Cask Data Application Platform (CDAP)

The Integrated Platform for Developers and Organizations to Build, Deploy, and Manage Data Applications

February 2015

Contents

Figures and Illustrations

Executive Summary

Though Apache Hadoop™ and related projects are driving big data innovation, a significant opportunity remains to realize the full potential business impact of open source big data solutions. To move beyond enterprise data warehouse offload and business intelligence use cases, developers must have the ability to deliver data applications—applications that combine real-time events with historical data—to deliver actionable, operational intelligence.

There are significant challenges to delivering data applications. The Hadoop stack is comprised of multiple technologies, each designed independently by different open source communities. Data is often tightly coupled to the programming paradigm, making reuse of data and applications difficult. Traditional concepts and tools for application life cycle management have not been included in the projects comprising Hadoop.

Cask Data, an open source big data software company, has delivered the flagship Cask Data Application Platform (CDAP) to address these challenges. CDAP provides an abstraction layer making it easier to manage multiple technology components. CDAP provides integration, offering developers higher-level capabilities built with Hadoop functional elements. Additionally, CDAP offers runtime services supporting the application lifecycle.

CDAP enables a new class of applications to drive greater business value, including those requiring real-time and batch processing. CDAP simplifies big data application development: more apps faster, with less dependence on Hadoop expertise. Finally, CDAP enables organizations to avoid compromising operational transparency and control of their big data apps by providing essential services such as security, logging, metrics, and lineage.

While the capabilities of CDAP apply to the broad range of Hadoop applications, CDAP is uniquely suited to address the challenges many organizations face with ETL (Extract, Transform, and Load) data

management. The unified real-time and batch capabilities of CDAP can transform both the development and operation of the next-generation of big data apps with customer intimacy, anomaly detection, and operationalized analytics.

Hadoop Background and Environment

Since Google published a research paper on MapReduce in 2004 and the Apache Hadoop open source project was created in 2005, the Hadoop ecosystem has emerged as the driving platform for innovation in big data applications. From its origins as a method for sifting through large amounts of web data for large Internet companies, Hadoop and its related open source projects are now in production at organizations in virtually all geographic regions and industry segments. Hadoop applications are used to help companies better understand and connect with their customers, to help organizations identify and protect themselves against threats, and to drive the optimization of complex systems. As the range of applications has diversified, Hadoop has evolved from a narrow application project. Companies like Cloudera, Hortonworks, and MapR have integrated and packaged the complex collection of open source projects into a big data operating system. Looking forward, there is a growing demand for additional applications and use cases, for a broader range of businesses and users. Customers would like to utilize the scale and cost advantages of Hadoop for traditional data management use cases such as ETL (extract, transform, and load), data warehousing, real-time analytics, and even transactions. More demand will arise for streaming analytics and event processing in real-time as Hadoop enables value out of the growing Internet of Things (IOT) ecosystem.

The evolution of Hadoop has exposed some significant challenges to organizations seeking to obtain the most value out of their data. Each of the projects (YARN, Hive, HBase, HDFS, Flume, Oozie, etc.) that comprise Hadoop include their own application programming interfaces (APIs) that are often low-level interfaces tightly coupled to the infrastructure components. While developers with skills in SQL, Java or

other programming languages could spend the time to learn these low-level details of Hadoop, it would be ideal if they didn't have to.

Once data is ingested into a Hadoop environment, it is very difficult for IT operations to understand who is accessing or modifying data, or who is deploying applications and consuming resources. As use cases expand, gaps are exposed in the existing Hadoop ecosystem. These gaps become a critical impediment to adoption, particularly for those applications requiring enterprise-grade security, real-time response or business-critical predictability. Finally, as usage of Hadoop within an organization grows, there is no framework for developers to reuse application logic and to expand the application base against their existing data.

All of these challenges exist if Hadoop is being used as an offline data analysis or visualization tool, but they become more acute if an organization is trying to drive action out of data, combining both real time event processing and insights from retrospective data. This new wave of applications, known as data applications, can create even more business value for organizations.

As we have seen with the evolution of other technologies, higher-level abstractions and interfaces emerge to accelerate the cycle of application innovation by developers. To enable data applications, a data application platform is needed. An ideal data application platform should:

- Abstract low-level Hadoop APIs to accelerate and simplify app development;
- Enable the reuse of application logic and higher utilization of data assets;
- Scale easily from a laptop to large clusters;
- Broker data accesses through a control plane layer to provide operational transparency and access control while avoiding any impact to throughput and performance;
- Bridge the gap between data science, app development, and production operations;

- Enable a new class of applications that would otherwise be impractical; and
- Enable rapid application development iterations to extract value from data.

Introducing Cask

Cask Data is the open source software company leading the effort to deliver a data application platform for Hadoop. Cask, formerly known as Continuuity, was founded in 2011 by a group of developers who had faced significant challenges deploying massive-scale Hadoop applications at companies such as Yahoo! and Facebook. They had a vision that creating an abstraction layer with higher-level capabilities could dramatically accelerate the development of applications, enable new levels of operational efficiency, and make a new range of applications possible. The company developed and hardened the platform over three years, and recently contributed the entire project to the open source community under the Apache 2 open source license. The mission of the company is to provide simple access to powerful technology, starting with the flagship product, the Cask Data Application Platform (CDAP).

Cask Data Application Platform Overview

The Cask Data Application Platform (CDAP) is an open source project for distributed data and applications and is available in a commercial subscription from Cask. CDAP is a layer of software running on top of Hadoop platforms such as the Cloudera Enterprise Data Hub or the Hortonworks Data Platform.

CDAP is comprised of three core elements:

- **Programs:** Standardized containers providing consistency for diverse processing paradigms
- **Datasets:** Libraries to build reusable data access patterns spanning multiple storage technologies
- **Runtime Services:** Services to provide support for development, debugging, deployment, and the management of applications

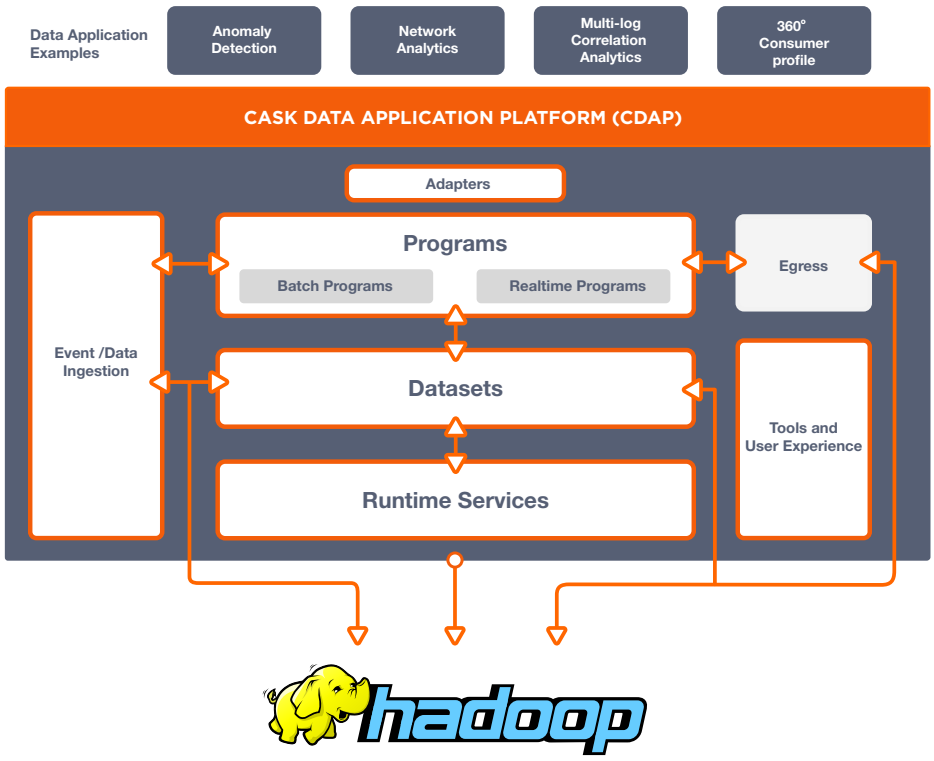

Figure 1—Functional Architecture of CDAP

The CDAP Software Development Kit (SDK) includes the developer APIs (Application Programming Interfaces) for creating both applications and accessing core CDAP services. CDAP defines and implements a collection of services and a set of SPIs (Service Provider Interfaces) that land applications and data on existing Hadoop infrastructure such as HBase, HDFS, YARN, MapReduce, Hive, and Spark. Cask will continue to expand platform support via additional services and SPIs, and the CDAP community will also expand platform support for both open source and proprietary software.

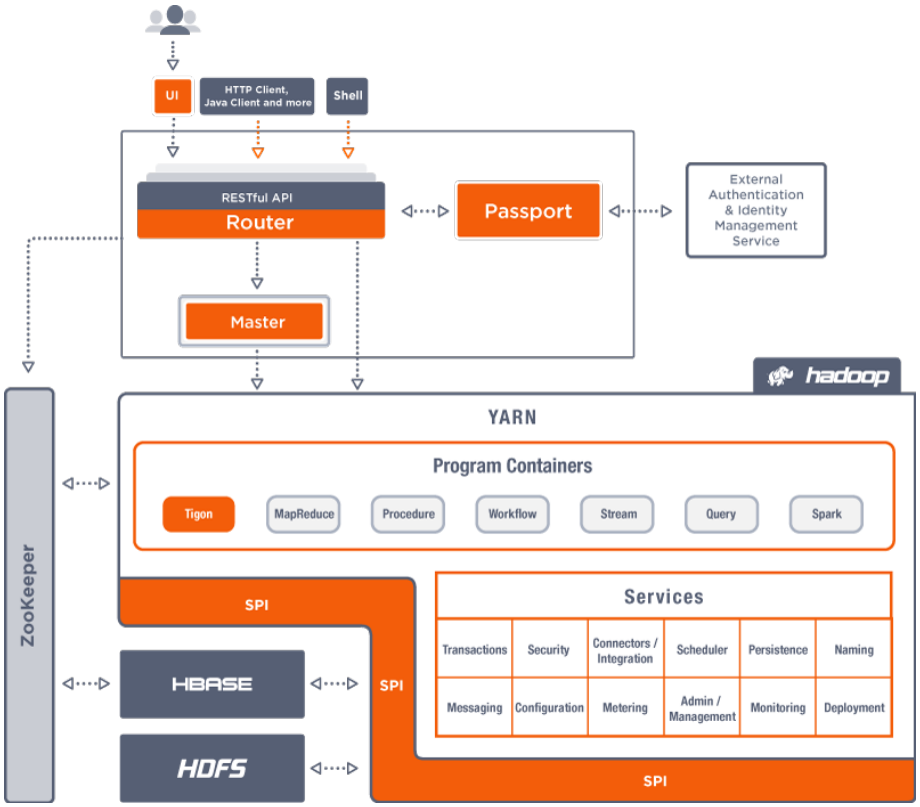

Figure 2—System Architecture view of CDAP

The CDAP platform is accessed via a router, which provides a highly available proxy for access to both applications and data. Data is ingested and queried through the router and applications are deployed and managed through the router. Data access, management, metrics, and security APIs are also exposed via HTTP RESTful APIs to easily integrate with other systems and users. The router provides named access, service discovery, and security for data and services that would otherwise be extremely complex to implement directly on a Hadoop environment.

The core services and SPIs are implemented directly against the underlying Hadoop infrastructure, with application containers operating natively within YARN and data operations performed directly against HDFS and HBase. This level of integration ensures there is virtually no added latency or performance impact from implementing on the CDAP abstraction layer. In addition, CDAP provides two other implementations of the platform: an in-memory mode for testing and a standalone mode for local development.

CDAP delivers value to customers three ways:

- For developers, the development framework, higher level APIs and integrated services dramatically simplify and accelerate data and application development;
- For organizations, the router and services provide data governance, metrics, security, and transparency enabling big data apps to be deployed and managed in production without compromising traditional enterprise application requirements; and
- For both individual developers and organizations, the ability to reuse data assets, provide applications as services, and drive a new class of applications using real-time streaming data delivers greater business value across a range of use cases.

The next step is to explore CDAP datasets and programs in detail.

CDAP Datasets

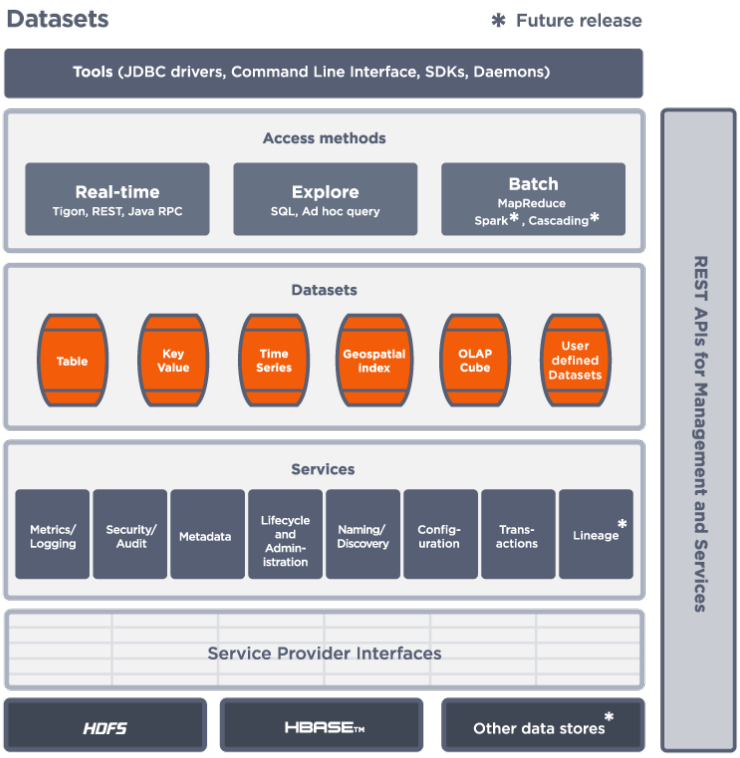

Figure 3—Functional Architecture of CDAP Data Abstraction

Datasets are one of the core elements by which CDAP enables developers to leverage data more effectively to build applications. Datasets are libraries used to abstract data spanning multiple storage technologies: today, using technologies such as HBase and HDFS. In the future, it will include data stored in other systems such as MongoDB and Apache Cassandra. The logical representations include a schema, domain-specific APIs, and different access pattern adapters that make the data accessible through multiple processing paradigms such as MapReduce, Spark, HIVE, real-time and Data RPC (Remote Procedure Calls). Data can be managed consistently across these different processing and storage paradigms. Schema can range from very simple byte arrays to more sophisticated data access patterns such as time-series or geospatial indexes. Because they are logical representations, datasets can incorporate multiple subordinate datasets as components

and include business logic. Datasets provide a simple mechanism for consistently updating multiple subordinate datasets using transactions. One example would be a dataset including all customer logins that could be used to derive another dataset providing a running total of customer logins. Applications or users seeking the total customer count would not have to build the logic to derive that total from the raw data.

The schema for datasets can be one of the commonly used data access patterns included with CDAP, or can be custom defined. These datasets are currently bundled with CDAP:

- **KeyValueTable** implements a key/value store as a table with a single column;
- **IndexedTable** implements a table with a secondary key using two embedded tables, one for the data and one for the secondary index;
- **TimeseriesTable** uses a table to store keyed data over time and allows querying of that data over ranges of time; and
- **ObjectStore** implements a table to read and write Java objects.

Cask has developed custom datasets that are available for use within CDAP, including an OLAP cube dataset and a geospatial index dataset. As the open source community around CDAP grows, more custom-defined datasets will become available covering a broader range of use cases.

CDAP Streams

A unique and essential type of dataset in CDAP is a stream. Streams are the method by which data is ingested into the platform. CDAP streams can be populated by existing methods for Hadoop data ingestion or streaming, such as Flume, Sqoop, or Kafka. However, CDAP streams can also be written to using many other technologies: HTTP RESTful APIs; the CDAP Command Line Interface (CLI); file-tailing daemons; and SDKs in Java, Ruby, and Python. This allows developers on other platforms to take advantage of Hadoop without worrying about Hadoop-specific ingestion technologies.

CDAP streams offer several capabilities unique to the platform.

- **Time stamping and ordering:** Without additional programming, streams ingested into a CDAP platform on Hadoop are automatically time-stamped and ordered predictably.
- **Horizontal scalability:** If the data flow for a stream outpaces the ability to write to disk, CDAP can initiate more stream writers for the same data flow. Despite the horizontal scaling, the stream itself is still represented logically as a single stream, and data from the multiple stream writers is aggregated, time-stamped, and ordered predictably.
- **Ability to process streams in multiple paradigms:** Because streams are a dataset, developers can run batch MapReduce, real-time streaming, and ad-hoc SQL queries directly on the stream.

Value of Datasets

- For developers, datasets offer rapid application development and enable new applications and services that would otherwise be impractical to implement without CDAP. The ability to take advantage of prebuilt services in the CDAP platform and the avoidance of programming to low-level data storage APIs are two elements that accelerate application development. Integrated ingestion tools, CDAP Streams, and the ability to expose useful data access patterns to other applications as HTTP RESTful APIs using datasets provides seamless data as a service and puts the focus back on application logic. The potential for processing directly on streams, both in real-time streaming and in batch, also expands the potential value derived from big data applications.
- Organizations benefit from the new application and service potential, but also gain additional value from CDAP. First and foremost is the added protection of a vital asset: data. CDAP provides data lineage, auditing, access controls, reporting, and other tools to ensure an organization's data assets are being utilized correctly. CDAP datasets offer investment protection in data—irrespective if an application is traditional ETL, real-time, batch; or whether applications are written in Spark, MapReduce or some other programming paradigm—data in CDAP datasets will remain consistent and highly-available.

CDAP Programs

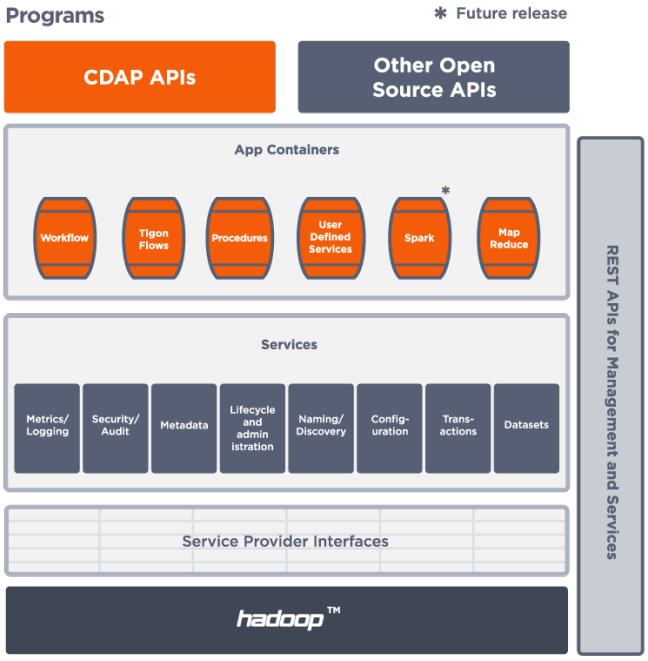

Figure 4—Functional Architecture of Application Abstraction

CDAP programs utilize the notion of containers to provide a standardized way to run and manage any type of application runtime.

It's important to distinguish CDAP application containers from Linux containers, such as those implemented by an application such as Docker. Linux containers exist on an individual node, and they can be used to run Hadoop software nodes such as a task worker or a data node.

CDAP containers are actually containers of distributed, horizontally scalable applications. Linux containers and virtual machine managers can be used to effectively run Hadoop infrastructure software, while CDAP containers are an effective means of implementing and running applications on Hadoop. CDAP containers provide not only distributed application packaging, isolation from the underlying Hadoop runtimes, and lifecycle management, but also provide shared runtime services such as log aggregation, metrics aggregation, service discovery, security and policies, and more.

There are different types of processing containers available within CDAP. The one most commonly used today is the batch processing container for MapReduce, but other processing frameworks—such as Apache Spark, Cascading and Pig—will be available in future releases. Running existing MapReduce jobs within a CDAP application container provides access to the shared services and tight integration with datasets. The capabilities injected in the containers removes a substantial burden from developers, allowing them to focus on business logic rather than operational requirements. Containers expose consistent, processing paradigm-agnostic HTTP RESTful APIs for managing the lifecycle of all application types. As most useful applications in big data comprise of more than just one MapReduce program, CDAP offers additional processing capabilities using Workflow containers—which allow MapReduce and other program types to be executed in a series, triggered by either an event or a schedule.

Another commonly used type of processing container is real-time stream processing. Stream processing in CDAP is implemented using Tigon Flows, comprised of one or more Flowlets that are wired together into a directed acyclic graph or DAG. Flowlets, the basic building blocks of a Flow, represent each individual processing node within a Flow. Flowlets consume data objects from their inputs and execute custom logic on each data object, allowing you to perform operations against Datasets as well as emit data objects to the Flowlet's outputs. The input to a Flow can be a CDAP Stream dataset; or a Flowlet can be used to either generate or pull the data from an external source (such as Twitter or Kafka, for example).

A crucial feature of CDAP that enables Applications in accessing Datasets is the CDAP transaction system, Tephra. Whether processing in batch or real-time, applications automatically execute data operations within transactions, providing framework-level guarantees that free developers from considering complex failure scenarios and infrastructure error handling. For example, a Flowlet processes the data objects received on its inputs one at a time. While processing a single input object, all operations—including the removal of the data from the input, reads and writes to Datasets, and emission of data to the outputs—are executed within a single transaction. This gives the Flowlet an exactly-

once processing guarantee and provides ACID (Atomicity, Consistency, Isolation, and Durability) properties:

- The process method runs under read isolation to ensure that it does not see dirty writes (uncommitted writes from concurrent processing) in any of its reads. It does, however, see its own writes.
- A failed attempt to process an input object leaves the data in a consistent state; it does not leave partial writes behind.
- All writes and emission of data are committed atomically; either all of them or none of them are visible.
- After processing completes successfully, all writes are persisted in a durable way.

Applications such as Flows have a tight integration with the CDAP transaction system, enabling the business logic within Flows to be non-idempotent. This provides a great deal of flexibility to developers in terms of the use cases that can be implemented in the system, drastically reduces the need for boilerplate code, and makes the move from development to production faster.

CDAP uses Optimistic Concurrency Control (OCC) to implement transactions. Unlike most relational databases that use locks to prevent conflicting operations between transactions, under OCC we allow these conflicting writes to happen. If two overlapping transactions modify the same row, then the transaction that commits first will succeed, but the transaction that commits last is rolled back due to a write conflict. In the CDAP system, MapReduce containers are also implemented as long transactions with many of the benefits of the transaction system, but they do not participate in conflict detection.

In addition to Flows, MapReduce, and existing Hadoop ecosystem open source technologies, CDAP includes additional application program types to make more advanced use cases such as “Data-as-a-Service” and “APIs-as-a-Service” simple. CDAP Procedures provide native access to datasets and allow you to build business logic over data and expose it as HTTP RESTful APIs.

CDAP User Services allow developers to build common discoverable services that can be easily accessed within different processing paradigms. A simple example is an IP-to-GeoLocation Lookup Service. It's a common service that can be used within MapReduce and Flows to join with the data being processed. Running them in a User Service application container makes it easy to register, discover and interact with the service from other application containers or from the external world.

Value of CDAP Programs

Utilizing CDAP programs offers developers significant advantages throughout the lifecycle of application development and deployment. In development mode, app containers all run within a single JVM using threads rather than processes, making it easier to debug, test locally, and utilize within a developer's Continuous Integration (CI) environment. Developers can have full confidence that the application can be deployed at scale in a distributed environment without any change to the application logic. As an application moves to production, the CDAP application runtime environment enables applications to be operationalized with low overhead for the developer. Organizations benefit from Programs because much of the complexity of Hadoop is handled by CDAP. The huge population of developers can quickly become Hadoop application developers, delivering value to their respective businesses. Finally, both organizations and individual developers benefit from the reuse, simplicity, and testability made possible by containers—developers accelerate their application development, and organizations increase the return-on-investment (ROI) of their spending on big data development.

CDAP Operations: Security and UI

The system architecture of CDAP includes a router proxy that brokers access to datasets and application containers and offers a significant advantage in implementing security on Hadoop. CDAP currently implements perimeter-level security through an authentication service named Passport. Passport supports delegation of authentication to an existing authentication backend—such as LDAP or Active Directory—through a pluggable authentication API supporting JAAS and JASPI

plugins. The CDAP Router integrates with Passport to authenticate all ingest and egress of data as well as management and lifecycle operations from external clients. Passport issues authentication tokens, which allow it to plugin to any existing enterprise identity management system. Passport insulates clients from requiring Kerberos credentials to work with the cluster.

The User Interface for CDAP provides administrators and operations with a complete view of the behaviors of applications running and data managed within CDAP. The starting overview shows which applications are currently installed and real-time graphs of all operations as the apps collect, process, store, and query data. Each statistic is per unit of time—events per second, bytes (or larger) per second, queries per second, based on a selectable time window. For each app, this information is displayed:

- **Collect:** the number of Streams consumed by the Application;
- **Process:** the number of Flows, MapReduce Jobs, and Workflows within the Application;
- **Store:** the number of Datasets used by the Application;
- **Query:** the number of Procedures in the Application; and
- **Busyness:** the percentage of time spent processing events by the Application.

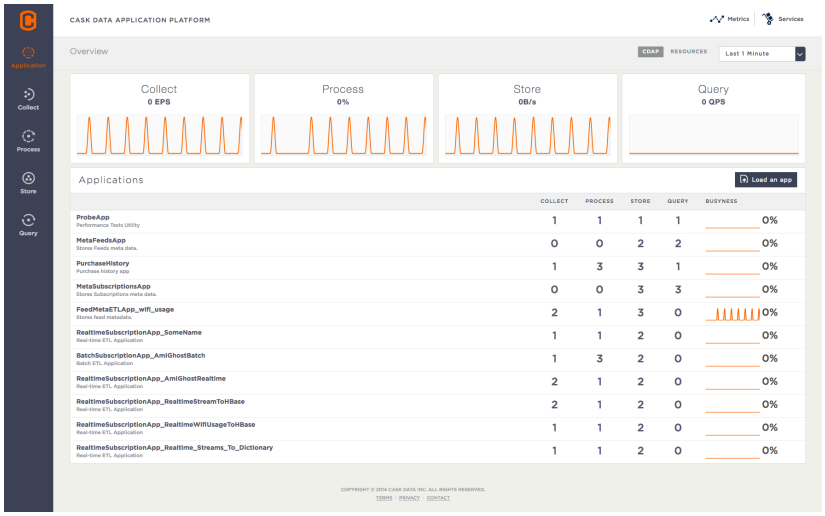

Figure 5—CDAP User Interface: Overview page

From this starting point, administrators can dive deeper into the status and behavior of CDAP applications and data through several useful lenses:

- **Resources:** Displays number of containers, number of cores, memory consumption (total bytes available and used memory), and storage (total bytes available and used disk space)
- **Metrics:** A customizable window where an admin can select which metric to graph over a specified period of time
- **Services:** A list of CDAP services with current status, horizontal scaling controls, and access to logging information where appropriate
- **Application:** Streams, Flows, Jobs, Workflows and Datasets associated with a particular application
- **Process:** A cross-application view of all Flows, MapReduce, and Workflows running
- **Store:** A cross-application view of all Datasets

The security capabilities and operational control and visibility offered in CDAP provide a bridge from the current status of Hadoop applications—which are often out of the operational path of business applications—to the future state where Hadoop applications can be part of an organization's core business operations.

Use Cases

The Cask Data Application Platform offers benefits across the full range of Hadoop use cases by enabling reuse of data and applications, and by providing operational controls difficult to implement without virtualization, a secured router, and supporting runtime services. There are two classes of use cases which are both common and critical and where the capabilities of CDAP are uniquely valuable. One specific class of applications is Extract Transform Load (ETL) apps, while a more general category are those that require a blend of batch and real-time analytics.

ETL (Extract, Transform, and Load)

The current class of ETL tools grew from a requirement to pull data from operational systems, where it would be invasive to perform analytics, into a data warehouse, where data could be made available to analysts and data scientists. The ETL function is quite complex as data sources and systems proliferate, and the need to integrate more types of data for greater insights grows. Some in the Hadoop community have postulated that ETL is no longer necessary as organizations can simply ingest all of their data into a Hadoop data lake and provide schema-on-read access to analysts. However, this argument unfairly discounts two of the core value propositions of traditional ETL tools. First, ETL tools provide data governance in the form of both access controls and data lineage. Second, ETL tools provide visual interfaces to reduce or eliminate the complexity of managing data. In the scenario where customers utilize Hadoop without ETL tools, both of these remain gaps.

Still, the promise of Hadoop replacing expensive and proprietary ETL tools remains; and CDAP makes great strides to close the gaps in both data governance and ease of use. CDAP provides security that integrates with existing authentication systems, and provides data lineage since all external data accesses are monitored through the router and internal data accesses are logged with context from the application container. Streams provide a turnkey method for ingestion, and datasets provide external systems easy access to data in access patterns that fit the application.

Absent CDAP, access to data in either the Hadoop Distributed File System (HDFS) or HBase requires connections to both the metadata servers and all data nodes, since data accesses in Hadoop are direct connections to the data nodes. This requires a relatively thick client and the ability to tolerate failures. With CDAP, external clients need only connect to the router, which then accesses the underlying Hadoop system. Finally, Cask has built and bundled an ETL application with CDAP to expose all of these capabilities to data integration developers, providing the ease of use customers desire. In a future release of CDAP, due in the second quarter of 2015, Cask expects to deliver base ETL capability on Hadoop without the need for developers to write a single line of code.

Real-time and Batch Processing in a Single Platform

The origins of Hadoop and the most common applications for Hadoop today are for batch processing, where response times aren't critical to the value of the application. Often, the results of Hadoop analytics are used to create new rules or business logic that is then deployed in a real-time streaming or event processing system that can handle data flows and provide responses in the time dictated by the business requirements. The real-time systems tend to have limited access to historical data, and the batch analytics systems can't be used for real-time decision support.

CDAP offers developers the opportunity to utilize different frameworks and systems to optimize your application for the response time and volume of data required, all in a single system. On one end of the spectrum is extreme real-time processing. Cask and AT&T Labs have collaborated to integrate AT&T's real-time complex event processing (CEP) engine used for line-speed packet processing with the CDAP transactional Stream processing engine. This effort has been contributed as the Apache-licensed open source project Tigon, which became available in a developer's release in Q3 2014. A production version of Tigon will be bundled with CDAP in a future release. Tigon provides CDAP with a very low-latency stream processing capability suitable for the most demanding real-time applications. At the other end of the spectrum is traditional batch processing, which CDAP also supports. In between, CDAP datasets offer developers the ability to analyze streams

or any subsets of data to support real-time response ranging from milliseconds to minutes, depending on the need. Standardized application containers make deployment and management consistent and simple.

This blend of real-time and batch processing has the potential to transform the next generation of big data apps in several domains:

- **Customer intimacy:** CDAP allows you to rapidly create targeting and user profile applications to drive website engagement and revenue. This includes providing recommendations on a real-time basis from predictive analytics for cross-promotion, delivering highly targeted ads for contextual experiences and ad forecasting applications, and creating large scale click stream analysis to drive customer account expansion
- **Anomaly detection:** CDAP enables development of applications to detect patterns and act on trends and anomalies over large streams of social, user-generated and machine-generated data. This could be used to detect and act upon fraud in financial transactions; breaches in IT systems; threats to public safety; and even medical anomalies, providing better and faster diagnosis for improved therapeutic outcomes.
- **Operationalized analytics:** Hadoop is already great for performing large-scale, ad-hoc, exploratory analysis of big data, i.e. asking new questions of old data. However, once the right questions have been asked, you may want to continuously know the answer as things happen: asking old questions of new data. This could be applied to predictive maintenance, supply chain management, and the optimization of complex systems such as manufacturing lines, computer networks, and transportation systems.

It is difficult to overestimate the economic value created by analytics applications that can blend real-time streaming with batch processing on historical data. A robust data application platform for Hadoop is a critical element to enabling these applications, and CDAP provides an open source foundation for the Hadoop developer ecosystem.

Engagement Process

The Cask Data Application Platform is freely available from Cask Data, and downloadable at http://cask.co/downloads. Java developers with less experience on Hadoop can download the SDK and sample applications and begin to modify applications to fit their needs. Developers with existing MapReduce code can easily port their applications to CDAP. Developers can build applications from the ground-up very rapidly by taking advantage of the rich services and simple APIs of CDAP.

For those developers interested in joining the CDAP open source community, to make contributions or provide input on requirements, the open source community site is http://cdap.io. Developers can join the mailing list and participate in the evolution of CDAP. The first step in the process of building the community was for Cask to contribute the project to open source under an Apache license. Moving forward, CDAP will become more community-driven and ultimately result in an open data application platform.

Many organizations planning to deploy CDAP in production will want a fully supported commercial implementation. CDAP commercial offerings are tailored from those getting started through to organizations putting applications into production. Contact Cask for details on pricing and subscription options.

Download CDAP
http://cask.co/downloads

