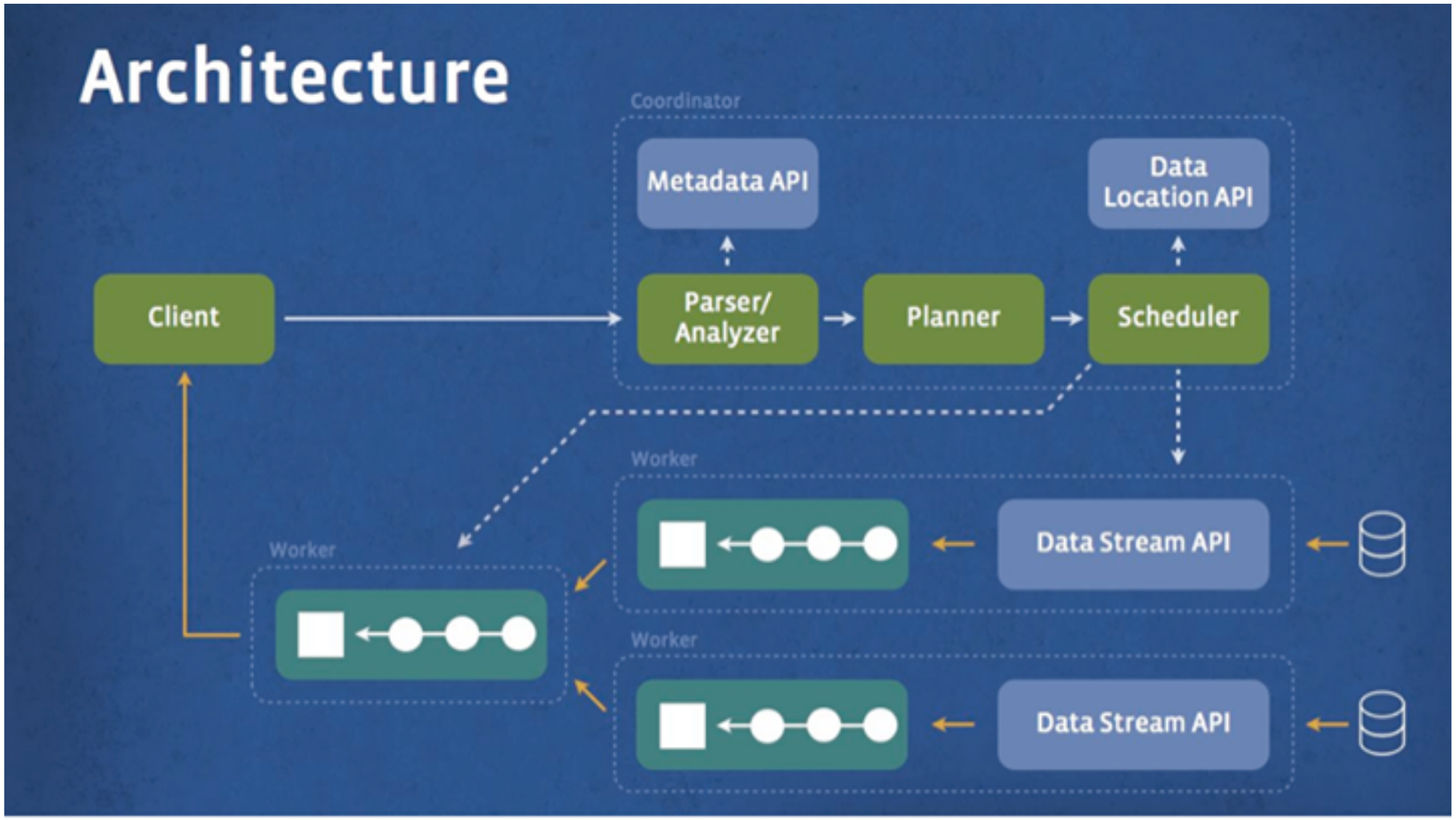
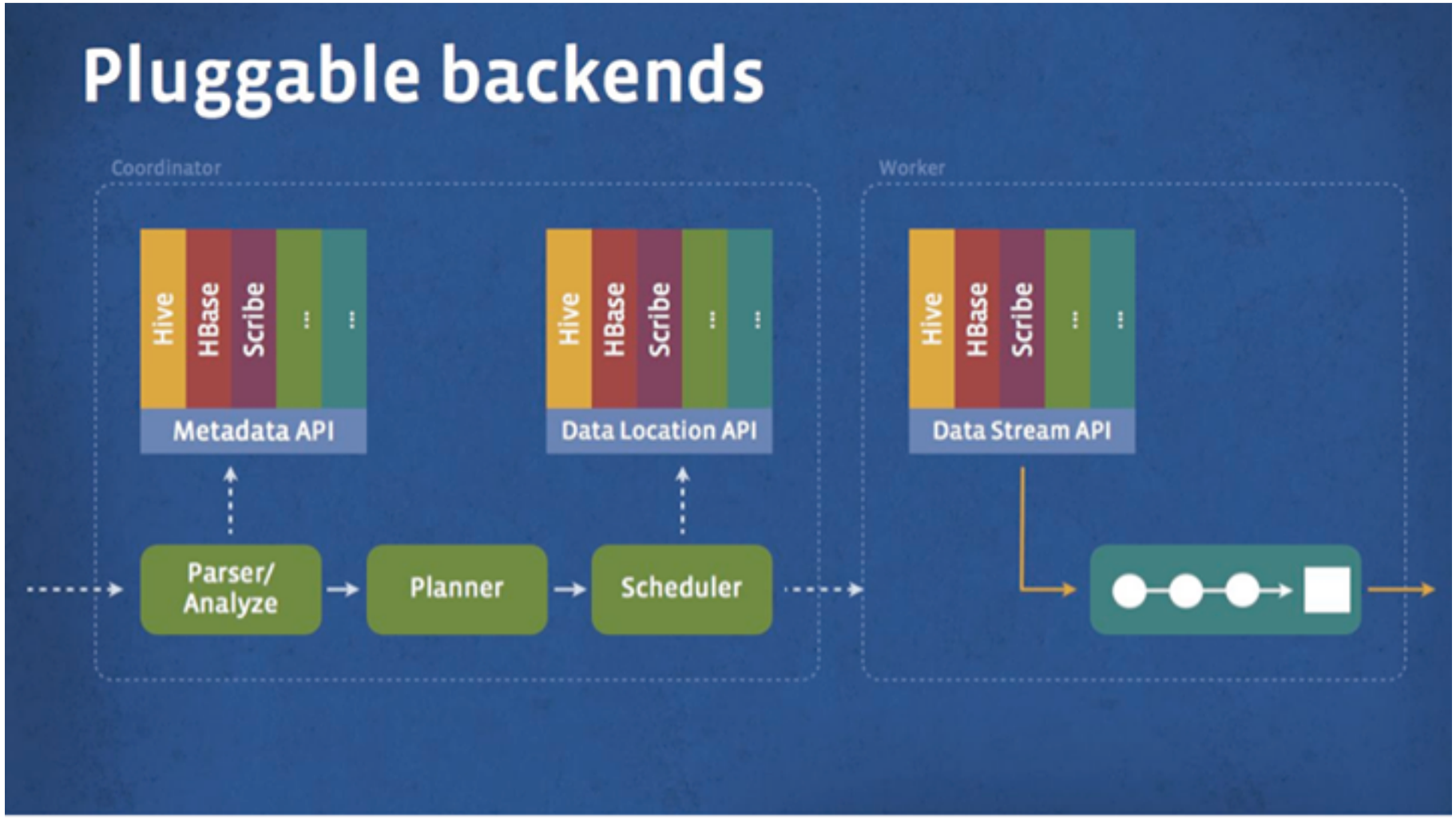
Sadayuki Furuhashi

Treasure Data, Inc.

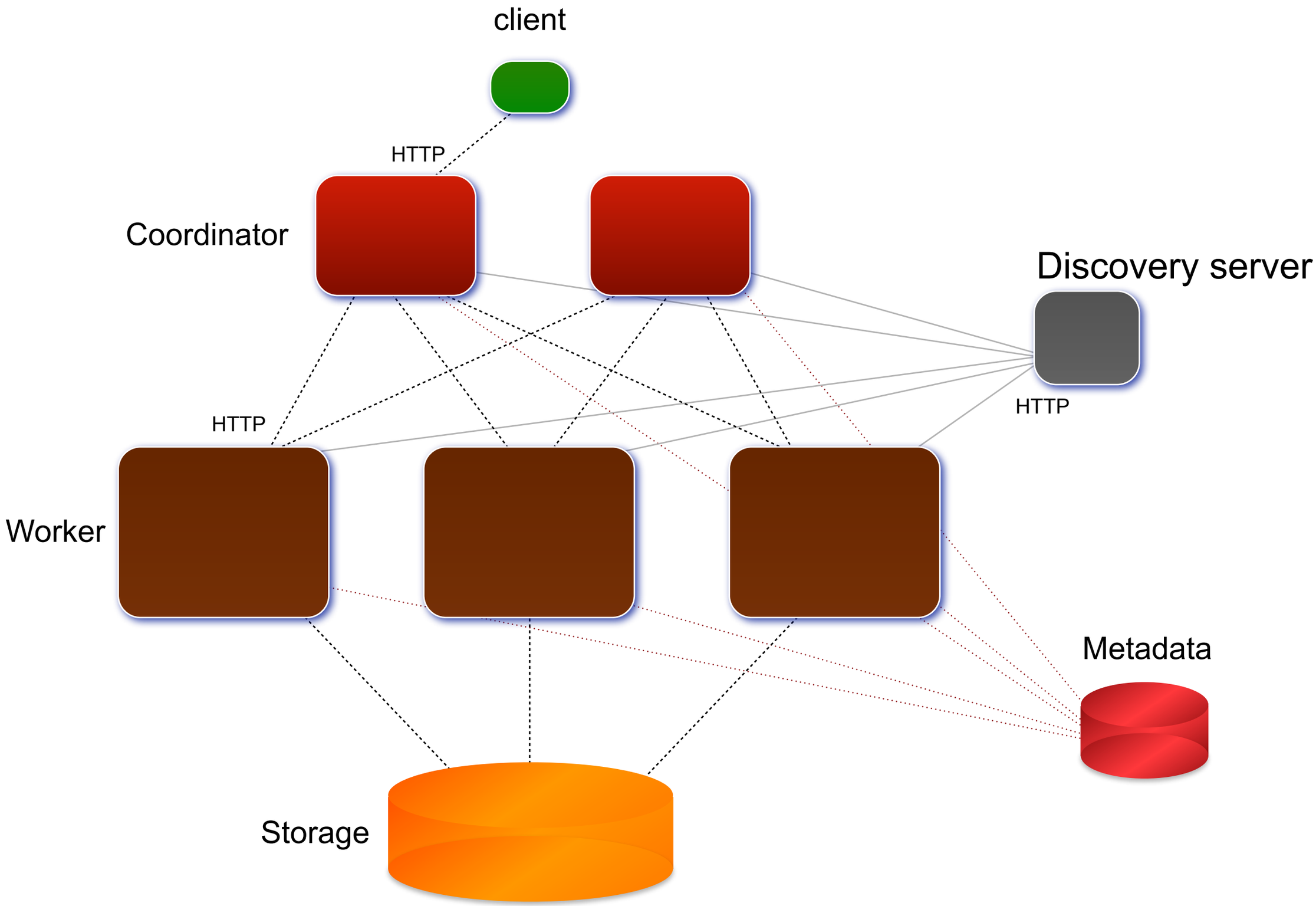
Founder & Software Architect



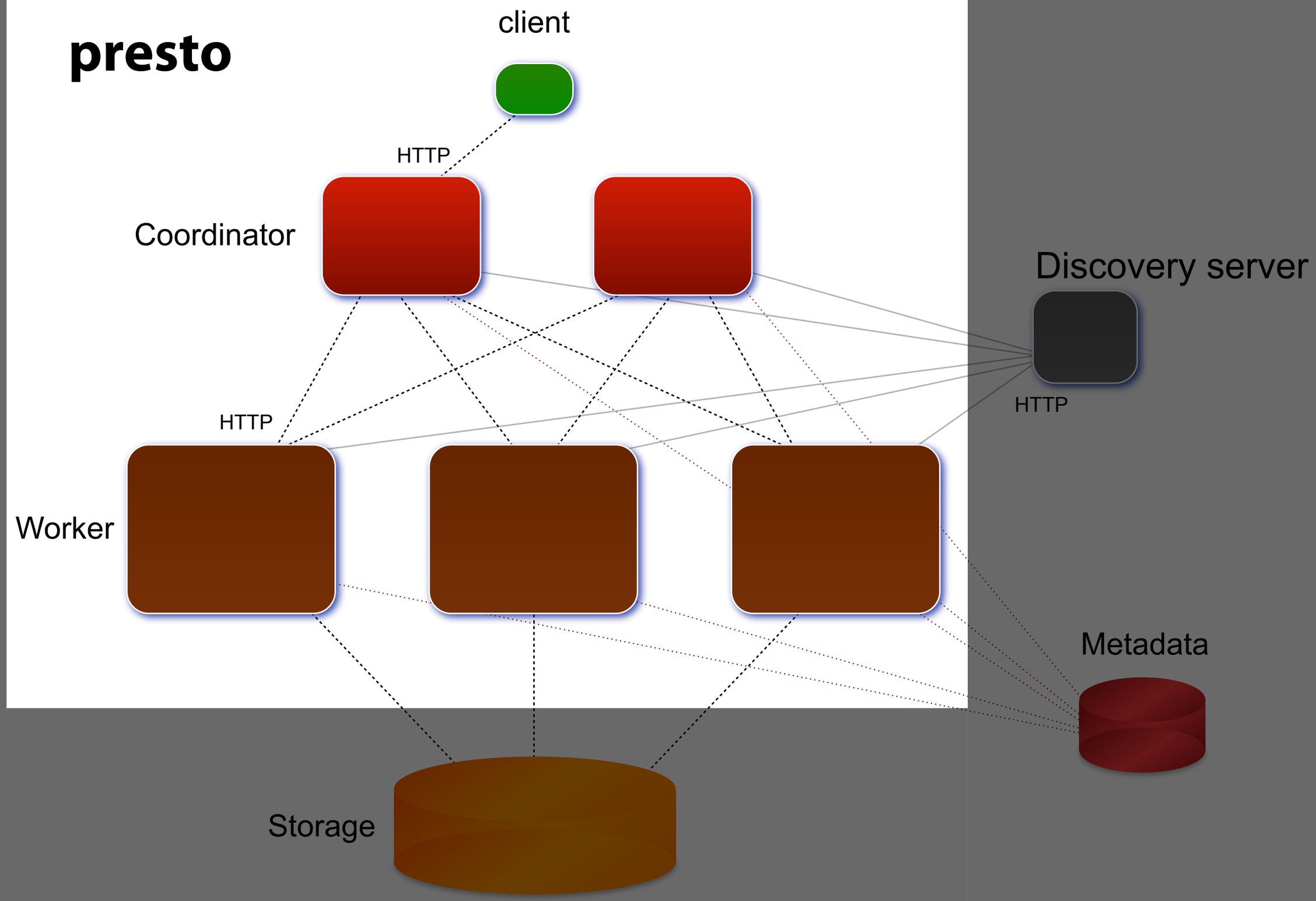
https://www.facebook.com/notes/facebook-engineering/presto-interacting-with-petabytes-of-data-at-facebook/10151786197628920



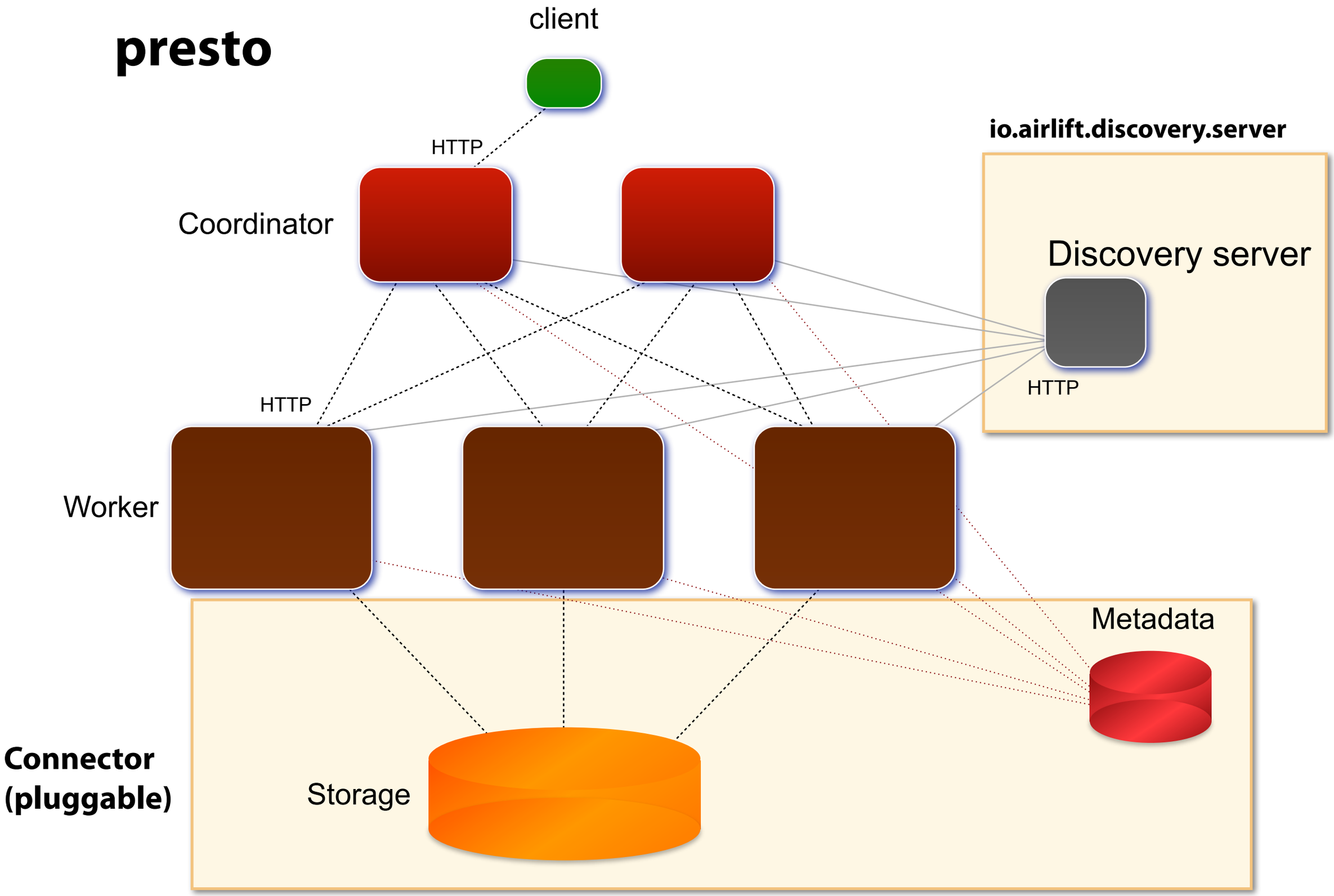
https://www.facebook.com/notes/facebook-engineering/presto-interacting-with-petabytes-of-data-at-facebook/10151786197628920


client
HTTP
Coordinator
Discovery server
HTTP
HTTP
Worker
Metadata
Storage

presto


client
HTTP
Coordinator
Discovery server
HTTP
HTTP
Worker
Metadata
Storage

presto


client
HTTP
io.airlift.discovery.server
Coordinator
Discovery server
HTTP
HTTP
Worker
Metadata
Connector
(pluggable)
Storage

Connector

A connector consists of 3 major components:

Metadata

- similar to Hive's metastore
- provide table schema to Presto

SplitManager

- similar to Hadoop's InputFormat
- partitioning, WHERE condition pushdown etc.

RecordCursor

- similar to Hadoop's RecordReader

presto-hive connector

Built-in hive connector

Hive Metadata

- read metadata from Hive Metastore
- Presto's Metadata is read-only (for now)

Hive SplitManager

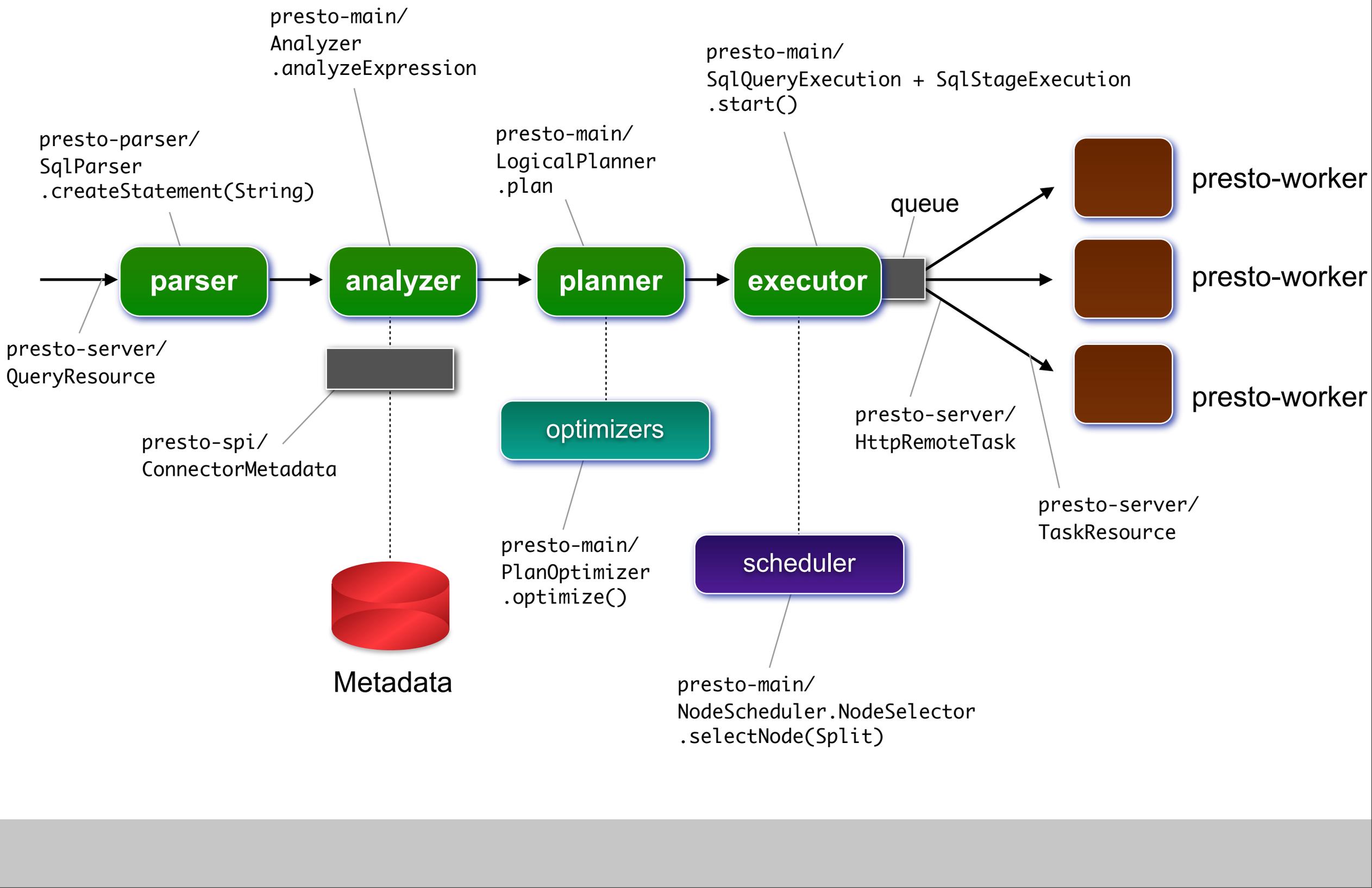
- Manages Hive's Partitions on HDFS

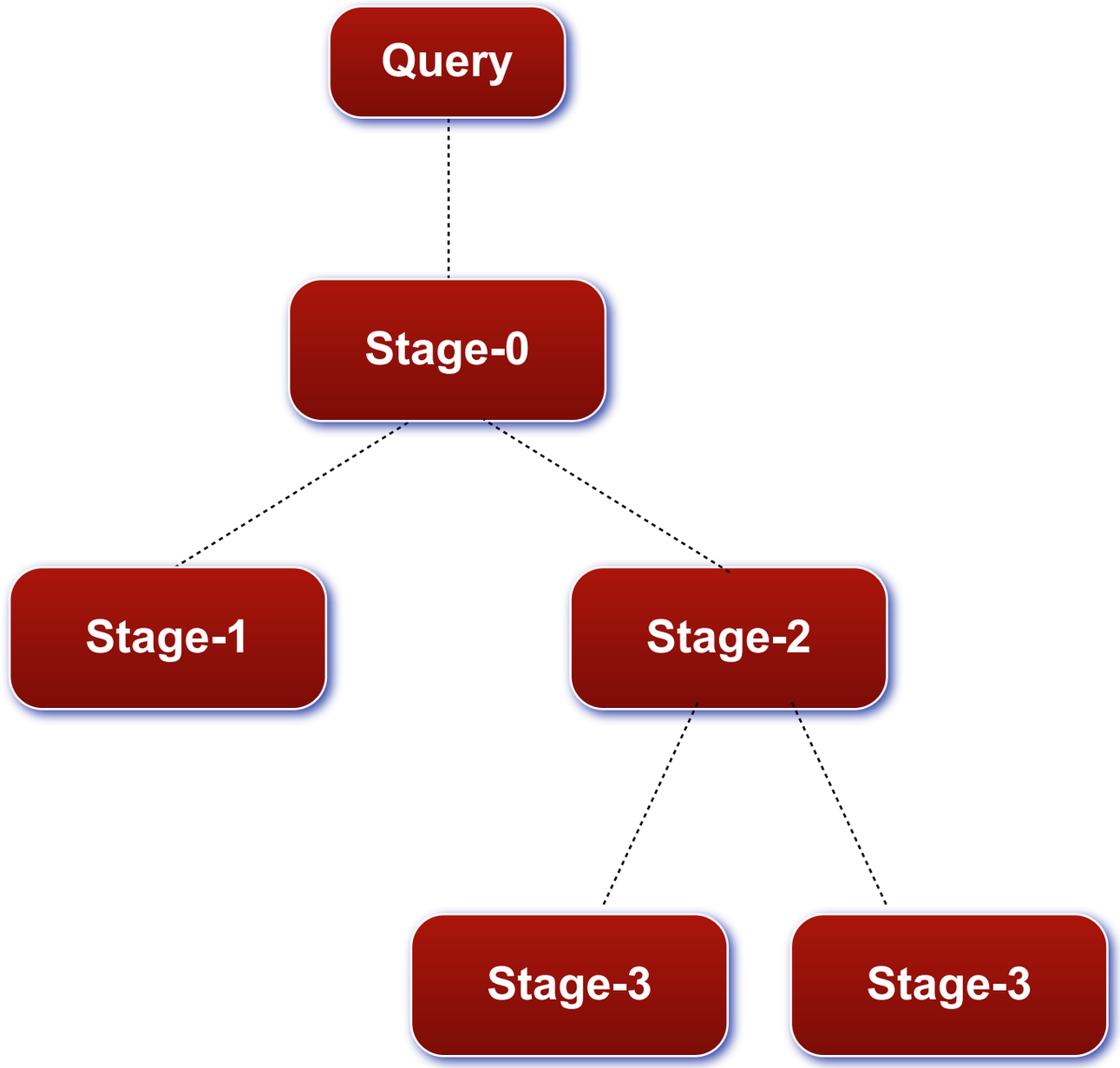
Hive RecordCursor

- Adapter class to use Hive's RecordReader

Connectors

- **example-http**
  - Metadata: Get table list from remote REST API server in JSON
  - RecordCursor: Get data from remote REST API server in CSV
- **jmx**
- **tpch**
- **information-schema**
- **system**


presto-main/
Analyzer
.analyzeExpression
presto-main/
SqlQueryExecution + SqlStageExecution
.start()
presto-parser/
SqlParser
.createStatement(String)
presto-main/
LogicalPlanner
.plan
queue
parser
analyzer
planner
executor
presto-worker
presto-worker
presto-worker
presto-server/
QueryResource
presto-spi/
ConnectorMetadata
optimizers
presto-server/
HttpRemoteTask
presto-server/
TaskResource
presto-main/
PlanOptimizer
.optimize()
scheduler
Metadata
presto-main/
NodeScheduler.NodeSelector
.selectNode(Split)


Query
Stage-0
Stage-1
Stage-2
Stage-3
Stage-3

Discussions

Logical types

they have a plan to have type customization system. Internally, it has only limited number of types. But users (or developers) can create any logical types that is composed of physical types.

HA

Once query runs, and if a task failed, Presto can't recover the task. Client needs to rerun the query.

Client heartbeat

Presto coordinator has heartbeat between client. If client doesn't poll for certain configured time, it kills the query.

Result downloading

Clients get query result from coordinator but coordinator itself doesn't store results. Worker stores it. Coordinator acts as a proxy for clients to download result from workers.

Scheduling

Memory consumption is limited by a configuration parameter (task.max-memory). If task exceeds the limit, the query fails. To prevent it, we need limitation in coordinator or clients.

CPU limit is implemented. Scheduler in workers can stop running operators (task consists of operators) so that it can schedule tasks fairly.

Spill to disk

Presto doesn't spill data to disk even if memory run out

JOIN

They have plan to implement JOIN across two large tables. one plan is to have "medium join"

Window function

They have plan to extend API of window functions (in next several weeks).

Window functions run on a single worker server, even if it has PARTITION BY.

Open-source

Primary development repository is the public repository on Github. They don't have internal repositories excepting connectors.

They use pull-request even in facebook.

https://github.com/facebook/presto/pull/832

Source code

- **airlift**
  - Slice
  - See also: Netty 4 at Twitter: Reduced GC Overhead
- **Google Juice**
  - Dependency injection
- **Jackson**
  - JSON serialization of model classes