**Ab Initio Lambda - Interface Layer**

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

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

All the hard work in data acquisition, batch processing and real-time ingestion are precursory steps in turning data into actionable intelligence; *The query*.

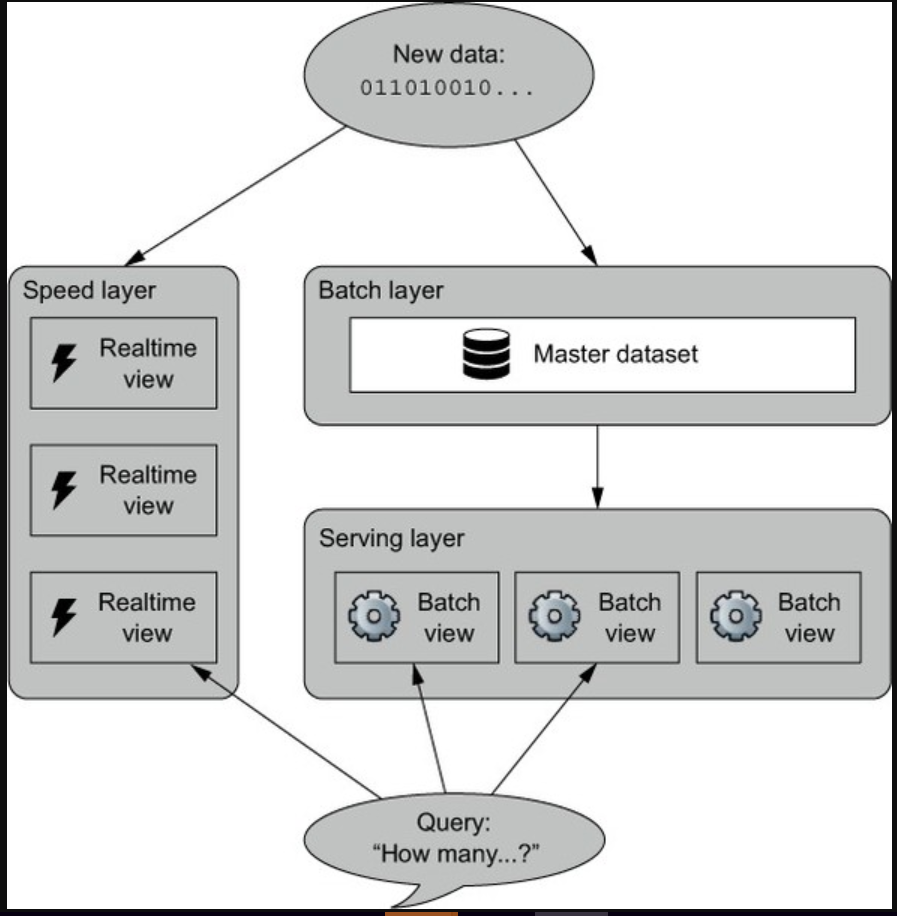
As Nathan simplifies, The Lambda Architecture, by these three equations:

batch view = function(all data)

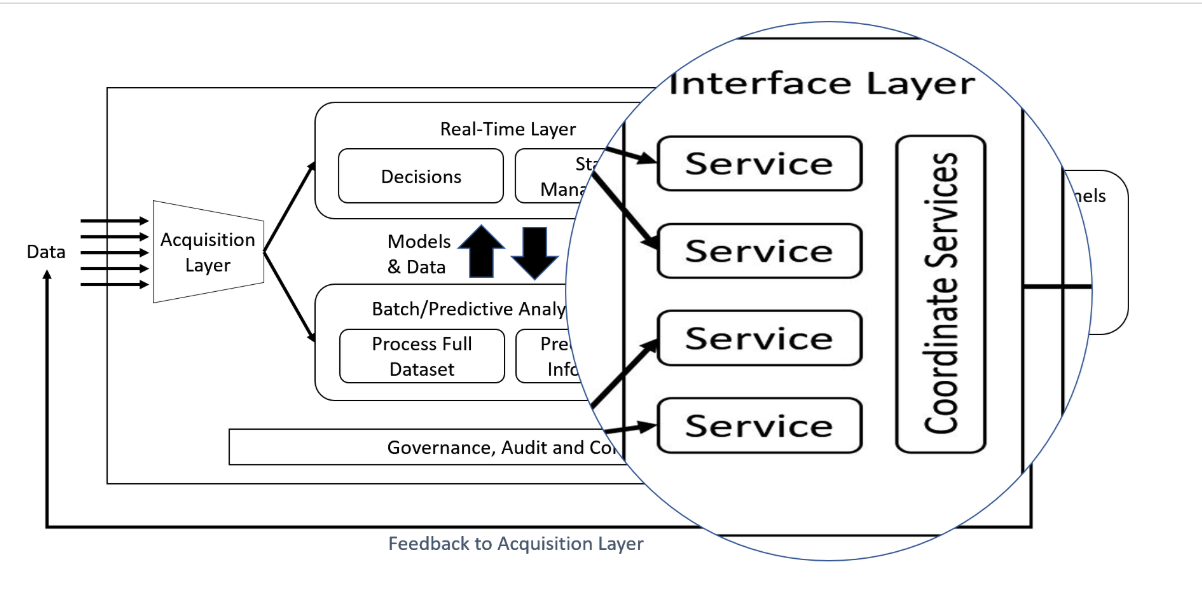
real-time view = function(real-time view, new data)

query = function(batch view. real-time view)

The Interface Layer is all about the *query*.



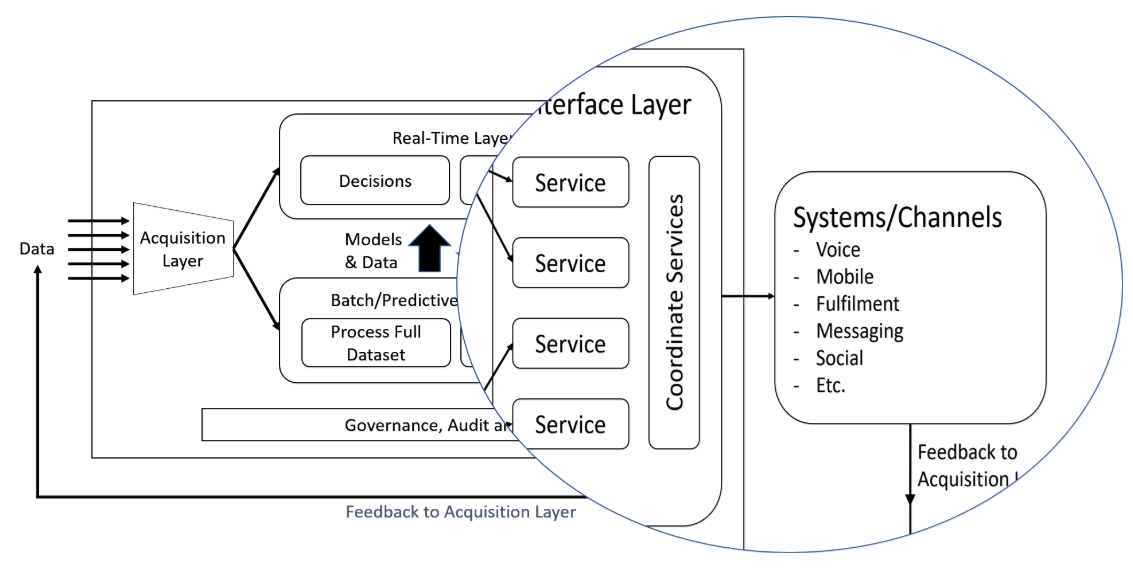
The query result set requires coordination of services interacting with the batch and real-time layers to provide consistency and accuracy.



Let’s put this in context with some use-cases for the Interface Layer;

* Enables interaction with channels and other external systems
* ‘Next Best Offer’ provided to a Mobile App
* Trigger a service message via SMS
* Product fulfilment
* Transaction (e.g. a loan or claim) completion
* Communication policy and personalisation can be applied
* Each channel/system generates event information back into the Acquire Layer
* 360-degree view of business

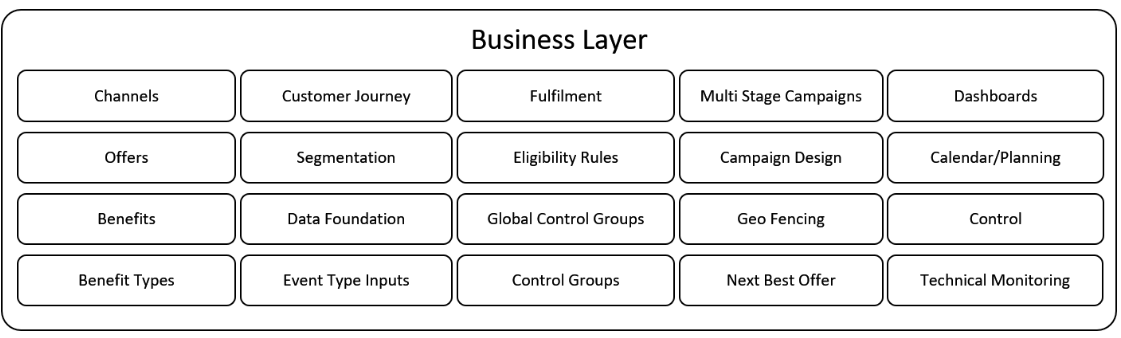
I'm sure you could expand this list significantly with your business drivers too.



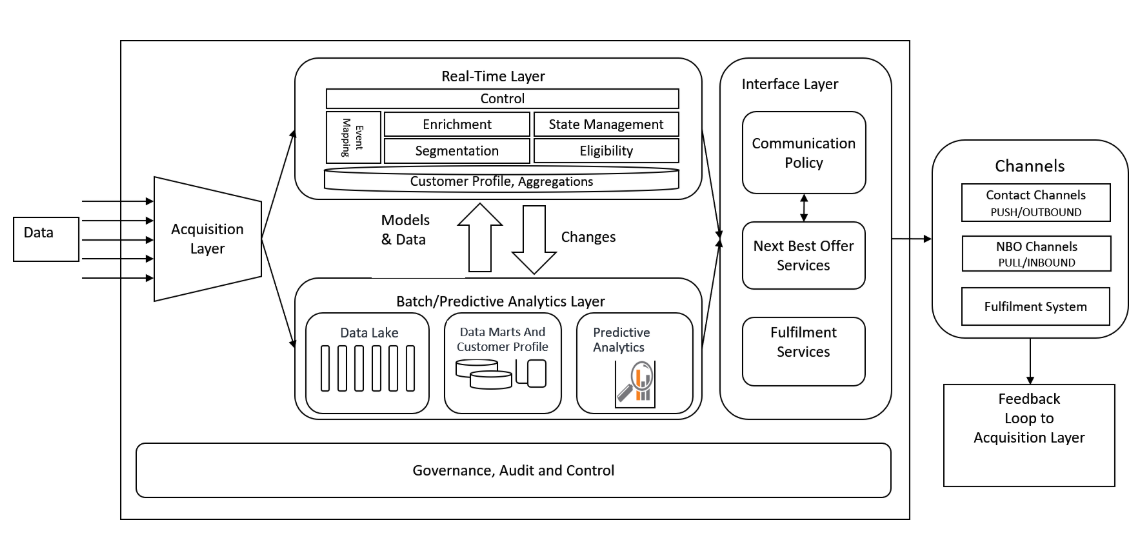
## A Customer Interaction Platform

A possible implementation of the Ab Initio Lambda Architecture would be a Customer Interaction Platform.

A typical Business Layer might consist of the following functional areas;



And mapping onto the Ab Initio Lambda Architecture would produce something like this;



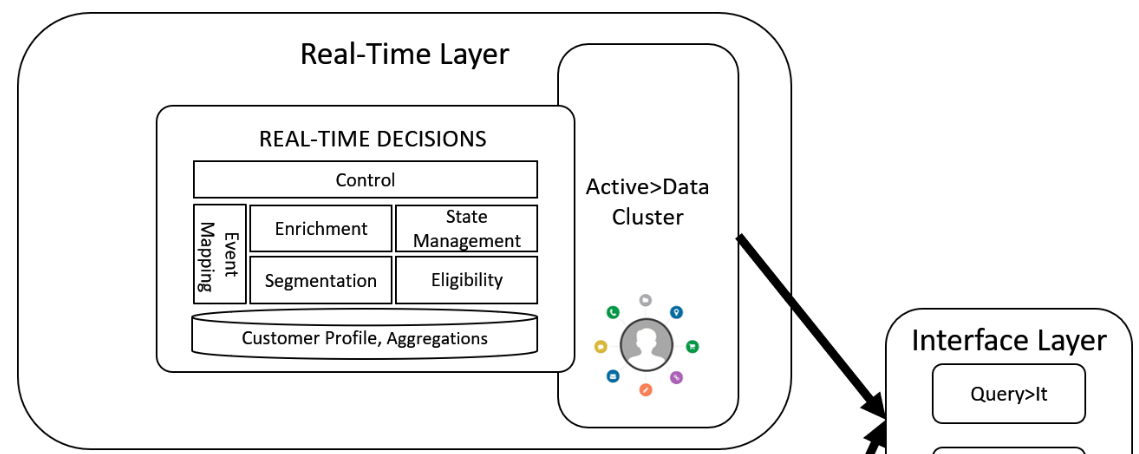
Just decide the timeliness of the information to fulfil a business function and then place that function in a Layer.

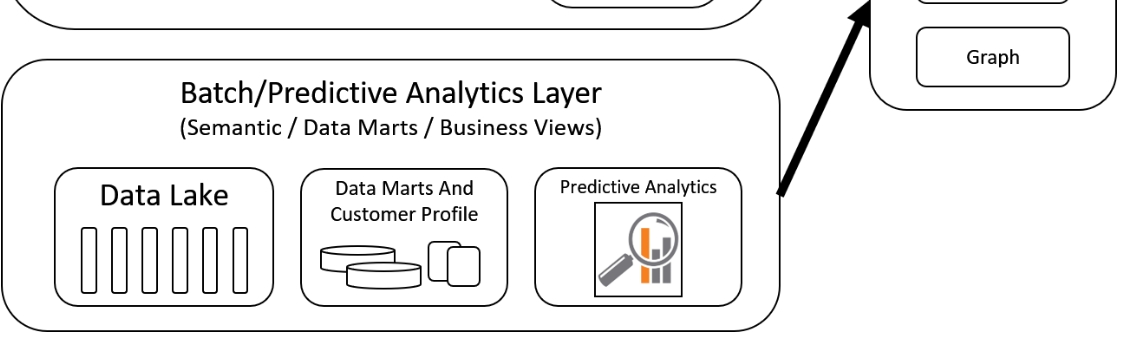
## How does this look like in Ab Initio?

The Interface Layer, to be successful needs to be able to integrate with anything, whether that be so-called legacy systems, the variety of modern digital applications we have today or whatever will come around the corner tomorrow.

The good news is that the Co>Operating System has been providing that piece of integration for decades, and always with an eye on the future while taking care of today. So, if in doubt, build a graph.

Of course, Query>It can interact with the Active>Data cluster to obtain that real-time in-memory data, along with any of the data stores on the Batch Layer.





There’s nothing beyond those standard enterprise integration patterns within the Interface Layer. In-memory active data stores can be accessed and combined with data anywhere; we have fulfilled Nathan’s Lambda function the query();

query = function(batch view. real-time view)

## 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 Governance, Audit and Control within the [Ab Initio Lambda Architecture](https://www.linkedin.com/pulse/ab-initio-lambda-overview-chris-day-/).