S90-09A^Q&As

SOA Design & Architecture Lab

Pass SOA S90-09A Exam with 100% Guarantee

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

https://www.pass4itsure.com/S90-09A.html

100% Passing Guarantee
100% Money Back Assurance

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

QUESTION 1

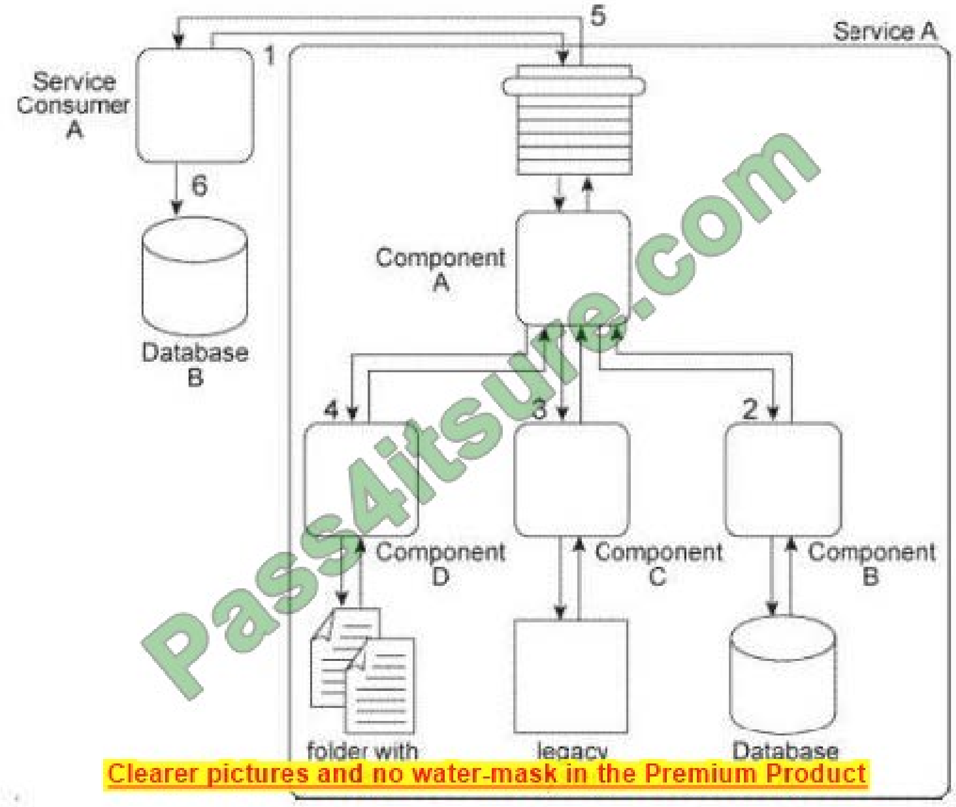
When Service A receives a message from Service Consumer A(1),the message is processed by Component A. This component first invokes Component B (2), which uses values from the message to query Database A in order to retrieve additional data. Component B then returns the additional data to Component A.

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 text 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 a task service that completes an entire business task on its own without having to compose other services. However, you have received many complaints about the reliability of Service A . Specifically, it has three problems. First, when Component B accesses Database A, it may not receive a response for several minutes when the database is being accessed by other applications in the IT enterprise. Secondly, the legacy system accessed by Component C frequently crashes and therefore becomes unavailable for extended periods of time. Third, for Component D to respond to Component A, it must first wait for the batch import of the files to occur. This can take several minutes during which Service Consumer A remains stateful and consumes excessive memory. What steps can be taken to address these three problems?

A. The Legacy Wrapper pattern can be applied so that Component B is separated to wrap the shared database, thereby allowing Component A to interact with this new service instead of directly interacting with the database. The Legacy Wrapper pattern can be applied again so that Component C is separated into a separate service that acts as a wrapper of the legacy system API. Component D can then be separated into a separate service and the Event-Driven Messaging pattern can be applied to establish a publisher- subscriber relationship between this new service and Component A and between Service A and Service Consumer A. The interaction between Service Consumer A and Component A is then redesigned so that Component A issues a message back to Service Consumer A when the event related to the batch import is triggered.

B. The Service Data Replication pattern can be applied so that Component B can access a replicated database instead of having to access the shared Database A directly. The Legacy Wrapper pattern can be applied so that Component C is separated into a separate service that acts as a wrapper of the legacy system API. Next, the Reliable Messaging pattern can be applied so that acknowledgements are issued from the new wrapper service to Component A, thereby enabling notifying Component A during times when the legacy system is unavailable. Finally, 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.

C. The Service Data Replication pattern can be applied so that Component B can access a replicated database instead of having to access the shared Database A directly. The Legacy Wrapper pattern can be applied so that Component C is separated into a separate service that acts as a wrapper of the legacy system API. Next, the Asynchronous Queuing pattern can be applied so that a messaging queue is positioned between Component A and the new wrapper service, thereby enabling communication during times when the legacy system is unavailable. Finally, Component D is separated into a new service and the Event-Driven Messaging pattern is applied to establish a publisher-subscriber relationship between this service and Component A and between Service A and Service Consumer A. The interaction logic is redesigned as follows: Component A interacts with Component B, the new wrapper service, and then issues a request to the new event-driven service. Upon receiving a response triggered by the event related to the batch import, Service A responds to Service Consumer A.

D. None of the above.

Correct Answer: C

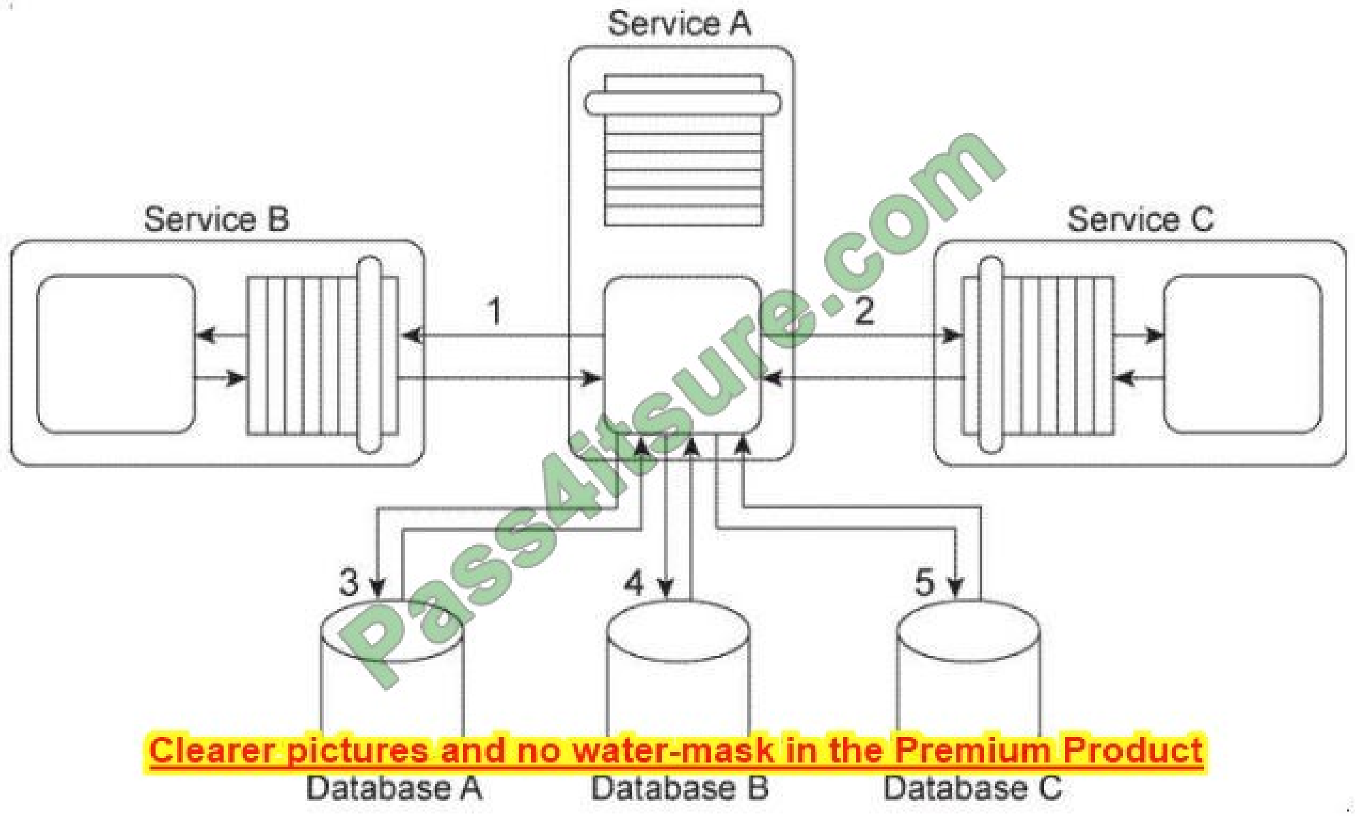
QUESTION 2

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 and several of the service consumers that access it have had to raise runtime exceptions due to these problems. What steps can be taken to solve these problems without compromising the normalization of the service inventory?

A. The Service Autonomy principle can be applied to increase the physical isolation of Service A and to reduce dependencies Service A has on external resources. In support of this, the Service Data Replication pattern can be applied in order to establish a dedicated database that contains replicated data from shared Databases A, B, and C . Furthermore, the Redundant Implementation pattern can be applied so that the logic Service A requires from Services B and C can be redundantly placed inside of Service A . This way, Service A avoids having to separately compose Services B and C

B. The Service Statelessness principle can be applied with the help of the State Repository pattern in order to establish a state database that Service A can use to defer state data it may be required to hold for extended periods. The Service Autonomy principle can also be applied in order to increase the physical isolation of Service A and to reduce dependencies Service A has on external resources. In support of this, the Service Data Replication pattern can be applied in order to establish a dedicated database that contains replicated data from shared Databases A, B, and C.

C. The Service Loose Coupling and Standardized Service Contract principles can be applied by introducing a separate utility service that provides centralized data access to the Databases A, B, and C, and exposes a standardized service contract that can be used by Service A . This will prevent Service A from direct dependencies on the shared databases in case any of them are replaced in the future. By following this approach, the Legacy Wrapper pattern is effectively applied via the introduction of the new utility service.

D. None of the above.

Correct Answer: B

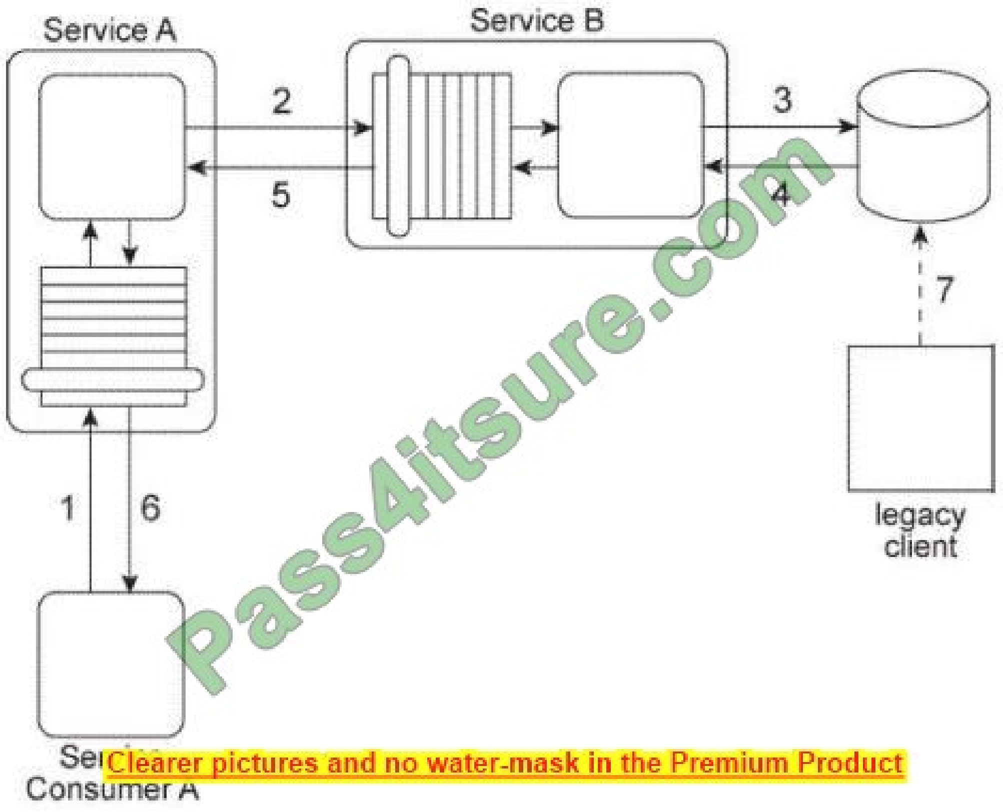
QUESTION 3

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.



Because Service A and Service B are not always available at the same times, messages are getting lost and several invocation attempts by Service Consumer A fail. What steps can be taken to solve this problem?

A. 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 .

B. The Asynchronous Queuing pattern can be applied so that a messaging queue is established between Service A and Service B . This way, messages are never lost due to the unavailability of Service A or Service B . The Service Agent pattern can be further applied to establish a service agent that makes a log entry and issues a notification when re-transmission attempts by the messaging queue exceeds a pre-determined quantity.

C. The Asynchronous Queuing pattern can be applied so that a messaging queue is established between Service Consumer A and Service A. This way, messages are never lost due to the unavailability of Service A or Service B. The Service Agent pattern can be further applied to establish a service agent that makes a log entry each time a runtime exception occurs.

D. None of the above.

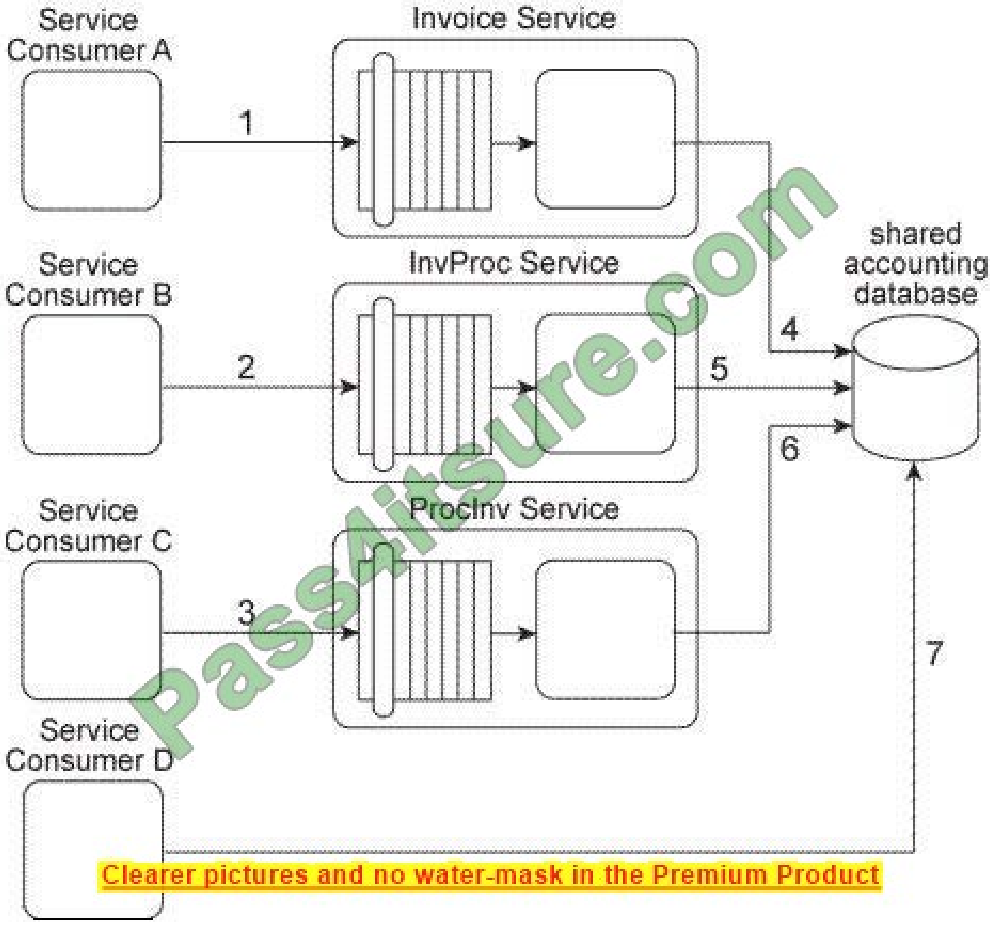
Correct Answer: A

QUESTION 4

Our service inventory contains the following three services that provide invoice-related data access capabilities: Invoice, InvProc, and Proclnv. These services were created at different times by different project teams and were not required to comply to any design standards. Therefore each of these services has a different data model for representing invoice data.

Currently each of these three services has one service consumer: Service Consumer A accesses the Invoice service(1). Service Consumer B (2) accesses the InvProc service, and Service Consumer C (3) accesses the Proclnv service. Each service consumer invokes a data access capability of an invoice-related service, requiring that service to interact with the shared accounting database that is used by all invoice-related services (4, 5, 6).

Additionally, Service Consumer D was designed to access invoice data from the shared accounting database directly (7), (Within the context of this architecture. Service Consumer D is labeled as a service consumer because it is accessing a resource that is related to the illustrated service architectures.)



Assuming that the Invoice service, InvProc service, and ProcInv service are part of the same service inventory, what steps would be required to fully apply the Official Endpoint pattern?

A. One of the invoice-related services needs to be chosen as the official service providing invoice data access capabilities. Service Consumers A, B, and C then need to be redesigned to only access the chosen invoice-related service. Because Service Consumer D does not rely on an invoice-related service, it is not affected by the Official Endpoint pattern and can continue to access the accounting database directly. The Service Abstraction principle can be further applied to hide the existence of the shared accounting database and other implementation details from current and future service consumers.

B. One of the invoice-related services needs to be chosen as the official service providing invoice data access capabilities. Service Consumers A, B, and C then need to be redesigned to only access the chosen invoice-related service. Service Consumer D also needs to be redesigned to not access the shared accounting database directly, but to also perform its data access by interacting with the official invoice-related service. The Service Abstraction principle can be further applied to hide the existence of the shared accounting database and other implementation details from current and future service consumers.

C. Because Service Consumers A, B, and C are already carrying out their data access via published contracts, they are not affected by the Official Endpoint pattern. Service Consumer D needs to be redesigned to not access the shared accounting database directly, but to perform its data access by interacting with the official invoice-related service. The Service Abstraction principle can be further applied to hide the existence of the shared accounting database and other implementation details from current and future service consumers.

D. None of the above.

Correct Answer: B

QUESTION 5

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. Additionally, a review of the logic of both services has revealed that some of the business rules used by the Client and Vendor services are actually the same. What steps can be taken to improve performance and reduce redundant business rule logic?

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, thereby reducing the redundancy of business rules logic. The Redundant Implementation pattern can then be applied to establish a scalable Rules service that is capable of supporting concurrent access from many service consumers.

B. The Redundant Implementation pattern can be applied to the Client and Vendor services, thereby establishing duplicate service 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 to isolate business rules logic into a central and reusable Rules service. Additionally, the Service Abstraction principle can be applied to hide the implementation details of new the Rules service.

D. None of the above.

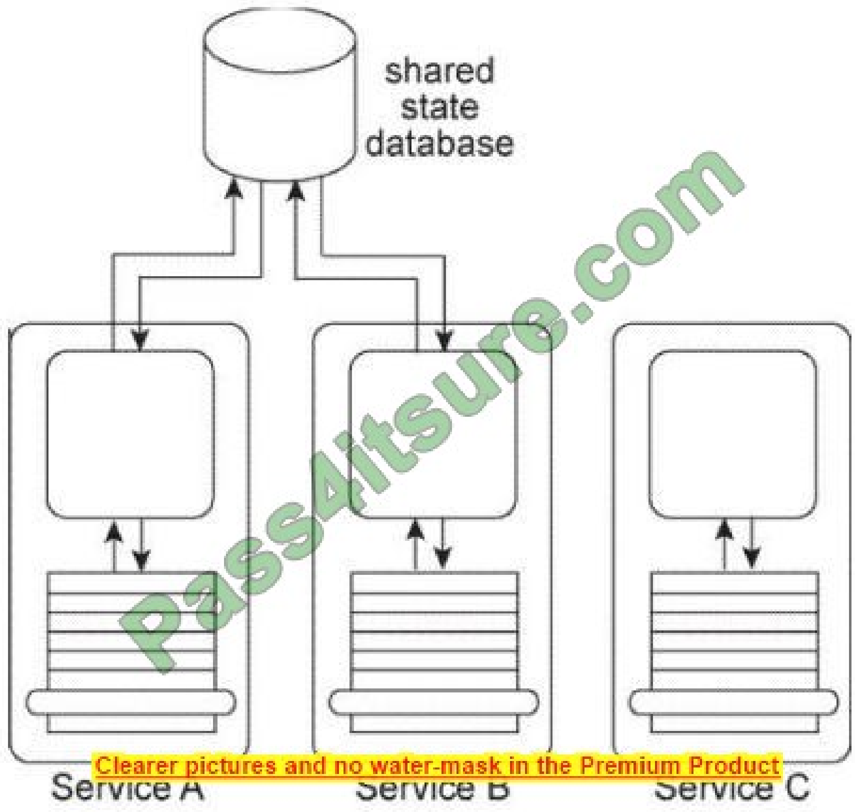
Correct Answer: A

QUESTION 6

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 7

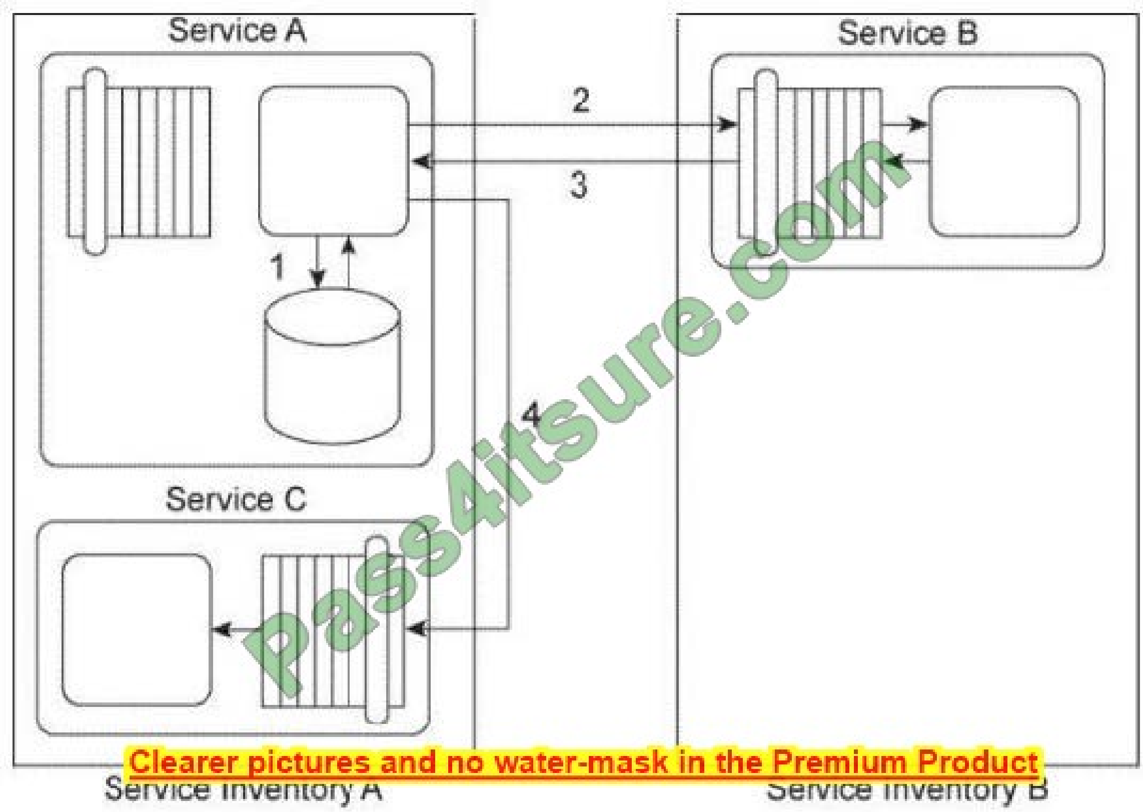
Service A is a task service that sends Service B a message (2) requesting that Service B return data back to Service A in a response message (3). Depending on the response received. Service A may be required to send a message to

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.

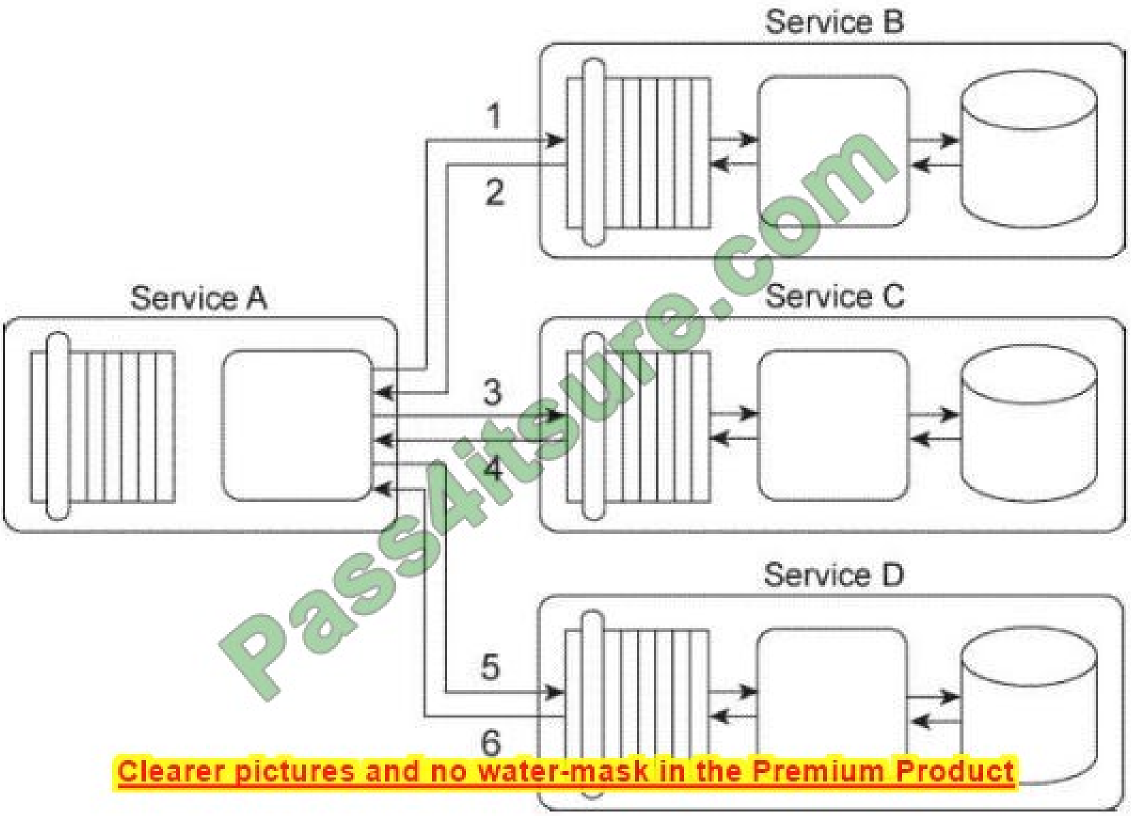
D. None of the above.

Correct Answer: B

QUESTION 8

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 given a requirement that all database updates must either be completed successfully or not at all. This means that if any of the three response messages received by Service A contain a failure code, all of the updates carried out until that point must be reversed. Note that if Service A does not receive a response message back from Services B, C, or D, it must assume that a failure has occurred. How can this service composition architecture be changed to fulfill these requirements?

A. The Reliable Messaging pattern can be applied to guarantee the delivery of positive or negative acknowledgements. This way, Service A will always be informed of whether a failure condition has occurred with any of the database updates performed by Services B, C, and D. Furthermore, the Service Loose Coupling principle can be applied to ensure that the request and response messages exchanged by the services do not contain any implementation details that would indirectly couple Service A to any of the databases.

B. The Atomic Service Transaction pattern can be applied individually to Services B, C, and D so that each of these services performs its own database update within the scope of an atomic transaction. If anyone update fails, that change can be rolled back on that database. Furthermore, the Service Loose Coupling principle can be applied to ensure that Service A is kept out of the scope of the atomic transaction so that it is not negatively coupled to the proprietary database technologies that are required to enable the atomic transaction functionality.

C. The Compensating Service Transaction can be applied to Service A so that when any one response message containing a failure code is received by Service A, it can invoke exception handling logic that will log the failed database updates. The Service Loose Coupling principle can be further applied to ensure that Services B, C, or D are not indirectly coupled to the exception handling logic, especially if Service A requires additional access to Services B, C, or D in order to collect more information for logging purposes.

D. None of the above.

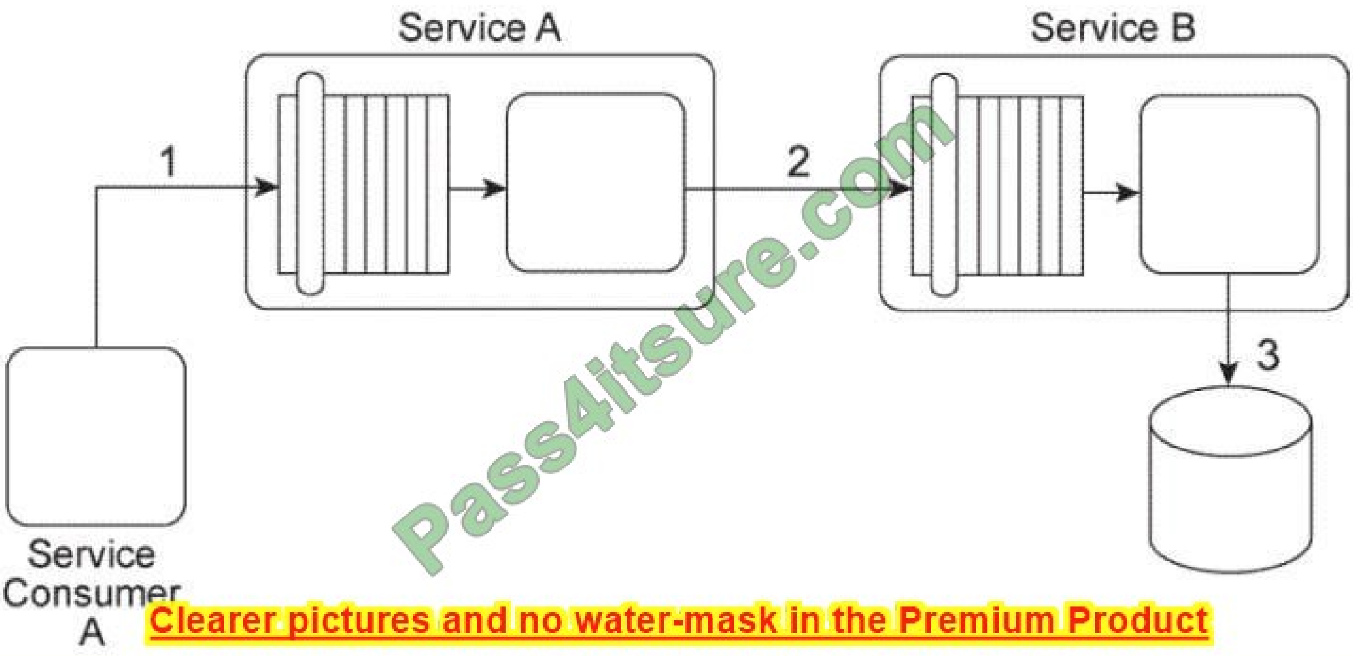
Correct Answer: D

QUESTION 9

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

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

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



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

A. Service Consumer A can be redesigned to use XML Schema B so that the SOAP message it sends is compliant with the service contract of Service A . The Data Model Transformation pattern can then be applied to transform the SOAP message sent by Service A so that it conforms to the XML Schema A used by Service B. The Standardized Service Contract principle must then be applied to Service B and Service Consumer A so that the invoice XML document is optimized to avoid unnecessary validation.

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

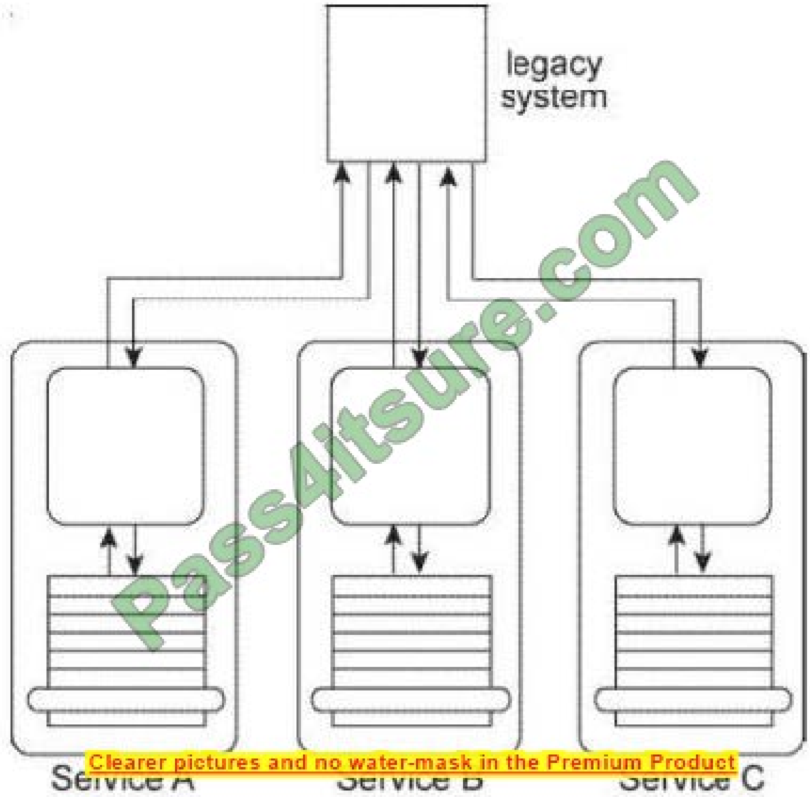
C. Service Consumer A can be redesigned to write the invoice document directly to the database. This reduces performance requirements by avoiding the involvement of Service A and Service B . It further supports the application of the Service Abstraction principle by ensuring that Service Consumer A hides the details of the data access logic required to write to the database.

D. None of the above.

Correct Answer: B

QUESTION 10

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



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 A, B, and C and any future services that are designed to rely upon it?

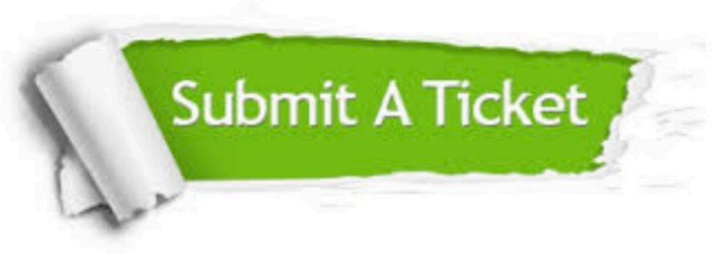
A. The Legacy Wrapper pattern can be applied together with the Standardized Service Contract principle to position a standardized service contract between the legacy system and any services that require access to it. This effectively establishes a new utility service dedicated to the encapsulation of the legacy system. When the legacy system is replaced, the utility service can keep its standardized service contract. To build the utility service, the Data Format Transformation pattern is applied to convert between the proprietary legacy system file format and the XML format used in the standardized service contract.

B. The Legacy Wrapper pattern can be applied together with the Official Endpoint pattern so that the Service A service contract is positioned as the sole access point for the legacy system. The Data Format Transformation pattern is applied to enable the conversion between the proprietary legacy system file format and the XML format used in the Service A service contract. Finally, the Contract Centralization pattern is applied so that Service A is forced to only access the legacy system via its published standardized service contract.

C. The Legacy Wrapper pattern can be applied together with the Data Format Transformation pattern and the Standardized Service Contract principle in order to establish an intermediate layer of standardized transformation logic that is positioned between the legacy system and Services A, B, and C. This way, if the legacy system is replaced, the services will not be affected because of the abstraction established by the standardized transformation layer.

D. None of the above.

Correct Answer: A


Submit A Ticket