Prometheus Financial Core Theory of Operation

> **Capital One's unique bank accounting approach as a basis for software design and implementation**

Acknowledgements

The heart of this theory is the work of Leonhardt de Waal, Distinguished Engineer, Capital One.

Table of Contents

List of figures

Capital One's Prometheus Financial Core theory of operation

The internal-to-Capital-One Prometheus program is an initiative is to develop a new Prometheus Financial Core and accompanying systems to replace costly legacy systems such as TS2, Shaw, Profile, and others and modernize Capital One's financial processing infrastructure.

The Prometheus program has three "pillars:" Enterprise Banking Platform, Prometheus Financial Core, and Enterprise Credit Bureau Reporting.

The Prometheus Financial Core is the focus of this theory of operation.

Summary of theory: separation of concern

Capital One's Prometheus Financial Core does everything that traditional banking cores do but is founded on an overarching principle not common in other banking cores: *separation of concern*.
For flexibility in design, implementation, and use of the system, the functions of the Prometheus Financial Core are distinct from each other. This separation of concern decomposes required functions into independent parts. In some traditional banking cores these functions are commingled, which complicates their design, implementation, and use.

This theory of operation describes how the Prometheus Financial Core's unique characteristics are based on separation of concern. These theoretical underpinnings make the Capital One Prometheus Financial Core different from other traditional banking cores:

- It is a programming platform.
- It maintains separation of customers, smart contracts, and accounts.
- It is independent of the time constraints of traditional banking cores.

How to read this theory

This theory is in two general parts.

1. The body of the theory discusses how the Capital One Prometheus Financial Core differs from traditional banking cores.
2. Persons with little or no familiarity with bank accounting should see the Basics of accounting and Learning materials. The theory assumes this fundamental knowledge as a starting point:

Programming platform

In general, most traditional banking cores are *hard-coded*: the definition of a financial product is mixed with the computer programming that processes customer transactions based on those products. Because of this commingling of product with program, new product definitions and changes to existing products can be expensive and difficult.

Capital One Prometheus Financial Core separation of concern
By contrast, Capital One's Prometheus Financial Core separates the programming from the definition of a product that it processes for each customer. The Prometheus Financial Core is the machinery to build financial products of any complexity from simpler building blocks. With the modern programming paradigm of a *domain-specific languages* (DSL), designed for Capital One's line of products, the programming aspects of banking are independent of the products themselves. New products can be defined and existing products can be modified with relative ease. It is unlikely that traditional banking cores have a corollary to this concept of a DSL.

There are several high-level parts of the Prometheus Financial Core:

- Financial instrument
- Smart contract

The relationships of the parts are described in the following table.

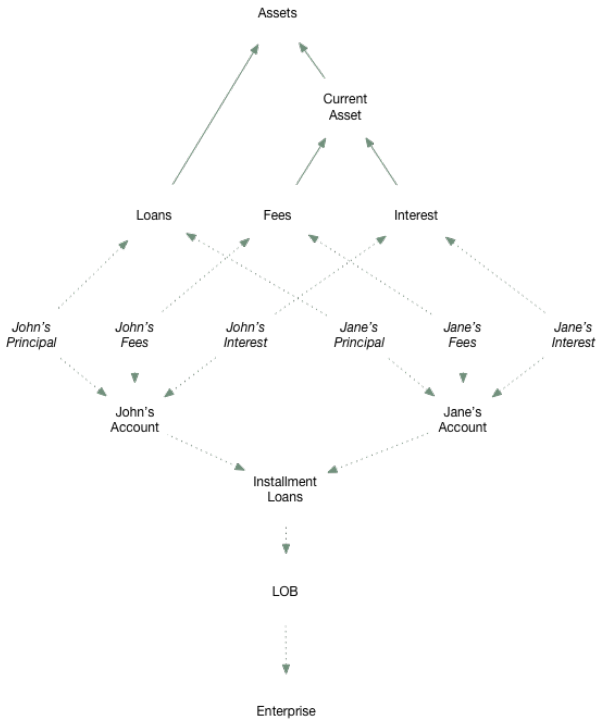
Function	Description
Financial instrument	A *financial instrument* encapsulates the structure and rules of a financial product, such as a checking account or installment loan. • The instrument acts as a template for a specific customer's *smart contract*. • Given a sequence of transactions, an instrument's definitions and program govern the evolution of the state of the customer's smart contract.

Smart contract	A *smart contract* is a unique instantiation of an instrument with terms tied to specific, unique accounting *buckets*. Buckets are "holding spots for calculations and other interpretative functions of the instrument. • The creation of a smart contract based on an instrument is called *provisioning*. • One customer's smart contract is independent of other smart contracts based on the same instrument. • The Prometheus Financial Core's domain-specific language (DSL) determines the effect of the transactions against buckets defined by the instrument from which the smart contract was instantiated.

From theory to programming in practice: financial instruments

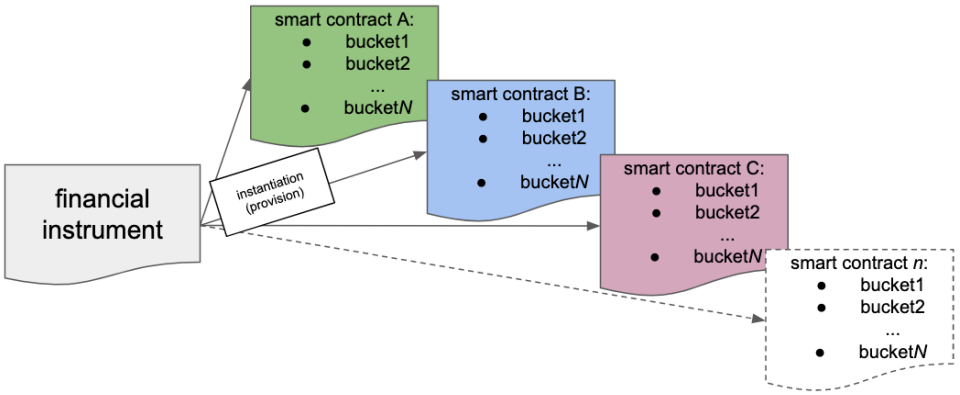
For information about how financial instruments are implemented based on this theory, see the developer guide *Creating Financial Instruments with the Prometheus Financial Core*.

Separation of customers, smart contracts, and accounts

In a traditional banking core, customer financial accounts are often commingled with customer transactions. Posting transactions to a customer's accounts is a lengthy, linear process. For a single transaction, *all* customer accounts must be traversed to identify the affected customer's account. To show balances to customers, transactions not yet posted are sometimes "papered over" so that the posting appears to have already taken place. Later, at actual posting, the "papering over" must be removed. **Capital One Prometheus Financial Core separation of concern** By contrast, Capital One's Prometheus Financial Core separates customers and their accounts from the transactions that affect them. Customer information is maintained in a system external to the Prometheus Financial Core.	***Representation of traditional chart of accounts*** 

When a transaction is received, the product-distinct domain-specific language described in Programming platform correlates in real time against the customer's accounts. Balances are thus current. There is no need to "paper over" transactions. Smart contracts and accounts can be associated when necessary; for example, a customer's interest-free loan via a merchant who must pay a fee to the bank for servicing the loan.

Relationship: instrument, smart contract, bucket



Customer/smart contract/account independence has several benefits:

- It increases the scalability of the system.
- It allows smart contracts to evolve quickly over account life.
- Contract changes can be applied retroactively, as discussed in Independence from time constraints.
- Regulatory changes can be more easily accommodated.

Independence from time constraints

Traditional banking cores are severely constrained by wall-clock or calendar time. They must "stop the clock" to exclude new transactions while they formally post transactions already accumulated in a bounded time period. This time-bound posting can frustrate customers and can hamper bank efficiency. Adjustments to past transactions are difficult because past transactions must often be "counter-transacted" by multiple correcting transactions.

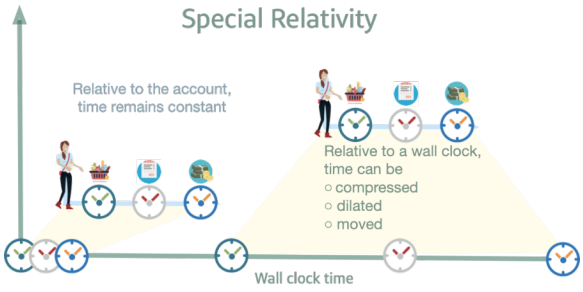
Capital One Prometheus Financial Core separation of concern
By contrast, Capital One's Prometheus Financial Core is time-independent. To avoid traditional banking time constraints, it decouples business logic from the clock on the wall. The posting of transactions to accounts is in real time without delay.
The system uses end-of-day (EOD) markers to mark the passage of time. These markers trigger the processing of Capital One's own transactions on customers' accounts, such as interest due or fees assessed. The most important timestamps are *posted or system time* (when the transaction was posted to the system) and *effective time* (when the economic value happened). Account state evolves as if all transactions were ordered by effective time.
The system creates events for postable transactions in the order in which transactions are observed, with some conditions:

- Future-dated transactions (that is, transactions that are out of proper sequence) are suspended until the future is "now".
- Intraday transaction order is not important for EOD processing.

Time-independence allows for *auditable* retroactive corrections:

- Unlimited changes can be made to past transactions.
- Retroactive transactions can be inserts, deletions, or modifications.
- Future-dated or ill-defined transactions can be suspended for later processing.
- Intervening transactions from the point of a retroactive transaction to the current state are automatically "replayed".

Prometheus Financial Core's unique idea of time

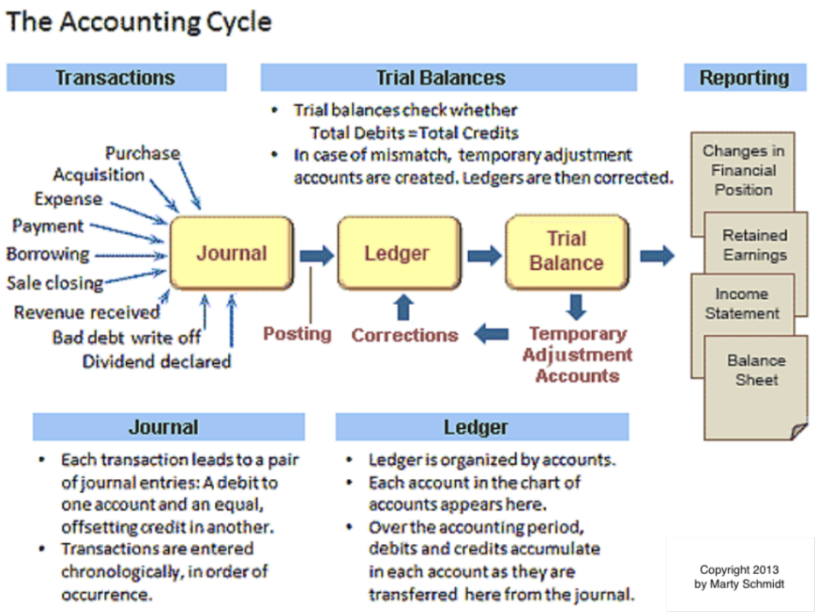


References

Basics of Accounting

Bank accounting is a large topic that can take a great deal of experience to fully understand. As a basis for understanding Capital One's Prometheus Financial Core, you should have a rudimentary knowledge of the following features of accounting in general.

Traditional accounting functions



Learning materials

1. Financial transaction. What is the transaction approach and balance sheet approach to measuring net income?
2. General Journal and General Ledger. What is the difference between entries in a general journal versus a general ledger?
3. Double entry accounting. What is double-entry bookkeeping?
4. Interest accrual. What is accrued interest?
5. Suspense account. What is a suspense account?
6. Bank reconciliation. What is a bank reconciliation?

Revision History

Prometheus Financial Core: Theory of Operation

Date	Description
2019-12-06	• Added link to developer guide *Creating Financial Instruments with the Prometheus Financial Core*. • Updated graphic Relationship: instrument, smart contract, bucket.
2019-09-26	Formal publication after team review.
2019-09-12	Complete rewrite. Draft for review by Prometheus Engineering Group, SF.

2019-08-01	Shared for viewing by anyone at Capital One.
2019-07-15	First created by Alex Lange.