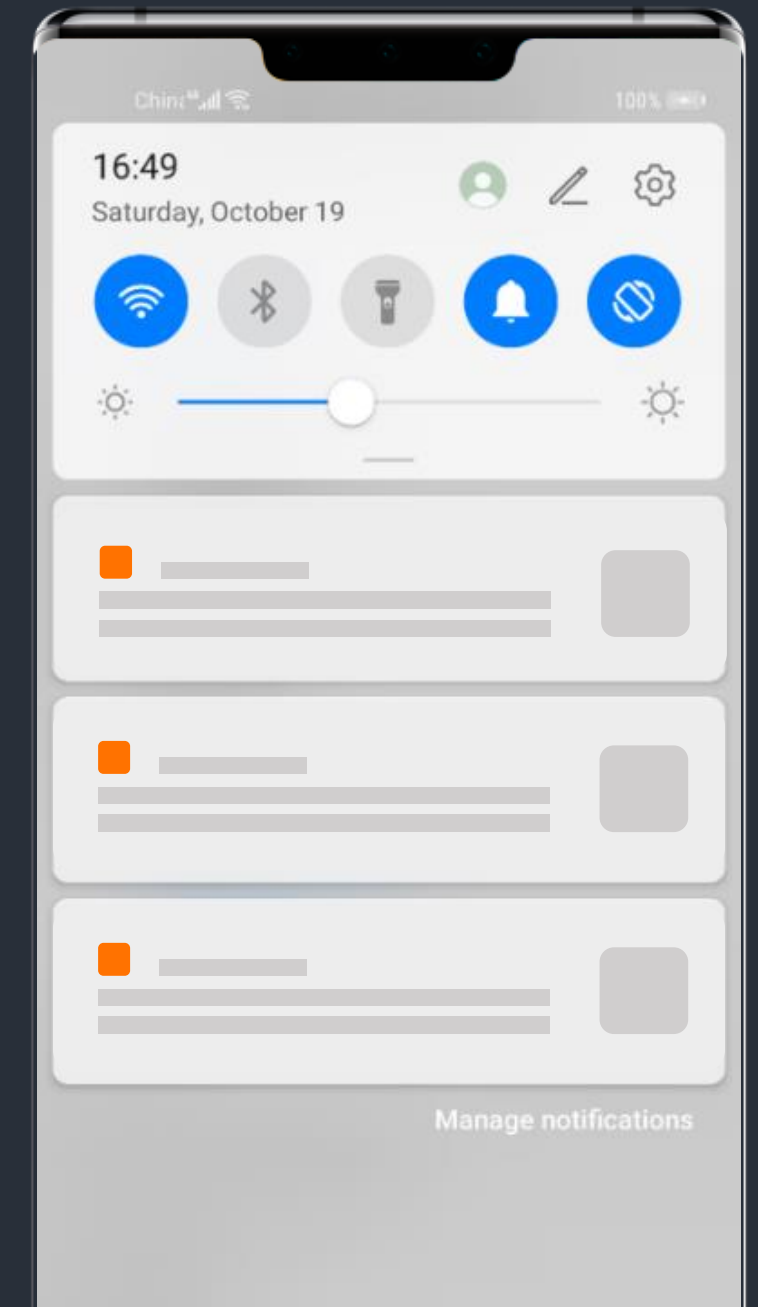
HUAWEI

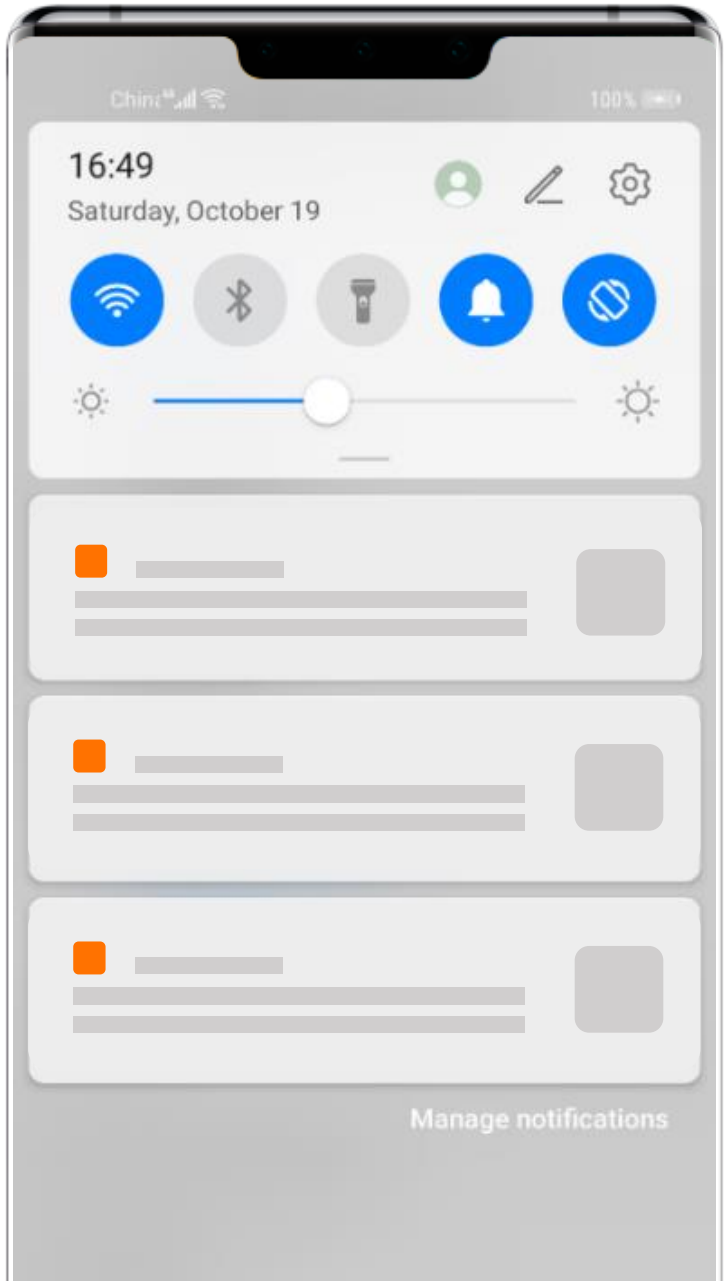
HMS INSIDE UNIVERSITIES

2021



CONTENIDO

HMS
NEARBY SERVICE



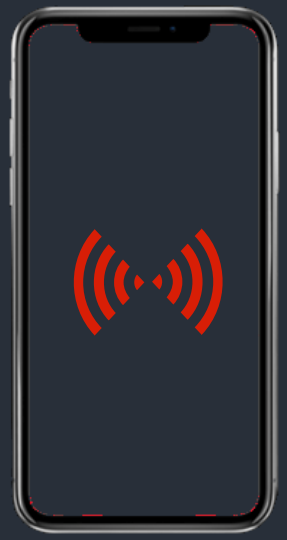
CONTENIDO


Introducción
Proceso de Integración
Preguntas y respuestas
1
2
3
4
5
6
Características
Ejemplo
Referencias

Huawei Nerby Service permite que las aplicaciones descubran fácilmente dispositivos cercanos y establezcan la comunicación con ellos mediante tecnologías como Bluetooth, Wi-Fi y Wi-Fi Direct.

Nearby Connection

- Descubre y transfiere datos con dispositivos cercanos.
- Los datos son cifrados en todas las conexiones



Nearby Message

- Permite a los suscriptores recibir códigos de uso compartido de editores y obtener información de Huawei Cloud.

Mutliplataforma

- A partir de HMS Core 5.0 se agregó soporte para Windows y IOS

Funciones Principales

Comunicación a corta distancia

- Descubrimiento
- Transmisión
- Networking
- Emparejamiento

Administración de balizas

- Registro del dispositivo
- Administración de mensajes
- Estadísticas de uso

Nearby Message

- Canal de mensajería
- Motor de mensajería

Ventajas competitivas

Descubrimiento rápido

Supere a la industria en términos de velocidad de descubrimiento de nodos.

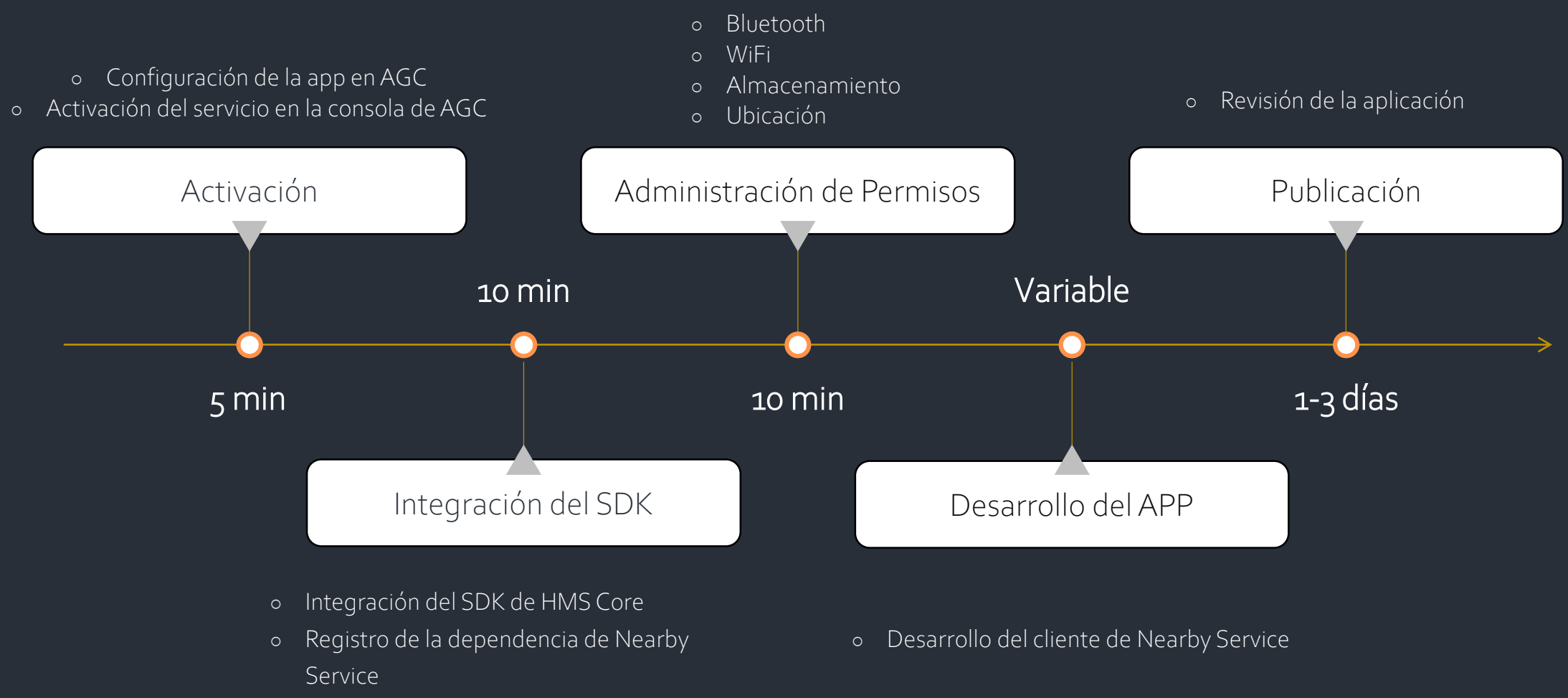
Alta velocidad de transmisión

Alcanza velocidades de 50 MB/s en transferencia de archivos.

Conexión a internet persistente

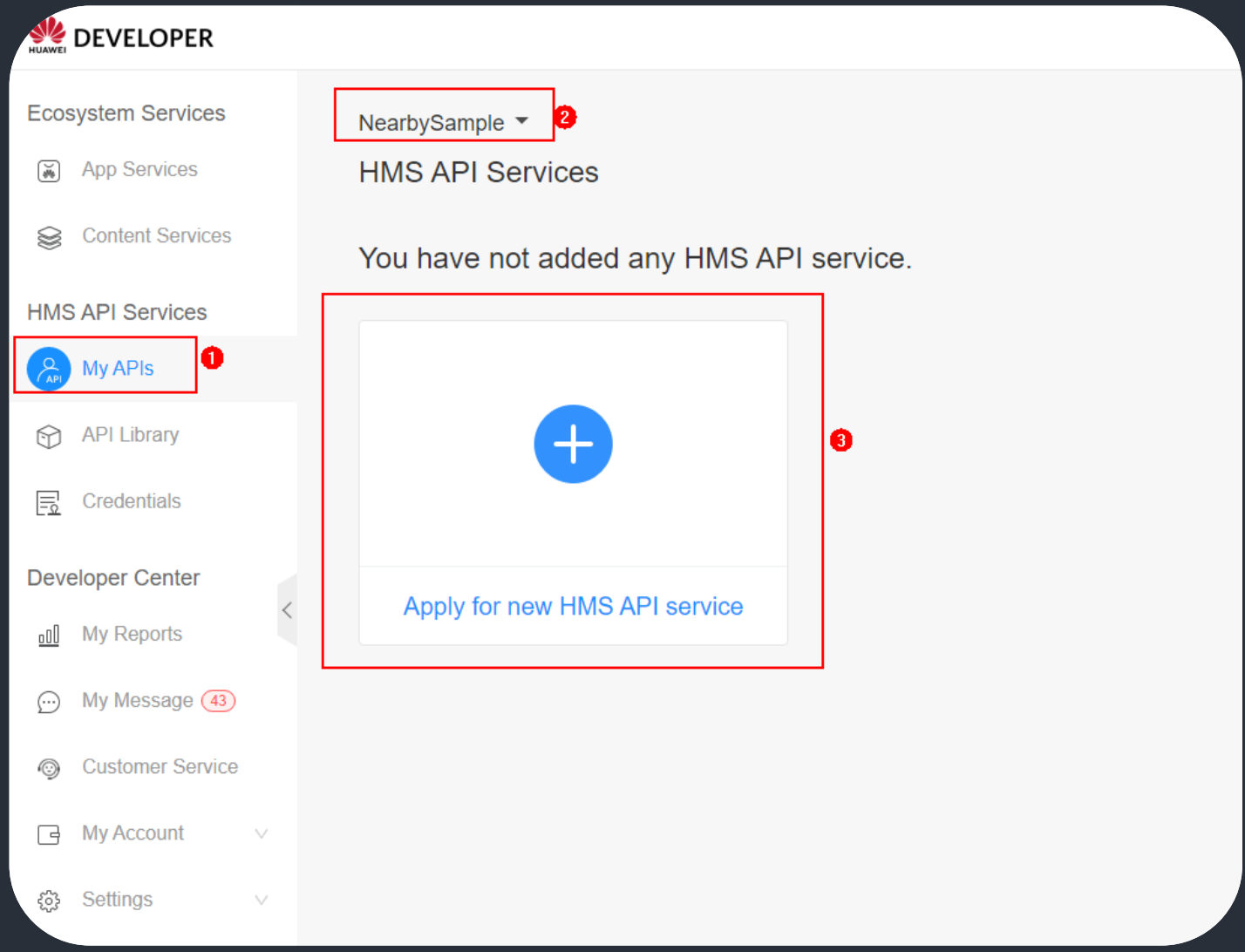
Los dispositivos no pierden su conexión a internet mientras usan Nearby Service

3 – Proceso de Integración



Activación del servicio

Puedes habilitar Nearby Service desde el panel de APIs en la consola de desarrollador o desde My projects en AppGallery Connect



Permisos

```
<!-- Permission to connect to the paired Bluetooth device -->
<uses-permission Android:name="Android.permission.BLUETOOTH" />
<!-- Permission to discover and pair with Bluetooth devices -->
<uses-permission Android:name="Android.permission.BLUETOOTH_ADMIN" />
<!-- Permission to check the Wi-Fi status -->
<uses-permission Android:name="Android.permission.ACCESS_WIFI_STATE" />
<!-- Permission to change the Wi-Fi status -->
<uses-permission Android:name="Android.permission.CHANGE_WIFI_STATE" />
<!-- Permission to obtain the coarse device location -->
<uses-permission Android:name="Android.permission.ACCESS_COARSE_LOCATION" />
<!-- Permission to obtain the accurate device location -->
<uses-permission Android:name="Android.permission.ACCESS_FINE_LOCATION" />
<!-- Permission to read the external storage -->
<uses-permission Android:name="Android.permission.READ_EXTERNAL_STORAGE"/>
<!-- Permission to write the external storage -->
<uses-permission Android:name="Android.permission.WRITE_EXTERNAL_STORAGE" />
```

Modo Cliente

```
public void startScanning() throws RemoteException {
        Log.e(TAG,"startScanning()");
        ScanOption.Builder discBuilder = new
ScanOption.Builder();
        discBuilder.setPolicy(Policy.POLICY_STAR);
        mDiscoveryEngine.startScan(serviceId,
scanEndpointCallback, discBuilder.build());
    }
```

Modo Servidor

```
public void startBroadcasting() throws
RemoteException {
        Log.e(TAG,"startBroadcasting()");
        BroadcastOption.Builder advBuilder = new
BroadcastOption.Builder();
        advBuilder.setPolicy(Policy.POLICY_STAR);
        mDiscoveryEngine.startBroadcasting(myName,
serviceId, connectCallback, advBuilder.build());
    }
```

DataCallback

Escucha el estado de la transmisión de datos

```
private final DataCallback dataCallback =
            new DataCallback() {
                @Override
                public void onReceived(String string, Data data) {
               }

                @Override
                public void onTransferUpdate(String string, TransferStateUpdate update) {
                }
            };
```

ScanEndpointCallback

Escucha los resultados de búsqueda de servicios activos

```
private final ScanEndpointCallback scanEndpointCallback =
            new ScanEndpointCallback() {
                @Override
                public void onFound(String endpointId, ScanEndpointInfo
discoveryEndpointInfo) {
}

                @Override
                public void onLost(String endpointId) {
}
            };
```

ScanEndpointCallback

Escucha los resultados de búsqueda de servicios activos

```
private final ScanEndpointCallback scanEndpointCallback =
            new ScanEndpointCallback() {
                @Override
                public void onFound(String endpointId, ScanEndpointInfo
discoveryEndpointInfo) {
}

                @Override
                public void onLost(String endpointId) {
}
            };
```

Enviando datos

Una vez que la conexión esté establecida, obtén una instancia de TransferEngine para comenzar a enviar información

```
mTransferEngine = Nearby.getTransferEngine(context);

Data data = Data.fromBytes(message.getBytes(Charset.defaultCharset()));
        Log.d(TAG, "myEndpointId " + endpointId);
        mTransferEngine.sendData(endpointId, data)
                .addOnSuccessListener((result) -> {
                    Log.i(“NearbyService”,”Data sent”)
                });
```

5 - Preguntas

6 - Referencias

Documentación de Nearby Service

Codelabs

Training