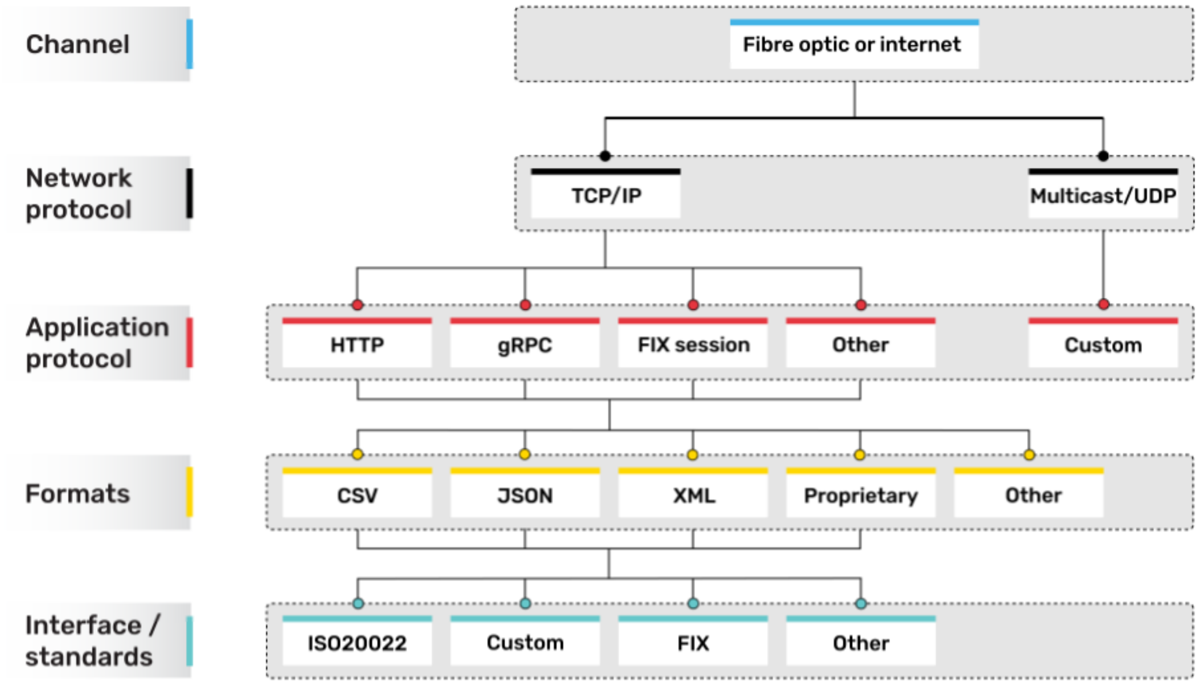
Planes, trains and automobiles .... bond CT-style

Protocols, standards, formats ... and other confusing details

Like the famous movie, getting from point A to point B in a certain time can involve various modes, methods and forms of transport. In the transaction data world in 2025, those transportation components will be key to the operation of any CT – directly affecting both the contributors and consumers of the data. However, while this is the very basis on which a CT platform will be built, there are no consistent or agreed market definitions for those components. ESMA's study outlined some of those components, although a number of important, practical elements were 'out of scope' of the assessment[1]. In order to help the discussion progress, we present our view of the 'taxonomy' that applies to the market as a whole and where those components sit:



Starting from the top, we see that - for the fixed income CT – some of the basic building blocks will have to be designed and developed in order to meet both the market, and regulator, expectations. The ESMA study *"acknowledge[ed] that the trading industry is mostly relying on FIX framework for real time market data feeds"* (although the study didn't distinguish between the equities and fixed income markets).

At Bondtape, we believe that, in order for the CT to succeed, it will have to offer and provide a variety of interfaces and protocols to ensure that costs to the data contributors are minimised and that ease of use for consumers of the CT data can be achieved.

Greater harmonisation of data standards and protocols is something that can be achieved during the first licence period, but will only happen if the CT itself succeeds and the market sees both the value and potential of the bond CT.

Contact us at info@bondtape.org to discuss.

[1] Study on data formats and transmission protocols, 25 January 2024 (ESMA12-437499640-2360) pages 10, 11 and 13

Explanatory notes and comments

Connectivity Channels

The channel is the technology that will ferry electronic signals from one system to another. Different channels have different properties (e.g. latency, cost, security).

Example channels:

- Fiber optic cable – this is where a physical cable(s) is run between the provider's servers and the CTP's servers (or CTP to consumer). This provides the CT and user with consistent very low latency and more ability to define resiliency parameters (e.g. ensuring two cables, two providers, two geographic routes if inter-data centre).
- Public internet – this is where data is ferried between a provider's servers and CTP's servers through whatever internet connections the two sides have (and then through the public internet). This doesn't have a material effect upon security because as all traffic would be encrypted in transit. Routing through the internet does impact latency (electronic signals have to travel through more network hops before reaching their destination) and more susceptible to latency variability and disruption (depending on internet traffic activity in different geographies).

Channel (/Network) Protocols

The agreed handling of the electronic signals on the connectivity channel (conflating the OSI model for ease) and what to do if things go missing, or things arrive out of order.

Example channel protocols:

- TCP/IP – this is ubiquitous protocol provides guarantees that both the sender and the receiver know whether the signals transmitted have been received. This means that if something temporarily disrupts the network connection the sender knows to resend the signals. The trade-off for this confidence is that it introduces tiny amounts of latency (delay) whilst extra signals are sent back and forth to confirm delivery. The internet is, largely, built on TCP.
- Multicast/RTP/UDP – this is basically just 'fire and forget', no frills, fast as possible, no overheads, some low percentage of things will be lost. When you're sending data over dedicated fibre optic cables, there can be some risk of loss that can be mitigated by, for example, dual multicast or call backs.

Application-level/Transport Protocols

How the applications agree to exchange data over the channel and protocols. e.g. If the channel was the Postal/Mail Service, channel protocol is 'registered delivery' vs 'first class', the transport protocol in this analogy would be an agreement between people to send a letter once a month and always use a novelty stamp. These allow developers to build upon agreed understandings of how two systems communicating with each other will behave.

Example application-level protocols:

- HTTP(S) – This is what the world wide web and RESTful APIs are based on. A simple general-purpose mechanism, not the fastest or the slowest, but incredibly well understood, flexible and documented (and good enough for the web). This protocol is based on TCP.
- gRPC – an alternative to HTTP this is another public protocol use on APIs (mostly those designed for 'call and response'). Improved performance, but there are technical trade-offs. This protocol is also based on TCP.
- Financial Information eXchange (FIX®) session layer – this TCP-based protocol is designed specifically for financial services and thus is much more opinionated about how it can be used and prioritises performance, reliability and connectivity. So, whereas HTTP can ferry any data, FIX® can only do key + value pairs encoded a specific way, it is optimised to be faster and deliver data in order and, obviously, is already present in the market (particularly equities). It's also worth noting that FIX® also defines message formats as part of its' 'family of standards'.
- Custom/proprietary – the larger trading venues have, in many cases, existing esoteric formats in which they encode data. These are designed for performant and reliable distribution, however, it places the burden on the consumer to read through their documentation to understand the details of the messages they're going to get and how to extract the data they care about (they usually provide helpful software and libraries).

Data Formats

The format of the data refers to how the information is structured when stored in bytes. Different formats are optimised for different things – whether it be real-time streaming (minimal bytes, implicit/rigid structure), flexibility (lots of bytes used to provide context and variable structure) or somewhere in the middle (structure, but limited features so fewer bytes).

Example data formats:

- JSON – born out of the internet (and pervasively used there) this is a way of writing structured information with not too much overhead (and thus supports fewer features). Designed for documents rather than streaming, like XML it can support nested structured data (e.g. parent data items that contain child data).
- XML – pre-dating JSON this is the original heavyweight structured data format, can describe anything and support a breadth of features, but uses up a lot of extra space and isn't good for streaming.
- CSV – simple tabular format used the world over for capturing application-agnostic datasets, though does not support structure.
- Proprietary – some vendors have their own proprietary way of structuring data to suit their own needs, such as real time streaming. FIX® uses a flavour of XML called FIXML.
- Other - there are various other data formats, each with their own trade-offs.

Interface

Interface refers to how data is laid out within the messages being sent. To stretch the metaphor used above with the Postal/Mail Service, the Interface would be the people agreeing that anytime a letter had bad news, it would be written right at the beginning – i.e. some pre-agreed understanding between parties as to the structure and presence of information within any kind of message.

Example interface specifications:

- ISO20022 – used by the SWIFT network, this is a standard that prescribes for 753 kinds of messages (across use cases) what each means, and what fields and metadata must be included, and how the values should be specified (e.g. decimal places, required or optional, how many letters in a currency). Such events are '*ForeignExchangeTradeCaptureReportV01*' or '*CorporateActionInstructionV12*'.
- Custom API – where a firm defines its' own specification/API that defines what fields for each kind of event required, and how they will be validated. It is worth noting that it is possible to build an API that conforms to ISO20022 (i.e. this is just about how the messages are structured, not how they're consumed).
- FIX® – the FIX® domain extends to define expected standard data to be populated for things like Trade Capture Reports, though these can be augmented if both sender and recipient agree what data is expected.

Data Standards

Data standards refer to the expected data to be provided in any kind of message, and what it means – so, if an identifier is required, the standard might be that it's an ISI and that any currencies confirm to ISO 4217.

Example data standards:

- ISO20022 – this comes with data standards as part of its specification
- Custom – the CTP could develop a standard, based on what the market wants. This would certainly be the case if the CTP has it define its own API.

DISCLAIMER