Documentación Técnica Final: Proyecto 27 AWS

Plataforma de Traducción Inteligente (CAT Tool)

Arquitecto Cloud AWS

3 de diciembre de 2025

Resumen

Este documento detalla la implementación completa de una arquitectura Serverless en AWS para traducción de documentos. A diferencia de versiones preliminares, esta entrega integra servicios de Inteligencia Artificial real (**Amazon Textract** para OCR, **Amazon Translate** para traducción neural) y un sistema de notificaciones desacoplado mediante **Amazon SNS**.

Índice

1. Arquitectura de la Solución

El flujo de datos implementado es el siguiente:

1. **Ingesta:** Usuario sube archivo (.txt o .pdf) a S3.
2. **Extracción:** Lambda detecta el formato; si es PDF usa Textract, si es TXT lee directo.
3. **Procesamiento:** Se segmenta el texto y se traduce con Amazon Translate.
4. **Persistencia:** Se guardan los segmentos bilingües en DynamoDB.
5. **Notificación:** Amazon SNS envía una alerta de finalización.
6. **Consumo:** Frontend web consulta los datos vía API Gateway.

2. Fase 1: Infraestructura Base

2.1. Almacenamiento (S3)

Bucket: `cat-tool-prod-v1`

- Carpeta `uploads/`: Disparador de eventos.
- Configuración: Event Notification activa hacia Lambda.

2.2. Base de Datos (DynamoDB)

Tablas en modo *On-Demand*:

- **TranslationProjects**: Estado del procesamiento.
- **TranslationSegments**: (PK: `project_id`, SK: `segment_id`). Almacena par origen-destino.

3. Fase 2: Backend con IA (Textract y Translate)

Se actualizó la función Lambda `cat-process-document` para manejar múltiples formatos y conectar con los servicios de IA reales.

3.1. Permisos IAM

El rol de ejecución fue actualizado con las siguientes políticas gestionadas:

- `AmazonTextractFullAccess`
- `TranslateFullAccess`
- `AmazonSNSFullAccess`

3.2. Código de la Función Lambda (Python 3.9)

```
import json
import boto3
import urllib.parse
import time

# Clientes AWS
s3 = boto3.client('s3')
dynamodb = boto3.resource('dynamodb')
translate = boto3.client('translate')
textract = boto3.client('textract')
sns = boto3.client('sns')

TOPIC_ARN = "arn:aws:sns:us-east-1:123456789012:TranslationAlerts"

def lambda_handler(event, context):
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])
    project_id = key.split('/')[-1]

    # 1. Extraccion de Texto (Logica Hibrida)
    full_text = ""
    if key.endswith('.pdf'):
        print("Detectado PDF. Usando Amazon Textract...")
        response = textract.detect_document_text(
            Document={'S3Object': {'Bucket': bucket, 'Name': key}}
        )
        # Concatenar lineas detectadas
        for item in response['Blocks']:
            if item['BlockType'] == 'LINE':
                full_text += item['Text'] + " "
    else:
        print("Detectado TXT. Leyendo directo S3...")
        obj = s3.get_object(Bucket=bucket, Key=key)
        full_text = obj['Body'].read().decode('utf-8')

    # 2. Segmentacion y Traduccion Real
    segments = full_text.split('.') # Segmentacion simple
    table = dynamodb.Table('TranslationSegments')

    with table.batch_writer() as batch:
        for i, source in enumerate(segments):
            if not source.strip(): continue

            # LLAMADA A AMAZON TRANSLATE REAL
            translation = translate.translate_text(
                Text=source,
                SourceLanguageCode='en',
                TargetLanguageCode='es'
            )
            target = translation['TranslatedText']

            batch.put_item(Item={
                'project_id': project_id,
                'segment_id': i,
                'source_text': source,
                'target_text': target
```

```
            })

    # 3. Notificacion SNS
    sns.publish(
        TopicArn=TOPIC_ARN,
        Message=f"El archivo {project_id} ha sido traducido exitosamente.",
        Subject="Traduccion Completada"
    )

    return {'statusCode': 200, 'body': 'Proceso Finalizado'}
```

Listing 1: Lógica de Procesamiento con IA Real

4. Fase 3: Sistema de Notificaciones (Amazon SNS)

Para notificar a los administradores o usuarios cuando un documento pesado termina de procesarse, se implementó Amazon SNS (*Simple Notification Service*).

4.1. Configuración del Tópico

1. Se creó un Tópico Estándar llamado `TranslationAlerts`.
2. Se creó una **Suscripción** de protocolo `Email`.
3. Se confirmó la suscripción mediante el enlace recibido en la bandeja de entrada.

Integración: La función Lambda, al finalizar el bucle de traducción, ejecuta el método `sns.publish()`, disparando el correo electrónico automáticamente.

5. Fase 4: Capa de Presentación (Frontend y API)

Para visualizar los resultados generados por la IA, se mantiene la arquitectura web.

5.1. API Gateway

Se expone una API REST con integración Proxy hacia Lambda para lectura de DynamoDB. Se resolvió el problema de serialización JSON implementando `DecimalEncoder`.

5.2. Frontend

Aplicación SPA alojada localmente (o en S3 Static Hosting) que consume la API. Permite visualizar lado a lado el texto original (extraído por Textract) y el traducido (por Translate).

6. Bitácora de Retos y Soluciones

Reto Técnico	**Contexto**	**Solución**
Procesamiento de PDF	S3 no permite leer contenido de archivos binarios/PDF directamente.	Se integró **Amazon Textract** para realizar OCR y extraer el texto plano antes de traducir.
Permisos de IA	La Lambda fallaba al invocar `translate_text`.	Se adjuntaron políticas `FullAccess` al Rol IAM para permitir el uso de los servicios de IA.
Notificaciones Asíncronas	Necesidad de saber cuándo termina un proceso largo.	Implementación de **SNS** al final del flujo de la Lambda.

Cuadro 1: Resumen de desafíos superados en la implementación final

7. Conclusión del Proyecto

El proyecto ha evolucionado de un prototipo simulado a una herramienta funcional de grado empresarial. La integración de **Amazon Textract** permite digitalizar documentos complejos, **Amazon Translate** ofrece traducciones de alta calidad sin intervención humana inicial, y **SNS** mejora la experiencia de usuario mediante notificaciones proactivas. La arquitectura es totalmente escalable y de pago por uso.