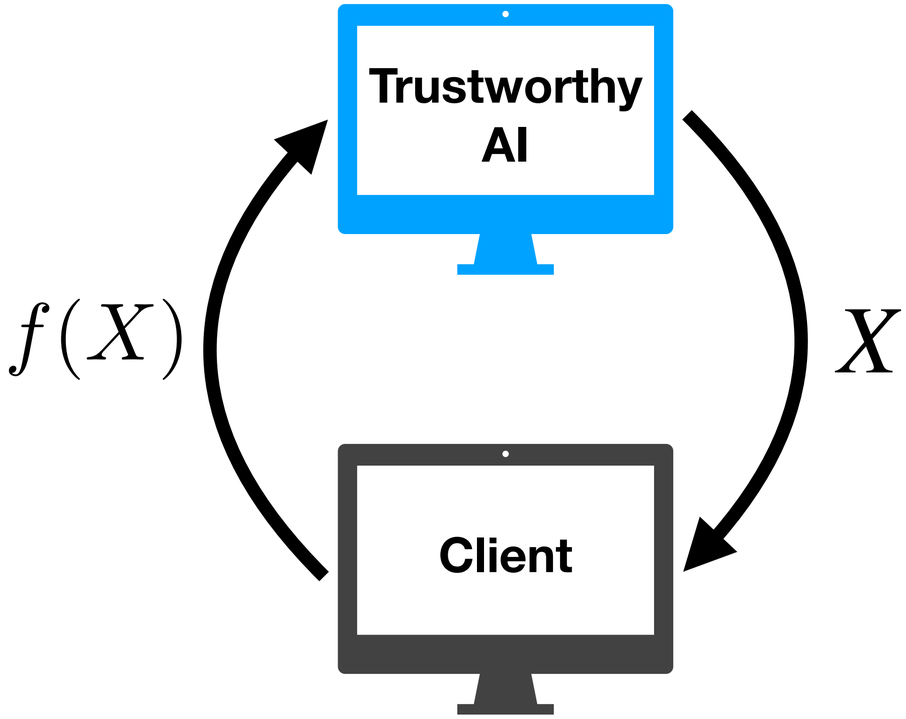
Integration of TrustworthySearch

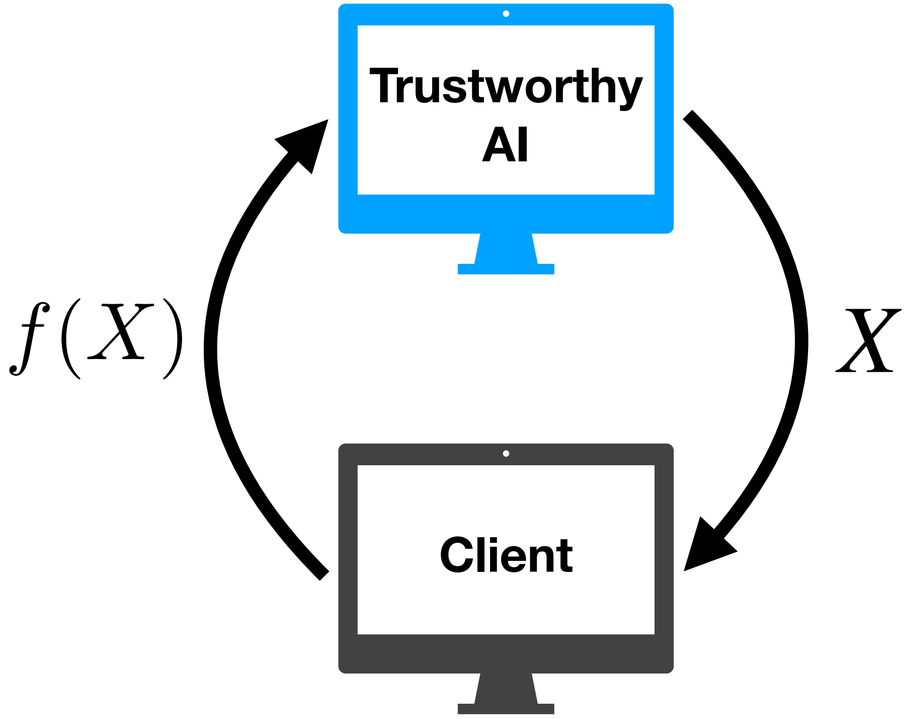
- gRPC service with Protocol Buffer (protobuf) serialization

```
service TrustworthySearch {
  rpc StartJob(JobRequest) returns (Job);
  rpc OpenSimStream(Job) returns (stream SimParams);
  rpc UploadSimResult(SimResult) returns (Empty);
  rpc GetJobResult(Job) returns (JobResult);
  rpc KillJob(Job) returns (Empty);
}
```

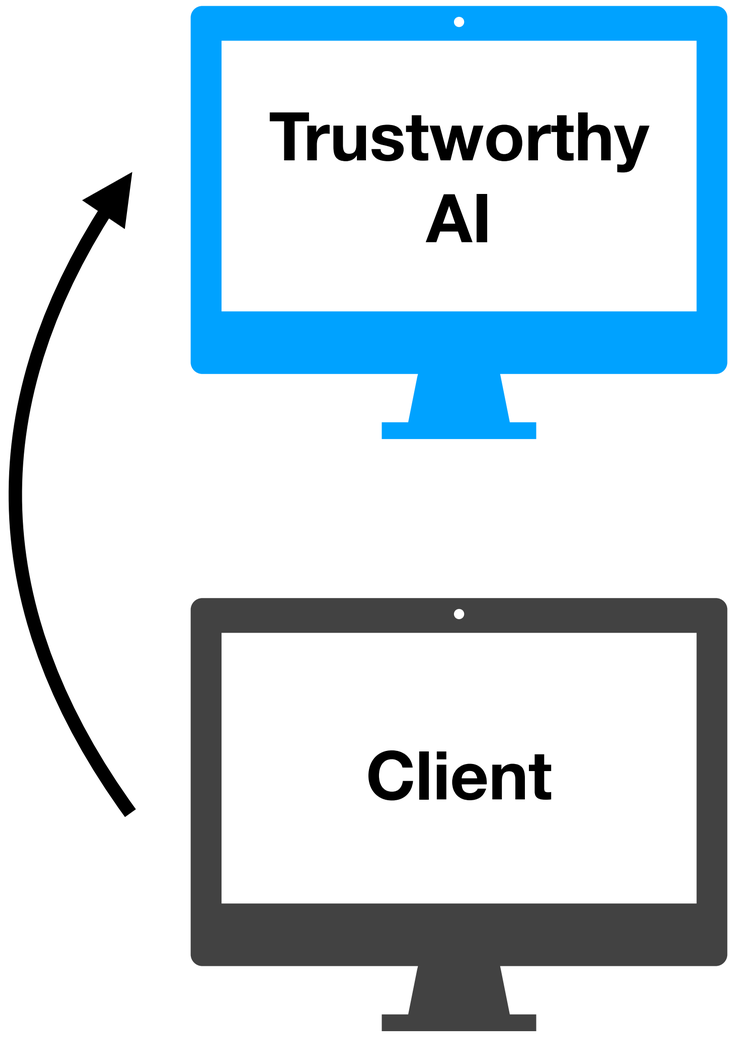


Naive version

- Communication occurs between a server and a client



Request



`StartJob(JobRequest)`

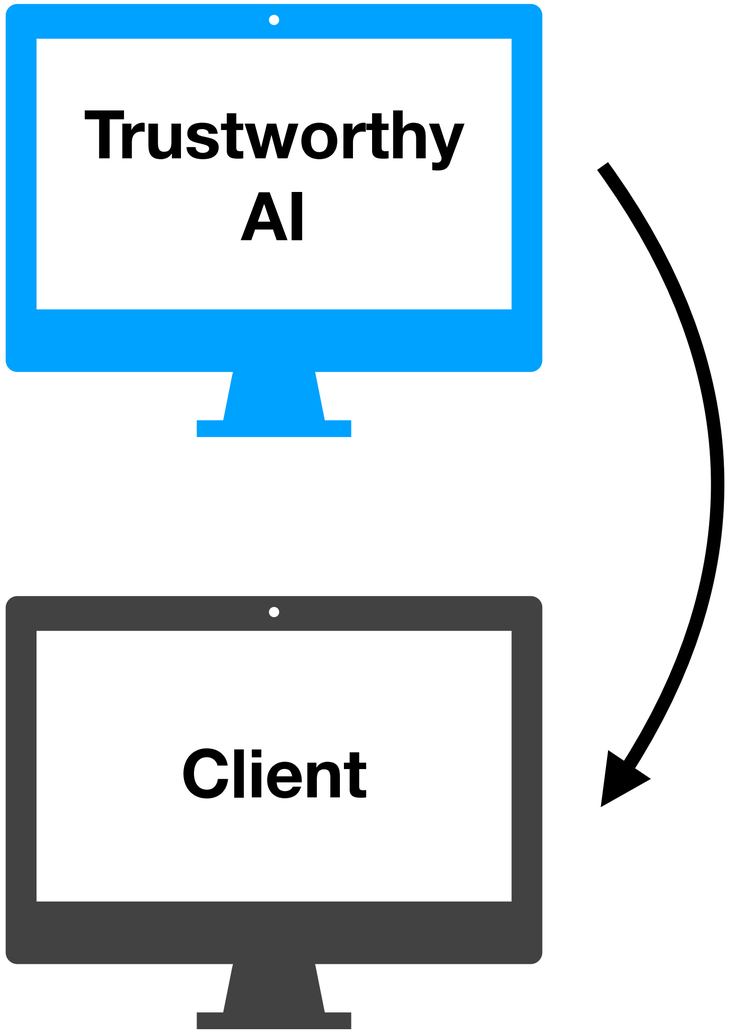
- Search space P_0
- Threshold γ
- # of simulations N

Example:

$P_0 = \mathcal{N}(0, I),\ \gamma = 2,\ N = 100$

Request

Reply



`StartJob(JobRequest)`

- Search space P_0
- Threshold γ
- # of simulations N

Example:

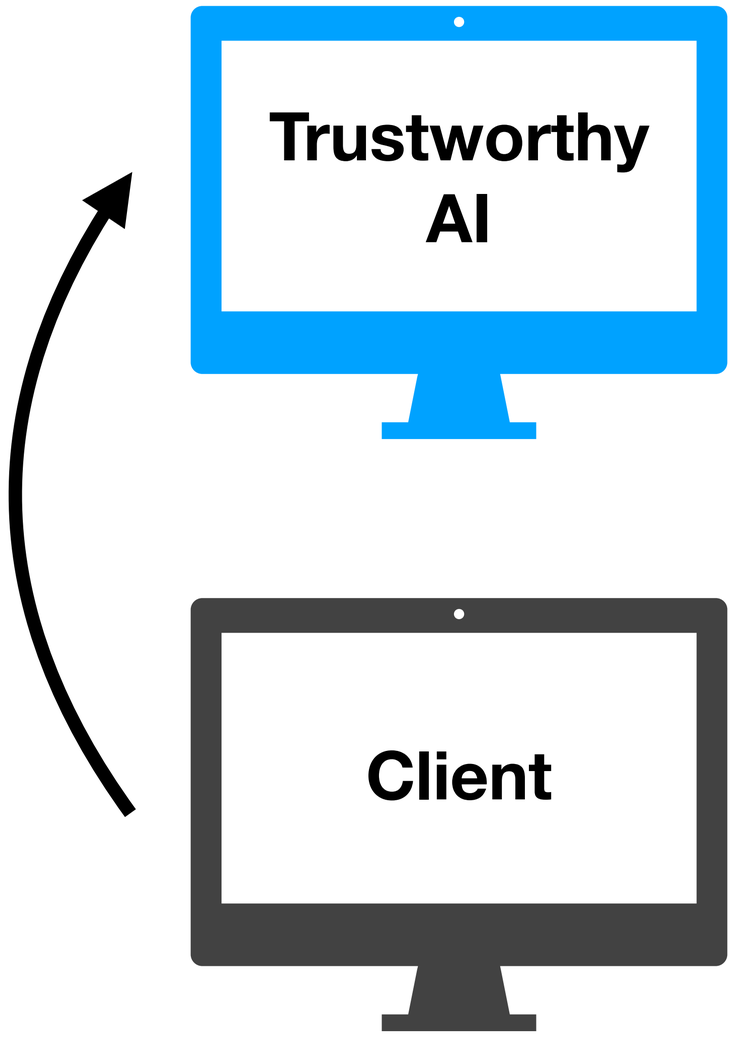
$P_0 = \mathcal{N}(0, I),\ \gamma = 2,\ N = 100$

`Job`

- Unique id for job

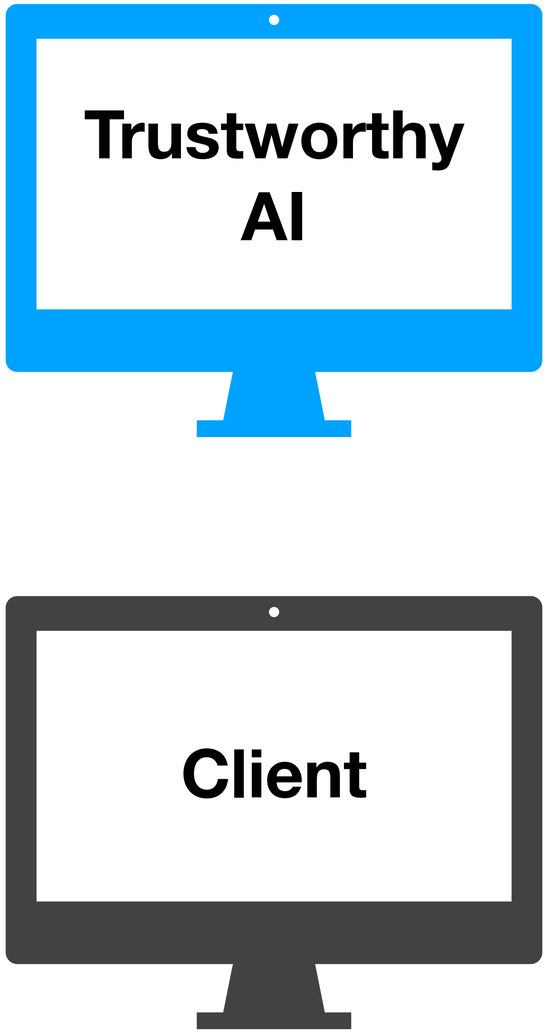
Request

OpenSimStream(Job)



Request

Replies

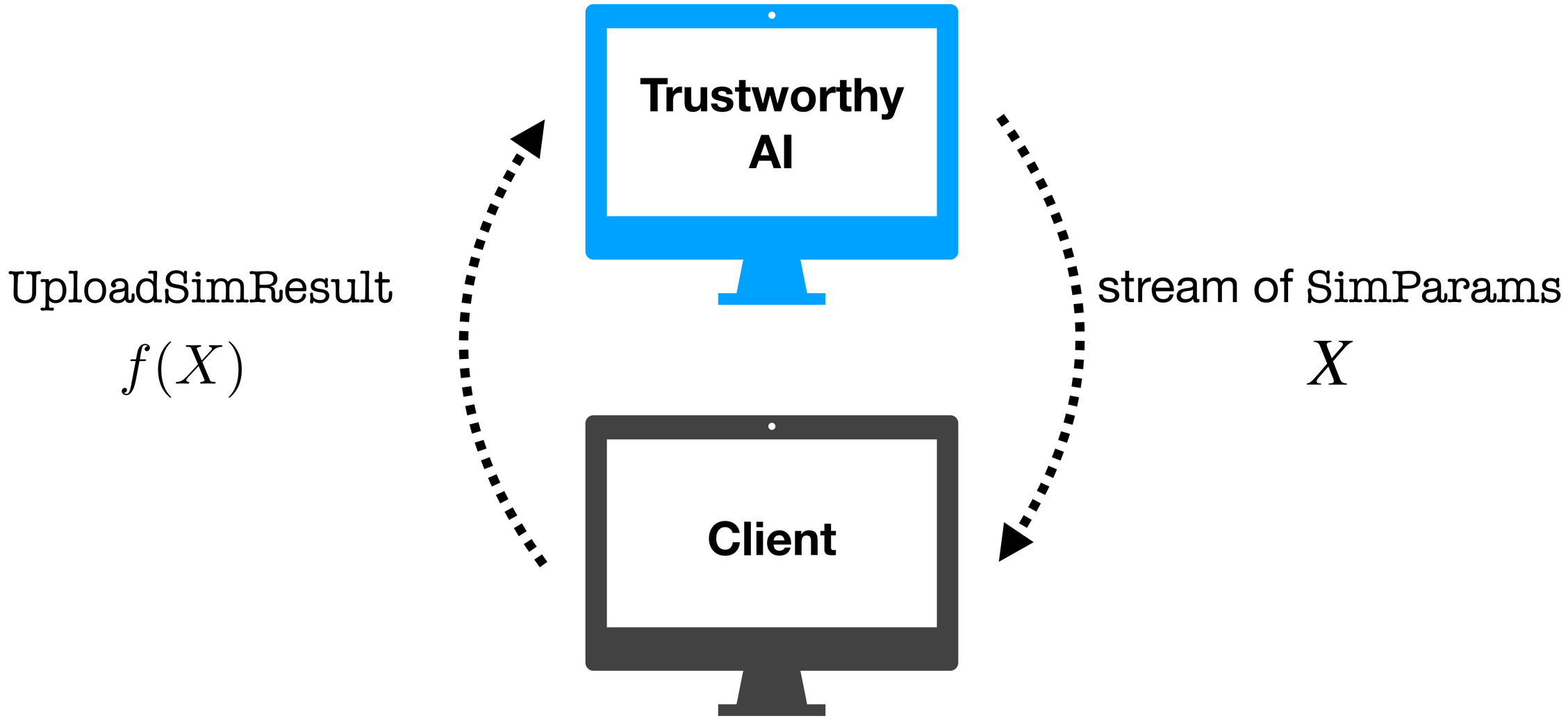


`OpenSimStream(Job)`

stream of `SimParams`

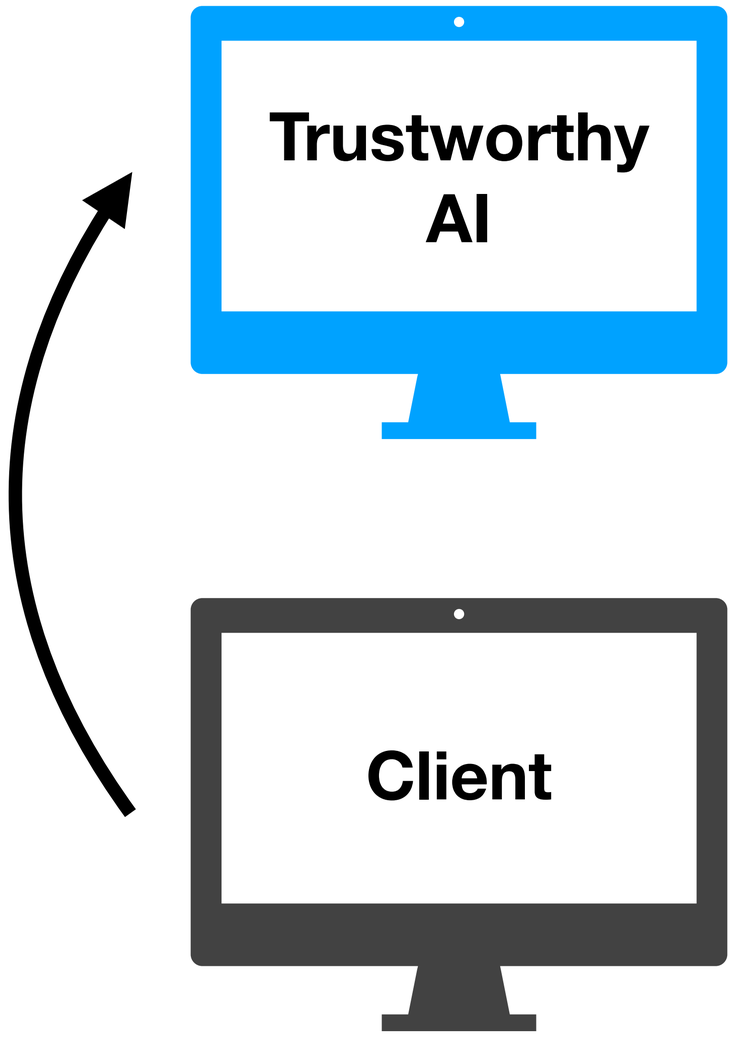
X

Asynchronous Loop



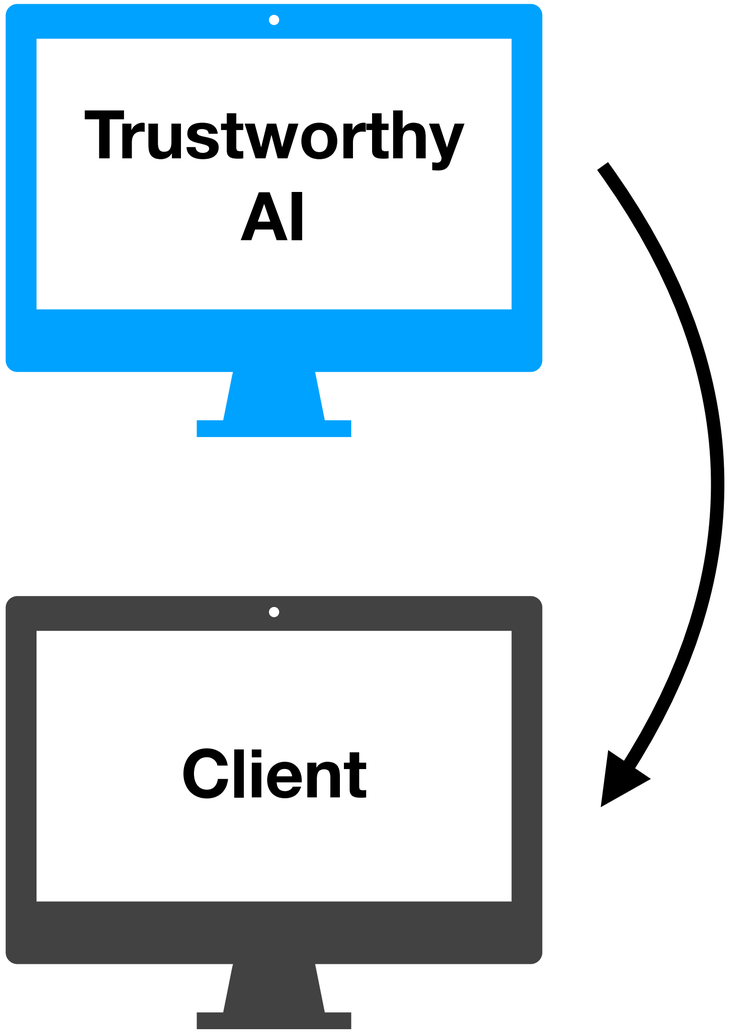
Request

GetJobResult(Job)



Request

Reply



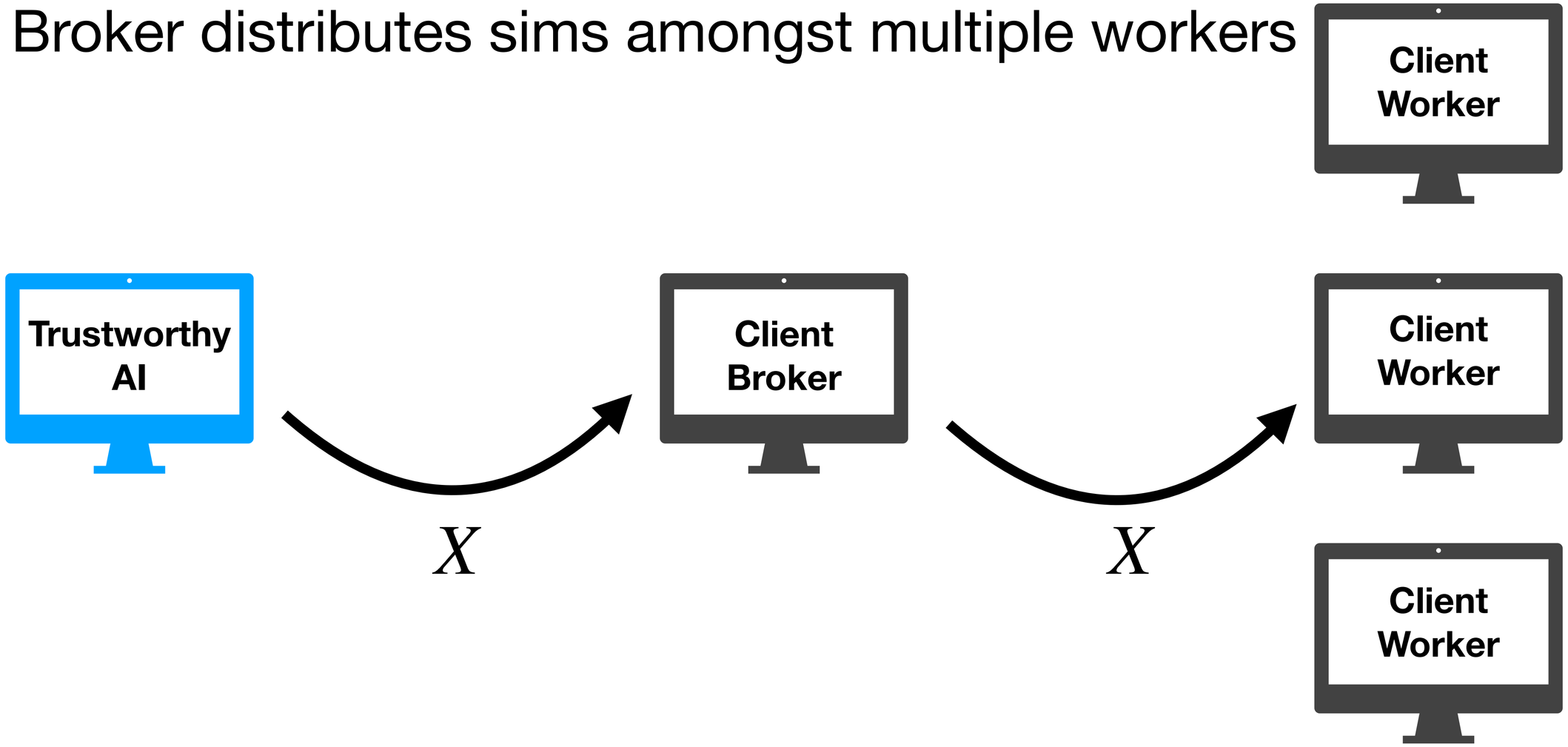
`GetJobResult(Job)`

`JobResult`

- Analysis of failures

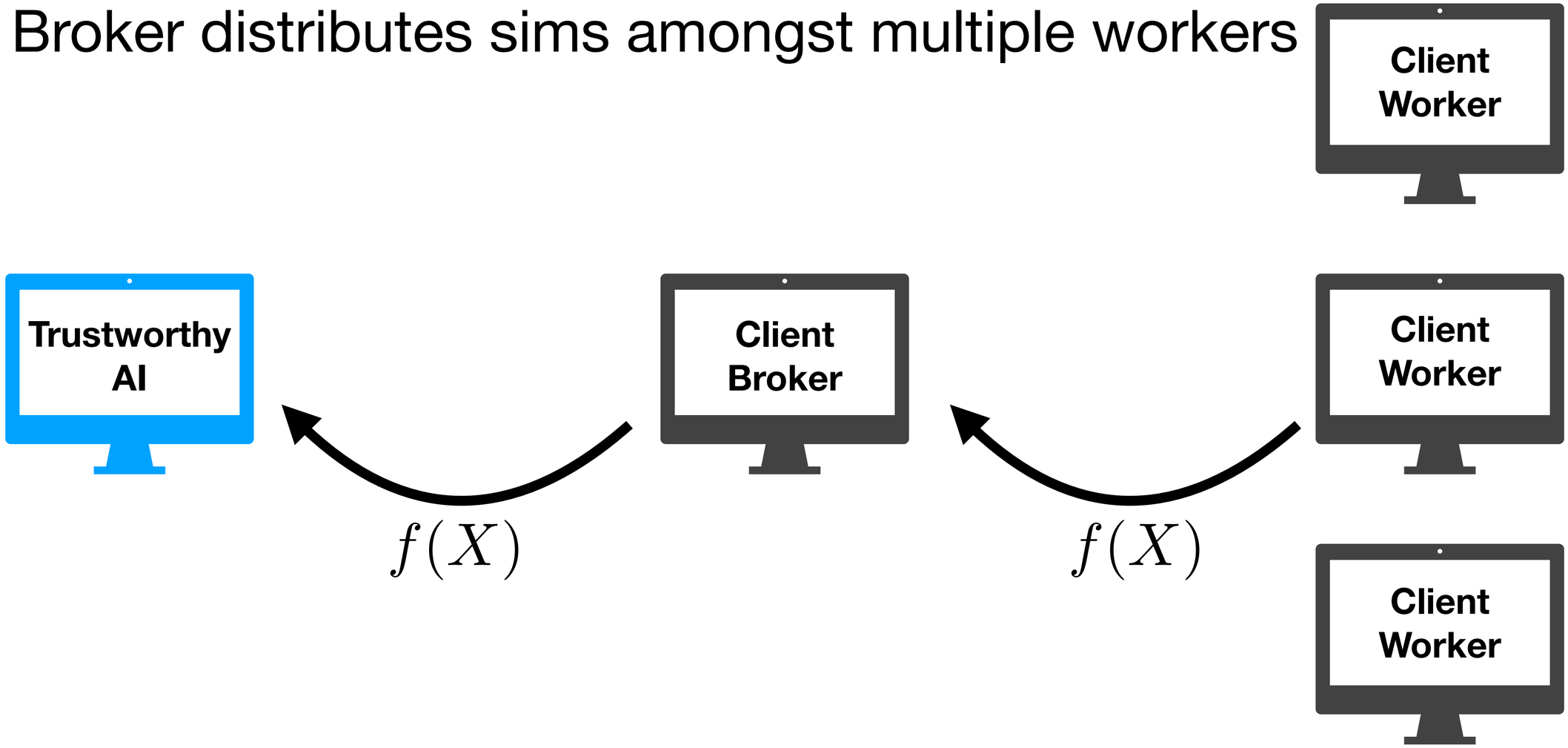
Brokered Communication

- Broker distributes sims amongst multiple workers



Brokered Communication

- Broker distributes sims amongst multiple workers



Brokered Communication

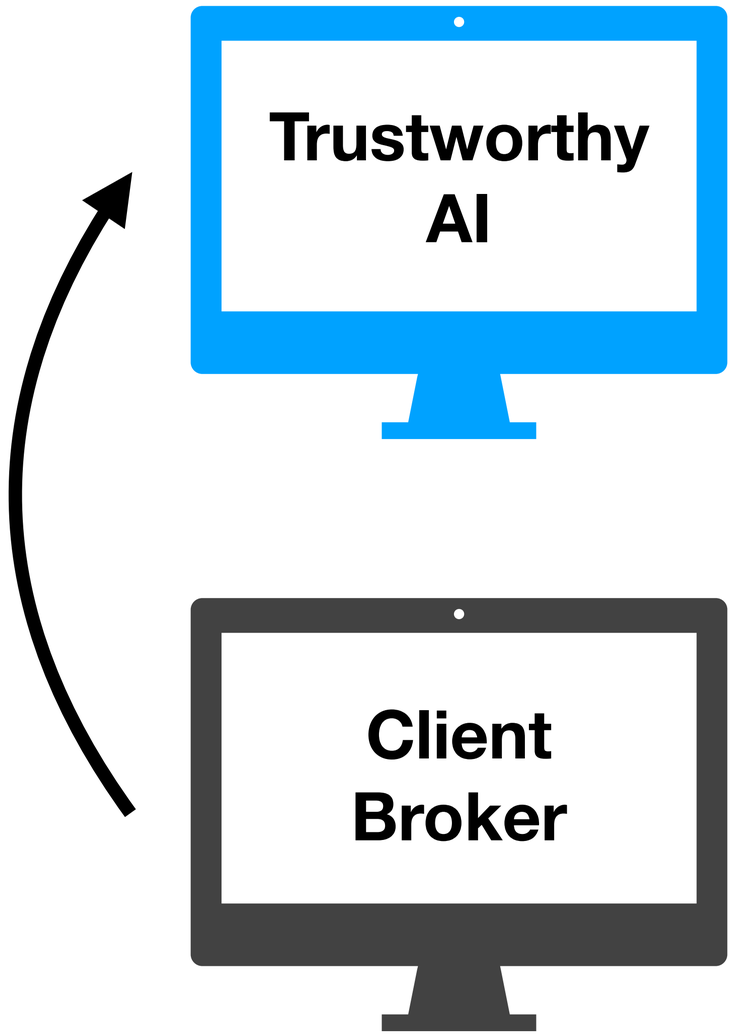
- Broker distributes sims amongst multiple workers

```
service Broker {
 rpc PushResult(SimResult) returns (Empty);
}
service Simulator {
 rpc Simulate(BrokerSimParams) returns (Empty);
 rpc RegisterBroker(BrokerPort) returns (Empty);
 rpc DeregisterBroker(BrokerPort) returns (Empty);
}
```

Request

`StartJob(JobRequest)`

- Search space P_0
- Threshold γ
- # of simulations N

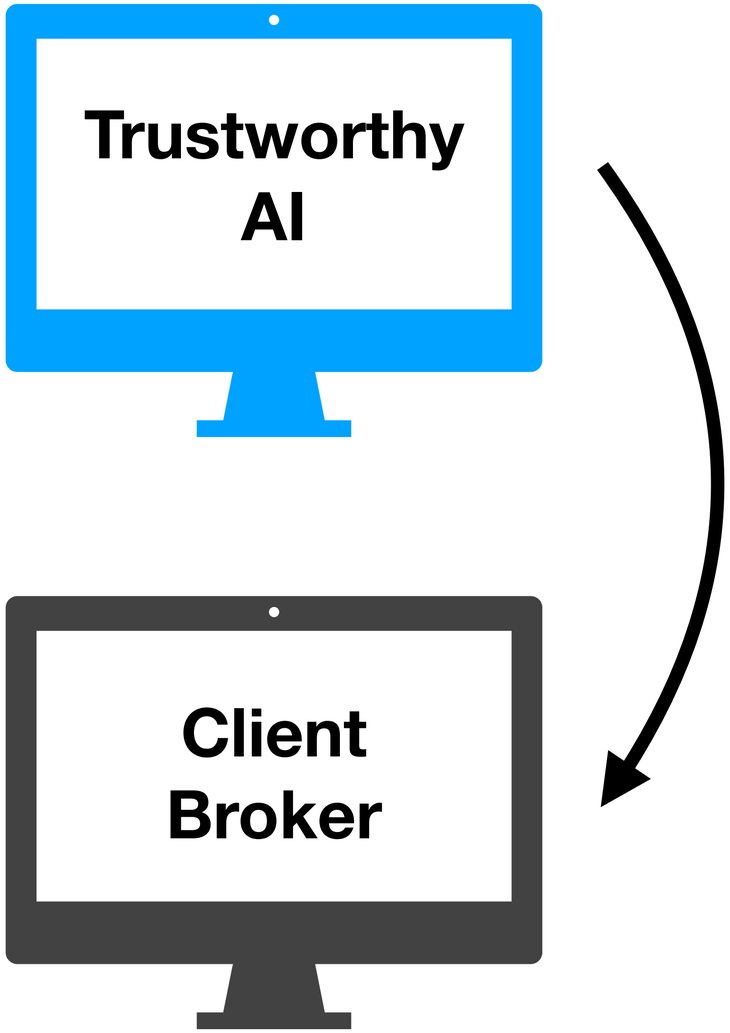


Example:

$P_0 = \mathcal{N}(0, I),\ \gamma = 2,\ N = 100$

Request

Reply



`StartJob(JobRequest)`

- Search space P_0
- Threshold γ
- # of simulations N

Example:

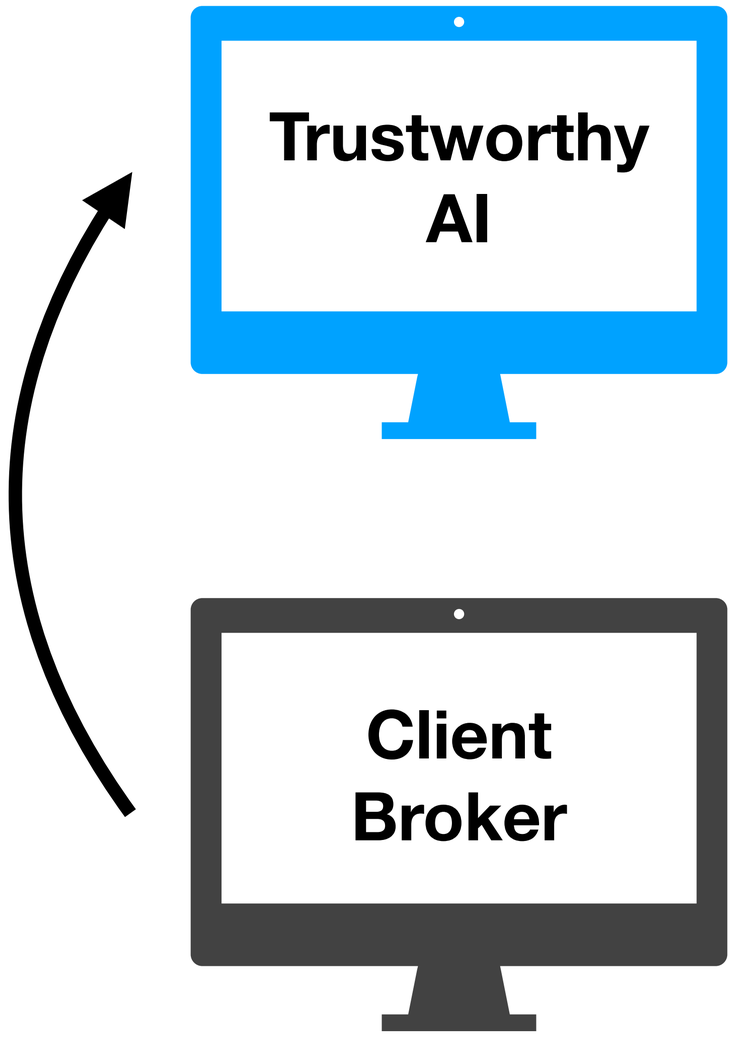
$P_0 = \mathcal{N}(0, I),\ \gamma = 2,\ N = 100$

`Job`

- Unique id for job

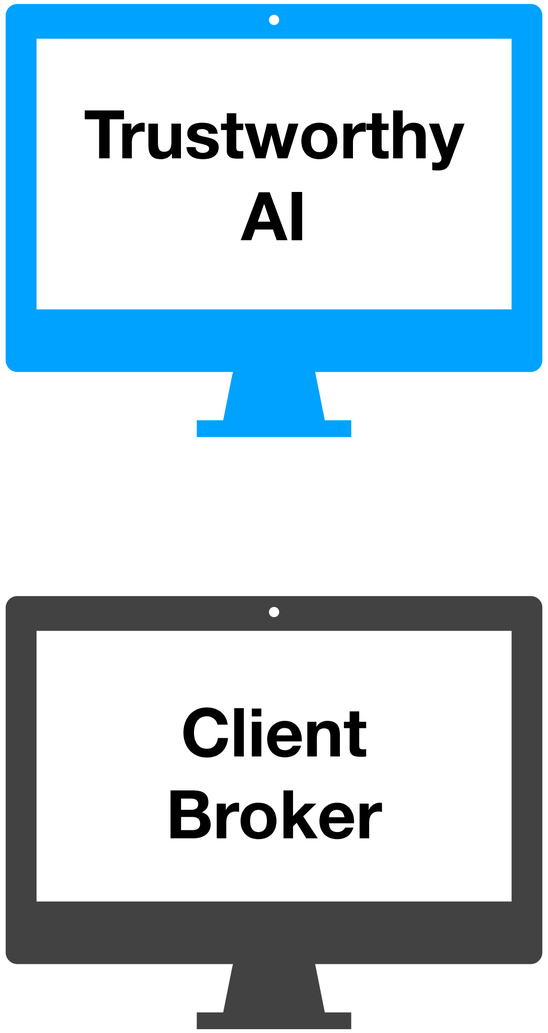
Request

OpenSimStream(Job)



Request

Replies



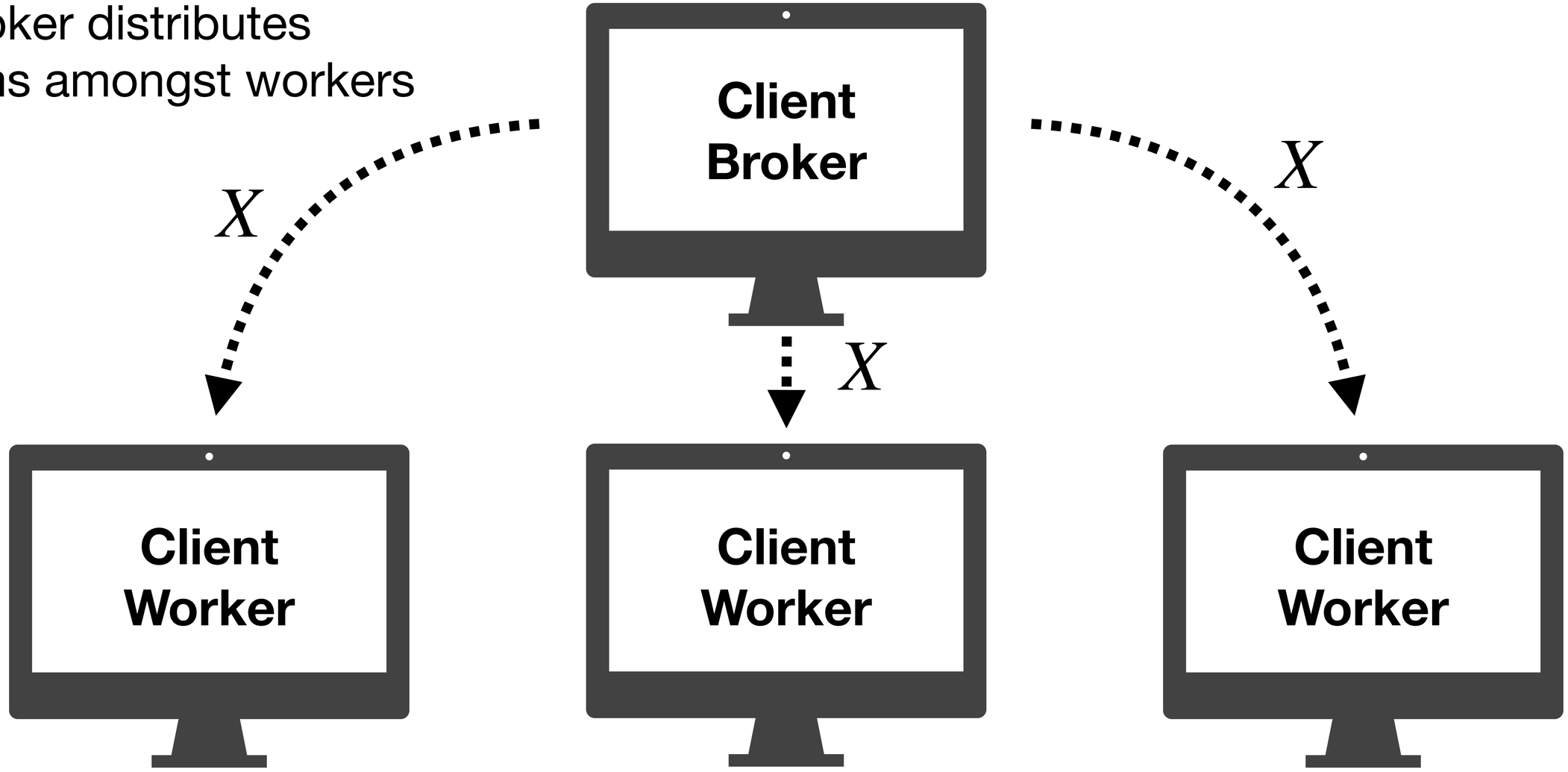
`OpenSimStream(Job)`

stream of `SimParams`

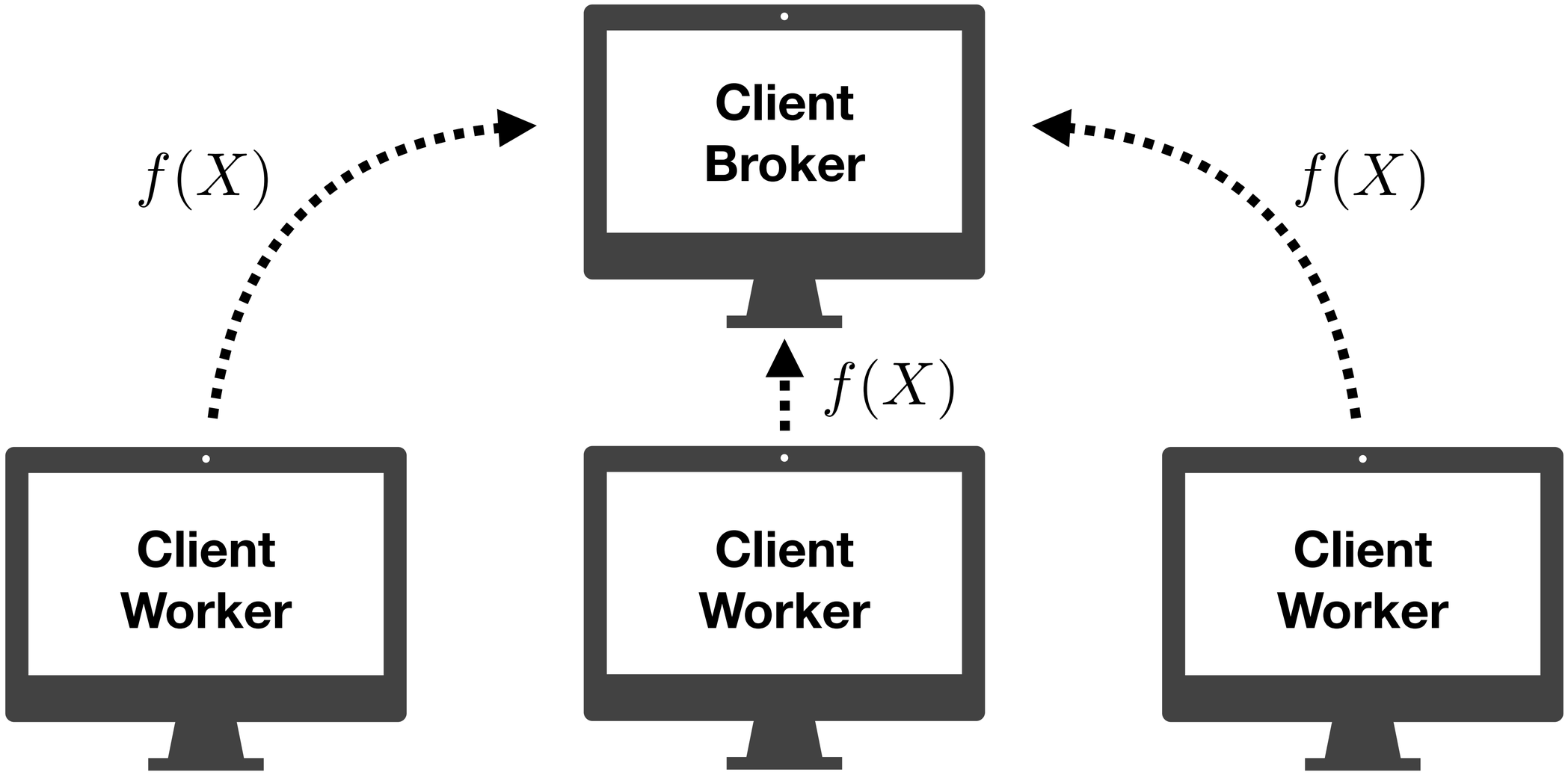
X

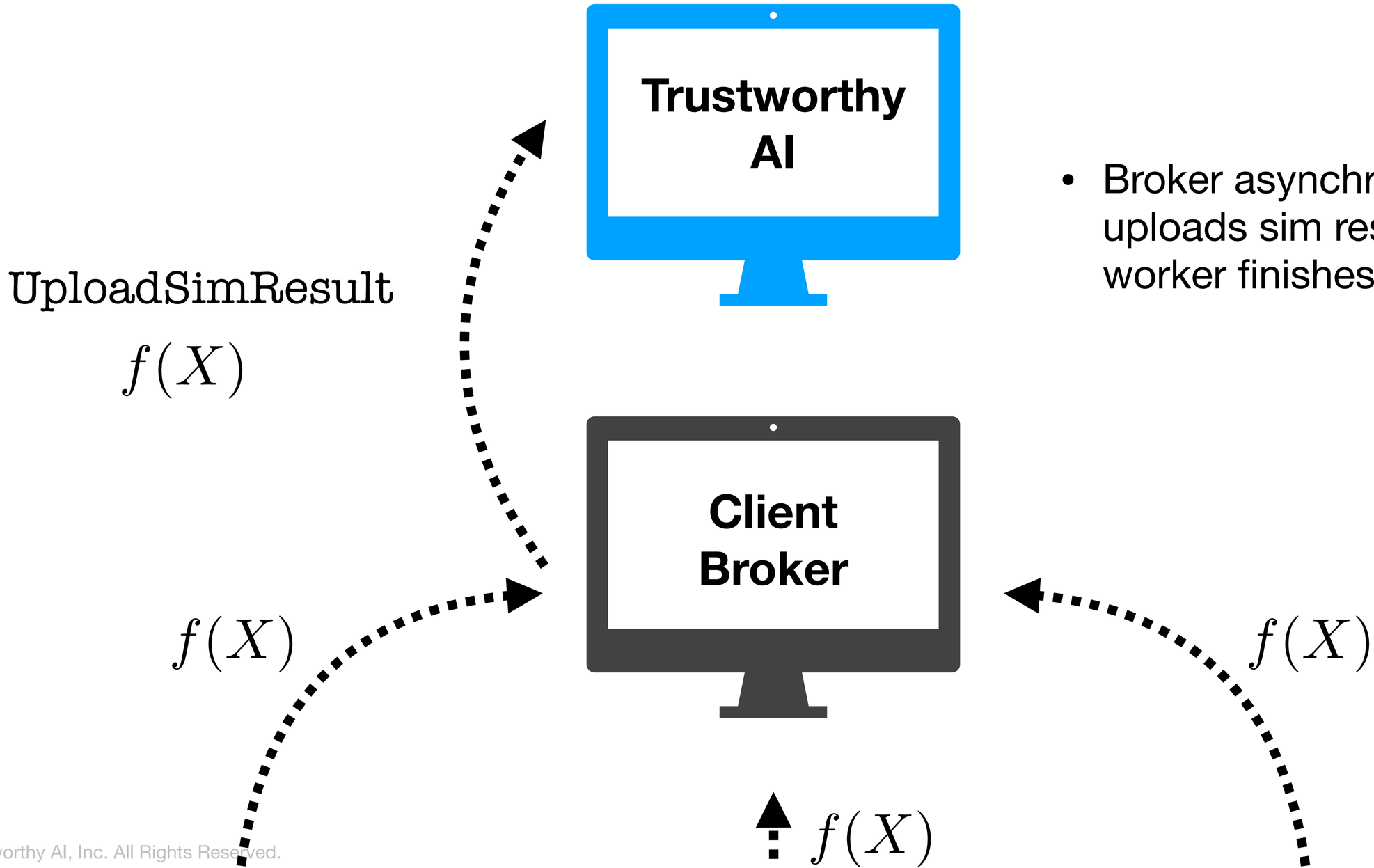
Simulate(BrokerSimParams)

- Broker distributes sims amongst workers



PushResult(SimResult)

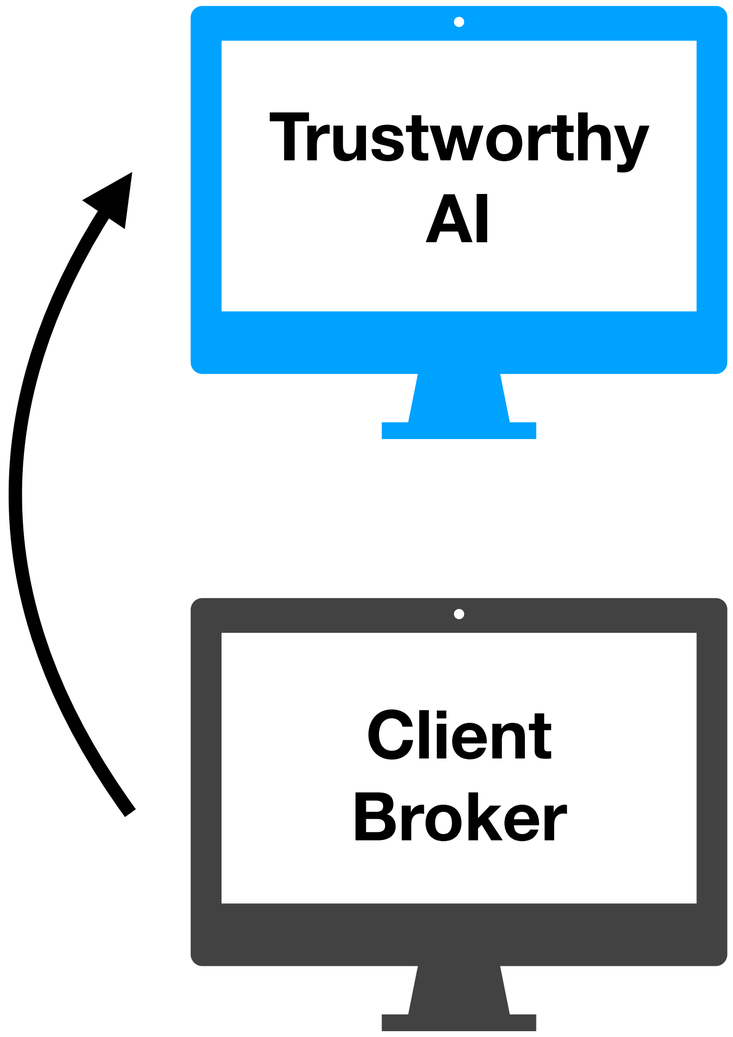




- Broker asynchronously uploads sim results as each worker finishes

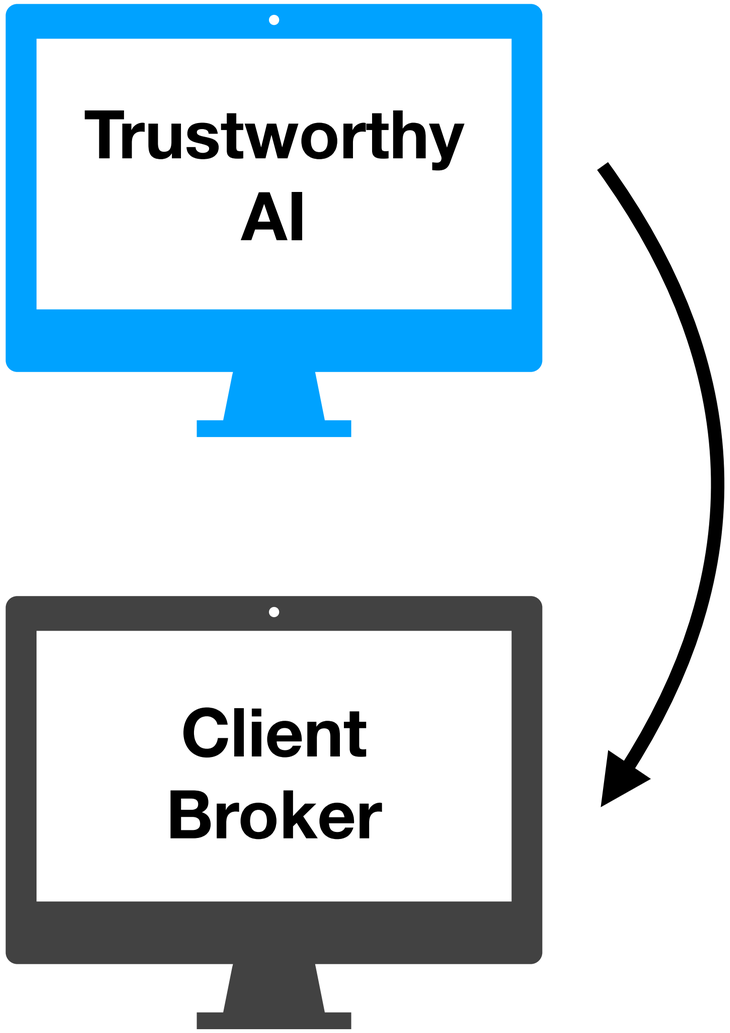
Request

GetJobResult(Job)



Request

Reply



`GetJobResult(Job)`

`JobResult`

- Analysis of failures