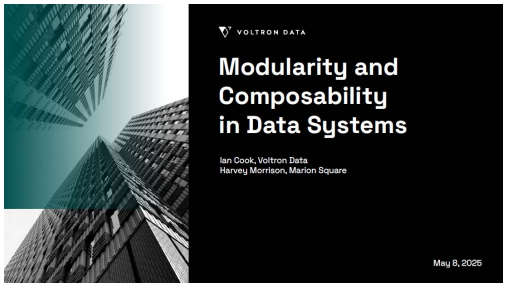
VOLTRON DATA

carahsoft



Modularity and Composability in Data Systems

Thank you for downloading this Voltron Data resource. Carahsoft is the distributor for Voltron Data's AI and ML solutions available via NASA SEWP V, ITES-SW2, NASPO ValuePoint, and many more contract vehicles.

To learn how to take the next step toward acquiring Voltron Data's solutions, please check out the following resources and information:

For Scale.AI overview:
carah.io/Voltron-Data

For upcoming events:
carah.io/Voltron-Events

For additional resources:
carah.io/Voltron-Resources

For additional Artificial Intelligence:
carah.io/ai-solutions

To set up a meeting:
VoltronData@carahsoft.com

To purchase, check out the contract vehicles available for procurement:
carah.io/procurement

VOLTRON DATA

Modularity and Composability in Data Systems

Ian Cook, Voltron Data
Harvey Morrison, Marion Square

May 8, 2025

Federal Data Challenges

Your Models Are Ready. Your Infrastructure Isn't.

Examples

- Federal agencies are embracing AI, ML, and real-time analytics
- But data pipelines remain CPU-bound, slow, and expensive
- OMB M-25-22 and DoD AI Strategy both call for efficient, scalable, and interoperable AI solutions

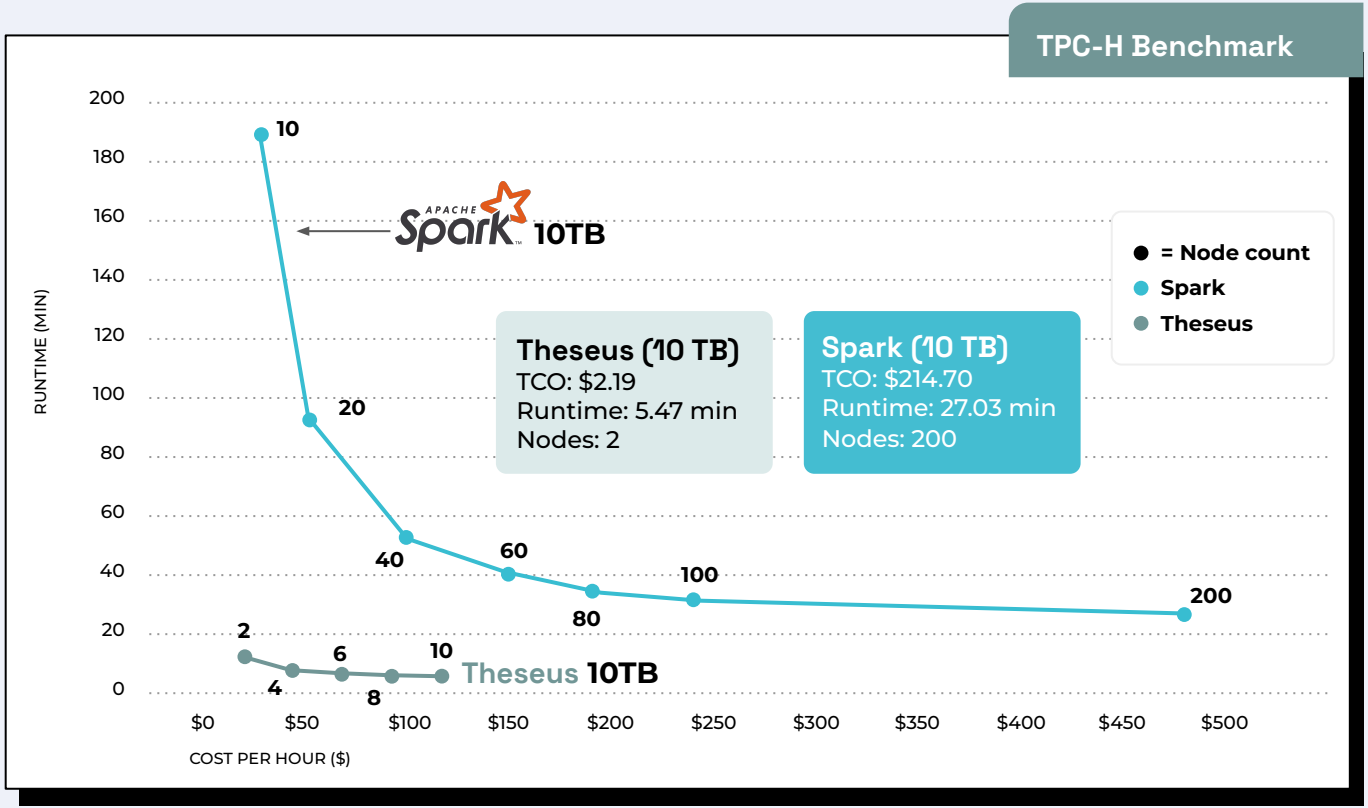
"Analyze large data volumes rapidly and affordably to support national security."

— DoD AI Strategy, 2025

Accelerate Analytics. Cut Costs. Improve Accuracy.

Challenge	Theseus Delivers	Impact
Long ETL Runtimes	20 hrs—20 min	Faster model iteration
Expensive Compute	200 CPU’s—2 GPU’s	100X Cost Savings
Limited Data Scope	Full Fidelity Training	Higher model accuracy

Theseus maximizes TCO



~100x cost reduction

~5x faster execution

100x fewer nodes

Theseus: Built for Interoperability, Trust, and Mission Scale

- Composable architecture built on Apache Arrow & Ibis
- Compatible with existing systems (cloud, on-prem, air-gapped)
- No vendor lock-in → aligns with **OMB M-25-22**
- Enables secure, rapid AI adoption at petabyte scale

“Voltron Data’s mission is to drive the marginal cost of analytics to zero.”

Modularity and Composability in Data Systems

01 Examples and key points

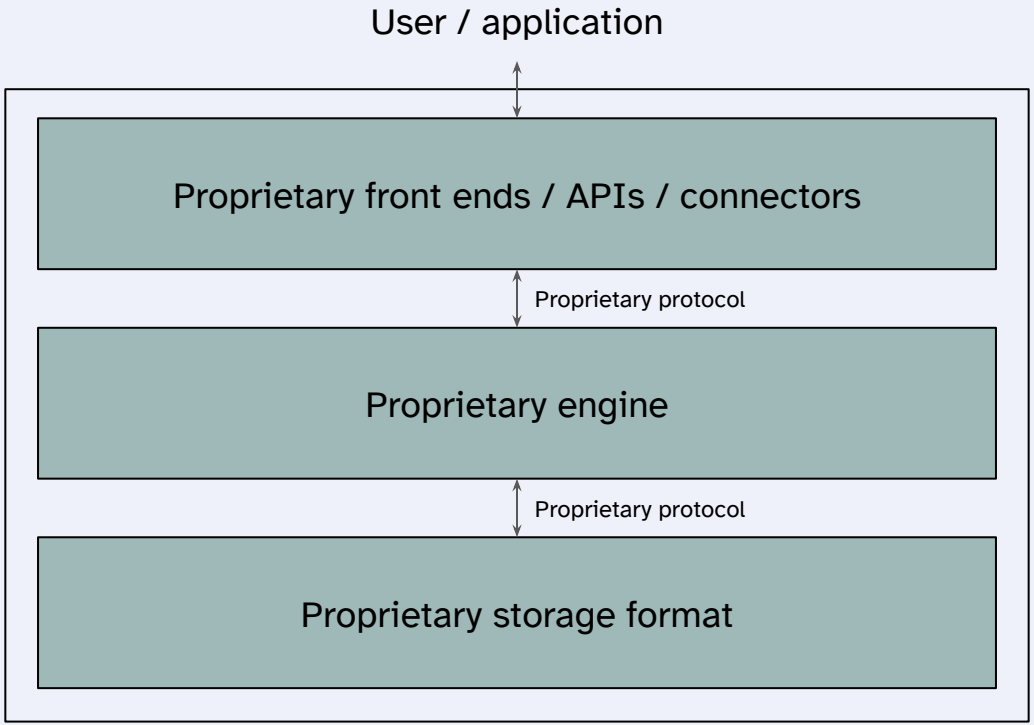
02 Terminology and history

03 Resources

01 Examples and key points

A vertically integrated data stack

Minimal example:



Vertically integrated data stacks

Examples

- **Relational database systems**
  - Oracle
  - PostgreSQL
  - ...
- **Traditional data warehouses**
  - Teradata
  - Vertica
  - ...
- **Cloud data warehouses** (in certain configurations)
  - Snowflake
  - BigQuery
  - ...
- **Custom-built in-house data systems**

Vertically integrated data stacks

Advantages

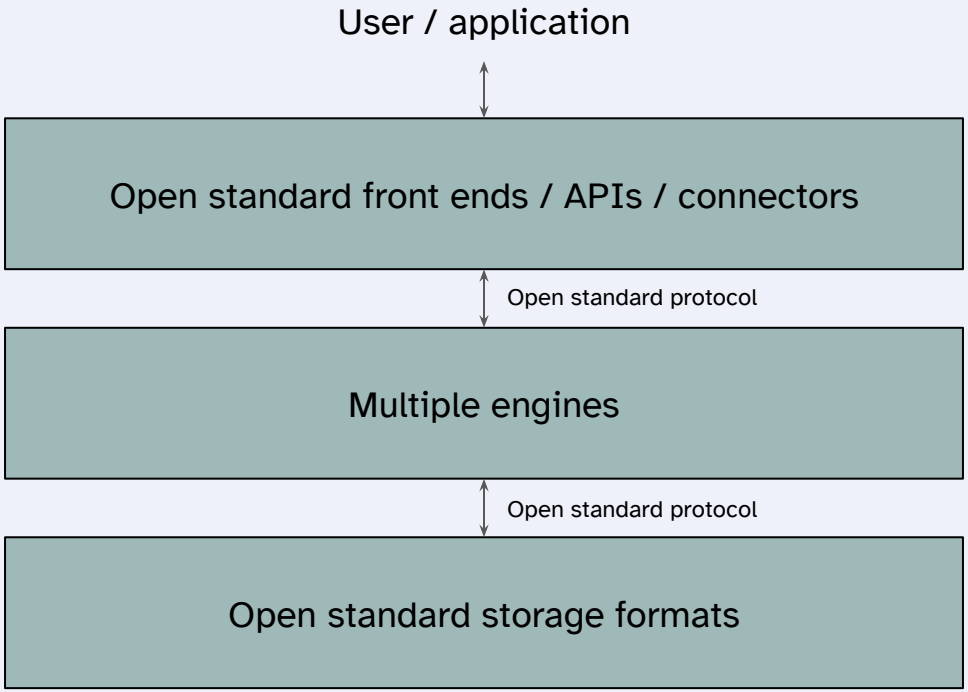
- Simplicity
  - For end users
  - For architects and administrators
- Full vendor support
- Theoretically: better features and performance
  - Because vendor need not compromise for the sake of interoperability

Disadvantages

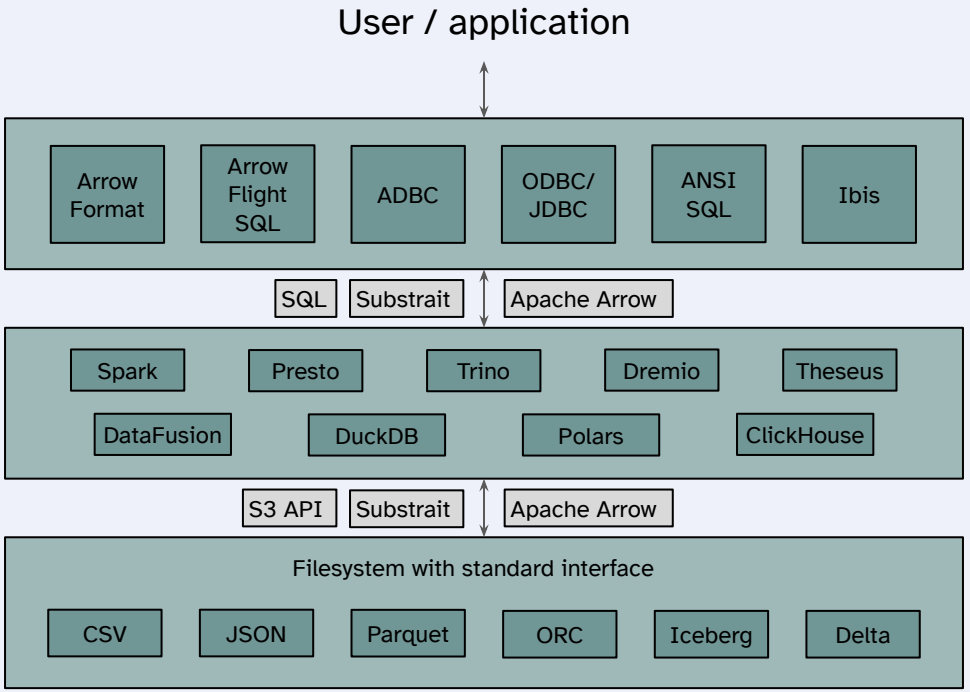
- Lack of choices
- Vendor lock-in
- Difficulty integrating third-party products
- Siloed architecture
- In practice: worse features and performance
  - Because vendor lacks focus and incentives to innovate

A composable data stack

Minimal example:



A composable data stack



Other elements of a composable data stack

- **Resource orchestration**
  - Kubernetes
- **Workflow orchestration**
  - Airflow
  - Prefect
  - Dagster
- **Planners and optimizers**
  - Calcite
- **Catalogs**
  - Iceberg REST catalog
  - Hive Metastore
  - Polaris Catalog
  - Unity Catalog
- **Streaming sources, sinks, and engines**
  - Kafka protocol
  - Flink
- **Observability**
  - OpenTelemetry
- **Lineage**
  - OpenLineage
- **Data Security**
  - Ranger-based standards and tools

Composable data stacks

Advantages

- Efficiency from specialization
- Flexibility and choice
- Freedom from vendor lock-in
- Ability to swap technologies without migration
- Ability to take advantage of modern hardware
- Ability to stay on technical forefront

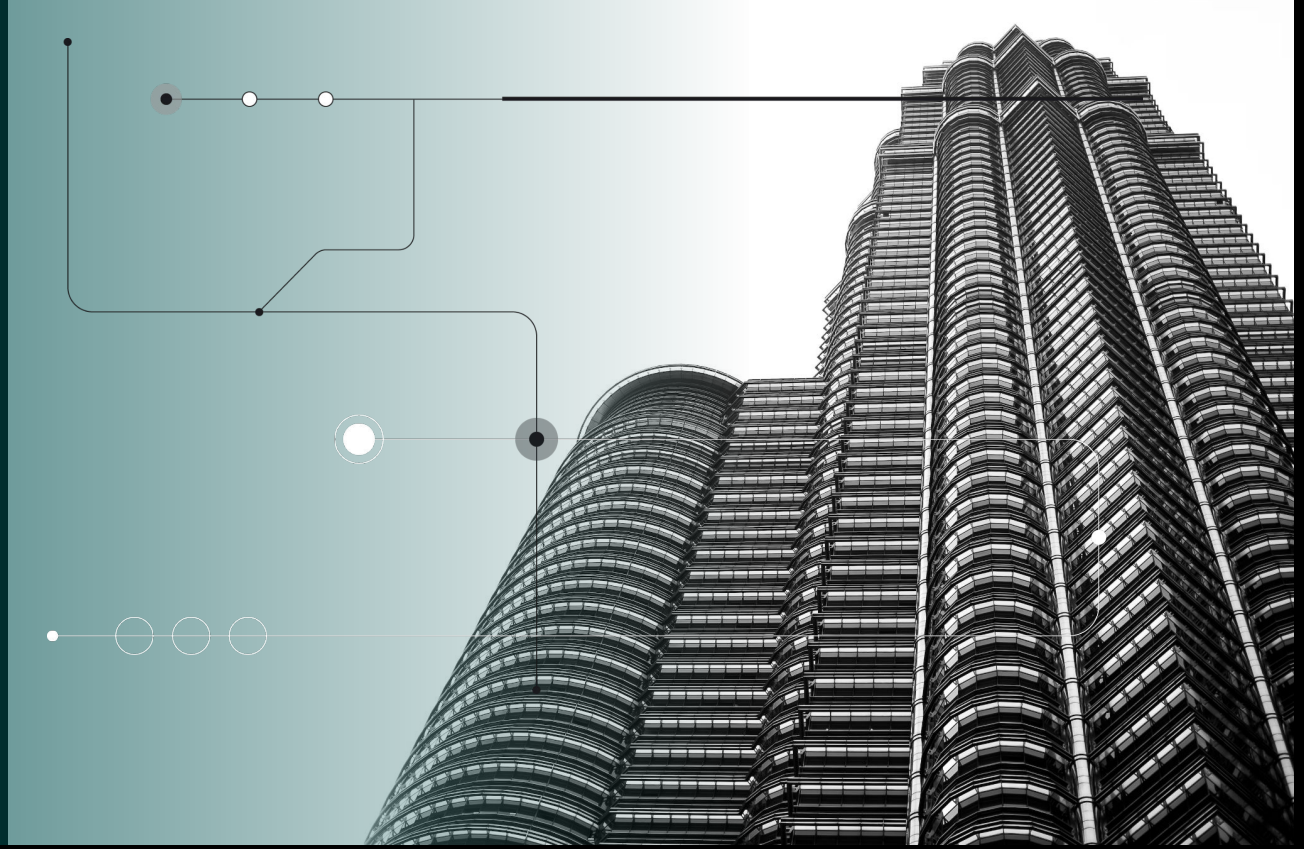
Disadvantages

- Higher cost and complexity to manage
- Need to engage with multiple vendors and open source communities

Composable data stacks

Not all-or-nothing

- Platforms like Snowflake and Databricks can separate compute and storage (external tables)
- Spark uses Arrow for data movement
- Snowflake supports Arrow as a result data format
- Vertically integrated data stacks support ODBC and JDBC (which are open standards)



02 Terminology and history

Terminology

- The components are **modular**
- The stack is **composable**

- The opposite of modular is **tightly coupled**
- The opposite of composable is **vertically integrated**

- **Monolithic** does not always imply tightly coupled

- Composable data **stack** and composable data **system** are synonymous
- Avoid the term composable data **platform**

Origins

- The concepts of modularity and composability arose in **architecture** in the 1930s
- They were applied to **computer hardware** in the 1950s–1960s
- In software engineering, these ideas became associated with **microservices**
  - And earlier with **service-oriented architecture (SOA)**
- Later, they became associated with **Hadoop**
  - But Hadoop never fully implemented them

Momentum

- Recent growth in awareness of modularity and composability in data systems
  - Data systems research community (VLDB, SIGMOD, CIDR)
  - Startup companies (Tabular, Voltron Data, InfluxDB, ...)
- Technical feasibility has improved and cost to implement has declined
  - Driven by maturity of Arrow, Parquet, engines, and table formats
- Growing frustration with vertically integrated stacks
  - Unexpectedly high costs
  - Customer pressure to separate compute and storage

03 Resources

Recommended Reading

- Origins in Hadoop
  - The Modern Data Architecture: The Deconstructed Database
  - Hadoop Is Dead. Long Live Hadoop.
- Separation of compute and storage
  - The Case for Independent Storage
- The vision for composable data systems
  - The Composable Data Management System Manifesto
  - The Composable Codex

Recommended Talks

- Overview of work toward composable data systems
  - The Future Roadmap for the Composable Data Stack (Wes McKinney)
- Example of a data platform built from modular parts
  - Building InfluxDB 3.0 with Apache Arrow, DataFusion, Flight and Parquet (Andrew Lamb)
- Subsequent sessions in this series