FIS Modern Banking Platform

CAPE RAS

Solution Architecture Document – CTE1 And CTE2

CAPE RAS Release 2.3

Release 3.11 GA

Published June 2024

Contents

About This Document

This document describes the CAPE RAS reporting architecture framework and its features. This document supports CAPE RAS v2.3.

A migration script is available in the following artifact link: https://artifactory.fis.dev/artifactory/epo-maven-release/mbp/caperas/ras-cds-Hive-Mig.zip.

Audience

This guide is written for users of the CAPE Reporting and Analysis Solution product.

How this Guide is Organized

This document is comprised of the following topics:

Topic	Description
Project Definition	Provides features and benefits of the CAPE RAS Solution Architecture.
Solution Architecture	Provides an overview of the Hadoop-detailed architecture with information on source systems, data acquisition, and the CAPE RAS user interface.
Security and Compliance	Provides an overview of the security framework.
Solution Approach and Strategy	Provides the data integration strategy for the solution architecture with source system analysis.
Non-Functional Components	Provides database-related maintenance required to keep data within Hive.
CTE1 vs CTE2	Provides the data flow difference between CTE1 and CTE2
References	Provides reference information for the CAPE RAS Data Dictionary and the CAPE RAS Data Model.

Conventions Used in This Guide

This guide may refer to aggregates and elements in service messages. The hierarchical path to an aggregate or element is denoted using the > symbol.

Assumptions

This guide assumes that all examples provided in this document are directory non-specific and in no way reflect live data from a database.

More Information

For additional details concerning FIS Modern Banking Platform functionality, refer to the *FIS Modern Banking Platform Reference* online Help system. This Help system will continue to be enhanced over time to describe all available functionality.

The complete FIS Modern Banking Platform documentation set is available at **FIS Client Portal**:

https://my.fisglobal.com/ > Modern Banking Platform product page

Project Definition

The CAPE RAS Solution Architecture provides a reporting and analysis solution for Modern Banking Platform.

- Hadoop-based data repository for Modern Banking Platform
- Batch and intraday processing of data acquisition, processing, and reporting
- Real-time streaming data acquisition, processing, and reporting
- Reporting framework for operational reports
- Reporting framework for MIS (Management Information System) reports
- Reporting framework for reconciliation reports
- Metadata driven reconciliation framework
- Metadata driven data processing platform (DPP)
- DPP in UI
- Ability to consume real-time events from BLOB storage for CTE2
- Framework for system reports (e.g., ETL Audit reports, Application Audit reports)
- Data Visualizer Tool - Dashboard Designer and Dashboard Viewer
- Change analytics
- Data provisioning for Compliance Suite extracts to downstream systems
- Foundation for data aggregation to downstream systems (e.g., statements and notices)
- Self-servicing (ad hoc) reporting capabilities (e.g., I-Gen / E-Gen and dashboard)
- Metadata management console
- Data lineage for self-reporting tool (I-Gen)
- Robust and simple metadata framework ensuring smooth integration of all ETL (extract, transform, and load) and Reporting services within a user interface
- Metadata-driven ETL framework featuring LowCode using a single master script
- Template-based metadata driven reporting framework
- Metadata-driven batch scheduler (via UI and backend)
- Configuration and user management
- Security framework (Encryption and Decryption, Data Masking, IdP (Single sign in), Apache Kerberos, signature validation of token)
- Auditing features —User audit log, report audit log, streaming data audit log
- Scheduling strategy
- High availability and disaster recovery strategy
- Deployment Strategy
    - Single click automated deployment
    - Stackable deployment
    - Rollback/backout strategy
    - OpenShift deployment
- Cloud implementation via Azure

The CAPE RAS Solution Architecture integrates the following source systems as part of the CAPE RAS:

- Retail Deposits

- Commercial Deposits
- Retail Lending
- Enterprise Customer
- Enterprise Organization
- Enterprise Banking Collateral
- EFT

Additional key features include:

- Multi-Tenancy
- Multi-Currency
- Multi-Lingual
- Locale based date, currency and decimal formats.
- Real-time integration
- Extensible/Configurable Datawarehouse Framework

Solution Architecture

Context Diagram

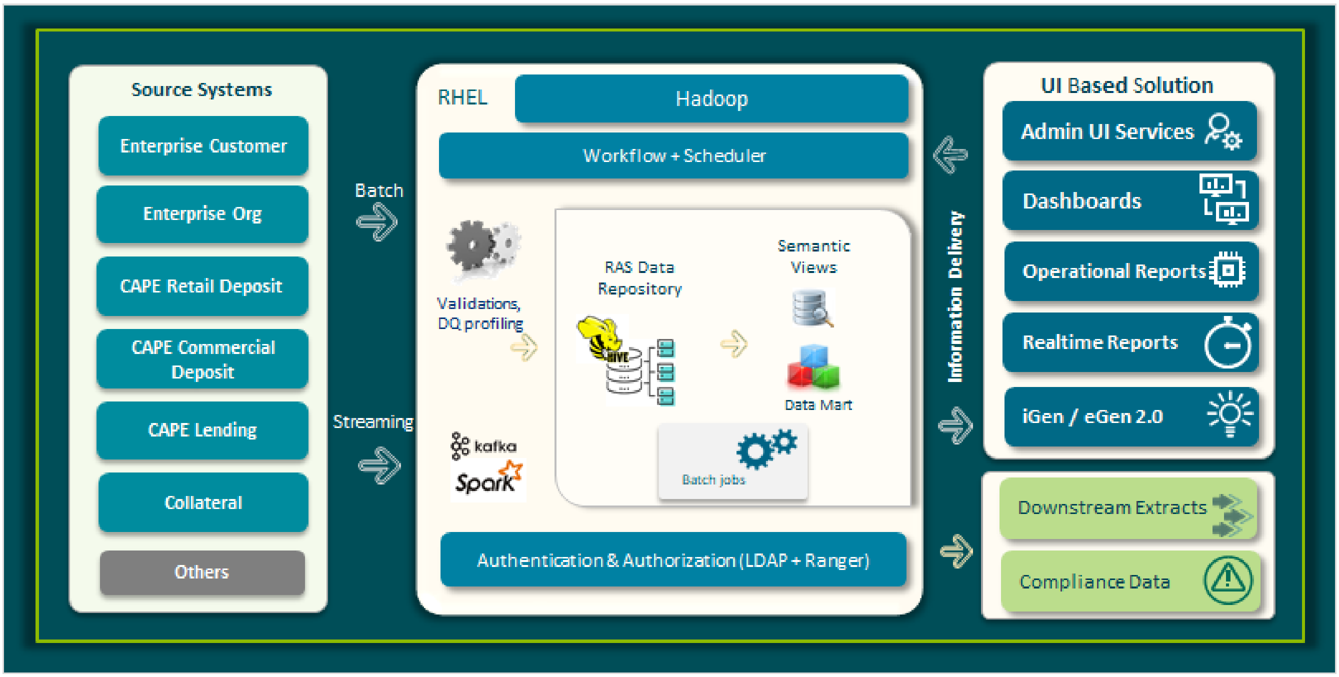


Figure 1. Solution Architecture Context Diagram

Hadoop Detailed Architecture

Overview

Apache Hadoop is a big data framework of the Apache Software Foundation that is extensively used for distributed storage and processing of data on a level that is enormous in terms of volume, variety, and velocity. The Hadoop platform can be used for data storage, processing, access, analysis, governance, security, operations, and deployment.

Hadoop operates on a cluster of nodes that involves huge amounts of data and hence the failure of a node is a high probability. The Hadoop platform is resilient in the sense that the Hadoop distributed file systems immediately upon sensing node failure divert the data among other nodes thus allowing the whole platform to operate without any interruptions.

Source Systems

SL No.	Source System	Source Data	Process Type
1	Retail Deposits	TDL Tables	Batch
			Intraday
		Servicing API	Real-time Streaming
2	Commercial Deposits	TDL Tables	Batch
			Intraday
		Servicing API	Real-time Streaming
3	Enterprise Customer	Flat File	Files sent to the FTP server in a secured manner
			Files retrieved from the FTP server into the Data Acquisition layer
			Batch
		Servicing API	Real-time Streaming
4	Enterprise Organization	Flat File	Files sent to FTP server in a secured manner
			Files retrieved from the FTP server into the Data Acquisition layer through
			Batch
5	Retail Lending	TDL Tables	Batch
			Intraday
		Servicing API	Real-time Streaming

SL No.	Source System	Source Data	Process Type
6	Collateral	Servicing API	Real-time Streaming
7	EFT	Flat File	Batch
8	ITC (CTE2 only)	TDL Tables	Batch
		Flat File	Batch

The source data type, format and database information are available in the metadata tables for processing. The metadata section provides detailed information about the Source data processing.

Supported Source Data Formats

- Flat Files
- Tables
- JSON's

Data Acquisition

The system retrieves the content of various data sources and stores it in the Hadoop platform within a scalable storage solution, such as the Hadoop Distributed File System (HDFS). The stored data is subsequently processed and stored in the semantic layer and serves as a source for the presentation layer.

Mode of Data Acquisition

- Batch
- Intraday
- Real Time Streaming

Batch Processing

Batch processing processes blocks of data that have been stored over a period. Modern Banking Platform processes the batch every night in a scheduled manner. The data is retrieved and processed using the Spark, Python, and SparkQL in the HDFS (Hadoop Distributed File System).

The Batch Date drives the batch process. The Batch Date is a process date and processing group that is passed as an argument (Sysdate-1) to the scheduler.

The source data for batch process includes flat files and the database. The batch process extracts this data from the source system, performs data validation and data profiling, and stores the data in HIVE tables.

Note:

Refer to the Data Processing section for details regarding data validation and data profiling. For international implementations, CAPE RAS consumes data via extract files and TDL from ITC source.

Intraday Processing

The Intraday Batch retrieves and processes data on a scheduled time interval, using Spark, Python, and SparkQL in the HDFS.

The Batch Timestamp drives the intraday batch process. The Batch Timestamp is a date timestamp that is passed as an argument along with the processing group to the scheduler.

The source data includes flat files and the database. The intraday batch process extracts this data from the source system, performs data validation and data profiling, and stores the data in HIVE tables.

Note:

Refer to the Data Processing section for details regarding data validation and data profiling.

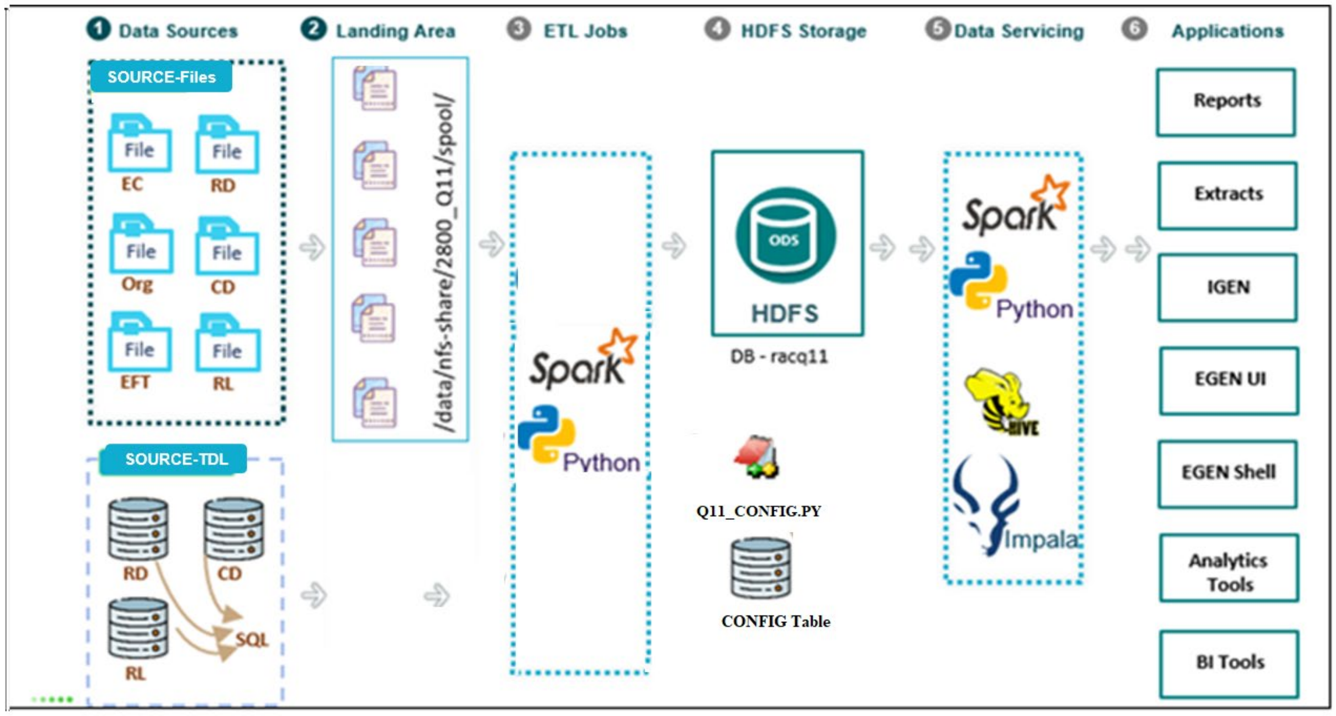


Figure 2. Data Processing

Real-Time Processing

Real-time data streaming uses data while it is motion through the server, processing data swiftly while simultaneously reacting to changing conditions in real time.

The real-time streaming process is achieved through a combination of Apache Kafka and Structured Spark Streaming. Structured Spark Streaming consumes the data and the data is adopted in the Events CDC inbox. The event payload JSON is transformed and stored in real-time (RT) tables in Hive.

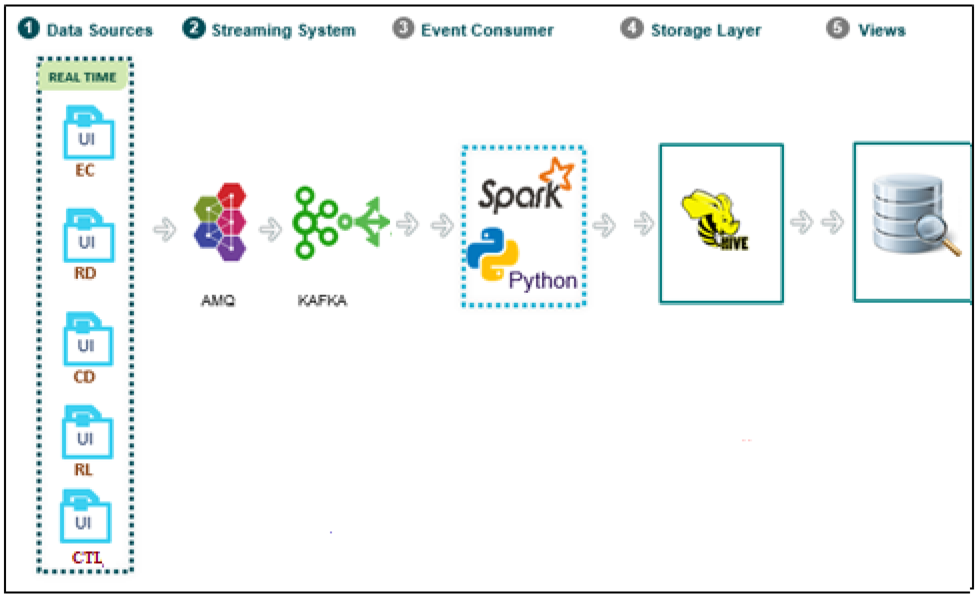


Figure 3. Real-time Process Flow

Real-time Process Flow

- Event broker processes transaction data from the core processors and the event service is enabled
- Kafka consumer process listens to the subscribed topic, consumes the events through Spark Structured Streaming, and stores the content in each component's Event Inbox table
- Data in the event payload is transformed and persisted in Hive tables based on the event tag/Hive column configurations
- Event data is stored in Real Time ODS (operational data store) tables
- Static data is stored in the parent table
- Repeating key-value pair is stored in child tables in Hive

Note:

Real-time data warehousing capability is implemented in Transaction History.

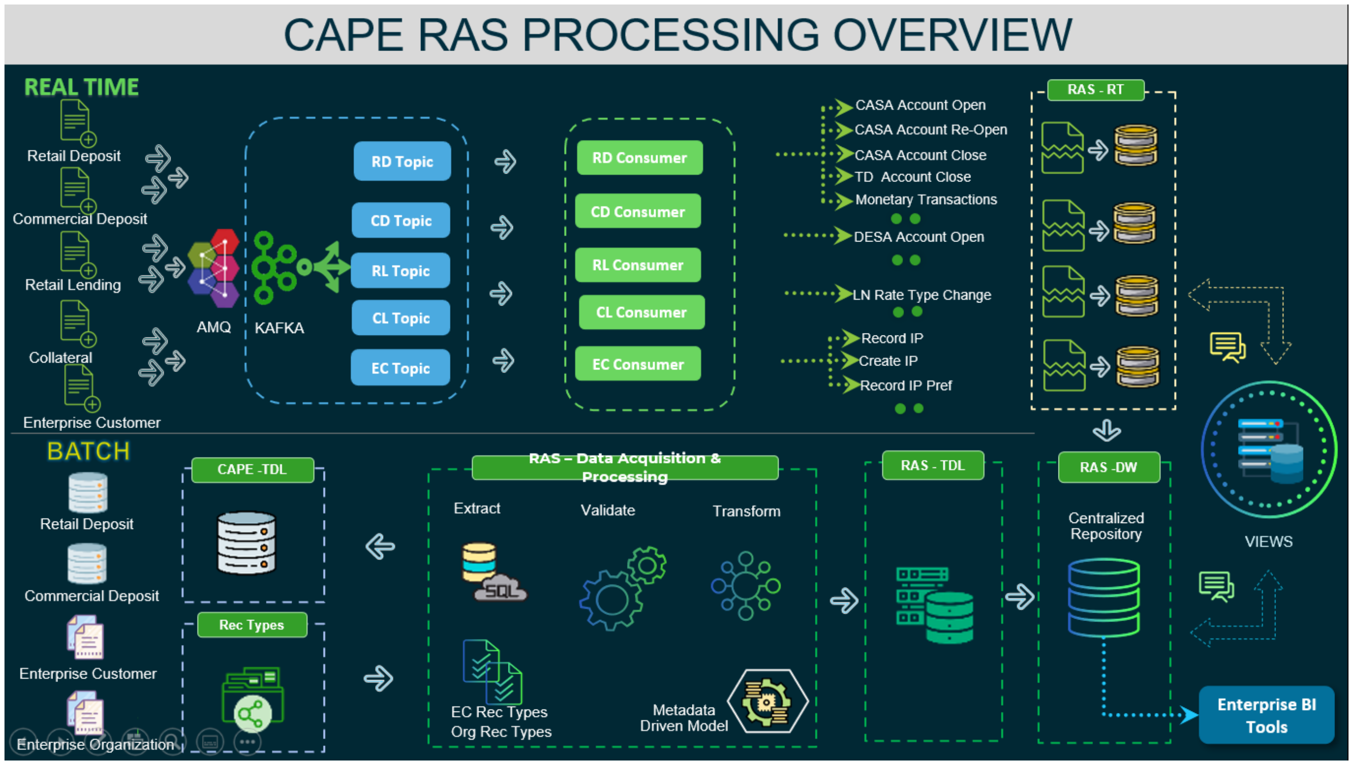


Figure 4. CAPE RAS Processing Overview

CAPE RAS and Core Reconciliation

Reconciliation reports maintain data consistency and integrity between source components and CAPE RAS. Monetary transactions are received through real-time events and persisted in the CAPE RAS database. Account balances persisted in CAPE RAS at the end of day are reconciled with Modern Banking Platform. Metadata driven reconciliation framework in CAPE RAS makes it easy to build new reconciliation jobs with outcomes of either report or hard stop.

CAPE RAS Reconciliations

- Posting Data and Account Balance
- GL Transaction and Monetary Transaction
- CAPE RAS SHIP Reconciliation

Note:

The *CAPE RAS Reconciliation Reports* document and the *CAPE RAS Self-Healing Intelligent Program (SHIP) Operations Guide* provide processing and detailed steps for each reconciliation.

Posting Data and Account Balance Reconciliation

CAPE RAS receives all posting transaction data through Posting TDL and Real-time Events.

The Monetary Transaction Event publishes sub-type events based on the posting identifier.

- Real-time table contains granular posting data
- History Posting tables contain all historical posting data in a denormalized tables structure
- The Transaction and Balances Reconciliation report reconciles the aggregated transactions based on credit/debit count, sum of credit/debit amount, and closing balances for each transacted account for the day between the core system and CAPE RAS database.

GL Transaction and Monetary Transaction (MTRECON)

Real-time transactions are published from the source and posted in the PSTG table.

Real-time events consumed by CAPE RAS are reconciled with the PSTG table.

CAPE RAS SHIP Reconciliation

CAPE RAS SHIP (Self-Healing Intelligent Programs) maintains data consistency and integrity between source components and CAPE RAS. These programs are built on the base functionality of reconcile, retrieve, and reload. The SHIP module is available via the UI in DPP.

In addition to reconciling data, SHIPs also retrieve missed data and reload it in target tables.

CAPE RAS SHIP performs the following reconciliations:

- SHIP1 – Events reconciliation, retrieve, and reload in batch
- SHIP2 – Previous Day GL Tran Reconciliation, Retrieve, and Load in Current Batch (Retail Deposits)

Data Processing Layer

The Data processing layer enables the execution of validation parameters and the storage of clean data in the data repository.

Data Processing Layer	
Data Validation	Source data undergoes data quality checks and data enrichments based on defined business rules. The detailed process is outlined in the DQ (data quality) Framework sections.
Data Profiling	Source data is reviewed and structure-analyzed, and potentials are identified before loading into the data repository.
Data Transformation	The Source-to-Target Mapping document is prepared and business rules are defined. The transformation functionalities include: • Data Repository transforms the source data based on the defined business rules with the same granularity as the source. • Derivation logic is applied to load columns required for reports or downstream extracts. • Columns to support the row lineage are created. • Data is denormalized for additional dimension attributes from the customer, account, product, location, and date dimension tables.
Data Aggregation	Aggregated data is stored in data marts to support requirements like dashboards, reports, and downstream extract.
Data Lineage	Allows the user to understand and visualize data as it flows from CAPE RAS source systems to reports generated via I-Gen.
Data Dictionary	Allows the user to view field information, such as name, description, relationships to other data, source, and format of all CAPE RAS data warehouse tables, from the CAPE RAS UI; this information can be used when generating reports.
Metadata Management	Specific technical metadata tables are populated as part of this process.
Exception and Error Handling	All errors and exceptions thrown are captured and stored during data processing. Error records are rejected, exceptions are loaded to the CAPE RAS exception tables, and data issues are captured as DQ exceptions.
Auditing	Data processing framework that captures the execution status and statistics of the entire process and maintains an end-to-end audit log.
Capabilities	CAPE RAS supports the following key functionalities: • Multi-Tenancy – CAPE RAS can host data from multiple banks in the same cluster with each bank on its own database. File folders are segregated based on the bank number/financial entity ID maintained in the metadata framework. CAPE RAS Consumer is implemented to support the multi-tenancy.

Data Processing Layer	
	• Multi-Currency – CAPE RAS supports multiple currencies and follows a structure similar to CAPE Core. • Multi-Lingual – CAPE RAS supports multiple languages. Based on the bank/FE (financial entity), the conversion team deploys packages to support multi-lingual capabilities.

Semantic Layer

A semantic layer is a business representation of core application data that helps end users access data autonomously using common business terms. The semantic layer maps complex data into familiar business terms such as product, customer, or account to offer a unified, consolidated view of data across the organization.

Semantic in the context of data and data application means “from the user’s perspective”. In real terms, it can be defined as abstracting the raw data and giving a semantic meaning for the business user. An abstraction layer factors complexity down to the level where each of the end business users need not replicate the logic to arrive at the desired output. The sematic layer is a force multiplier for data consumers and guarantees correct results, database performance, enhanced user performance and acceptance.

CAPE RAS enables FE-specific semantic layers with the following abstraction strategy:

- A metadata-driven approach generates semantic abstraction as per the individual bank’s requirement.
- The underlying physical structure is translated into recognizable business names and organized into a format that seems logical to the front-line business person through the centralized metadata repository.
- During tenant setup, update the friendly names to the business metadata tables so that same names appear in all the reports and for all bank/branch users.

For example, a semantic layer of loan data makes it easier for the Lending team to ask ad hoc questions. A financial entity could model separate abstractions specific to deposits data, channels data, accounting data as per user requirements.

A major advantage of this metadata-driven approach is that it bypasses the requirement for redeveloping the entire abstraction process for every financial entity.

CAPE RAS User Interface

The CAPE RAS user interface (UI) is built using the Responsive UI framework which internally uses the BOD framework.

The UI is comprised of the following components:

- Reports interface
- I-Gen
- E-Gen
- Data definition
- Metadata management interface
- Data masking
- Data Visualizer tool – Dashboard Builder and Dashboard Viewer
- Download extracts modules

- DDP in UI
- Data governance

The basic flow is provided from the end user perspective.

Users can access the CAPE RAS data repository from the CAPE RAS UI, batch mode, or through third-party BI (buisness information dashboards) or analytical tools.

The evaluation of Admin UI and Servicing UI along with its accessibility and usage is explained in detail in the *CAPE RAS User Guide.*

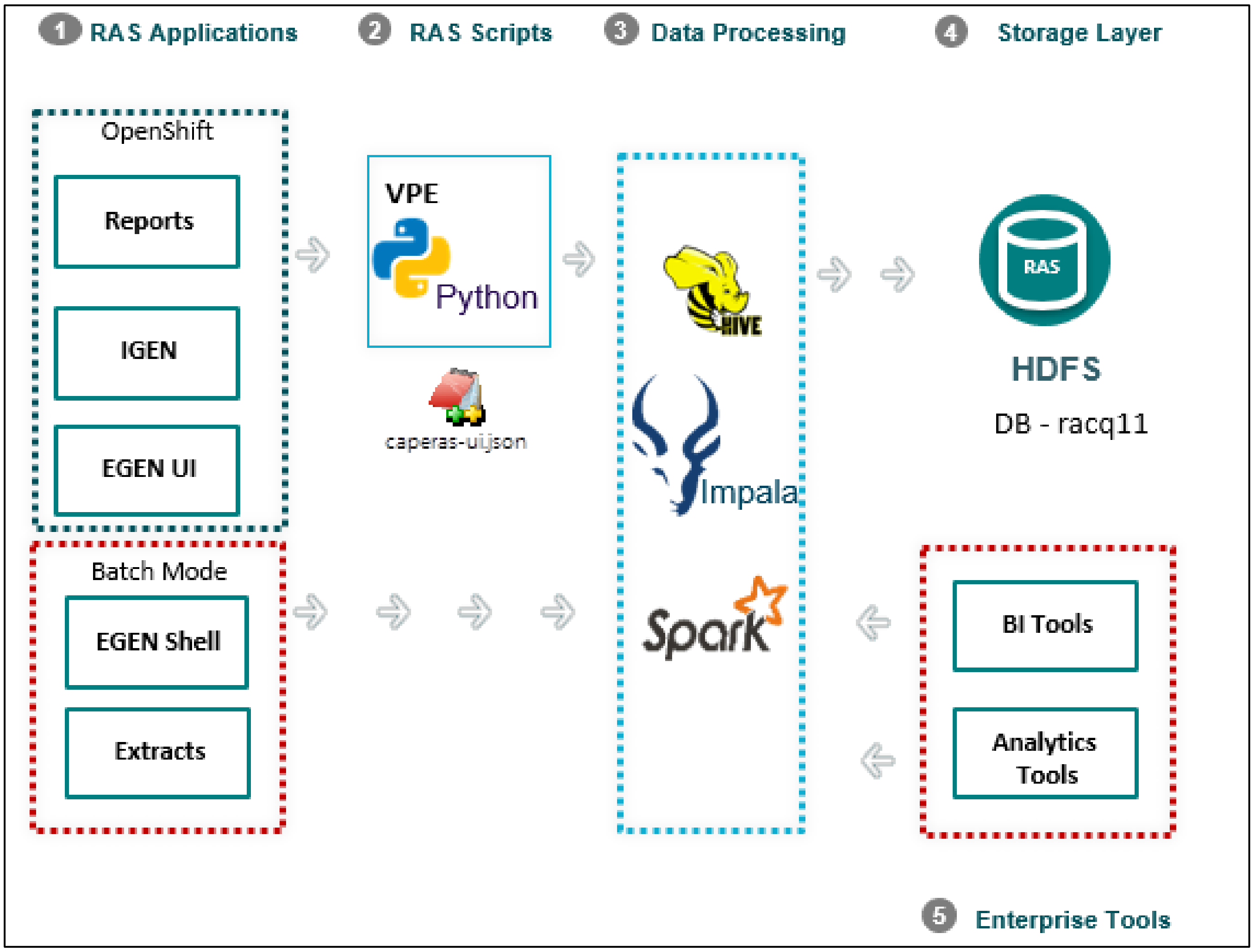


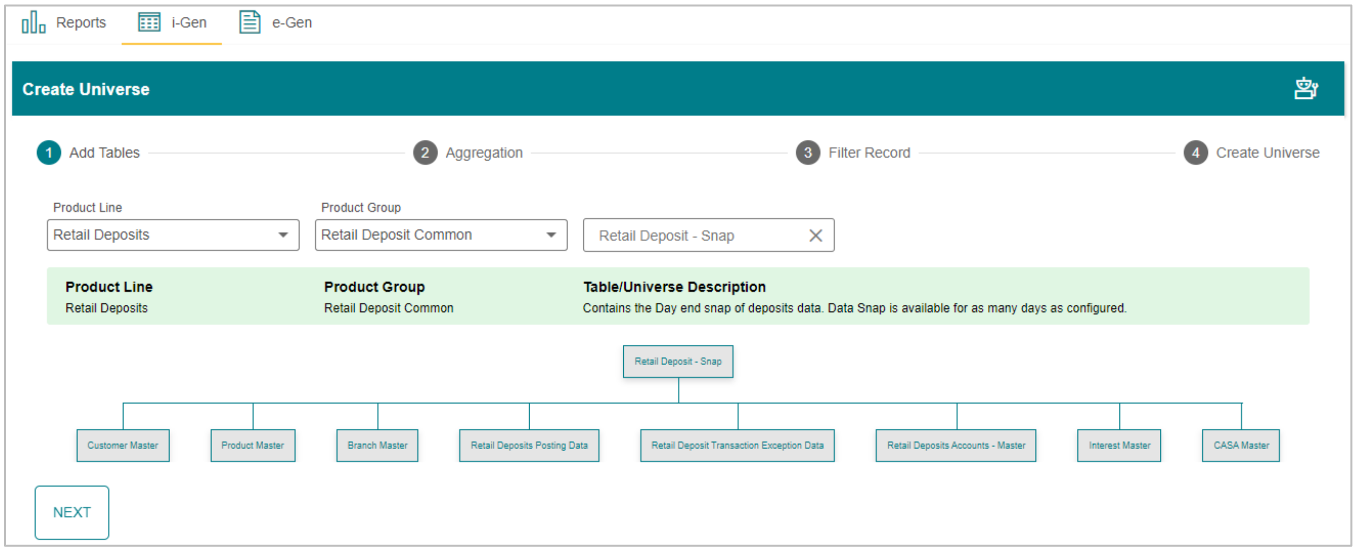
Figure 5. CASE RAS User Interface Architecture

I-Gen – Report Builder

I-Gen is a Python-based self-servicing business analysis and ad hoc report building feature of CAPE RAS that empowers non-technical business users to visualize, understand, analyze specific business questions, and quickly build ad hoc reports. I-Gen provides flexibility in functionality, access to multiple source data, ease of use, highly accessible/understandable semantic data, reusable data components like Universal data sets, and agility to create insightful information.

I-Gen Report Extract Features

- Universe is a virtual semantic layer or business representation of banking data that helps users access data autonomously using common business terms.
- Universe represents a collection of data elements to define a context in business, such as Loans, Deposits, and Customer. I-Gen enables users to create a universe as per their business requirements.
- Filter Condition enables users to filter data from a predefined universe or business tables to analyze data or create ad hoc reports. I-Gen offers filter conditions that include:
  - Less Than operator <
  - Greater than operator >
  - Not equal to <>
  - Equal to =
  - AND
  - OR
  - NOT
  - LIKE
- Aggregate Condition enables users to aggregate data on specific columns that are grouped together to create summary information or the report. I-Gen offers aggregate conditions that include:
  - MIN
  - MAX
  - COUNT
  - SUM
  - AVG.
- Ad hoc reports or datasets can be downloaded and stored.



Product Line Retail Deposits
Product Group Retail Deposit Common
Table/Universe Description Contains the Day end snap of deposits data. Data Snap is available for as many days as configured.

Retail Deposit - Snap
Customer Master
Product Master
Branch Master
Retail Deposits Posting Data
Retail Deposit Transaction Exception Data
Retail Deposits Accounts - Master
Interest Master
CASA Master

Figure 7. I-Gen Report Builder

I-Gen Report Builder Features

- Create templates from the list of tables along with required columns.
- Save or delete a user-defined universe
- Save, publish, and schedule reports generated via I-Gen
- Build ad hoc reports from the predefined universe with filter and aggregate conditions
- Provide one or more filter conditions to specific columns(e.g., <, >, <>, =, AND, OR, NOT, LIKE)
- Provide an aggregation factor to specific columns (e.g., MIN,MAX, COUNT, SUM, AVG)
- Download and store reports generated on ad hoc basis
- Visualize the flow of data added in the reports generated via I-Gen
- Use report formats such as Grid view, MultiTab, MultiGroup, and MultiHeader
- Download the configuration of all universe/reports into a single zipped file, like Flyway DB Loader.
- Option to clone I-Gen reports
- Create Process Master metadata entry to schedule a report

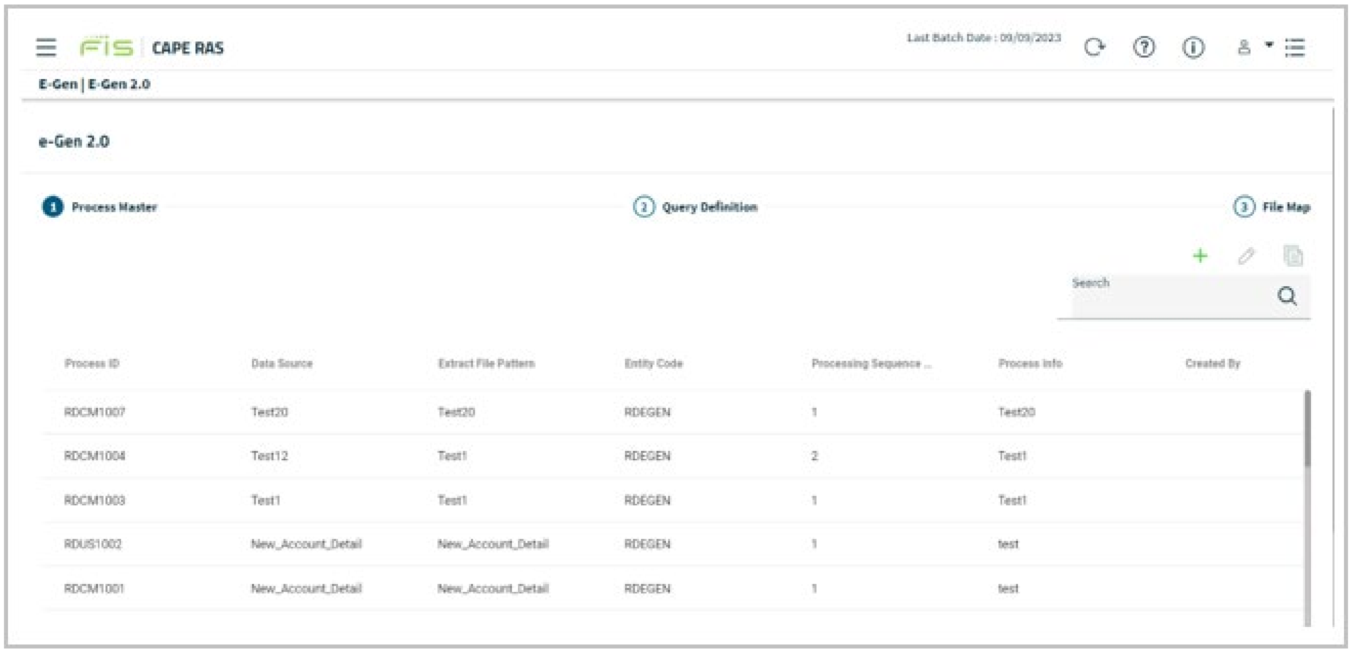
IGen Query Builder

I-Gen is a value-added feature within I-Gen that enables users to create or edit a universe via SQL Query using Query Builder. This skips the stepper process, saving time and effort.

E-Gen Report Extract

E-Gen is an inherent extract building capability to generate user-defined extracts.

Data extraction is the act or process of retrieving data out of data sources for further data processing or data storage. The import into the intermediate extracting system is usually followed by data transformation and the addition of metadata prior to export to another stage in the data workflow.



Process ID	Data Source	Extract File Pattern	Entity Code	Processing Sequence ...	Process Info	Created By
RDCM1007	Test20	Test20	RDEGEN	1	Test20	
RDCM1004	Test12	Test1	RDEGEN	2	Test1	
RDCM1003	Test1	Test1	RDEGEN	1	Test1	
RDUS1002	New_Account_Detail	New_Account_Detail	RDEGEN	1	test	
RDCM1001	New_Account_Detail	New_Account_Detail	RDEGEN	1	test	

Figure 2.<E-Gen Extract List>

E-Gen Report Extract Features

- Create ad hoc extracts by selecting the required columns and applying needed filters
- Download extracts in .txt format
- Configure custom extracts using rules and query parameters .
- Merge extracts

E-Gen 2.0 Query Builder

E-Gen 2.0 is a value-added feature within E-Gen that enables users to define an extract. The user can generate the extract output directly by providing the SQL query instead of going through the multi-step E-Gen process. The user can define the extract file as either fixed width or delimited, and can define the delimiter in metadata. The field width, arrangement of columns, and sorting records for the extract are configurable.

UI Level Authentication

- User-level security (authentication of users) validates the identity of a user through the IdP process. IdP manages the roles and the user association with the roles.
- Object-level security controls the visibility to business logical objects based on a user's role. This is set up in object-level security for metadata repository objects, such as business models and subject areas, and for Web objects, such as dashboards and dashboard pages, which are defined in the Presentation Catalog. For example, users in a particular department can view only the subject areas that belong to their department.
- Data-level security controls the visibility of data (content rendered in subject areas and dashboards) based on the user's association to data.
- Data masking masks any preconfigured columns based on user role/IdP authorization in the UI for view only and PDF downloads. It covers:
    - Standard extracts
    - Custom extracts generated via E-Gen
    - Custom/operational/standard reports (only account number masking is enabled)

Note:

Based on user role, an authorized user can toggle between mask/unmask.

IDP/SSO Integration

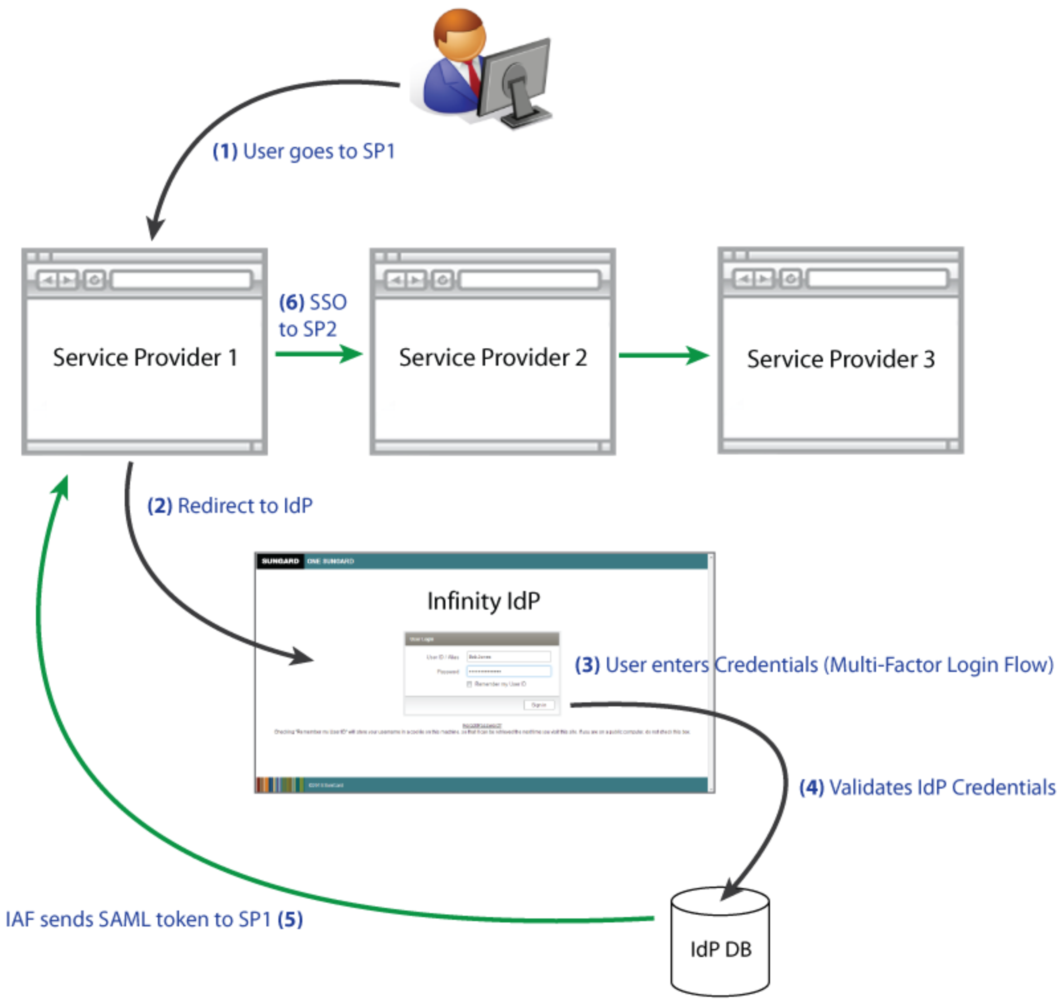
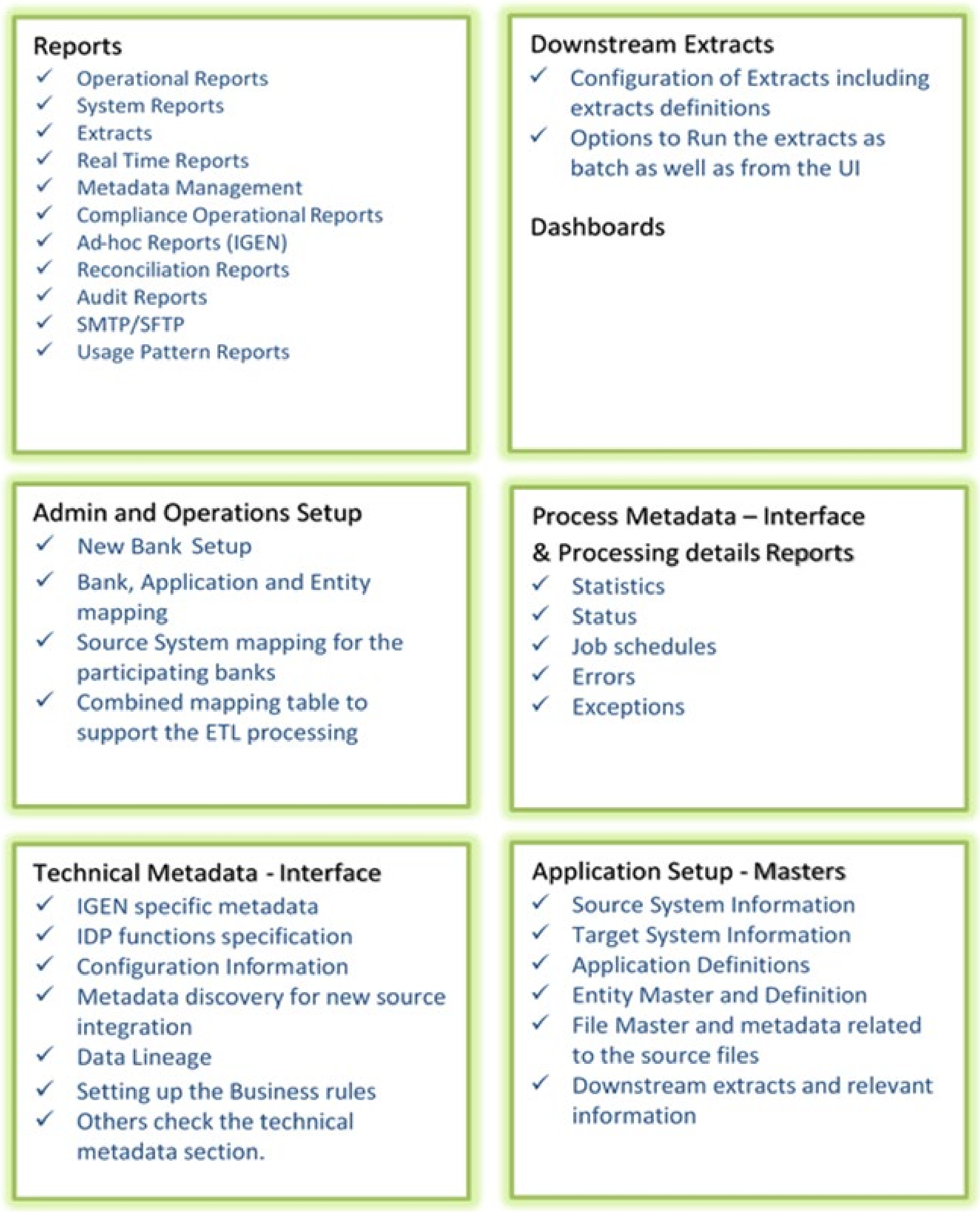


Figure 9. IDP/SSO Integration

The CAPE RAS UI supports the following functions:



Reports
- ✓ Operational Reports
- ✓ System Reports
- ✓ Extracts
- ✓ Real Time Reports
- ✓ Metadata Management
- ✓ Compliance Operational Reports
- ✓ Ad-hoc Reports (IGEN)
- ✓ Reconciliation Reports
- ✓ Audit Reports
- ✓ SMTP/SFTP
- ✓ Usage Pattern Reports

Downstream Extracts
- ✓ Configuration of Extracts including extracts definitions
- ✓ Options to Run the extracts as batch as well as from the UI

Dashboards

Admin and Operations Setup
- ✓ New Bank Setup
- ✓ Bank, Application and Entity mapping
- ✓ Source System mapping for the participating banks
- ✓ Combined mapping table to support the ETL processing

Process Metadata – Interface & Processing details Reports
- ✓ Statistics
- ✓ Status
- ✓ Job schedules
- ✓ Errors
- ✓ Exceptions

Technical Metadata - Interface
- ✓ IGEN specific metadata
- ✓ IDP functions specification
- ✓ Configuration Information
- ✓ Metadata discovery for new source integration
- ✓ Data Lineage
- ✓ Setting up the Business rules
- ✓ Others check the technical metadata section.

Application Setup - Masters
- ✓ Source System Information
- ✓ Target System Information
- ✓ Application Definitions
- ✓ Entity Master and Definition
- ✓ File Master and metadata related to the source files
- ✓ Downstream extracts and relevant information

Figure 10. Functions Support

CAPE RAS Extension Capabilities

CAPE RAS is packaged to include a set of source systems and related tables.The component architecture can be extended to support client-specific and country-specific implementation needs.

The following diagram displays extension features as per implementation needs:

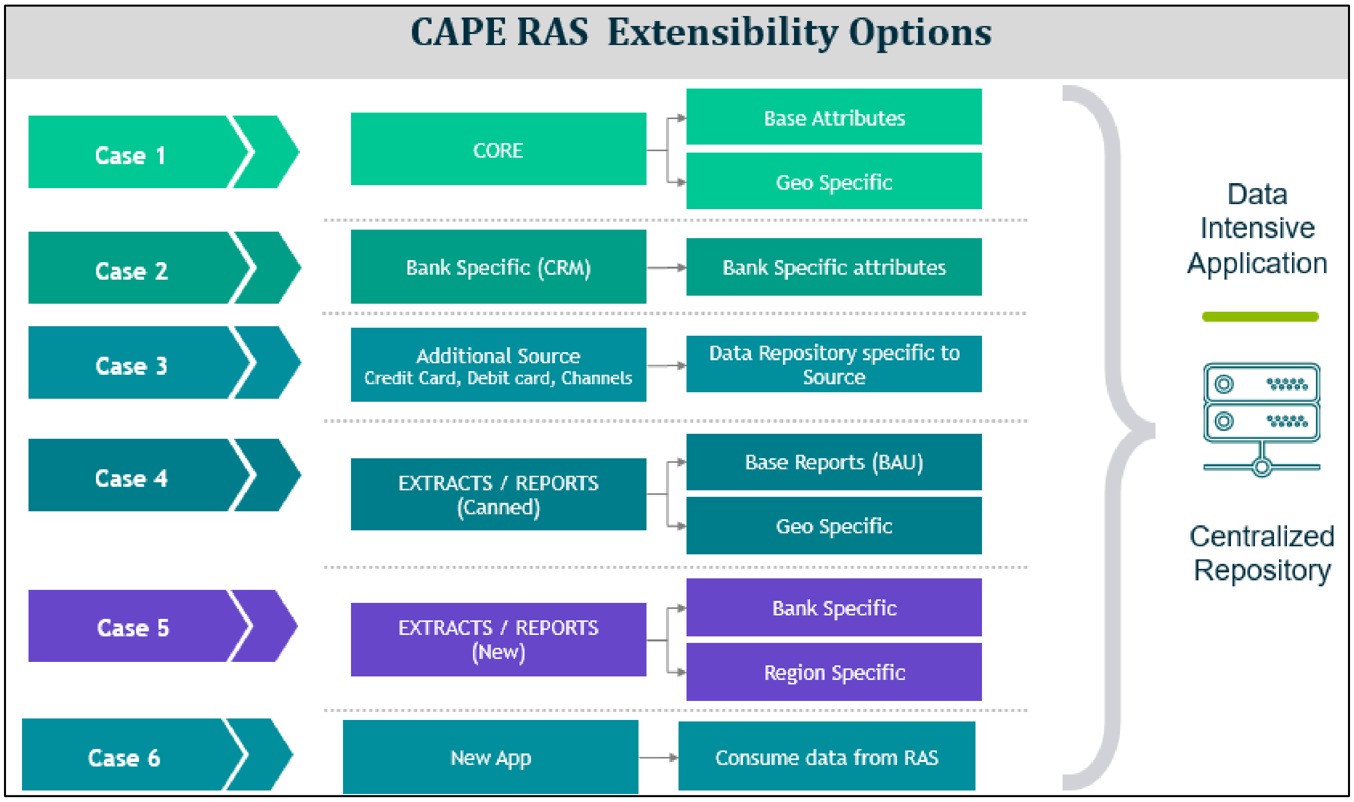


Figure 11. Extensibility Options

Extension Type 1 – Core

The first level of extension provides options for the implementation bank to build extensions on the core and implement the changes on CAPE RAS. For example, the bank captures additional KYC (Know Your Customer) information which is not in the base platform but has been built as an extension in the core. Extensions can be created in CAPE RAS without impacting the base code.

Note:

The customer view should be enhanced if attributes from the new tables are required for reporting and extracts

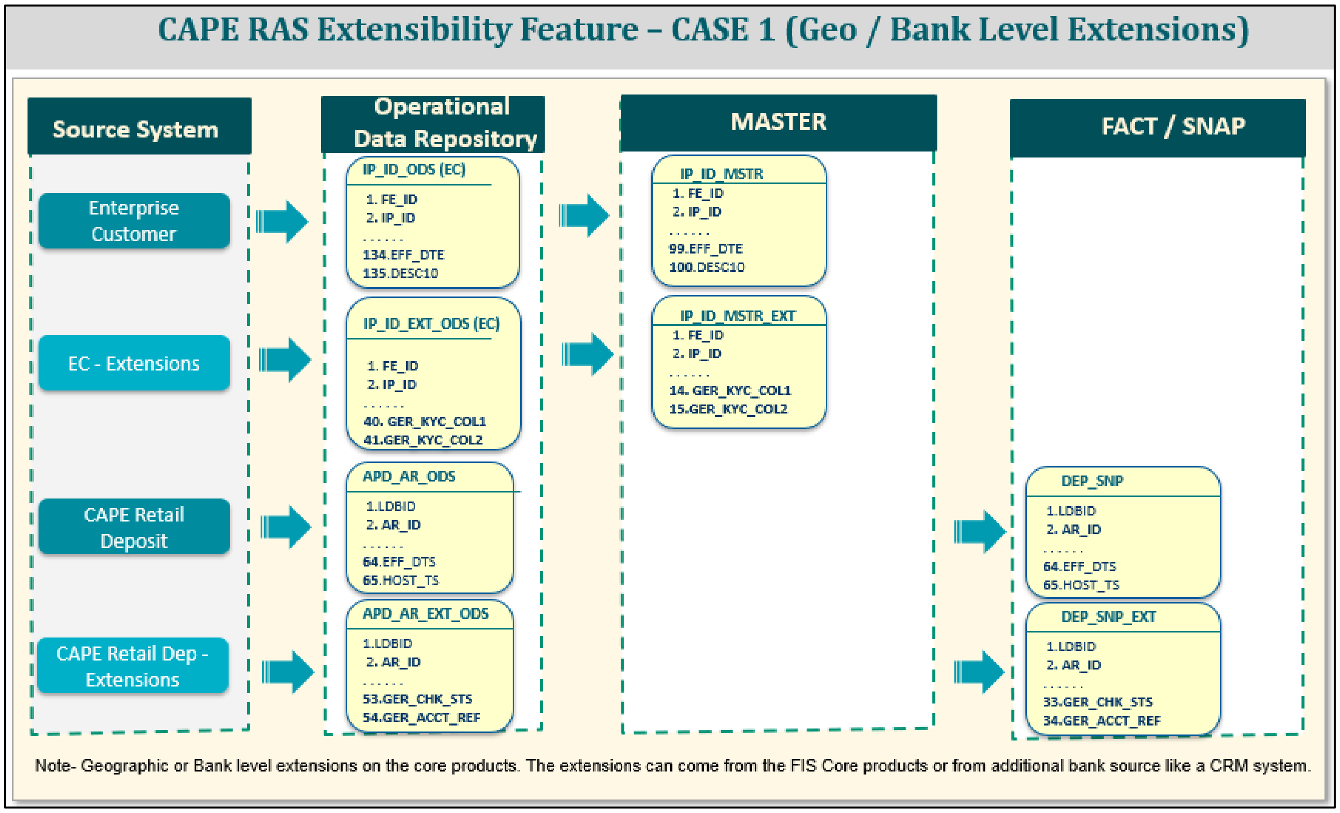


Figure 12. Extensibility Feature Case 1 - Core

Extension Type 2 – Bank Specific (CRM)

An implementation can integrate bank-specific enterprise components like Customer Information Systems (CIS) with CAPE RAS.

Note:

The naming convention used in the ODS layer can be specific to the applications used. This helps the operations team in the case of triaging.

To load data to the data warehouse tables (e.g., IP Master), build the mapping between the source CIS systems and the IP Master. Base the ETL (extract, transform, and load) processing on this new mapping.

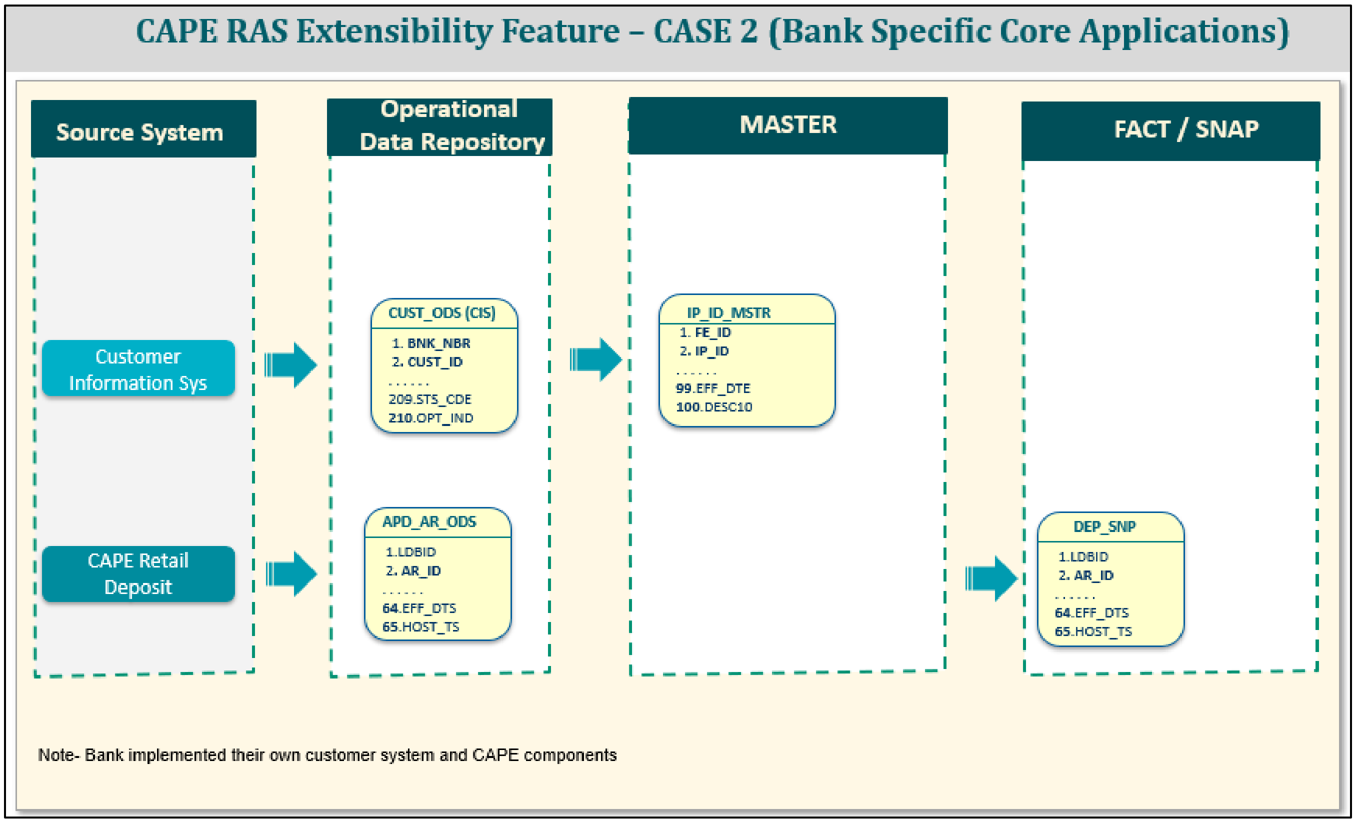


Figure 14. Extensibility Feature Case 2 – Bank-Specific (CRM)

Extension Type 3 – Additional Source (Credit Card, Debit Card, Channels)

The bank can integrate credit card, Enterprise general ledgers (GLs), and debit card or other additional systems that might or might not be integrated with the FIS Customer or Accounts system. The data can be ingested and loaded to the CAPE RAS system.

Note:

The naming convention used in the ODS layer can be specific to the applications used. This helps the operations team in the case of triaging.

To integrate data with the data warehouse (DW) tables, add data mapping and ETL processing to bring additional attributes to the existing DW tables.

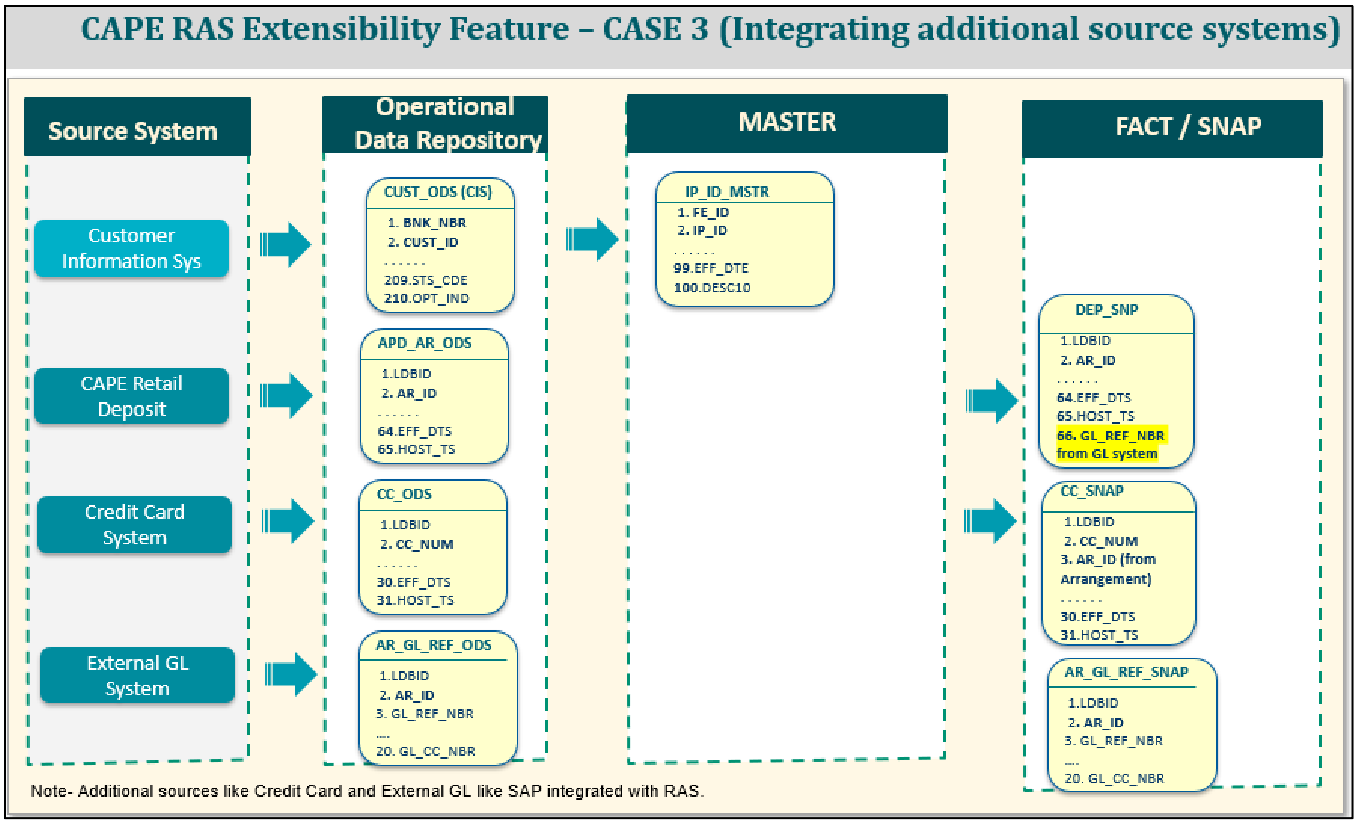


Figure 15. Extensibility Feature Case 3 – Additional Source

Extension Type 4 – Standard Extracts and Reports

CAPE RAS contains standard reports and extracts that can be run in a batch model as well as invoked from the CAPE RAS UI.

A financial entity can customize reports and extracts based on banking or geographical needs. Place the report object in the extension path. The report servicing layer checks for the specific reporting object in the extension path, runs it if available or runs it from the base path if the extensions are not available.

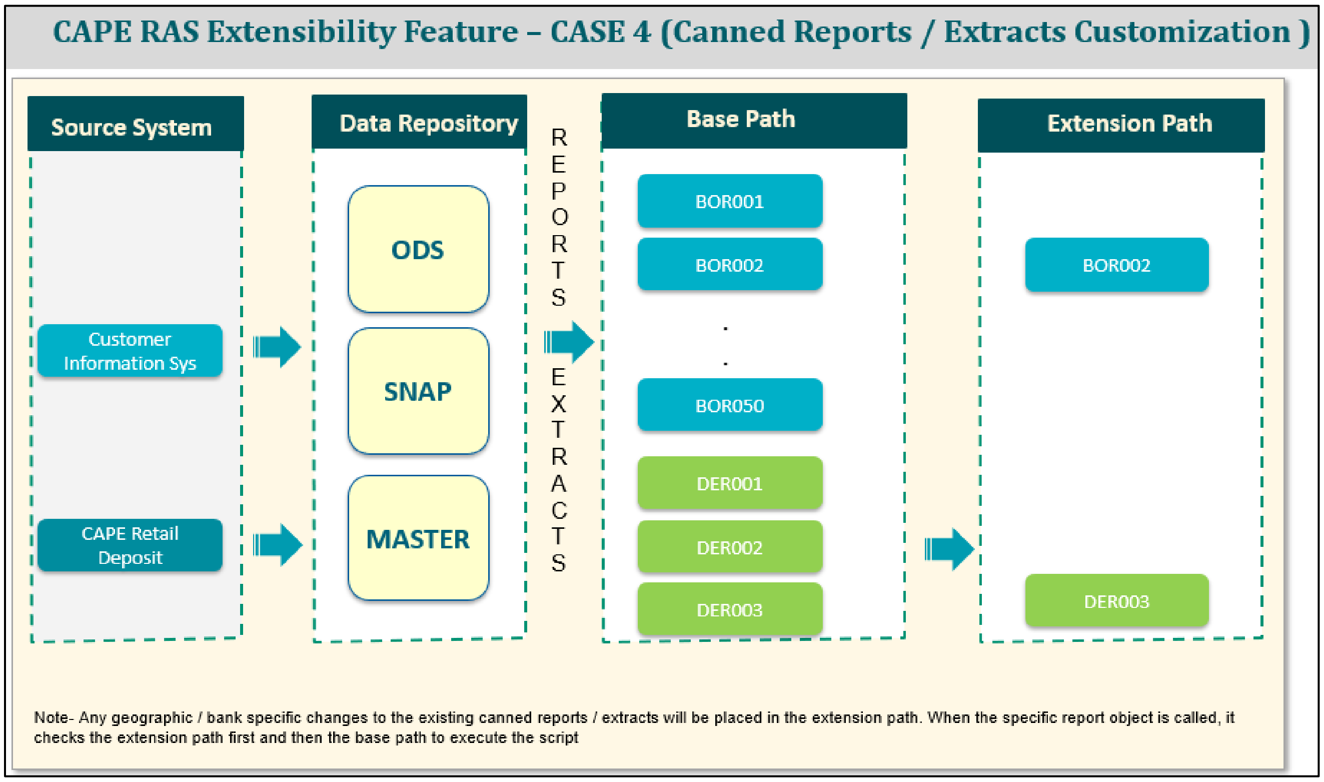

CAPE RAS Extensibility Feature – CASE 4 (Canned Reports / Extracts Customization)
Source System
Customer Information Sys
CAPE Retail Deposit
Data Repository
ODS
SNAP
MASTER
REPORTS
EXTRACTS
Base Path
BOR001
BOR002
BOR050
DER001
DER002
DER003
Extension Path
BOR002
DER003
Note- Any geographic / bank specific changes to the existing canned reports / extracts will be placed in the extension path. When the specific report object is called, it checks the extension path first and then the base path to execute the script

Figure 16. Extensibility Feature Case 4 – Standard Extracts and Reports

Extension Type 5 – New Extracts and Reports

Banks can create their own reports or add extracts to the CAPE RAS metadata list. After added to the metadata list, a user can invoke the reports/extracts from the CAPE RAS UI, as well as schedule them in the batch process.

New reports must use one of the CAPE RAS UI templates. If the reports must generate Excel or PDF content, update the respective scripts and metadata information in CAPE RAS.

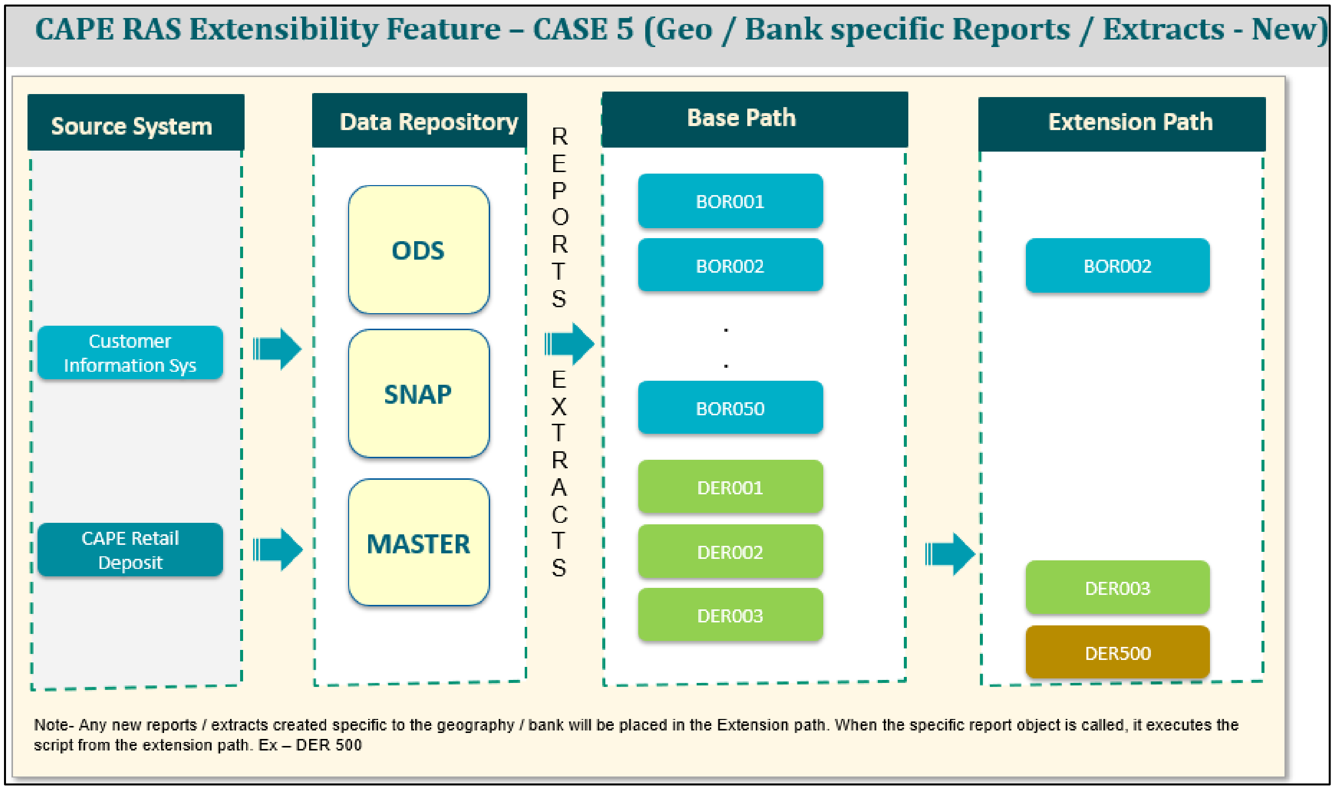

CAPE RAS Extensibility Feature – CASE 5 (Geo / Bank specific Reports / Extracts - New)
Source System
Customer Information Sys
CAPE Retail Deposit
Data Repository
ODS
SNAP
MASTER
REPORTS
EXTRACTS
Base Path
BOR001
BOR002
BOR050
DER001
DER002
DER003
Extension Path
BOR002
DER003
DER500
Note- Any new reports / extracts created specific to the geography / bank will be placed in the Extension path. When the specific report object is called, it executes the script from the extension path. Ex – DER 500

Figure 17. Extensibility Feature Case 5 – New Extracts and Reports

Extension Type 6 – New Applications

CAPE RAS acquires data from multiple source systems, transforms it, and integrates and loads it in the centralized data repository. The DW schema maintains the customer and account master along with a daily snapshot of the account balances.

Any application that requires integrated data from the core system can source the data from CAPE RAS.

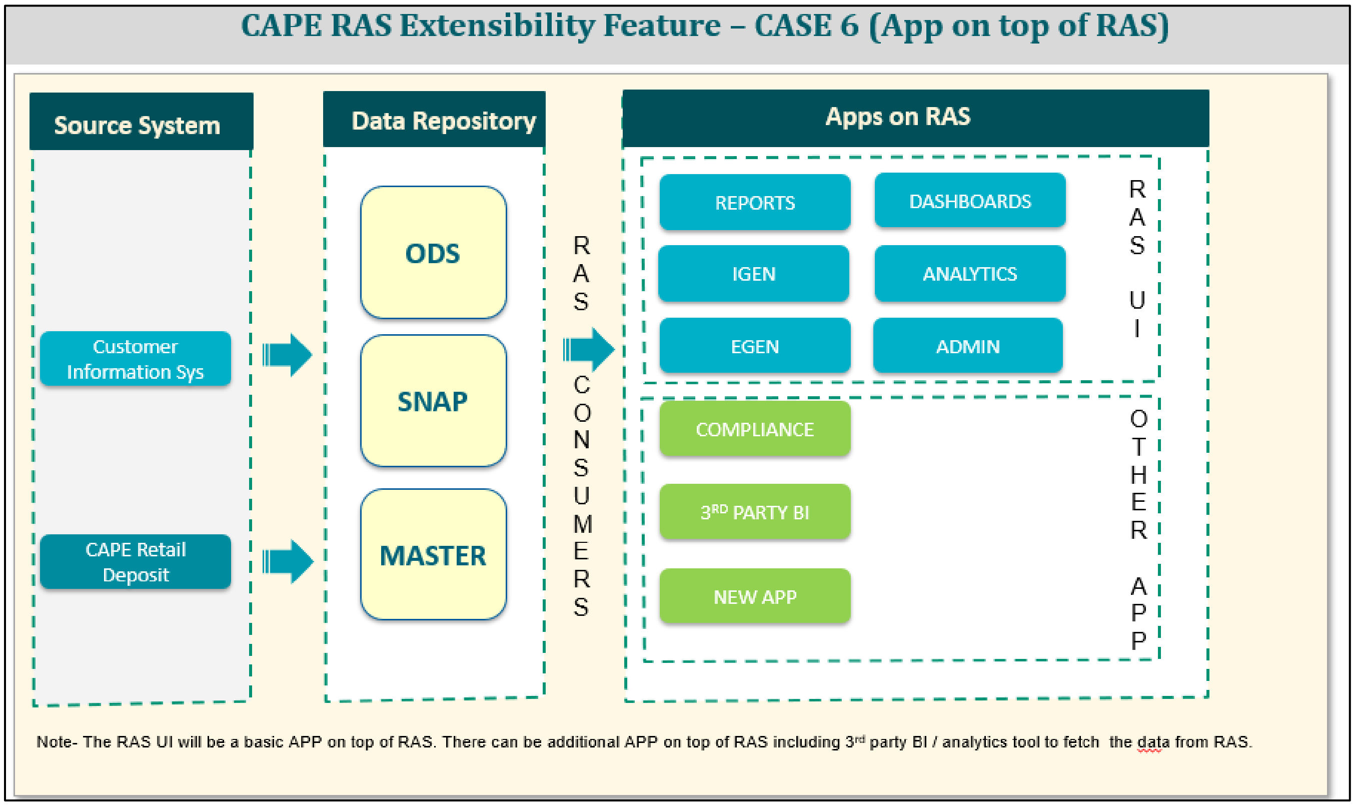

CAPE RAS Extensibility Feature – CASE 6 (App on top of RAS)
Source System
Customer Information Sys
CAPE Retail Deposit
Data Repository
ODS
SNAP
MASTER
RAS CONSUMERS
Apps on RAS
REPORTS
DASHBOARDS
IGEN
ANALYTICS
EGEN
ADMIN
RAS UI
COMPLIANCE
3RD PARTY BI
NEW APP
OTHER APP
Note- The RAS UI will be a basic APP on top of RAS. There can be additional APP on top of RAS including 3rd party BI / analytics tool to fetch the data from RAS.

Figure 18. Extensibility Feature Case 6 – New Applications

Extension Capabilities

- Additional data attributes to support a specific client implementation
- Additional data attributes to support a specific geography
- New reports requirements beyond standard report list
- New extracts requirement beyond standard extracts
- New dashboards requirement beyond standard dashboards
- New applications beyond CAPE RAS

Component Based Solution

CAPE RAS can be built as a modular product with logical boundaries between components for customizable and transparent implementation. CAPE RAS is designed to minimize dependencies between modules.

Each component shall be version-controlled for continuos integration.

Implimenting only those components preferred by the finanical entity provides flexibility in component development, maintenance, and upgrade.

This approach enhances:

- Customized product implementation
- Independent and reusable components closely aligned with core components
- Reduced complexity
- Predefined interdependence
- Improved maintainability
- Country-specific or client-specific extension support
- Configurable components based on parallel bach flow

The following component architecture diagram displays multiple scenarios of CAPE RAS implementation.

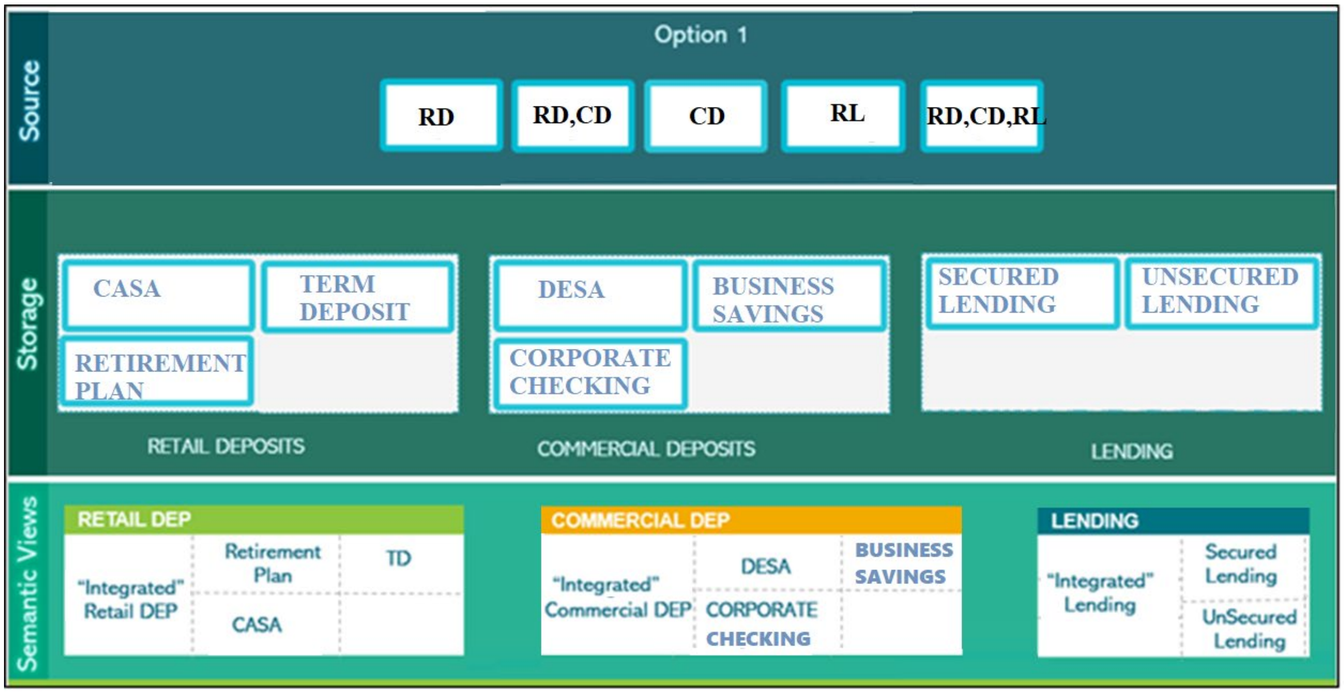


Figure 19. Component-Based Solution

CAPE RAS Base Components

- Common Scripts
  - Data Quality Process
  - User Interface
  - Enterprise Organization
  - Enterprise Customer
  - Retail Deposits
  - Commercial Deposits
  - Retail Lending
  - Enterprise Banking Collateral
  - EFT
  - Metadata Management Console
  - Backward Compatibility
- Product extensions based on following extensions
  - Country-specific extensions
  - Client-specific extensions

Deployment Model

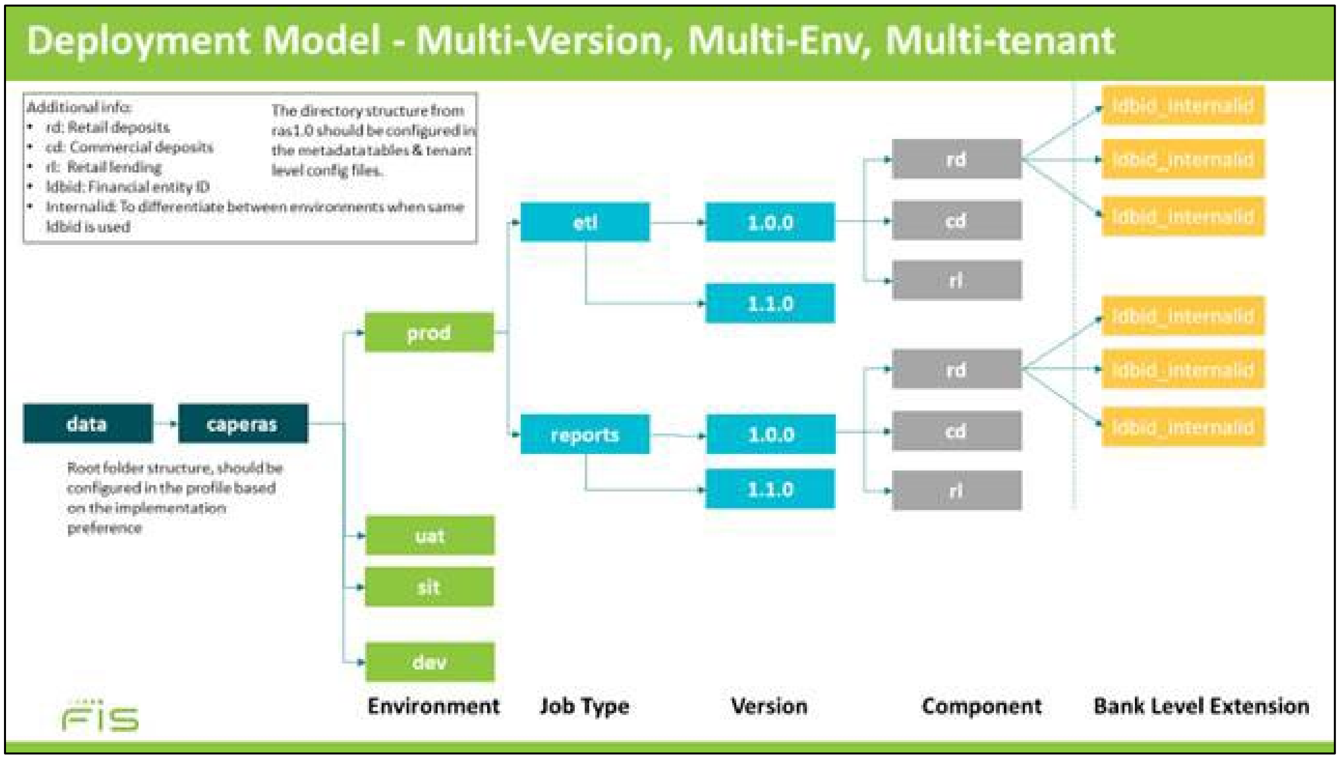
CAPE RAS supports multiple versions, multiple environments, and multitenancy on the same cluster.

Note:

The folder structure is critical to support the multi functionalities.

Key Features

- One-click deployment using Flyway
- Automated stackable deployment
- Rollback/backout process



Additional info:
- rd: Retail deposits
- cd: Commercial deposits
- rl: Retail lending
- ldbid: Financial entity ID
- Internalid: To differentiate between environments when same ldbid is used

The directory structure from ras1.0 should be configured in the metadatatables & tenant level config files.

Root folder structure, should be configured in the profile based on the implementation preference

Figure 20. Deployment Model

DFD/Network Architecture – Hadoop

CAPE RAS Data Flow Diagram (DFD) Process

- RT process
    - Source data from Modern Banking Platform is streamed through Kafka on a real-time basis.
    - The Kafka consumer is written based on the Spark Structured Streaming method and is placed on the edge node where it listens to events in the Kafka port.
    - As soon as events are available, Kafka consumer reads them and writes them directly into the CDC Inbox table (Event Repo table) in Hive, without landing into HDFS (Hadoop Distributed File System).
    - Each product line has a separate CAPE RAS Event Repo Hive table.
    - CDC to RT Engine processes events stored in the CAPE RAS Event Repo tables, parses them, and loads data into the Hive tables.
- Batch Process
    - Source data is pushed from Enterpise components (e.g., Enterprise Customer, Enterprise Organization as a flat file.
    - For Modern Banking Platform core processing components (e.g., Deposits, Loans), source data is pulled directly from the Oracle tables.
    - ETL Spark jobs in the edge nodes acquire the data, transform it, and load it to HDFS storage.
    - Reports and downstream extracts jobs are triggered from the Enterprise Scheduler (Airflow, Zena) and the output files are stored in the NAS drive.
    - The complete batch flow is processed via the combined parallel batch flow model of Core and CAPE RAS, which in turn enhances the performance.
- Data Servicing and Presentation
    - Presentation Layer – Users run the report from the CAPE RAS UI using the BOD framework deployed in the OpenShift container.
    - Servicing Layer – The UI invokes Python services hosted in the Virtual Python Environment (VPE) which performs user authorization and calls the reporting scripts.
    - The reporting scripts initiates the queries using Hive/Impala connection and the response is returned to the calling UI in JSON format.

CTE 1 CAPE RAS DFD (Data Flow Diagram)

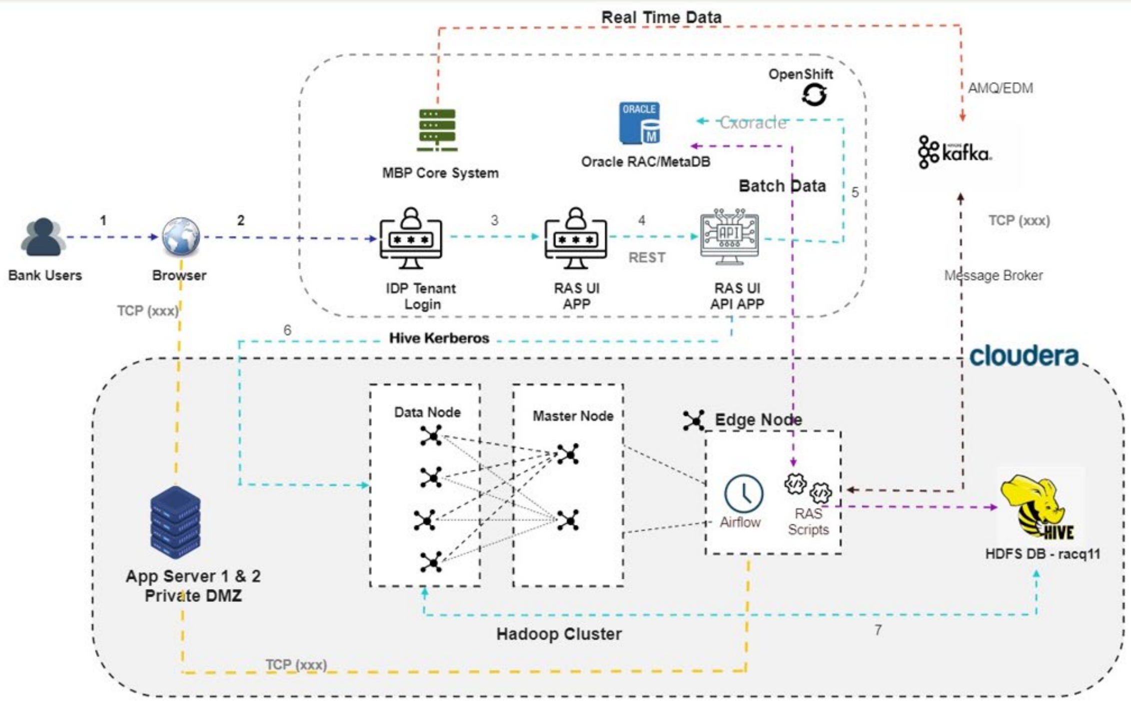


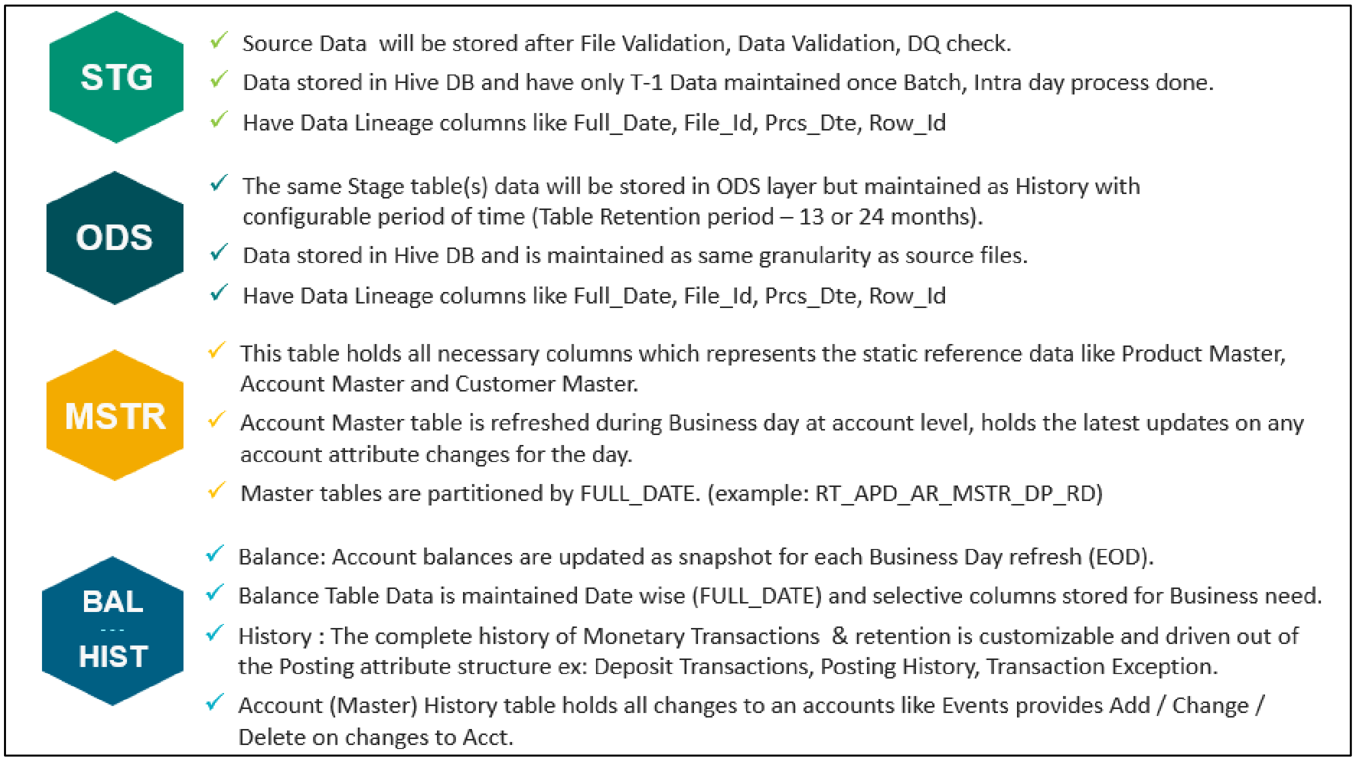
Figure 21. Data Servicing and Presentation

CAPE RAS – Data Architecture

CAPE RAS Data Repository is comprised of the following components.

Business Tables

- ODS – Data is maintained at the same granularity as source files.
- Master – Master data represents static reference data (e.g., Account Master, Customer Master).
- History – The history table is applicable for monetary transactions and the master information like Customer and Account.
- Balance – Data is maintained per date.



STG
- ✓ Source Data will be stored after File Validation, Data Validation, DQ check.
- ✓ Data stored in Hive DB and have only T-1 Data maintained once Batch, Intra day process done.
- ✓ Have Data Lineage columns like Full_Date, File_Id, Prcs_Dte, Row_Id

ODS
- ✓ The same Stage table(s) data will be stored in ODS layer but maintained as History with configurable period of time (Table Retention period – 13 or 24 months).
- ✓ Data stored in Hive DB and is maintained as same granularity as source files.
- ✓ Have Data Lineage columns like Full_Date, File_Id, Prcs_Dte, Row_Id

MSTR
- ✓ This table holds all necessary columns which represents the static reference data like Product Master, Account Master and Customer Master.
- ✓ Account Master table is refreshed during Business day at account level, holds the latest updates on any account attribute changes for the day.
- ✓ Master tables are partitioned by FULL_DATE. (example: RT_APD_AR_MSTR_DP_RD)

BAL --- HIST
- ✓ Balance: Account balances are updated as snapshot for each Business Day refresh (EOD).
- ✓ Balance Table Data is maintained Date wise (FULL_DATE) and selective columns stored for Business need.
- ✓ History : The complete history of Monetary Transactions & retention is customizable and driven out of the Posting attribute structure ex: Deposit Transactions, Posting History, Transaction Exception.
- ✓ Account (Master) History table holds all changes to an accounts like Events provides Add / Change / Delete on changes to Acct.

Figure 22. Business Tables

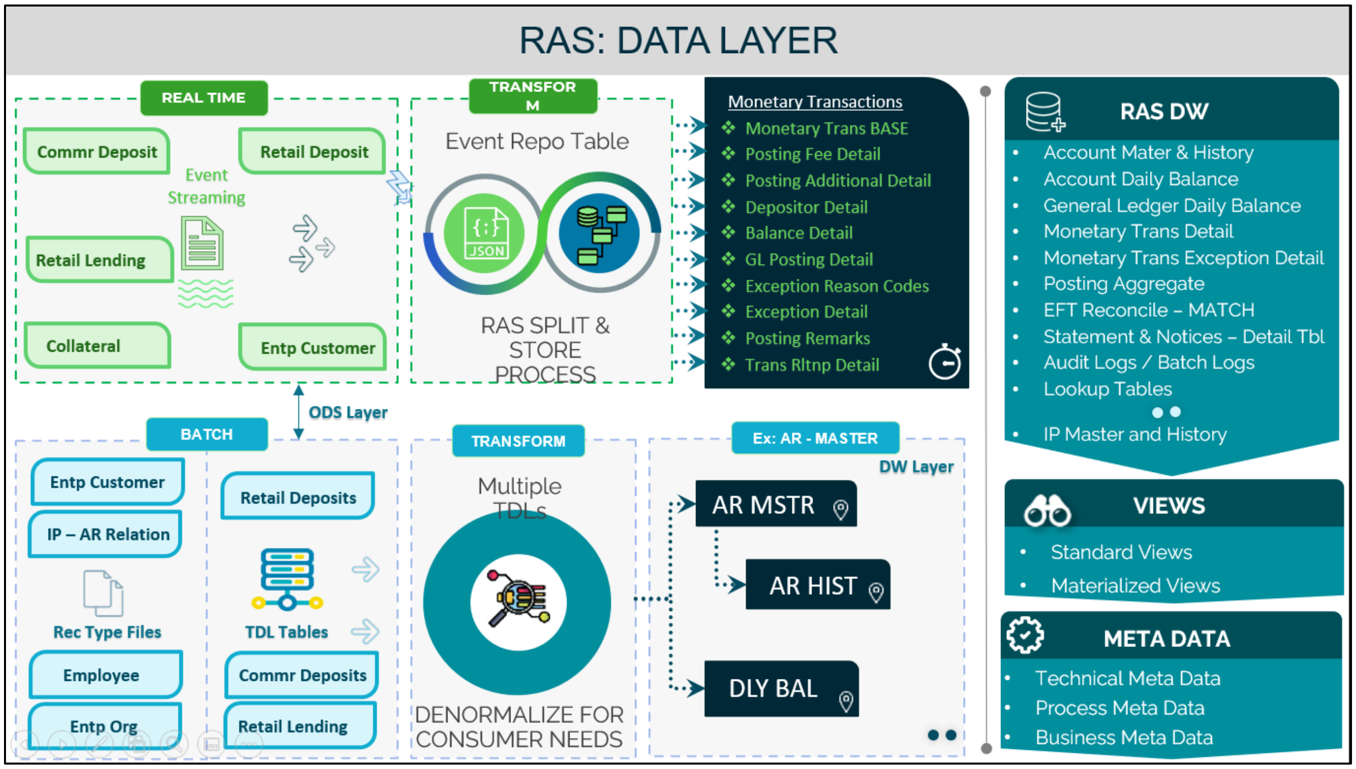
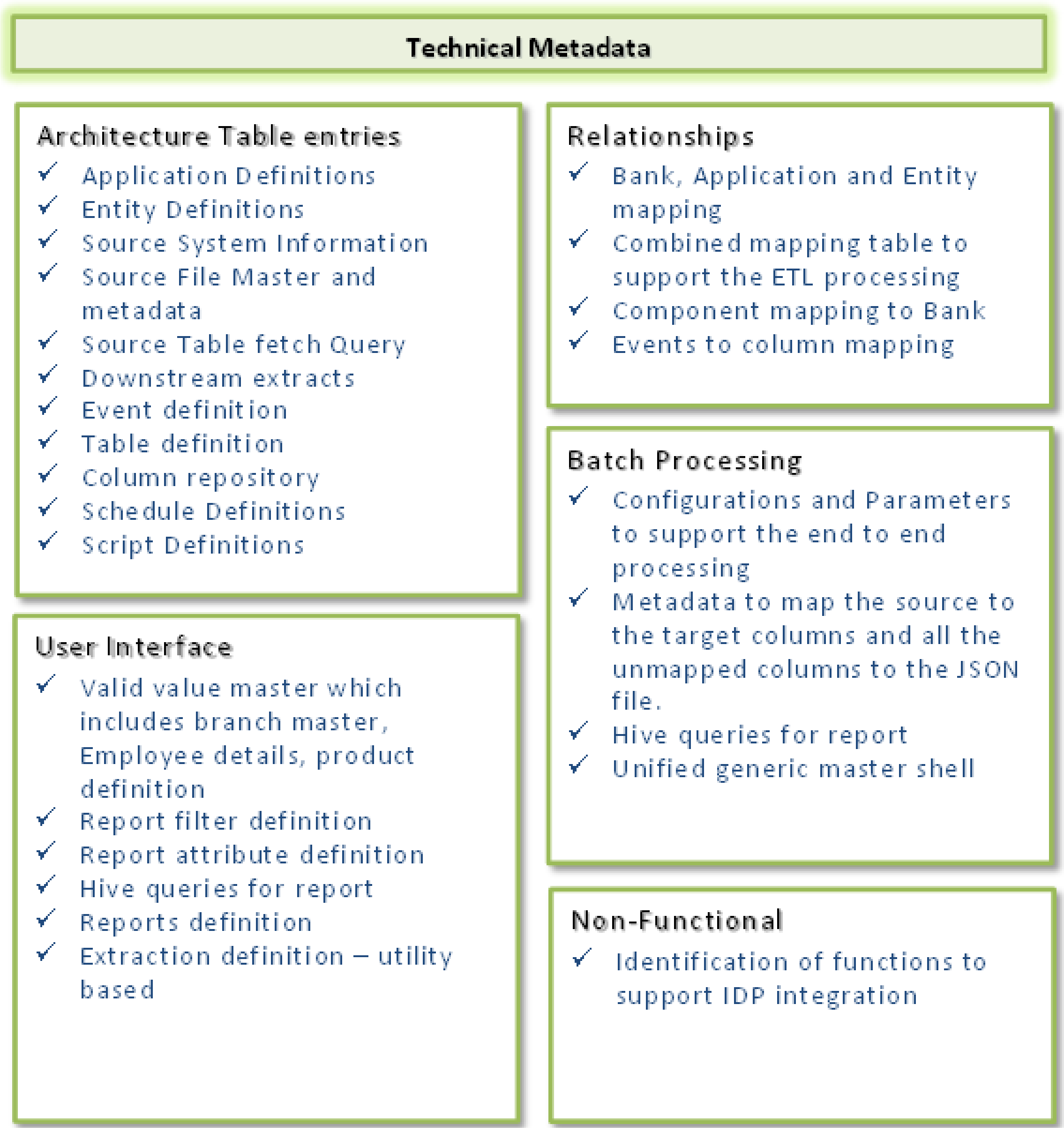


Figure 22. Data Architecture

Metadata Tables

The metadata tables support technical, business, and process metadata.



Technical Metadata

Architecture Table entries

- ✓ Application Definitions
- ✓ Entity Definitions
- ✓ Source System Information
- ✓ Source File Master and metadata
- ✓ Source Table fetch Query
- ✓ Downstream extracts
- ✓ Event definition
- ✓ Table definition
- ✓ Column repository
- ✓ Schedule Definitions
- ✓ Script Definitions

Relationships

- ✓ Bank, Application and Entity mapping
- ✓ Combined mapping table to support the ETL processing
- ✓ Component mapping to Bank
- ✓ Events to column mapping

User Interface

- ✓ Valid value master which includes branch master, Employee details, product definition
- ✓ Report filter definition
- ✓ Report attribute definition
- ✓ Hive queries for report
- ✓ Reports definition
- ✓ Extraction definition – utility based

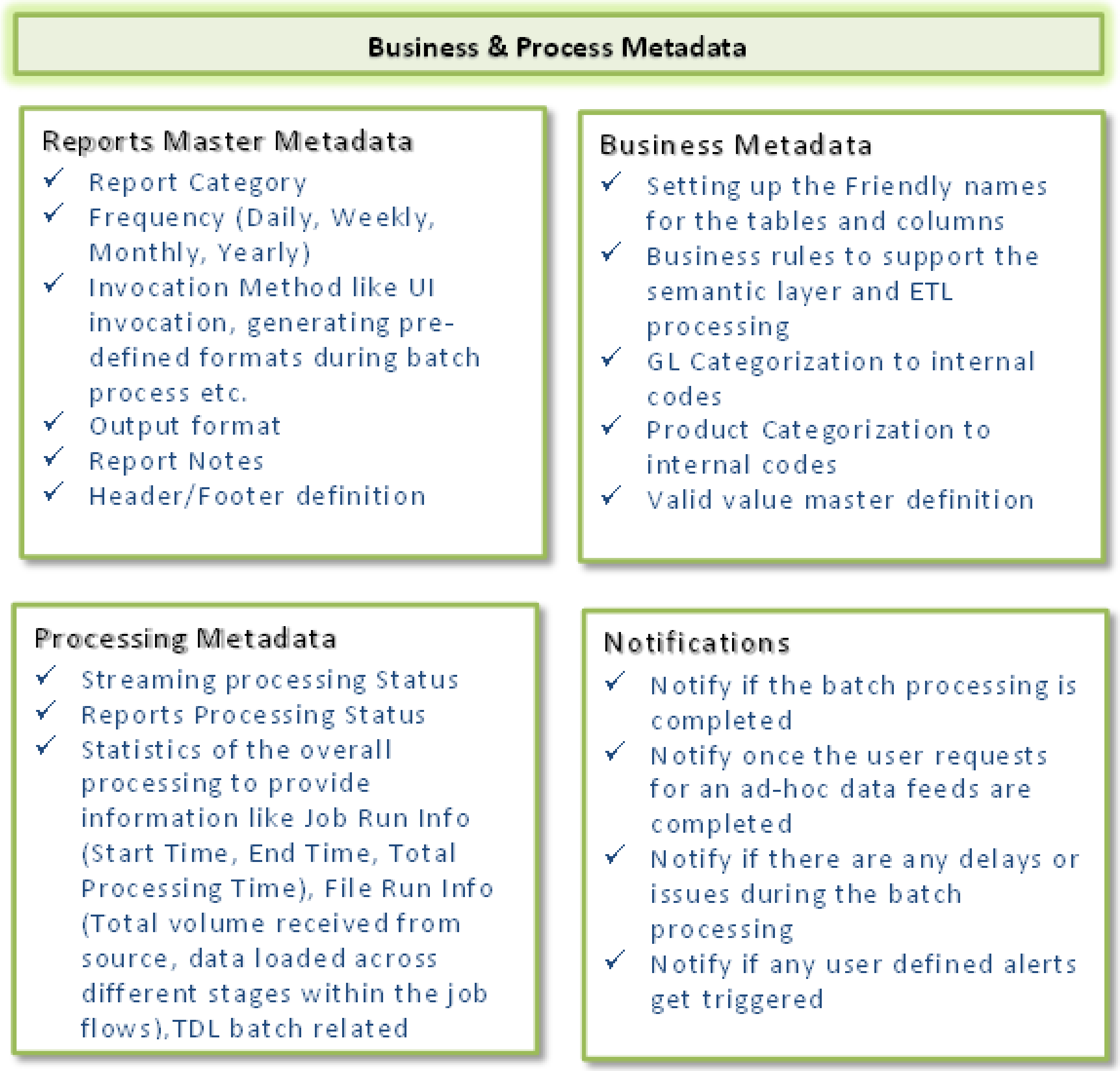
Batch Processing

- ✓ Configurations and Parameters to support the end to end processing
- ✓ Metadata to map the source to the target columns and all the unmapped columns to the JSON file.
- ✓ Hive queries for report
- ✓ Unified generic master shell

Non-Functional

- ✓ Identification of functions to support IDP integration

Figure 23. Technical Metadata Table



Business & Process Metadata

Reports Master Metadata
- ✓ Report Category
- ✓ Frequency (Daily, Weekly, Monthly, Yearly)
- ✓ Invocation Method like UI invocation, generating pre-defined formats during batch process etc.
- ✓ Output format
- ✓ Report Notes
- ✓ Header/Footer definition

Business Metadata
- ✓ Setting up the Friendly names for the tables and columns
- ✓ Business rules to support the semantic layer and ETL processing
- ✓ GL Categorization to internal codes
- ✓ Product Categorization to internal codes
- ✓ Valid value master definition

Processing Metadata
- ✓ Streaming processing Status
- ✓ Reports Processing Status
- ✓ Statistics of the overall processing to provide information like Job Run Info (Start Time, End Time, Total Processing Time), File Run Info (Total volume received from source, data loaded across different stages within the job flows),TDL batch related

Notifications
- ✓ Notify if the batch processing is completed
- ✓ Notify once the user requests for an ad-hoc data feeds are completed
- ✓ Notify if there are any delays or issues during the batch processing
- ✓ Notify if any user defined alerts get triggered

Figure 24. Business and Process Metadata Table

Security and Compliance

The Security framework adheres to established security principles.

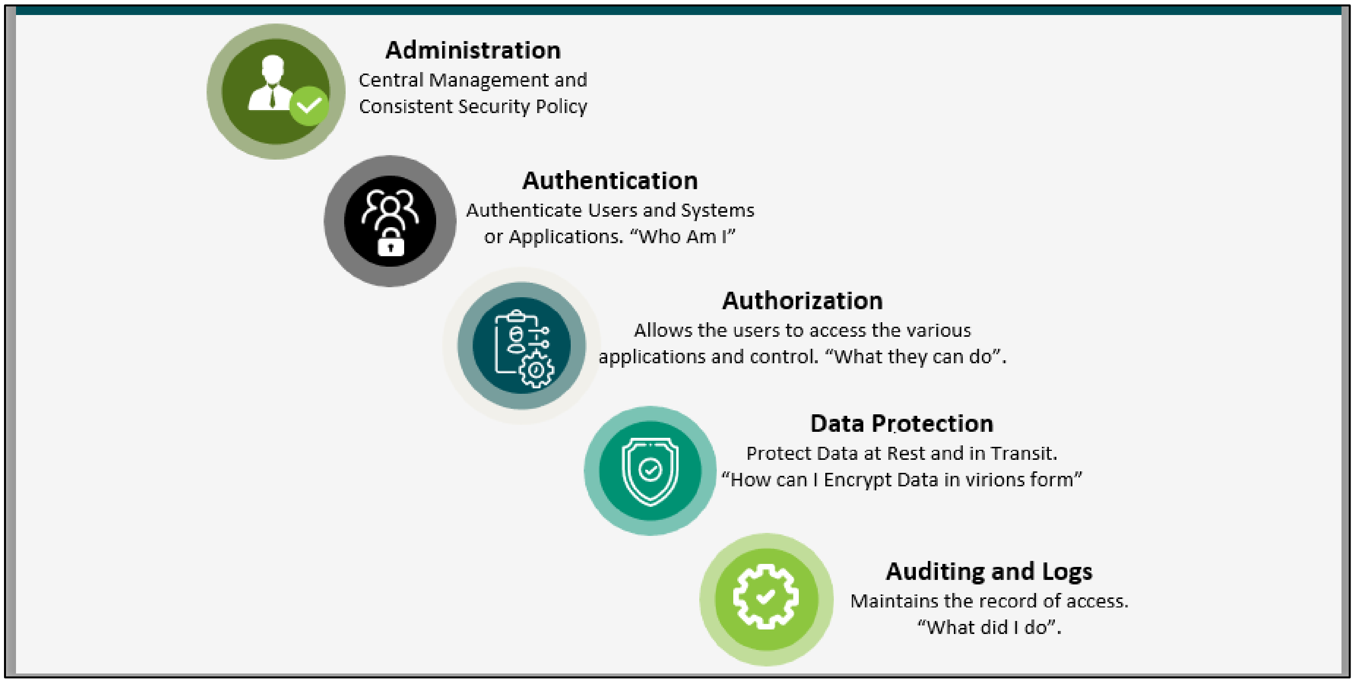

Administration
Central Management and Consistent Security Policy
Authentication
Authenticate Users and Systems or Applications. "Who Am I"
Authorization
Allows the users to access the various applications and control. "What they can do".
Data Protection
Protect Data at Rest and in Transit. "How can I Encrypt Data in virions form"
Auditing and Logs
Maintains the record of access. "What did I do".

Figure 25. Security Framework

- Apache Ranger framework enables, monitors, and manages comprehensive data security across the Hadoop platform
- IdP framework manages user-level security and access control
- Encryption and Decryption
  - Standard Enterprise Encryption tools like Vormetric secure the NAS Spool Storage file system
  - AES-based encryption is used for the data nodes on the Hadoop cluster
- Data Masking feature masks configurable sensitive columns within extracts
- Data exchange from OpenShift to Hadoop cluster is protected using Transport Layer Security (TLS)
- Comprehensive and distinctive maintenance of logs (e.g., user access logs, batch process logs) exist for compliance and auditing purposes

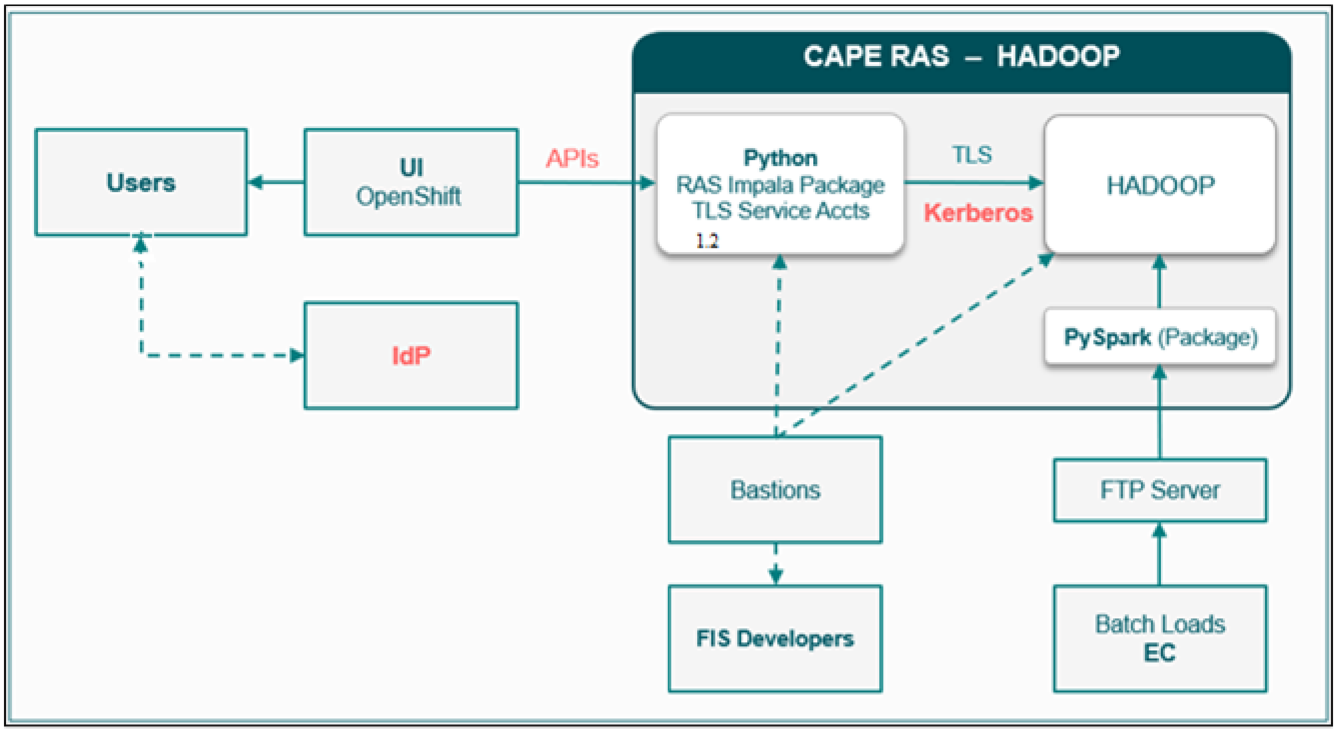


Figure 26. Hadoop-based CAPE RAS Ecosystem

Solution Approach and Strategy

Data Integration Strategy

Source System Analysis

The system evaluates source data against predefined subject areas and conducts a gap analysis. The gap analysis helps redefine the model and checks for additional data elements.

Batch/Stream Processing Framework Features

- Defines the extract, transform and load process, or FTP process if required, to transfer to other servers or systems
- Uses technical metadata
- Includes daily, weekly, monthly, hourly, etc.

Data Quality Framework Features

- Null value handling
- Invalid data handling
- Lookup validation
- DQ Error handling
- Business Rule validations

Data Profiling Framework Features

- Data quality and credibility – After analyzing data, the application can help eliminate duplications or anomalies
- Predictive decision-making – Use profiled information to stop small mistakes from becoming big problems

Data Loading Framework Features

- Loading Methods
- Handling of Bad Records

Restart Strategy Features

- Restart ability in the event of failure during the process
- Step ID parameter for all jobs can be fetched from the operational metadata tables
- Override option restarts the process from the beginning

Exception Handling Features

- All steps have proper exception handling and an alerts mechanism. Errors are captured in the operational metadata tables.

Backward Compatibility

The CAPE RAS Data Interface enables backward compatibility using the following design principles:

- New data extensions due to extensions in the Source Data components (e.g., Modern Banking Platform Core or Enterprise Components) are designed to ensure no impact to existing consumers of data.
- If a column must be modified for an existing table, a new column is added if it impacts an existing extract/reporting processes (versus changing the existing column). This prevents impact to existing consumers of this column. The existing column is depreciated after 18 months.

Note:

Dates of depreciation of any column are documented in the CAPE RAS Data Model documentation published with each GA release.

Scheduling Strategy Features

- CAPE RAS jobs are segregated into critical and non-critical categories and are run in parallel along with Core jobs based on their dependencies. Critical Jobs can be given higher priority than the non-critical jobs to meet the service-level agreement (SLA) time.
- The Enterprise scheduling tool (e.g., Zena, Airflow) triggers jobs within the FIS data centers.
- Prerequisite for CAPE RAS batch process:
  - Extracts required from the source system (e.g., Enterprise Customer, Deposits) for CAPE RAS batch processing.
  - Place all required input extracts in the CAPE RAS Server spool path. The spool path is created for a tenant during the tenant setup.
  - Check if a batch is already running for the tenant by querying the CAPE RAS Batch Metadata table - “btch_stat” column from tb_pm_batch_info tables. Provide the batch status. Enterprise Scheduler starts the batch workflow for COB (current business date).
  - A separate scheduling flow is available for each tenant based on the component and the specific release created during tenant setup. FIS uses the Airflow scheduler in the development region.
  - Airflow DAG is defined in a YAML template. The YAML template is a collection of all tasks/jobs organized in a way that reflects their relationships and dependencies. The YAML flow along with the batch flow document can automatically generate from the scheduler group-based metadata tables.
  - If a task/job fails, rerun or skip the failed task/job based on instruction given in the workflow document.
  - After batch completion, CAPE RAS outbound extracts are available in the specific spool path.

Non-Functional Components

Database Approach and Strategy

This section provides database-related maintenance required to keep data within Hive.

Performance Tuning

The performance tuning measures are handled throughout the life cycle of data and as per the design principles of the tools and technologies used (e.g., Hadoop, Python, Spark, Hive).

Purge Process

The purge process is based on the history retention specification available in the metadata tables and scripts.

Table Classification	How It Processes
Master Tables (SW)	Metadata Configurable (File Master) to purge given tables
Real Time ODS Tables	Handled by ETL scripts
Real Time Stage Tables	Handled by Ad Hoc scripts
Balance Tables (DW)	Deletes only Month End Balance and beyond 5 years data

Disaster Recovery Strategy

The Disaster Recovery (DR) Strategy is run as per implementation requirements. DR must be set up for all Hadoop components and Appliction servers.

The Disaster Recovery Strategy is outlined for different contexts and is implemented as per bank requirements.

The Disaster Recovery Strategy is captured in the DR documentation created during implementation.

Infrastructure

The tools and technologies used in the CAPE RAS are provided below. The version and flavors change during implementation and are captured in the Deployment guide.

Tools Used

Tools	Software License	Application Code Review	Usage
HTML	Open Source	Code Scan Required	UI
CSS	Open Source	Code Scan Required	UI
BOD	Open Source	Code Scan Required	UI
Cloudera Data Platform	CDP – Subscription	Manual Scan Required	Repository
Python	Open Source	Manual Scan Required	Scripting
Java	Enterprise	Manual Scan Required	Scripting

Certified Technical Environment (CTE1)

Refer to the Modern Banking Platform Certified Technical Environment (CTE) document for CTE1.

Certified Technical Environment (CTE2)

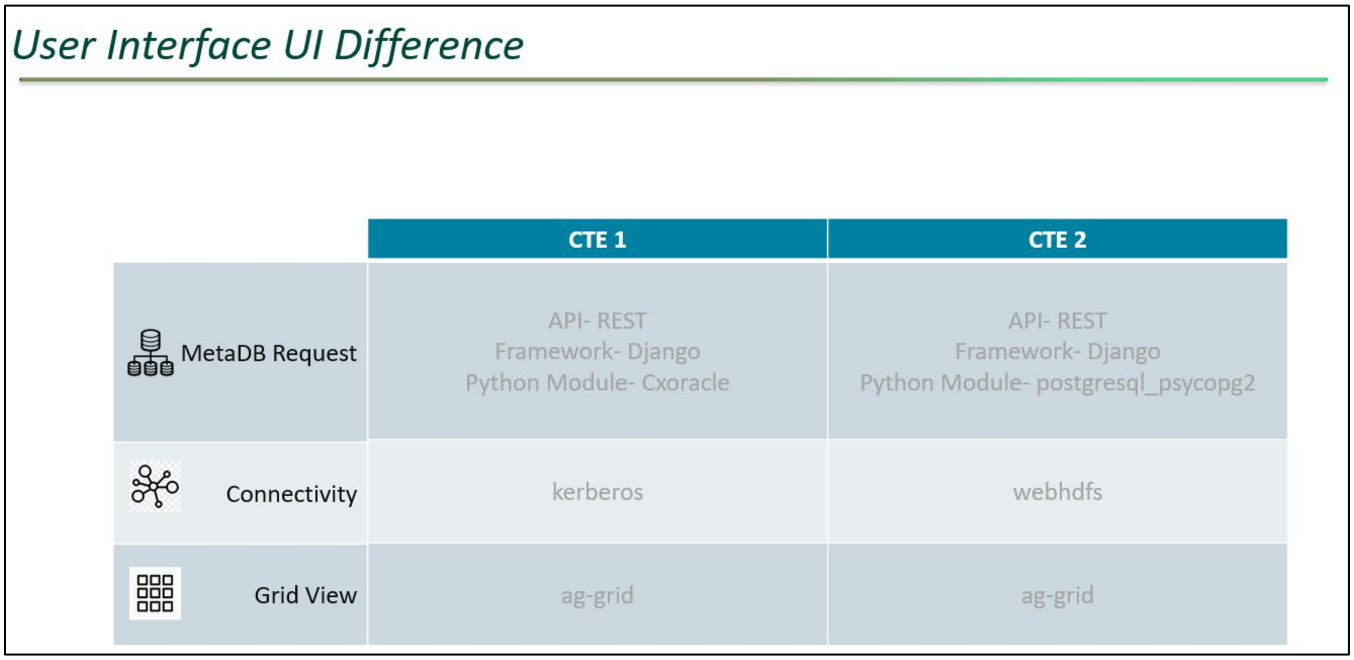
Refer to the Modern Banking Platform Certified Technical Environment (CTE) document for CT21.

CTE1 versus CTE2

CAPE RAS Solution supports a multi-cloud environment.

- CTE1 is hosted in a private cloud
- CTE2 is hosted in an Azure-based public cloud.

The following diagram depicts the data flow difference between CTE1 and CTE2.



User Interface UI Difference

	CTE 1	CTE 2
MetaDB Request	API- REST Framework- Django Python Module- Cxoracle	API- REST Framework- Django Python Module- postgresql_psycopg2
Connectivity	kerberos	webhdfs
Grid View	ag-grid	ag-grid

Figure 27. User Interface UI Difference

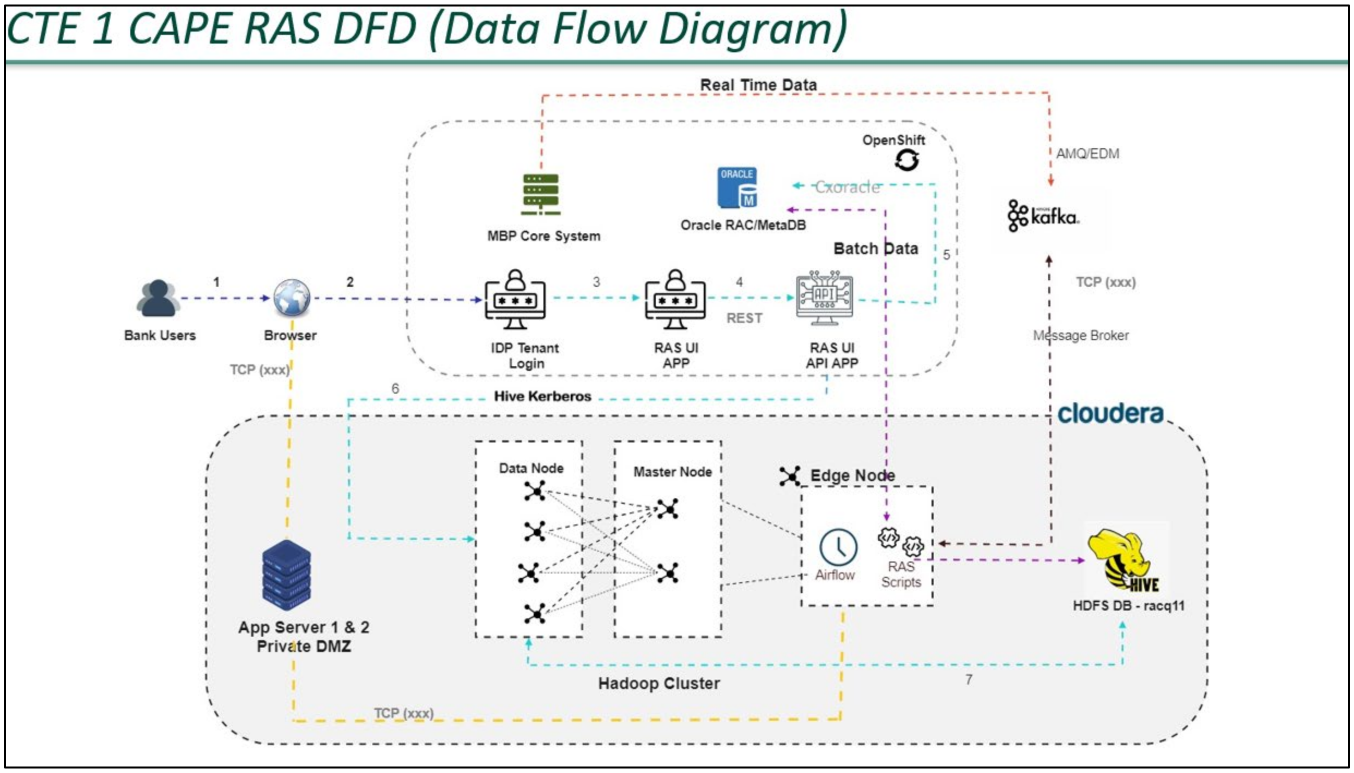
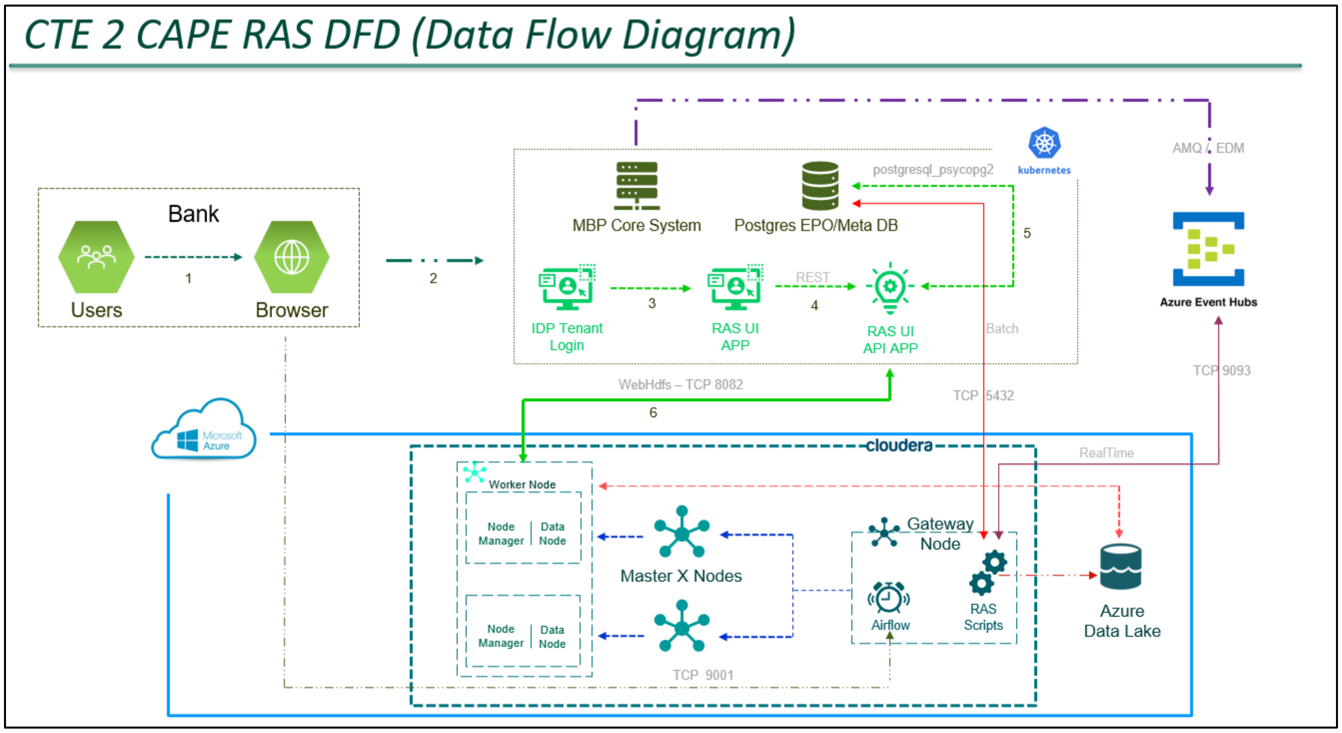


Figure 27. CTE1 CAPE RAS DFD (Data Flow Diagram)



References

Subject	CAPE RAS Links
CAPE RAS Data Dictionary	Available on CAPE RAS UI for viewing and downloading.
CAPE RAS Data Model	Available on CAPE RAS UI for viewing and downloading.

Revision History

Ver	Date	Author	Description
2.0	27-Jun-24	Krishna Davey	Modern Banking Platform v3.11 GA publication. • In *Project Definition* section, added metadata driven reconciliation framework and DPP, ability to consume real-time events from BLOB (CTE2), and locale-based date, currency, and decimal formats. • In *Solution Architecture* > *Source Systems*, added ITC for CTE2. • In *Solution Architecture* > *Data Acquisition* > *Batch Processing*, added note that CAPE RAS consumes data via extract files and TDL from ITC source. • In *Solution Architecture* > *Data Acquisition* > *Real-Time Processing*, added note re: real-time data warehousing in Transaction History. • In *Solution Architecture* > *CAPE RAS User Interface*, added DDP and data governance capabilities. Removed *Dashboard Designer* sub-section. For I-Gen Report Builder, added option to close I-Gen reports and ability to schedule reports.
1.0	18-May-23	Brian Osesek	Modern Banking Platform v3.10 publication (initial client publication)
0.13	14-Aug-22	Krishna Davey	Added 3.9 New features/Enhancements for CTE1/CTE2
0.12	10-Jun-22	Krishna Davey	Added 3.8 New features/Enhancements for CTE1/CTE2
0.11	29-Jun-20	Anand Ramamurthy	Added the flows for the batch and real-time, reconciliation process.
0.10	29-Jun-20	Vigneshwar Prabhu / T Mani	Modified HDP to CDP.
0.09	10-Sep-19	Vigneshwar Prabhu	Updated Modularity and Extensibility. Added Real-time Event stream processing and Real-time vs Batch Reconciliation process.
0.08	18-Feb-19	Vigneshwar Prabhu	Added RAS Extension capabilities on client and geography-specific implementation concepts.
0.07	14-Feb-19	Vigneshwar Prabhu	Added Component Based Solution, I-Gen, and New tables addition. Removed QA section.

0.06	08-Feb-19	Vigneshwar Prabhu	Updated tabulation across all sections
0.05	04-Feb-19	Vigneshwar Prabhu	Updated Semantic layer, Security, and DR Framework
0.04	01-Feb-19	Vigneshwar Prabhu	Added UI Strategy / Review Comments
0.03	21-Jan-19	Vigneshwar Prabhu	Included Tabular columns, Atlas, and Ranger details
0.02	03-Jan-19	Vigneshwar Prabhu	Modified with Review comments
0.01	31-Dec-18	Vigneshwar Prabhu	Initial draft