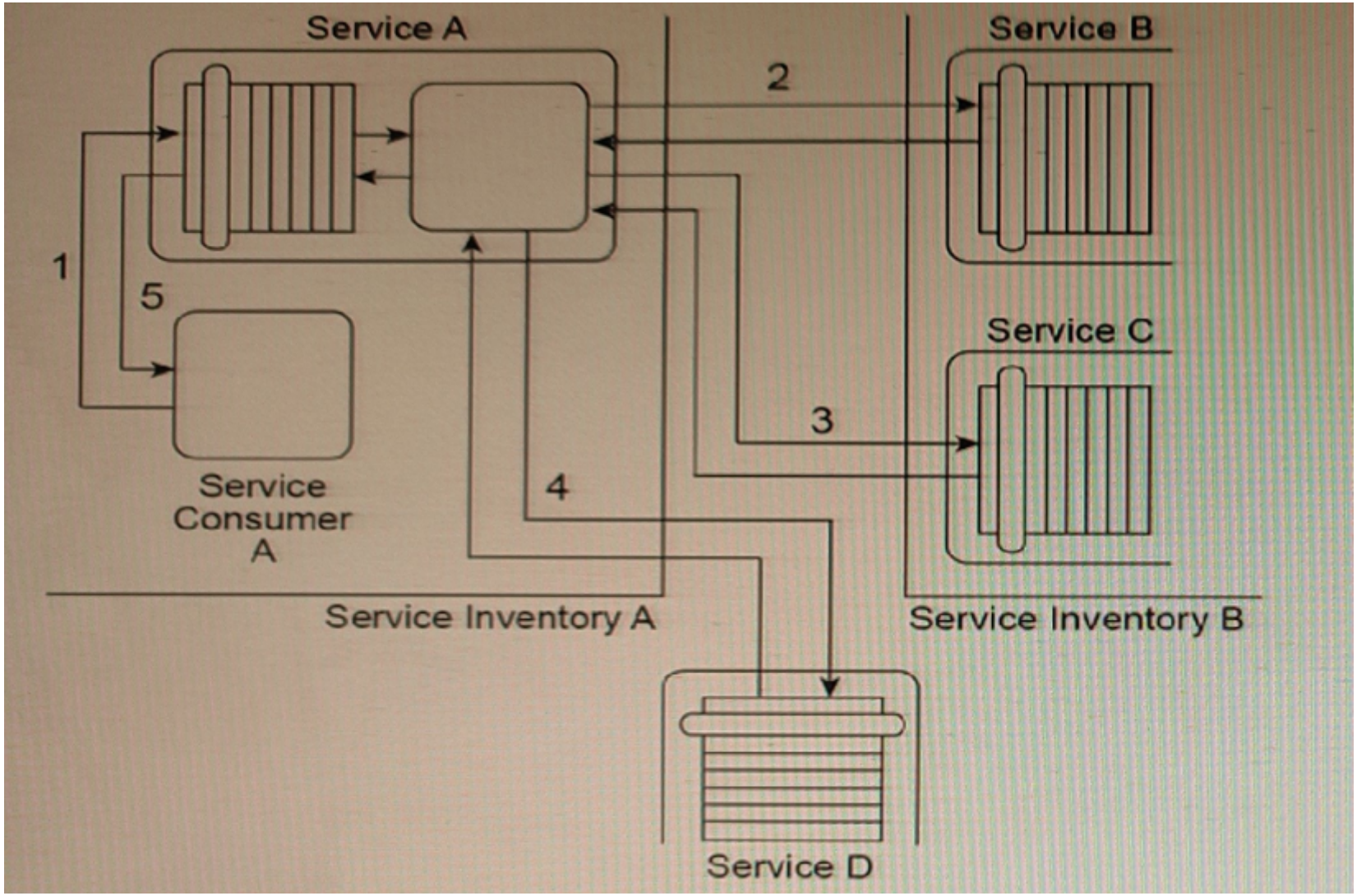
Explanation:

The Lightweight Endpoint pattern can be applied to establish lightweight capabilities that can return related entity data directly to service consumers, simplifying the composition logic by minimizing the number of services required to support simple business functions like order management or bill payment. This approach provides a standardized and simplified interface for the legacy system, reducing the complexity of the integration process with the entity services, and enabling them to focus on their core functionality.

Question 7

Question Type: MultipleChoice

Refer to Exhibit.



Service Consumer A and Service A reside in Service Inventory

Options:

A- Service B and Service C reside in Service Inventory B. Service D is a public service that can be openly accessed via the World Wide Web. The service is also available for purchase so that it can be deployed independently within IT enterprises. Due to the rigorous application of the Service Abstraction principle within Service Inventory B, the only information that is made available about Service B and Service C are the published service contracts. For Service D, the service contract plus a service level agreement (SLA) are made available. The SLA indicates that Service D has a planned outage every night from 11:00pm to midnight.
You are an architect with a project team that is building services for Service Inventory A. You are told that the owners of Service Inventory A and Service Inventory B are not generally cooperative or communicative. Cross-inventory service composition is tolerated, but not directly supported. As a result, no SLAs for Service B and Service C are available and you have no knowledge about how available these services are. Based on the service contracts you can determine that the services in Service Inventory B use different data models and a different transport protocol than the services in Service Inventory A. Furthermore, recent testing results have shown that the performance of Service D is highly unpredictable due to the heavy amount of concurrent access it receives from service consumers from other organizations. You are also told that there is a concern over how long Service Consumer A will need to remain stateful while waiting for a response from Service A.
What steps can be taken to solve these problems?

A- The Event-Driven Messaging pattern can be applied to establish a subscriber-publisher relationship 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 can be applied to position a central messaging queue between Service A and Service B and between Service A and Service C. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B and between Service A and Service C. The Redundant Implementation pattern can be applied so that a copy of Service D is brought in-house and

made part of Service Inventory A.

B- The Asynchronous Queuing pattern can be applied to position a central messaging queue between Service A and Service B and between Service A and Service C and so that a separate messaging queue is positioned between Service A and Service Consumer A. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B and between Service A and Service C. The Redundant Implementation pattern can be applied so that a copy of Service D is brought in-house. The Legacy Wrapper pattern can be further applied to wrap Service D with a standardized service contract that is in compliance with the design standards used in Service Inventory A.

C- The Containerization pattern can be applied to establish an environment for Service A to perform its processing autonomously. This gives Service A the flexibility to provide Service Consumer A with response messages consistently. The Asynchronous Queuing pattern can be applied so that a central messaging queue is positioned between Service A and Service B, between Service A and Service C, and between Service A and Service D. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B and between Service A and Service C.

D- The Asynchronous Queuing pattern can be applied to position a message queue between Service A and Service B, between Service A and Service C, and between Service A and Service D. Additionally, a separate messaging queue is positioned between Service A and Service Consumer A. The Data Model Transformation and Protocol Bridging patterns can be applied to enable communication between Service A and Service B, between Service A and Service C, and between Service A and Service D. The Redundant Implementation pattern can be applied so that a copy of Service D is brought in-house. The Legacy Wrapper pattern can be further applied to wrap Service D with a standardized service contract that is in compliance with the design standards used in Service Inventory B.

Answer:

D

Explanation:

The Asynchronous Queuing pattern is applied to position a messaging queue between Service A, Service B, Service C, Service D, and Service Consumer A. This ensures that messages can be passed between these services without having to be in a stateful mode.

The Data Model Transformation and Protocol Bridging patterns are applied to enable communication between Service A and Service B, Service A and Service C, and Service A and Service D, despite their different data models and transport protocols.

The Redundant Implementation pattern is applied to bring a copy of Service D in-house to ensure that it can be accessed locally and reduce the unpredictability of its performance.

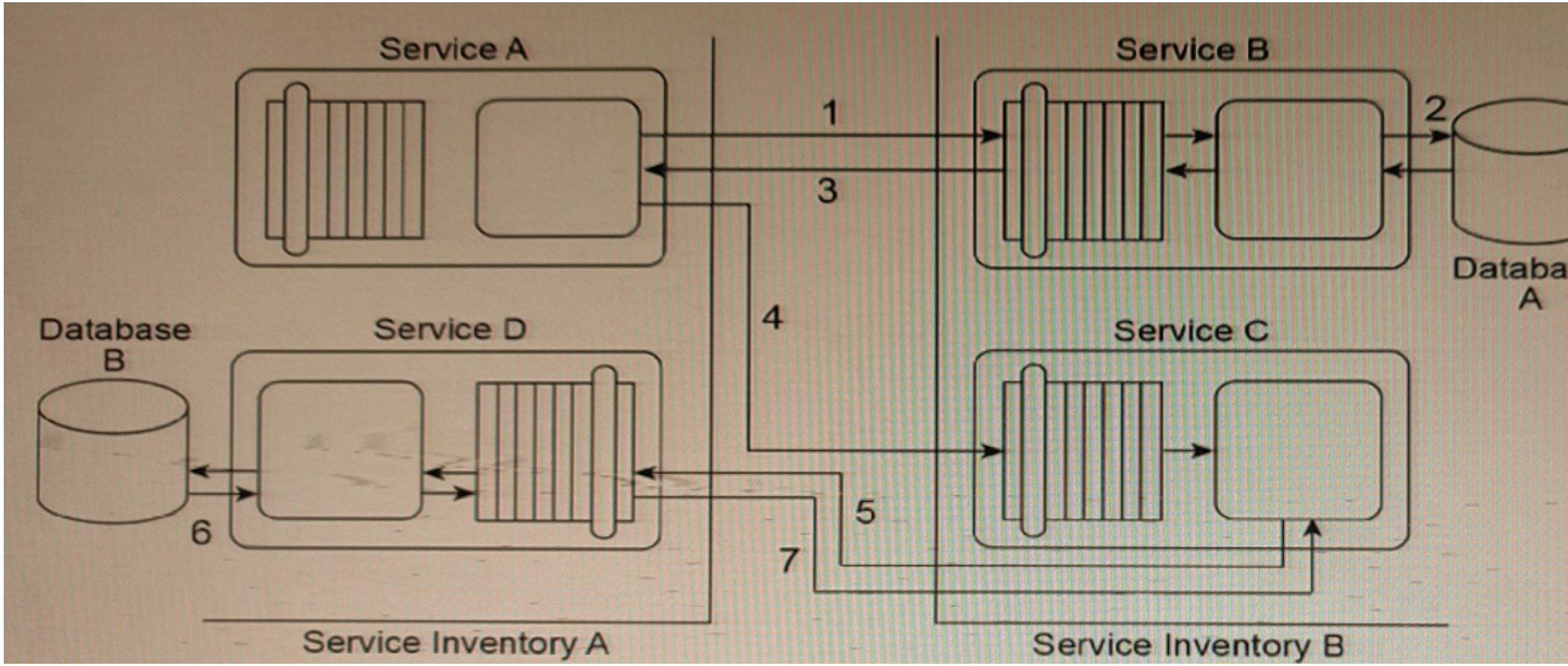
The Legacy Wrapper pattern is applied to wrap Service D with a standardized service contract that complies with the design standards used in Service Inventory B. This is useful for service consumers who want to use Service D but do not want to change their existing applications or service contracts.

Overall, this approach provides a comprehensive solution that addresses the issues with Service A, Service B, Service C, and Service D, while maintaining compliance with the Service Abstraction principle.

Question 8

Question Type: MultipleChoice

Refer to Exhibit.



Service A sends a message to Service B (1). After Service B writes the message contents to Database A (2), it issues a response message back to Service A (3). Service A then sends a message to Service C (4). Upon receiving this message, Service C sends a message to Service D (5), which then writes the message contents to Database B (6) and issues a response message back to Service C (7).

Service A and Service D are located in Service Inventory

Options:

A- Service B and Service C are located in Service Inventory B.

You are told that In this service composition architecture, all four services are exchanging invoice-related data in an XML format. However, the services in Service Inventory A are standardized to use a different XML schema for invoice data than the services in Service Inventory B. Also, Database A can only accept data in the Comma Separated Value (CSV) format and therefore cannot accept XML-formatted data. Database B only accepts XML-formatted data. However, it is a legacy database that uses a proprietary XML schema to represent invoice data that is different from the XML schema used by services in Service Inventory A or Service Inventory B. What steps can be taken to enable the planned data exchange between these four services?

A- The Data Model Transformation pattern can be applied so that data model transformation logic is positioned between Service A and Service B, between Service C and Service D, and between the Service D logic and Database B. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between Service A and Service C, and between the Service B logic and Database A.

B- The Protocol Bridging pattern can be applied so that protocol conversion logic is positioned between the Service B logic and Database A. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between Service A and Service B, between Service A and Service C, between Service C and Service D, and between the Service D logic and Database B.

C- The Data Model Transformation pattern can be applied so that data model transformation logic is positioned between Service A and Service B, between Service A and Service C, between Service C and Service D, and between the Service D logic and Database B. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between the Service B logic and Database A.

D- The Protocol Bridging pattern can be applied so that protocol conversion logic is positioned between Service A and Service B, between Service A and Service C, and between Service C and Service D. The Data Format Transformation pattern can be applied so that data format transformation logic is positioned between the Service B logic and Database A and between the Service D logic and

Database B.

Answer:

C

Explanation:

This solution addresses the two main challenges in the service composition architecture: the different XML schema used by services in Service Inventory A and Service Inventory B, and the incompatible data formats of the two databases.

By applying the Data Model Transformation pattern, data model transformation logic can be inserted to map the invoice-related data between the different XML schemas used by the services in Service Inventory A and Service Inventory B. This can be done at the appropriate points in the message flow: between Service A and Service B, between Service A and Service C, between Service C and Service D, and between the Service D logic and Database B.

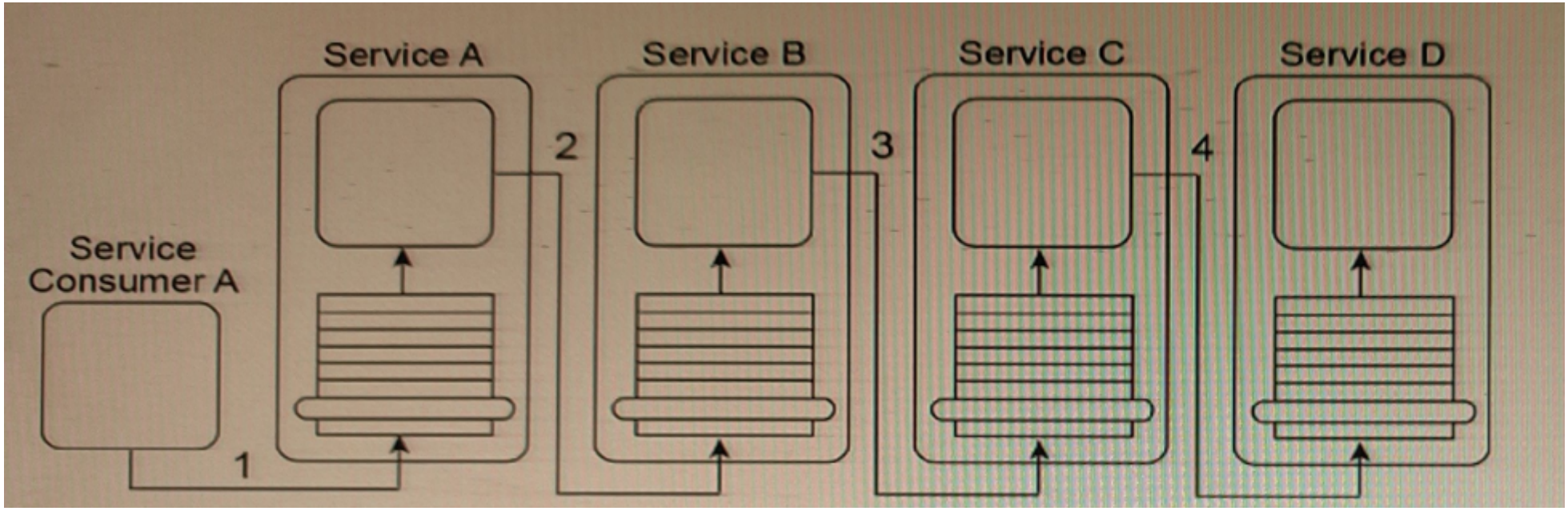
By applying the Data Format Transformation pattern, data format transformation logic can be inserted to convert the XML-formatted data used by the services to the CSV format required by Database A, and to convert the proprietary XML schema used by Database B to the XML schema used by the services. This can be done between the Service B logic and Database A.

The Protocol Bridging pattern is not necessary in this case because all services are already communicating using the same protocol (presumably HTTP or a similar protocol).

Question 9

Question Type: MultipleChoice

Refer to Exhibit.



Service Consumer A sends a message to Service A (1), which then forwards the message to Service B (2). Service B forwards the message to Service C (3), which finally forwards the message to Service D (4). However, Services A, B and C each contain logic that reads the contents of the message to determine what intermediate processing to perform and which service to forward the message to. As a result, what is shown in the diagram is only one of several possible runtime scenarios.

Currently, this service composition architecture is performing adequately, despite the number of services that can be involved in the transmission of one message. However, you are told that new logic is being added to Service A that will require it to compose one other service to retrieve new data at runtime that Service A will need access to in order to determine where to forward the message to. The involvement of the additional service will make the service composition too large and slow.

What steps can be taken to improve the service composition architecture while still accommodating the new requirements and avoiding an increase in the amount of service composition members?

Options:

A- The Service Instance Routing pattern can be applied to introduce a Routing service to provide a centralized service to contain routing-related business rules. This new Routing service can be accessed by Service A and Service C so they can determine where to forward messages to at runtime. The Service Reusability principle can be further applied to ensure that the logic in all remaining services is designed to be multi-purpose and reusable.

B- The Asynchronous Queuing pattern can be applied together with a Routing service that is invoked by messages read from a messaging queue. This new Routing service can replace Service B and can be accessed by Service A and Service C so they can determine where to forward messages to at runtime. The Service Loose Coupling principle can be further applied to ensure that the new Routing service remains decoupled from other services so that it can perform its routing functions independently from service contract invocation.

C- The Intermediate Routing pattern can be applied together with the Service Agent pattern by removing Service B or Service C from the service composition and replacing it with a service agent capable of intercepting and forwarding the message at runtime based on pre-defined routing logic. The Service Discoverability principle can be further applied to ensure that Service A can be found by any future service consumers.

D- The Intermediate Routing pattern can be applied together with the Service Agent pattern to establish a service agent capable of intercepting and forwarding the message at runtime based on pre-defined routing logic. The Service Composability principle can be further applied to ensure that all services are designed as effective service composition participants.

Answer:

B

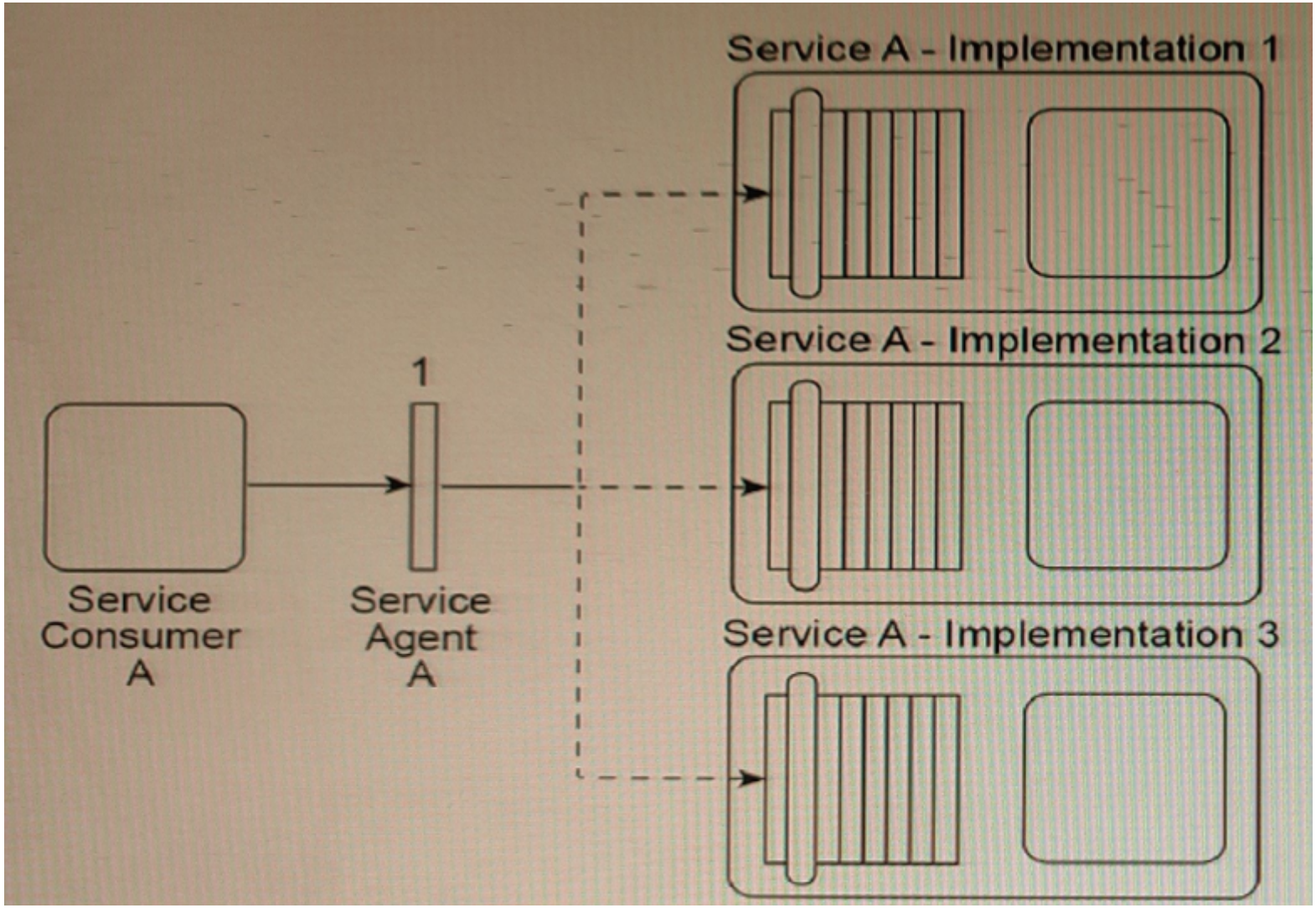
Explanation:

This solution addresses the issue of the service composition becoming too large and slow by introducing a new Routing service that is invoked by messages read from a messaging queue. This allows Service A and Service C to determine where to forward messages to at runtime without the need for additional services in the composition. The Service Loose Coupling principle is applied to ensure that the new Routing service remains decoupled from other services so that it can perform its routing functions independently from service contract invocation.

Question 10

Question Type: **MultipleChoice**

Refer to Exhibit.



Service Consumer A sends a message to Service

Options:

A- There are currently three duplicate implementations of Service A (Implementation 1, Implementation 2 and Implementation 3). The message sent by Service Consumer A is intercepted by Service Agent A (1), which determines at runtime which implementation of Service A to forward the message to. All three implementations of Service A reside on the same physical server.
You are told that despite the fact that duplicate implementations of Service A exist, performance is still poor at times. You are also informed that a new service capability will soon need to be added to Service A to introduce functionality that will require access to a shared database being used by many other clients and applications in the IT enterprise. This is expected to add further performance demands on Service A.
How can this service architecture be changed to improve performance in preparation for the addition of the new service capability?

A- The Standardized Service Contract principle can be applied to ensure that the new service capability extends the existing service contract in a manner that is compliant with current design standards. The Redundant Implementation pattern can be applied to establish separate implementations of Service A that include duplicate databases with copies of the data that Service A requires from the shared database.

B- The Service Autonomy principle can be applied to further isolate the individual implementations of Service A by separating them onto different physical servers. When the new service capability is added, the Service Data Replication pattern can be applied to give each implementation of Service A its own copy of the data it requires from the shared database.

C- The Service Loose Coupling principle can be applied together with the Standardized Service Contract principle to ensure that Service Consumer A is not indirectly coupled to the shared database after the new service capability is added to the service contract. The Legacy Wrapper pattern can be applied to establish a new utility service that will provide standardized data access service capabilities for the shared database.

D- The Service Autonomy principle can be applied to further isolate the individual implementations of Service A by separating them onto different physical servers. When the new service capability is added, the State Repository pattern can be applied to give each implementation of Service A its own copy of the data it requires from the shared database.

Answer:

B

Explanation:

By separating the individual implementations of Service A onto different physical servers, they can be isolated from each other and from other clients and applications in the IT enterprise, which can help improve performance. Additionally, using the Service Data Replication pattern to give each implementation of Service A its own copy of the data it requires from the shared database can help reduce the load on the shared database and improve performance. This can be especially important when a new service capability is added that requires access to the shared database, as it can help ensure that the performance of Service A is not impacted by the additional demands placed on the shared database.