S90.09 Q&As

SOA Design & Architecture Lab

Pass SOA S90.09 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.geekcert.com/s90-09.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by SOA Official Exam Center

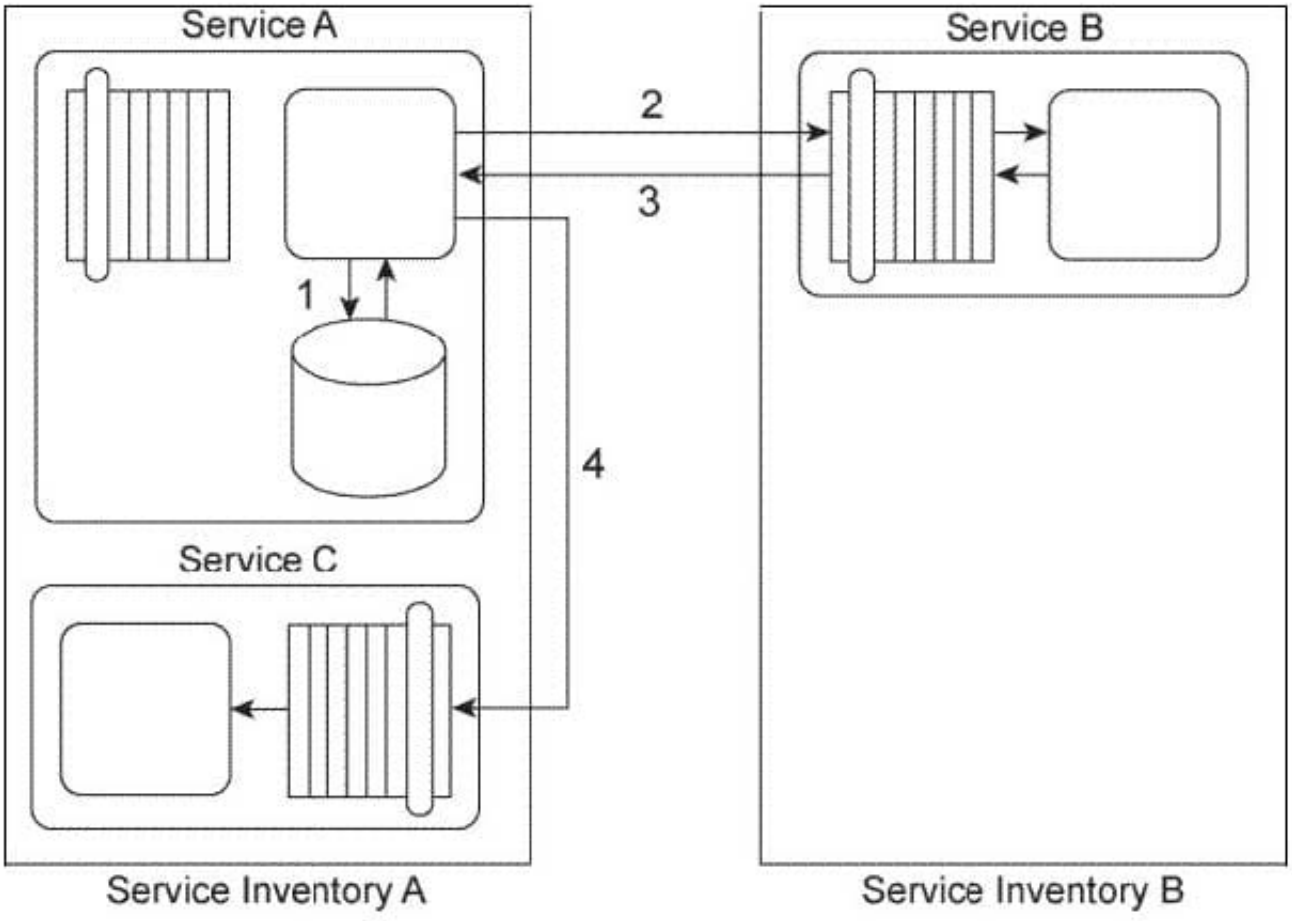
- **Instant Download** After Purchase
- **100% Money Back** Guarantee
- **365 Days** Free Update
- **800,000+** Satisfied Customers

QUESTION 1

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 were designed with service contracts based on different design standards than the services in Service Inventory B. As a result, Service A and Service B use different data models to represent the data they need to exchange. Therefore, Service A and Service B cannot currently communicate. Furthermore, Service C is an agnostic service that is heavily accessed by many concurrent service consumers. Service

C frequently reaches its usage thresholds during which it is not available and messages sent to it are not received. How can this service composition architecture be changed to avoid these problems?

A. The Data Model Transformation pattern can be applied by establishing an intermediate processing layer between Service A and Service B that can transform a message from one data model to another at runtime. The Intermediate Routing and Service Agent patterns can be applied so that when Service B sends a response message, a service agent can intercept the message and, based on its contents, either forward the message to Service A or route the message to Service C . The Service Autonomy principle can be further applied to Service C together with the Redundant Implementation pattern to help establish a more reliable and scalable service architecture.

B. The Data Model Transformation pattern can be applied by establishing an intermediate processing layer between Service A and Service B that can transform a message from one data model to another at runtime. The Asynchronous Queuing pattern can be applied to establish an intermediate queue between Service A and Service C so that when Service A needs to send a message to Service C, the queue will store the message and retransmit it to Service C until it is successfully delivered. The Service Autonomy principle can be further applied to Service C together with the Redundant Implementation pattern to help establish a more reliable and scalable service architecture.

C. The Data Model Transformation pattern can be applied by establishing an intermediate processing layer between Service A and Service B that can transform a message from one data model to another at runtime. The Intermediate Routing and Service Agent patterns can be applied so that when Service B sends a response message, a service agent can intercept the message and, based on its contents, either forward the message to Service A or route the message to Service C . 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.

D. None of the above.

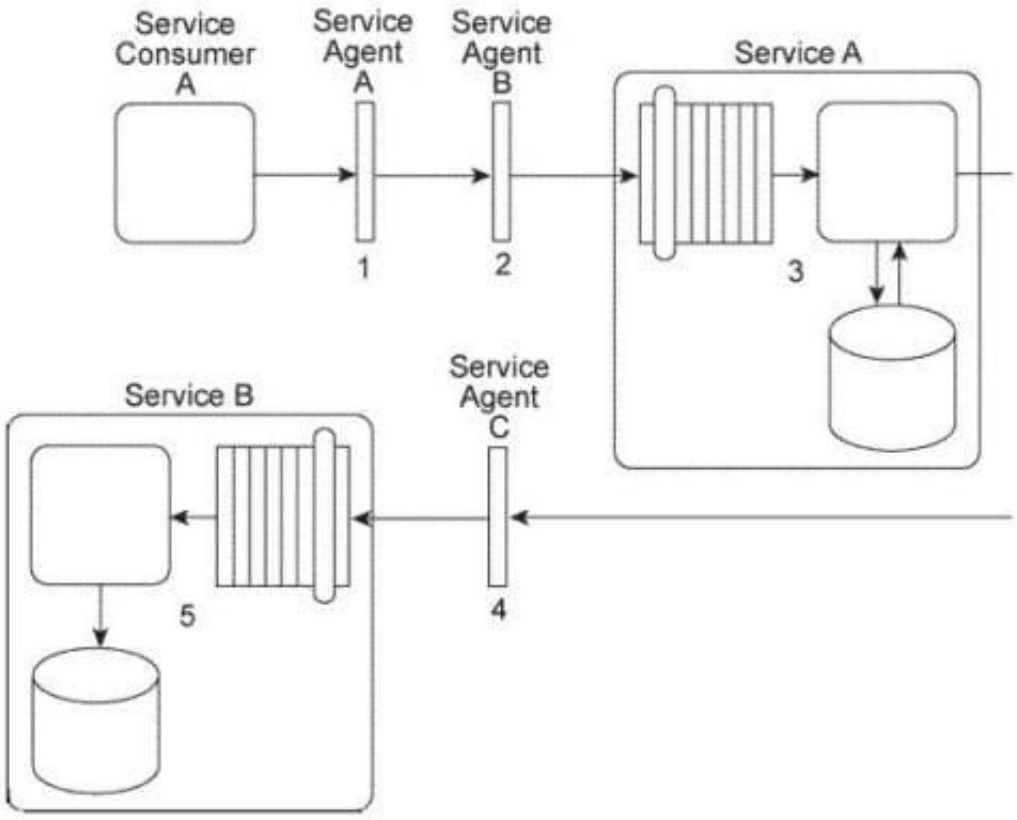
Correct Answer: B

QUESTION 2

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 3

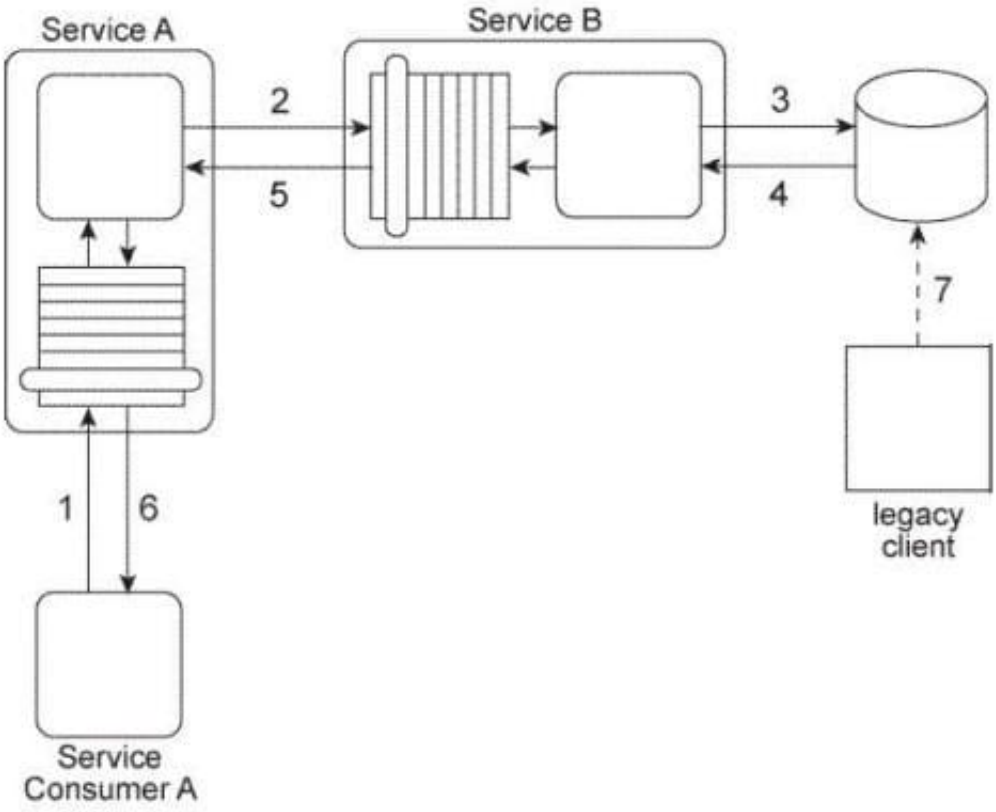
Service A is an entity service that provides a Get capability that returns a data value that is frequently changed.

Service Consumer A invokes Service A in order to request this data value (1). For Service A to carry out this request, it must invoke Service B (2), a utility service that interacts (3.4) with the database in which the data value is stored. Regardless of whether the data value changed, Service B returns the latest value to Service A (5), and Service A

returns the latest value to Service Consumer A (6).

The data value is changed when the legacy client program updates the database (7) When this change happens is not predictable. Note also that Service A and Service B are not always available at the same time.

Any time the data value changes. Service Consumer A needs to receive it as soon as possible. Therefore, Service Consumer A initiates the message exchange shown in the Figure several times a day. When it receives the same data value as before, the response from Service A is ignored. When Service A provides an updated data value, Service Consumer A can process it to carry out its task.



The current service composition architecture is using up too many resources due to the repeated invocation of Service A by Service Consumer A and the resulting message exchanges that occur with each invocation. What steps can be taken to solve this problem?

A. The Event-Driven Messaging pattern can be applied by establishing a subscriber- publisher relationship between Service A and Service B . This way, every time the data value is updated, an event is triggered and Service B, acting as the publisher, can notify Service A, which acts as the subscriber. The Asynchronous Queuing pattern can be applied between Service A and Service B so that the event notification message sent out by Service B will be received by Service A, even when Service A is unavailable.

B. The Event-Driven Messaging pattern can be applied by establishing a subscriber- publisher relationship between Service Consumer A and Service A . This way, every time the data value is updated, an event is triggered and Service A, acting as the publisher, can notify Service Consumer A, which acts as the subscriber. The Asynchronous Queuing pattern can be applied between Service Consumer A and Service A so that the event notification message sent out by Service A will be

received by Service Consumer A, even when Service Consumer A is unavailable.

C. The Asynchronous Queuing pattern can be applied so that messaging queues are established between Service A and Service B and between Service Consumer A and Service A . This way, messages are never lost due to the

unavailability of Service A or Service B.

D. None of the above.

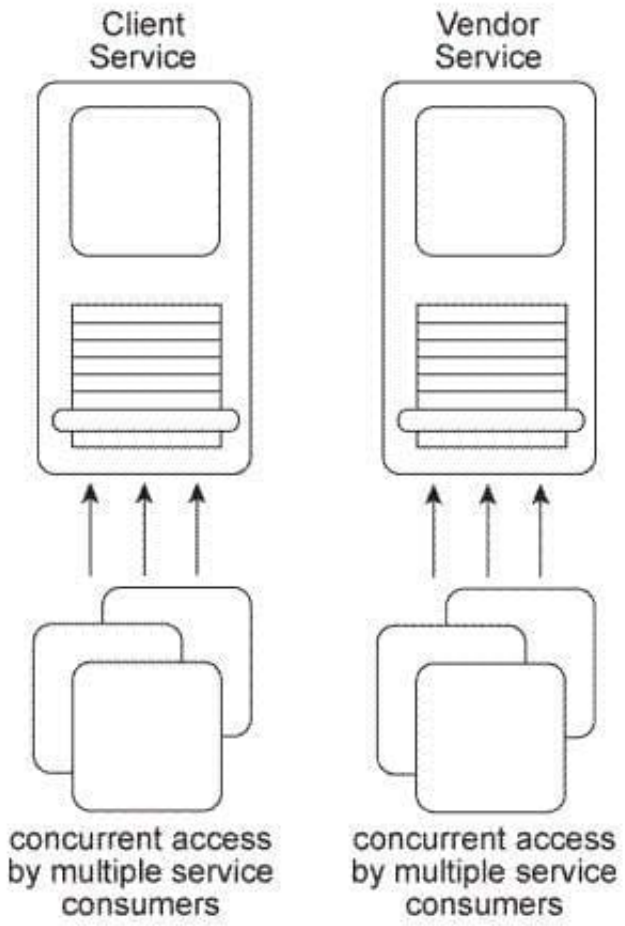
Correct Answer: D

QUESTION 4

The Client and Vendor services are agnostic services that are both currently part of multiple service compositions. As a result, these services are sometimes subjected to concurrent access by multiple service consumers.

The Client service is an entity service that primarily provides data access logic to a client database but also provides some calculation logic associated with determining a client\\'s credit rating. The Vendor service is also an entity service that provides some data access logic but can also generate various dynamic reports.

After reviewing historical statistics about the runtime activity of the two services, it was discovered that the majority of concurrent runtime access is related to the processing of business rules. With the Client service, it is the calculation logic that is frequently required and with the Vendor service it is the dynamic reporting logic that needs to be accessed separately from the actual report generation.



Currently, due to the increasing amount of concurrent access by service consumers, the runtime performance of both the Client and Vendor services has worsened and has therefore reduced their effectiveness as service composition members. What steps can be taken to solve this problem without introducing new services?

A. The Rules Centralization pattern can be applied by extracting the business rule logic from the Client and Vendor services and placing it into a new Rules service. This will naturally improve the runtime performance of the Client and Vendor services because they will no longer be subjected to the high concurrent access of service consumers that require access to the business rules logic.

B. The Redundant Implementation pattern can be applied to the Client and Vendor services, thereby establishing duplicate implementations that can be accessed when a service reaches its runtime usage threshold. The Intermediate Routing pattern can be further applied to provide load balancing logic that can, at runtime, determine which of the redundant service implementations is the least busy for a given service consumer request.

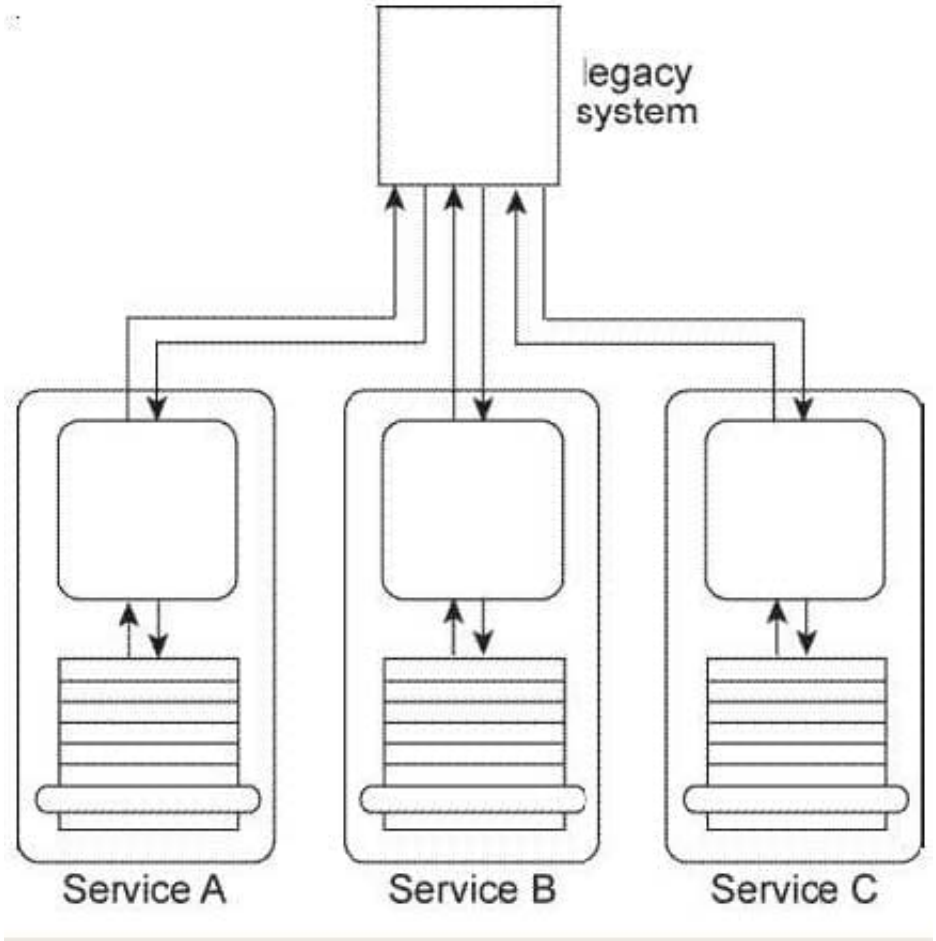
C. The Rules Centralization pattern can be applied together with the Redundant Implementation pattern to establish a scalable Rules service that is redundantly implemented and therefore capable of supporting high concurrent access from many service consumers. The Service Abstraction principle can be further applied to hide the implementation details of the Rules service.

D. None of the above.

Correct Answer: B

QUESTION 5

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