**Cognite Functions Template for Time Series Analysis**

[**Version Control** 2](#_Toc55391430)

[1. Introduction 3](#_Toc55391431)

[1.1 Stakeholders 3](#_Toc55391432)

[1.2. Requirements 3](#_Toc55391433)

[1.3. Vocabulary 3](#_Toc55391434)

[2. Goals 3](#_Toc55391435)

[3. Existing Solution 3](#_Toc55391436)

[4. Solution Overview 3](#_Toc55391437)

[5. Application architecture 4](#_Toc55391438)

[5.1 Interfaces 4](#_Toc55391439)

[6. Information architecture 4](#_Toc55391440)

[6.1 Data sources 4](#_Toc55391441)

[6.2 Data model 4](#_Toc55391442)

[6.3 Data architecture 4](#_Toc55391443)

[6.4 Data quality and data validation 4](#_Toc55391444)

[7. Deployment 4](#_Toc55391445)

[8. Security Architecture 5](#_Toc55391446)

[8.1 Identity and Authentication 5](#_Toc55391447)

[8.2 Authorization and access control 5](#_Toc55391448)

[9. Technical Architecture 5](#_Toc55391449)

[9.1 Environments 5](#_Toc55391450)

[10. Operations 5](#_Toc55391451)

[10.1 Monitoring and alerting 5](#_Toc55391452)

[10.2. Logging 5](#_Toc55391453)

[10.3. Job scheduling 5](#_Toc55391454)

[10.4. Roles and Responsibilities 5](#_Toc55391455)

[11. Known Gaps and Issues 6](#_Toc55391456)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version Control** | | | | |
| Version | Date | Change Descritpion | Responsible | Approver |
| 1.0 | 07.09.23 | Project scoping | Vetle Nevland |  |
| 2.0 | 06.10.23 | First Cognite Function deployed | Vetle Nevland |  |
| 2.1 | 21.11.23 | Generalization to multiple time series inputs/outputs | Vetle Nevland |  |
| 2.2 | 05.01.24 | Support for date-specific aggregates | Vetle Nevland |  |

# Introduction

*Problem statement - Here we shall describe what the problem is and what are we trying to achieve, source systems, use stories (functional requirements, business rules).*

The goal of this project is to generate a template for using Cognite Functions to read time series from CDF, perform transformations using the Cognite Python SDK, populate new time series object(s) with the transformed data, and finally deploying the time series to a dataset (here: Center of Excellence – Analytics, devoted to data produced using this template). The template supports transformation of single and multiple time series inputs and outputs, and endless calculation possibilities from desired libraries in Python.

## Stakeholders

|  |  |  |
| --- | --- | --- |
| **#** | **Stakeholder** | **Role** |
| 1 | Stephanie Raoul | CoE Analytics dataset owner.  Peer reviewer. |
| 2 | Vetle Nevland | Development of Cognite Functions template for time series |
| 3 | Anders Brakestad | Peer programmer.  Peer reviewer. |
| 3 | CDF Operation Team | Access granting and creation of dataset |

## 1.2 Requirements

*Use cases and requirements should be detailed here*

Running the project requires read/write access to CDF resources: Time Series, Files and Functions, installation of Cognite Python SDK and basic knowledge of Python. See the project’s GitHub page ([AkerBP-DataOps/deos-cognite-functions-template (github.com)](https://github.com/AkerBP-DataOps/deos-cognite-functions-template)) for detailed requirements.

Potential use cases are broad:

* Easily set up basic to advanced calculations of arbitrary time series, avoiding cumbersome preprocessing of data
* Deploy new time series to CDF and let the associated calculations run on a prescribed schedules
* Develop basic machine learning workflows for high-frequent prediction of time series data

## 1.3 Vocabulary

*Terminology which the target audience might be unfamiliar with and that this document understanding depends of. Some examples below.*

|  |  |
| --- | --- |
| **Term** | **Description** |
| CDF | Cognite Data Fusion |
| Asset | CDF resource type which acts as a digital representation of objects from the physical world |
| Time Series | CDF resource type to store data points in temporal sequence |
| SME | Subject Matter Expert |
| SDK | Software Development Kit |
| Dev | Development |
| Prod | Production |

# Goals

*KPIS*:

Make a versatile framework for easily setting up and perform advanced transformations on arbitrary time series data, and schedule the calculations to run and update the time series at desired frequency.

KPIs:

* Reduce time to insight through rapid deployment of desired calculations on arbitrary time series, mitigating time used for preprocessing and organizing data
* Lower the threshold for data scientists and SMEs to independently run semi-advanced transformations of time series that is otherwise not available/possible in CDF Charts

# Existing Solution

*Describe current solution if exists*

User interfaces such as Charts and applications like Grafana already exists for visualization and analysis, but is limited to rather basic transformations. Only very simple templates for deploying Cognite Functions exist, with no particular use case.

# Solution Overview

*Describe the solution to the problem statement, including CDF resources and core functionalities as well as all infrastructure and resources used outside CDF which are used/deployed in the solution. Be sure to include diagrams which will make the understanding of the overall design easier to the reader.*

The solution involves extracting time series from CDF, perform transformations using Cognite Python SDK and deploy the new time series to a devoted dataset in a CDF project using Cognite Functions scheduling. The solution also details the process of granting required read/write access from/to CDF in order to permit deployment. The solution is summarized as follows:

1. Install the Cognite Python SDK to your development environment
2. Create an account at Cognite Hub
3. Apply for write access to designated dataset in Cognite Fusion Dev (akerbp-dev).
4. Using Cognite Python SDK:
   1. Interactively authenticate with a Cognite Client
   2. Extract desired time series
   3. Create a Cognite Function that performs transformations on the input time series to produce a new output time series
   4. Populate the new dataset with the new time series data
   5. Deploy schedule to frequently update the signal
5. Apply for read/write access to CDF’s testing environment (akerbp-test)
6. Testing phase:
   1. Set up tests to validate quality of our solution
   2. Run the tests
   3. Proceed when all tests have passed
7. Apply for write access to CDF’s production environment (akerbp)
8. Finalize the dataset, ensure governance and deploy to the akerbp tenant

# Application architecture

The overall architecture and purpose of this solution, highlighting specific design or technological features and components. Include diagrams where appropriate.

Simplified, the architecture can be depicted as a workflow loop where Cognite Fusion Prod marks the start and end destination. Dataset of interest is extracted from Cognite Fusion Prod, transformed with Cognite Functions using the Python SDK, written to Cognite Fusion Dev to finalize development, then to Cognite Fusion Test to run test procedures, and eventually deployed as a new time series in a governed dataset in Cognite Fusion Prod. Once the new time series is deployed, and the governed dataset resides in CDF, time series can be visualized/analyzed in integrated or external dashboards like Charts or Grafana.

## Interfaces

Describe if any.

The diagram below illustrates the architecture.

A diagram of a process

Description automatically generated

# Information architecture

*Define the information strategy or business solution through the definition of the company’s business information assets, their sources, structure, classification and associations.*

## Data extraction

Data is extracted from the PI Time Series dataset in CDF Clean.

## Data sources

*Include all data sources, including CDF resources as well as any outside data used in the solution.*

The primary data source is PI, which ingests time series data into CDF Clean through PI Extractors, then converted to CDF Time Series objects with subsequent contextualization (no modifications to this end).

## Create new dataset

A particular dataset, Center of Excellence – Analytics, is to be dedicated to the time series produced in this project. As such, the CDF Operations team creates this dataset for us after our request for write access has been accepted.

## Data transformations

Initial transformations have already been performed to contextualize the data from CDF Raw to CDF Clean as a Time Series resource type. The transformations that are referred to in this project are done through the Python SDK, which we interchangeably refer to as “calculations”. The calculations can be implemented using any desired Python library.

## Data model

*How the data will be modelled in CDF*

No modifications are done on the source data, metadata and other data connected in a CDF model. Relationships with assets are sustained.

## Data quality and data validation

*Standards of data quality, operations, and Service Level Agreements (SLAs).*

Data quality and integrity is accounted for through the SIT testing framework. See separate documentation of testing.

# Deployment

*The necessary steps and tools to deploy the solution*

Deployment of the complete dataset is done internally in CDF with the necessary write access.

Cognite Functions are deployed by generating a designated folder for your calculation in the project template, and then run the Jupyter notebook “run\_functions.ipynb” with relevant input parameters for the “handle.py” script. See the project’s Github page, [AkerBP/deos-cognite-functions-template (github.com)](https://github.com/AkerBP/deos-cognite-functions-template), for deployment guidance.

# Security Architecture

## Identity and Authentication

*On CDF, the solution follows the existing identity and authentication regime.*

The solution follows the existing identity and authentication regime as implemented in CDF. Authentication using tokens and secrets is conducted through the Cognite Python SDK. Details are provided on the project’s GitHub page: [AkerBP-DataOps/deos-cognite-functions-template (github.com)](https://github.com/AkerBP-DataOps/deos-cognite-functions-template)

## Authorization and access control

*For data ingestion into CDF, a service account, api key, access groups with the required capabilities, and finally a dataset in which the data will be written is required. Please provide the information below.*

There are no specific access restrictions on the produced dataset. General access permissions apply.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stage** | **Tenant** | **Service account** | **Authorization** | **Data Set** |
| Development |  |  |  |  |
| Test |  |  |  |  |
| Prod |  |  |  |  |

# Technical Architecture

## Environments

*Technologies in which the solution is deployed and what runs where (ex: google cloud, windows server, etc)*

Transformations with Python SDK are performed in designated Python classes and run in a Jupyter Notebook in VS Code. Deployment of Cognite Functions are realized through authentication and connection with an Azure AD tenant.

# Operations

## Monitoring and alerting

**Technical:**

**Business:**

## Logging

## Job scheduling

## Roles and Responsibilities

*Contacts: Tech Leader, SME and Support.*

# Known Gaps and Issues