Free Questions for S90.09 by dumpssheet

Shared by Castro on 12-12-2023

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

Service Consumer A sends Service A a message containing a business document (1). The business document is received by Component A, which keeps the business document in memory and forwards a copy to Component B (3). Component B first writes portions of the business document to Database A (4). Component B writes the entire business document to Database B and then uses some of the data values from the business document as query parameters to retrieve new data from Database B (5). Next, Component B returns the new data back to Component A (6), which merges it together with the original business document it has been keeping in memory and then writes the combined data to Database C (7). The Service A service capability invoked by Service Consumer A requires a synchronous request-response data exchange. Therefore, based on the outcome of the last database update, Service A returns a message with a success or failure code back to Service Consumer A (8). Databases A and B are shared and Database C is dedicated to the Service A service architecture.

There are several problems with this architecture: The business document that Component A is required to keep in memory (while it waits for Component B to complete its processing) can be very large. Especially when Service A is concurrently invoked by multiple service consumers, the amount of runtime resources it uses to keep this data in memory can decrease the overall performance of all service instances. Additionally, because Database A is a shared database that sometimes takes a long time to respond to Component B, Service A can take a long time to respond back to Service Consumer A . Currently, Service Consumer A will wait for a response for up to 30 seconds after which it will assume the request to Service A has failed and any subsequent response messages from Service A will be rejected. What steps can be taken to solve these problems?

Options:

A- The Service Statelessness principle can be applied together with the State Repository pattern in order to extend Database C so that it also becomes a state database allowing Component A to temporarily defer the business document data while it waits for a response from Component B . The Service Autonomy principle is applied together with the Legacy Wrapper pattern to isolate Database A so that it is encapsulated by a separate wrapper utility service. The Compensating Service Transaction pattern is applied so that if the response time of Service A exceeds 30 seconds, a notification is sent to a human administrator to raise awareness of the fact that the eventual response of Service A will be rejected by Service Consumer A .

B- The Service Statelessness principle can be applied together with the State Repository pattern in order to establish a state database that Component A can defer the business document data to while it waits for a response from Component B . The Service Autonomy principle can be applied together with the Service Data Replication pattern to establish a dedicated replicated database for Component B to access instead of the shared Database A . The Asynchronous Queuing pattern can be applied to establish a messaging queue between Service Consumer A and Service A so that Service Consumer A does not need to remain stateful while it waits for a response from Service A .

C- The Service Statelessness principle can be applied together with the State Repository pattern in order to establish a state database that Component A can defer the business document data to while it waits for a response from Component B . The Service Autonomy principle can be applied together with Service Abstraction principle, the Legacy Wrapper pattern, and the Service Facade pattern in order to isolate Database A so that it is encapsulated by a separate wrapper utility service and to hide the Database A implementation from Service A and to position a Facade component between Component B and the new wrapper service. This Facade component will be responsible for compensating the unpredictable behavior of Database A .

D- None of the above.

Answer:

B

Question 2

Question Type: MultipleChoice

Service Consumer A sends Service A a message containing a business document (1). The business document is received by Component A, which keeps the business document in memory and forwards a copy to Component B (3). Component B first writes portions of the business document to Database A (4). Component B writes the entire business document to Database B and then uses some of the data values from the business document as query parameters to retrieve new data from Database B (5). Next, Component B returns the new data back to Component A (6), which merges it together with the original business document it has been keeping in memory and then writes the combined data to Database C (7). The Service A service capability invoked by Service Consumer A requires a synchronous request-response data exchange. Therefore, based on the outcome of the last database update, Service A returns a message with a success or failure code back to Service Consumer A (8). Databases A and B are shared and Database C is dedicated to the Service A service architecture.

There are several problems with this architecture: First, the response time of Database A is often poor, resulting in Component B taking too much time to provide a response to Component A . This results in Component A consuming too many runtime resources while it holds the business document in memory and it also causes unreasonable delays in responding to Service Consumer A . Additionally, Database B is being replaced with a different database product that supports a proprietary file format. This will disable the current interaction between Component B and the new Database B . What steps can be taken to solve these problems?

Options:

A- The State Repository pattern is applied so that Component A can defer the business document data to a state database while it waits for a response from Component B . The Service Data Replication pattern is applied so that Component B can interact with a database that is replicated from the shared Database A . This will improve performance and reliability that will affect both Component A and Service Consumer A . Finally, the Legacy Wrapper pattern is applied so that Database B is wrapped in a standardized contract. This will establish a new wrapper utility service that will allow Database B to be replaced with a different database product without affecting Service A . Furthermore, the Data Format Transformation pattern can be applied within the new wrapper utility service to enable it to convert to and from the new proprietary file format.

B- The State Repository pattern is applied so that Component A can defer the business document data to a state database while it waits for a response from Component B . The Asynchronous Queuing pattern can be applied so that a messaging queue is established between Service Consumer A and Service A, thereby guaranteeing delivery and avoiding Service Consumer A from being tied up too long waiting for Service A to respond. Finally, the Data Format Transformation pattern can be applied to enable Component B to convert to and from the new proprietary file format introduced by the database product that is replacing Database B .

C- The Legacy Wrapper pattern is applied so that Database B is wrapped in a standardized contract. This will establish a new wrapper utility service that will allow Database B to be replaced with a different database product without affecting Service A . The Data Format Transformation pattern can be applied within the new wrapper utility service to enable it to convert to and from the new proprietary file format. The Service Data Replication pattern is applied so that Component B can interact with a database that is replicated from the shared Database B, regardless of what database product is used to replace Database B . The Service Abstraction principle can be further applied to hide the implementation details, including the changes mentioned in this solution, from Service Consumer A .

D- None of the above.

Answer:

A

Question 3

Question Type: MultipleChoice

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?

Options:

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 Faade components are added: one between Component A and each of the new wrapper utility services. This way, the Faade components can compensate for any change in behavior that may occur as a result of the separation. The Service Compos ability 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 Compos ability 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.

Answer:

B

Question 4

Question Type: **MultipleChoice**

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 state ful and consumes excessive memory. What steps can be taken to address these three problems?

Options:

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.

Answer:

C

Question 5

Question Type: **MultipleChoice**

You are told that in this service composition architecture, all four services are exchanging invoice related data in an XML format. The services in Service Inventory A are standardized to use a specific XML schema for invoice data. Design standards were not applied to the service contracts used in Service Inventory B, which means that each service uses a different XML schema for the same kind of data. Database A and Database B can only accept data in the Comma Separated Value (CSV) format and therefore cannot accept XML formatted data. What steps can be taken to enable the planned data exchange between these four services?

Options:

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

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

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

D- None of the above.

Answer:

A

Question 6

Question Type: **MultipleChoice**

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

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

Options:

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

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

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

D- None of the above.

Answer:

C

Question 7

Question Type: MultipleChoice

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?

Options:

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.

Answer:

C

Question 8

Question Type: MultipleChoice

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 despite the fact that duplicate implementations of Service A exist, performance is still poor at times. Also, you are informed that a new service capability will soon need to be added to Service A that will introduce functionality that will require access to a shared database that is used by many other clients and applications in the IT enterprise. This is expected to add further performance demands on Service A . How can this service architecture be changed to improve performance in preparation for the addition of the new service capability?

Options:

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

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

C- The Service Loose Coupling principle is applied together with the Standardized Service Contract principle to ensure that Service Consumer A is not indirectly coupled to the shared database after the new service capability is added to the service contract. The Legacy Wrapper pattern can be applied to establish a new utility service that will provide standardized data access service capabilities for the shared database. D . None of the above.

Answer:

B