New Discount Samples

# Overview

We will cover the main picture of the Dynamics retail discount extensibility and then we will explain the two new discount samples.

# Pricing engine main picture

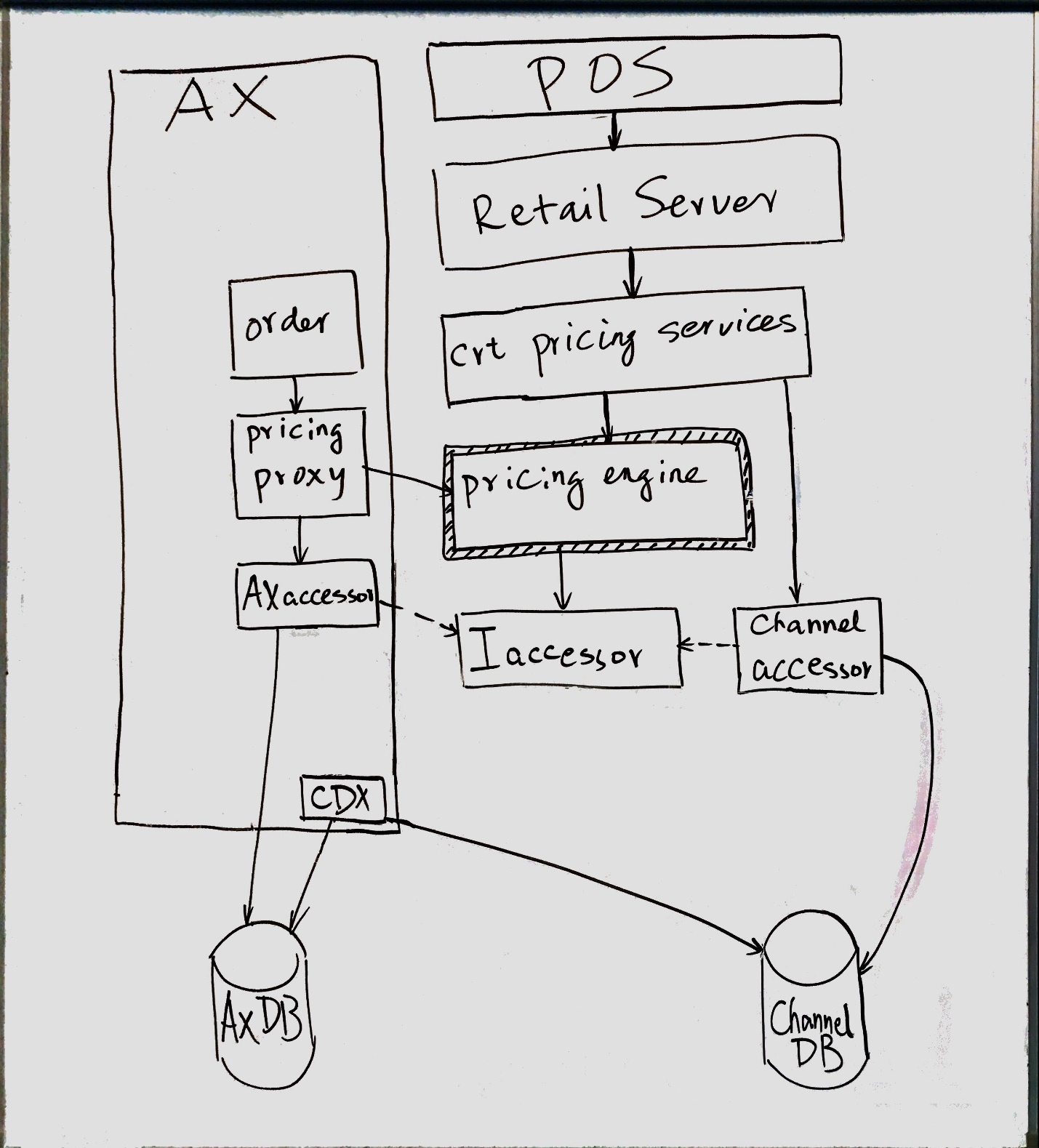
## One pricing engine serves all retail channels

All retail channels: store POS, call center orders and store front orders share the same pricing engine. Call center is part of AX, while POS and store front utilize commerce runtime framework via retail server, so we have two pricing engine invocation patterns.

1. Retail server calls pricing services via commence runtime framework, which in turn calls pricing engine.
2. AX calls pricing engine directly, as AX does not understand commerce runtime.

In addition, to accommodate the difference in data access, we have IPricingDataAccessor. When retail server calls pricing engine via pricing services, pricing services would construct channel version of the data accessor, while AX would construct its own.

The following diagram illustrates the main flow and difference between AX and channel side implementations.



## Extensibility.

Say, we are creating a new discount type.

### First, the schema, the form and CDX

The number in parenthesis corresponds the number in the diagram.

1. [AX schema extension (5)] new table(s)
2. [AX form extension (6)] RetailPeriodicDiscount form extension
3. [Channel database schema (8)] new table(s), view(s) and/or stored procedure(s).
4. [CDX extension (7)] [manual setup](https://ax.help.dynamics.com/en/wiki/add-customer-preference-data-to-a-channel-database/).

### Secondly, core pricing engine

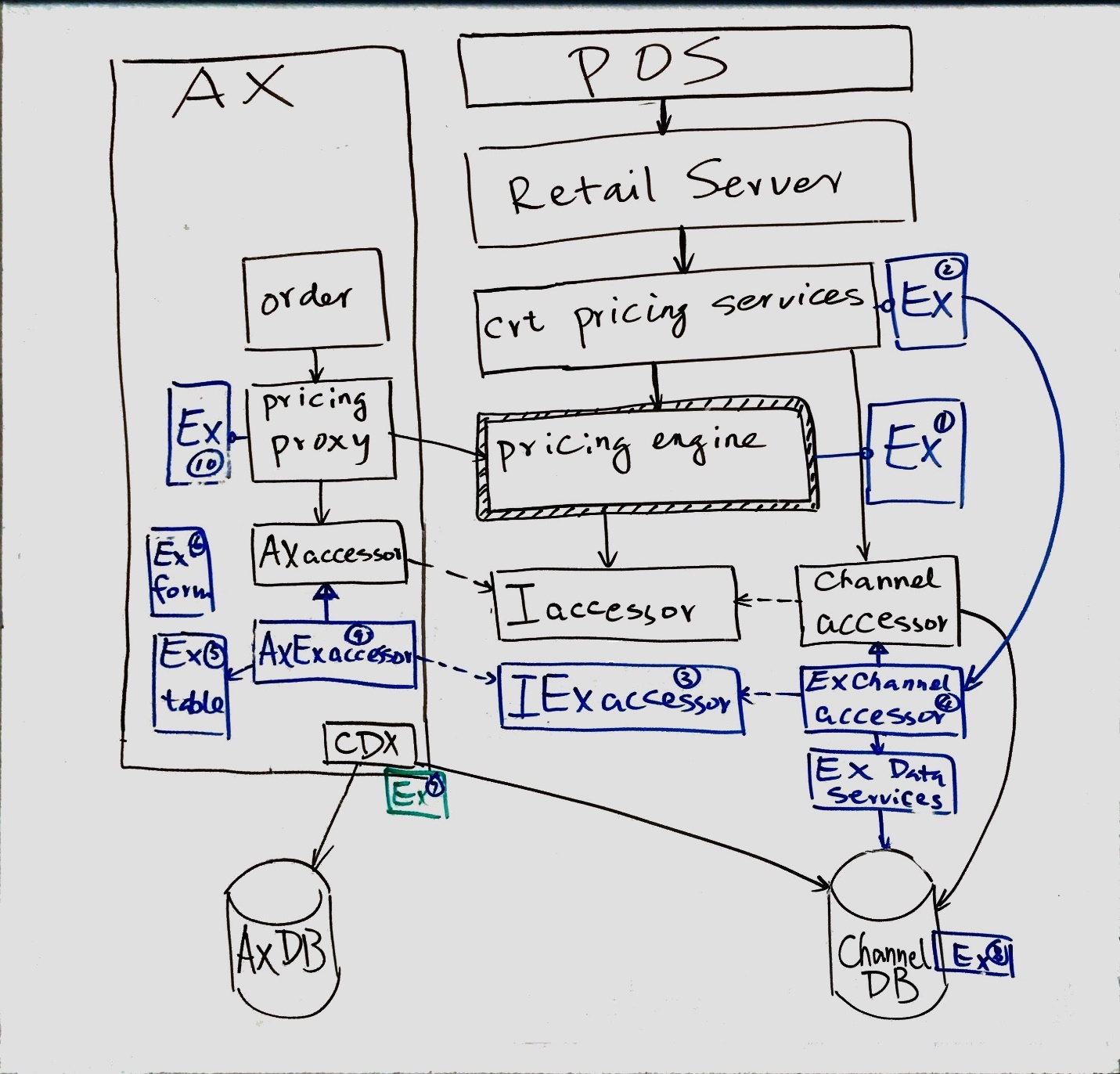
1. [pricing engine extension (1)] Create a new discount type.
2. [pricing engine extension (1)] [Register the discount type in retail discount store](https://blogs.msdn.microsoft.com/retaillife/2017/02/17/dynamics-retail-discount-extensibility-register-a-new-discount-type/) that loads discount data from database and instantiates discount types
3. [pricing data accessor interface extension (3)]

### Thirdly, channel stack

1. [pricing services extension (2)] Register the retail discount store extension at the top of commerce runtime pricing services extension. Instantiate the extension data accessor and pass it to the pricing engine.
2. [pricing data accessor channel extension (4)] Derive from PricingDataServiceManager and implement the extension interface.

### Lastly, the AX stack

1. [pricing data accessor AX extension (9)] derive from RetailPricingDataManager and RetailPricingDataManagerSimulator
2. [AX pricing proxy (10)] Equivalent to channel side commerce runtime pricing services extension: register the retail discount store extension. Instantiate the extension AX data accessor and pass it to the pricing engine.



# New discount samples

## Discount offer line filter with validation period

This is the simplest extension. It’s to customize existing discount offer (a.k.a. simple discount) to enable discount line level validation period (think happy hour).

There is no change beyond validation period filtering. This is important as in general, we recommend a new discount type if there is a behavior change that affects how discount is calculated.

## Discount offer with amount cap

### Project Extensions.PricingDataServicesSample

It covers 2 parts. You may consider split them into 2 projects:

1. Data model: class DiscountAmountCap
2. Channel side data services

### Project Extensions.PricingEngineSample

This is the core of customization

1. Interface of data accessor extension: IPricingDataAccessorSample.
2. New discount type: AmountCapDiscount.cs
3. Customized retail discount store: CustomizedRetailDiscountStore

### Project Extensions.PricingServicesSample

Commerce runtime pricing service override

### Project Extensions.PricingEngine.SampleTests

Sample test is covered in PricingSampleTests document.

# References

[Blog discussions about retail discounts](https://blogs.msdn.microsoft.com/retaillife/2017/01/07/retail-discount-concurrency-control-pricing-zone/)