

Continuous Deployment for Data Science (CDDS) Extension for KNIME Business Hub

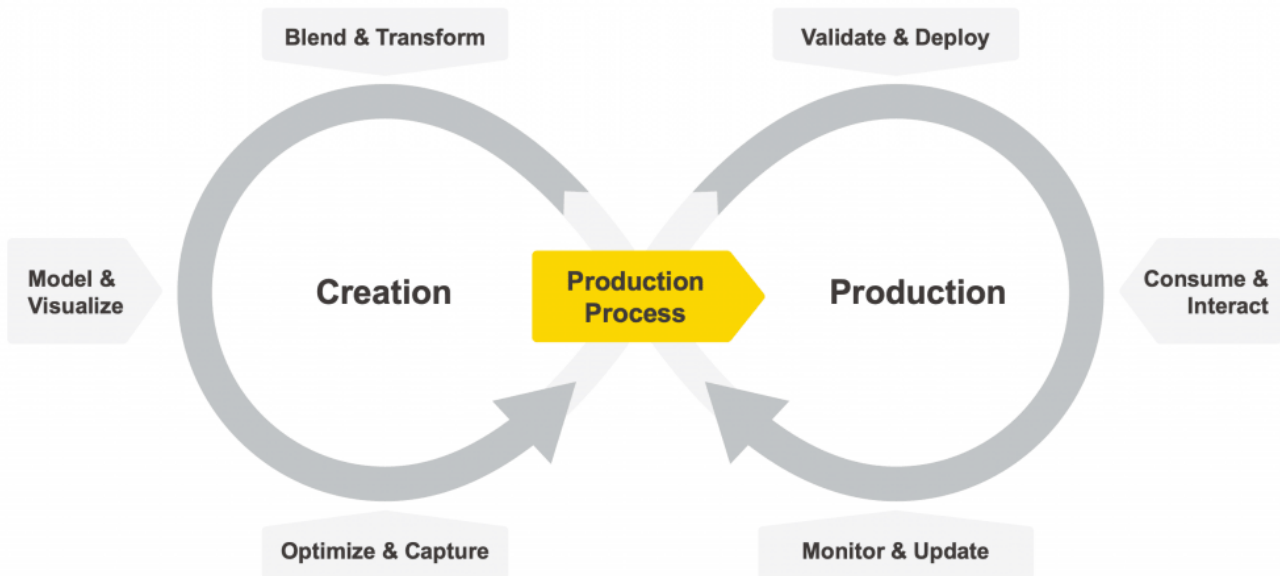
KNIME AG, Zurich, Switzerland
Version 1.9 (last updated on 2023-07-19)



Table of Contents

Introduction	1
Concepts Overview	2
Workflow Project	2
Stages	2
Event Log	3
Environments and their Separation	3
CDDS on the Hub	5
Spaces	5
Execution Contexts	5
Admin Guide	6
User and access management	6
Automation vs. Interaction	6
Event logging	9
User Guide	10
Developing workflow projects	10
Deploying projects via the user data app	12
Continuous Deployment Sophistication	14
Level 1: Validation of a production workflow	14
Level 2: Automatic creation of the production workflow	14
Level 3: Monitoring of the production workflow and automatic re-training	15
Workflow Project	15
Training Environment	16
FAQs	17
Installation Guide	20
Prerequisites	20
Installation Workflow	21

Introduction



Companies are continuously creating new and updating existing data science applications, often based on Machine Learning models. Administrators of these applications need to ensure that these applications follow corporate governance practices before they can be moved into production. With an increasing number of applications, this task becomes more and more difficult and time consuming for the administrator. Automated processes are needed that enable a continuous deployment of data science applications. The aim of CDDS is to solve this problem.

Unlike traditional devops, moving data science into production requires additional functionality. Looking at the [data science lifecycle](#), CDDS is an extension to KNIME Business Hub that intends to provide a solution for:

- **Loading** and Registering workflow packages that are deemed ready for production.
- **Validate & Deploy**: Validate production workflows before they get deployed.
- **Monitor & Update**: Monitor the results being generated by a deployed production workflow and, if required, automatically update and re-deploy it.

Although most organizations share similar concerns regarding their deployment practices, the deployment process itself can vary greatly. Factors such as the size of the organization, industry, resources, and IT and governance requirements determine what the deployment process needs to look like. Therefore, the CDDS extension provides a customizable solution. The extension is built with KNIME workflows which can be adjusted and extended.

This guide shows what the extension is capable of and how it could be used. It also explains how the extension can be adjusted or extended to personal needs.

Concepts Overview

CDDS uses concepts that are explained here. It has been designed to run on KNIME Business Hub and leverages its concepts and features, such as spaces, execution contexts and deployments. See the [user guide](#) for KNIME Business Hub.

Workflow Project

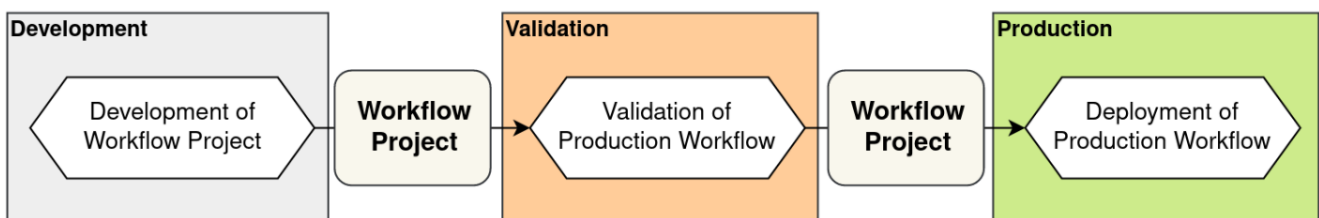
CDDS does not process single workflows but workflow projects that contain potentially multiple workflows. In the simple scenario this will just be a single “production workflow”, i.e. the workflow that should be deployed in production. However, in more sophisticated scenarios there are more workflows part of a workflow project, as described in a [later section](#).

The workflow project concept is used throughout this document. For now, we only talk about the simple scenario where a workflow project contains a single production workflow.

It is important to note that workflow projects are distinguished by their names, implying that each project must have a unique name. Deploying a workflow project with the same name again, results in the previous deployment being updated.

Stages

Deploying a workflow project into production means going through multiple stages. The entire project (containing all its directories and associated workflows) is moved across the stages. The following diagram shows the flow between the development, validation and production stages.



- 1. Development:** The data scientist develops a production workflow and creates a workflow project that should be deployed to production. During development, the data scientist tests the production workflow and verifies that it is performing its intended function. Once done, they manually start the deployment process by putting the workflow project into validation.
- 2. Validation:** The production workflow is validated. If validation passes, the workflow project moves to the production stage.

3. **Production:** The production workflow gets deployed.

Event Log

It is important to keep track of exactly what happens at any changed state through the process, not only of the process being triggered but exactly what the results of that action are. To accomplish that, the CDDS extension uses a database to log events. For example, the execution of the validation process on a project would contain a record of that process being executed as well as detailed results of that process if, for example, it failed. Throughout the CDDS extension, event logging is implemented. It can of course be extended or modified as required by an organization.

Environments and their Separation

Usually, it is desired to keep different stages separated from each other. This makes on the one hand sense as different stages require different computational resources or access to different data sources. On the other hand, separating the stages into different environments improves the overview and observability of the processes and can aid in managing security and access.

As shown in the figure above, the process is separated into three stages. Accordingly, there are three environments.

- **Development Environment:** Workflow development is done by data scientists using KNIME Analytics Platform, while KNIME Hub is used for collaboration and manual testing. As an entry point for CDDS, there is Development space on the Hub required where data scientists save their workflow projects once ready for production.
- **Validation Environment:** Validation is usually a lightweight task but should not be executed in the production environment.
- **Production Environment:** Only validated deployments should live here.

The environment separation can, again, be adjusted. The different stages do not need to be separated into exactly three environments. The CDDS extension can be adjusted to fit organizational needs and preferences.

The movement of workflow projects between different environments should be initiated by the destination environment, particularly when transitioning from validation to production. It is important to consider security and access when transferring workflow projects, as the validation environment should not be permitted to load projects into production. Instead, the production environment should have permission to access the validation environment and

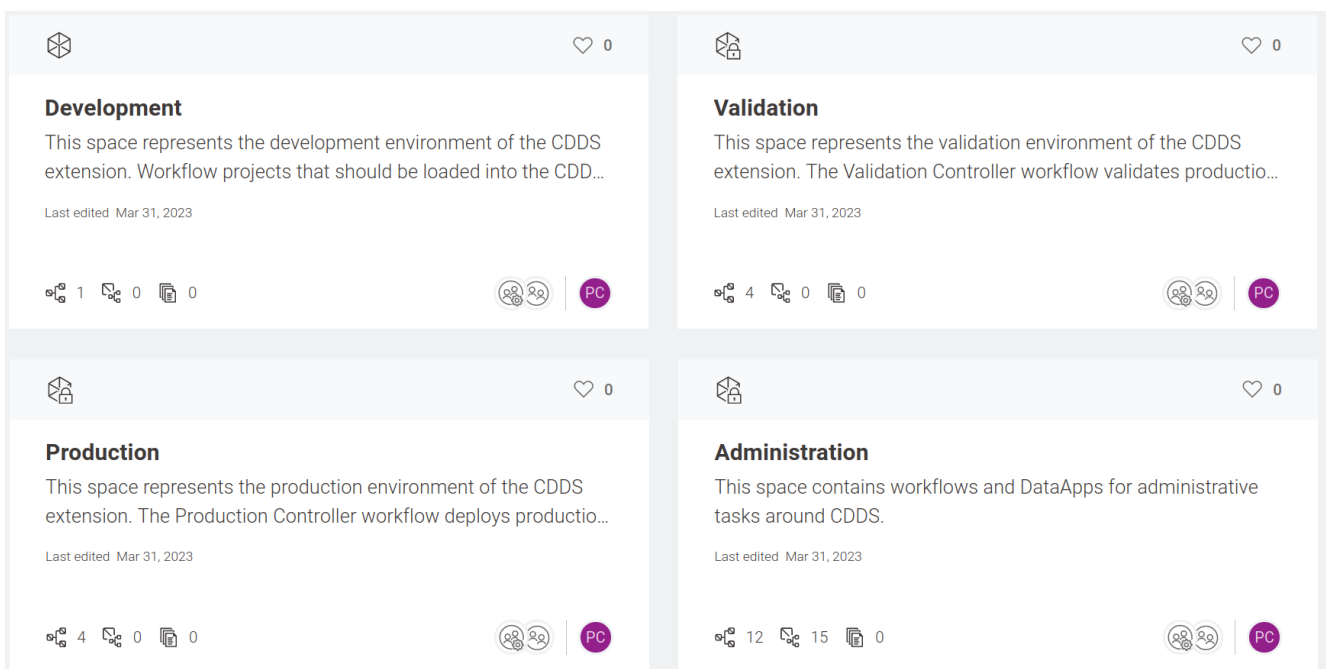
retrieve workflow projects that have been successfully validated.

CDDS on the Hub

Once the CDDS extension is **installed in KNIME Hub**, the concepts of stages and environments get materialized by spaces.

Spaces

An environment is represented by a space on KNIME Hub. Different environments could live in the same team or in different ones. They could even be separated across different Hub installations. In the default setup, all the spaces live in the same team.



On the Hub, environments are separated by spaces. In the default setup, a space for each environment exists and all spaces live in the same team.

In addition to the environment spaces (Development, Validation, Production), there is also an Administration space. This space contains data apps and workflows that support the admin observing the CDDS processes and states.

Execution Contexts

Besides segregating the environments' locations, it is recommended to maintain their execution separated. Each environment may require different computational resources and system access. On KNIME Hub, this separation is achieved with **execution contexts**.

Admin Guide

Admins are responsible for the training, validation and production environments.

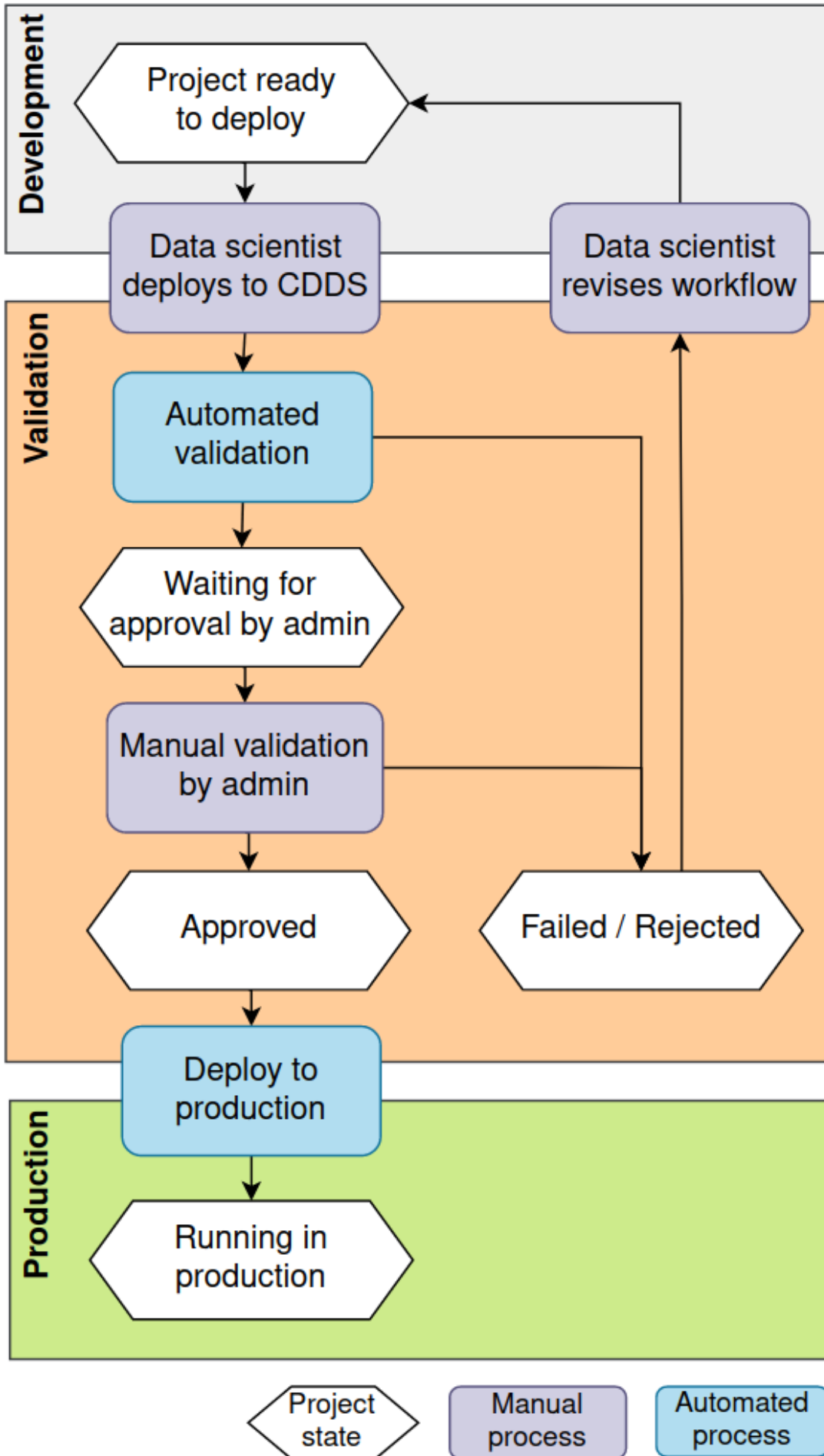
User and access management

The access to each space can be customized to specific users. It is recommended to limit access to the Training, Validation, and Production spaces to admins. Data scientists will only be granted access to the Development space.

The Administration space contains a workflow named “CDDS User Data App”. This workflow is supposed to be deployed as a data app by the admin and **shared** with team members that will develop workflow projects and load them into CDDS. It is basically the gateway for users to interact with the CDDS extension. The workflow is configured during the installation process with the application password of the admin. These credentials are used to allow a user to load a workflow project into the Validation space which has access restricted to the admin.

Automation vs. Interaction

The CDDS extension automates many processes using **triggers** and allows users and admins to observe project states and interact with them using **data apps**. The following is an overview of the states a project can be in and of the automated and manual processes that move a project between the states.



1. Once the development of a project has been finished, it is **ready to deploy**. The data scientist uses a data app to **deploy the project to CDDS**. See also the user guide.
2. The project will **automatically be validated** by an automated workflow. General and project-independent validations are performed, such as checking whether the production workflow contains deprecated nodes. If the automated validation fails, the project is **rejected**. If the validation is successful, the project is **waiting for manual approval by the admin**.
3. The admin can inspect the project waiting for **manual validation** in a data app and either **approve** or **reject** it. If approved, the deployment will be automatically **deployed to production**.
4. Once deployed, the project is **running in production** and available.
5. Rejected projects and their reasons for rejection can be inspected by the data scientist in the user data app. The data scientist will **revise the project and its workflows** accordingly and re-deploy it.

Admin data app for interaction

The admin data app allows the admin to observe the validation and production stages. It shows the projects either waiting for validation and the projects running in production. Projects waiting for validation can be either approved or rejected. If rejected, a reason needs to be provided that instructs the data scientist how to improve the project. Projects running in production can be inspected and removed if desired.

The screenshot displays the 'Admin View' of the Continuous Deployment for Data Science (CDDS) interface. At the top, there is a 'Refresh' button and a 'Data Science Team' dropdown menu. The main content is divided into two panels: 'Validation' and 'Production', each showing 2 projects.

Validation Panel:

- Header: Validation (2 Projects)
- Project to validate: Document Classifier, Fraud Detection Scorer
- Rejection reason: Rejected because... (empty text area)
- Review the workflow: Document Classifier (with a help icon), Submitted on: 2023-04-14 12:22
- Buttons: Approve, Reject

Production Panel:

- Header: Production (2 Projects)
- Select deployment: Churn Prediction, Customer Segmentation
- Selected deployment: Churn Prediction (with a help icon)
- Running since: 2023-04-14 12:20
- Buttons: Deactivate, Remove

The admin data app shows projects in validation and production and offers different interactions.

Event logging

For each event within the CDDS extension, such as a workflow project moved to another state or validation of a workflow project failed, a record is written as log entry into the configured database. In the Administration space under the Data Apps folder, the Event data app can be found. It can be run to monitor the events written to the database.

User Guide

Users develop workflow projects and deploy them to CDDS via the user data app. Once deployed to CDDS, the state of the project can be observed. If a project is rejected during validation, it needs to be revised and improved.

Developing workflow projects

A workflow project follows a certain folder structure to be able to be processed by CDDS.

1. Create a new folder with a name describing the project (e.g. "Fraud Detection"). Project names must be unique.
2. Create subfolders for each workflow being part of the workflow project.
 - a. A *Training* folder if the project should contain a training workflow.
 - b. A *Production* folder if the project should contain a production workflow.
 - c. A *Monitoring* folder if the project should contain a monitoring workflow.
3. Create the workflows (workflow names are not constrained) and place them inside the corresponding subfolders. There should be only one workflow in each subfolder. See the [Workflow Structures](#) section below for information on how to create the workflows.

Depending on the use case and type of project, different workflows might be required. The table below provides an overview of which workflows are required for which kind of workflows.

Project ☒/ Required workflows ☒☒	Production ☒	Training ☒	Monitoring ☒
Level 1: ETL workflow	☒		
Level 1: Machine Learning w/o integrated deployment	☒		
Level 2: Machine Learning w/ integrated deployment		☒	
Level 3: Machine Learning w/ integrated deployment and monitoring		☒	☒

Note that if your project only contains a production workflow, a "Production" folder is not required. The workflow could just be placed within the first level of the project folder.

Workflow Structure

The CDDS extension does not require a production workflow to have a certain structure. It can be any workflow that needs to be deployed to production, e.g., an ETL workflow or a prediction workflow. It does require that all data and external access required by the workflow is defined within the workflow - in other words, the workflow is self-contained to be run.

The CDDS extension expects Training and Monitoring workflows in the project structure to follow certain design patterns. CDDS will fail if the specific design patterns are not respected.

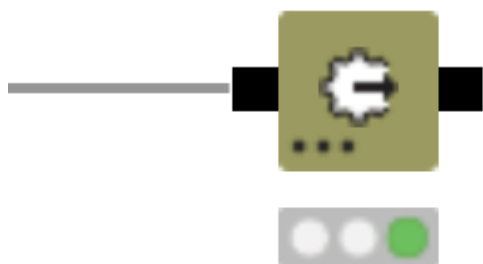
Training Workflow

The training workflow will be executed by CDDS and is expected to train an ML model and produce the prediction workflow that will later be deployed as the production workflow. #

For the creation of the production workflow, the [Integrated Deployment](#) extension should be used. Examples making use of the extension can be found on [KNIME Hub](#).

The last node of the training workflow is expected to be a [Workflow Service Output](#) node, configured with a *Workflow Port Object* as input and output. It outputs the created production workflow.

Workflow Service Output



The training workflow needs to output the created prediction workflow.

A concrete example workflow can be found in the Administration space in the “Examples” directory. The “Level_2_Fraud Detection with ID” example contains a corresponding training workflow.

Monitoring Workflow

The monitoring workflow will be executed by CDDS and is expected to evaluate the performance of the production workflow. The subject of evaluation depends highly on the use case, e.g. the data or model might be checked for drift.


CDDS expects the monitoring workflow to output the information whether a re-training is required and should be initialized. The “Configure Monitoring Output” component should be used to create the proper output. It can be configured via its dialog and allows the user to select a boolean column containing the information whether a re-training is required (true if required, false otherwise). Optionally, a string column can be selected containing information that will be logged by CDDS. This information could contain, e.g., the reason why re-training is required (or not). Only the first row of the columns is respected.


An example monitoring workflow that can trigger a relearning can be found in the Administration space in the “Examples” directory with name “Level_3_Fraud Detection with Monitoring”.

Deploying projects via the user data app

The user data app allows users to observe the projects in the development, validation and production stages. Projects in development can be deployed and rejected projects and their reason can be inspected.

Continuous Deployment for Data Science (CDDS) Refresh ↻


 [Data Science Team](#) | User View ?

 **Development** 6 Projects ?


Project to deploy

Churn Prediction
Customer Insights (ETL)
Customer Segmentation
Document Classifier
Fraud Detection Scorer
Quarterly Finance Report

Submit

 **Validation** 1 Projects ?


Submitted on	Project
2023-04-14 12:22	Fraud Detection Scorer

 **Rejected Projects** 1 Projects ?

Project to inspect

Document Classifier

Failed on	Rejection reason
2023-04-14 12:25	Documentation missing

 **Production** 2 Projects ?

Running since	Project
2023-04-14 12:23	Customer Segmentation
2023-04-14 12:20	Churn Prediction

The user data app shows projects in development, validation and production and offers different interactions.

The user data app is deployed by the admin and **shared** with the users. Users don't have access to the underlying workflow itself.

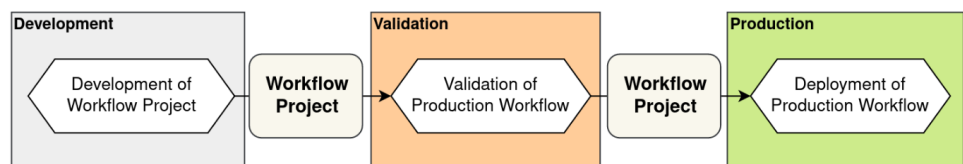
Continuous Deployment Sophistication

When productionizing data science, simply moving a production workflow through the process might not be enough. Data and models tend to change over time so constant monitoring of the deployed production of what happens to the results of the workflow can also be required. As monitoring workflows that contain models are very use case dependent tasks, there is not a general solution that can be provided. Instead, the data scientist creating the production workflow is also responsible for creating proper monitoring workflows.

In the case of monitoring, there are different ways of handling the case when the monitoring workflow detects that a production workflow needs to be updated. With the **Integrated Deployment** extension, KNIME allows the data scientist to create a training workflow that can automatically create the production workflow.

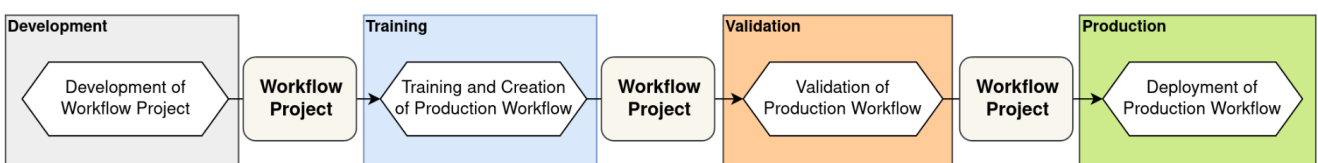
CDDS allows different features to be enabled depending on the requirements and the sophistication of the workflow projects being deployed. So far, this guide was focusing on the simple scenario, i.e. Level 1. In the following, two further sophistication levels are explained.

Level 1: Validation of a production workflow



In the simplest scenario, a data scientist creates a workflow that should be deployed into production. This workflow could in general be any workflow, e.g., an ETL workflow or a prediction workflow. For this scenario, the training environment is not required. The workflow project only contains the production workflow which gets validated and pushed to production. If a user wants to update the production workflow, they will manually update the workflow and trigger the deployment process again.

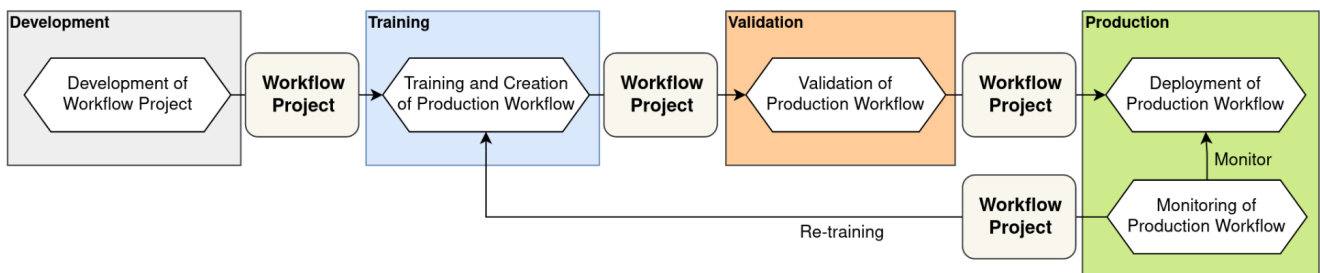
Level 2: Automatic creation of the production workflow



In a modeling use case, a data scientist can use the Integrated Deployment extension to create the production workflow automatically. In this scenario, the workflow project only

contains a training workflow and a Training stage is added to the process which comes before Validation. In the training stage, the training workflow will be executed and the production workflow will be added to the workflow project. Then, similar to scenario 1, the production workflow gets validated and pushed to production. Now, if the user wants to simply re-train and update the production workflow, the training environment can again be triggered by the user to restart the deployment process.

Level 3: Monitoring of the production workflow and automatic re-training



This scenario is the most sophisticated and represents data science best practice. Suppose a data scientist wants to have automatic monitoring of the production workflow in place. In that case, they add a monitoring workflow to the workflow project that contains a training workflow. This monitoring workflow also gets pushed to the production environment where it is executed on a schedule. If the monitoring workflow detects that a re-training is needed, it initiates the process automatically. No manual interaction by the data scientist is required.

Workflow Project

As already explained in the [concepts overview](#), the CDDS extension works with workflow projects instead of single workflows. The structure and content of workflow projects change for Level 2 and Level 3 use cases.

What this means is that the data scientist needs to create a series of related workflows that together move through the different stages of the CDDS extension. To make this task of grouping related workflows together, the CDDS relies on a workflow project which potentially contains multiple workflows:

- **Training workflow:** Trains a model and creates the production workflow using the [Integrated Deployment](#) extension. This workflow is optional and only required for Level 2 and 3, i.e., if one wants the training workflow to create the production workflow containing the model. This workflow would be located in the “Training” subfolder of a workflow project.

- **Production workflow:** The workflow that will eventually be deployed. If a training workflow is part of the workflow project, this workflow will be created automatically. It does not need to be a prediction workflow that applies a machine learning model, it could also be, e.g., an ETL workflow. This workflow would either be located in the root or the “Production” subfolder of a workflow project.
- **Monitoring workflow:** Monitors the deployed production workflow and, if required, initiates a re-training. This workflow is optional and only required for Level 3, i.e., if one wants to monitor the deployed production workflow. This workflow would be located in the “Monitoring” subfolder of a workflow project.

The demands for the sophistication level vary between projects. Hence, CDDS has been developed so that it knows which stages to execute based on the contents of the workflow project. The data scientist might want to use monitoring and automatic re-training and, hence, creates a workflow project containing training and monitoring workflows. If there is no need for monitoring and automatic re-training, those workflows would not be provided and those stages would not be executed, i.e. the workflow project would only contain a production workflow and would start from the validation stage before moving to the production stage.

Training Environment

In addition to the Validation and Production environments explained in a previous [section](#), Level 2 allows to separate the process also into a Training environment.

- **Training Environment:** Training machine learning models often requires a lot of computational resources, e.g. Deep Learning models might require a GPU and a special setup. Therefore, executing training workflows in a separate environment with proper resources available makes sense.

FAQs

The CDDS extension has been designed for changes, modifications, and extensions. This section answers many of the common questions and tasks organizations are interested in.

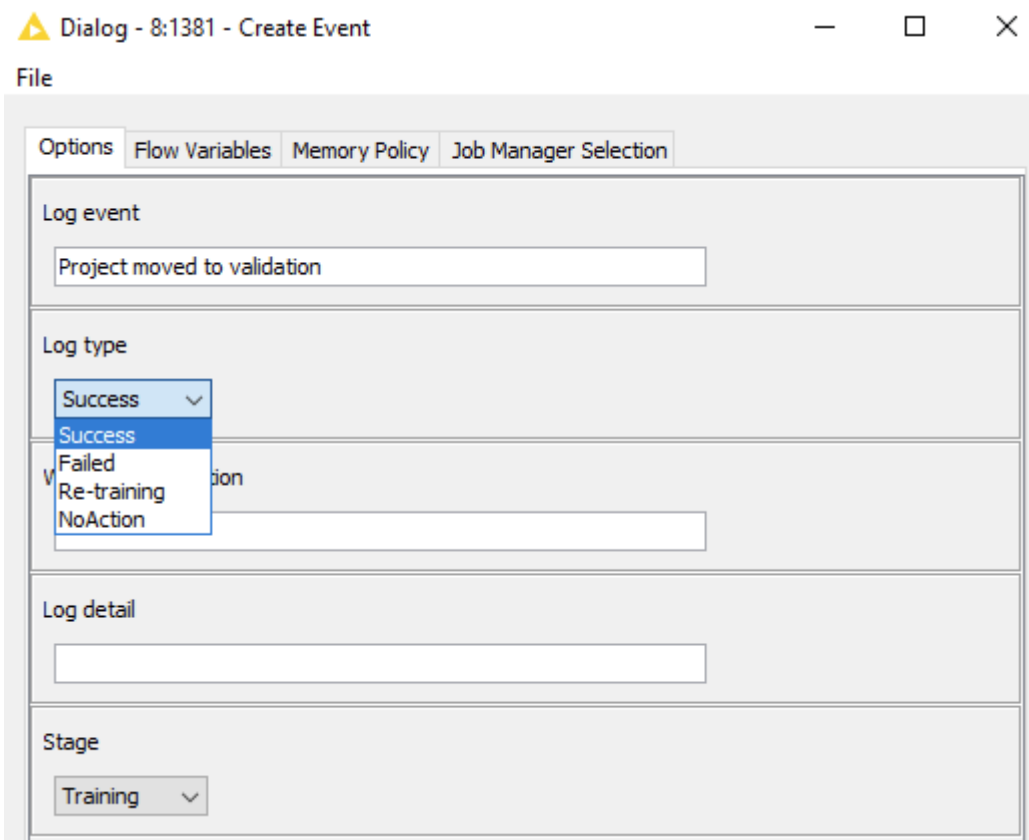
Can I change the location of the environments?

Yes. You would need to adapt the environment connection components (see Administration > Components > Connectors). There exists one for the Development, Training, Validation, and Production environments. Each of those needs to point to the location of the corresponding environment.

You can even separate the environments across different Hub installations. The connection components allow connecting to other Hub instances, so that e.g. Production could be completely separated.

Can I add additional logging to the different processes?

Yes, by using the Create Event and Log Event components, you can add additional logging at any stage. The Create Event components configurations are the following:



The screenshot shows a dialog box titled "Dialog - 8:1381 - Create Event". It has a "File" menu and four tabs: "Options", "Flow Variables", "Memory Policy", and "Job Manager Selection". The "Options" tab is active and contains the following fields:

- Log event:** A text input field containing "Project moved to validation".
- Log type:** A dropdown menu with "Success" selected. The dropdown list shows "Success", "Failed", "Re-training", and "NoAction".
- Log detail:** An empty text input field.
- Stage:** A dropdown menu with "Training" selected.

The components use a predefined schema. Log event, workflow description, and log detail can be specified by the user. Log Type and Stage are predefined and used by the existing workflows to record actions. All can be modified and extended.

Can I change the database used for logging?

Yes. You can configure this during installation. In a running setup, you would need to adapt the Database Connector component (see Administration > Components > Connectors) and run the “Initialize Event Logging Database” under the Administration > Setup.

The workflow can also be used to reset the database or change the schema for logging. Note that you would also need to adapt the “Create Event” and “Log Event” components if you change the schema.

Can I add more validation steps?

Yes. You can add further components to the “Validation Controller” workflow.

Can I do notifications instead of automatic retrain/redeploy?

Yes. You would modify the “Monitoring Controller” workflow.

Can I control access to the different environments?

Yes. This would require you to configure the access to the spaces (Training, Validation, Production) accordingly or to separate the different environments onto separate teams or Hub installations.

Can I ensure that all production jobs are checked for up-to-date components?

KNIME Hub contains a feature that automatically updates components. But of course, this means that any required validation process on those updated components as explained here is not done. If you want all components to go through the CDDS process, you would turn off the automatic updating of components. You would then create a monitoring workflow that checks whether components are up to date. If components need to be updated, the monitor can trigger an automatic retrain or notifications as necessary.

Can I change the number of “environments”?

Yes. You would need to modify and possibly add more trigger / controller workflows to match your setup. The critical components that would need to be modified include:

- Transfer Workflows

Can I have backups / alternate storage of the project assets?

Yes. You would add the appropriate connectors to each controlling workflow. The critical components that would need to be modified in each controlling workflow include:

- Transfer Workflows

Does the CDDS extension require me to follow a data standard for running training, validation, or monitor workflows?

No. We leave that up to the user and assume that all data required is defined within the individual workflow in such a way that the workflow calls it correctly.

Can I use GitHub with the CDDS extension?

Yes. Note that GitHub can not “process” workflows as it does not understand KNIME workflows, so you will always be executing workflows within KNIME. However, you can use GitHub as a backup device or alternately as your “storage of record” to contain all your workflows in their various states. If you just want to capture production workflows in GitHub, you would modify the Production Controller. If you wanted to capture all stages, then you would need to modify the other controller workflows as well. The following components would need to be modified:

- Transfer Workflows

Can I add workflow-specific validation?

Yes. The CDDS extension can be extended to satisfy this requirement. You would write a project specific validation workflow. You would add a subdirectory to the project folder called “Validation” and place the workflow there. In the Validation Controller, you would add an additional component that checks for the existence of a workflow under the validation folder and, if it exists, executes that workflow. You would need to add your own control logic for deciding on success criteria and could use the logging functions to capture the results of that additional validation check.

Can I validate Python code within my workflows?

Yes. If you are repeatedly reusing Python code, then best practice would be to write Python KNIME Nodes. In this way, you could use any external Python code validation mechanism you already have and be assured that the Python Nodes used in KNIME were validated. For “open Python code” used within KNIME Python scripting nodes, you would use the Workflow Summary Extractor node to pull the Python code being used. You would then need to have that code externally validated by your Python validation environment. You could extend the validation controller workflow to handle the external validation, capture any relevant information and make decisions based on that. You could also pass that external information to the log by calling the appropriate components.

Installation Guide

The KNIME CDDS extension is a predefined setup of folder structures and workflows within Hub spaces. To ease the process of setting CDDS up on Business Hub, a corresponding installation workflow can be used.

Prerequisites

1. Analytics Platform with version 5.1 or higher.
2. To run the installation workflow, you need to create an **application password** on the Business Hub. The application password will be used by the installation workflow to connect to the Hub and create required resources, such as spaces. This password will also be configured in the CDDS workflows to move workflows from one space to another, i.e. the application password should not be deleted after the installation process.
3. An **execution context** is required in the Hub team the CDDS extension should be installed to.
4. For event logging, the CDDS extension requires a Postgres database to be configured. While of course an external database could be used and configured, there is also a Hub internal one that could be used. The CDDS installation workflow requires the password of that database if it should be used. See the following instructions on how a Business Hub admin can retrieve the password.



In order to use the Hub internal Postgres database you need to enable the database first in the Admin Console, under *PostgreSQL Database > Enable CDDS database*.

How to retrieve the password for the internal Hub CDDS database

To configure CDDS using a database running with Hub, a Business Hub admin with access to the Kubernetes cluster the Hub is running in needs to retrieve the database password. To get the required credentials, you need to access the instance the Business Hub is running on and run the following command:

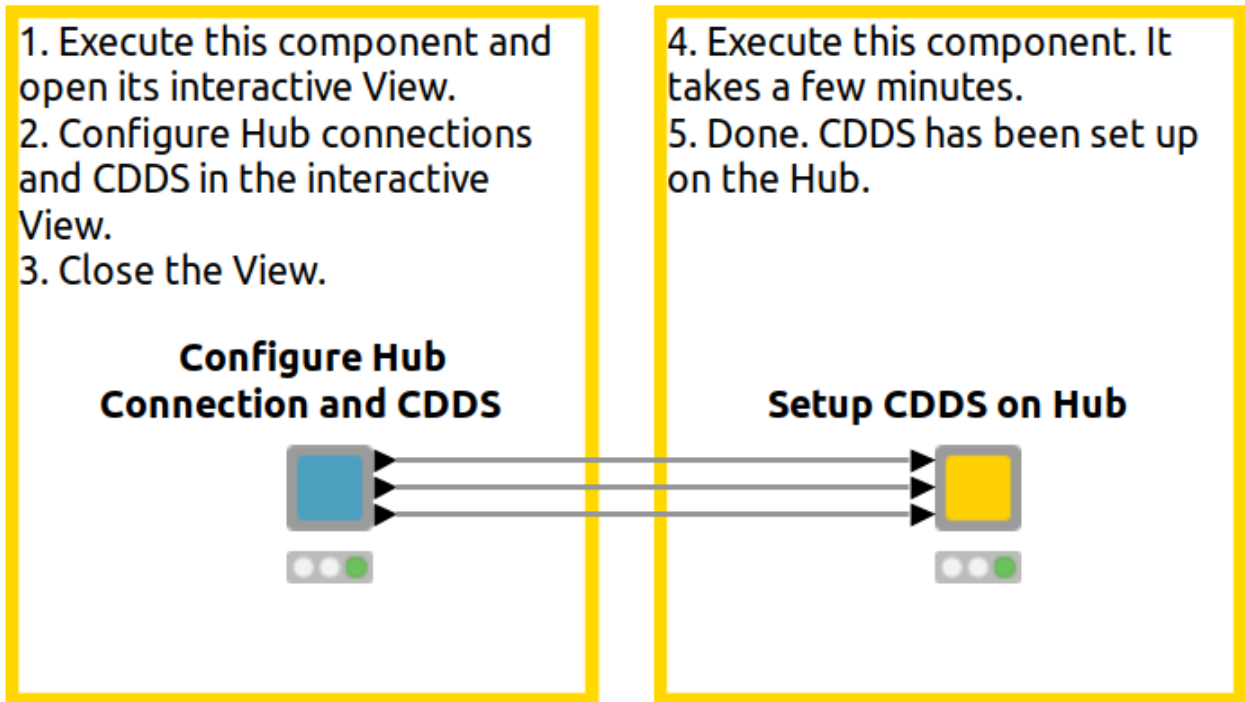
```
kubectl -n knime get secret cdds.cdds-postgres-cluster.credentials -o yaml
```

This will return a file that contains the `password`. Please notice that it is base64 encrypted. In order to get the decrypted password, you can run the following command:

```
echo <PASSWORD> | base64 -d
```

Installation Workflow

Find and download the [Install CDDS using a KNIME data app](#) workflow to the Analytics Platform.



Installation workflow executed in KNIME Analytics Platform.

Configuration

Open the workflow and execute the first component (note: the workflow must not be executed on the Hub but in Analytics Platform). Open the view of this component to configure the installation.

Hub connection

The connection information of the Hub where CDDS should be installed needs to be configured.

1. **Hub URL:** The location of the Hub. Enter the URL of the Hub here.
2. **User Credentials:** The application password, see [prerequisites](#) above.

Hub URL

User Credentials (Application Password)

User

Password

Connect to Hub

Database used for event logging

The database that is used to store CDDS event logs needs to be configured. It needs to be a PostgreSQL database.

1. **Use Hub internal database:** A Hub internal database can be used. This might be useful for testing, as it removes the need for an external database. For production purposes, an external database is recommended.
2. **Postgres connection information:** Enter the hostname, database name and credentials.

Event log database (PostgreSQL)

Use Hub internal database

Hostname

Database credentials

Password

Database name

CDDS configuration

Configure different settings of CDDS.

1. **Hub team:** Select the team on the Hub the CDDS spaces should be located in. All spaces will be located in the same team.
2. **CDDS level:** Select the **level of sophistication**. Depending on the level, a training and monitoring capabilities are set up.

3. **Space names:** Choose names for the created spaces.

CDDS configuration

Hub Team

Admin Team ▼

CDDS level

Level 1: Validation of a production workflow ▼

Name of the Validation space

Validation

Name of the Production space

Production

Name of the Development space

Development

Name of the Administration space

Administration

Environment contexts

Select the execution contexts of the different spaces. CDDS workflows running in these spaces will be using the selected execution context.

Execution contexts

Execution context of the Validation environment

4.7.2 ▼

Execution context of the Production environment

4.7.2 ▼

Execution context of the Administration environment

4.7.2 ▼

Automate pipelines with trigger deployments

Installation

Once configured, close the component view. Now, the workflow can be fully executed. The execution takes a few minutes. The workflow will automatically create the configured spaces on the Hub and upload all necessary folders and workflows.

Also, trigger workflows will automatically be deployed as trigger deployments and connections between the different spaces and to the database are configured.

Once the workflow is successfully executed, the CDDS extension is ready to be used. See the [CDDS on the Hub](#) section to get started with CDDS.

KNIME AG
Talacker 50
8001 Zurich, Switzerland
www.knime.com
info@knime.com