PassQueen
a.k.a dumpsqueen.com

Architecting Composite Applications and Services with TIBCO

Tibco TB0-118

Version Demo

Total Demo Questions: 13

Total Premium Questions: 136

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Volume A	61
Topic 2, Volume B	55
Total	136

QUESTION NO: 1

An insurance claim service must support policy holders submitting individual claims and hospitals submitting batches of claims. Policy holders expect responses while they wait. Hospitals do not require a response until the next day. What are the two most important considerations in deciding whether a single operation can service both consumer requirements? (Choose two.)

A. minimizing the number of operation

B. differing consumer response time contracts

C. differing exception handling requirements

D. use of a common data structure

ANSWER: B C

Explanation:

The key considerations are *differing consumer response time contracts* and *differing exception handling requirements*. A service operation's contract includes not only the request data but also the message-exchange pattern and the expected timing of a response. A policy-holder channel requires a synchronous, interactive response within a short and predictable interval, whereas a hospital batch channel can use asynchronous submission and deferred processing. Before exposing one operation for both, the architect must determine whether the operation can clearly and reliably support both response-time expectations without making the contract ambiguous for either consumer.

Exception behavior is equally important because the two channels need different fault semantics. An interactive claimant generally needs an immediately actionable validation or processing result. Batch submitters commonly need durable acceptance, per-claim error reporting, retry handling, reconciliation, and a later completion status. A single operation is appropriate only if it can preserve these distinct interaction and error-handling contracts through explicit correlation, acknowledgements, and reporting behavior. TIBCO BusinessWorks supports both request-response and one-way/asynchronous service interactions, making the selected messaging pattern and fault contract central design decisions. See the TIBCO ActiveMatrix BusinessWorks documentation and the W3C WSDL messaging-pattern specification.

QUESTION NO: 2

Which three statements are true about the lifecycles of service consumers and service providers? (Choose three.)

A. The two lifecycles can be related because any component could be both a service consumer and a service provider.

B. The service provider lifecycle must be complete before the service consumer lifecycle can begin.

C. Interactions between the lifecycle of a service consumer and the lifecycle of a service provider result in the creation of a service utilization contract.

D. The two lifecycles interact because the service consumer uses the service of the service provider.

ANSWER: A C D

Explanation:

Service consumer and provider lifecycles are distinct but necessarily connected in a service-oriented architecture. "The two lifecycles can be related because any component could be both a service consumer and a service provider." is true because a composite application component can invoke another service while simultaneously exposing functionality for use by other components. Its role is determined by the particular interaction, not by an exclusive, permanent classification.

"Interactions between the lifecycle of a service consumer and the lifecycle of a service provider result in the creation of a service utilization contract." is also true. A utilization contract expresses the agreement governing consumption of a provider's published service, including the service interface and the consumer's intended use. This connects consumer requirements with provider capabilities and supports controlled service reuse.

"The two lifecycles interact because the service consumer uses the service of the service provider." is true because service consumption is the operational point at which the independently managed consumer and provider lifecycles meet. TIBCO

composite applications use explicitly defined service interfaces and references to model these provider-consumer relationships. See the TIBCO ActiveMatrix Service Grid documentation and TIBCO BusinessWorks Container Edition documentation for TIBCO service and composite-application concepts.

QUESTION NO: 3

A TIBCO ActiveMatrix Service Bus-based order fulfillment application makes outbound requests to an inventory service. Based on the geography and quantity of the order, the request must be routed to one of a number of different specialized inventory service instances. What are two possible ways to route the requests in ActiveMatrix? (Choose two.)

A. configure the order fulfillment application to reference the different inventory service instances

B. use XPath to create conditional references from the order fulfillment composite to each inventory service instance

C. create a Mediation Flow containing route tasks and route cases

D. create a dynamic binding to the inventory service and use XPath to compute the routing

ANSWER: C D

Explanation:

Creating a Mediation Flow containing route tasks and route cases is a declarative ActiveMatrix Service Bus solution for content-based routing. A route task evaluates configured route cases against the message content and directs the request through the appropriate branch. The geography and quantity values can therefore be used as conditions that select the specialized inventory endpoint required for that order. This keeps routing logic in the mediation layer, where message inspection, transformation, and service-routing concerns are intended to reside.

Creating a dynamic binding to the inventory service and using XPath to compute the routing is also appropriate when the target service address or endpoint must be determined at runtime. XPath can evaluate values in the outgoing request and construct or select the destination dynamically, allowing the same service invocation to be directed to different inventory instances without statically fixing one endpoint. This is particularly useful where routing targets are determined from message data and may change independently of the composite's static wiring. These approaches reflect ActiveMatrix Service Bus support for mediation-based routing and dynamic endpoint resolution. See the TIBCO ActiveMatrix Service Bus documentation and the TIBCO ActiveMatrix Service Bus product overview.

QUESTION NO: 4

Which coordination pattern (from the perspective of a source system) is most appropriate for keeping a push-based cache of values current?

A. In-Out

B. Out-In

C. In-Only

D. Out-Only

ANSWER: D

Explanation:

Out-Only is the appropriate coordination pattern because the source system's responsibility in a push-based cache design is to publish changed values, or change notifications, to the cache without waiting for a business response. When a source record is created, updated, or deleted, the source emits a one-way outbound message so that the cache can apply the corresponding change and remain current. This supports asynchronous propagation, reduces coupling between the source application and the cache implementation, and avoids making the source transaction depend on immediate cache processing or availability.

From the source system's perspective, the interaction therefore has an outbound direction and no required return message: it is "out" because the source initiates delivery, and "only" because no response is part of the message-exchange contract. This is the natural model for event-driven cache refresh, particularly where updates may be queued, retried, or processed independently by the receiving cache service. TIBCO composite applications commonly use messaging and service interactions to decouple producers from downstream consumers; the one-way exchange semantics align with this approach. See the WSDL 2.0 specification for one-way message-exchange concepts and the TIBCO ActiveMatrix BusinessWorks documentation for TIBCO integration and messaging documentation.

QUESTION NO: 5

You have a composite with a single SOAP/HTTP service and a component providing the service. Deploying the composite on a single node does not provide sufficient capacity to handle the load. What should you do?

A. use a mediation component to distribute the load

B. deploy the SOAP/HTTP service on one node and the component on a different node

C. deploy multiple instances of the component on different nodes

D. add more nodes to the system

ANSWER: C

Explanation:

deploy multiple instances of the component on different nodes is the appropriate scaling approach. In an ActiveMatrix composite, the service is implemented by the component, so the processing capacity that handles incoming service requests is increased by creating additional runtime instances of that implementation. Placing those instances on separate nodes distributes execution across the available infrastructure and provides horizontal scale for the composite without changing the service contract exposed to consumers.

This deployment pattern also improves resilience: if a node hosting one component instance becomes unavailable, other deployed instances can continue processing requests, subject to the configured runtime and binding behavior. Capacity planning therefore consists not merely of adding machines, but of configuring the component implementation to run with sufficient instances and assigning those instances across nodes in the ActiveMatrix environment. The SOAP/HTTP service remains the consumer-facing endpoint while the runtime routes work to the available component instances. TIBCO describes ActiveMatrix Service Grid as a platform for deploying and managing distributed service applications; its documentation and product resources are available at TIBCO ActiveMatrix Service Grid documentation and TIBCO ActiveMatrix Service Grid product information.

QUESTION NO: 6

What is the root element of a TIBCO ActiveMatrix Service Bus design?

A. SCA component

B. SCA composite

C. mediation flow

D. Spring implementation type

ANSWER: B

Explanation:

The correct answer is **SCA composite**. TIBCO ActiveMatrix Service Bus is based on Service Component Architecture (SCA), where an application or service assembly is modeled as a composite. A composite is the top-level design artifact: it provides the containing boundary for the application's components, services, references, properties, and the wires that

connect participating components. Consequently, it is the root from which the service-oriented design is defined and assembled for deployment.

Within this composite, developers can include implementations such as mediation logic, Java or Spring-based components, and other SCA component types. The composite also exposes services to consumers and declares references to external services, making it the central unit for both assembly and runtime configuration. This organization lets ActiveMatrix Service Bus represent an integration solution as a coherent, deployable SCA application rather than as an isolated implementation artifact. The OASIS SCA Assembly Model describes a composite as the assembly construct that groups components, services, references, and wires: OASIS SCA Assembly Model Specification. TIBCO product documentation is available through the TIBCO ActiveMatrix Service Bus documentation page.

QUESTION NO: 7

Which three statements are true about database access? (Choose three.)

A. TIBCO ActiveMatrix Adapter for Database can update a database.

B. TIBCO ActiveMatrix BusinessWorks can install triggers in the database and be notified of database changes.

C. TIBCO ActiveMatrix Adapter for Database can install triggers in the database and be notified of database changes.

D. TIBCO ActiveMatrix BusinessWorks can update a database.

ANSWER: A C D

Explanation:

TIBCO ActiveMatrix Adapter for Database can update a database because it provides database services that perform SQL operations, including insert, update, delete, and query processing. This lets an integration application interact with database tables through configured adapter services rather than requiring every database operation to be implemented directly in application code.

TIBCO ActiveMatrix Adapter for Database can install triggers in the database and be notified of database changes through its publication capability. The adapter uses database-side mechanisms, such as triggers and its associated polling or notification design, to detect relevant changes and publish the resulting events to messaging consumers. This makes it suitable for integrating database changes with event-driven enterprise systems.

TIBCO ActiveMatrix BusinessWorks can update a database through its JDBC activities. A BusinessWorks process can establish a JDBC connection and execute SQL update statements or stored procedures as part of a process flow, allowing process orchestration to create, modify, or remove relational data. See the TIBCO ActiveMatrix BusinessWorks documentation and the TIBCO ActiveMatrix Adapter for Database documentation for product documentation and configuration guidance.

QUESTION NO: 9

Which three factors should be considered when deciding whether business functionality might be best implemented as a Web Service? (Choose three.)

A. distribution of potential service components and potential service clients

B. estimated number of lines of code of the potential service implementation

C. the platform on which the underlying implementation will run

D. whether the invocation overhead is significant compared to the actual work performed

E. performance requirements of the potential service consumers

ANSWER: A D E

Explanation:

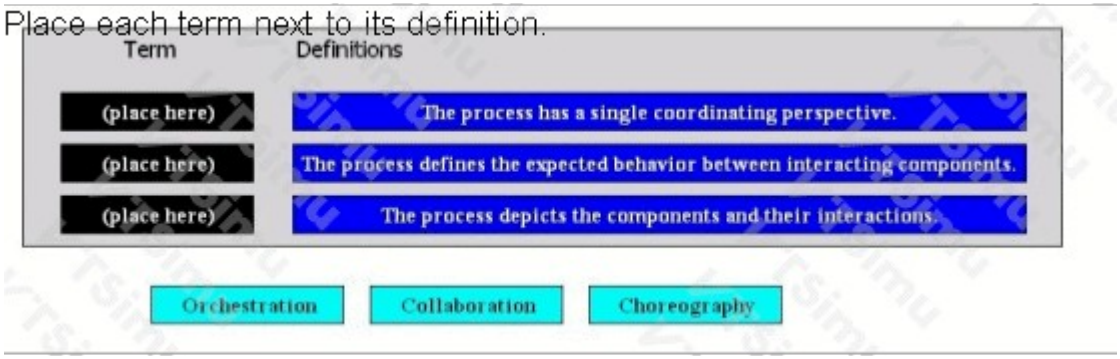
Web-service suitability is primarily an architectural decision about location, interaction cost, and consumer needs. “distribution of potential service components and potential service clients” is essential because services are intended to expose functionality across process, application, organizational, or network boundaries. A remotely accessible service interface can decouple clients from the implementation and enable interoperability where consumers and providers are distributed.

“whether the invocation overhead is significant compared to the actual work performed” is equally important. A Web Service invocation introduces message creation, serialization, transport, security processing, and request handling. The business operation should perform enough useful work to justify those costs; otherwise, a local API or a coarser-grained service operation may be the better design.

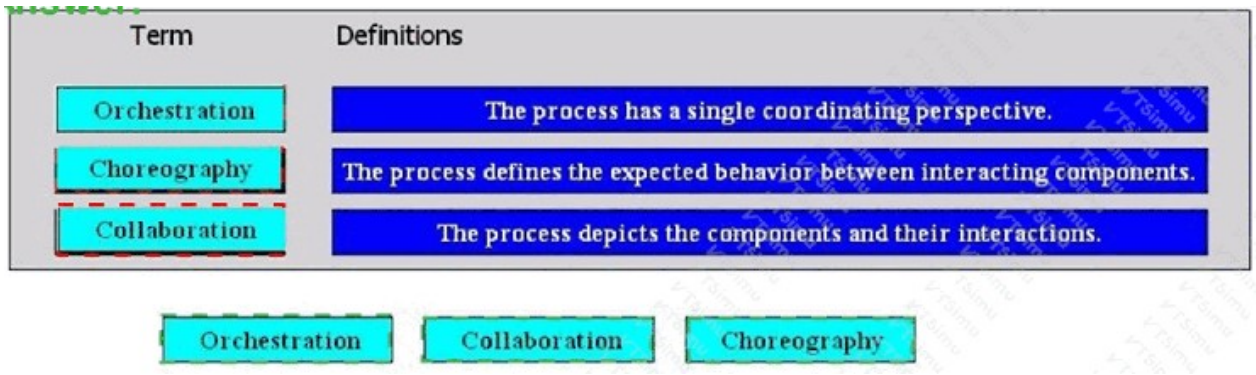
“performance requirements of the potential service consumers” must also guide the decision. Consumer response-time, throughput, availability, and payload requirements determine whether synchronous service calls are appropriate and influence interface granularity, binding selection, and whether asynchronous messaging is needed. TIBCO’s BusinessWorks documentation describes service-oriented integration capabilities and supported service bindings, while its architecture materials emphasize designing integrations around service contracts and runtime requirements. See the TIBCO ActiveMatrix BusinessWorks documentation and TIBCO SOA resources.

QUESTION NO: 11 - (DRAG DROP)

DRAG DROP



ANSWER:



Explanation:

The correct mapping distinguishes three complementary ways of describing a business process or composite application interaction. **Orchestration** is the process view with a single coordinating perspective. A central process definition controls the sequence of activities and invokes, receives from, or coordinates the participating services. This is the familiar executable-process viewpoint used when one application or process engine owns the flow of work.

Choreography defines the expected behavior between interacting components. Rather than expressing one participant’s internal control flow, it establishes the interaction protocol: which messages or actions are expected between participants and the order or conditions under which those interactions occur. This makes choreography useful as a shared behavioral contract across independent services or organizations.

Collaboration depicts the components and their interactions. It provides the broader interaction picture by showing the participants involved and the communication or message exchanges connecting them. In BPMN terminology, a collaboration commonly presents pools, participants, and message flows, making the relationship structure visible at a diagram level.

These concepts are aligned with BPMN's formal distinction between process models, choreography models, and collaboration models. The OMG BPMN 2.0 specification describes collaboration as the interaction of participants and identifies choreography as a way to define expected participant behavior. For a practical overview of BPMN collaboration and choreography diagrams, see Visual Paradigm's BPMN collaboration guide.

QUESTION NO: 13

When wrapping a backend system, what does a Web Service always provide that the native system interface does not?

A. standardized service operation semantics

B. an HTTP-based interface

C. a common data model

D. access independent of implementation technology

ANSWER: D

Explanation:

access independent of implementation technology is correct because a Web Service wrapper exposes a backend capability through a platform-neutral service contract rather than requiring consumers to use the backend system's proprietary API, programming language bindings, transport conventions, or runtime environment. The consumer interacts with the published service description and exchanged messages, while the wrapper is responsible for translating those interactions to the native backend interface. This separation is the key architectural value of service enablement: a legacy application, packaged application, database-oriented system, or custom component can be consumed through a consistent interoperable boundary without exposing its implementation details. In SOAP-based services, the WSDL contract formally describes service operations, messages, and bindings, allowing clients developed with different technologies to communicate with the service. The World Wide Web Consortium describes WSDL as an XML language for describing network services and their operations: WSDL 2.0. Likewise, the SOAP specification defines a messaging framework designed for decentralized, distributed environments: SOAP Version 1.2, Part 1. The wrapper may use HTTP and may transform data or standardize operations, but the essential guarantee supplied by the service boundary is technology-independent access.