

FL Platforms

2022_Fall



FL Platforms

Open FL Platform (Active),

<https://github.com/Kwangkee/FL/blob/main/FL@Platform.md#open-fl-platform-active>

Open FL Platform (Etc),

<https://github.com/Kwangkee/FL/blob/main/FL@Platform.md#open-fl-platform-etc>

FL Benchmark, <https://github.com/Kwangkee/FL/blob/main/FL@Platform.md#fl-benchmark>

Commercial FL Platform,

<https://github.com/Kwangkee/FL/blob/main/FL@Platform.md#commercial-fl-platform>

Open FL Platforms

Open FL Platform (Active)	Affiliation
Flower: A Friendly Federated Learning Framework, https://flower.dev/	U of Cambridge, Samsung
FedScale: A scalable and extensible federated learning engine and benchmark, http://fedscale.ai/	U of Michigan
FedML: https://fedml.ai/	USC & FedML Inc.
FLSim: https://github.com/facebookresearch/FLSim	Meta/Facebook Research

Open FL Platform (Etc)	Affiliation
TensorFlow Federated (TFF), https://www.tensorflow.org/federated?hl=ko	Google
PySyft, https://github.com/OpenMined/PySyft	OpenMined
FedAI, https://www.fedai.org/	WeBank
Open Federated Learning (OpenFL) - An Open-Source Framework For Federated Learning, https://github.com/intel/openfl	Intel
IBM Federated Learning, https://ibmfl.mybluemix.net/	IBM Research.
OpenFed, https://github.com/FederalLab/OpenFed	--
FL_PyTorch, https://github.com/burlachenkok/flpytorch	--

FLRA: A Reference Architecture for Federated Learning Systems

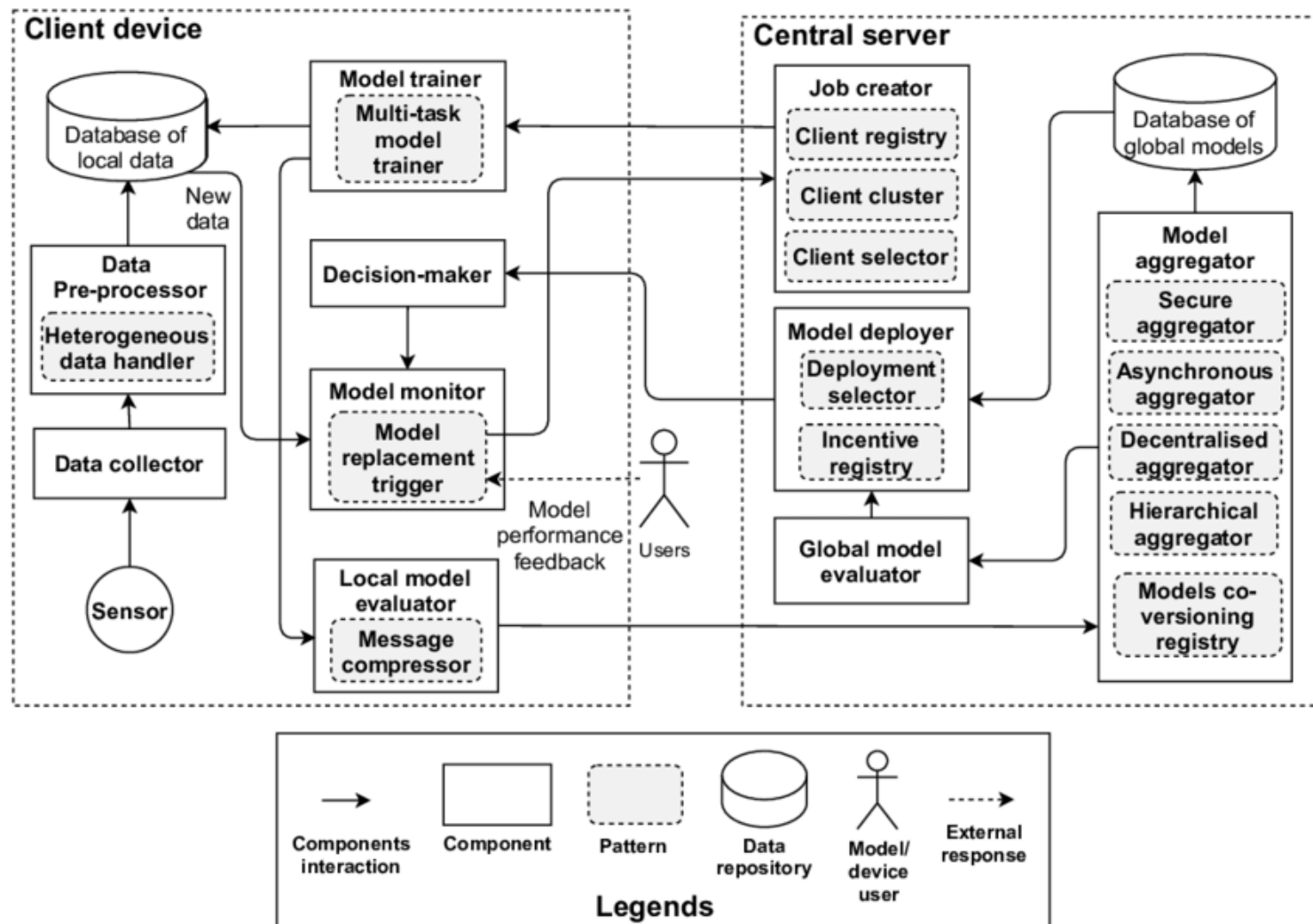
FLRA: A Reference Architecture for Federated Learning Systems, <https://arxiv.org/abs/2106.11570>

Although much effort has been put into federated learning from the machine learning perspectives, our previous systematic literature review on the area shows that there is a distinct lack of considerations for software architecture design for federated learning. In this paper, we propose **FLRA, a reference architecture for federated learning systems, which provides a template design for federated learning-based solutions.**

The FLRA reference architecture consists of a pool of architectural patterns that could address the frequently recurring design problems in federated learning architectures. The FLRA reference architecture can serve as a **design guideline to assist architects and developers with practical solutions for their problems**, which can be further customised.

FLRA: A Reference Architecture for Federated Learning Systems

FLRA: A Reference Architecture for Federated Learning Systems, <https://arxiv.org/abs/2106.11570>



FLRA: A Reference Architecture for Federated Learning Systems

FLRA: A Reference Architecture for Federated Learning Systems, <https://arxiv.org/abs/2106.11570>

The central servers interacts with a massive number of client devices that are both system heterogeneous and statistically heterogeneous. The magnitude of client devices number is also several times larger than that of the distributed machine learning systems [18,24]. To increase the model and system performance, client devices can be selected every round with predefined criteria (e.g., resource, data, or performance) via **client selector** component.

Job creation	Mandatory	Job creator	Initialises training job and global model
	Optional	Client registry	Improves system's maintainability and reliability by maintaining client's information
		Client cluster	Tackles statistical heterogeneity & system heterogeneity by grouping clients with similar data distribution or resources before aggregation
		Client selector	Improves model & system's performance by selecting high performance client devices
Data collection & preprocessing	Mandatory	Data collector	Collects raw data through sensors or smart devices deployed
		Data preprocessor	Preprocesses raw data
	Optional	Heterogeneous Data Handler	Tackles statistical heterogeneity through data augmentation methods

FLRA: A Reference Architecture for Federated Learning Systems

FLRA: A Reference Architecture for Federated Learning Systems, <https://arxiv.org/abs/2106.11570>

Local model training.

Once the client receives the job from the central server, the model trainer component performs model training based on configured hyperparameters (number of epochs, learning rate, etc.). In the standard federated learning training process proposed by McMahan in [28], only model parameters (i.e., weight/gradient) are mentioned to be sent from the central server, whereas in this reference architecture, the models include not only the model parameters but also the hyperparameters.

Model evaluation.

The **local model evaluator** component measures the performance of the local model and uploads the model to the model aggregator on the central server if the performance requirement is met. In distributed machine learning systems, the performance evaluation on client devices is not conducted locally, and only the aggregated server model is evaluated. However, for federated learning systems, local model performance evaluation is **required for system operations such as client selection, model co-versioning, contributions calculation, incentive provision, client clustering, etc.**

Model training	Mandatory	Model trainer	Trains local model
		Local model evaluator	Evaluates local model performance after each local training round
		Model aggregator	Aggregates local models to produce new global model

FLRA: A Reference Architecture for Federated Learning Systems

FLRA: A Reference Architecture for Federated Learning Systems, <https://arxiv.org/abs/2106.11570>

this technique is particularly relevant when faced with nonIID data which can produce personalised model that may outperform the best possible shared global model [18]

The conventional design of a federated learning system that relies on a central server to orchestrate the learning process might lead to a single point of failure. **A decentralise aggregator** performs model exchanges and aggregation in decentralised manner to improve system reliability. The known uses of decentralised aggregator include BrainTorrent [31] and FedPGA [15]. **Blockchain can be employed as a decentralised solution for federated learning systems.**

Optional	Multi-task model trainer	Improves model performance (personalisation) by adopting multi-task training methods
	Message compressor	Improves communication efficiency through message size reduction to reduce bandwidth consumption
	Secure aggregator	Improves data privacy & system security through different secure multiparty computation protocols
	Asynchronous aggregator	Improves system performance by reducing aggregation pending time of late client updates
	Decentralised aggregator	Improves system reliability through the removal of single-point-of-failure
	Hierarchical aggregator	Improves system performance & tackle statistical heterogeneity & system heterogeneity by aggregating models from similar clients before global aggregation
	Model co-versioning registry	Improves system's accountability by recording the local models associated to each global models to track clients' performances

FLRA: A Reference Architecture for Federated Learning Systems

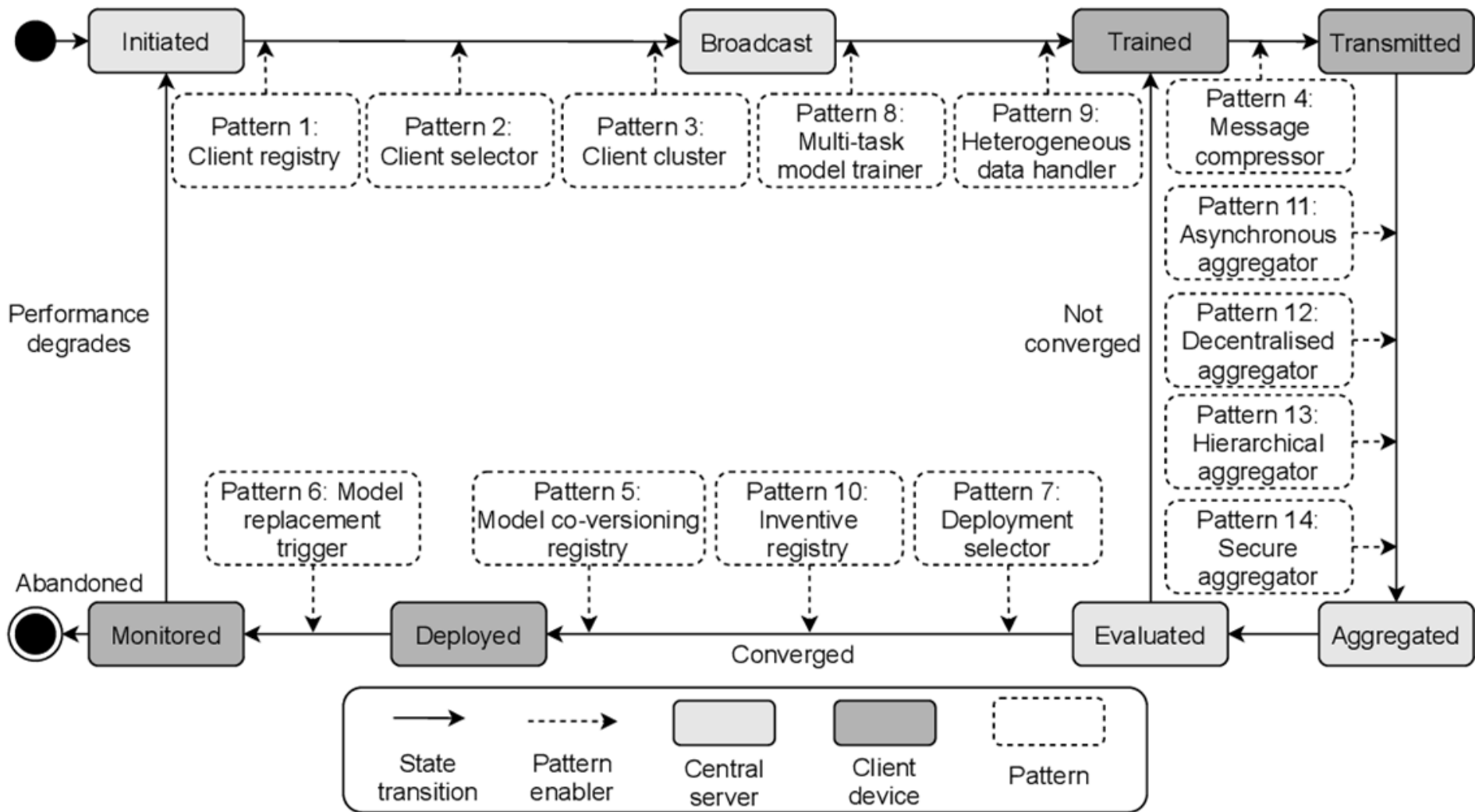
FLRA: A Reference Architecture for Federated Learning Systems, <https://arxiv.org/abs/2106.11570>

The **incentive registry** component maintains all the client devices' incentives based on their contributions and agreed rates to motivate clients to contribute to the training. Blockchain has been leveraged in FLChain [3] and DeepChain [36] to build a incentive registry.

Model deployment	Mandatory	Model deployer	Deploys completely-trained-models
		Decision maker	Decides model deployment
	Optional	Deployment selector	Improves model performance (personalisation) through suitable model users selection according to data or applications
		Incentive registry	Increases clients' motivatability
Model monitoring	Mandatory	Model monitor	Monitors model's data inference performance
	Optional	Model replacement trigger	Maintains system & model performance by replacing outdated models due to performance degrades

Architectural patterns for FL

Architectural patterns for the design of federated learning systems, <https://arxiv.org/abs/2101.02373>



Blockchain-based trustworthy federated learning architecture

Towards Trustworthy AI: Blockchain-based Architecture Design for Accountability and Fairness of Federated Learning Systems, <https://ieeexplore.ieee.org/abstract/document/9686048>

<https://github.com/Kwangkee/FL/blob/main/FL%40CSIRO.md#towards-trustworthy-ai>

- However, federated learning systems struggle to achieve and embody responsible AI principles. In particular, federated learning systems face accountability and fairness challenges due to multi-stakeholder involvement and heterogeneity in client data distribution. To enhance the accountability and fairness of federated learning systems, we present a blockchain-based trustworthy federated learning architecture.
- We designed the architecture based on a reference architecture for federated learning system named FLRA [6]. <https://arxiv.org/abs/2106.11570>

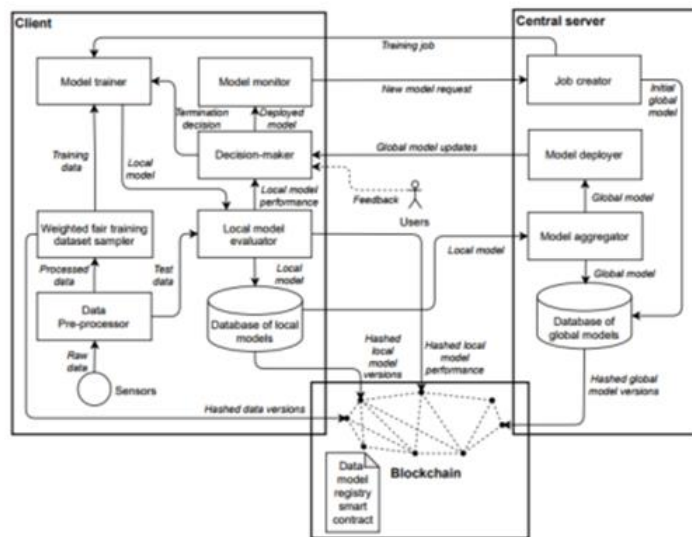


Fig. 1: Blockchain-based Trustworthy Federated Learning Architecture

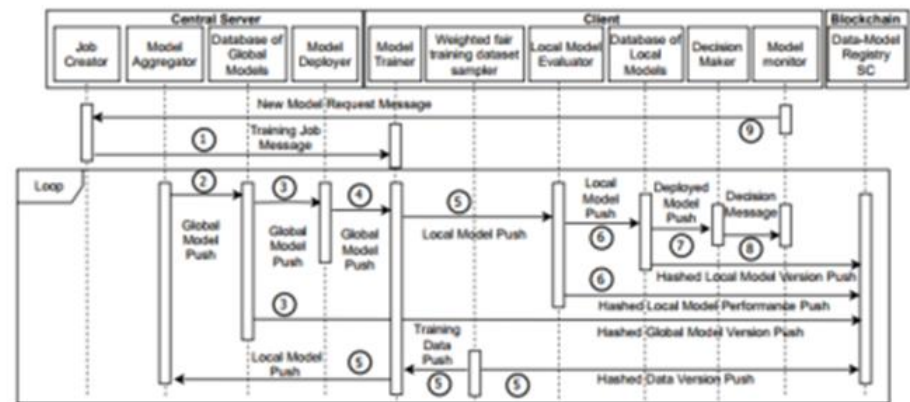


Fig. 2: Sequence Diagram of Blockchain-based Trustworthy Federated Learning Process

Flower

Flower, <https://flower.dev/>

A Friendly Federated Learning Framework, A unified approach to federated learning. Federate any workload, any ML framework, and any programming language.

Github: <https://github.com/adap/flower>

Slack: <https://friendly-flower.slack.com/ssb/redirect>

Summit: <https://flower.dev/conf/flower-summit-2021>

Flower: A Friendly Federated Learning Research Framework, <https://arxiv.org/abs/2007.14390>

Youtube: <https://www.youtube.com/watch?v=t5WdERBPQfk&t=1s>

On-device Federated Learning with Flower, <https://arxiv.org/abs/2104.03042>

Youtube: <https://www.youtube.com/watch?v=QJEX5c0y1I8&t=2s>

Flower use case

Scaling Flower with Multiprocessing, <https://towardsdatascience.com/scaling-flower-with-multiprocessing-a0bc7b7aace0>

Github: https://github.com/matturche/flower_scaling_example

Learn how to scale locally your Federated Learning experiments using the Flower framework and multiprocessing with PyTorch.

How to solve the issue:

This problem that you might have encountered, can be solved quite easily. Since the memory is not released until the process accessing it is released, then we simply need to encapsulate the part of our code that need to access the GPU in a sub-process, waiting for it to be terminated until we can continue to execute our program. Multiprocessing is the solution, and I will show you how to do it using PyTorch and Flower.

Differentially Private Federated Learning with Flower and Opacus, <https://towardsdatascience.com/differentially-private-federated-learning-with-flower-and-opacus-e14fb0d2d229>

Github: https://github.com/matturche/flower_opacus_example

OpenFL

[intel]

OpenFL: An open-source framework for Federated Learning, <https://arxiv.org/abs/2105.06413>

Github: <https://github.com/intel/openfl>

YouTube: Federated Learning in Healthcare Use Cases | Intel Software,

<https://www.youtube.com/watch?v=z5jJsvvfKbM>

YouTube: SOSCON Russia 2021 [English]. OpenFL: Python library for Federated Learning. Olga

Perepelkina, Intel, <https://www.youtube.com/watch?v=Zso2oYsEgw0>

인텔-펜실베이니아大, 연합학습으로 환자 보안 유지하며 뇌종양 식별하는 AI 개발,

<http://www.airtimes.kr/news/articleView.html?idxno=16331>

<https://www.apheris.com/blog-top7-open-source-frameworks-for-federated-learning>

IBM Federated Learning, <https://ibmfl.mybluemix.net/>

Github: <https://github.com/IBM/federated-learning-lib>

OpenMined, <https://www.openmined.org/>

Github: <https://github.com/OpenMined/PySyft>

FedML: A Research Library and Benchmark for Federated Machine Learning, <https://fedml.ai/>

Github: <https://github.com/FedML-AI>

OpenFL

[intel] OpenFL: An open-source framework for Federated Learning, <https://arxiv.org/abs/2105.06413>

Abstract. Federated learning (FL) is a computational paradigm that enables organizations to collaborate on machine learning (ML) projects without sharing sensitive data, such as, patient records, financial data, or classified secrets.

Open Federated Learning (OpenFL) is an open-source framework for training ML algorithms using the data-private collaborative learning paradigm of FL. OpenFL works with **training pipelines built with both TensorFlow and PyTorch**, and can be easily extended to other ML and deep learning frameworks.

Here, we summarize the motivation and development characteristics of OpenFL, with the intention of facilitating its application to existing ML model training in a production environment.

Finally, we describe the first use of the OpenFL framework to train consensus ML models in a **consortium of international healthcare organizations**, as well as how it facilitates the first computational competition on FL.

Our ambition is that federations, such as the FeTS Initiative, will not serve as ad hoc collaborations for specific research efforts, but will serve as permanent networks for researchers in the healthcare, financial, industrial, and retail industries to more effectively train, deploy, monitor, and update their AI algorithms over time.

FeTS : <https://www.fets.ai>

OpenFL

[intel] OpenFL: An open-source framework for Federated Learning, <https://arxiv.org/abs/2105.06413>

3.2 First Computational Competition on Federated Learning

As the first challenge ever proposed for federated learning, the FeTS challenge 2021 intends to address these hurdles towards both the creation and the evaluation of tumor segmentation models. Specifically, the FeTS 2021 challenge uses clinically acquired, multi-institutional MRI scans from the BraTS 2020 challenge [18–20], as well as from various remote independent institutions included in the collaborative network of a real-world federation.

Federated Tumor Segmentation Challenge 2021, <https://fets-ai.github.io/Challenge/>

Compared to the BraTS challenge, the ultimate goal of the FeTS challenge is divided into the following two tasks:

1. Task 1 (“Federated Training”) aims at **effective weight aggregation methods for the creation of a consensus model given a pre-defined segmentation algorithm for training**, while also (optionally) accounting for network outages.
2. Task 2 (“Federated Evaluation”) aims at **robust segmentation algorithms**, given a pre-defined weight aggregation method, evaluated during the **testing phase on unseen datasets** from various remote independent institutions of the collaborative network of the fets.ai federation.

Nevermined

Nevermined, <https://www.nevermined.io/>

Nevermined is a data ecosystem solution that provides the capabilities of building bespoke networks where different entities can share and monetize their data and make an efficient and secure usage of it even with untrusted parties.

Github: <https://github.com/nevermined-io>

Docs: <https://docs.nevermined.io/>

Blog: <https://docs.nevermined.io/Blog/>

YouTube: Leveraging blockchain to unlock data for federated learning,

<https://www.youtube.com/watch?v=A0A9hSIPhKI>

Description: This demo we will be using Flower and Nevermined with the goal of the model to train classify images given two different datasets.

Data Monetization for Enterprises, <https://multimedia.getresponse.com/getresponse-ylcbE/documents/12e30a17-5321-49cd-a613-963451401b07.pdf>

<https://www.nevermined.io/solutions/details/data-sharing#federated-learning>

<https://www.nevermined.io/solutions/details/data-governance#incentives>

- One of the most powerful features of blockchain technology is the ability to issue incentives, rewards, and penalties for specific activity. In Bitcoin, the incentive manifests as block rewards for miners operating computers with the largest computational power. This ability to bake in incentives that promote participant behavior provides a fundamental shift in how digital ecosystems operate, including how they are monetized as well as governed.

STADLE

STADLE: <https://www.stadle.ai/>

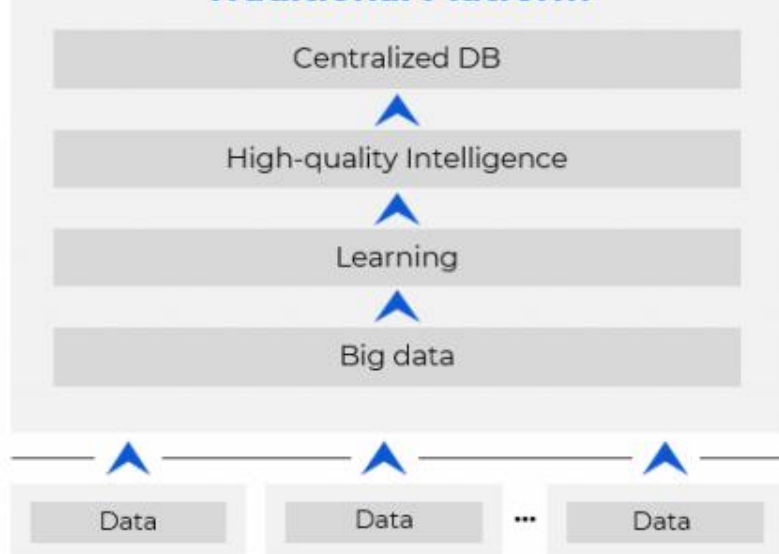
TieSet Website: <https://tie-set.com/>

Doc/Install: <https://stadle-documentation.readthedocs.io/en/latest/overview.html>

Introduction to TieSet, <https://www.youtube.com/watch?v=Nk9gdsFBKGs>

YouTube: <https://www.youtube.com/channel/UCv3NW3foNBRv12q-ymKa37A>

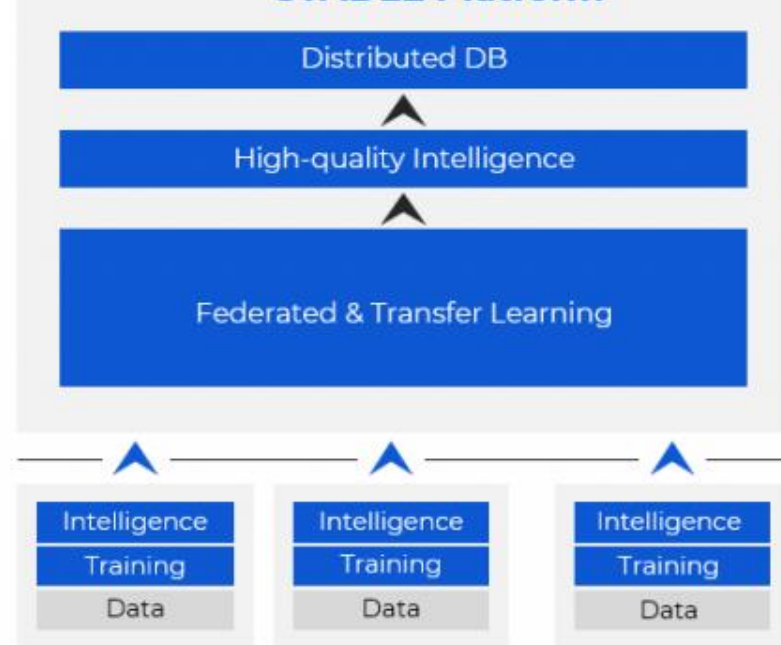
Traditional Platform



Before (Typical Big Data Companies)

Must upload full data to generate AI model,
can't work in offline mode.

STADLE Platform



After (TieSet)

Able to generate AI models on local devices, no
need to upload data, works in offline mode.

STADLE

STADLE: <https://www.stadle.ai/>

TieSet Website: <https://tie-set.com/>

Doc/Install: <https://stadle-documentation.readthedocs.io/en/latest/overview.html>

Introduction to TieSet, <https://www.youtube.com/watch?v=Nk9gdsFBKGs>

YouTube: <https://www.youtube.com/channel/UCv3NW3foNBRv12q-ymKa37A>

TieSet Team 1 min intro, <https://www.youtube.com/watch?v=vURWKP1jrv0>



STADLE

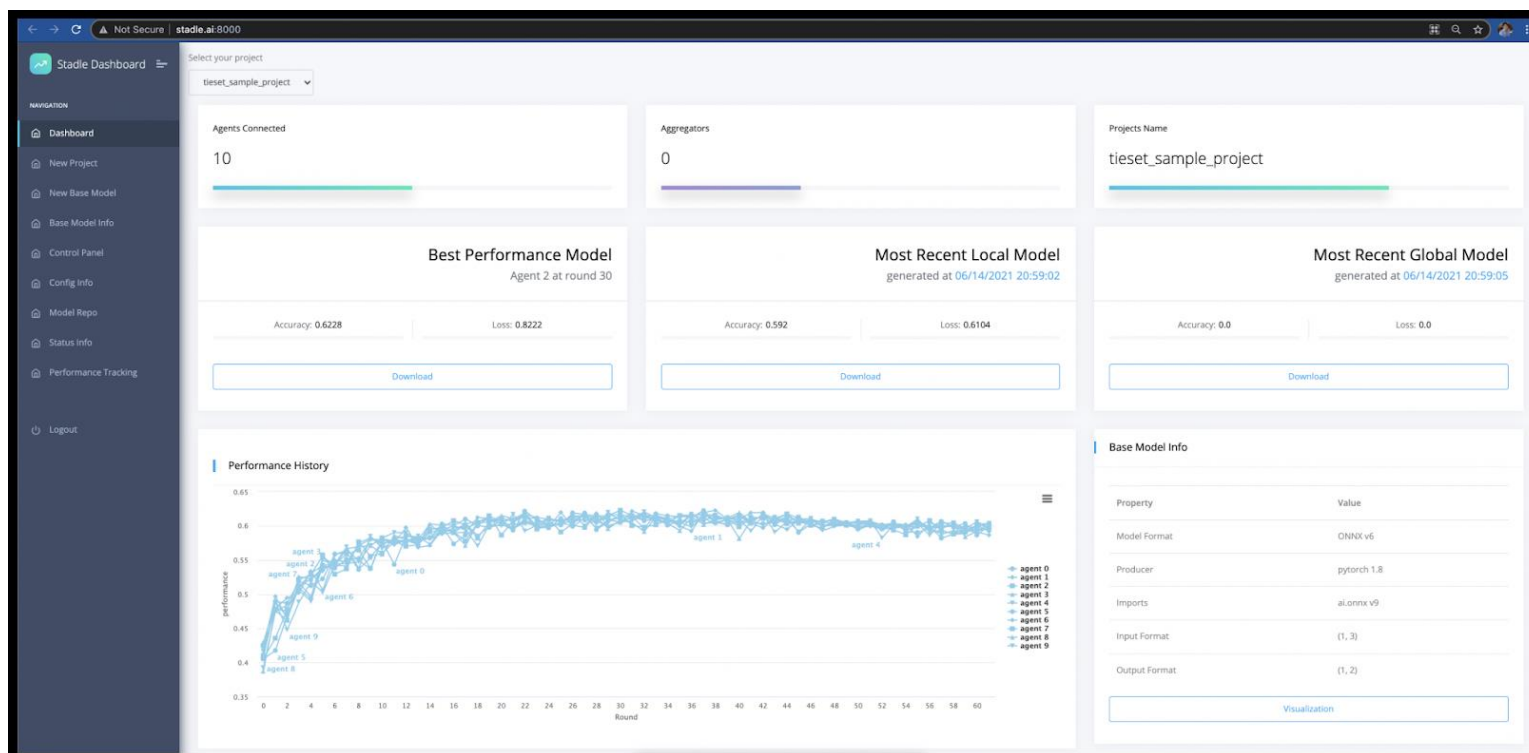
STADLE: <https://www.stadle.ai/>

TieSet Website: <https://tie-set.com/>

Doc/Install: <https://stadle-documentation.readthedocs.io/en/latest/overview.html>

News: <https://re-how.net/all/1380746/>

Now that we have completed the development of the basic functions for commercialization, we have decided to carry out a private release. With this release, further functional improvements and commercialization of this platform To promote this, we are looking for a partner who can carry out both technical verification and verification verification.



PowerFlow

PowerFlow: <https://integrate.ai/powerflow/>

Design a federated learning system in seven steps, <https://integrate.ai/blog/design-a-federated-learning-system-in-seven-steps-pftl/>

Etc.

Federated Learning in Heterogeneous Environments,
<https://www.youtube.com/watch?v=651VFm2vlhA>