handytec®

We make it possible

Informe técnico

Proyecto	Estrategia de datos y plataforma de analítica
Cliente	Banco Solidario S.A.
Autor	Jonathan Arana
Versión	1.0
Fecha	2022-04-07
Estado	Final

Contenido

1. Antecedentes 3
2. Herramientas 3
2.1. Azure Repos 3
2.1.1. Flujo GIT 4
2.2. Azure Pipelines 5
3. Flujos 5
3.1. Ingesta 5
3.2. Ingeniería de Datos 6
4. Observaciones y Recomendaciones 8

1. Antecedentes

Como parte del proyecto "Estrategia de datos y plataforma de analítica", se ha realizado una estrategia de DataOps, la cual considera CI/CD (Continuos Integration and Continuos Deployment) para las diferentes fases del proyecto.

Para la implementación de la estrategia se ha optado por la herramienta Azure DevOps, tomando en cuenta la integración con todas las herramientas que se utilizarán en la arquitectura del proyecto.

- Azure Data Factory
- Azure Databricks

2. Herramientas

Azure DevOps es una suite de herramientas de Microsoft con integración nativa a su nube Microsoft Azure, dentro de esta suite se utilizarán las siguientes herramientas:

- Azure Repos
- Azure Pipelines

2.1. Azure Repos

Es un conjunto de herramientas de control de versiones que se puede usar para administrar el código.
Siempre es recomendable utilizar una herramienta de control de versiones.
Azure Repos proporciona dos tipos de control de versiones:

- Git: control de versiones distribuido
- Control de versiones de Team Fundation (TFVC): control de versiones centralizado

Git
GIT es el sistema de control de versiones más utilizado actualmente, muy robusto al ser distribuido, esto significa que la copia del código es un repositorio de control de versiones completo. Estos repositorios locales son completamente funcionales y permiten trabajar sin conexión o de forma remota.
El trabajo se confirma localmente, para luego sincronizar la copia con el servidor.
TFVC
TFVC es un sistema de control de versiones centralizado. Normalmente, los miembros del equipo solo tienen una versión de cada archivo en sus equipos de desarrollo. Los datos históricos se conservan únicamente en el servidor. Las bifurcaciones se basan en las rutas de acceso y se crean en el servido.

Para el proyecto se ha decidido utilizar repositorios GIT, por ser distribuido y las herramientas a utilizar tienen integración total.

2.1.1. Flujo GIT

Ramas principales/colaboración.
Por defecto Git crea la rama **main** se recomienda tener al menos 2 ramas.

- **main** (rama que contiene la versión estable productiva)
- **develop** (rama de desarrollo/integración para funciones)

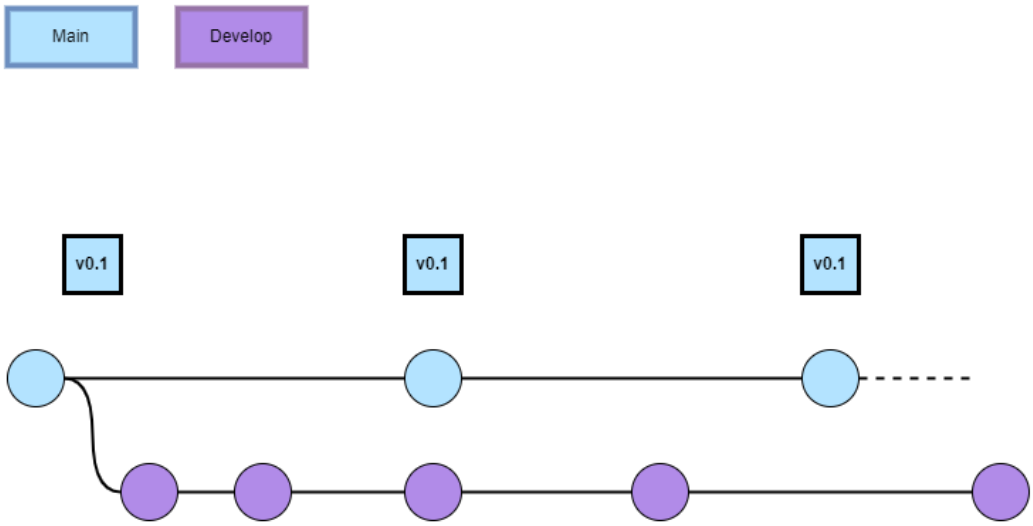

Diagrama 1: Ramas principales

Ramas de funciones
Para las funciones nuevas debe existir una rama propia, que se puede enviar al repositorio central para copia de seguridad/colaboración.
Pero deben ramificarse de la rama **develop**. Nunca deben interactuar directo con la rama **main.**
A estas ramas el estándar de Git las nombra **feature,** ejemplos de nombrado de ramas de función nueva.

- feature/nombreFuncionNueva
- feature/nombreTareaJira
- feature/nombreDominio

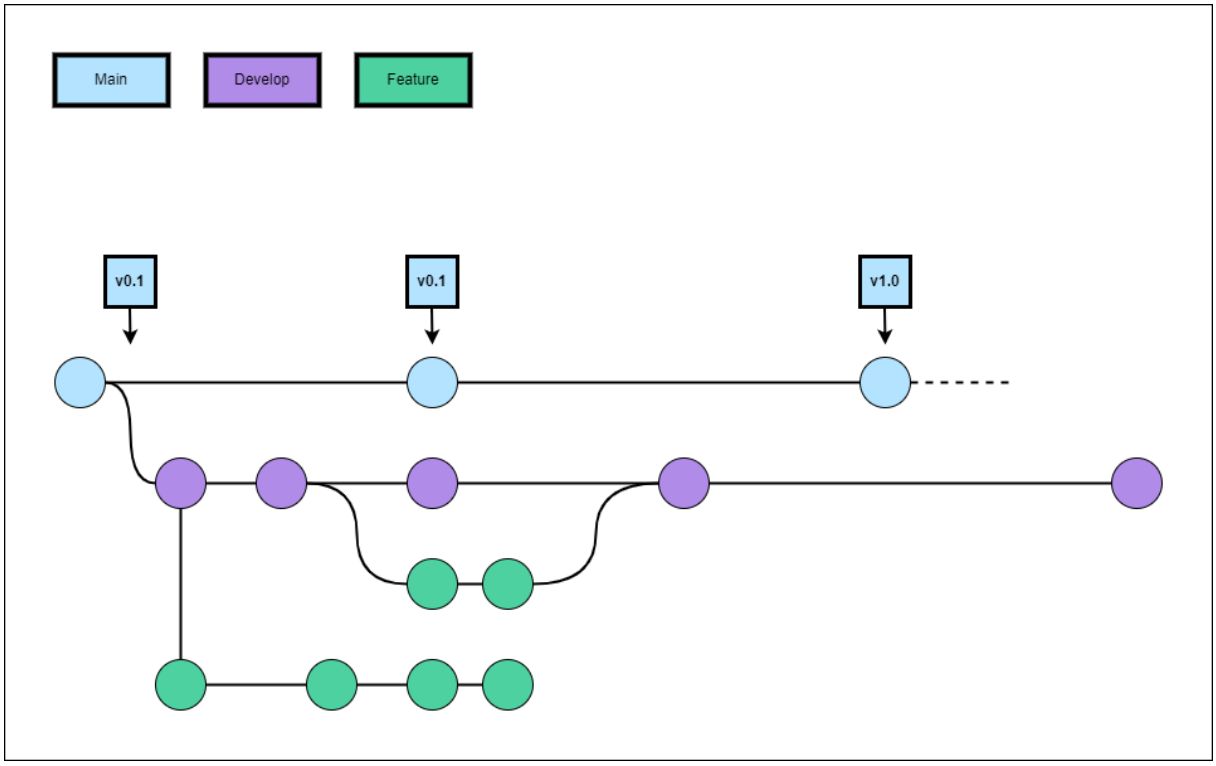

Diagrama 2:Rama de función

2.2. Azure Pipelines

Azure Pipelines compila y prueba automáticamente proyectos de código para que estén disponibles para otros usuarios.
Funciona con prácticamente cualquier tipo de proyecto y lenguaje de programación.
Poseen integración continua (CI) y despliegue continuo (CD).
Integración Continua (CI) es la práctica que usan los equipos de desarrollo de automatizar la combinación y prueba de código.
Despliegue Continuo (CD) es un proceso por el que el código se ha probado e implementado en uno o varios entonces de prueba y producción. Ayudando a incrementar la calidad del desarrollo.

3. Flujos

3.1. Ingesta

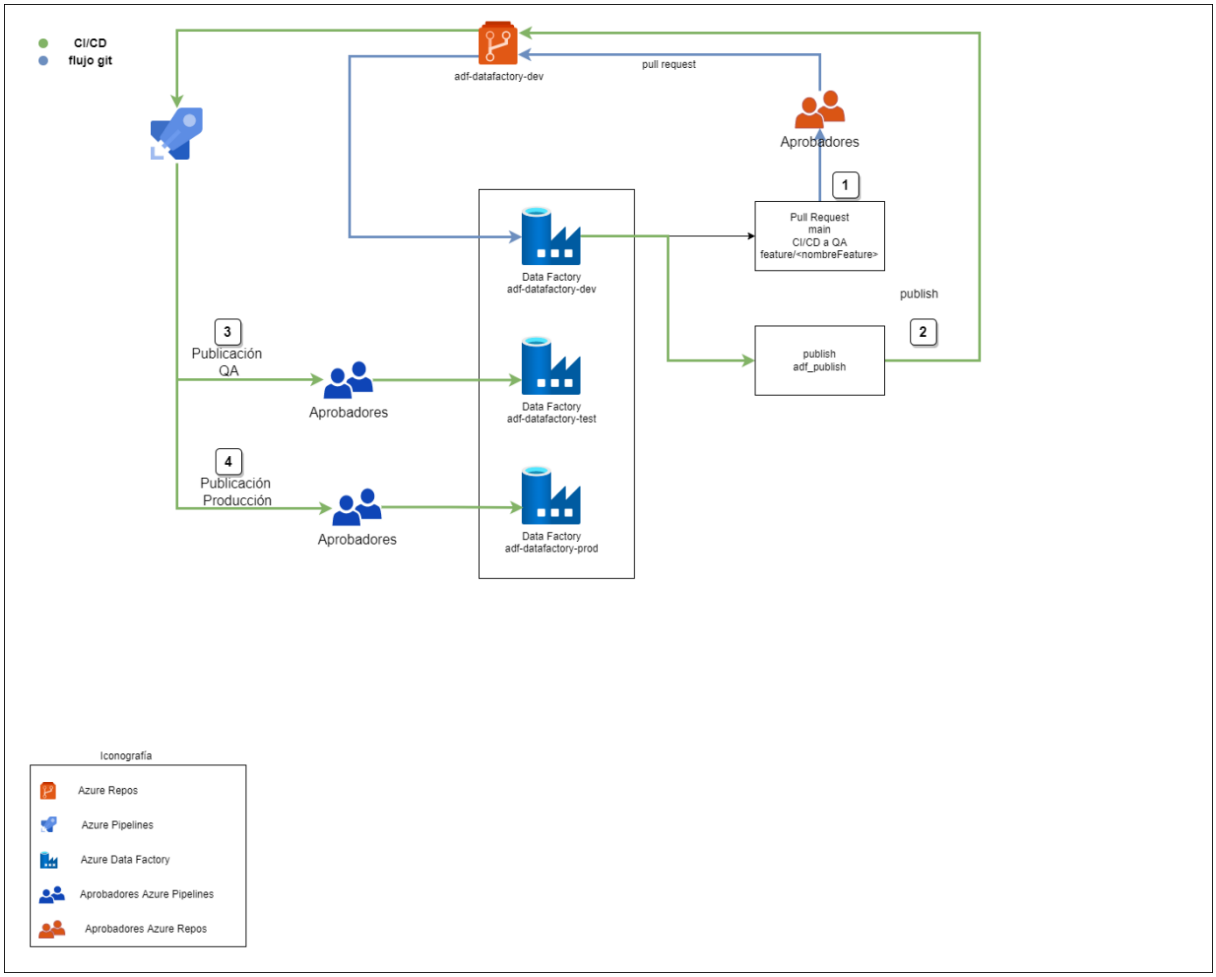

Diagrama 3: Ingesta

3.2. Ingeniería de Datos

Ambiente no productivo

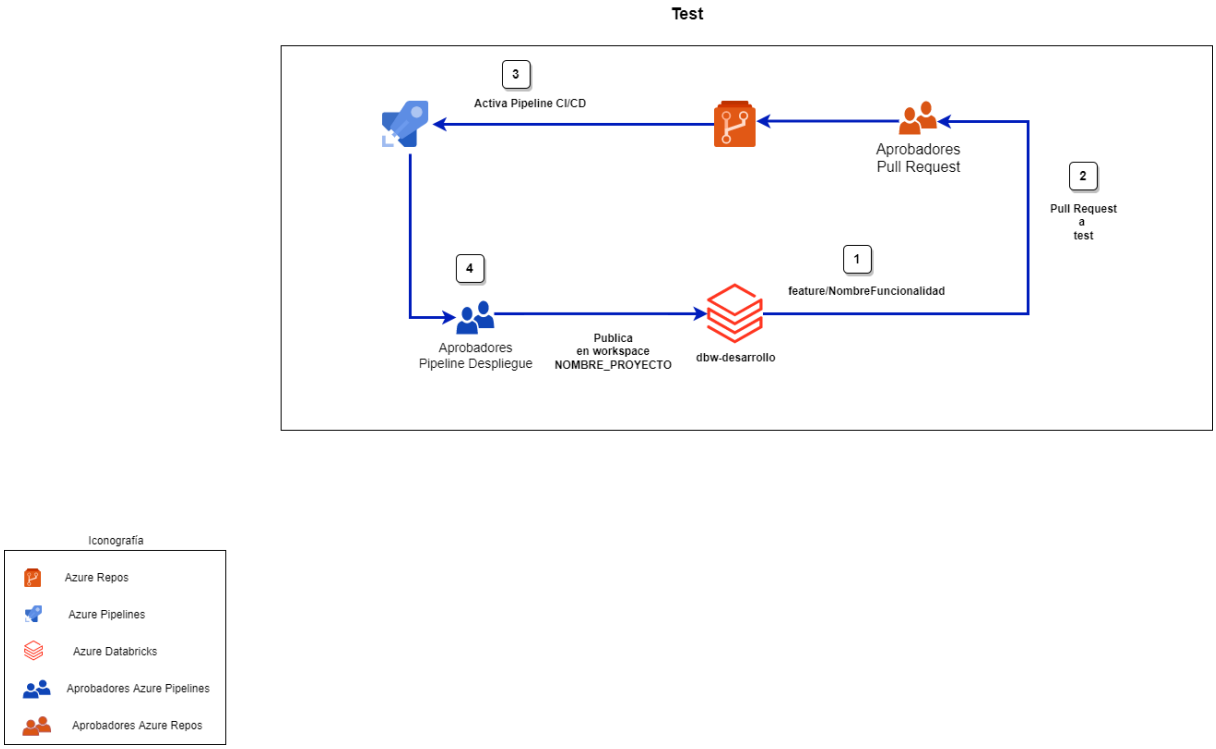

Diagrama 4: Ingeniería de datos test

Ambiente Productivo

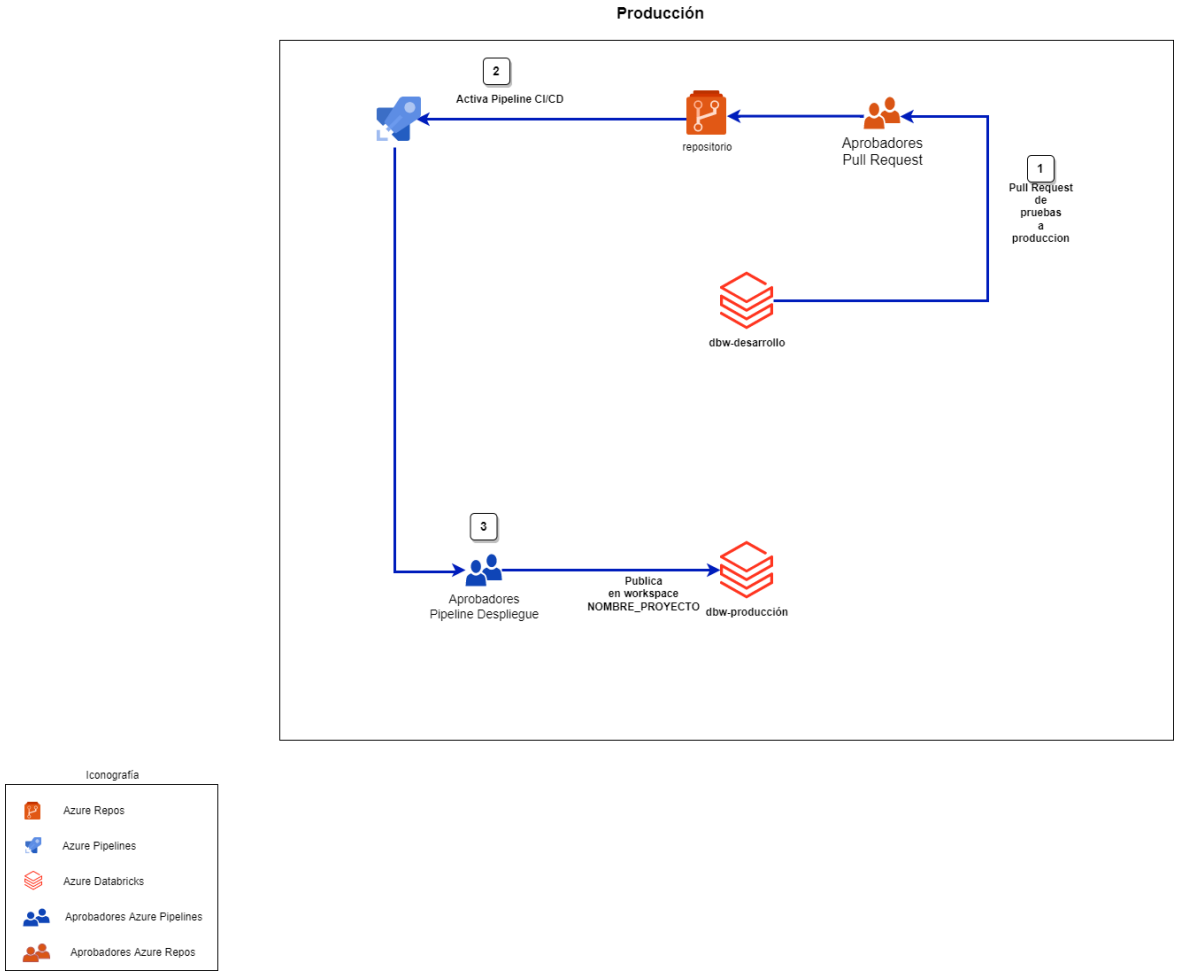

Diagrama 5: Ingeniería de datos productivo

4. Observaciones y Recomendaciones

- La herramienta Data Factory recomienda utilizar el flujo de CI/CD con una sola rama de colaboración, y no borrar la rama que se crea automáticamente llamada: **adf_publish**
- Se recomienda usar el flujo git de al menos 2 ramas de colaboración que represente a un ambiente productivo y no productivo.