
	<h1>SISTEMA INTEGRAL DE MENSAJERÍA</h1> <h2>Guía para envío de avisos push</h2>	
---	---	---



MINISTERIO  
PARA LA TRANSFORMACIÓN  
DIGITAL Y DE LA  
FUNCIÓN PÚBLICA

SECRETARÍA GENERAL DE  
ADMINISTRACIÓN DIGITAL



# SISTEMA INTEGRAL DE MENSAJERÍA

<p><b>Guía para envío de avisos PUSH</b></p>	<p>Versión: 2.0</p> <p>FECHA: 30/06/2024</p>
--	--



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



### Registro de Modificaciones

Versión	Fecha	Descripción	Autor	Revisado
1.0	01/07/2016	Versión inicial	SGIADSC	
1.1	25/03/2021	Actualización para explicar la integración con FCM y nuevos campos del servicio web de envío de mensajes	SGAD	
1.2	12/04/2021	Se incluye funcionamiento de mensajes de datos y notificaciones silenciosas	SGAD	
2.0	30/06/2024	Se incluye nuevo funcionamiento de registro push (FCM)	SGAD	

## Índice

1.	Introducción .....	4
2.	Guía para envío de avisos push Android .....	5
2.1	Crear un proyecto de Firebase.....	5
2.2	Registrar la aplicación con Firebase.....	7
2.3	Agregar archivo de configuración de Firebase .....	9
2.4	Agregar el SDK de Firebase a la aplicación .....	11
2.5	Editar el fichero manifiesto de la aplicación.....	12
2.6	Acceder al token de registro del dispositivo FCM .....	13
2.7	Registrar Servicio PUSH en la Plataforma SIM .....	13
2.7.1	Creación rol en Google Cloud .....	14
2.7.2	Creación cuenta de servicio .....	15
2.7.3	Generación fichero clave JSON .....	18
2.7.4	Incluir clave JSON en SIM.....	19
3.	Guía para envío de avisos push iOS.....	20
3.1	Habilitar las notificaciones Push en la aplicación móvil.....	20
3.2	Crear un certificado de avisos Push .....	21
3.3	Registrar Servicio PUSH en la plataforma SIM .....	23
3.4	Registro del terminal en APNS .....	24
4.	Registro de terminal en la plataforma SIM .....	25
5.	Envío de avisos PUSH .....	27
5.1.	Mensajes de datos / notificaciones silenciosas .....	30
6.	Gestión del estado de los avisos push.....	31



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



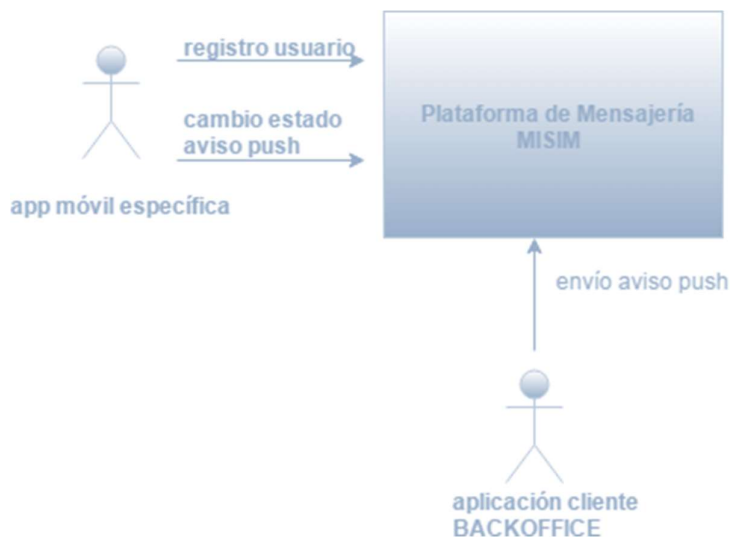
### 1. Introducción

En este documento se especifican las posibilidades de interconexión de una app móvil que quiera hacer uso de los servicios disponibles en la Plataforma de Mensajería SIM.

Se especificarán las siguientes funcionalidades:

- Registro de un terminal en SIM para la recepción de avisos push. Para poder realizar envíos de avisos PUSH a usuarios a través de la plataforma SIM es necesario que los responsables de las aplicaciones móviles realicen los pasos descritos a continuación. El procedimiento a seguir dependerá del sistema operativo de la aplicación móvil: Android o iOS.
- El envío de avisos push por parte de la aplicación cliente.
- Gestión del cambio de estado de los avisos push enviados.

El esquema es el siguiente:





# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



## 2. Guía para envío de avisos push Android

Los responsables de las aplicaciones móviles Android deberán realizar los siguientes pasos.

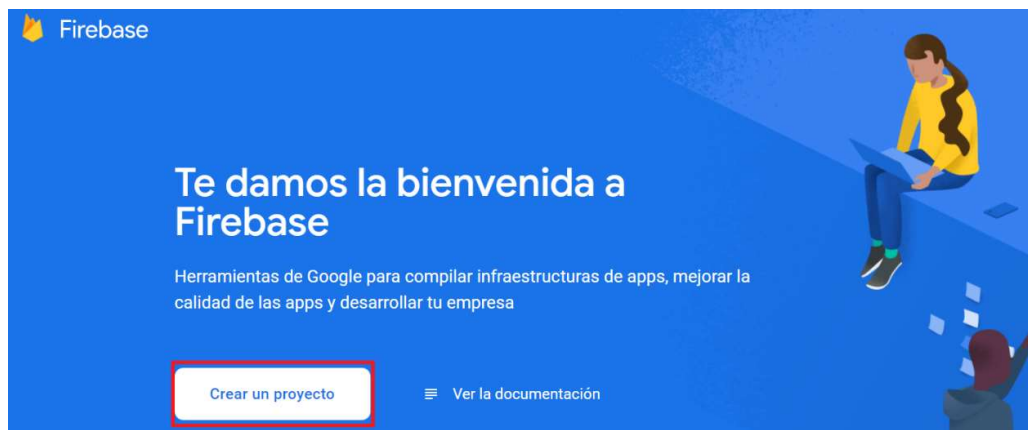
Se muestran los pasos ofrecidos por Firebase en su documentación a la fecha de realización del documento.

### 2.1 Crear un proyecto de Firebase

Para poder realizar envíos a aplicaciones móviles para Android es necesario previamente crear un proyecto de Firebase y conectarlo a la app.

Para ello, se deben seguir los siguientes pasos:

1. Acceder a [Firebase console](#).
2. Hacer clic en Crear un proyecto e introducir el nombre del mismo:





# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



✕ Crear un proyecto(paso 1 de 3)

Comencemos con el  
nombre de tu proyecto®

Ingresa el nombre de tu proyecto

El nombre del proyecto es obligatorio.

my-awesome-project-id

☐ Acepto las [condiciones de Firebase](#)

Continuar

Si ya se dispone de un proyecto de Google Cloud, se podrá seleccionar de un menú desplegable para agregarle recursos de Firebase.

3. Hacer clic en Continuar.
4. (Opcional) Configurar Google Analytics para el nuevo proyecto y hacer clic en Continuar:

✕ Crear un proyecto(paso 2 de 3)

Google Analytics es una solución de estadísticas ilimitada y gratuita que permite usar la orientación, los informes y otras funciones en Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions y Cloud Functions.

Google Analytics habilita las siguientes funciones:

- Pruebas A/B ⓘ
- Usuarios que no experimentan fallas ⓘ
- Segmentación de usuarios y orientación a ellos en los productos de Firebase ⓘ
- Activadores de Cloud Functions basados en eventos ⓘ
- Predicción del comportamiento de los usuarios ⓘ
- Informes ilimitados y gratuitos ⓘ

☒ Habilitar Google Analytics para este proyecto  
Recomendado

Anterior Continuar

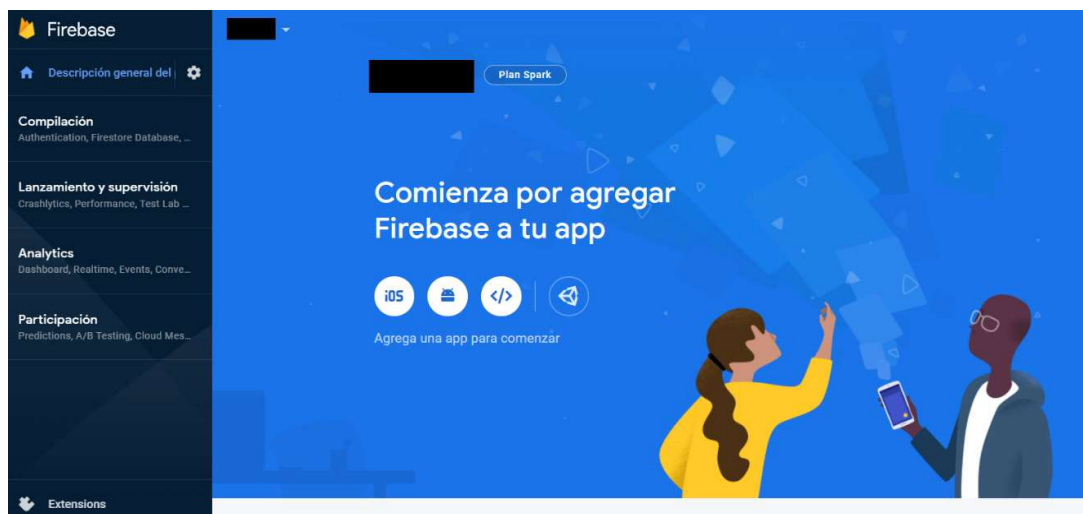
5. Hacer clic en Crear proyecto.

Una vez finalizada la creación del proyecto, debería visualizarse una pantalla como la siguiente:



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push

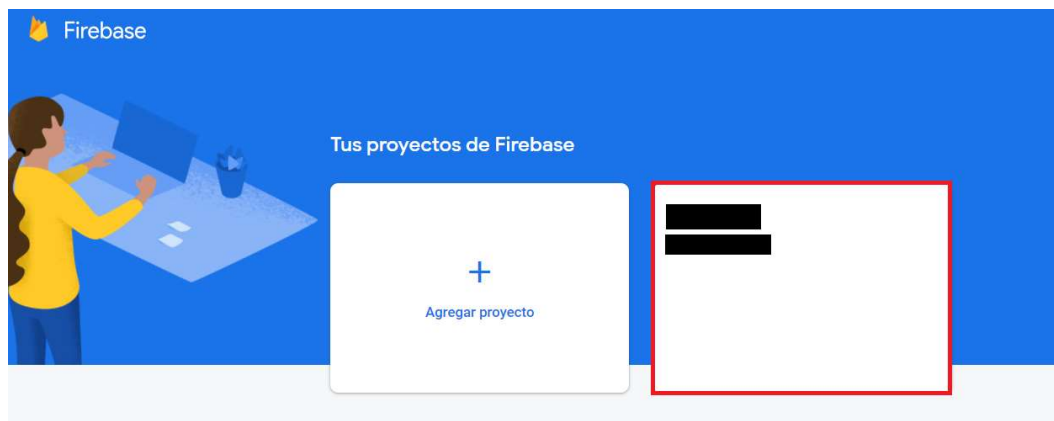


## 2.2 Registrar la aplicación con Firebase

Para usar Firebase en la aplicación para Android, se debe registrar la aplicación con el proyecto de Firebase.

Para ello, se deben seguir los siguientes pasos:

1. Acceder a [Firebase console](#).
2. Clicar en el proyecto al que se debe asignar la aplicación. En este caso, se seleccionará el nuevo proyecto creado previamente:

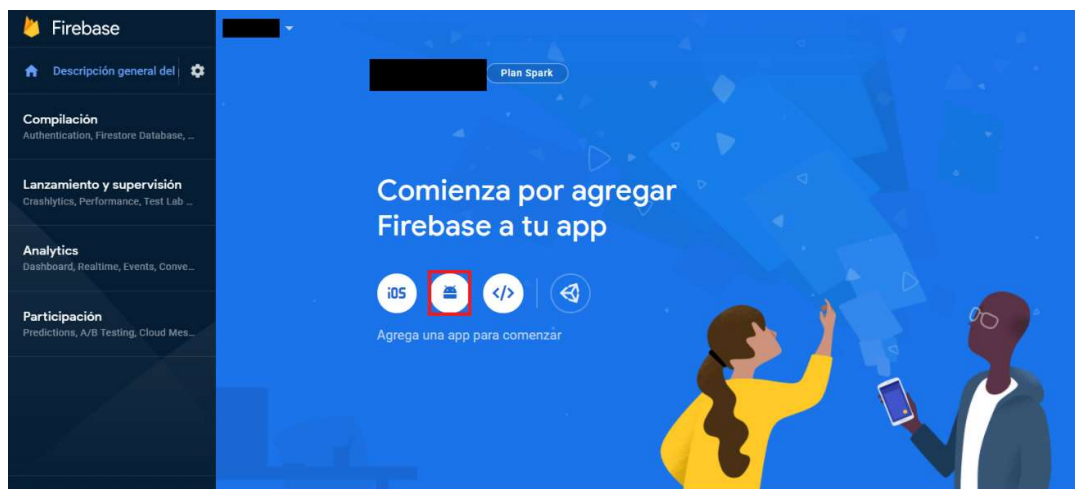


3. En el centro de la página de descripción general del proyecto, hacer clic en el icono de Android o en Agregar app para iniciar el flujo de trabajo de configuración.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



4. Completar el formulario emergente introduciendo los datos solicitados, siendo estos:

- Nombre del paquete de Android: nombre que identifica de forma única la aplicación en el dispositivo y en Google Play Store. Suele referirse al ID de aplicación y generalmente puede obtenerse buscando el nombre del paquete de la app en el archivo build.gradle (por ejemplo: com.yourcompany.yourproject)
- (Opcional) Sobrenombre de la app: identificador interno para Firebase. Solo puede verlo el usuario gestor de la aplicación.
- (Opcional) Certificado de firma SHA-1 de depuración





5. Hacer clic en Registrar app.

## 2.3 Agregar archivo de configuración de Firebase

Tras registrar la app en Firebase, debe procederse a configurar la aplicación para el uso de Firebase. Para ello, en los siguientes pasos del formulario anterior se debe:

1. Agregar el archivo de configuración de Firebase para Android a la aplicación, como se indica a continuación y en el formulario:
  - a. Hacer clic en **Descargar google-services.json** para obtener el archivo de configuración de Firebase para Android (*google-services.json*).
  - b. Transferir el archivo de configuración al directorio del módulo (nivel de app) de la aplicación en cuestión.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push

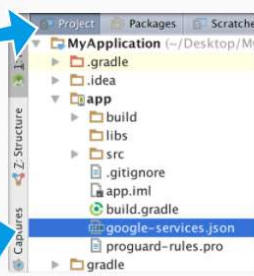


2 Descargar archivo de configuración Instrucciones para Android Studio a continuación | [Unity](#) [C++](#)

[Descargar google-services.json](#)

Cambia a la vista **Proyecto** de Android Studio para ver el directorio raíz de tu proyecto.

Coloca el archivo `google-services.json` que acabas de descargar en el directorio raíz del módulo de tu app para Android.



**google-services.json**

[Siguiente](#)

3 Agregar el SDK de Firebase

4 Próximos pasos

2. Agregar el complemento de `google-services` a los archivos Gradle de la aplicación a fin de habilitar los productos de Firebase.
  - a. Agregar las reglas para incluir el complemento de servicios de Google al archivo Gradle (*build.gradle*) de nivel de raíz (a nivel de proyecto). Además, revisar que tiene el repositorio Maven de Google.

```
buildscript {  
  
    repositories {  
        // Check that you have the following line (if not, add  
it):  
        google() // Google's Maven repository  
    }  
  
    dependencies {  
        // ...  
  
        // Add the following line:  
        classpath 'com.google.gms:google-services:4.3.5' //  
Google Services plugin  
    }  
}
```



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



```
}  
  
allprojects {  
    // ...  
  
    repositories {  
        // Check that you have the following line (if not, add  
it):  
        google() // Google's Maven repository  
        // ...  
    }  
}
```

- b. En el archivo Gradle (generalmente *app/build.gradle*) del módulo (a nivel de app), aplicar el complemento Gradle de los servicios de Google:

```
apply plugin: 'com.android.application'  
// Add the following line:  
apply plugin: 'com.google.gms.google-services' // Google  
Services plugin  
  
android {  
    // ...  
}
```

## 2.4 Agregar el SDK de Firebase a la aplicación

El complemento de *google-services* para Gradle añadido en el paso anterior carga el fichero de configuración *google-services.json* descargado previamente.

A continuación, se deben declarar las dependencias de los productos de Firebase que se quieren usar en la aplicación.

En este caso, para permitir el envío de notificaciones push, será necesario declarar en el archivo Gradle de la aplicación la dependencia correspondiente a Firebase Cloud Messaging o FCM de la siguiente manera:

```
dependencies {  
    // ...  
  
    // Import the Firebase BoM
```



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



```
implementation platform('com.google.firebase:firebase-bom:26.6.0')

// When using the BoM, you don't specify versions in Firebase
library dependencies

// Declare the dependencies for the desired Firestore products
implementation 'com.google.firebase:firebase-messaging'
```

Una vez declaradas las dependencias, será necesario sincronizar la aplicación para garantizar que la aplicación hace uso de las versiones necesarias de cada dependencia.

## 2.5 Editar el fichero manifiesto de la aplicación

Se debe agregar lo siguiente al fichero del manifiesto de la aplicación:

- Un servicio que extienda *FirebaseMessagingService*. Esto es obligatorio si se desea administrar los mensajes además de recibir notificaciones en aplicaciones en segundo plano. Para recibir notificaciones en aplicaciones en primer plano, recibir la carga útil de datos, enviar mensajes ascendentes, etc., se debe extender este servicio.

Por ejemplo:

```
<service
    android:name=".java.MyFirebaseMessagingService"
    android:exported="false">
    <intent-filter>
        <action android:name="com.google.firebase.MESSAGING_EVENT"
    />
    </intent-filter>
</service>
```

- (Opcional) Dentro del componente de la aplicación, elementos de metadatos para configurar el ícono y el color predeterminados de la notificación. Android usa estos valores cada vez que los mensajes entrantes no tienen un ícono ni un color configurado de manera explícita.

```
<!-- Set custom default icon. This is used when no icon is set
for incoming notification messages.
See README(https://goo.gl/l4GJaQ) for more. -->
<meta-data
```



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



```
android:name="com.google.firebase.messaging.default_notification_
icon"
    android:resource="@drawable/ic_stat_ic_notification" />
<!-- Set color used with incoming notification messages. This is
used when no color is set for the incoming
notification message. See README (https://goo.gl/6BKBk7) for
more. -->
<meta-data

android:name="com.google.firebase.messaging.default_notification_
color"
    android:resource="@color/colorAccent" />
```

## 2.6 Acceder al token de registro del dispositivo FCM

Cuando se inicia la aplicación móvil por primera vez en un dispositivo, el SDK de FCM genera un token de registro para la instancia de la app cliente. Si el objetivo son dispositivos individuales o la creación de grupos de dispositivos, es necesario extender *FirebaseMessagingService* y anular *onNewToken* para acceder a este token.

En esta sección, se describe cómo recuperar el token actual del dispositivo. Debido a que el token puede rotar después del primer inicio de la aplicación, se recomienda siempre recuperar el token de registro con la actualización más reciente.

El token de registro puede cambiar en las siguientes situaciones:

- La app se restablece en un dispositivo nuevo.
- El usuario desinstala y vuelve a instalar la app.
- El usuario borra los datos de la app.

Para acceder al token actual del dispositivo, se debe llamar al siguiente método:

```
FirebaseMessaging.getInstance().getToken()
```

Este token deberá ser informado a la hora de registrar el usuario correspondiente en SIM, de acuerdo a lo especificado en el punto 4 del manual.

## 2.7 Registrar Servicio PUSH en la Plataforma SIM

Una vez efectuada toda la configuración de la aplicación móvil en Firebase, es necesaria la creación de un servicio PUSH asociado a la aplicación en la Plataforma de SIM para permitir los envíos de notificaciones push.



# SISTEMA INTEGRAL DE MENSAJERÍA



## Guía para envío de avisos push

Previamente se debe generar el fichero json con las credenciales y remitirlo al soporte SIM.

### 2.7.1 Creación rol en Google Cloud

Lo primero que deberemos hacer es crear un rol que nos permita el envío de notificaciones, para ello en la **consola de Google Cloud** accedemos a la pestaña **Roles**:

Start your Free Trial with \$300 in credit. Don't worry—you won't be charged if you run out of credits. [Learn more](#)

Google Cloud test Search (/) for resources, docs, products, and more

IAM & Admin Roles + CREATE ROLE CREATE ROLE FROM SELECTION — DISABLE DELETE

Roles for "test" project

A role is a group of permissions that you can assign to principals. You can create a role and add permissions to it, or copy an existing role and adjust its permissions. [Learn more](#)

Filter Enter property name or value

<input type="checkbox"/>	Type	Title ↑	Used in	Status	
<input type="checkbox"/>		<a href="#">Access Approval Approver</a>	Access Approval	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Approval Config Editor</a>	Access Approval	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Approval Invalidator</a>	Access Approval	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Approval Viewer</a>	Access Approval	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Context Manager Admin</a>	Access Context Manager	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Context Manager Editor</a>	Access Context Manager	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Context Manager Reader</a>	Access Context Manager	Enabled	⋮
<input type="checkbox"/>		<a href="#">Access Transparency Admin</a>	Organization Policy	Enabled	⋮
<input type="checkbox"/>		<a href="#">Actions Admin</a>	Actions	Enabled	⋮
<input type="checkbox"/>		<a href="#">Actions Viewer</a>	Actions	Enabled	⋮
<input type="checkbox"/>		<a href="#">Activity Analysis Viewer</a>	Other	Enabled	⋮
<input type="checkbox"/>		<a href="#">Admin</a>	Cloud Talent Solution	Enabled	⋮
<input type="checkbox"/>		<a href="#">Admin of Tenancy Units</a>	Service Consumer Management	Enabled	⋮
<input type="checkbox"/>		<a href="#">Advisory Notifications Admin</a>	Other	Enabled	⋮
<input type="checkbox"/>		<a href="#">Advisory Notifications Viewer</a>	Other	Enabled	⋮
<input type="checkbox"/>		<a href="#">AI Platform Admin</a>	AI Platform	Enabled	⋮
<input type="checkbox"/>		<a href="#">AI Platform Developer</a>	AI Platform	Enabled	⋮
<input type="checkbox"/>		<a href="#">AI Platform Job Owner</a>	AI Platform	Enabled	⋮
<input type="checkbox"/>		<a href="#">AI Platform Model Owner</a>	AI Platform	Enabled	⋮
<input type="checkbox"/>		<a href="#">AI Platform Model User</a>	AI Platform	Enabled	⋮

Seleccionamos Create Role y a continuación veremos la siguiente pantalla.

Rellenamos los datos como se indica en la imagen y posteriormente asignamos los permisos:



# SISTEMA INTEGRAL DE MENSAJERÍA



## Guía para envío de avisos push

**IAM & Admin**

- IAM
- Identity & Organization
- Policy Troubleshooter
- Policy Analyzer
- Organization Policies
- Service Accounts
- Workload Identity Federation
- Workforce Identity Federation
- Labels
- Tags
- Settings
- Privacy & Security
- Identity-Aware Proxy
- Roles**
- Audit Logs
- Essential Contacts
- Asset Inventory
- Quotas & System Limits
- Groups

**Create Role**

Custom roles let you group permissions and assign them to principals in your project or organization. You can manually select permissions or import permissions from another role. [Learn more](#)

**Title \***  
Cliente SIM Role  
16 / 100 characters

**Description**  
Created on: 2024-03-07  
22 / 256 characters

**ID \***  
ClienteSIMRole

**Role launch stage**  
General Availability

[+ ADD PERMISSIONS](#)

**1 assigned permission**

**Filter** Enter property name or value

Permission	Status
cloudmessaging.messages.create	Supported

Some permissions might be associated with and checked by third parties. These permissions contain the third party's service and domain name in the permission prefix.

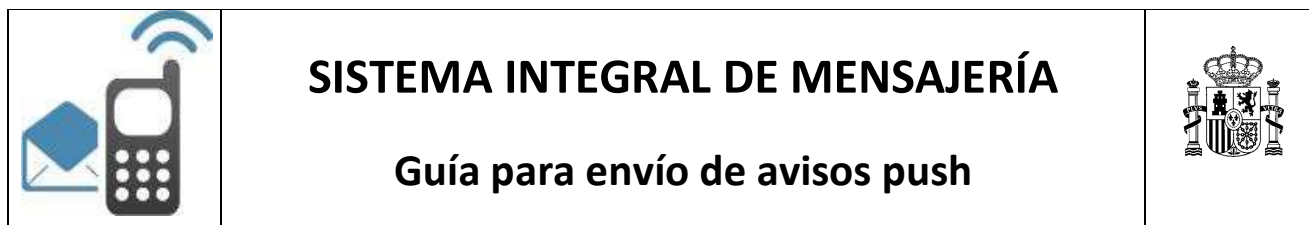
[SHOW ADDED AND REMOVED PERMISSIONS](#)

[CREATE](#) [CANCEL](#)

En nuestro caso el único permiso necesario es **cloudmessaging.messages.create** que es el que permite el envío de notificaciones, no seleccionaremos ningún permiso más para mejorar la seguridad.

### 2.7.2 Creación cuenta de servicio

A continuación, crearemos la cuenta de servicio que se usará en SIM para el envío de notificaciones. Para ello desde el menú **Service Accounts** seleccionamos **Create Service Account**:



**IAM & Admin** | Service accounts | **+ CREATE SERVICE ACCOUNT** | DELETE | MANAGE ACCESS | REFRESH

**Service accounts for project "test"**

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts](#)

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies](#)

Filter: Enter property name or value

Email	Status	Name ↓	Description	Key ID	Key creation date	OAuth 2.0 Client ID	Actions
test-667@test-89ca6.iam.gserviceaccount.com	Enabled	test		bd57a8c31a9d262595c894e0a9167cd039e3a1a0	Mar 4, 2024	105235232108772720340	⋮
firebase-adminsdk-utww@test-89ca6.iam.gserviceaccount.com	Enabled	firebase-adminsdk		b9c547fe4abf42241cf4325770bed36fd5859e38	Mar 4, 2024	110838780664031757209	⋮

Nos aparecerá la siguiente pantalla con los datos a rellenar:

**IAM & Admin** | **Create service account**

**1 Service account details**

**Service account name**  
Cliente SIM

Display name for this service account

**Service account ID \***  
cliente-sim-957

Email address: cliente-sim-957@test-89ca6.iam.gserviceaccount.com

**Service account description**  
Describe what this service account will do

**CREATE AND CONTINUE**

**2 Grant this service account access to project (optional)**

**3 Grant users access to this service account (optional)**

**DONE CANCEL**

Rellenamos el nombre de la cuenta y pulsamos en continuar.

Después buscamos y seleccionamos el rol que hemos creado anteriormente y lo asociamos a la cuenta de servicio:





# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



IAM & Admin

← Create service account

✓ Service account details

2 Grant this service account access to project (optional)

Grant this service account access to test so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Select a role IAM condition (optional) ?

Filter Cliente SIM Role

Cliente SIM Role  
Created on: 2024-03-07

DONE

MANAGE ROLES

Y finalmente pulsamos en **Done** para terminar el proceso de creación.

IAM & Admin

← Create service account

✓ Service account details

2 Grant this service account access to project (optional)

Grant this service account access to test so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Role Cliente SIM Role IAM condition (optional) ?

Created on: 2024-03-07

+ ADD ANOTHER ROLE

CONTINUE

3 Grant users access to this service account (optional)

DONE CANCEL

### 2.7.3 Generación fichero clave JSON

Una vez creada la cuenta, entramos en ella desde el listado de cuentas de servicio:

IAM & Admin

IAM

Identity & Organization

Policy Troubleshooter

Policy Analyzer

Organization Policies

Service Accounts

Workload Identity Federation

Workforce Identity Federation

Service accounts

+ CREATE SERVICE ACCOUNT

DELETE

+ MANAGE ACCESS

REFRESH

Service accounts for project "test"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts.](#)

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies.](#)

Filter

Enter property name or value

<input type="checkbox"/>	Email	Status	Name	Description	Key ID	Key creation date	OAuth 2 Client ID	Actions
<input type="checkbox"/>	cliente-sim-932@test-89ca6.iam.gserviceaccount.com	Enabled	Cliente SIM	No keys			109435955413068367708	
<input type="checkbox"/>	firebase-adminsdk-lutww@test-89ca6.iam.gserviceaccount.com	Enabled	firebase-adminsdk		b9c547fe4abf42241cf4325770bed36fd5859e38	Mar 4, 2024	110838780664031757209	
<input type="checkbox"/>	test-667@test-89ca6.iam.gserviceaccount.com	Enabled	test		bd57a8c31a9d262595c894e0a9167cd039e3a1a0	Mar 4, 2024	105235232108772720340	

A continuación, en la pestaña **Keys** seleccionamos **Add Key** y **Create new key**:

IAM & Admin

IAM

Identity & Organization

Policy Troubleshooter

Policy Analyzer

Organization Policies

Service Accounts

Workload Identity Federation

Workforce Identity Federation

Labels

Tags

← Cliente SIM

DETAILS

PERMISSIONS

KEYS

METRICS

LOGS

Keys

Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the [Google Cloud SDK](#).

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using [organization policies](#).

[Learn more about setting organization policies for service accounts](#)

ADD KEY

Create new key

Upload existing key

Key creation date

Key expiration date

En el tipo de clave seleccionamos JSON:

Create private key for "Cliente SIM"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

☒ JSON

Recommended

☐ P12

For backward compatibility with code using the P12 format

CANCEL

CREATE

Página 18 de 31

Al darle a **Create** se descargará el fichero JSON con la clave. **Este fichero es importante custodiarlo adecuadamente por motivos de seguridad**, ya que nos dará acceso al envío de notificaciones en el proyecto. **Este archivo es el que habrá que incluir en SIM** para poder realizar los envíos de notificaciones push.

Finalmente, en el listado veremos la clave que hemos generado y su estado: Activa

**IAM & Admin**

- IAM
- Identity & Organization
- Policy Troubleshooter
- Policy Analyzer
- Organization Policies
- Service Accounts**
- Workload Identity Federation
- Workforce Identity Federation
- Labels
- Tune

← Cliente SIM

DETAILS PERMISSIONS **KEYS** METRICS LOGS

**Keys**

⚠ Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the [Virtual Machine Authentication Library](#).

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using [organization policies](#).

[Learn more about setting organization policies for service accounts](#)

**ADD KEY**

Type	Status	Key	Key creation date	Key expiration date	
	Active	559958e3707579113546415b47e4c44f976c51bf	Mar 7, 2024	Jan 1, 10000	

En cualquier momento podremos desactivar la clave para evitar accesos no deseados o incluso eliminarla y generar una nueva.

### 2.7.4 Incluir clave JSON en SIM

Desde el menú de Servicio, incluiremos como documento asociado a nuestra aplicación la clave para la cuenta de servicio.

Dicho servicio debe tener como canal “Canal Notificaciones PUSH”.

En la ventana de creación del servicio se debe seleccionar como plataforma Android, ya que la aplicación a integrar está desarrollada para dicho entorno.

En este paso, se ha de subir un archivo JSON, el cual a continuación se detalla cómo obtenerlo:

**Plataforma (\*):**

☒ [Android](#)

**Archivo JSON (\*):**

confNotificacionPushFCM.json

Subir JSON

Descargar archivo JSON

Borrar archivo JSON actual

La gestión de la clave se hará a través de esta pantalla, sustituyendo o eliminando este fichero cuando sea necesario.



## SISTEMA INTEGRAL DE MENSAJERÍA

### Guía para envío de avisos push



### **3. Guía para envío de avisos push iOS**

Para poder realizar envíos de avisos push a usuarios a través de la plataforma SIM, los responsables de las aplicaciones móviles iOS deberán realizar los siguientes pasos.

#### **3.1 Habilitar las notificaciones Push en la aplicación móvil**

Para poder realizar envíos a aplicaciones móviles iOS es necesario que dicha aplicación tenga habilitadas las notificaciones Push en su App ID del Member Center.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



**ID**

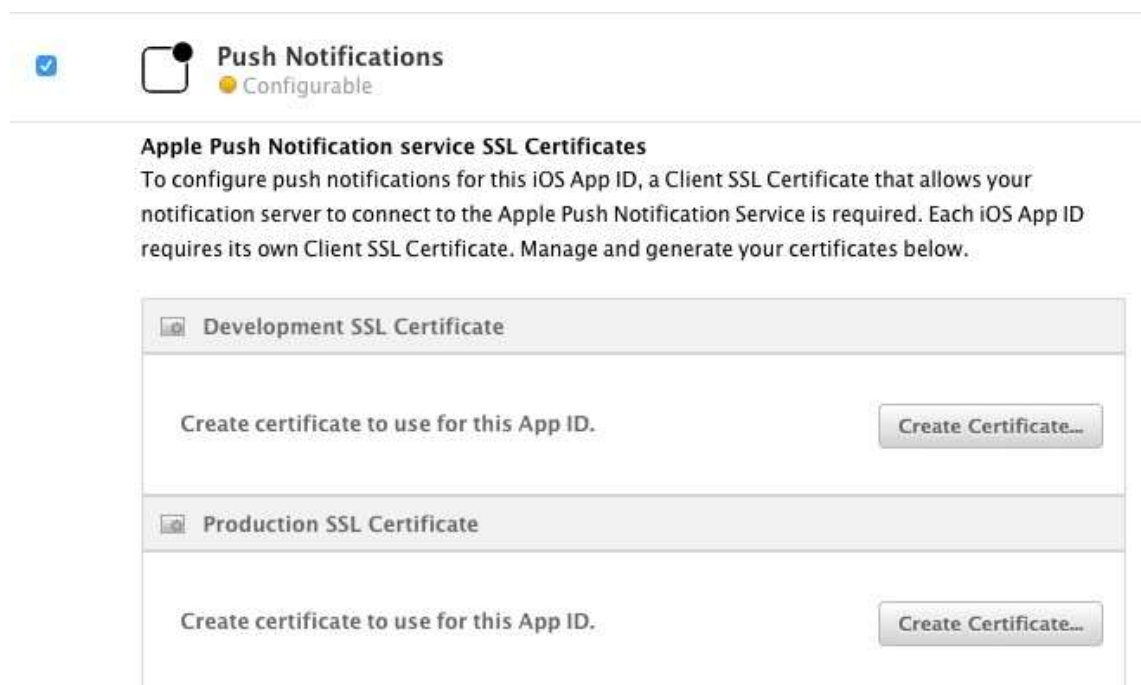
Name: Xcode iOS App ID [REDACTED]  
Prefix: S93JHWHY2R  
ID: [REDACTED]

Application Services:

Service	Development	Distribution
<b>App Group</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Associated Domains</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Data Protection</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Game Center</b>	<input checked="" type="radio"/> Enabled	<input checked="" type="radio"/> Enabled
<b>HealthKit</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>HomeKit</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Wireless Accessory Configuration</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>iCloud</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>In-App Purchase</b>	<input checked="" type="radio"/> Enabled	<input checked="" type="radio"/> Enabled
<b>Inter-App Audio</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Apple Pay</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Wallet</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled
<b>Push Notifications</b>	<input checked="" type="radio"/> Configurable	<input checked="" type="radio"/> Configurable
<b>VPN Configuration &amp; Control</b>	<input type="radio"/> Disabled	<input type="radio"/> Disabled

### 3.2 Crear un certificado de avisos Push

Se debe crear un certificado de avisos Push de producción para la aplicación que será usado en el lado del servidor. Es habitual que en el servidor se use con el formato .p12



El proceso de crear un certificado es exactamente el mismo que para un certificado de desarrollo o distribución.

Para crear un fichero .p12 de nuestro certificado seguiremos los siguientes pasos:

1. Descargamos el certificado creado y lo instalamos en nuestro llavero (Keychain). Basta con un doble click para que sea instalado.

Name	Kind	Expires	Keychain
Apple Development iOS Push Services: com[redacted]	certificate	Jan 4, 2017, 1:49:58 AM	login
[redacted]	private key	--	login

2. Pulsamos botón derecho sobre el certificado Push.

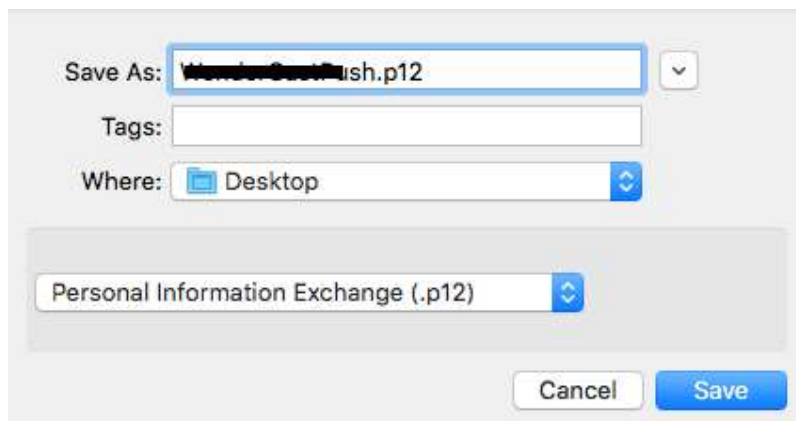
Apple De	New Identity Preference...	9:58 AM	login
Jack V			login
Apple De	Copy "Apple Development iOS Push Services: com[redacted]"	1:47:29 AM	login
Wend	Delete "Apple Development iOS Push Services: com[redacted]"		login
Apple Pr		1:49:20 PM	login
com.app	Export "Apple Development iOS Push Services: com[redacted]"	1:47:52 AM	login
com.app		1:16:09 PM	login
Develop	Get Info	1:05:48 PM	login
Develop	Evaluate "Apple Development iOS Push Services: com[redacted]"	1:05:48 PM	login

3. Elegimos el formato .p12 para exportar y el nombre que deseemos.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



4. Debemos establecer una contraseña para el fichero .p12 que posteriormente será usada en el servidor para poder abrir el fichero .p12.

### 3.3 Registrar Servicio PUSH en la plataforma SIM

Para poder realizar el envío a la aplicación móvil de notificaciones desde la plataforma de mensajería SIM es necesario registrar un nuevo Servicio en la plataforma.

Dicho servicio debe tener como canal “Canal Notificaciones PUSH”.

En la ventana de creación del servicio se debe seleccionar como plataforma iOS ya que la aplicación a integrar está desarrollada para dicho entorno.

En el apartado “APNS Ruta certificado”, se debe introducir el nombre del fichero generado en el punto anterior de la guía.

En el apartado “APNS Password certificado”, se debe introducir la clave del fichero p12 generado en el punto anterior de la guía.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



ESTÁS EN: > CONFIGURACIÓN DE APLICACIONES > SERVICIOS

Zoom: - +

### CONFIGURACIÓN

- Configuración General
  - Servidores SMTP
  - Proveedores SMS
  - Proveedores MISIM
  - Receptores SMS
  - Servidores Push
  - Servidores Web Push
- Configuración Aplicaciones
  - Aplicaciones
  - Gestión Aplicaciones
  - Servicios**
  - Gestión Servicios
  - Planificaciones
  - Servicios Móviles
- Configuración Seguridad
  - Usuarios
- Configuración Organismos
  - Organismos
  - Altas masivas
- Utilidades
  - Explorador de archivos
  - Consultas SQL
  - Notificaciones
  - Configuraciones
  - Registro de entrada
  - Decodificador
  - Gestión Passbook
  - Ejecución de Jobs
  - Lista de EndPoints
  - Ver Certificados
  - Auditoría
  - Contactos

### GESTIÓN

- Gestión de Envíos
- Estadísticas Generales
- Usuarios Push
- Usuarios Web Push
- Envío de Mensajes

### CREACIÓN SERVICIO

**Datos Generales**

Aplicación (\*):

Canal (\*):

Nombre (\*):

Descripción:

Activo: ☐

Nº Max. Envíos (\*):

Nº Max. Reintentos:

Historificación: ☒ 30 días ☐ 60 días ☐ 90 días ☐ Otro  días

*Si el periodo de tiempo es superior a 90 días debe indicar el motivo*

Conservación: ☒ 1 año ☐ 2 años ☐ 3 años ☐ Otro  años

*Si el periodo de tiempo es superior a 3 años debe indicar el motivo*

Plataforma (\*): ☐ Android ☐ iOS

Agrupar notificaciones cada (\*):

(Android) FCM API Key (\*):

(iOS) APNS Ruta Certificado (\*):

(iOS) APNS Password Certificado (\*):

(iOS) APNS Topic:

(\*) Campos obligatorios

**Contactos**

Responsable técnico (\*):

Email (\*):

Responsable funcional (\*):

Email (\*):

(\*) Campos obligatorios

**Informes**

Activo: ☐

### 3.4 Registro del terminal en APNS

En la aplicación cliente no es necesario establecer ninguna configuración especial para el uso de notificaciones Push. Podemos certificar que la aplicación está bien configurada a través del Member Center si nuestro target del proyecto, en la sección Capabilities, tiene activada la opción de Push Notifications.





Steps: ✓ Add the "Push Notifications" entitlement to your App ID

Es importante que el proyecto tenga el mismo BundleID que hemos definido en nuestro portal Member Center con avisos Push.

## 4. Registro de terminal en la plataforma SIM

En este apartado se define cómo realizar el registro de un usuario para permitir el envío de avisos PUSH. El endpoint expuesto por SIM será común para aplicaciones móviles basadas en Android o en iOS.

*Módulo de registro de usuario en avisos push*

- **Preproducción:**

[https://pre-misim.redsara.es/misim-bus-webapp/rest/registro?\\_wadl](https://pre-misim.redsara.es/misim-bus-webapp/rest/registro?_wadl)

- **Producción:**

[https://misim.redsara.es/misim-bus-webapp/rest/registro?\\_wadl](https://misim.redsara.es/misim-bus-webapp/rest/registro?_wadl)

Esta interfaz será utilizada una vez que el usuario se ha dado de alta en una aplicación móvil para la recepción de avisos push. A través de ella, la aplicación móvil deberá comunicar a la plataforma de mensajería SIM su identificador de registro, el identificador de la plataforma y el nombre de usuario.

La interfaz desplegada por SIM para la recepción de esta información admite los siguientes parámetros de entrada:

- **Servicio:** identificador del servicio asociado a la aplicación móvil en SIM
- **Identificador de registro (IdRegistro):** identificador obtenido por la app móvil en su registro en la plataforma de notificación correspondiente.
- **Plataforma de registro:** identificador de la plataforma a la que pertenece la aplicación móvil (FCM, APNS, MPNS):
  - 1 - Android
  - 2 - Apple
- **Usuario y contraseña:** autenticación de la aplicación cliente en SIM. El servicio web expuesto dispone de Authentication Basic recibiendo por este método el usuario y la contraseña de la aplicación cliente que quiere registrar un usuario en la plataforma.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



Opcionalmente también se pueden enviar:

- **Identificador de usuario:** es el identificador del usuario en la plataforma SIM. Asocia el terminal a un usuario ya registrado en la plataforma.
- **Identificador de dispositivo:** identificador del terminal móvil registrado para la recepción de los avisos push.

De acuerdo a la información que la aplicación móvil envíe a SIM, se pueden distinguir varias casuísticas, en función de si se envía el identificador de dispositivo, el identificador de usuario o ambos. Así, tendremos los siguientes casos:

- **Opción 1:** se envía el identificador del usuario. En este caso, la aplicación móvil dispone de identificación de usuario y quiere asociar varios dispositivos al mismo usuario, de manera que el envío de avisos push se realizará de manera simultánea a todos los dispositivos del usuario que estén registrados en la plataforma.
- **Opción 2:** se envía el identificador de dispositivo. La aplicación móvil requiere de una identificación propia del dispositivo, y el dispositivo quedará registrado en SIM con este identificador.
- **Opción 3:** no se envía ni identificador de dispositivo ni identificador de usuario. La aplicación móvil no dispone de identificador de usuario ni una identificación propia de dispositivo. Como no se envían estos datos, se generará en la plataforma SIM un idDispositivo y un idUsuario, que es el mismo que idDispositivo, y que se devuelve a la aplicación móvil.
- **Opción 4:** se envía identificador de dispositivo e identificador de usuario.

A continuación, se muestra con un ejemplo el registro de usuario.

[http://urlDelEntorno/misim-bus-webapp/rest/registro?\\_wadl](http://urlDelEntorno/misim-bus-webapp/rest/registro?_wadl)

*Endpoint:* [urlDelEntorno](#)

*Resource:* [/misim-bus-webapp/rest/registro](#)

Los parámetros que se deberán enviar son:

- **Servicio:** Id del servicio al que se quiere asociar a la app móvil
- **IdRegistro:**
  - Token generado al registrar el terminal en FCM (para obtenerlo, revisar punto 2.6 del manual)
  - Token generado al registrar el terminal en APNS (sin espacios)
- **Plataforma:** Plataforma a la que pertenece la aplicación móvil, en este caso
  - 1 – Android
  - 2 – Apple



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



Opcionalmente, se podrán enviar los siguientes parámetros:

- **IdDispositivo:** Identifica el terminal
- **IdUsuario:** Identificador del usuario en la Plataforma SIM. Asocia el terminal a un usuario ya registrado en la plataforma.

Por otro lado, este servicio web dispone de Authentication Basic, recibiendo por este método el usuario y la contraseña de la aplicación cliente que quiere registrar un usuario push en la plataforma.

En la siguiente figura se muestran los campos de petición y de respuesta:



## 5. Envío de avisos PUSH

En este apartado se especifica cómo realizar el envío de un aviso push a través de la plataforma.

El endpoint expuesto por SIM para el envío de avisos push es genérico, y se usa para el envío de otro tipo de mensaje (SMS, mail), luego habrá que particularizar los campos correspondientes.

El endpoint está disponible en la siguiente dirección:

- Preproducción:  
<https://pre-misim.redsara.es/misim-bus-webapp/EnvioMensajesService?wsdl>
- Producción:  
<https://misim.redsara.es/misim-bus-webapp/EnvioMensajesService?wsdl>

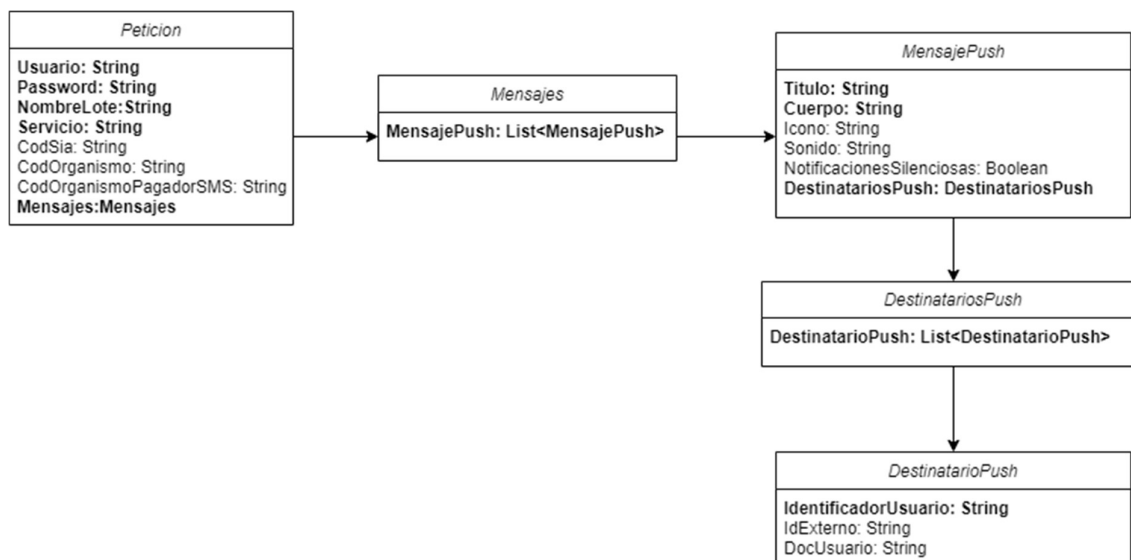


# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



La petición xml tendrá los siguientes campos:



Campos genéricos:

- Usuario: usuario de la aplicación cliente de SIM
- Password: password de la aplicación cliente de SIM
- NombreLote: nombre del lote para poder usar la funcionalidad de envío por lotes
- Servicio: servicio de envío de avisos push asociado a la aplicación
- Opcionalmente:
  - CodSia
  - CodOrganismo
  - CodOrganismoPagadorSMS

Campos del aviso push:

- Título: título del aviso push a enviar
- Cuerpo: cuerpo del aviso push.
- Opcionalmente:
  - Icono
  - Sonido
  - NotificacionesSilenciosas: informar un 1 para el envío del mensaje push como notificación silenciosa. Solo para envíos push en iOS.

Campos del destinatario push:



# SISTEMA INTEGRAL DE MENSAJERÍA



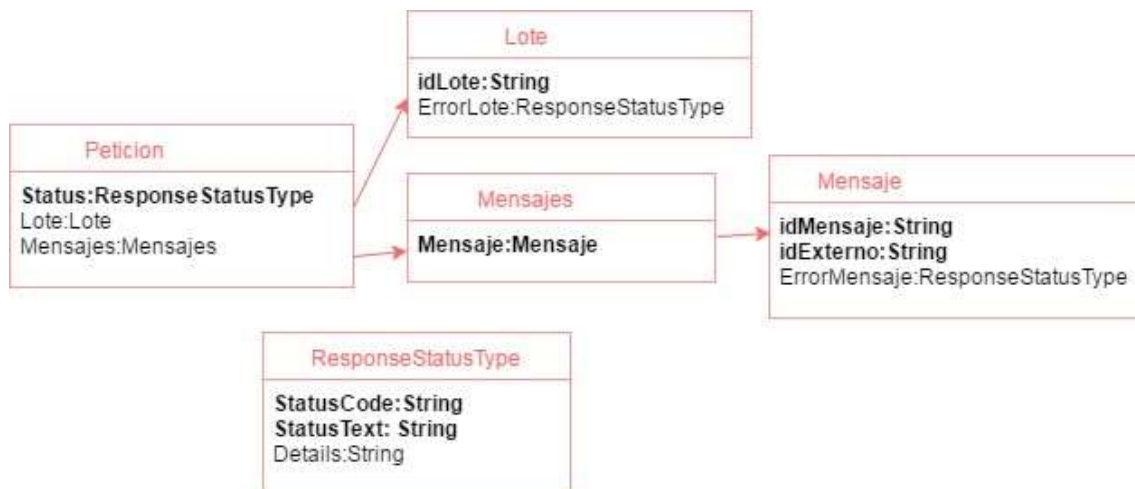
## Guía para envío de avisos push

- IdentificadorUsuario: identificador del usuario al cual se quiere enviar el aviso push
- Opcionalmente:
  - DocUsuario
  - IdExterno

A continuación, se muestra una petición XML de ejemplo:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:pet="http://misim.redsara.es/misim-bus-webapp/peticion">
  <soapenv:Header/>
  <soapenv:Body>
    <pet:Peticion>
      <pet:Usuario>appMOVILpush</pet:Usuario>
      <pet:Password>appMOVILpush</pet:Password>
      <pet:NombreLote>662_1230</pet:NombreLote>
      <pet:Servicio>662</pet:Servicio>
      <pet:Mensajes>
        <pet:MensajePush>
          <pet:Titulo>Titulo 2</pet:Titulo>
          <pet:Cuerpo>{"idServicioMovil":"2","title":"Newsletter del Portal de Administración
Electrónica", "bodyMessage":"Newsletter consulta tus notificaciones YA!", "url":
"http://google.com", "icono":""}</pet:Cuerpo>
          <pet:DestinatariosPush>
            <pet:DestinatarioPush>
              <pet:IdentificadorUsuario>662_1230</pet:IdentificadorUsuario>
            </pet:DestinatarioPush>
          </pet:DestinatariosPush>
        </pet:MensajePush>
      </pet:Mensajes>
    </pet:Peticion>
  </soapenv:Body>
</soapenv:Envelope>
```

La respuesta de salida tendrá la siguiente estructura:





# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



Un ejemplo de respuesta del endpoint, para la petición anteriormente mostrada:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <Respuesta xmlns="http://misim.redsara.es/misim-bus-webapp/respuesta">
      <Status>
        <StatusCode>1000</StatusCode>
        <StatusText>OK</StatusText>
        <Details>Petición procesada correctamente</Details>
      </Status>
      <Lote>
        <idLote>48642</idLote>
      </Lote>
      <Mensajes>
        <Mensaje>
          <idExterno/>
          <idMensaje>60872</idMensaje>
        </Mensaje>
      </Mensajes>
    </Respuesta>
  </soapenv:Body>
</soapenv:Envelope>
```

Cada una de las aplicaciones clientes que deseen enviar avisos push a través de la Plataforma de Mensajería y no dispongan de app móvil propia, se les habilitará un servicio específico en nuestra app móvil genérica (Mi Buzón gob.es). De esta forma, el usuario interesado en recibir avisos push se suscribirá en el servicio habilitado, y la aplicación cliente enviará los avisos a través de SIM al usuario suscrito.

### 5.1. Mensajes de datos / notificaciones silenciosas

El servicio de envío de mensajes de SIM permite, de acuerdo a sus especificaciones, el envío de mensajes de datos para Android y notificaciones silenciosas para iOS.

Para ello, los usuarios pueden utilizar las siguientes características del servicio web:

- **Mensajes de datos:** todos los parámetros informados en el campo <Cuerpo> de cada <MensajePush> a enviar se incluyen en el campo "data" del JSON de la notificación que se envía a FCM. Por tanto, las aplicaciones consumidoras del servicio pueden enviar en el campo <Cuerpo> los campos que consideren necesarios para que luego la aplicación móvil los procese.
- **Notificaciones silenciosas:** en la especificación de cada <MensajePush> existe un campo <NotificacionesSilenciosas> en el que se puede informar un 1 para activar este tipo de envíos propio del sistema iOS.



# SISTEMA INTEGRAL DE MENSAJERÍA

## Guía para envío de avisos push



### 6. Gestión del estado de los avisos push

La app móvil podrá especificar a la Plataforma de Mensajería SIM la lectura o cambio de estado de un aviso push. Para ello SIM expondrá el siguiente web service, que será invocado por la app móvil.

El webservice estará disponible en la siguiente dirección:

*Preproducción*

[https://pre-misim.redsara.es/misim-bus-webapp/rest/gestionNotificacionesPush?\\_wadi](https://pre-misim.redsara.es/misim-bus-webapp/rest/gestionNotificacionesPush?_wadi)

*Producción*

[https://misim.redsara.es/misim-bus-webapp/rest/gestionNotificacionesPush?\\_wadi](https://misim.redsara.es/misim-bus-webapp/rest/gestionNotificacionesPush?_wadi)

Como parámetros de entrada, habrá que especificar:

- **notificacionId:** identificador del aviso push
- **status:** estado en que se encuentra el aviso:
  - 1 = Recibido
  - 2 = Leído
- **idUsuario:** identificador del usuario en la Plataforma de Mensajería

Dicho web service dispone de Authentication Basic, recibiendo por este método el usuario y la contraseña de la aplicación que quiere actualizar el estado de un aviso push en la Plataforma de Mensajería.

La Plataforma de Mensajería responderá a la petición con los siguientes campos:

- **statusCode:** código de la respuesta
- **statusText:** descripción de la respuesta
- **details:** detalle de la descripción

En la siguiente figura se muestran los campos de la petición y respuesta:

