

# New Horizons for Complex Event Processing



Daniel Chait

*Complex event processing is already viewed by many as an integral part of today's auto/algo trading platforms. Using market data handling as an example, Daniel Chait, Managing Director, Lab49, a financial markets technology consultancy, asserts that CEP has a much wider range of applications.*

Over the past few years, complex event processing (CEP) technology has grown from a niche category rooted in academia to one of the central pillars in the modern financial services technology stack. In this article, I will discuss the reasons why this new technology has become so widely adopted, demonstrate one of its main new uses, and preview where I see this industry headed.

## History and trends

To understand CEP, and the central role it has come to play in financial services, it is useful to consider the broader market and technology environments within which this development has taken place. Over the past decade, data volumes have skyrocketed, new sources of liquidity have mushroomed, sell-side firms have created internal crossing networks, trading has

become increasingly sophisticated and trading volumes in complex derivatives have spiked, collectively far outstripping the stock markets.

For these reasons, the old, batch-based technology model no longer suffices. In that model, firms would record their activity throughout the day, typically using a relational database, and then run a batch of programmes at the end of the day to process this data. For example, a fund would send orders to its broker throughout the trading day, storing these in a position-keeping database along the way. Then, at the close of the day, they (or their prime broker on their behalf) would re-compute their value at risk (VaR) as well as their net exposures to various factors, and use these closing values the next day to inform their trading decisions. The problem is that those nightly numbers are quickly out-of-date. As markets sped up and volumes rose, traders began demanding these numbers again at the middle of the day, then several times per day. These intraday batches came to be requested so frequently that systems designed to handle a single nightly run simply could not scale up to support the business needs.

This model has yielded to a new paradigm of continuous, event-driven systems. In this design, rather than store data for processing at a later stage, we set up the processes and then, as data arrives, act upon it immediately. For example, we might create a system to continuously re-price off-the-run US treasury bonds using a model dictating that each bond should be computed according to some

calculated spreads from a corresponding on-the-run bond. Then, with each new tick in the price of the on-the-run bond, we will trigger a cascade of analytic code to re-calculate the new prices for all the related off-the-run bonds. CEP engines are designed to enable this continuous, event-driven model.

### Buy side takes control

Another interesting effect of the acceleration in the markets is that buy-side firms have been pressured to adopt cutting-edge technology, which for a long time was the purview only of the large sell-side institutions. And as buy-side firms look to take more control over their trading functions (for example direct market access, high-speed algorithmic trading, real-time risk), they are increasingly developing tighter technology integration with their broker/dealers. These two effects have demanded increasing sophistication of their IT operations. CEP, as one of the key new technologies enabling real-time financial services, is seeing rapid adoption within buy-side firms as part of this overall trend.

CEP first gained widespread adoption in automated and algorithmic trading. Auto/algo trading systems are ideally suited to CEP for several reasons: they often require low latency and high throughput; they employ complex logic in response to outside market events; and they frequently require connections to several outside trading venues. Before the advent of CEP engines, firms had to write much of this code themselves, adding to the cost, complexity and risk of building algorithmic trading solutions. Now, using a CEP platform, developers can focus on creating the business logic to implement a particular algorithm and let the platform handle details such as connecting to exchanges, triggering trading decision logic and maintaining consistency even in the face of network outages, hardware failures etc.

Additionally, certain features of CEP products are specifically geared towards automated and algorithmic traders. For example, firms need a way to test out their trading strategies and execution algorithms prior to deployment. Most CEP products provide sophisticated features for feeding in simulated data, recording and playing back live data, and debugging applications, providing a rich environment for teams to develop, test and refine their strategies.

### Expanding CEP use cases

Because CEP was first adopted for algorithmic trading systems, it is still often thought of primarily in those terms. However, in fact there are many other interesting applications of CEP across the financial markets and firms are expanding their use to many new applications.

*“... buy-side firms have been pressured to adopt cutting-edge technology, which for a long time was the purview only of the large sell-side institutions.”*

To demonstrate how and why CEP engines are being deployed to meet new challenges, we'll take the example – drawn from recent Lab49 projects – of a system for handling market data within a hedge fund. Firms engaged in trading activity must connect to several sources of market data, run checks on that data to ensure that it is clean and valid, and convert it to standard formats. Additionally, they would like notifications when problems occur such as a feed becoming unavailable. By architecting a market data system around a CEP engine, firms can achieve these goals faster and better than they otherwise could (Figure 1 overleaf illustrates such a system). Here's why:

- **Feed handling** – CEP platforms all come with many pre-built feed handlers for common market data formats, wherein the vendors have already done much of the heavy lifting involved in connecting to the sources, parsing their proprietary data formats, handling control messages and re-connects, etc. In cases where clients want to connect to a data source for which the vendor does not already have a feed handler, the platform will provide a framework for creating new input adapters to plug in additional market data sources.
- **Data standardisation** – CEP engines can easily enable canonicalisation (the process of standardising data about the same entity from different sources within a common reference schema). Examples of data that may need canonicalisation include order size fields represented in lots of 1,000 versus single units, different date formats, various ways to represent complex structured products, etc. ▶

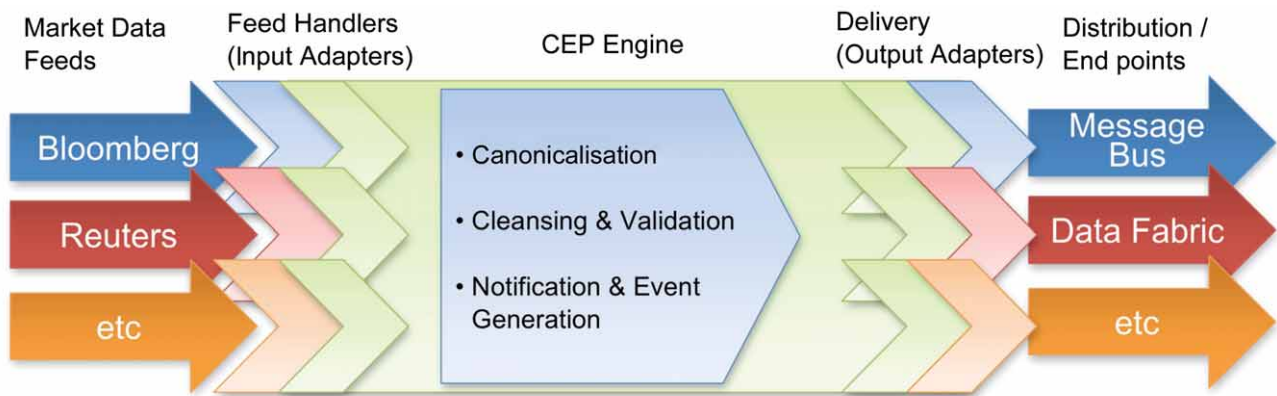


Figure 1: Architecture of a CEP-based market data system

Through their event models and schema definition languages, CEP engines allow developers to easily define their desired canonical format and then to build sophisticated transformation code that converts each incoming message as it arrives.

- **Cleansing and validation** – The central programming paradigm enabled by CEP platforms is very well suited to the cleansing and validation of market data. Essentially, cleansing and validating market data comes down to recognising patterns in the data and filtering out anomalous events which fall outside recognised parameters. A classic technique is to keep a trailing average window for a set of values and to reject any new events which fall outside normal distribution. These bad values can result from any number of errors, including fat-finger problems and transmission glitches. The programming languages and design patterns incorporated into CEP engines are ideally suited to solving this problem as they include the inherent capability for recognising patterns, creating rolling time-based data sets and computing statistics based upon streams of messages.

- **Alerts** – A market data system should be able to send notifications when problems occur. CEP engines can help with this by using their ability to monitor streams and detect patterns, and their features for hooking into email servers, posting messages to web services and other such integration points for notifications and alerts.


- **Data delivery** – Having accepted market data feeds and performed the relevant cleansing and normalisation, a market data system must handle delivery of the data to other systems, which will then

consume the data (these may include P&L systems, auto/algorithm trading systems, analytic spreadsheets and more). This brings two main challenges which CEP engines can solve. Delivery poses an integration problem, resulting from the fact that there are a host of various downstream systems that may consume the data, each with their own application programming interfaces (APIs). Just as CEP engines include both pre-built input adapters and adapter frameworks which help when connected incoming data streams, they also include output adapters for popular systems (including message buses, relational databases and distributed data caches) as well as toolkits for easily creating custom output adapters to proprietary systems. Market data delivery also presents challenges in producing data for different consumption patterns. For example, if the data is being delivered to a high-speed trading platform which is using it to make automated trading decisions based on sub-second arbitrage opportunities, the delivery system needs to produce data as fast as possible. On the other hand, if that same market data is updating a spreadsheet that a trader watches to help make investment decisions, it may not need to be updated more than once every few minutes or so, since humans cannot process and react to data that fast. CEP engines support the capability to throttle output streams according to these types of constraints, allowing customers to readily access very high performance when needed and to limit message traffic, and thus network load, for less sensitive applications.

Other use case categories including pricing, real-time risk and P&L calculation, and portfolio management follow many of the same patterns and present equally compelling use cases. ►

## CEP's inflection point

Prior to 2007, very few in the finance sector were aware of CEP products and even fewer had actually begun to implement such systems within a few specialised areas. 2007 was a breakout year in which the volume of articles, conferences and events on CEP was matched by the arrival of new vendors and use cases, confirming its status as a mainstream technology. Based on an understanding of the sector developed through Lab49's partnerships with vendors and implementations with clients, I sense that CEP is at a major inflection point right now.

Looking out to 2008 and beyond, I expect CEP technologies to mature from workgroup- or desk-level tools to enterprise-wide platforms. Tools have started to evolve to support the demands placed on other enterprise technologies such as: high availability; fault tolerance; better deployment, management and configuration tools; and tighter integration with entitlement, authentication and security infrastructure. At the same time, customers are pushing for more sophisticated end-user tools (customisable dashboards, Excel integration, ad hoc query and business intelligence tools, etc.) to lessen their reliance on IT departments every time a business user needs to see some data. For instance, the most recent version upgrades to StreamBase include features for better error handling, finer-grained security and support for several different fault tolerant deployment scenarios. Likewise, Coral8 has recently added 64-bit Windows support, an updated portal product, real-time Excel integration and enhanced security capabilities. Lastly, the emergence of related technologies such as low-latency messaging systems, grid computing platforms and data fabrics has led to a confusing patchwork of different systems. Vendors will increasingly look for tighter integration of the various overlapping feature sets these products provide, yielding simpler and more comprehensive suites of platforms from fewer vendors. For instance, BEA has added closer integration with distributed cache products as well as publish/subscribe messaging features to their event server products. 

## A Brief Buyer's Guide to CEP

Selecting a CEP product today can be a complicated matter. The various vendors in the market have taken vastly different approaches to several key aspects of their offerings and it can be complicated to sort it all out. Here are a few tips:

**Performance** – Performance matters, of course. Bear in mind, though, performance of a CEP product is composed of several aspects, including plug-in performance, scalability as queries and users are added, and effective monitoring and reporting of performance statistics in real time. Don't take the vendor's word for it that their product can handle a certain number of messages per second. Take a test drive under real-world scenarios to see how it actually stacks up.

**Features** – Feature sets vary greatly. For instance, some products focus on small, fast and lightweight runtime engines, others on massive scalability and high availability. Some include many adapters for plugging in all kinds of outside data, while others leave the coding to you. Other variables include: security; platform-independence; and richness of integrated development environments. Deciding which of these features is most important to you should guide your evaluation process.

**Design** – From SQL-like to rules-based, CEP products differ in their approach to creating applications in their systems. Some include rich visual programming environments which may or may not be equivalent in power and expressiveness to their text-based programming model. Other differences include: persistent model vs. purely transient; supporting updates and deletes vs. insert-only; and multi- vs. single-threaded.

**Platform or solution?** – Some products provide significant business-focused applications out of the box, others provide ready-to-use templates to use as starting points for building applications, while others focus on the platform and leave the application development to you. Usually I advise firms to buy the functionality where they do not feel they have a compelling competitive advantage and to build the pieces where they feel they can gain an edge. If purchasing a CEP platform with the intention of using the business-specific modules it provides (e.g. electronic trading, market liquidity, pricing and many more), then take into account the ramifications of using logic that every other customer will also have access to.