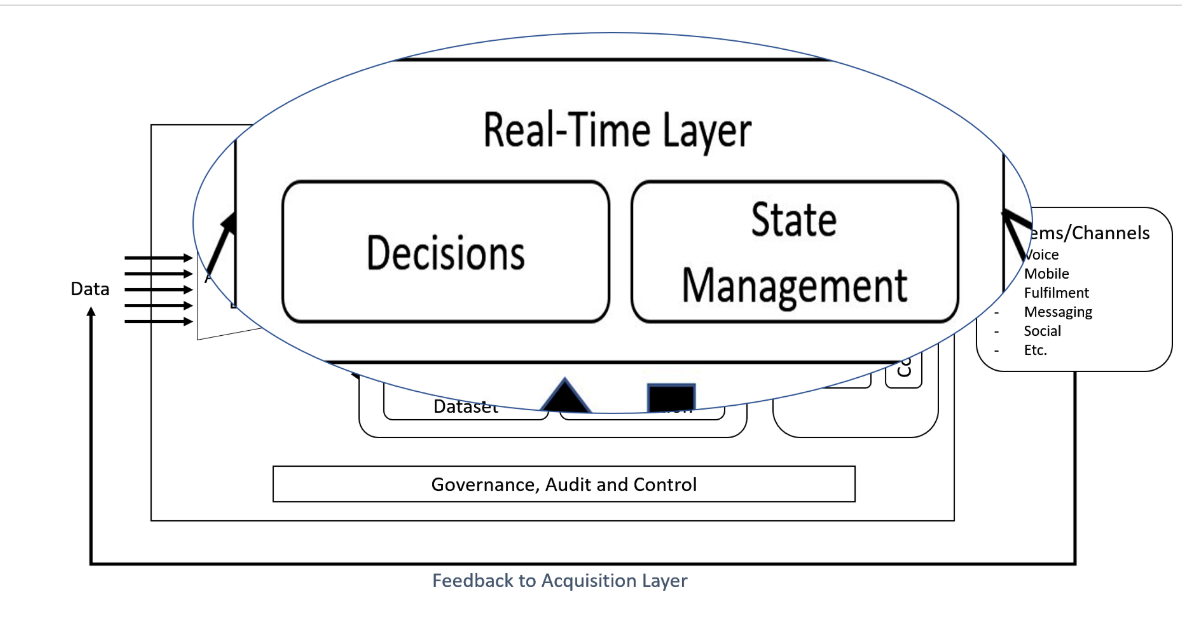
**Ab Initio Lambda - Real-Time Layer**

In this fourth article we continue to dig deeper into the [Ab Initio Lambda Architecture](https://www.linkedin.com/pulse/ab-initio-lambda-overview-chris-day-/) and look at the Real-Time Layer.

In previous posts we looked at the [Acquisition Layer](https://www.linkedin.com/pulse/ab-initio-lambda-acquisition-layer-chris-day-/) and the [Batch Layer](https://www.linkedin.com/pulse/ab-initio-lambda-batch-layer-chris-day-/).

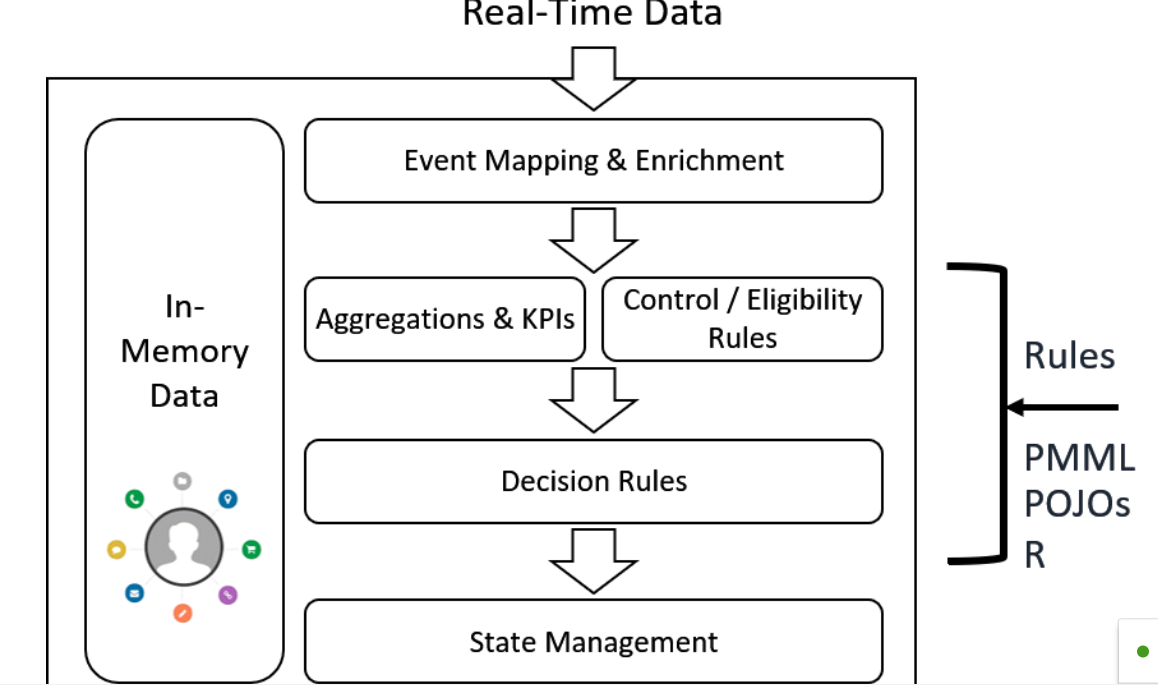


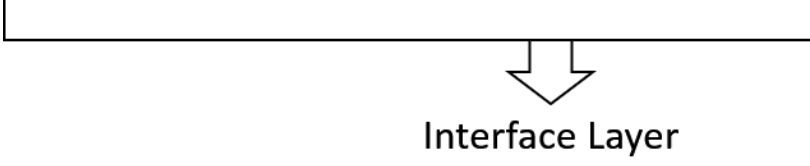
The Real-Time Layer's evolution is dramatic over the last few years as the digital revolution heads into its adolescence phase. Initially, the Real-Time Layer would focus on looking at recent data, to compensate for newly arrived data post the last batch run. Processing of the recent data would be at the lowest level of granularity to achieve the smallest latencies.

real-time view = function(real-time view, new data)

## How does this look today?

Today, however, the business and our customers drive the need for rapid feedback and sophisticated decisions to enable the modern digital transformation.





We still need to facilitate the real-time views as new data arrives using incremental computation as opposed to the recomputation of the Batch Layer. However, to reduce the latency even further from both a business and technology perspective, advanced data processing needs to happen in-line with the data.

Only then can we address some of the business and technical challenges;

* Sophisticated decisions to trigger the right business actions, such as Product offering, Fraud Alerts, Service Interaction and Claim/Loan approval etc.
* Low Latency or lower than our competition
* Platform is Business Critical from the get-go
* Rapidly changing Business Rules through a much shortened life-cycle
* Maintain in real-time the customer profiles, aggregations, KPI's and state
* Provide fast access now and can scale to faster
* 360-degree view of customer and business activity

### Event Mapping & Enrichment

The incremental computations that are part of the original Lambda architecture remain, but this has evolved, hence this precursory stage.

Once the data starts to arrive, we need to begin mapping those events and enriching the information quickly. (This dovetails nicely in the Collect, Detect, Act paradigm of the event and state-based processing contained within any Customer Interaction Platform).

We enrich data upstream to reduce redundant computation and storage which is generally a good practice. Also, we don’t have the luxury of latency as we do in the Batch Layer, so In-memory data processing rules the day for real-time interactions with your customers.

### More on “Collect, Detect, Act”

At a high level, complex event processing fits a strategy of “Collect, Detect, Act.”

* Simple event processing — The processing of individual events in isolation.
* Complex event processing — The processing of aggregates of events.

Which can involve one or both of the following:

* Aggregating data (sum, distinct count, and so on)
* Updating state and making state transitions

### Aggregations & KPI’s along with Control/Eligibility Rules

As we handle the data outside of processed batch cycles, we need windowed aggregation at scale and the management system for those business rules.

Which drives the following requirements

* Shared variables for tasks like “Maintain the balance for each account” or “Track whether an alert has been issued”.
* Windowed aggregation functions to keep the record from the end of the last batch to now.
* A similar ability for the compound metrics that form KPI’s, in essence, a holistic view across the real-time ingestion.
* Empower the business too so our processing can adapt to new business rules confidently, effectively and with the utmost efficacy.

Of course, we need this functionality at scale, across our geographically dispersed landscape with low-latency to keep ahead of the competition.

### Decision Rules

To enable the business to be a champion of change, we require a rapid evolution of business rules throughout the Agile delivery lifecycle. That means optimisation of the delivery process, we need agility, but auditable control and accountability.

This requirement drives the need for a intuitive rules architecture to handle those decision trees.

### State Management

The original Lambda architecture and its basic needs have now matured for real-time interactions, and that means state management.

We combine the classical functional requirements of the real-time view, basically aggregations mirroring the Batch Layer functionality to fill the gaps, with the point in time process metrics.

A simple use-case could be a block on a customer credit card, opt-in to a time-boxed promotion, operational coverage of a network, all measures of a state within a business process.

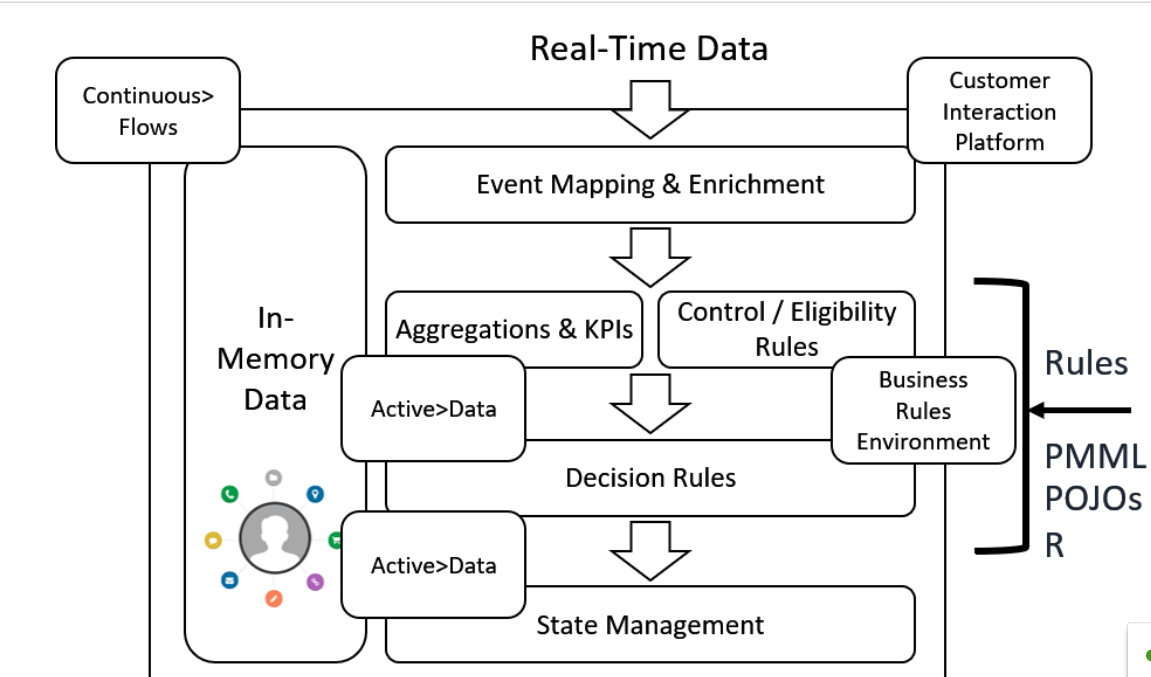
Together, we can maintain in real-time the customer profiles, aggregations, KPI’s and process state which is available quickly and coherently to give that 360-degree view of your customer and business activity.

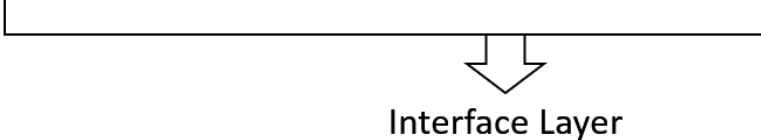
## How does this look in Ab Initio?

With a combination of Continuous>Flows, Active>Data Cluster and the Business Rules Environment, we support the rapid feedback and sophisticated decisions required by modern digital transformations.

At a technical level, this means data and event processing must be low-latency, in-memory and materially scalable.

The materiality comes from supporting Agile business development within the rules management system, coupled with a replication, distribution and resilience that occurs out of the box with Continuous>Flows and the Active>Data Cluster. All critical components that affect the bottom line and improve the top line, hence materially scalable.





### Acquire>It publishing to Continuous>Flows.

Acquire>It handles the acquisition of Real-Time data>It, although this time we are publishing to Continuous>Flows, whether that input comes from Social Media, Kestrel, Kafka, JMS, Amazon Kinesis, Storm, Flume, Sabre, Tickers and of course those trickle feeds from any other operational source. Continuous>Flows gives us the consistency, conformity and extensibility that we need for entry into the Real-Time layer.

Of course, Continuous>Flows doesn’t care where it resides, and that’s been the case yesterday, today and tomorrow. It is genuinely heterogeneous and handles all the real-world scenarios, whether that’s the complexity of FpML or the intricacy of ASN.1 or whatever evolving data definition languages come along.

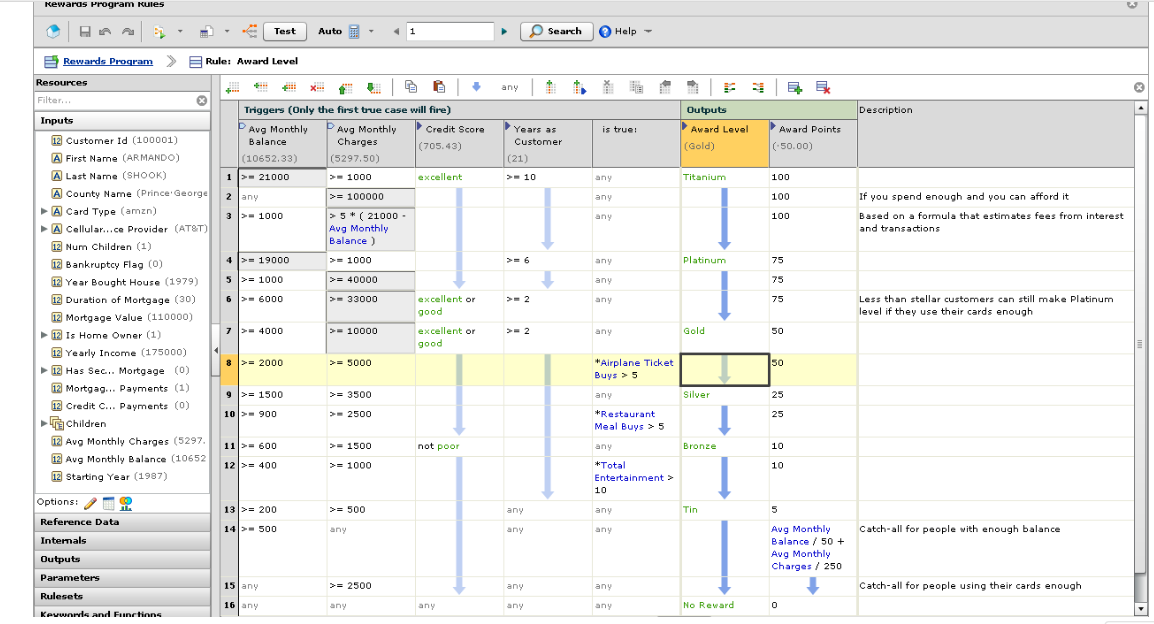
Continuous>Flows runs anywhere.

### Business Rules Environment

Supporting sophisticated business rules that are rapidly changing means we need analyst-driven application development.That translates into a Business Rules Management System and a Business Rules Engine.

That environment should also adhere to the [Business Rules Manifesto](http://www.businessrulesgroup.org/brmanifesto.htm).

Ab Initio’s Business Rules Environment (BRE) can preserve business knowledge, dramatically increase speed to production, enable automation of tasks that could not previously be automated, reduce programming team size by 70% or more, and provide businesses with magnitudinal annual savings. Ab Initio’s BRE makes it possible for business experts to interact directly with the data, building and testing rules in an intuitive, spreadsheet-like environment. Using the BRE, analysts can develop and update their rules without ever involving software developers in the process, and they can run simulations and “what-if” scenarios to test rules and explore options. The BRE can reduce the time it takes to get rules into production by as much as 95%.



### PMML Models

The Predictive Model Markup Language provides a way for analytic applications to describe and exchange predictive models produced by data mining and machine learning algorithms.

Ab Initio can translate PMML to DML using the pmml-to-dml utility, and then use this DML in the BRE or any Ab Initio product that can consume DML.

This translation means your Data Scientists and analysts can continue to use products such as KNIME, R and Spark to develop the models and then leverage the power of Ab Initio for deployment at scale. The list of PMML powered vendors is [here](http://dmg.org/pmml/products.html).

### Active>Data Cluster

Ab Initio has been handling shared variable collections for a while, but Active>Data truly couples the variables with processing. This Active Logic is pushed down into the in-memory data store – making your data genuinely active, using the familiar DML language.

Think about the current in-memory database world, the spin-offs of Apache Ignite and the like. The model is “pull out and process”, that’s the old world. The new world keeps the processing with the data, and have that data distributed, anywhere and in-memory.

* Replicated
* Distributed
* Resilient, 24/7 operation
* Service-based computation
* In-memory data store
* DML driven

In the field, some Ab Initio customers are running with over 400 million customer accounts held within the Active>Data Cluster. Just think of the new levels of service with all the channel information held in-memory, driving a better response time and improved service.

## Want to know more?

If you want a full featured demo, then engage with your Ab Initio Account Manager. If you are in a knowledge repression culture, then drop me a line and I’ll help you connect.

In the next article, we focus on the Interface Layer within the [Ab Initio Lambda Architecture](https://www.linkedin.com/pulse/ab-initio-lambda-overview-chris-day-/?lipi=urn%3Ali%3Apage%3Ad_flagship3_pulse_read%3Bcta0o1sLRTqniI8bGzmSRA%3D%3D).