beacon

ACCELERATE TECHNOLOGY

Beacon White Papers

Beacon's Data Warehouse and Bi-Temporal Data Model

BEACON PLATFORM, Inc. 5 Hanover Square, 20th Floor, Suite 2001, New York NY 10004

Introduction

A key part of the Beacon platform stack is the integrated data warehouse: an object-oriented database – the Beacon Object Database – that forms a data fabric that all Beacon analytics and applications can access.

In addition to storing data in a flexible and uniform way, Beacon's data models allow for easy access to versioned data – that is,
as data changes over time, applications access the most recent data in most cases and are able to access old versions when required, with a clear audit trail of changes.

We call this versioned data structure "bi-temporal" data because there are two times involved: the time the data is associated with (the "as-of" time), and the time the data was actually updated (the "entry time"). For example, a calculation might generate data corresponding to a run for a particular business date, like the end of a month, but that calculation might be run several times: once on the business date, once a few days later, and once a month later. In that example, the "as-of" time is the business date for each of the updates, but the "entry time" is different for each of the updates. All versions of the calculated data are available in Beacon.

This time-based data versioning fits nicely into Beacon's object models as well. At the bottom of the hierarchy of objects in Beacon's dependency graph, there are environment settings that control parameters, such as the current date and time. Beacon makes it easy to override those environment settings to switch to any past date and time, and Beacon's bi-temporal data automatically shifts to use the version appropriate for that environment time. This makes it easy to account for and drill into the impact of data changes on important metrics for a business, such as profit and loss calculations. For example, Beacon clients can "time travel" to run reports as of any market date and position date, with full transparency into all inputs and updates.

Beacon clients can "time travel" to run reports as of any market date and position date, with full transparency into all inputs and updates.

About Beacon Platform, Inc.

Beacon Platform, Inc. was founded in 2014 by the front office trading and risk technologists who created SecDB, Athena and Quartz at Goldman Sachs, JP Morgan and Bank of America Merrill Lynch. Leveraging the experience and lessons learned during their extensive careers at major investment banks, Beacon's founders have created the only third-party solution on the market that delivers a cloud-based, end-to-end development and production platform. With Beacon's open architecture, transparent source code, and automated infrastructure solutions, we give financial and quantitative developers the tools they need so that they can focus on the business rather than plumbing and process. And for business users, we deliver fully integrated applications for analytics, pricing, risk management, and more. Beacon has over 60 employees with offices in the United States, UK, Germany, and Japan.

www.beacon.io

Risk Awards 2017 Winner — Beacon Fintech start-up of the year

RiskTech 100 2018 Rising Star

Risk.net Market Technology Awards 2018 — Beacon Platform Pricing and analytics Structured products/ cross-asset

In-Memory Namespaces and the Beacon Object Database

Beacon supports arbitrary data sources, such as popular relational databases, cloud vendor database products, and big data repositories such as Hadoop. Clients can choose to configure their data infrastructure as they find appropriate.

However, at the core of Beacon's data model is the Beacon Object Database. This data fabric acts like a distributed file system, where objects are stored by name in a structure that supports directories and subdirectories. Under the hood, the Beacon Object Database uses MongoDB, but developers never access the Mongo database with direct MongoDB API calls – instead they use the Beacon Object Database API. Adding a layer of abstraction to data access allows Beacon clients to use many different underlying database technologies.

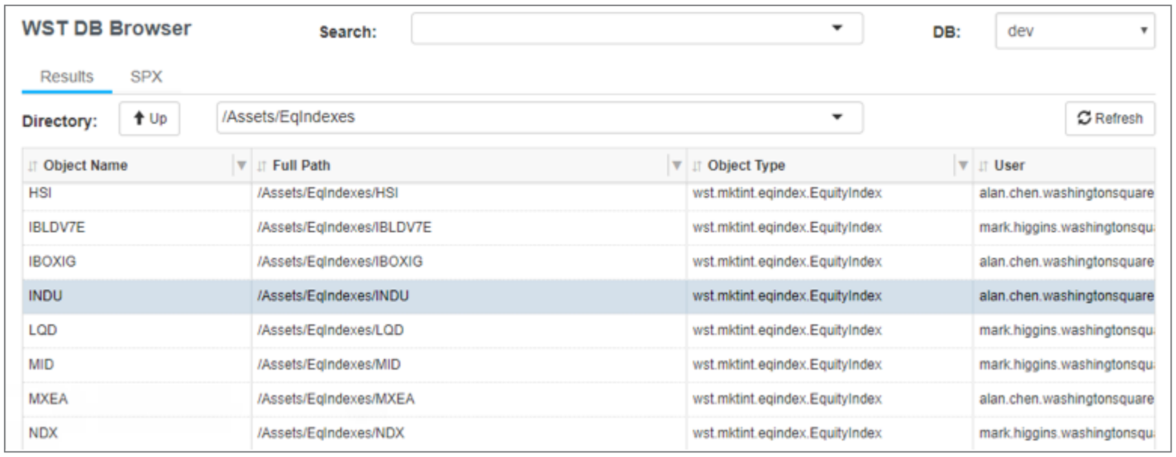
Adding a layer of abstraction to data access allows Beacon clients to use many different underlying database technologies.

Thanks to that abstraction, Beacon provides useful logical architecture on top of the raw database functionality. For example, the Beacon Object Database supports "symlinks", where a subdirectory in one Beacon Object Database can link to a subdirectory in a different physical Beacon Object Database. Symlinks let us share data across physical databases inside a client environment and also allow Beacon to share common data with all its clients.

Client Case Study: Month-End Close

One of Beacon's clients manages oil inventories around the world. The client was using a legacy CTRM system to report P&L and market exposure related to their inventory position. Their biggest pain point was managing end-of-month P&L. Due to outdated terminal software, they could only receive actualized end-of-month inventory levels three days after month end. Once they received the inventory levels, they could not simply update them and re-close the prior month because their system could not filter out any changes that happened after month end, such as new hedges, actualized secondary costs, historical market data corrections from vendors, and changes in mark-to-market curves. As a workaround, the client had to complete each month's close on the last day of the month, export all end-of-month closing data, and then spend days doctoring it outside of the legacy CTRM system. Needless to say, such a manual process often resulted in human errors and incorrect monthly reports.

After implementing Beacon, the client was able to take advantage of the bi-temporal data model and gained the ability to update inventory levels as of the last day of the prior month. They can now enter all "as of" information as it becomes available and safely re-close the prior month at any time. They no longer need to worry about post month-end changes slipping through the cracks and producing erroneous reports. By implementing Beacon, they have also saved several man-days of manual work per month and made end-of-month reports 100% accurate.



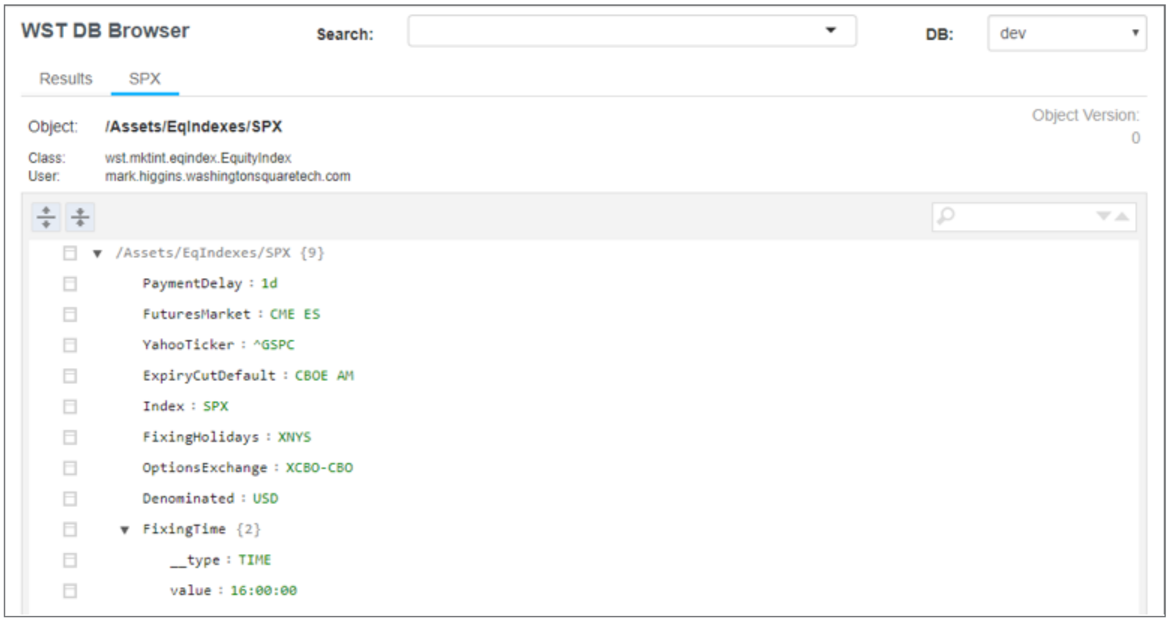
WST DB Browser Search: DB: dev

Results SPX

Directory: Up /Assets/EqIndexes Refresh

Object Name	Full Path	Object Type	User
HSI	/Assets/EqIndexes/HSI	wst.mktint.eqindex.EquityIndex	alan.chen.washingtonsquare
IBLDV7E	/Assets/EqIndexes/IBLDV7E	wst.mktint.eqindex.EquityIndex	mark.higgins.washingtonsqu
IBOXIG	/Assets/EqIndexes/IBOXIG	wst.mktint.eqindex.EquityIndex	alan.chen.washingtonsquare
INDU	/Assets/EqIndexes/INDU	wst.mktint.eqindex.EquityIndex	alan.chen.washingtonsquare
LQD	/Assets/EqIndexes/LQD	wst.mktint.eqindex.EquityIndex	mark.higgins.washingtonsqu
MID	/Assets/EqIndexes/MID	wst.mktint.eqindex.EquityIndex	mark.higgins.washingtonsqu
MXEA	/Assets/EqIndexes/MXEA	wst.mktint.eqindex.EquityIndex	alan.chen.washingtonsquare
NDX	/Assets/EqIndexes/NDX	wst.mktint.eqindex.EquityIndex	mark.higgins.washingtonsqu

Figure 1: View of the Beacon Object Database. This shows the contents of the folder /Assets/EqIndexes, which holds objects representing various equity indexes.



WST DB Browser Search: DB: dev

Results SPX

Object: **/Assets/EqIndexes/SPX** Object Version: 0

Class: wst.mktint.eqindex.EquityIndex
User: mark.higgins.washingtonsquaretech.com

```
▾ /Assets/EqIndexes/SPX {9}
    PaymentDelay : 1d
    FuturesMarket : CME ES
    YahooTicker : ^GSPC
    ExpiryCutDefault : CBOE AM
    Index : SPX
    FixingHolidays : XNYS
    OptionsExchange : XCBO-CBO
    Denominated : USD
  ▾ FixingTime {2}
        __type : TIME
        value : 16:00:00
```

Figure 2: View of the data associated with a particular object, named /Assets/EqIndexes/SPX, and corresponding to the S&P 500 equity index.

The objects in the Beacon Object Database can be referenced from any process running in the platform. The Beacon Object Database is used as the only data source by most applications in Beacon. These objects can also be used as interfaces to other databases, so that developers who use Beacon have a consistent, uniform interface to all the data they need. For example, many clients connect Beacon to their existing in-house databases via secure VPN. Instead of having to source data from institutional data sources on an ad hoc basis for every analysis, Beacon requires developers to do the work of interfacing with a data

Instead of having to source data from institutional data sources on an ad hoc basis for every analysis, Beacon requires developers to do the work of interfacing with a data source only once.

source only once. Thereafter, every developer and end-user application enjoys the benefit of having just one right way to access that data forever.

When an object is loaded into a Beacon process from the Beacon Object Database, it is loaded as a fully functional business object, using the data in the database to set its properties. For example, loading the object "/Assets/EqIndexes/SPX" into memory creates an equity index object configured to represent the S&P index. However, besides simply representing an index, that object can also do interesting calculations – calculate forward equity prices, interpolate implied volatilities for equity option prices, note which futures market is based on the index, and so on. That translation from "saved set of data" to "fully featured business object" happens automatically, so developers do not need to rebuild that translation over and over.
Objects loaded into a process live in an in-memory namespace, which looks like an in-memory cache of the Beacon Object Database distributed file system. That is, objects have the same names and are cached in session to avoid reloading a common object more than once.

Bi-Temporal Time Series Data

Beacon's model for bi-temporal reference data is based on a framework where an object in the Beacon Object Database can point to versioned data based on the environment settings for the current date.Most financial market data can be represented as "time series": market data measurements taken at a range of times.

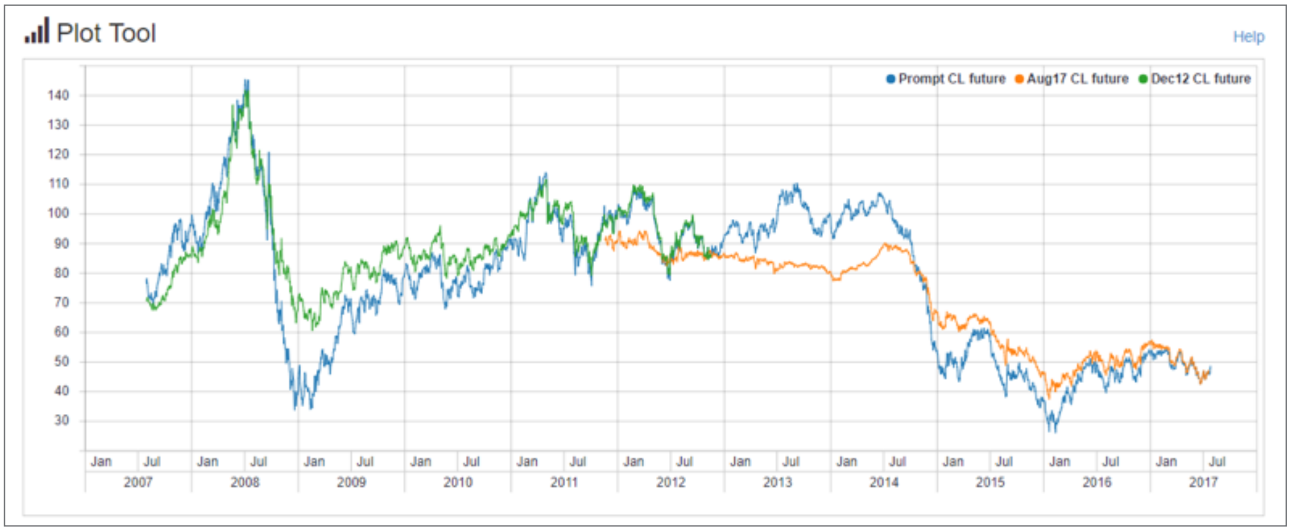


Figure 3: Financial time series examples. The blue line shows the "prompt", or closest-to-settlement, futures price for crude oil trading on the NYMEX; the green line shows the crude oil December 2012 futures price; the orange line shows the crude oil August 2017 futures price.

For daily time series – those with one point per date, such as the official settlements of futures prices shown in Figure 3 – the "as-of" time is the settlement date. In most cases, those daily points are written to the database on the settlement date, but,
occasionally, incorrect values are written to the database and need to be amended at a later time.

Important calculations, such as measures of profit and loss or risk metrics, depend on those market data values, and amendments to them can have significant impact. Since Beacon's time series are bi-temporal, it is straightforward to identify all the updates to the point for a particular "as-of" date. In Beacon, no data is ever lost, an audit trail is in place to identify the changes, and isolating the impact of data changes on calculated values is straightforward.

Bi-Temporal Reference Data

In Beacon, no data is ever lost, an audit trail is in place to identify the changes, and isolating the impact of data changes on calculated values is straightforward.

Reference data in financial markets, as opposed to market data, refers to data that is fairly static: the dates defining a coupon schedule for a corporate bond, the lot size of a futures contract, holiday calendars, and so on.

While a piece of reference data might change only rarely, reference data of some kind or another change almost every day, and, as with market data, those changes can impact important business calculations.

Beacon applies its bi-temporal data model to reference data as well, so that old versions of reference data can be easily reproduced. As a result, Beacon's bi-temporal model for reference data makes it easy to recalculate the value of a portfolio on some date in the past, quantify the impact of changes in different types of reference data,and drill into the results -and no data is ever lost.

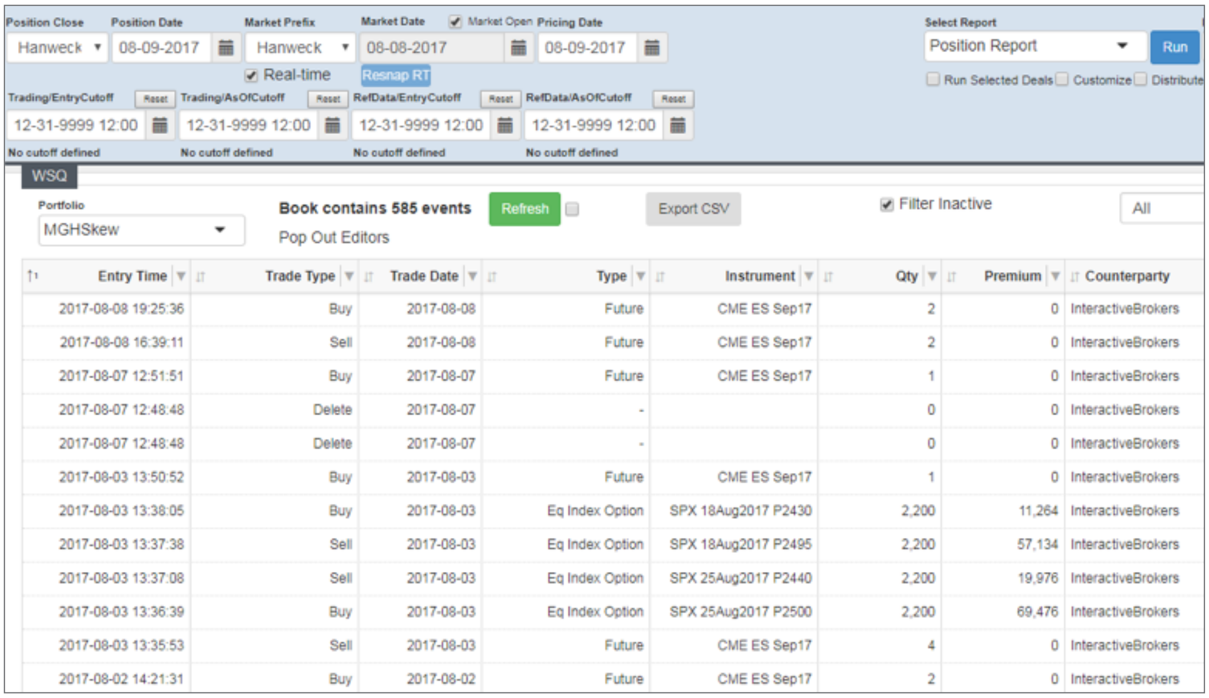
Bi-Temporal Trade and Position Data

The Beacon Standard platform product includes a "deal model" for representing financial transactions. This deal model flexibly handles all the complexity of modern derivatives trading: amendments, lifecycle events, such as option exercises and coupon payments, deal validation workflows, feeds of deal data to downstream systems, such as accounting platforms, and so on.

Beacon's ability to reproduce market states "as of" any date is critical for many business functions where old environments need to be fully recreated.

As with market data and reference data, bi-temporality is central to Beacon's deal model as well. A "deal" in Beacon is a container for a series of trade "events," ordered by time. The first event is the "open" event: the initial buy or sell of the financial instrument. If the trade is amended, a new event is added to the event series that backs out the open event's position and replaces it with the new position. That is, the original event is never deleted; instead a new event in the series is created.

If the amend was booked on a date after the original booking, Beacon makes it easy to view the positions "as of" the original date, showing the original position. Beacon's ability to reproduce market states "as of" any date is critical for many business functions where old environments need to be fully recreated.



Position Close: Hanweck | Position Date: 08-09-2017 | Market Prefix: Hanweck | Market Date: 08-08-2017 | ✓ Market Open | Pricing Date: 08-09-2017 | ✓ Real-time | Resnap RT

Trading/EntryCutoff: 12-31-9999 12:00 (No cutoff defined) | Trading/AsOfCutoff: 12-31-9999 12:00 (No cutoff defined) | RefData/EntryCutoff: 12-31-9999 12:00 (No cutoff defined) | RefData/AsOfCutoff: 12-31-9999 12:00 (No cutoff defined)

Select Report: Position Report | Run | Run Selected Deals | Customize | Distribute

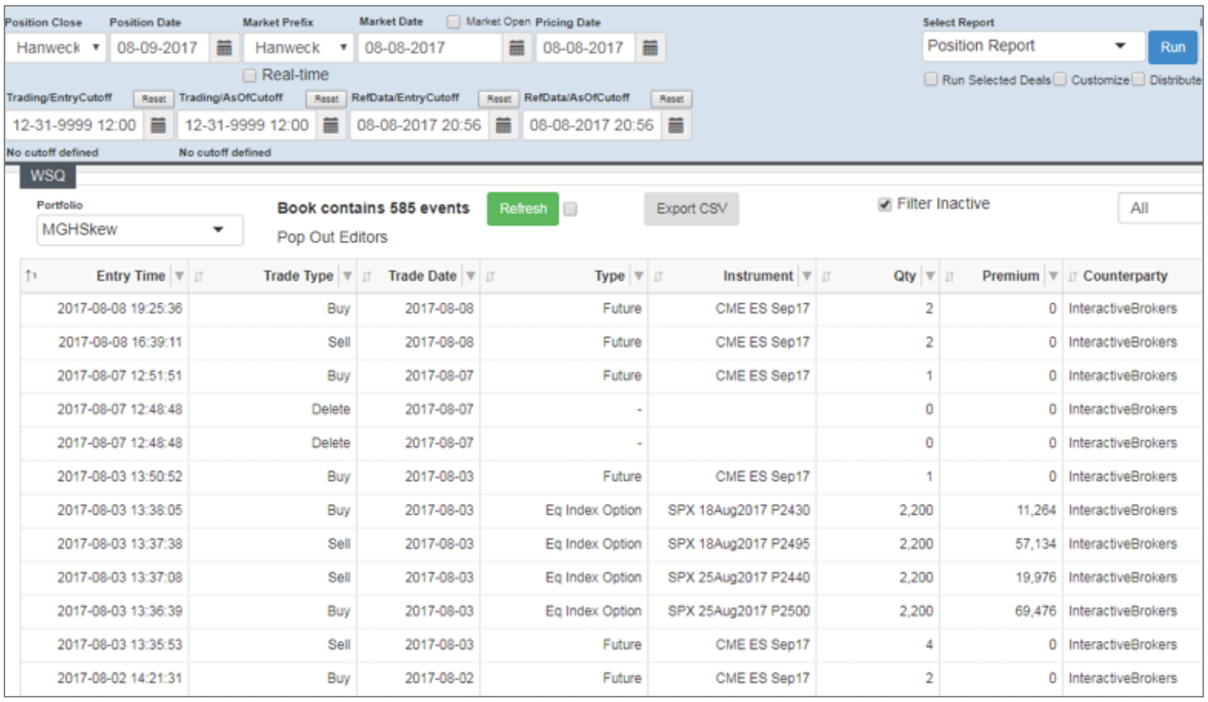
WSQ

Portfolio: MGHSkew | Book contains 585 events | Refresh | Export CSV | ✓ Filter Inactive | All

Pop Out Editors

Entry Time	Trade Type	Trade Date	Type	Instrument	Qty	Premium	Counterparty
2017-08-08 19:25:36	Buy	2017-08-08	Future	CME ES Sep17	2	0	InteractiveBrokers
2017-08-08 16:39:11	Sell	2017-08-08	Future	CME ES Sep17	2	0	InteractiveBrokers
2017-08-07 12:51:51	Buy	2017-08-07	Future	CME ES Sep17	1	0	InteractiveBrokers
2017-08-07 12:48:48	Delete	2017-08-07	-		0	0	InteractiveBrokers
2017-08-07 12:48:48	Delete	2017-08-07	-		0	0	InteractiveBrokers
2017-08-03 13:50:52	Buy	2017-08-03	Future	CME ES Sep17	1	0	InteractiveBrokers
2017-08-03 13:38:05	Buy	2017-08-03	Eq Index Option	SPX 18Aug2017 P2430	2,200	11,264	InteractiveBrokers
2017-08-03 13:37:38	Sell	2017-08-03	Eq Index Option	SPX 18Aug2017 P2495	2,200	57,134	InteractiveBrokers
2017-08-03 13:37:08	Sell	2017-08-03	Eq Index Option	SPX 25Aug2017 P2440	2,200	19,976	InteractiveBrokers
2017-08-03 13:36:39	Buy	2017-08-03	Eq Index Option	SPX 25Aug2017 P2500	2,200	69,476	InteractiveBrokers
2017-08-03 13:35:53	Sell	2017-08-03	Future	CME ES Sep17	4	0	InteractiveBrokers
2017-08-02 14:21:31	Buy	2017-08-02	Future	CME ES Sep17	2	0	InteractiveBrokers

Figure 4: The Beacon Trade Blotter enables users to define custom environments within which to run reports and analyze positions. This screenshot was taken on the real-world current date of 8-9-2017. Position Date = 8-09-2017 means live positions are used. Market Date = 08-08-2017, Market Open = True and Pricing Date = 08-09-2017 means that for all pricing, risk and analytics, the previous day's closing market data is rolled forward to the current Pricing Date, as appropriate, and replaced with real-time market data where available.



Position Close: Hanweck | Position Date: 08-09-2017 | Market Prefix: Hanweck | Market Date: 08-08-2017 | Market Open | Pricing Date: 08-08-2017 | Real-time

Trading/EntryCutoff: 12-31-9999 12:00 (No cutoff defined) | Trading/AsOfCutoff: 12-31-9999 12:00 (No cutoff defined) | RefData/EntryCutoff: 08-08-2017 20:56 | RefData/AsOfCutoff: 08-08-2017 20:56

Select Report: Position Report | Run | Run Selected Deals | Customize | Distribute

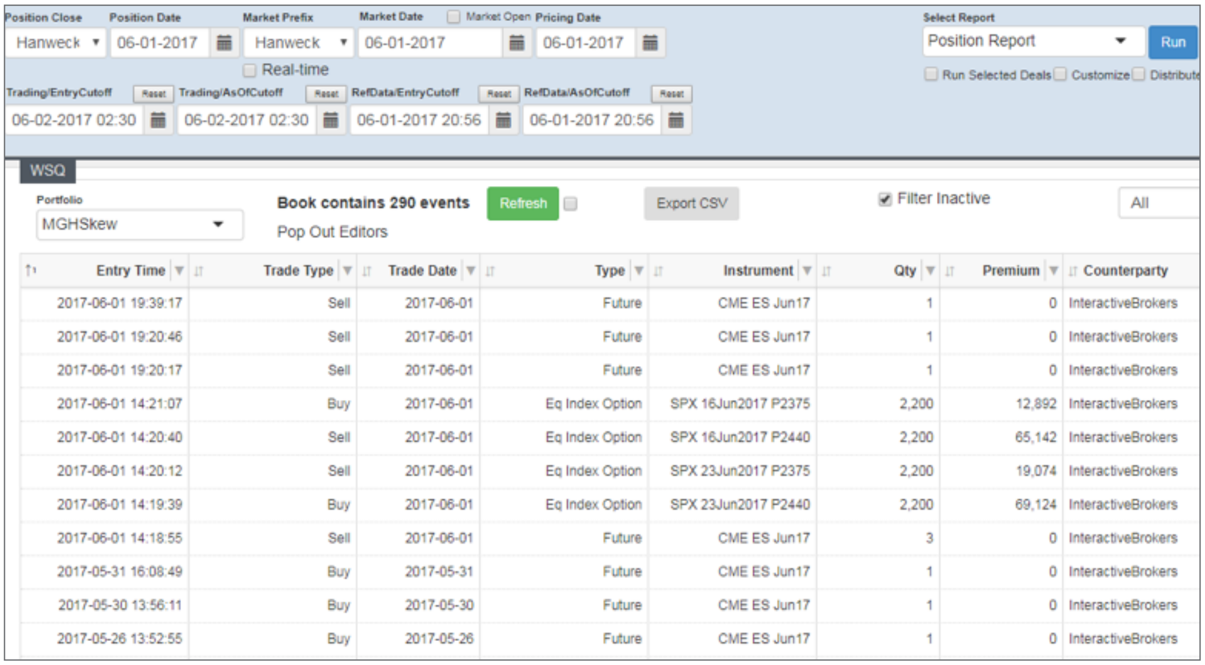
WSQ

Portfolio: MGHSkew | Book contains 585 events | Refresh | Export CSV | ✓ Filter Inactive | All

Pop Out Editors

Entry Time	Trade Type	Trade Date	Type	Instrument	Qty	Premium	Counterparty
2017-08-08 19:25:36	Buy	2017-08-08	Future	CME ES Sep17	2	0	InteractiveBrokers
2017-08-08 16:39:11	Sell	2017-08-08	Future	CME ES Sep17	2	0	InteractiveBrokers
2017-08-07 12:51:51	Buy	2017-08-07	Future	CME ES Sep17	1	0	InteractiveBrokers
2017-08-07 12:48:48	Delete	2017-08-07	-		0	0	InteractiveBrokers
2017-08-07 12:48:48	Delete	2017-08-07	-		0	0	InteractiveBrokers
2017-08-03 13:50:52	Buy	2017-08-03	Future	CME ES Sep17	1	0	InteractiveBrokers
2017-08-03 13:38:05	Buy	2017-08-03	Eq Index Option	SPX 18Aug2017 P2430	2,200	11,264	InteractiveBrokers
2017-08-03 13:37:38	Sell	2017-08-03	Eq Index Option	SPX 18Aug2017 P2495	2,200	57,134	InteractiveBrokers
2017-08-03 13:37:08	Sell	2017-08-03	Eq Index Option	SPX 25Aug2017 P2440	2,200	19,976	InteractiveBrokers
2017-08-03 13:36:39	Buy	2017-08-03	Eq Index Option	SPX 25Aug2017 P2500	2,200	69,476	InteractiveBrokers
2017-08-03 13:35:53	Sell	2017-08-03	Future	CME ES Sep17	4	0	InteractiveBrokers
2017-08-02 14:21:31	Buy	2017-08-02	Future	CME ES Sep17	2	0	InteractiveBrokers

Figure 5: This screenshot was taken on the real-world current date of 8-9-2017. Position Date = 8-09-2017 means live positions are used. Market Date = 08-08-2017, Market Open = False and Pricing Date = 08-08-2017 means that for all pricing, risk and analytics is based on the previous day's close. Sliding the RefDataEntryCutoff into the future would enable the user to pull in later updates to the closing market data, if any.



Position Close: Hanweck
Position Date: 06-01-2017
Market Prefix: Hanweck
Market Date: 06-01-2017
Market Open
Pricing Date: 06-01-2017
Real-time
Trading/EntryCutoff: 06-02-2017 02:30
Trading/AsOfCutoff: 06-02-2017 02:30
RefData/EntryCutoff: 06-01-2017 20:56
RefData/AsOfCutoff: 06-01-2017 20:56
Select Report: Position Report
Run Selected Deals
Customize
Distribute

Portfolio: MGHSkew
Book contains 290 events
Pop Out Editors
Filter Inactive: ✓
All

Entry Time	Trade Type	Trade Date	Type	Instrument	Qty	Premium	Counterparty
2017-06-01 19:39:17	Sell	2017-06-01	Future	CME ES Jun17	1	0	InteractiveBrokers
2017-06-01 19:20:46	Sell	2017-06-01	Future	CME ES Jun17	1	0	InteractiveBrokers
2017-06-01 19:20:17	Sell	2017-06-01	Future	CME ES Jun17	1	0	InteractiveBrokers
2017-06-01 14:21:07	Buy	2017-06-01	Eq Index Option	SPX 16Jun2017 P2375	2,200	12,892	InteractiveBrokers
2017-06-01 14:20:40	Sell	2017-06-01	Eq Index Option	SPX 16Jun2017 P2440	2,200	65,142	InteractiveBrokers
2017-06-01 14:20:12	Sell	2017-06-01	Eq Index Option	SPX 23Jun2017 P2375	2,200	19,074	InteractiveBrokers
2017-06-01 14:19:39	Buy	2017-06-01	Eq Index Option	SPX 23Jun2017 P2440	2,200	69,124	InteractiveBrokers
2017-06-01 14:18:55	Sell	2017-06-01	Future	CME ES Jun17	3	0	InteractiveBrokers
2017-05-31 16:08:49	Buy	2017-05-31	Future	CME ES Jun17	1	0	InteractiveBrokers
2017-05-30 13:56:11	Buy	2017-05-30	Future	CME ES Jun17	1	0	InteractiveBrokers
2017-05-26 13:52:55	Buy	2017-05-26	Future	CME ES Jun17	1	0	InteractiveBrokers

Figure 6: This screenshot was taken on the real-world current date of 8-9-2017. Positions are based on Position Date = 6-01-2017. Market Data for all pricing, risk and analytics is based on the same date. Sliding the TradeEntry-Cutoff or the TradingAsOfCutoff into the past or future would enable the user to pull in later updates to the positions.

In addition to providing an immutable audit trail for trade activity, Beacon's event series architecture lets traders associate lifecycle events with their original deals. For example, a currency option might exercise into a currency forward contract; in Beacon, the exercise is treated as another event in the event series of the original deal which evolves the positions from the option to the forward. In other trading systems, the exercise is often booked as a separate trade which can be difficult to link back to the original option trade.

Moving the Clock: Accessing Prior Data Versions

A core piece of Beacon's in-memory namespace is a "dependency graph," which defines connections between values on the business objects loaded from the Beacon Object Database. For example, one business object might be a financial instrument, such as a bond future. As is the case with every financial instrument in Beacon, the bond future object knows what its contract terms are, how to price itself, and how to evolve itself through lifecycle events.

Every financial instrument in Beacon knows what its contract terms are, how to price itself, and how to evolve itself through lifecycle events.

The price of the bond future is calculated based on the cheapest-to-deliver bond in the deliverable basket for that future, plus a spread (calculated on the fly) to match the market futures price. Beacon's dependency graph connects the bond future instrument price to the underlying market data, so that if those market data change, the price will recalculate efficiently.

At the bottom of the dependency graph, there are environment settings such as the "current date," which determine the appropriate version of the bi-temporal data (time series, reference data, trade data, and so on) to be used.

The relevant dates and times that define the "current date" environment for bi-temporal data are:

- **Market Data Date:** the "as-of" date for market data in the bi-temporal time series model.
- **RefData AsOf Cutoff:** the time of date on Market Data Date that the close corresponds to; also the time part of the "as-of" date/time.
- **RefData Entry Cutoff:** the time cutoff for entry time of market data in the bi-temporal time series model. If there is more than one update to a time series point for a piece of market data on the Market Data Date, moving the Entry Cutoff back and forth across those update times will change the market data value used for pricing and risk calculations.
- **Trade As-Of Cutoff:** the "as-of" time cutoff for trade events; any events with an as-of time after this are excluded from positions.
- **Trade Entry Cutoff:** the entry time cutoff for trade events; any events with an entry time after this are excluded from positions. For example, if a deal was entered before the close time on day 1, then amended on day 2, you could choose an as-of time for the amendment to be before the close time on day 1 (affecting the official close for day 1), but still be able to easily slide the Trade Entry Cutoff back to the close time for day 1 to see the original set of closing positions and reproduce the original behavior.

Conclusions

The data warehouse is tightly integrated with Beacon's business object model and powerful dependency graph framework, which makes interacting with objects and their associated data easy: there is no need for many separate implementations of "data loaders" to translate data into functionality.

Beacon's integrated data warehouse, in the form of the Beacon Object Database, gives a business a one stop shop for data: any new data is integrated into the warehouse only once, and it is then available for all developers and all applications, in development and in production, forever. The data warehouse is tightly integrated with Beacon's business object model and powerful dependency graph framework, which makes interacting with objects and their associated data easy: there is no need for many separate implementations of "data loaders" to translate data into functionality.

Many kinds of data in Beacon – including market data, reference data, and trade data – use bi-temporal data, which defines two times for each data point: an "as-of" time corresponding to what time the data point represents; and an "entry time" corresponding to the time the update happened. Updates to a particular data point result in new versions being created: for a single as-of time there might be updates with many different entry times.

Beacon's dependency graph, through its environment settings, makes it easy to control which versions of data are used for different kinds of functionality. This flexibility makes it easy to audit data changes, reproduce old behavior, and quantify the impact of data changes.

All versions of data are saved in the database, and no data is ever lost. Beacon's dependency graph, through its environment settings, makes it easy to control which versions of data are used for different kinds of functionality. This flexibility makes it easy to audit data changes, reproduce old behavior, and quantify the impact of data changes.

For more information, please contact info@beacon.io