## 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.passapply.com/s90-09.html<br>100\% Passing Guarantee<br>100\% Money Back Assurance

Following Questions and Answers are all new published by SOA Official Exam Center
*) 800,000+ Satisfied Customers


## QUESTION 1

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 2

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.

VCE \& PDF
https://www.passapply.com/s90-09.html
2024 Latest passapply S90.09 PDF and VCE dumps Download


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 3

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.


How can the application of the Orchestration pattern improve this architecture?
A. The application of the Orchestration pattern will result in an environment whereby the State Repository and Service Data Replication patterns are naturally applied, allowing the shared state database to be replicated for Services A and B so that each task service can have its own dedicated state database. The Process Centralization pattern can also be applied to Services A and B, so that their logic is physically centralized, turning them into orchestrated task services.
B. The application of the Orchestration pattern will result in an environment whereby the Process Abstraction and Process Centralization patterns are naturally applied to Services A, B, and C, resulting in a clean separation of nonagnostic task services from newly designed agnostic services with reuse potential. Also, the State Repository pattern can be applied by the availability of a central state database that can be shared by Services A and
C. This database can be made available as a local part of the environment so that Services $A$ and $B$ can avoid remote communication.
D. The application of the Orchestration pattern will result in an environment whereby the Compensating Service Transaction is naturally applied, resulting in the opportunity to create sophisticated exception logic that can be used to compensate for the performance problems caused by Services A and B having to remotely access the state database. The Process Abstraction and Service Broker patterns are also naturally applied, enabling the separation of non-agnostic logic and agnostic logic while providing common transformation functions required to overcome any disparity in the service contracts that will need to be created for the new agnostic services.
E. None of the above.

Correct Answer: B

## QUESTION 4

When Service A receives a message from Service Consumer $A(1)$,the message is processed by Component $A$. This component first invokes Component $B(2)$, which uses values from the message to query Database A in order to retrieve additional data. Component B then returns the additional data to Component A .

Component $A$ then invokes Component $C$ (3), which interacts with the API of a legacy system to retrieve a new data value. Component $C$ then returns the data value back to Component $A$.

Next, Component A sends some of the data it has accumulated to Component D (4), which writes the data to a te>X file that is placed in a specific folder. Component $D$ then waits until this file is imported into a different system via a regularly scheduled batch import. Upon completion of the import, Component $D$ returns a success or failure code back to Component A.

Component A finally sends a response to Service Consumer A (5) containing all of the data collected so far and Service Consumer A writes all of the data to Database B (6).

Components A, B, C. and D belong to the Service A service architecture. Database A, the legacy system, and the file folders are shared resources within the IT enterprise.


Service A is an entity service with a service architecture that has grown over the past few years. As a result of a service inventory-wide redesign project, you are asked to revisit the Service A service architecture in order to separate the logic provided by Components B, C, and D into three different utility services without disrupting the behavior of Service A as it relates to Service Consumer A. What steps can be taken to fulfill these requirements?
A. The Legacy Wrapper pattern can be applied so that Component B is separated into a separate wrapper utility service that wraps the shared database. The Asynchronous Queuing pattern can be applied so that a messaging queue is positioned between Component A and Component C, thereby enabling communication during times when the legacy system may be unavailable or heavily accessed by other parts of the IT enterprise. The Service Facade pattern can be applied so that a Facade component is added between Component $A$ and Component $D$ so that any change in behavior can be compensated. The Service Autonomy principle can be further applied to Service A to help make up for any performance loss that may result from splitting the component into a separate wrapper utility service.
B. The Legacy Wrapper pattern can be applied so that Component $B$ is separated into a separate utility service that wraps the shared database. The Legacy Wrapper pattern can be applied again so that Component C is separated into a separate utility service that acts as a wrapper for the legacy system API. The Legacy Wrapper pattern can be applied once more to Component D so that it is separated into another utility service that provides standardized access to the file folder. The Service Facade pattern can be applied so that three Facade components are added: one between Component A and each of the new wrapper utility services. This way, the Facade components can compensate for any change in behavior that may occur as a result of the separation. The Service Composability principle can be further applied to Service A and the three new wrapper utility services so that all four services are optimized for participation in the new service composition. This will help make up for any performance loss that may result from spliting the three components into separate services.
C. The Legacy Wrapper pattern can be applied so that Component $B$ is separated into a separate utility service that wraps the shared database. The Legacy Wrapper pattern can be applied again so that Component C is separated into a separate utility service that acts as a wrapper for the legacy system API. Component D is separated into a separate service and the Event-Driven Messaging pattern is applied to establish a publisher-subscriber relationship between this new service and Component A. The interaction between Service Consumer A and Component A is then redesigned so
that Component $A$ first interacts with Component $B$ and the new wrapper service. Service $A$ then issues a final message back to Service Consumer A. The Service Composability principle can be further applied to Service A and the three new wrapper utility services so that all four services are optimized for participation in the new service composition. This will help make up for any performance loss that may result from splitting
the three components into separate services.
D. None of the above.

Correct Answer: B

## QUESTION 5

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 6

Service Consumer A sends a message to Service A. There are currently three duplicate implementations of Service A (Implementation 1, Implementation 2, 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 after Service A was deployed, each of its three implementations was claimed by a different IT department, which means each implementation of Service A has a different owner. You are informed that a new service capability will soon need to be added to Service A. This service capability will introduce new business logic specific to Service A as well as logic required to access a shared database. What steps can be taken to ensure that the service owners will each add the service capability in a consistent manner to their respective implementations of Service A?
A. The Contract Centralization pattern can be applied so that when the new service capability is added, the Service A service contract will become the primary contact point for Service A. This will avoid Service Consumer A or any other potential service consumer from being designed to access the shared database directly. The Service Abstraction principle can be applied to further hide the implementation details so that Service Consumer A and other service consumers are unaware of the fact that the shared database is being accessed.
B. The Legacy Wrapper pattern can be applied to establish a new wrapper utility service that will provide standardized data access service capabilities for the shared database. This will avoid Service A from having to access the shared database directly and will further support the application of the Service Loose Coupling principle between Service A and the new utility service. By abstracting the data access logic into the wrapper service, there is no need to add the new service capability to each implementation of Service A.
C. The Standardized Service Contract principle is applied to ensure that the new service capability is consistently added to the service contract of each implementation and that it extends the existing Service A service contract in a manner that is compliant with current design standards. The Service Loose Coupling principle is applied to ensure that the new service capability remains decoupled from the underlying logic and implementation so that Service Consumer A does not become indirectly coupled to any new logic or to the shared database.
D. None of the above.

Correct Answer: C

## QUESTION 7

Service A is an entity service that provides a set of generic and reusable service capabilities. In order to carry out the functionality of any one of its service capabilities, Service A is required to compose Service B
(1) and Service C (2) and Service A is required to access Database A (3), Database B (4), and Database C (5). These three databases are shared by other applications within the IT enterprise.

All of service capabilities provided by Service A are synchronous, which means that for each request a service consumer makes. Service $A$ is required to issue a response message after all of the processing has completed.

Depending on the nature of the service consumer request, Service A may be required to hold data it receives in memory until its underlying processing completes. This includes data it may receive from either Service A or Service B or from any of the three shared databases.

Service A is one of many entity services that reside in a highly normalized service inventory. Because Service A provides agnostic logic, it is heavily reused and is currently part of many service compositions.

## Service A

Service B


You are told that Service A has recently become unstable and unreliable. The problem has been traced to two issues with the current service architecture. First, Service B, which is also an entity service, is being increasingly reused and has itself become unstable and unreliable. When Service B fails, the failure is carried over to Service A.Secondly, shared Database B has a complex data model. Some of the queries issued by Service A to shared Database B can take a very long time to complete. What steps can be taken to solve these problems without compromising the normalization of the service inventory?
A. The Redundant Implementation pattern can be applied to Service A, thereby making duplicate deployments of the service available. This way, when one implementation of Service A is too busy, another implementation can be accessed by service consumers instead. The Service Data Replication pattern can be applied to establish a dedicated database that contains an exact copy of the data from shared Database B that is required by Service A.
B. The Redundant Implementation pattern can be applied to Service B, thereby making duplicate deployments of the service available. This way, when one implementation of Service $B$ is too busy, another implementation can be accessed by Service A instead. The Service Data Replication pattern can be applied to establish a dedicated database that contains an exact copy of the data from shared Database B that is required by Service A .
C. The Redundant Implementation pattern can be applied to Service B, thereby making duplicate deployments of the service available. This way, when one implementation of Service $B$ is too busy, another implementation can be accessed by Service A instead. The Service Data Replication pattern can be applied to establish a dedicated database that contains a copy of the data from shared Database B that is required by Service A. The replicated database is designed with an optimized data model in order to improve query execution performance.
D. None of the above.

Correct Answer: C

## QUESTION 8

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 9

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).

Services A, B, and C each contain logic that reads the content of the message and, based on this content, determines which service to forward the message to. As a result, what is shown in the Figure is 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 in order 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?
A. The Rules Centralization pattern can be applied to establish a centralized service that contains routing-related business rules. This new Rules service would replace Service B and could be accessed by Service A and Service C in order for Service A and Service C to determine where to forward a message to at runtime. The Service Composability principle can be further applied to ensure that all remaining services are designed as effective service composition participants.
B. The Asynchronous Queuing pattern can be applied together with the Rules Centralization pattern to establish a Rules service that encapsulates a messaging queue. This new Rules service would replace Service B and could be accessed by Service $A$ and Service $C$ in order for Service $A$ and Service $C$ to determine where to forward a message to at runtime. The Service Composability principle can be further applied to ensure that all remaining services are designed as effective service composition participants.
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 predefined routing logic. The Service Composability principle can be further applied to ensure that all remaining services are designed as effective service composition participants.
D. None of the above.

Correct Answer: C

## QUESTION 10

Service Consumer A invokes Service A (1). The logic within Service A is required to retrieve three independent data values from Services B, C, and D and to then return these data values back to Service Consumer A.

To accomplish this, Service A begins by sending a request message to Service B (2). After receiving a response message with the first data value from Service B, Service A sends a request message to Service C (3). After it receives a response message with the second data value from Service $C$, Service $A$ then sends a request message to Service $D$ (4). Upon receiving a response message with the third data value from Service D. Service A finally sends its own response message (containing all three collected data values) back to Service Consumer A.

Service Consumer A and Service A reside in Service Inventory 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 PM to midnight.


You are an architect with a project team 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 about 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 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 Redundant Implementation pattern is applied so that a copy of Service $D$ is brought in- house and made part of Service Inventory A.
B. 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 so that a separate messaging queue is positioned between Service A and Service Consumer
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 Redundant Implementation pattern is applied so that a copy of Service D is brought in- house for fail-over purposes. The Legacy Wrapper pattern is further applied to wrap Service D with a standardized service contract that is in compliance with the design standards used in Service Inventory
A. This wrapper utility service first attempts to access the external service, but if that service is
unavailable it will access the redundant internal service instead.
D. 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 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 .
E. None of the above.

Correct Answer: B

