DUMPSARENA

SOA Design & Architecture Lab

SOA S90.09

Version Demo

Total Demo Questions: 5

Total Premium Questions: 40

Buy Premium PDF

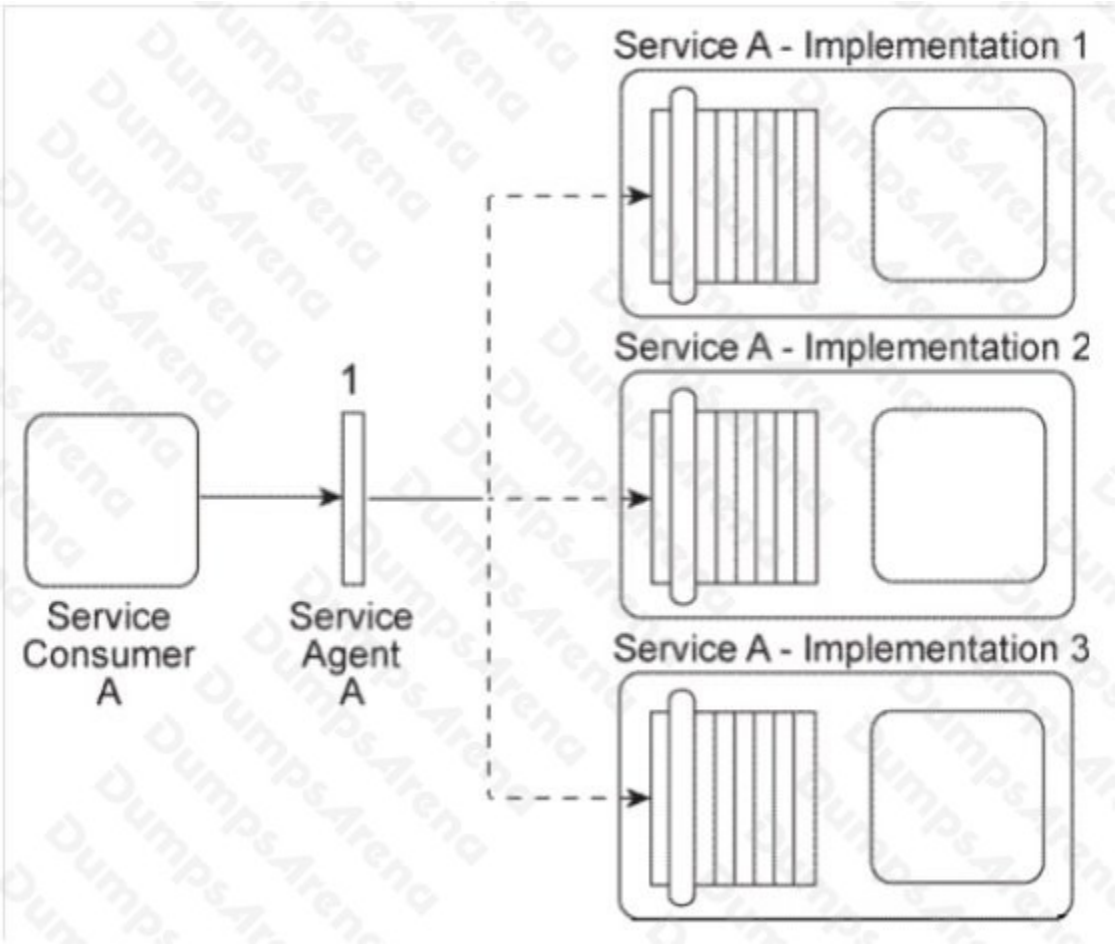
https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

QUESTION NO: 1

It has been confirmed that Policy A and Policy B are, in fact, the same policy and that the security credential check performed by Service Agent B also needs to be carried out on messages sent to Service B .



How can this service composition architecture be changed to reduce the redundancy of policy content and fulfill the new security requirement?

A. The Policy Centralization pattern can be applied so that Policy A and Policy B are combined into the same policy. The policy enforcement logic is removed from Service Agent C and Service Agent A is then used to enforce the policy for messages sent to Service A and Service B . Service Agent B can be used to perform the security credential check for Service A and Service B .

B. The Policy Centralization pattern can be applied so that Policy A and Policy B are combined into the same policy. The Service Agent pattern is then applied to introduce a new service agent (called Service Agent D) which carries out the validation and enforcement of Policy A and Policy Service Agent B can be moved so that it performs the security credential check for Service B, but not for Service A .

C. The Policy Centralization pattern can be applied so that Service Agent A is changed to enforce the policy for messages sent to Service A and Service B and to perform the security credential check for Service A and Service B .

D. None of the above.

ANSWER: A

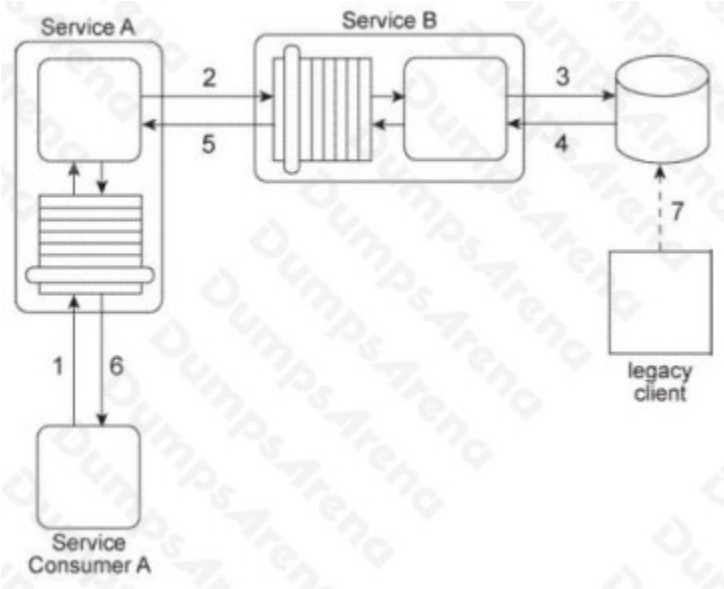
QUESTION NO: 2

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.

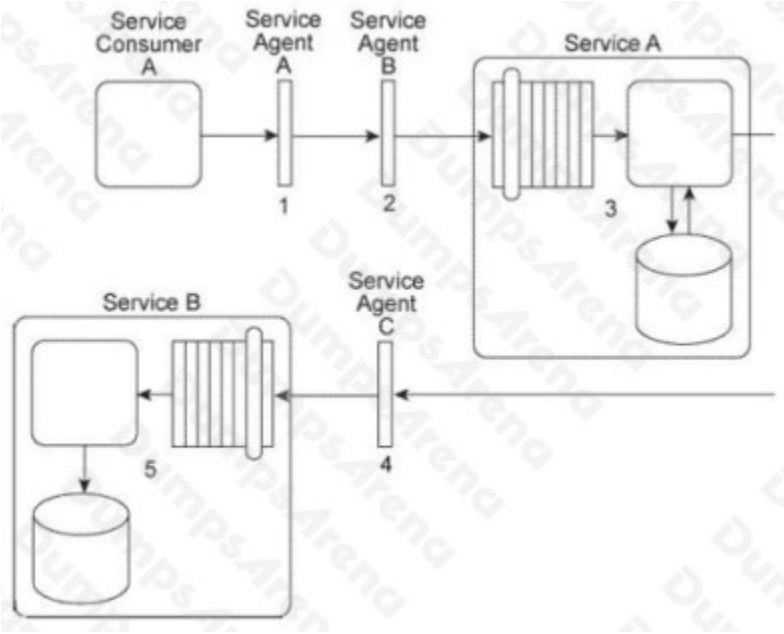
D. None of the above.

ANSWER: D

QUESTION NO: 3

Service Consumer A sends a message to Service A. Before the message arrives with Service A, it is intercepted by Service Agent A (1). which checks the message for compliance to Policy A that is required by Service A. If the message fails compliance, Service Agent A will not allow it to proceed and will instead write the message contents to a log. If the message does comply to the policy, it continues to be transmitted toward Service A, but before it arrives it is intercepted by Service Agent B (2), which validates the security credentials in the message header. If the security credential validation fails, the message is rejected and a runtime exception is raised. If the security credentials are validated, the message is sent to Service A.

Upon receiving the message, Service A retrieves a data value from a database and populates the message header with this data value (3) prior to forwarding the message to Service B. Before the message arrives at Service B. it is intercepted by Service Agent C (4) which checks the message for compliance with two policies: Policy B and Policy C. Policy B is identical to Policy A that was checked by Service Agent A. To check for compliance to Policy C. Service Agent C uses the data value added by Service A. If the message complies with both of the policies, it is forwarded to Service B (5), which stores the message contents in its own database.



You are told that Policy B and Policy C have changed. Also, in order to carry out the compliance check of Policy C, Service Agent C will now require a new data value from the Service B database. How can this service composition architecture be changed to fulfill these new requirements?

A. The Policy Centralization pattern can be applied so that only one service agent is used to enforce Policy A and Policy B. Service A is redesigned to first query Service B for the value required by Service Agent C to check the compliance of the updated Policy C. If the compliance check is successful, the message is sent to Service B .

B. The Policy Centralization pattern can be applied so that only one service agent is used to enforce Policy A and Policy Service Consumer A is redesigned to first query Service B for the value required by Service Agent C. This way, Service Consumer A can include this value in the message header prior to sending the message to Service A .

C. The Policy Centralization pattern can be applied so that only one service agent is used to enforce Policy A and Policy B. The policy enforcement logic for Policy C is removed from Service Agent C and instead embedded within the logic of Service B . This way, Service B can itself retrieve the value required to check compliance with Policy If the message received is not in compliance, Service B will reject it.

D. None of the above.

ANSWER: D

QUESTION NO: 4

Upon reviewing these requirements it becomes evident to you that the Orchestration compound pattern will need to be applied. However, there are additional requirements that need to be fulfilled. To build this service composition architecture, which patterns that is not associated with the Orchestration compound pattern need to also be applied? (Be sure to choose only those patterns that relate directly to the requirements described above. Patterns associated with the Orchestration compound pattern include both the required or core patterns that are part of the basic compound pattern and the optional patterns that can extend the basic compound pattern.)

A. Atomic Service Transaction

B. Compensating Service Transaction

C. Data Format Transformation

D. Data Model Transformation

E. Event-Driven Messaging

F. Intermediate Routing

G. Policy Centralization

H. Process Centralization

I. Protocol Bridging

J. Redundant Implementation

K. Reliable Messaging

L. Service Data Replication

M. State Repository

ANSWER: C L

QUESTION NO: 5

Service A is an orchestrated task service that is invoked by a separate composition initiator (1) and then sends a request message to Service C (2). Service C queries Database B to retrieve a large data record (3) and provides this data in a response message that is sent back to Service A. Service A temporarily stores this data in a central state database (4) and then sends a request message to Service D (5), which accesses a legacy system API to retrieve a data value (6). Service D then sends this data value in a response message back to Service A. The data in the state database is subsequently retrieved by Service A (7) and merged with the newly received data value. This combined data is written to Database A (8), which triggers an event that results in the invocation of Service B (9).

Service B is an orchestrated task service that sends a request message to Service D (10). which accesses a legacy system API to retrieve a data value (11) and then sends this data value in a response message back to Service B. Service B temporarily stores this data in a central state database (12) and then sends a request message to Service E (13), which performs a runtime calculation and then responds with the calculated data value back to Service B. The data in the state database is then retrieved by Service B (14) and merged with the calculated data value. Service B then uses the merged data to complete its business task.

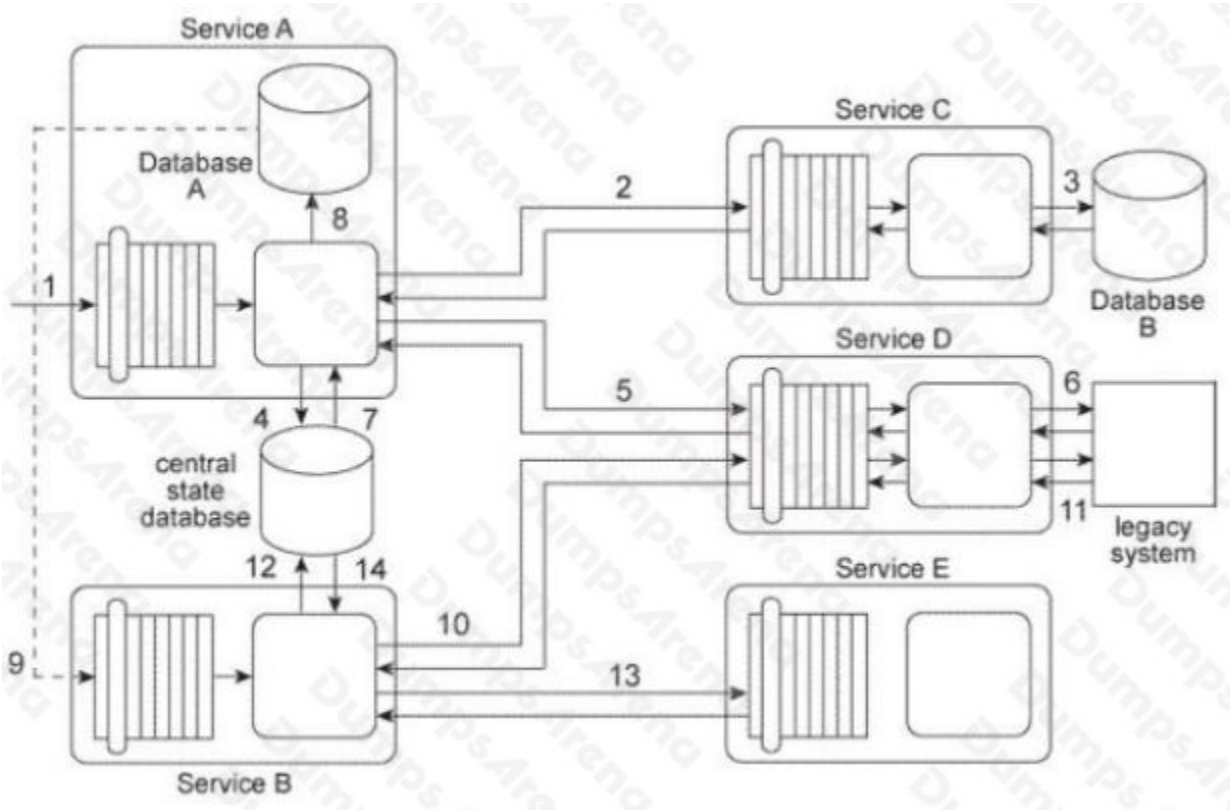
The following specific problems and requirements exist:

• Database B uses a proprietary data format that is not compliant with the XML format used by all of the services in this service composition architecture This incompatibility needs to be solved in order to enable the described service message exchanges.

• The service contract provided by Service D does not comply with the data model standards that were applied to the other services and therefore uses a different data model to represent the same type of data that is exchanged. This incompatibility needs to be solved in order to enable communication with Service D.

• Database B is a shared database that can be accessed by other services and applications within the IT enterprise, which causes unpredictable runtime performance. This performance problem needs to be solved in order to make the runtime behavior of Service C more predictable.

• For performance and maintenance reasons, Service A and Service B need to be deployed in the same physical environment where they can share a common state database.



Upon reviewing these requirements it becomes evident to you that the Enterprise Service Bus compound pattern will need to be applied. However, there are additional requirements that need to be fulfilled. To build this service composition architecture, which patterns that is not associated with the Enterprise Service Bus compound pattern need to also be applied? (Be sure to choose only those patterns that relate directly to the requirements described above. Patterns associated with the Enterprise Service Bus compound pattern include both the required or core patterns that are part of the basic compound pattern and the optional patterns that can extend the basic compound pattern.)

A. Atomic Service Transaction

B. Compensating Service Transaction

C. Data Format Transformation

D. Data Model Transformation

E. Event-Driven Messaging

F. Intermediate Routing

G. Policy Centralization

H. Process Centralization

I. Protocol Bridging

J. Redundant Implementation

K. Reliable Messaging

L. Service Data Replication

M. State Repository

ANSWER: H L M