

8. End-to-end ML platforms

Reducing the complexity

Until now we saw...

- Getting a model to production takes many steps
 - Many different stakeholders will participate
 - Many different tools will be used
 - Even after the deployment the pipelines will evolve
- => The process is complex and it takes lots of time and maintenance**



Solution

- A tool that treats multiple steps from development to deployment
=> end-to-end ML platform
- More and more platforms is appearing
- Several are open-source, mostly they are managed
- They mostly started as in-house solutions (Google, Uber, Netflix, AirBnB...)
- They focus on different parts of the process (no unified definition)
- Can be framework-specific (TensorFlow Extended) or framework agnostic (Liminal) (both open-source)
- Cloud providers offer many services, including MLOps toolstack

Platform comparison



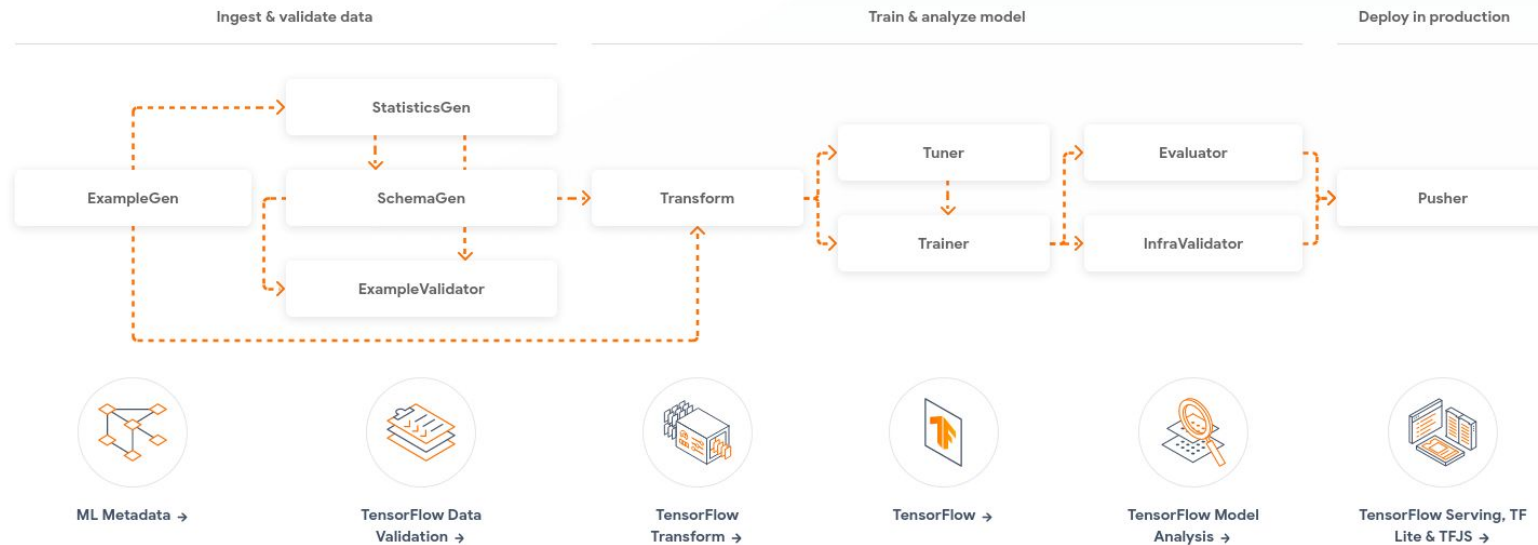
<https://valohai.com/mlops-platforms-compared/>

TensorFlow Extended (TFX)

- Google's solution for deploying their proper models
- Robust and scalable
- Open-sourced in 2019

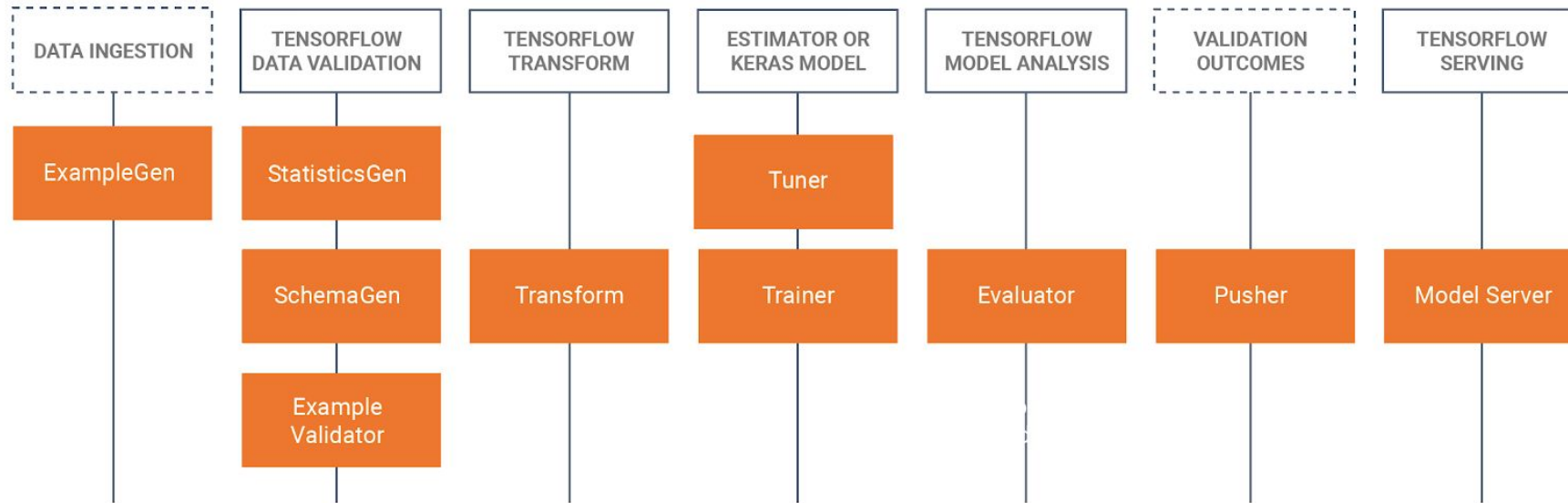
How it works

A TFX pipeline is a sequence of components that implement an ML pipeline which is specifically designed for scalable, high-performance machine learning tasks. Components are built using TFX libraries which can also be used individually.



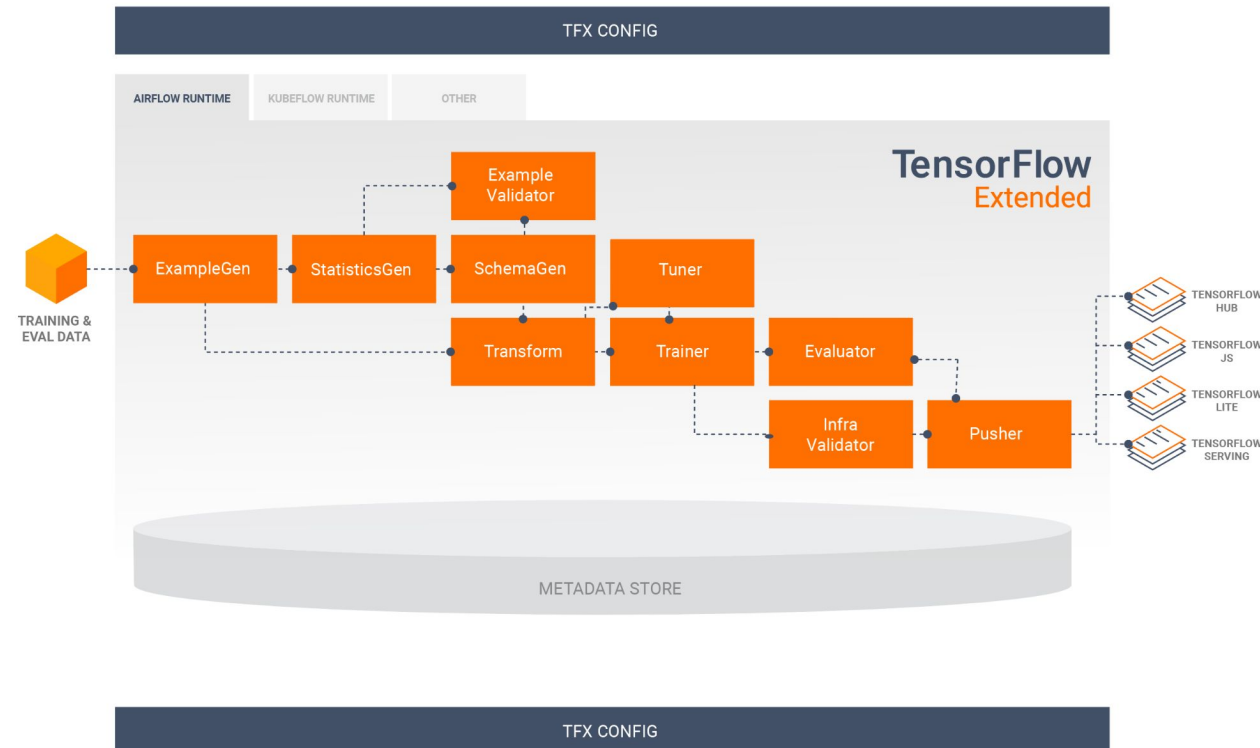
TFX: how it works

- It is based on TensorFlow libraries
- Many libraries: potential problems with compatibility



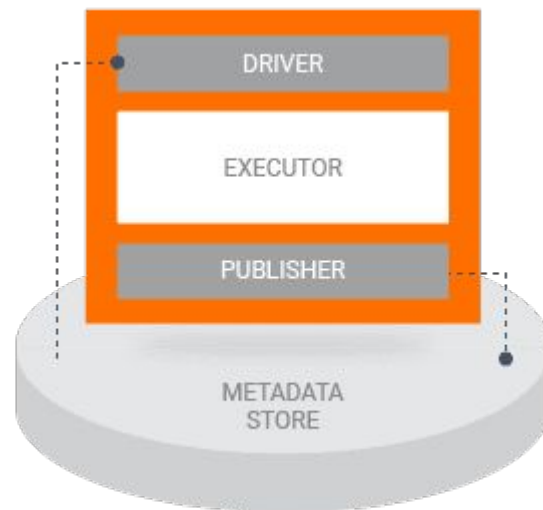
TFX: how it works

- Modular: modules can be used independently or to build pipelines
- Output from one module is input for the next one
- Reduction of boilerplate code to connect modules and make the pipeline work



TFX: metadata store

- Stores the information about the state of the system after an execution of a component
- Stores the outputs of the components:
 - information about the models, the data they were trained on, and the evaluation metrics
 - execution records for every component
 - lineage of the data objects as they flow through the pipeline.
- Makes the pipeline data and task-aware

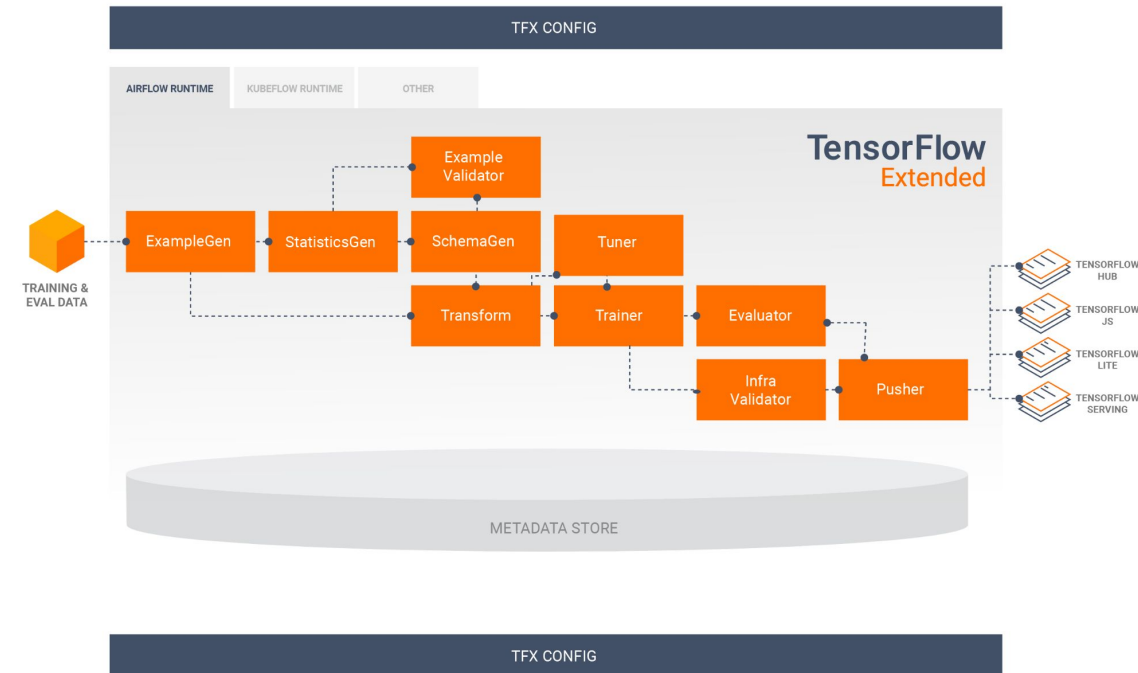
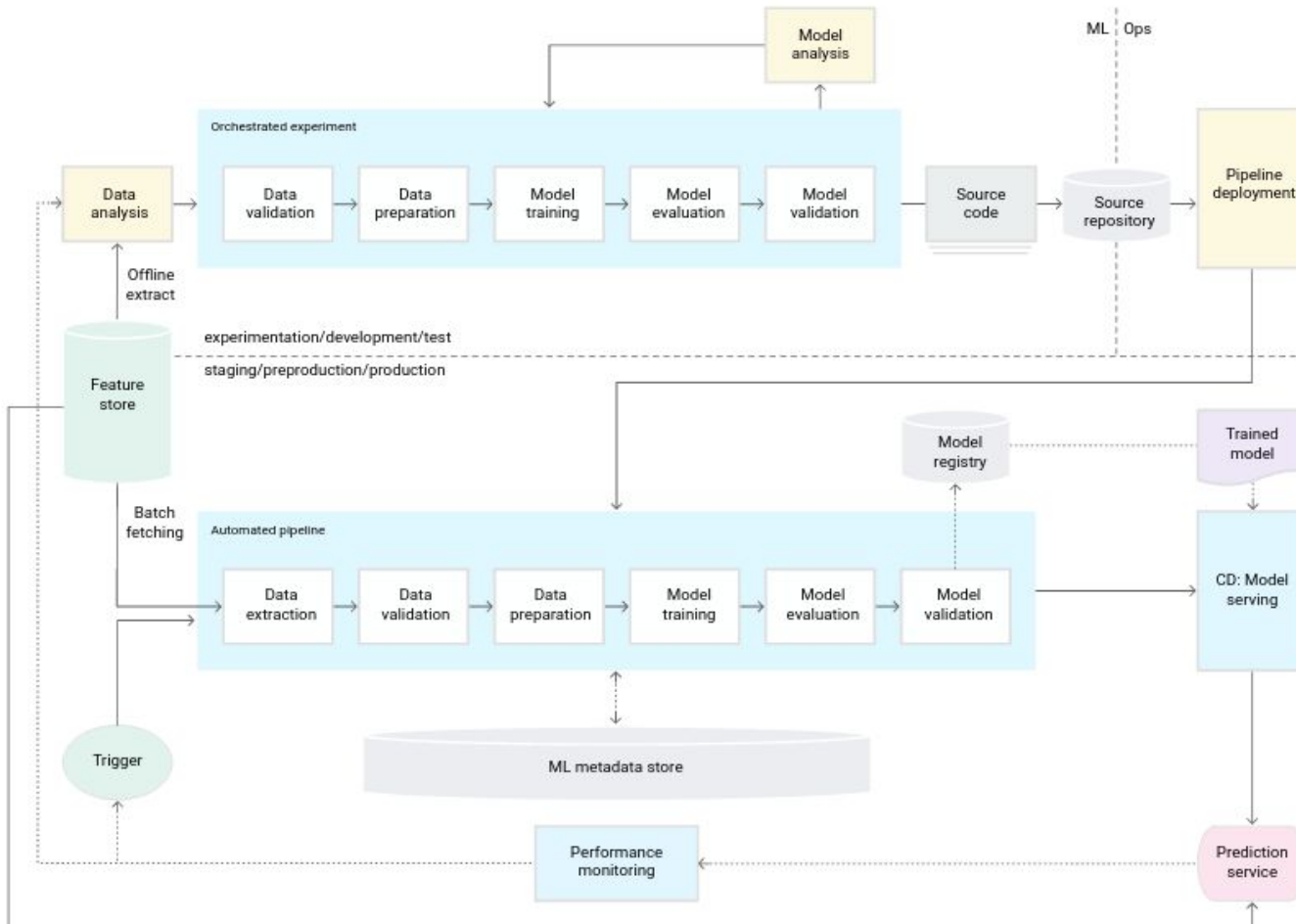


▼ ExecutionResult at 0x7efe254dfcd0	
.execution_id	1
.component	► CsvExampleGen at 0x7efdaa209b10
.component.inputs	{}
.component.outputs	[examples]
	▼ Channel of type 'Examples' (1 artifact) at 0x7efdaa209f10
.type_name	Examples
.artifacts	[0]
	▼ Artifact of type 'Examples' (uri: /home/petra/tfx/artifacts/CsvExampleGen/examples/1) at 0x7efe254f9650
.type	<class 'tfx.types.standard_artifacts.Examples'>
.uri	/home/petra/tfx/artifacts/CsvExampleGen/examples/1
.span	0
.split_names	["train", "eval"]
.version	0

What will we do

- Exercise with TFX in colab:
https://colab.research.google.com/github/tensorflow/tfx/blob/master/docs/tutorials/tfx/components_keras.ipynb
- Understand the functioning of the individual components of TFX

Comparison of the initial pipeline and TFX



To go from here

- Learn TFX at Coursera: <https://www.coursera.org/learn/ml-pipelines-google-cloud>
- To install and try it locally: <https://www.adaltas.com/en/2021/03/05/tfx-overview/>
- Look into one of the platforms: <https://valohai.com/mlops-platforms-compared/>