ORACLE

Design and Implementation of Siebel–WhatsApp Messaging Using Oracle Digital Assistant

This whitepaper provides a comprehensive overview of the Siebel–WhatsApp–Oracle Digital Assistant (ODA) integration architecture. It examines the messaging and push notification framework that connects Siebel CRM with WhatsApp via ODA, detailing webhook handling, end-to-end message flow, and deployment approaches that enable secure, scalable, and Oracle-compliant enterprise communication.

February, 2026, Version [1.0]
Copyright © 2026, Oracle and/or its affiliates
Public

Purpose statement

This document provides a solution approach and is for informational purposes only. It is intended solely to provide valuable insights and guidance on the subject matter while serving as a reference for stakeholders. This document does not constitute a binding agreement or official policy.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the solution described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Copyright © 2026, Oracle and/or its affiliates / Public

ORACLE

Table of contents

List of Figures

Copyright © 2026, Oracle and/or its affiliates / Public

Overview

Siebel CRM (Customer Relationship Management) is a comprehensive enterprise platform designed to manage and automate customer-centric business processes across sales, marketing, and customer service. Known for its robustness and flexibility, Siebel enables organizations to maintain a unified, 360-degree view of customer interactions. By supporting highly configurable business workflows and multiple deployment models, Siebel continues to serve the complex operational and integration requirements of large-scale enterprises across diverse industries.

Oracle Digital Assistant (ODA) is an AI-powered conversational platform that enables organizations to build, deploy, and manage intelligent digital assistants across multiple channels, including WhatsApp. ODA facilitates natural language interactions, orchestrates business logic, and integrates seamlessly with backend enterprise systems such as Siebel CRM. By leveraging ODA skills, events, and channel webhooks, enterprises can deliver automated, real-time, and contextual customer interactions while maintaining centralized control and governance.

The Siebel–WhatsApp–ODA integration enables bidirectional messaging and push notification capabilities by connecting Siebel CRM with WhatsApp through ODA-managed webhooks and events. This integration supports real-time message handling, event-driven notifications, and secure communication flows. By deploying the webhook service within the Siebel Tomcat environment, the solution simplifies architecture, reduces operational dependencies, and ensures compliance with Oracle-approved deployment and security standards.

Solution Overview

Optimized for enterprise messaging and event-driven communication, this integration seamlessly connects Siebel CRM with WhatsApp through Oracle Digital Assistant. The solution can be implemented using an optimized Java-based webhook service deployed within the Siebel Tomcat environment. It is designed for streamlined deployment and simplified operations, providing a unified integration layer that enables organizations to deliver real-time customer interactions and notifications while meeting enterprise security and compliance requirements.

This solution enables end-to-end WhatsApp messaging and push notification capabilities through the following key components and features:

- **Unified Webhook Service:** Handle inbound WhatsApp messages from Meta and outbound responses through a single Java service deployed within Siebel's Tomcat server.
- **ODA Skill Integration:** Route messages through Oracle Digital Assistant skills for intent handling, orchestration, and response generation.
- **Event-Driven Notifications:** Trigger WhatsApp push notifications using ODA Events based on business events and workflows in Siebel CRM.
- **Simplified Architecture:** Eliminate external VM or Node.js dependencies by consolidating messaging and notification handling within the Siebel environment.
- **Secure Communication Flow:** Leverage Oracle-approved deployment models and HTTPS-based communication for secure message exchange.

Copyright © 2026, Oracle and/or its affiliates / Public

Integration Architecture Overview

The architecture of the Siebel–WhatsApp–Oracle Digital Assistant (ODA) integration is thoughtfully designed around three core components: Siebel CRM, Oracle Digital Assistant, and WhatsApp (Meta). Siebel CRM serves as the enterprise system of record, enabling business events and triggers to initiate customer communications. Oracle Digital Assistant functions as the orchestration layer, processing requests, managing skills, and coordinating message flows, while WhatsApp acts as the customer-facing channel for real-time messaging and notifications.

A Java-based webhook service deployed within the Siebel Tomcat environment acts as the integration interface, securely handling inbound requests from Meta, forwarding messages to ODA, and returning responses back to WhatsApp users. The same service also facilitates outbound push notifications by invoking ODA Events, ensuring efficient routing through skill-associated channels. This architecture minimizes latency, reduces external dependencies, and provides a streamlined, secure, and scalable solution aligned with Oracle enterprise deployment standards.

Technical Deep Dive: Whatsapp – ODA – Siebel Integration

This section details the technical architecture and component interactions that enable real-time messaging and push notifications between WhatsApp users and Siebel CRM through Oracle Digital Assistant. At the core of this integration is a Java-based webhook service deployed within the Siebel Tomcat environment, which bridges WhatsApp (Meta) and ODA while allowing Siebel-triggered events to initiate outbound notifications in a secure and scalable manner.

Architecture Overview
The architecture is composed of three primary components: the WhatsApp (Meta) Platform, Oracle Digital Assistant, and the Siebel CRM Environment.

- **WhatsApp (Meta) Platform**: This layer represents the customer-facing communication channel. Users interact with a Business WhatsApp Account associated with an ODA skill. Incoming messages are delivered by Meta to the configured webhook endpoint, and outbound responses are routed back to users through the same channel, ensuring real-time, bidirectional communication.

- **Oracle Digital Assistant (ODA):** This layer acts as the conversational intelligence and orchestration engine. ODA receives inbound messages forwarded by the webhook service, processes them using configured skills, intents, and dialog flows, and generates appropriate responses. ODA also exposes an Events API, which enables event-driven outbound messaging for push notifications initiated by backend systems such as Siebel.

- **Siebel CRM Environment:** This layer includes the Siebel application and the embedded Java-based webhook service deployed within the Siebel Tomcat server. The service handles inbound webhook requests from Meta, forwards messages to ODA, and relays responses back to WhatsApp. Additionally, based on configured business triggers within Siebel, the same service invokes ODA Events to initiate outbound push notifications, repeating the message flow from ODA back to WhatsApp users. This consolidated deployment model reduces external dependencies while maintaining enterprise security and operational consistency.

Copyright © 2026, Oracle and/or its affiliates / Public

Technical Diagram

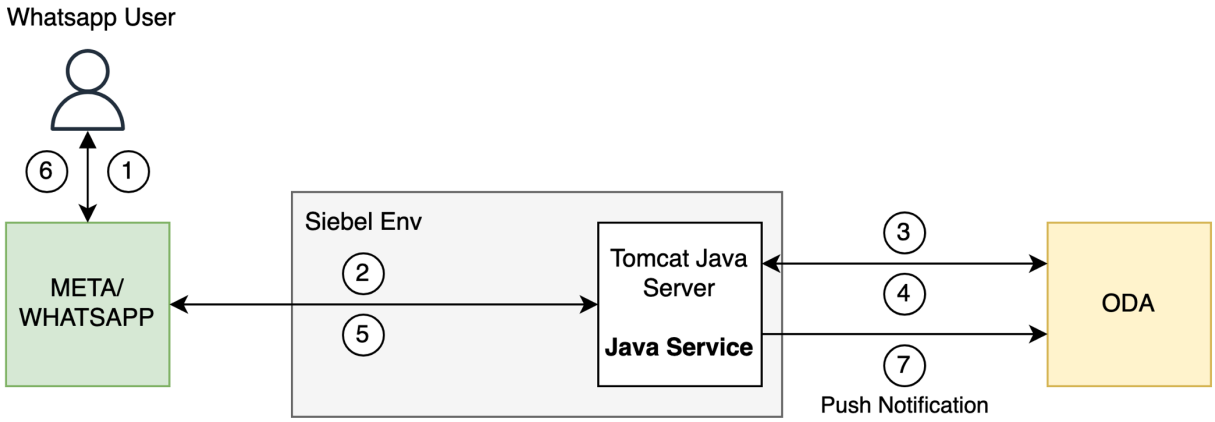

Figure 1: Architecture diagram

Steps

1. User sends a WhatsApp message to the Business WhatsApp Account associated with the ODA skill.
2. Meta forwards the message request to the webhook implemented as a Java service deployed in the Siebel Tomcat server.
3. The webhook service forwards the message to Oracle Digital Assistant (ODA).
4. ODA processes the request and returns the response to the webhook service.
5. The webhook service forwards the response to the WhatsApp (Meta) platform.
6. The user receives the response from the Business WhatsApp Account.
7. Based on configured business triggers, the same Java service initiates push notifications by invoking ODA Events, which repeat the flow from steps 4 through 6.

Copyright © 2026, Oracle and/or its affiliates / Public

Prerequisites

- Whatsapp – Meta Business Account
- Oracle Digital Assistant

Configuration

Configuring Meta Business Suit

1. Go to https://developers.facebook.com/ and Login using Facebook and create a Meta Business Account if not already created. Click on "My Apps" from the top Menu bar.

2. Click on Create App.

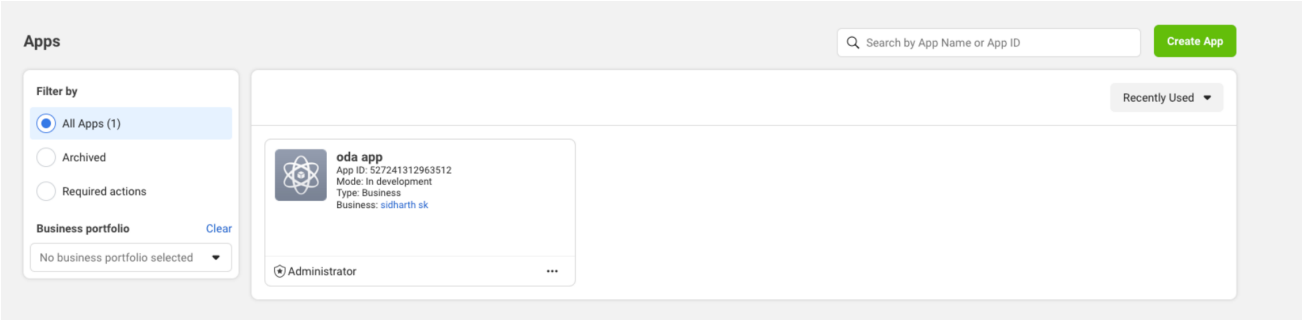

Figure 2: Creating Meta app

Copyright © 2026, Oracle and/or its affiliates / Public

3. Enter your Details and Choose "Other" option in Use Cases Tab and click Next

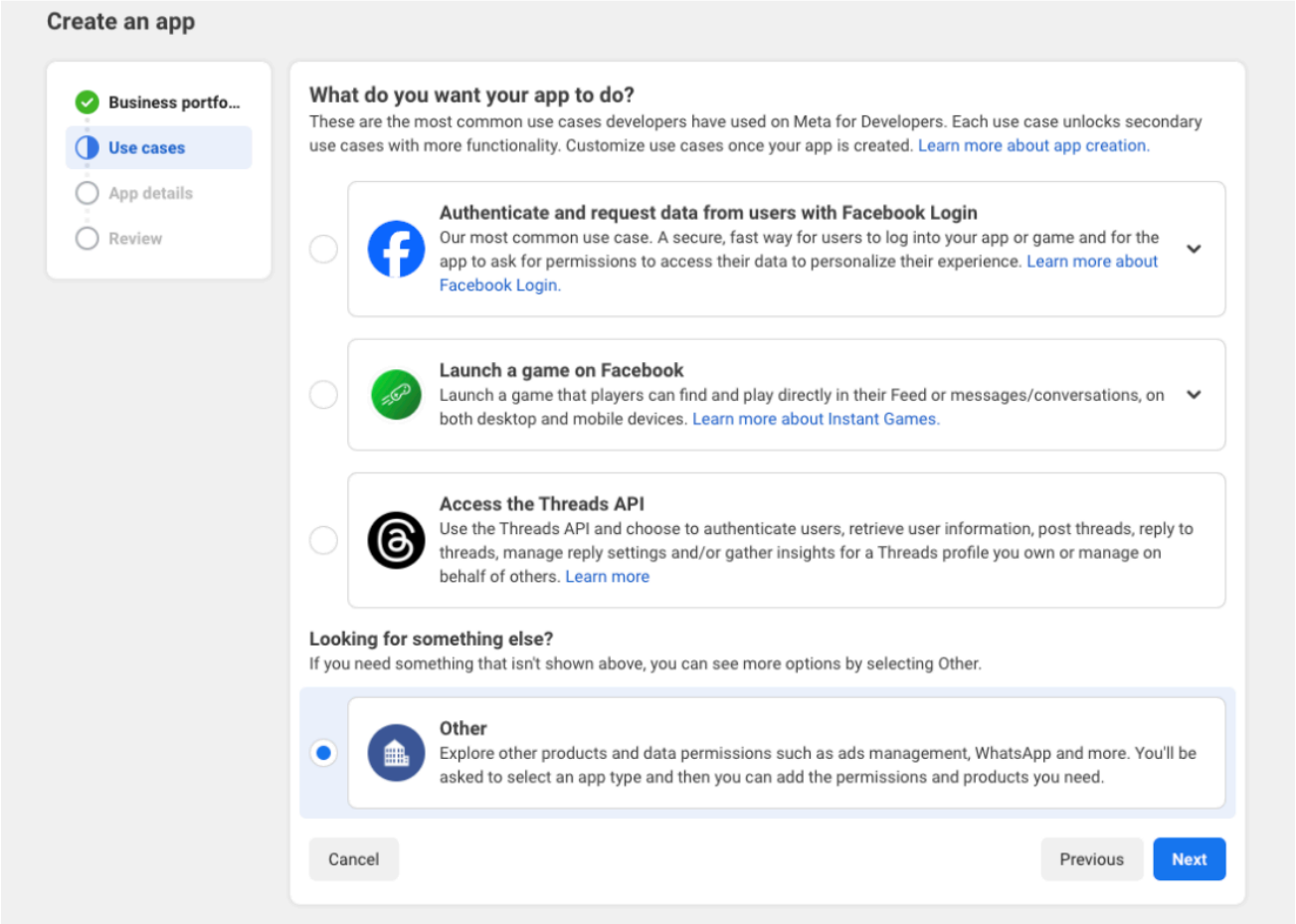

Figure 3: Creating Meta account

4. For the Type Choose "Business" and click Next, Enter your Details in the next Tab.

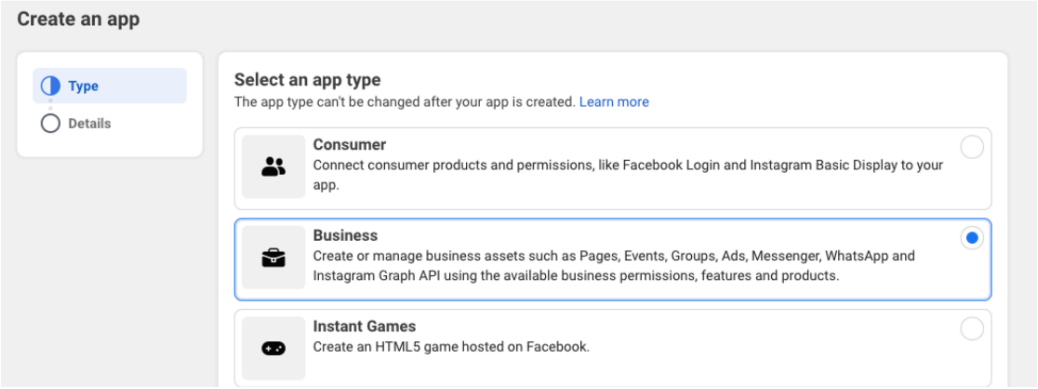

Figure 4: Setting Meta account Type

5. In the "Add products to your app" page Choose Whatsapp - integrate with Whatsapp.

Copyright © 2026, Oracle and/or its affiliates / Public

6. After landing on Quickstart page, Click on "Start using the API".

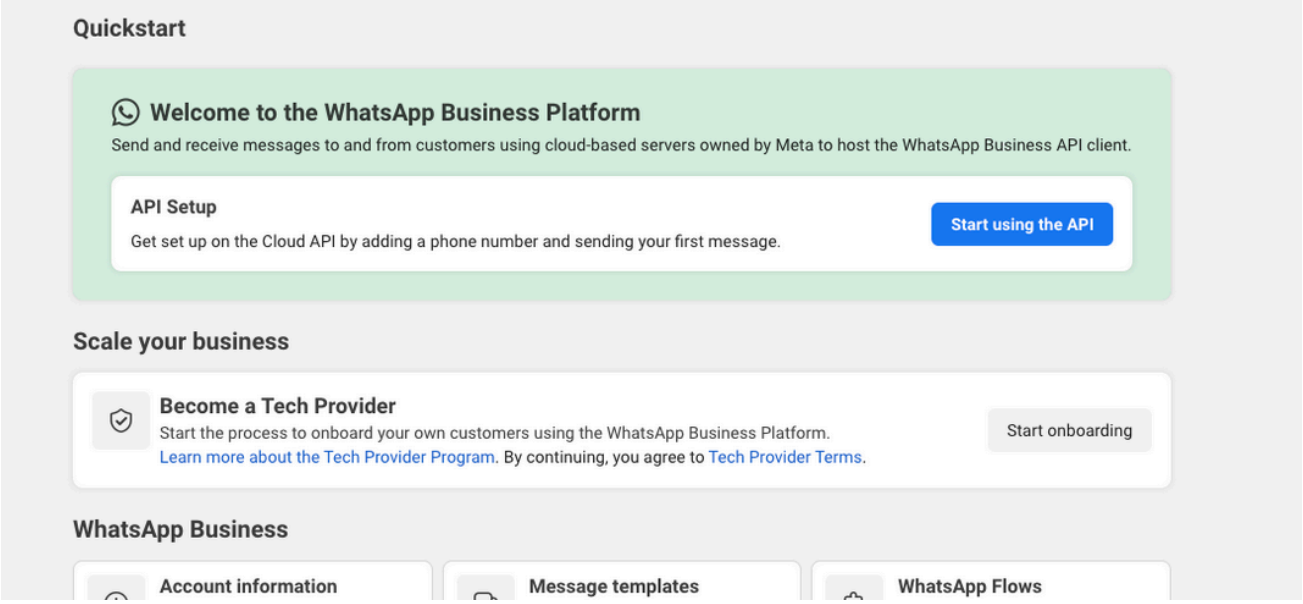

Figure 5: Meta Business API Configuration

7. Copy the **access Token**, this will be used for authorizing webhook requests. Copy the **Temporary phone number** which will be the Bot that receives and sends messages. You can test a flow by entering your own number in the "To" section by verifying the number via OTP Whatsapp Sends. After testing, you can add a phone number in the From field and verify it using a code which WhatsApp sends. This is actual business number which you want to use for messaging your customer. This number will get added to the from number on the get started page.

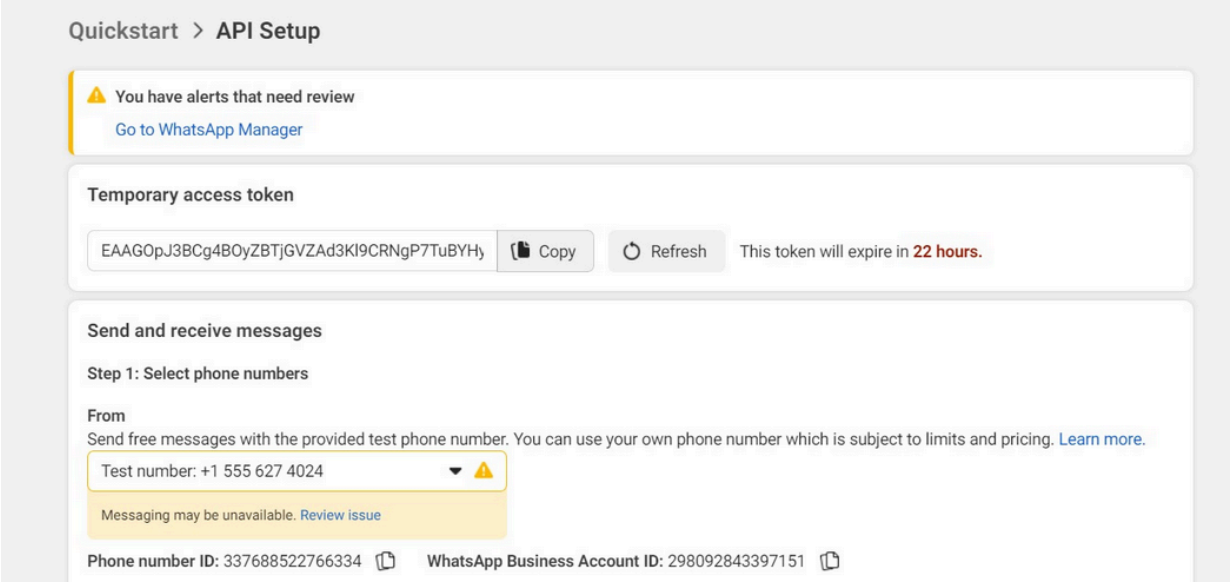

Figure 6: Meta API Setup

Copyright © 2026, Oracle and/or its affiliates / Public

Configuring ODA

1. Create a Skill with all the required Intents. For Demo purpose we created a Skill- **wpodaskill**.

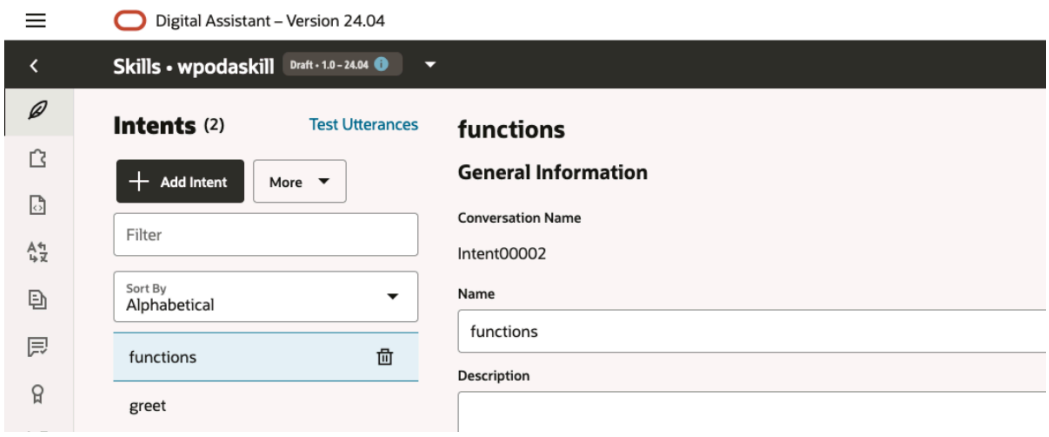

Figure 7: ODA Intents

2. In the Flow Designer create Flows for all the intents previously created.

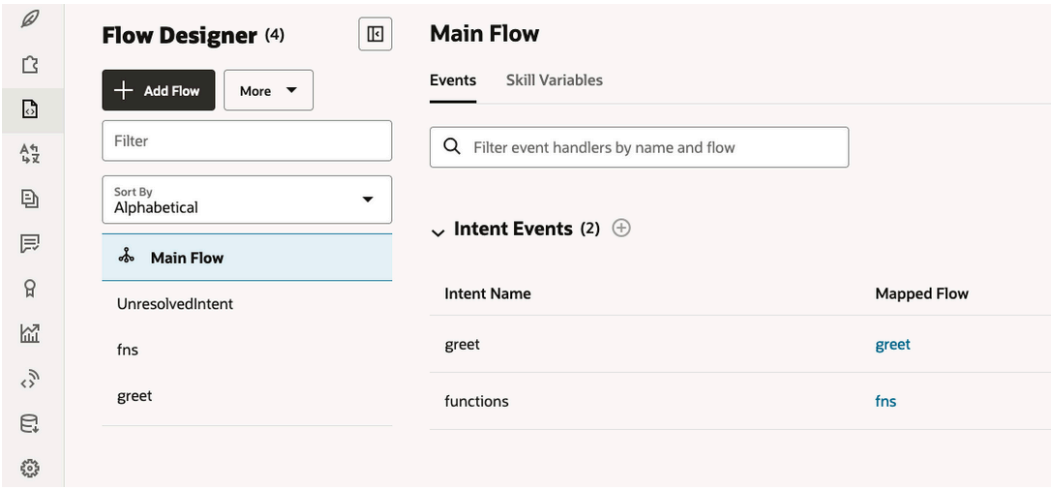

Figure 8: ODA Flow designer

3. Go to Channels from ODA Development Menu.

4. Click on "Add Channel" to Create a Channel and enter below details-
   Channel Type: Webhook
   Route To: Choose the skill created in step 1 (in the demo its **wpodaskill**)
   Payload Version: Conversation Model
   Outgoing Webhook URI: www.oracle.com (This is temporary value, we need to replace this with URL of the webhook server after deployment)
   Click on Create
5. Copy the **Secret Key** and **Webhook URL**
6. Enable the Channel

Copyright © 2026, Oracle and/or its affiliates / Public

ORACLE

Deploy Webhook in Siebel for ODA – Siebel Communication

1. **Build and Test Locally**

    - Build the Java application. Test the application on a local Tomcat server.
      Application archive: SiebelExternalServicePOC.zip
      (update Secret key and Webhook URL, TOKEN value (from Step 7 - Setting up Meta Business Suit). MYTOKEN can be any text value which we need to enter same in Webhook config page of WhatsApp as well.)

2. **Generate WAR File**

    - Generate the WAR file from the application source.

3. **Copy WAR to Remote Server**

    - Securely copy the WAR file to the target VM using SCP via corporate proxy.
    - Example (details masked):
    - scp -o ProxyCommand='nc -X connect -x <proxy-host>:<proxy-port> %h %p' \
    - -i ~/.ssh/id_rsa <app-name>.war <user>@<vm-ip>:/home/<user>/<path>

4. **Deploy WAR into Dockerized Tomcat**

    - Copy the WAR file into the Tomcat webapps directory inside the Docker container:
    - docker cp <app-name>.war <container-name>:/<tomcat-path>/webapps
    - Access the container shell if required: docker exec -it <container-name> bash

5. **Restart Tomcat Server**

    - Restart the Tomcat server (or the container) to load the new application.

6. **Test Using Postman**

    - Invoke the REST endpoint exposed by the application.

7. **Endpoint (example):**
   https://<host-ip>/<app-context>/siebel/eventPusher/pushEvent
   Sample Payload (masked):{
   "channel": "<ODA_EVENT_CHANNEL_URL>",
   "secret": "<SECRET_KEY>",
   "message": "sample message",
   "userid": "<USER_ID>"
   }
   Note Webhook_host_URL.

Copyright © 2026, Oracle and/or its affiliates / Public

ODA Certificate Import (SSL Handshake Issue)

Error:
Java application running on **Siebel Tomcat inside Docker** fails to invoke an HTTPS REST API with:

SSLHandshakeException: PKIX path building failed

1. **Download SSL Certificate**
    - Download the server .pem certificate from the target HTTPS endpoint.
2. **Convert Certificate**
3. openssl x509 -outform der -in cert.pem -out cert.crt
4. **Import into JVM Truststore**
    - Copy the certificate into the Docker container.
    - Import it into the JVM truststore used by Tomcat:
5. keytool -import -alias <alias> -file cert.crt \
6. -keystore <truststore-path> -storepass <password>
7. **Restart Server**
    - Restart Tomcat / container.

Updating values in ODA, Meta

1. Update Outgoing Webhook URI in ODA Channel (Step 4 - Setting up ODA): <Webhook_host_URL>/bot/message

Figure 9: ODA Outgoing URL

2. In Meta Business Suit, Go to "Configuration" Tab (From the left menu options under Whatsapp). Edit the values by entering Callback URL as: <Webhook_host_URL>/webhook and **Verify token** as the value provided in **MYTOKEN** earlier (Step 3- Setting up Webhook Node Server locally). Click Verify and save.

Copyright © 2026, Oracle and/or its affiliates / Public

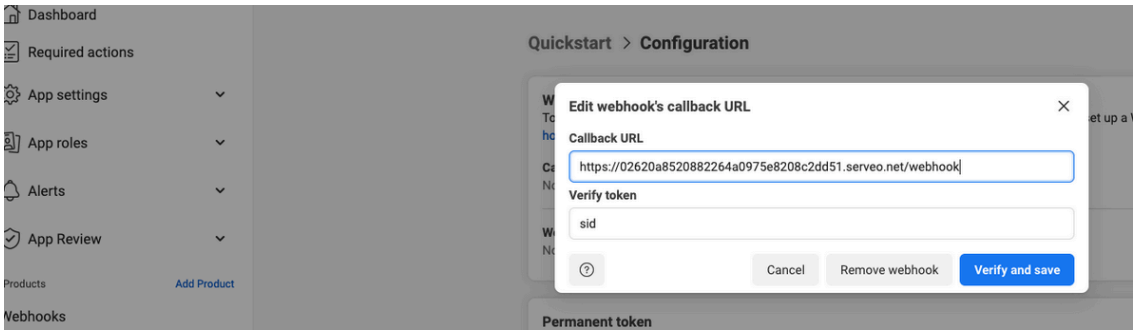

Figure 10: Callback URL - webhook

3. Once you have saved the webhook, click on manage and give the messaging permissions.

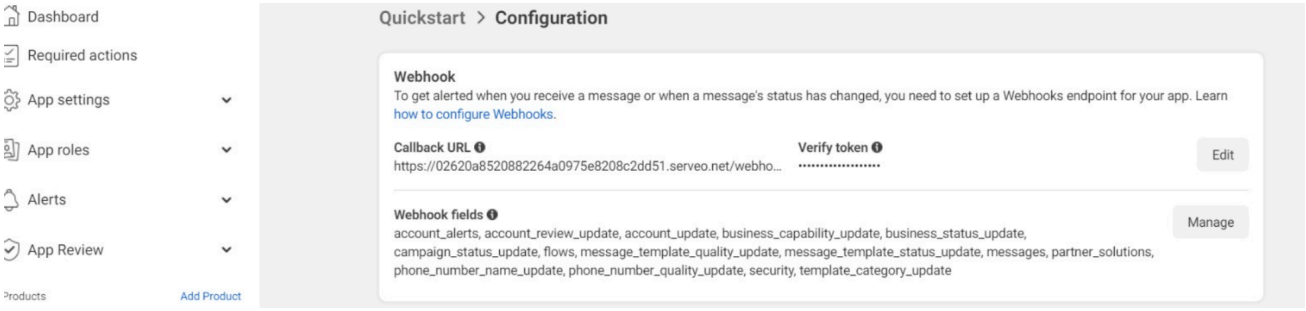

Figure 11: Meta settings - Webhook

4. Now Send a message to the Trial Number provided to you to run the flow. (Step 7 - Setting up Meta Business Suit)

Copyright © 2026, Oracle and/or its affiliates / Public

ODA Configuration for Events

ODA Provides Events Siebel Web service only supports Basic Auth whereas ODA supports OAuth 2.0. Since eScripts directly in Siebel cannot be used for building sha256 HMAC signatures headers, a JAVA service was deployed in the Docker Container Tomcat server where Siebel is hosted. This can be invoked internally from Business services or Workflows.

1. Identify the source of the event: Identify the data that needs to be sent to the users or a predefined message that needs to be sent to users on a certain events occurring.
2. In Digital Assistant, register the event type.

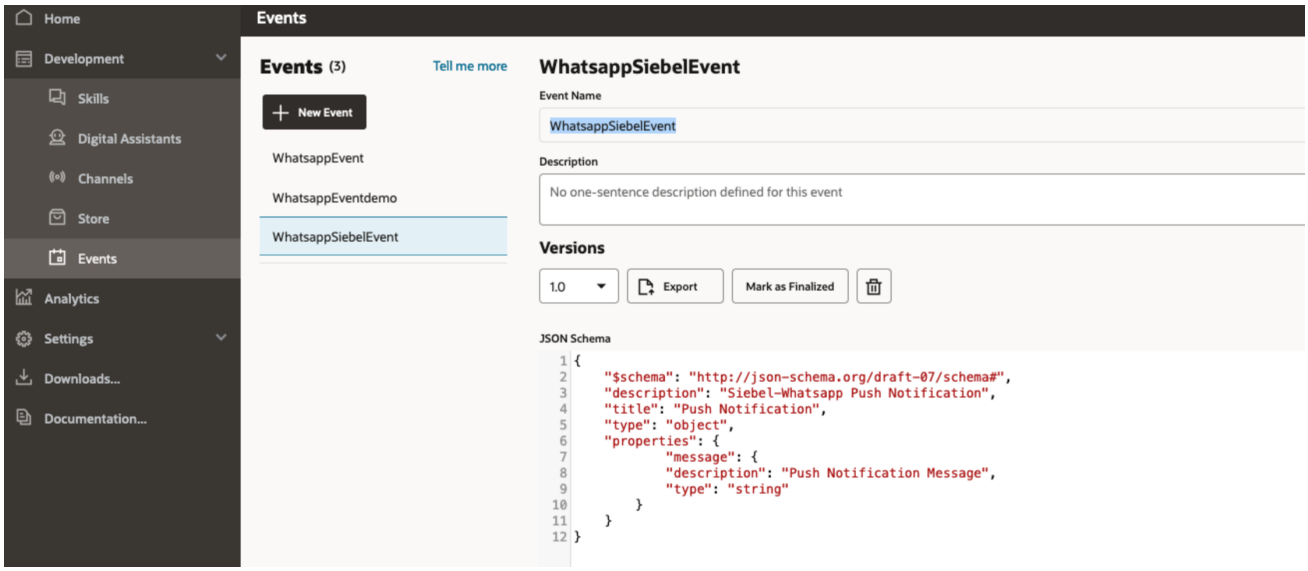

Figure 12: ODA Events type

Click ☰ to open the side menu, select Development > Events, click New Event, and enter a name for the event type. Define the JSON Schema like below (the structure of event).

```
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "description": "Siebel-Whatsapp Push Notification",
    "title": "Push Notification",
    "type": "object",
    "properties": {
        "message": {
            "description": "Push Notification Message",
            "type": "string"
        }
    }
```

When you have finished work on the schema and want to freeze its contents, click Mark as Finalized. At this point, you can use this event type in a skill.

3. Configure a skill to consume events of that event type and add that skill to a digital assistant: In a skill, create a flow with the Notify User component to consume the event. (This component is only available for dialog flows developed in Visual mode.) At runtime, when the event is generated, the event is passed to

Copyright © 2026, Oracle and/or its affiliates / Public

the skill. You can use expressions to access the event's data and context.

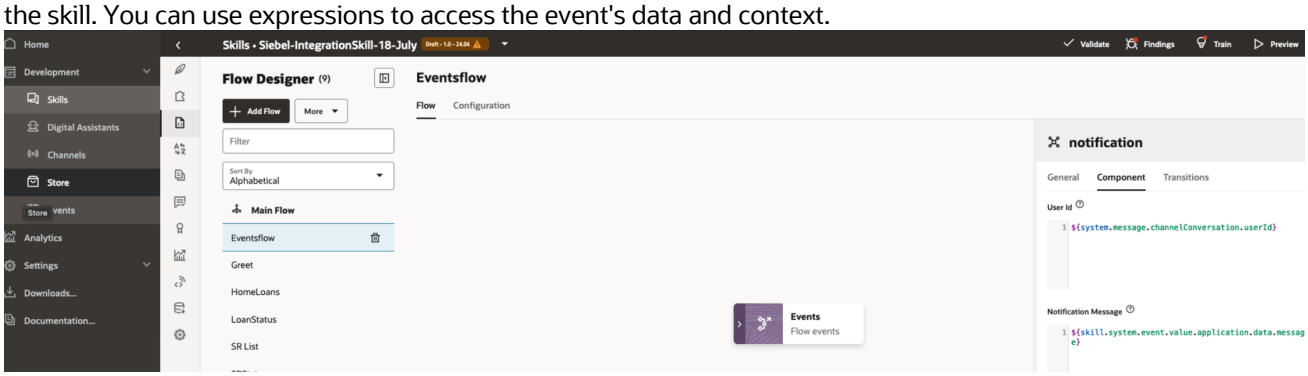

Figure 13: ODA Skills settings 1

4.

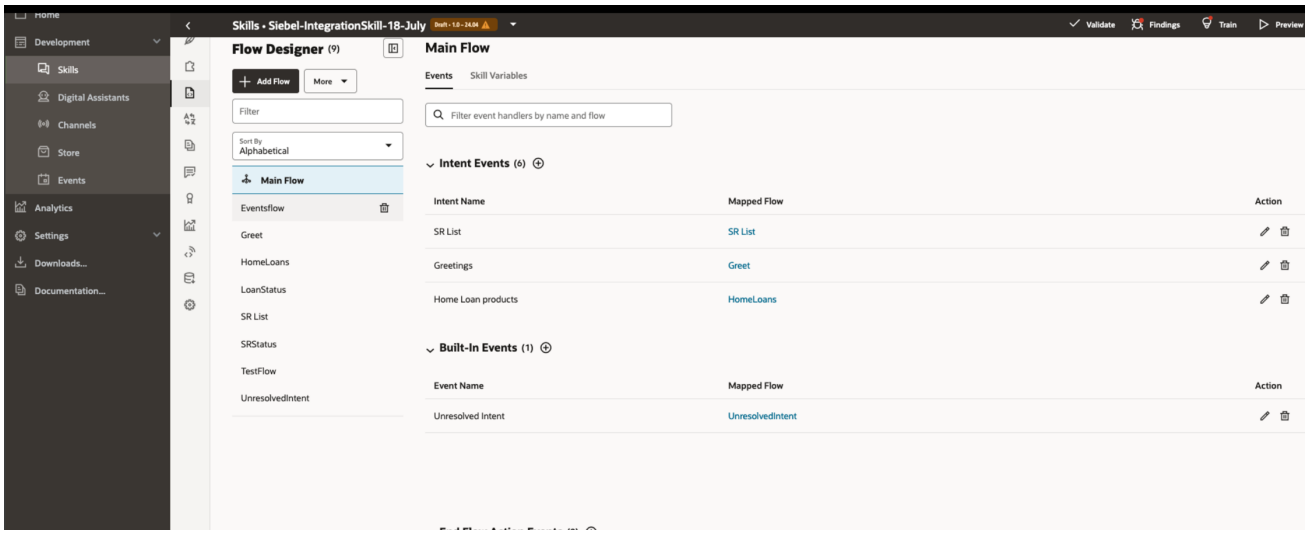

Figure 14: ODA Skills settings 2

5. Create an Events channel and map it to the digital assistant and enable it. Note the Secret key and inbound URL.

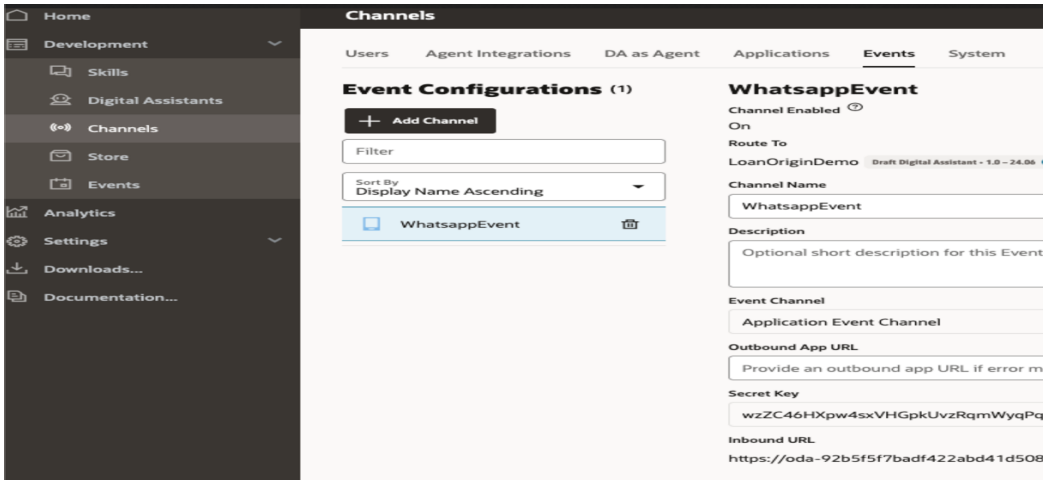

Figure 15: ODA Channel settings 3

Copyright © 2026, Oracle and/or its affiliates / Public

From the created Events channel, copy the inbound URL and secret key that will be used to Push events from Siebel.

Payload of Event

```
const event = {

        specversion: "1.0",
        type: "WhatsappSiebelEvent",   // Replace with event name
        source: "https://my-event-source",  // Replace with your event source
        id: "123e4567-e89b-12d3-a456-426614174000",  // Replace with your unique
event ID
        //time: new Date().toISOString(),
        channelname: "ODA_WhatsApp",
        version: "1.0",
        userid: "918281385304",
        datacontenttype: "application/json",
        data: eventData

      };
```

Must have **userid** key to have value of customers Phone number, which will be used to send whatsapp message to that particular number.

Siebel Set Field Trigger (eScript)

In Opportunity Business Component, whenever the Status of the Opportunity is updated a Push Notification must be sent to the Corresponding Phone Number associated to the Contact of the Opportunity. A trigger is written in the **Opportunity** Business component BusComp_SetFieldValue eScript.

```
function BusComp_SetFieldValue (FieldName)
{

    var status = this.GetFieldValue("Status");
    if (status !== "Submitted") {
        return (ContinueOperation);
    }
    var channel = "https://oda-92b5f5f7badf422abd41d5086ed61beb-
da7.data.digitalassistant.oci.oraclecloud.com/events/v2/listeners/appevent/channe
ls/50d29821-7026-41c1-ac49-00406085c44f";
    var secret = "wzZC46HXpw4sxVHGpkUvzRqmWyqPqSaA";
    var message ="this is coming from Siebel testing";
    var name = this.GetFieldValue("Name");


    var bo = TheApplication().GetBusObject("Opportunity");
    var bc = bo.GetBusComp("Opportunity");
    bc.ClearToQuery();
    bc.SetSearchSpec("Name", name);
    bc.ExecuteQuery(ForwardOnly);

    var phnumber = "";

    if (bc.FirstRecord()) {

        phnumber = bc.GetFieldValue("Key Contact Work #");
    }

    bc = null;
    bo = null;
```


Copyright © 2026, Oracle and/or its affiliates / Public

```
    var httpService = TheApplication().GetService("EAI HTTP Transport");
    var httpInputs = TheApplication().NewPropertySet();
    var httpOutputs = TheApplication().NewPropertySet();

    var url = "http://sai-ENT.company.com:4431/SiebelExternalService-
POC/siebel/eventPusher/pushEvent";
    var requestBody = '{"channel":"' + channel + '", ' +
        '"secret":"' + secret + '", ' +
        '"message":"' + message + '", ' +
        '"userid":"' + phnumber +'"}';

    httpInputs.SetProperty("HTTPRequestURLTemplate", url);
    httpInputs.SetProperty("HTTPRequestMethod", "POST");
    httpInputs.SetProperty("HTTPContentType", "application/json");
    httpInputs.SetProperty("CharSetConversion", "UTF-8");
    httpInputs.SetProperty("HTTPAccept", "*/*");
    httpInputs.SetValue(requestBody);

    httpService.InvokeMethod("SendReceive", httpInputs, httpOutputs);
   return (CancelOperation);
}
```


Copyright © 2026, Oracle and/or its affiliates / Public

Benefits

This initial beta release is ready to transform your development cycle with the following key features:

- **Unified Messaging Architecture**: Centralizes inbound WhatsApp messages and outbound push notifications through a single Java-based webhook deployed within the Siebel Tomcat environment, reducing architectural complexity and operational overhead.
- **Real-Time, Bidirectional Communication**: Enables seamless two-way communication between customers and Siebel CRM via WhatsApp, ensuring timely responses, conversational continuity, and improved customer engagement.
- **Event-Driven Push Notifications**: Leverages Oracle Digital Assistant Events to trigger proactive WhatsApp notifications based on Siebel business events and workflows, enabling timely, contextual customer outreach.
- **Enterprise-Grade Security and Compliance**: Adheres to Oracle-approved deployment models using HTTPS, OAuth 2.0 (via ODA), and controlled internal service calls, ensuring secure and compliant message exchange.
- **Reduced External Dependencies**: Eliminates the need for standalone VMs or third-party Node.js services by consolidating integration logic within the existing Siebel infrastructure, simplifying maintenance and governance.
- **Scalable and Extensible Design**: Supports high message volumes and can be easily extended to additional ODA skills, events, or messaging use cases without major architectural changes.

Future Works

- **Multi-Channel Expansion**: Extend the same integration framework to additional ODA-supported channels such as SMS, Web Chat, or Microsoft Teams for a unified omnichannel experience.
- **Advanced Event Payloads**: Enrich ODA Events with structured business data (order status, case updates, SLA alerts) to deliver more personalized and actionable notifications.
- **Template-Based Messaging**: Introduce configurable WhatsApp message templates managed within Siebel or ODA to standardize outbound communications and support regulatory requirements.

Conclusion

Modern Siebel development is often slowed by manual scripting, repetitive technical tasks, and constant context switching between tools. Siebel eScripts Code Assist fundamentally transforms this workflow. By embedding AI-powered code generation, explanations, and workspace synchronization directly within the familiar Visual Studio Code environment, we're not just adding convenience—we're redefining the developer experience.

Your team can shift from time-consuming boilerplate work to delivering high-quality customizations with greater speed, accuracy, and confidence. This integration is built to make developers more productive, code more consistent, and releases more reliable, all while securely leveraging your own OCI tenancy.

Call To Action

Ready to Elevate Customer Engagement with Siebel and WhatsApp?

Don't let fragmented communication channels and manual notification processes limit your customer experience. Take the next step toward a unified, event-driven messaging strategy by enabling real-time WhatsApp interactions and proactive push notifications directly from Siebel CRM.

Contact us directly at siebel_coe_grp@oracle.com to discuss your requirement.

Copyright © 2026, Oracle and/or its affiliates / Public

ORACLE

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Copyright © 2026, Oracle and/or its affiliates. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.