Practice Exam: Salesforce Certified Platform Integration Architect

This practice paper follows the historical arc of the Salesforce Certified Platform Integration Architect credential, a Salesforce integration certification that tests architectural reasoning rather than product trivia. Early Force.com integrations gave way, particularly after Salesforce acquired MuleSoft in 2017, to API-led orchestration and governed event-driven designs. The pivotal development was the system-process–experience model, which made contracts, resilience, and data authority explicit.

2006–2013: Salesforce API Integration Foundations

Early Salesforce API integration established interfaces, ownership, and transaction semantics as architectural concerns. SOAP preceded REST, while direct database access yielded to contract-based services.

Questions 1–2: Authority and Idempotency

1. Which statement best defines a system of record for an enterprise data domain?

- A. The authoritative source whose values govern downstream copies.
- B. The first system in which the data is captured.
- C. The principal application used by end users.
- D. The system performing the greatest number of integrations.

Answer: A

A **system of record** is authoritative for a defined data domain and determines the accepted value when copies differ. Option B identifies an *ingestion point*, while C and D describe *usage characteristics* that do not confer authority.

2. An order request is delivered twice after a network timeout. Which property makes the second delivery safe?

- A. Atomicity across all participating systems.
- B. Idempotency through a stable key and duplicate detection.
- C. Eventual consistency without reconciliation.
- D. Encryption in transit.

Answer: B

Idempotency ensures that repeated requests produce the same business result as the original request. Atomicity concerns *all-or-nothing transaction behaviour* and does not itself prevent duplicate processing.

2014–2017: Salesforce MuleSoft Integration and the Salesforce Certified Platform Integration Architect

API-led design matured as MuleSoft became central to Salesforce's integration strategy, culminating in the 2017 acquisition. This was the central turning point: reusable API layers replaced unmanaged point-to-point dependencies with explicit contracts.

Questions 3–5: API-Led Orchestration and Delivery

3. Which MuleSoft API-led layer orchestrates a business transaction across multiple system APIs?

- A. System API.
- B. Process API.
- C. Experience API.
- D. Object API.

Answer: B

The **process layer** composes system APIs into reusable business transactions; experience APIs present those results to particular channels. An experience API normally consumes process APIs, so treating it as the cross-system orchestrator *misstates the layering*.

4. A MuleSoft flow receives an HTTP 503 while processing an ERP request. What is the most appropriate handling?

- A. Bounded retry with backoff, followed by error handling.
- B. Immediate unlimited retries.
- C. Commit partial data and suppress the error.
- D. Convert every non-2xx response into success.

Answer: A

HTTP 503 normally indicates a temporary failure, so **bounded backoff** preserves a recovery path without overwhelming the service Unlimited immediate retries amplify an outage and may exhaust integration resources.

5. An ERP must be notified after a Salesforce record is approved without delaying the approving user's transaction. Which mechanism best fits?

- A. Synchronous Apex callout.
- B. Outbound message.
- C. REST request from a Visualforce page.
- D. Salesforce Batch Apex.

Answer: B

An **outbound message** queues an asynchronous call to an external service, avoiding the latency of a synchronous request. An Apex callout blocks the transaction and may roll back Salesforce work when the external service is unavailable.

2018–2021: Salesforce Data Integration at Scale

Bulk API 2.0 and event technologies shifted emphasis from request-response exchanges toward high-volume and near-real-time integration. Salesforce data integration consequently became a problem of throughput, change capture, and authoritative ownership.

Questions 6–8: Bulk Movement and Event-Driven Consistency

6. Ten million records must be loaded into Salesforce, where throughput matters more than immediate visibility. Which interface is most appropriate?

- A. Bulk API 2.0.
- B. Streaming API.
- C. Apex REST callouts.
- D. Metadata API deployment.

Answer: A

Bulk API 2.0 supports asynchronous, high-volume ingestion and avoids the overhead of request-by-request processing. Streaming API delivers changes rather than bulk-loading records, while Apex callouts are constrained by request and governor limits.

7. A subscriber needs near-real-time notifications of standard Account changes, including updates made by other systems. Which capability is the best starting point?

- A. Change Data Capture.
- B. Bulk API.
- C. Apex trigger.
- D. Scheduled Apex.

Answer: A

Change Data Capture publishes standard-object changes to the event bus, including updates made outside the subscriber's code. An Apex trigger can also publish events, but it consumes platform resources and is less direct for broad change capture.

8. ERP and Salesforce both permit customer telephone numbers to be edited. Which rule most reliably prevents divergent records?

- A. Apply last-writer-wins in both systems.
- B. Synchronize edits from both systems without validation.
- C. Assign one system authority and reconcile exceptions.
- D. Compare all records every second.

Answer: C

Explicit authority defines which value is valid, while reconciliation detects and repairs divergence. Uncontrolled bidirectional updates can oscillate, and last-writer-wins makes the result dependent on timing rather than policy.

2022–Present: Secure and Resilient Orchestration

In this era, the Salesforce integration architect must treat identity, transactional reliability, and fault containment as central architectural concerns. Secure integration no longer ends at authentication, nor reliable integration at a successful request.

Questions 9–12: Identity and Failure Containment

9. How should an Apex integration avoid embedding an external service credential in source code?

- A. Store it in a Named Credential referenced by Apex.
- B. Store it in a Custom Metadata record.
- C. Place it in a Static Resource.
- D. Concatenate it into a request header.

Answer: A

A **Named Credential** keeps authentication material outside source code and supports secure authenticated requests. Custom Metadata can hold configuration, but it is *not the dedicated mechanism* for secret storage and lifecycle management.

10. A MuleSoft integration connects to Salesforce server-to-server without an interactive user login. Which Salesforce-supported OAuth flow is the strongest choice?

- A. OAuth 2.0 implicit flow.
- B. OAuth 2.0 JWT bearer flow.
- C. OAuth username-password flow.
- D. Authorization Code flow with PKCE.

Answer: B

JWT bearer flow authenticates a server-to-server integration through a digital certificate and integration user, avoiding a stored user password. Authorization Code with PKCE suits user-facing clients but depends on delegated user consent, which is absent in this scenario.

11. After creating a Salesforce order, a service must publish a corresponding event without risking event loss. Which pattern addresses the failure window?

- A. Synchronous callout before the transaction commits.
- B. Transactional outbox followed by asynchronous publication.
- C. Unrelated platform event in a later transaction.
- D. Recursive Apex trigger.

Answer: B

Writing an **outbox record** in the same transaction as the order makes the order and event intent atomic; a relay publishes it afterwards. A pre-commit callout can expose uncommitted data or publish an event for work that subsequently rolls back.

12. A Salesforce process must coordinate an ERP, payment provider, and partner API while tolerating partial outages. Which architecture is strongest?

- A. One synchronous controller chain with a shared transaction.

- B. Direct endpoint-to-endpoint communication forming a mesh.
- C. MuleSoft process layer using queues, retries, idempotency, and compensation.
- D. Bidirectional nightly CSV exchange.

Answer: C

An **orchestration layer** isolates participants, enforces stable contracts, and contains failures through controlled retries and compensating actions. A synchronous controller chain creates a distributed transaction and a large blast radius, while direct endpoint meshes amplify coupling.

Conclusion: The Salesforce Certified Platform Integration Architect Mindset

Success on the Salesforce Certified Platform Integration Architect exam depends on reasoning across ownership, contracts, data movement, identity, and failure. The historical shift toward API-led orchestration explains why reusable APIs, MuleSoft patterns, ever capture, and resilient messaging recur throughout this paper.

Frequently Asked Questions

How should I use this paper?

Complete it without notes, review every explanation, and revisit any topic that required guesswork.

What should I revise first?

Prioritise API selection, data ownership, MuleSoft layering, authentication, and resilient failure handling.

Are these official exam questions?

No. These are original practice questions designed to rehearse the architectural reasoning assessed by the certification.

Salesforce

PLAT-ARCH-204 Exam

Salesforce Certified Platform Integration Architect

Thank you for Downloading PLAT-ARCH-204 exam PDF Demo

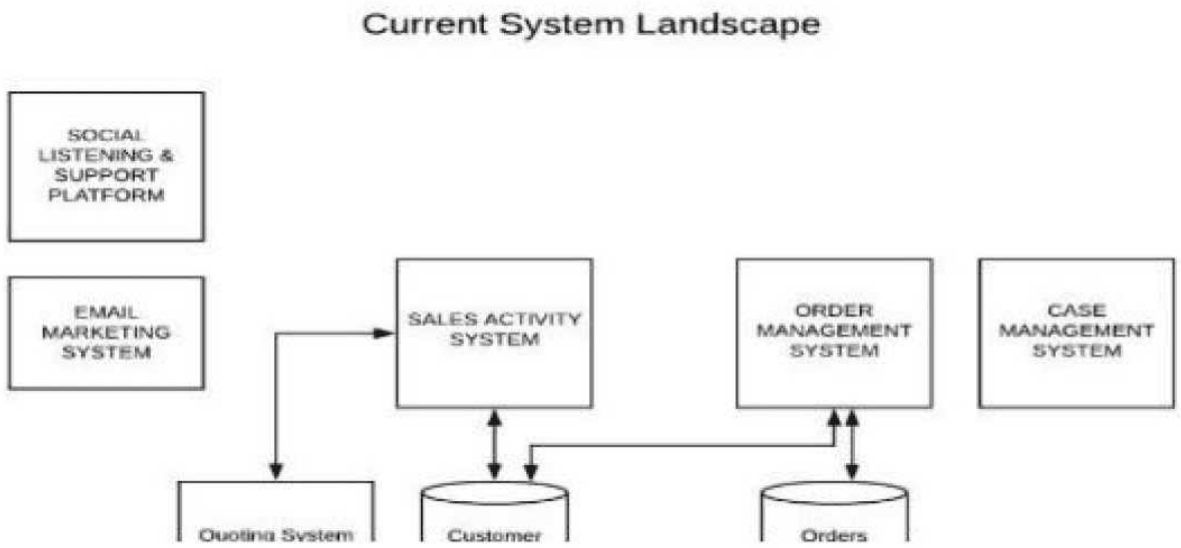
You can also try our PLAT-ARCH-204 Full Version Download

https://www.certkillers.net/Exam/PLAT-ARCH-204

Version: 23.0

Question: 1

A large business-to-consumer (B2C) customer is planning to implement Salesforce CRM to become a customer-centric enterprise. Below is the B2C customer's current system landscape diagram.



The goals for implementing Salesforce include:

Develop a 360-degree view of the customer.

Leverage Salesforce capabilities for marketing, sales, and service processes.

Reuse Enterprise capabilities built for quoting and order management processes.

Which three systems from the current system landscape can be retired with the implementation of Salesforce?

A. Order Management, Case Management, and Email Marketing

B. Sales Activity, Order Management, and Case Management

C. Email Marketing, Sales Activity, and Case Management

Answer: C

Explanation:

Comprehensive and Detailed 250 to 350 words of Explanation From Salesforce Pl14atform Integration Architect documents: In the rol15e of a Salesforce Platform Integration Architect, evaluating a legacy landscape requires a clinical mapping of current system functions against Salesforce's native capabilities, while strictly adhering to the "Constraints and Goals" provided by the business. The objective here is to maximize the ROI of the Salesforce implementation by consolidating redundant systems into the core platform.

According to Goal 2, the business intends to utilize Salesforce for Marketing, Sales, and Service processes. Salesforce is architected to handle these three domains through its core clouds: Marketing Cloud (replacing the legacy Email Marketing System), Sales Cloud (replacing the Sales Activity System), and Service Cloud (replacing the Case Management System). By consolidating these three specific functions into Salesforce, the organization achieves Goal 1, which is the creation of a 360-degree view of the customer. When these activities occur on a single platform, the data is unified, eliminating the silos that existed in the previous landscape.

However, the architect must also respect the technical constraints defined in Goal 3, which explicitly states the need to reuse enterprise capabilities built for quoting and order management. In architectural design, this designates the "Quoting System" and the "Order Management System" as systems of record that must remain in the future-state landscape. These systems likely contain complex, proprietary logic or are tightly coupled with back-end ERP systems like SAP Business Suite, making them "non-negotiable" for retirement at this stage.

Therefore, because Email Marketing, Sales Activity, and Case Management map directly to Salesforce's primary strengths and are not excluded by the "reuse" requirement, they are the three systems that should be retired. This strategic retirement simplifies the integration architecture, allowing the architect to focus on building robust integration patterns (such as Request-Reply or Fire-and-Forget) between Salesforce and the remaining Quoting and Order Management systems.

Question: 2

An integration architect has built a solution using REST API, updating Account, Contact, and other related information. The data volumes have increased, resulting in higher API calls consumed, and some days the limits are exceeded. A decision was made to decrease the number of API calls using bulk updates. The customer prefers to continue using REST API to avoid architecture changes. Which REST API composite resources should the integration architect use to allow up to 200 records in one API call?

A. Batch

B. SObject Tree

C. Composite

Answer: B

Explanation:

When designing high-volume integrations, the Salesforce Platform Integration Architect must distinguish between standard REST resources and "Composite" resources to optimize API consumption. The Salesforce REST API provides several composite resources to group multiple operations into a single call, thereby reducing the overhead of multiple HTTP requests and helping to stay within daily API limits.

According to Salesforce documentation on Composite Resources, the sObject Tree resource (/services/data/vXX.X/composite/tree/) is specifically designed to handle multiple records in a single request. While it is primarily marketed for creating complex hierarchies (parent-child relationships), it has a unique limit that allows for up to 200 records to be processed in a single call. These records can even be unrelated records of the same type. This is a significant advantage over the standard Batch and Composite resources.

The Composite resource and the Batch resource both have a much lower limit of 25 subrequests per call. While each subrequest in a Batch call could technically be a collection operation, the question specifically asks for the resource that natively supports the "200 records" threshold preferred for bulk-style updates within the REST framework. By utilizing the sObject Tree resource, the architect can bundle 200 record updates into a single transaction, effectively reducing API consumption by a factor of 200 compared to individual REST calls. This aligns with the requirement to avoid major architectural changes (like switching to the Bulk API 2.0) while solving the immediate problem of exceeding daily governor limits. In the context of the Integration Architect exam, understanding these specific payload limits is crucial for selecting the most efficient "Request-Reply" or "Data Synchronization" pattern.

Question: 3

A business requires automating the check and updating of the phone number type classification (mobile vs. landline) for all incoming calls delivered to its phone sales agents. The following conditions exist:

At peak, the call center can receive up to 100,000 calls per day.

The phone number type classification is a service provided by an external service API.

Business is flexible with timing and frequency to check and update the records (throughout the night or every 6-12 hours is sufficient).

A Remote-Call-In pattern and/or Batch Synchronization (Replication via ETL: System -> Salesforce) are determined to work with a middleware hosted on customer premise. In order to implement these patterns and mechanisms, which component should an integration architect recommend?

A. ConnectedApp configured in Salesforce to authenticate the middleware

B. An API Gateway that authenticates requests from Salesforce into the middleware (ETL/ESB)

C. Remote Site Settings configured in Salesforce to authenticate the middleware

Answer: A

Explanation:

In this scenario, the architecture involves a Remote-Call-In pattern or Batch Synchronization, where an external system (the middleware or ETL tool) initiates communication with Salesforce to update records. For any external system to securely access Salesforce APIs and perform these updates, it must be authenticated and authorized.

The Connected App is the foundational framework that allows an external application to integrate with Salesforce using APIs and standard protocols, such as OAuth 2.0 and SAML. By configuring a Connected App, the architect can define which permissions (Scopes) the middleware has, such as the ability to access data via the REST or Bulk API. This is the correct choice because the middleware needs to "log in" to Salesforce to push the phone classification data back into the Account or Contact records.

Option B, an API Gateway, is typically used to manage and secure requests going out of an organization to external services, or to provide a facade for on-premise APIs; it does not handle the inbound authentication into Salesforce itself. Option C, Remote Site Settings, is a configuration used solely to permit Salesforce to make outbound calls to a specific external URL (for example, if Salesforce were calling the phone classification service directly via Apex).

Given that the business is flexible with timing (allowing for nightly or 12-hour syncs) and handles 100,000 calls, a Batch Synchronization pattern via an ETL tool is highly efficient. The ETL tool will authenticate against the Connected App using a secure OAuth flow (such as the JWT Bearer Flow for server-to-server integration), retrieve the new phone numbers, call the external classification API, and then bulk-update the Salesforce records. This setup ensures a secure, scalable, and manageable integration that respects Salesforce's security architecture while meeting the high-volume data requirements of the call center.

Question: 4

A company captures orders and needs to send them to the Order fulfillment system. The user is not

required to have confirmation from the Order fulfillment system. Which system constraint question should be considered when designing an integration to send orders from Salesforce to a fulfillment system?

A. What latency is acceptable for orders to reach the fulfillment system?

B. Can the fulfillment system implement a contract-first Outbound Messaging interface?

C. Which system will validate order shipping addresses?

Answer: A

Explanation:

When designing an integration where the user does not require immediate confirmation, the architect is moving away from a synchronous "Request-Reply" pattern toward an asynchronous "Fire-and-Forget"16 or "Batch Processing" pattern. In such scenarios, the most critical architectural constraint is defining the latency requirements.

Latency dictates the technical choice of the integration tool. If the fulfillment system needs the order within seconds of creation to begin a high-speed picking process, the architect might choose Salesforce Outbound Messaging or an Apex Callout triggered by a Platform Event. If the system only needs to process orders once an hour or overnight, a Batch ETL process is more appropriate. Understanding the acceptable delay (latency) ensures that the solution meets business expectations without over-engineering for real-time performance where it isn't required.

While Option B (Outbound Messaging) is a valid technical capability, it is a specific solution rather than a high-level "system constraint question" that drives the initial design phase. Option C (Address Validation) is a functional requirement regarding data integrity, but it does not define the architectural framework of the integration as effectively as latency does. By identifying the latency threshold, the architect can determine if the integration should be near real-time, hourly, or daily, which in turn influences how the system handles error recovery, retries, and transaction volumes.

Question: 5

Northern Trail Outfitters requires an integration to be set up between one of its Salesforce orgs and an External Data Source using Salesforce Connect. The External Data Source supports Open Data Protocol. Which configuration should an integration architect recommend be implemented in order to secure requests coming from Salesforce?

A. Configure a certificate for OData connection.

B. Configure Special Compatibility for OData connection.

C. Configure Identity Type for OData connection.

Answer: C

Explanation:

Salesforce Connect is a powerful tool for data virtualization, allowing users to view and manage data in external systems (via OData) as if it were stored natively in Salesforce. However, a critical security decision during the setup of an External Data Source is determining the Identity Type.

The Identity Type determines how the external system authenticates the Salesforce user. There are two primary options:

Named Principal: Every Salesforce user accesses the external system using the same set of credentials. This is easier to maintain but provides less granular security tracking in the target system.

Per User: Each individual Salesforce user must provide their own credentials for the external system. This ensures that the data visible in Salesforce respects the user's specific permissions in the external source.

Configuring the Identity Type is the fundamental way an architect secures OData requests because it defines the authentication boundary between the platforms. While certificates (Option A) can be used for transport layer security, the "Identity Type" configuration is the specific Salesforce Connect setting that governs how a session is authorized. Option B (Special Compatibility) is a technical setting used to handle non-standard OData implementations and does not directly relate to securing the request. By recommending the correct Identity Type, the architect ensures that the integration adheres to the "Principle of Least Privilege," ensuring that users only see the external data they are authorized to access.

Thank You for trying PLAT-ARCH-204 PDF Demo

To try our PLAT-ARCH-204 Full Version Download visit link below

https://www.certkillers.net/Exam/PLAT-ARCH-204

Start Your PLAT-ARCH-204 Preparation

[Limited Time Offer] Use Coupon "CKNET" for Further discount on your purchase. Test your PLAT-ARCH-204 preparation with actual exam questions.