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

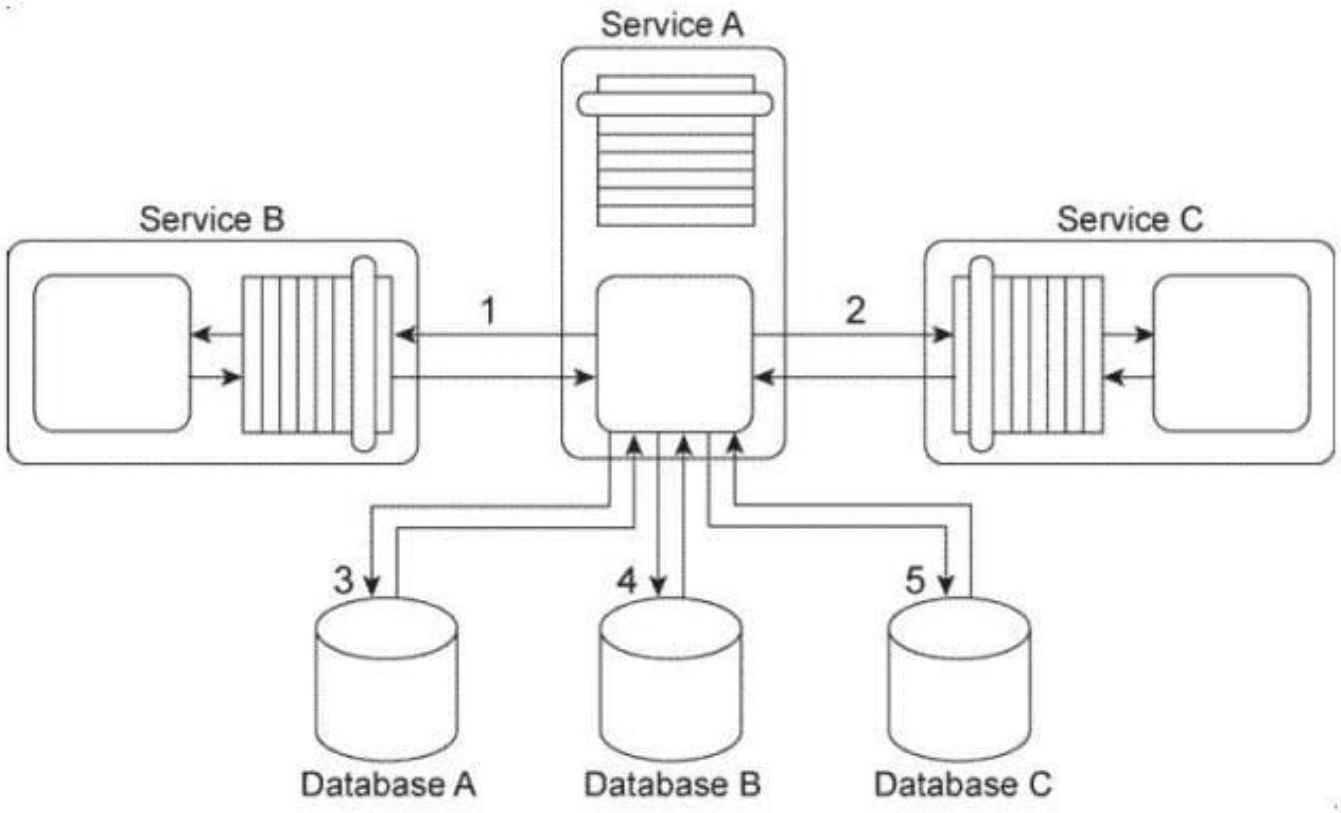
Service A is an entity service that provides a set of generic and reusable service capabilities. In order to carry out the functionality of any one of its service capabilities, Service A is required to compose Service B

(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.



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 2

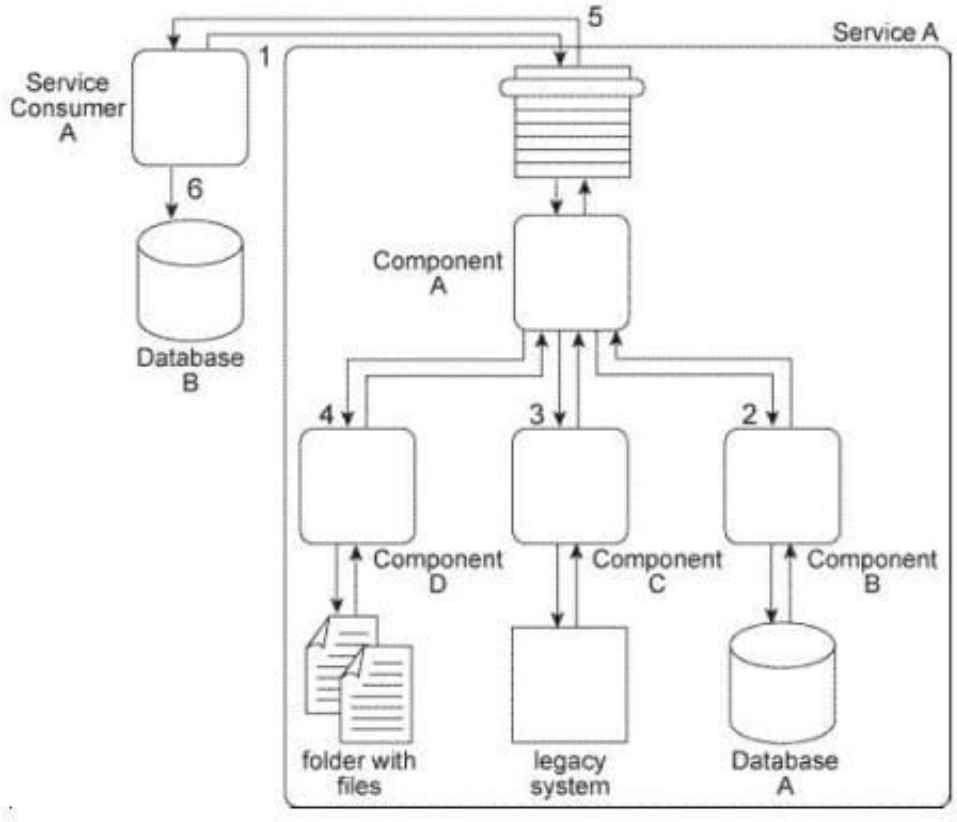
When Service A receives a message from Service Consumer A(1),the message is processed by Component A. This component first invokes Component B (2), which uses values from the message to query Database A in order to retrieve additional data. Component B then returns the additional data to Component A.

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 splitting 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.

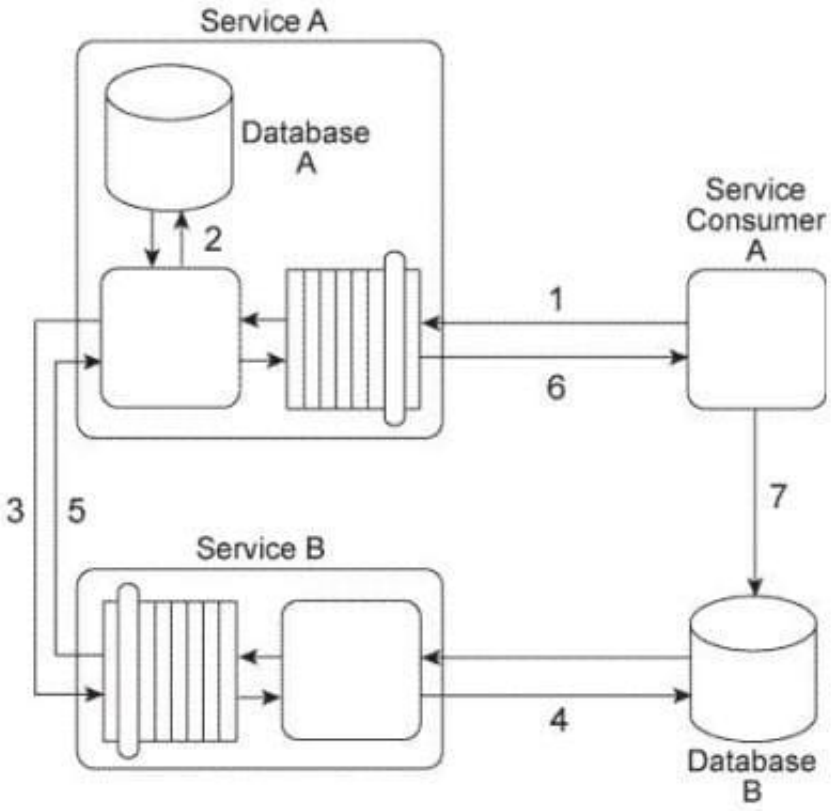
Correct Answer: B

QUESTION 3

Service Consumer A sends a message with a business document to Service A (1), which writes the business document to Database A (2). Service A then forwards the business document to Service B (3), which writes the business document to Database B (4).

Service B then responds to Service A with a message containing a failure or success code (5) after which Service A responds to Service Consumer A with a message containing a failure or success code (6). Upon receiving the message, Service Consumer A updates a log table in Database B (7). The log entry is comprised of the entire business document.

Database A is dedicated to the Service A service architecture and Database B is a shared database.



There are two problems with this service composition architecture that you are asked to address: First, both Service Consumer A and Service B need to transform the business document data from an XML format to a proprietary Comma Separated Value (CSV) in order to write the data to Database B. This has led to redundant data format transformation

logic that has been difficult to keep in synch when Database B changes. Secondly, Service A is an entity service that is being reused by several other service compositions. It has lately developed reliability problems that have caused the service to become unavailable for extended periods. What steps can be taken to solve these problems?

A. The Legacy Wrapper pattern can be applied so that data access to Database B is separated into a new wrapper utility service. This way, the Data Format Transformation pattern only needs to be applied within the logic of this new service which will expose a standardized contract that both Service Consumer A and Service B can access. The Asynchronous Queuing pattern can be applied so that messaging queues are established between Service Consumer A and Service A and between Service A and Service B . The Service Autonomy principle can be further applied to Service A in order to establish a more isolated and reliable surrounding infrastructure.

B. The Legacy Wrapper pattern can be applied so that data access to Database B is separated into a new wrapper utility service. This way, the Data Format Transformation pattern only needs to be applied within the logic of this new service which will expose a standardized contract that both Service Consumer A and Service B can access. The Reliable Messaging pattern can be applied so that acknowledgements are passed between Service Consumer A and Service A and between Service A and Service B . The Service Composability principle can be further applied to Service A in order to optimize its service architecture for improved participation in multiple service compositions.

C. The service composition can be redesigned with the application of the Contract Centralization pattern so that instead of writing the business document to Database B, Service Consumer A sends the business document to Service B instead. This way, Service B would provide the only location where data format transformation logic for Database B needs to be carried out, which further supports the application of the Service Reusability principle. The Reliable Messaging pattern can be applied so that acknowledgements are passed between Service Consumer A and Service A and between Service A and Service B . The Service Composability principle can be further applied to Service A in order to optimize its service architecture for improved participation in multiple service compositions.

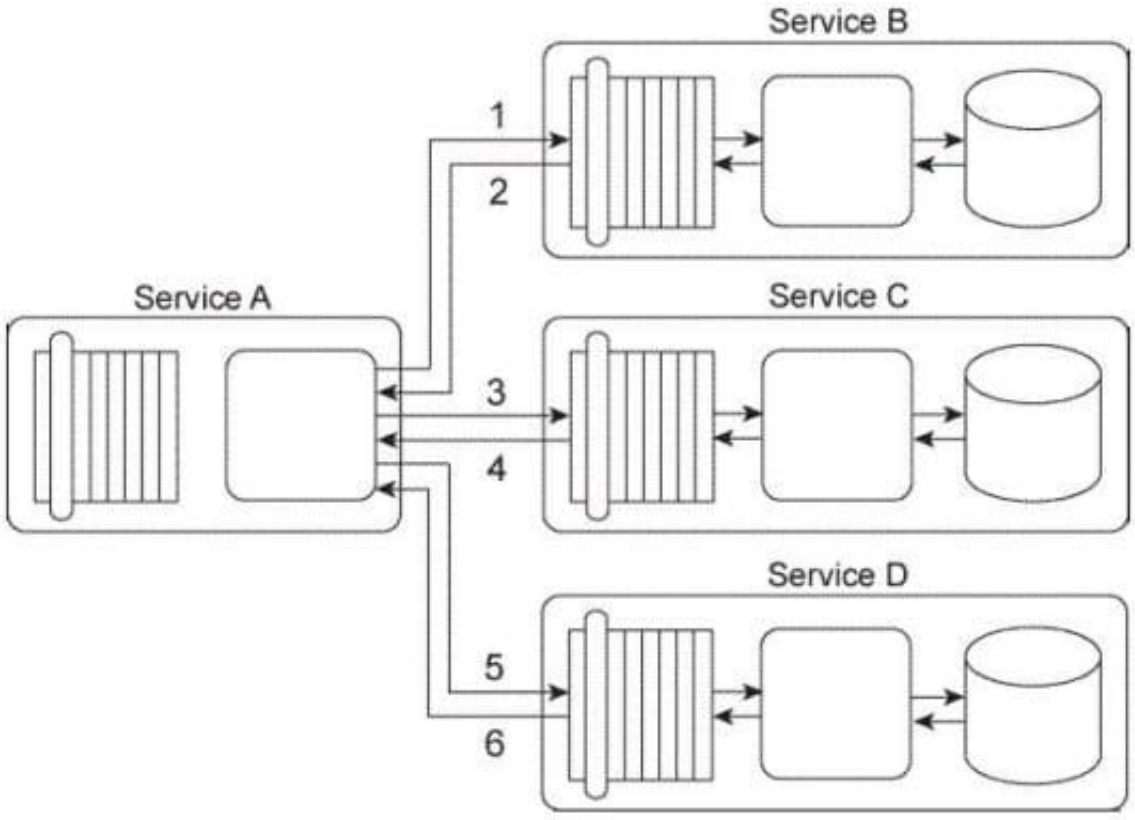
D. None of the above.

Correct Answer: A

QUESTION 4

Service A is a task service that is required to carry out a series of updates to a set of databases in order to complete a task. To perform the database updates Service A must interact with three other services, each of which provides standardized data access capabilities.

Service A sends its first update request message to Service B (1), which then responds with a message containing a success or failure code (2). Service A then sends its second update request message to Service C (3), which also responds with a message containing a success or failure code (4). Finally, Service A sends a request message to Service D (5), which responds with its own message containing a success or failure code (6).



You\\'ve been asked to change this service composition architecture in order to fulfill a set of new requirements: First, if the database update performed by Service B fails, then it must be logged by Service

A. Secondly, if the database update performed by Service C fails, then a notification e-mail must be sent out to a human administrator. Third, if the database update performed by either Service C or Service D fails, then both of these updates must be reversed so that the respective databases are restored back to their original states. What steps can be taken to fulfill these requirements?

A. Service A is updated to perform a logging routine when Service A receives a response message from Service B containing a failure code. Service A is further updated to send an e-mail notification to a human administrator if Service A receives a response message from Service C containing a failure code. The Atomic Service Transaction pattern is applied so that Services A, C, and D are encompassed in the scope of a transaction that will guarantee that if the database updates performed by either Service C or Service D fails, then both updates will be rolled back.

B. The Compensating Service Transaction pattern is applied to Service B so that it invokes exception handling logic that logs failed database updates before responding with a failure code back to Service

A . Similarly, the Compensating Service Transaction pattern is applied to Service C so that it issues an e-mail notification to a human administrator when a database update fails. The Atomic Service Transaction pattern is applied so that Services A, C, and D are encompassed in the scope of a transaction that will guarantee that if the database updates performed by either Service C or Service D fails, then both updates will be rolled back. The Service Autonomy principle is further applied to Service A to ensure that it remains consistently available to carry out this sequence of actions.

C. The Atomic Service Transaction pattern is applied so that Services A, C, and D are encompassed in the scope of a transaction that will guarantee that if the database updates performed by either Service C or Service D fails, then both updates will be rolled back. The Compensating Service Transaction pattern is then applied to all services so that the scope of the compensating transaction includes the scope of the atomic transaction. The compensating exception logic that is added to Service D automatically invokes Service B to log the failure condition and Service C to issue the e-mail notification to the human administrator. This way, it is guaranteed that the compensating logic is always executed

together with the atomic transaction logic.

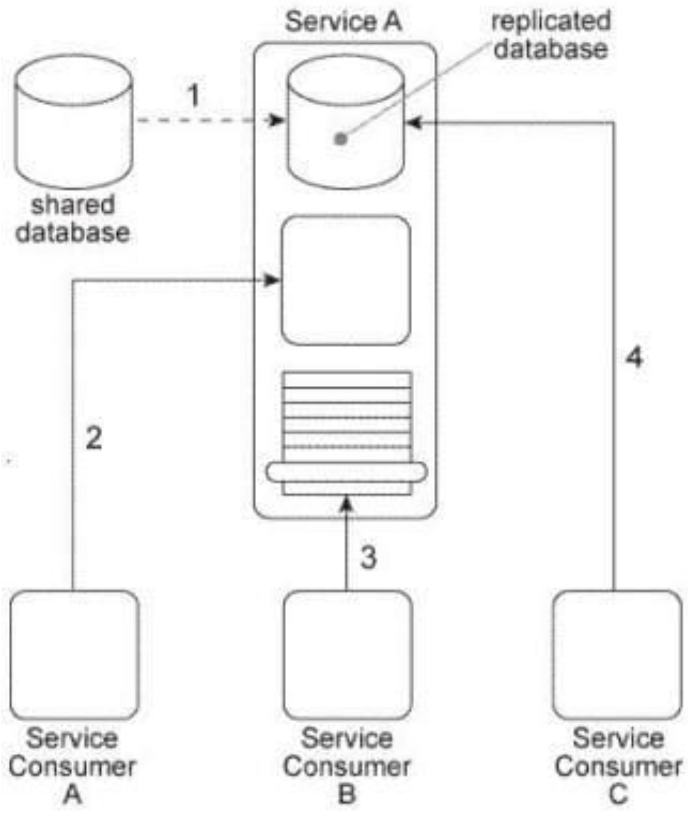
D. None of the above.

Correct Answer: A

QUESTION 5

Service A is a utility service that provides generic data access logic to a database that contains data that is periodically replicated from a shared database (1). Because the Standardized Service Contract principle was applied to the design of Service A, its service contract has been fully standardized.

Service A is being accessed by three service consumers. Service Consumer A accesses a component that is part of the Service A implementation by invoking it directly (2). Service Consumer B invokes Service A by accessing its service contract (3). Service Consumer C directly accesses the replicated database that is part of the Service A implementation (4).



You\\'ve been told that the reason Service Consumers A and C bypass the published Service A service contract is because, for security reasons, they are not allowed to access a subset of the operations in the WSDL definition that expresses the service contract. How can the Service A architecture be changed to enforce these security restrictions while avoiding negative forms of coupling?

A. The Contract Centralization pattern can be applied to force all service consumers to access the Service A architecture via its published service contract. This will prevent negative forms of coupling that could lead to problems when the database is replaced. The Service Abstraction principle can then be applied to hide underlying service architecture details so that future service consumers cannot be designed to access any part of the underlying service implementation.

B. The Contract Centralization pattern can be applied to force service consumers to access the Service A architecture via its published service contract only. The Service Loose Coupling principle can then be applied to ensure that the centralized service contract does not contain any content that is dependent on or derived from the underlying service implementation.

C. The Concurrent Contracts pattern can be applied to Service A in order to establish one or more alternative service contracts. This allows service consumers with different levels of security clearance to continue accessing the service logic via its published service contracts.

D. None of the above.

Correct Answer: C