Energy Interoperation Common Transactive Services (CTS) Version 1.0

Committee Specification Draft ~~02~~03

28 ~~April 2022~~March 2024

This stage:

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd03/ei-cts-v1.0-csd03.pdf (Authoritative)

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd03/ei-cts-v1.0-csd03.html

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd03/ei-cts-v1.0-csd03.docx

Previous stage:

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd02/ei-cts-v1.0-csd02.pdf (Authoritative)

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd02/ei-cts-v1.0-csd02.html

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd02/ei-cts-v1.0-csd02.docx

~~**Previous stage:**~~
~~(Authoritative)~~

Latest stage:

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/ei-cts-v1.0.pdf (Authoritative)

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/ei-cts-v1.0.html

https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/ei-cts-v1.0.docx

Technical Committee:
OASIS Energy Interoperation TC

Chairs:
William T. Cox (wtcox@coxsoftwarearchitects.com), Individual
David Holmberg (david.holmberg@nist.gov), NIST
~~William T. Cox (wtcox@coxsoftwarearchitects.com), Individual~~

Editor:
Toby Considine (toby.considine@unc.edu), University of North Carolina at Chapel Hill

Related work:
This document replaces or supersedes:

- Common Transactive Services 1.0. The Energy Mashup Lab Specification. Edited by William T. Cox, Toby Considine 30 November 2020. https://www.theenergymashuplab.org/s/cts-1-0-draft-20201130.pdf.

This document is related to:

- *Energy Interoperation Version 1.0*. Edited by Toby Considine, 11 June 2014. OASIS Standard. http://docs.oasis-open.org/energyinterop/ei/v1.0/os/energyinterop-v1.0-os.html. Latest version: http://docs.oasis-open.org/energyinterop/ei/v1.0/energyinterop-v1.0.html. and its TeMIX Profile
- *Energy Market Information Exchange (EMIX) Version 1.0.* Edited by Toby Considine. Latest version: http://docs.oasis-open.org/emix/emix/v1.0/emix-v1.0.html.

- *WS-Calendar Platform Independent Model (PIM) Version 1.0*. Edited by William Cox and Toby Considine. Latest version: http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/ws-calendar-pim-v1.0.html.
- *Schedule Signals and Streams Version 1.0.* Edited by Toby Considine and William T. Cox. Latest version: http://docs.oasis-open.org/ws-calendar/streams/v1.0/streams-v1.0.html.

Abstract:

Common Transactive Services (CTS) permits energy consumers and producers to interact through energy markets by simplifying actor interaction with any market. CTS is a streamlined and simplified profile of the OASIS Energy Interoperation (EI) specification, which describes an information and communication model to coordinate the exchange of energy between any two Parties that consume or supply energy, such as energy suppliers and customers, markets and service providers.

Status

This document was last revised or approved by the OASIS Energy Interoperation TC on the above date. The level of approval is also listed above. Check the "Latest stage" location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://~~www~~groups.oasis-open.org/~~committees~~communities/tc~~_home.php?wg_abbrev=energyinterop~~-community-home2?CommunityKey=d3a22f3f-887d-465e-bb0f-018dc7d3f405#technical.

Any individual may submit comments to the TC at Technical-Committee-Comments@oasis-open.org. TC members should send comments on this document to the TC's email list.~~ Others should send comments to the TC's public comment list, after subscribing to it by following the instructions at the "" button on the TC's web page at .~~

This document is provided under the RF on Limited Terms Mode of the OASIS IPR Policy, the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this document, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC's web page (https://www.oasis-open.org/committees/energyinterop/ipr.php).

Note that any machine-readable content (Computer Language Definitions) declared Normative for this Work Product is provided in separate plain text files. In the event of a discrepancy between any such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

Key words:

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

Citation format:

When referencing this document, use the following citation format~~ should be used~~:

[Energyinterop-CTS-v1.0]

Energy Interoperation Common Transactive Services (CTS) Version 1.0. Edited by Toby Considine. 28 ~~April 2022~~March 2024. OASIS Committee Specification Draft ~~02~~03. https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/csd03/ei-cts-v1.0-csd03.html. Latest stage: https://docs.oasis-open.org/energyinterop/ei-cts/v1.0/ei-cts-v1.0.html.

Notices:

Table of Contents

Table of Tables

Table of Figures

1 Introduction

The Common Transactive Services (CTS) is an application profile of the OASIS Energy Interoperation 1.0 (**[EI]**) specification, with most optionality and complexity stripped away. CTS defines the messages for transactive energy, leaving communication details unspecified. Transactive energy names the collaboration techniques to balance energy supply and energy demand at every moment even as power generation becomes decentralized and as the ownership of energy assets becomes more diverse.

The purpose of CTS is to enable broad semantic interoperation between systems in transactive energy-based markets, or in any markets whose products are commodities distinguished chiefly by time of delivery. These time-volatile commodities are termed resources, and the interactions defined in CTS are common to any market used to manage resources over time.

To encourage broad adoption, CTS uses terms from financial markets in preference to the relatively obscure terms used in specialized energy markets. Among these is the use of the term instrument for a tradable asset, or a negotiable item. In CTS, the term instrument encompasses a quantity of a Resource delivered at a particular time for a particular duration. A transaction is created when a buyer and seller agree on the price for an instrument.

Transactive resource markets coordinate ~~Resource~~resource supply and ~~Resource~~resource use through markets that trade instruments. The initial research into transactive resource markets used a market to allocate heat from a single furnace within a commercial building. Transactive resource markets balance supply and demand over time using automated voluntary transactions between market participants.

Examples of transactable resources include, but are not limited to, electrical energy, electrical power, natural gas, and thermal energy such as steam, hot water, or chilled water. The capability to transmit such time-dependent resources is also a transactable resource, as instruments can be defined for transmission rights as well as for the services that maintain grid frequency or voltage.

When we apply transactive resource markets to the distribution of power or energy, we refer to it as transactive energy. A significant driver of transactive energy is the desire to smooth supply and demand variability, or alternatively, to match demand to variable supply. We anticipate this variability to increase as additional variable and distributed generation sources are connected to the power grid. The reader can find an extended discussion of Transactive Energy (TE) in the EI specification **[EI]**

A goal of CTS is to enable systems and devices developed today or in the future to address the challenges of increasing distributed energy resources. CTS enables distributed actors to participate in markets deployed today or in the future. ~~The reader can find an extended discussion of Transactive Energy (TE) in the **[EI]** specification.~~

CTS defines interactions between actors in energy markets. We do not identify whether an ~~Actor~~actor is a single integrated system, or a distributed collection of systems and devices working together. See Section 1.5~~1.5~~ for a discussion of the term Actor in this specification. Autonomous market actors must be able to recognize patterns and make choices to best support their own needs.

CTS assumes the perspective of a trader, that is of a market participant. [EI] was developed with significant input form Economists and energy market regulators, and it relies on language from economics and regulation. The Committee deliberately chose to seek guidance from financial traders and to use their language. Many data elements and message types have been renamed to align with FIX-based markets.

CTS messages are simple and strongly-typed~~,~~ and make no assumptions about the systems or technologies behind the actors. Rather, CTS defines a technology-agnostic minimal set of messages to enable interoperation through markets of participants irrespective of internal technology. In a similar manner, CTS does not specify the internal organization of a market, but rather a common set of messages that can be used to communicate with any transactive energy market.

The Common Transactive Services, strictly speaking, are a definition of the payloads and exchange patterns necessary for a full-service environment for interaction with markets. In other words, CTS describes the message payloads to be exchanged, defining the semantic content and ordering of messages. Any message exchange mechanism may be used, including but not limited to message queues and Service-Oriented mechanisms.

In a Service-Oriented Architecture [SOA] environment, the semantic payloads are those sent and returned by the *services* described. CTS enables any SOA or other framework to exchange equivalent semantic information without presuming the specific messaging system(s) or architecture used, thus allowing straightforward semantic interoperation.[1] See Section 2.3~~.~~. Th

1.1 Application of the Common Transactive Services

The purpose of this specification is to codify the common interactions and messages required for energy markets. Any system able to use CTS should be able to interoperate with any CTS-conforming market with minimal or no change to system logic. The full protocol stack and cybersecurity requirements for message exchange between systems using CTS are out of scope.

Systems that can be represented by CTS actors include but are not limited to:

- Smart Buildings/Homes/Industrial Facilities
- Building systems/devices
- Business Enterprises
- Electric Vehicles
- Microgrids
- Collections of IoT (Internet of Things) devices

TE demonstrations and deployments have seldom been interoperable—each uses its own message model and its own market dynamics. ~~Many early implementations required transmitting information far beyond that needed for transactions to remote or cloud-based decision aggregators termed markets.~~ Such markets discount local decision making while introducing new barriers to resilience such as network failure. Others rely on a single price-setting supplier. Systems built to participate in these demonstrations and deployments have been unable to interoperate with other implementations. The intent of this specification is to enable systems and markets developed for future deployments to interoperate even as the software continues to evolve.

CTS does not presume a Market with a single seller (e.g., a utility). CTS recognizes two parties to a transaction, and the role of any Party can switch from buyer to seller from one transaction to the next. Each Resource Offer (Tender) has a Side attribute (Buy or Sell). when each transaction is committed (once the product has been purchased) it is owned by the purchaser, and it can be re-sold as desired or needed.

A CTS~~-operated~~ micromarket may balance power over time in a traditional distribution system attached to a larger power grid or it may bind to and operate a stand-alone autonomous microgrid **[SmartGridBusiness].**

1.2 Support for Developers

Specific coding, message, and protocol recommendations are beyond the scope of this specification which specifies information content and interactions between systems. The Common Transactive Services payloads are described using the Universal Modelling Language **[UML] and** defined in XML schemas **[XSD]**. Many software development tools can accept artifacts in UML or in XSD to enforce proper message formation. To further support message interoperability, two additional common serializations are defined:

(1) This specification provides **[JSON]** schemas compatible with JSON Abstract Data Notation **[JADN]** format.

(2) This specification provides **[SBE]** schemas. The FIX Simple Binary Encoding ~~**[SBE]**~~ specification is used in financial markets~~.~~ and for general high-performance messaging—SBE is designed to encode and decode messages using fewer CPU instructions than standard encodings and without forcing memory

[1] SOA is occasionally mis-described as a *client-server* approach. In distinction, services are requested by an Actor, and fulfilled by another Actor. In SOA the services offered are key, and the actors take different roles in different interactions.

management delays. SBE-based messaging is used when very high rates of message throughput are required. ~~This specification will deliver schemas for generating SBE messages based on the common message content.~~

Naming Conventions

This specification follows some naming conventions for artifacts defined by the specification, as follows:

For the names of elements and the names of attributes within XSD files and UML models, the names follow the lowerCamelCase convention, with all names starting with a lower-case letter. For example,

```
<element name="componentType" type="ei:ComponentType"/>
```

For the names of types within XSD files, the names follow the UpperCamelCase convention with all names starting with an upper-case letter ~~prefixed~~suffixed by “type-“. For example,

```
<complexType name="ComponentServiceType">
```

For clarity in UML models the suffix “type” is not always used.

For the names of intents and for attributes in the UML models, names follow the lowerCamelCase convention, with all names starting with a lower-case letter, EXCEPT for cases where the intent represents an established acronym, in which case the entire name is in upper case.

JSON and where possible SBE names follow the same conventions.

1.3 Editing Conventions

For readability, element names in tables appear as separate words. The actual names are lowerCamelCase, as specified above, and as they appear in the UML models, and in the XML and JSON schemas.

All elements in the tables not marked as “optional” are mandatory. This is the opposite of the convention used in the specification of FIX.

Information in the **FIX Field** column is non-normative, and includes in parentheses zero or more FIX Tags that are related to the field. This provides guidance for those integrating CTS markets to interoperate with markets supporting the FIX Protocol.

Information in the **Meaning** column of the tables is normative. Information appearing in the **Notes** column is explanatory and non-normative.[2]

Examples and Appendices are non-normative. In particular, architectural and functional examples are presented only to support narrative description. The specific processes, structures, and algorithms are out of scope.

1.4 FIX and the Language of Trading

As noted above, this specification strives to apply the language of financial trading to resource markets. FIX is the language of trading.

We thank members of the FIX Trading Community (https://www.fixtrading.org/) for their extensive input and close reading. FIX was formed in 1991 to connect the global ecosystem of venues, asset managers, banks/brokers, vendors and regulators by standardizing the communication among participants. FIX relies on 4 key principles:

- Creating and maintaining robust open standards across the across the trade life-cycle with its pre-trade, trade, and post-trade environments.
- Providing advice and counsel to regulatory bodies in a transparent and unbiased way.
- Seeking ways to improve the trading process front to back for the global financial services industry.

[2] In ISO and IEC standards, portions that are not normative are *informative*. OASIS uses the term *non-normative*.

- Providing FIX members with a neutral, collaborative environment to come together through member-driven conferences and other critical forums to promote, support and educate.

This specification relied strongly on their assistance.

~~1.4~~1.5 Use of terms Actors and Facets in this specification

This specification defines message content and interaction patterns.

The EI 1.0 specification in 2011~~,~~ presumed web services for interactions. That specification described a Service-Oriented Architecture (SOA) approach. Service orientation complements loose integration and organizes distributed capabilities that may be in different ownership domains by focusing solely on requested results rather than on mechanisms. [EI] uses the language of web services to describe all interactions.

There is a growing use of the descriptive term “cloud-native computing” for extending the architecture and technologies developed for use in clouds not only in data centers but to edge computing, where IoT devices reside. A discussion of the rapidly-evolving topics of cloud-native computing and edge computing is beyond the scope of this specification.

At the time of this specification, typical architectures decompose applications into smaller, independent building blocks that are easier to develop, deploy and maintain. A single market participant in energy may be embodied as several of these independent blocks (actors). ~~Message queues provide loosely-coupled communication and coordination within and among these distributed applications. Message queues enable asynchronous communication, i.e., the endpoints that are producing and consuming messages interact with the queue, not each other. Publishers can add requests to the queue without waiting for their processing. Subscribers process messages only when they are available.~~

~~In~~For the Internet of Things (IoT), the term Actor ~~is preferred as~~begins and ends at the interfaces to things. The “actor model” makes no assumptions of the mechanisms or even motives internal to an Actor. An Actor is simply a thing that acts. The Actor implementation may be by a traditional computer, a cloud node, a human behind a user interface, or any device ~~in~~on the Internet of things.

In transactive energy, ~~we see~~the actor model supports the diversity ~~supported by the term Actor in the~~of IoT and of markets. An energy seller may be a generator or a solar panel or a virtual power plant or a demand responsive facility or a financial entity. An energy buyer or seller may be a home or commercial facility or an embedded device or a microgrid or an energy district. A ~~Marketplace~~Market acts to match Tenders~~, but~~. An Actor may ~~also participate to buy~~ take a market-maker role, buying and/or ~~sell~~selling power for itself. An energy storage system may act as a buyer or as a seller at any time.

~~Architectures MAY decompose applications into smaller independent Actors.~~ We use the term “Facet” to name a coherent ~~se~~set of ~~interactions~~messages that~~ such~~ an Actor may use to communicate with other Actors. An Actor submits tenders to buy or to sell. An Actor may operate a Market. If the architecture ~~requires~~includes a telemetry ~~for~~Actor, measuring Resource flow (metering), ~~one of many facets supported by a Resource-consuming~~then that Actor MAY ~~provide it, or separate Actor MAY present only~~represent the ~~telemetry, logically and physically separated from~~ Market or the ~~Resource-consuming Actor.~~market participant or even a third party. This specification makes no requirement as to how to distribute or make use of these facets.

While this specification discusses messages between Actors, it establishes no requirement or expectation of specific implementation. While this specification uses the language of Actor and Facet, there is no architectural expectation linked to this language. One could apply the terms Actor and Facet throughout the [EI] specification. A traditional [EI] application consisting of a several unitary systems each presenting all facets as web services described by WSDL can be conformant so long as it uses a compatible set of information payloads.

~~A discussion of the rapidly-evolving topic of cloud-native computing is beyond the scope of this specification. This specification does not require that implementations conform to any specific implementation of cloud-native computing. Cloud-native and edge computing have informed the language of this specification, just as web services and SOA informed [EI].~~

~~1.5~~1.6 Security and Privacy

Service requests and responses are generally considered public actions of each interoperating system, with limitations to address privacy and security considerations (see Appendix B). Service actions are independent from private actions behind the interface (i.e., device control actions). A Facet is used without needing to know all the details of its implementation. Consumers of services generally pay for results, not for effort.

~~1.5.1~~1.6.1 Security Considerations

Size of transactions, costs of failure to perform, confidentiality agreements, information stewardship, and even changing regulatory requirements can require that similar transactions be expressed within quite different security contexts. Loose integration using the service-oriented architecture (SOA) style assumes careful definition of security requirements between partners. It is a feature of the SOA approach that security is composed in to meet the specific and evolving needs of different markets and transactions. Security implementation is free to evolve over time and to support different needs. The Common Transactive Services allow for this composition, without prescribing any particular security implementation.

The best practice in cloud-native computing is to use Zero-Trust security **[ZeroTrust]**. Zero Trust security requires authentication and authorization of every device, person, and application. The best practice is to encrypt all messages, even those between the separate components of an application within the cloud.

This specification makes no attempt to describe methods or technologies to enable Zero Trust interactions between Actors.

~~1.5.2~~1.6.2 Privacy Considerations

Detailed knowledge of offers to buy or sell or knowledge of energy inputs and outputs for an Actor may reveal information on actions and operations. For example, transactions or tenders may indicate whether a production line is starting or stopping, or anticipated energy needs, or who has been buying or selling power. Making such information public may be damaging to actors. Similarly, an adverse party may be able to determine the likelihood that a dwelling is presently occupied.

The essence of any transaction is the agreement of a Party to sell, and of another a Party to buy. The identity of the buyer and the identity of the seller are each part of the transaction. Some transaction notifications may hide the identity of the buyer from the seller. Some transaction notifications may hide the identity of the seller from the buyer. Some transactions, such as a double auction, may be between the market participants as a whole, and not with any particular counterparty. Where this is required, the Market itself may be designated as the counterparty in a notification.

Both security and privacy considerations are addressed in Appendix B.

~~1.6~~1.7 Semantic Composition

The semantics and interactions of CTS are selected from and derived from OASIS Energy Interoperation [EI]. EI references two other standards, [EMIX] and [WS-Calendar], and uses an earlier Streams definition. We adapt, update, and simplify the use of the referenced standards, while maintaining conformance.

See Appendix C, Semantic Composition from Energy Interoperation, EMIX, and WS-Calendar.

- EMIX describes price and product for electricity markets.
- WS-Calendar communicates schedules and sequences of operations. CTS uses the [Streams] optimization, which is a standalone specification, rather than part of EI 1.0.
- EI uses the vocabulary and information models defined by those specifications to describe the services that it provides. The payload for each EI service references a product defined using [EMIX]. EMIX schedules and sequences are defined using [WS-Calendar]. Any additional schedule-related information required by [EI] is expressed using [WS-Calendar].

- Since ~~[EI] was~~OASIS published~~,~~ [EI], a semantically equivalent but simpler [Streams] specification was developed in the OASIS WS-Calendar Technical Committee. CTS uses that simpler [Streams] specification.

See Appendix C, Semantic Composition from Energy Interoperation, EMIX, and WS-Calendar~~All terms used in this specification are as defined in their respective specifications.~~

.

In [EI], the fundamental resource definition was the [EMIX] Item, composed of~~:~~ a resource name, a unit of measure, a scale factor, and a quantity. For example, a specific EMIX ~~item~~Item may define a Market denominated in 25 ~~MW-hour~~MWh bids. ~~[EI] defined how to buy and sell items during specific intervals defined by a duration and a start time. The Quotes, Tenders, and Transactions that are the subject of [EI] added specific prices and quantities to the item and interval. EMIX optionally included a location, i.e., a point of delivery for each [EI] service.~~

In CTS, we group and name these elements as a Resource, Product, and Instrument. These terms are defined in Section 2.2.4, "Markets and Venues"

Note that the informational elements in a fully defined tender or transaction are identical to those described in EMIX. The conceptual regrouping enables common behaviors including Market discovery and interoperation between Actors built on different code bases.

~~1.6.1 Conformance with Energy Interoperation~~

~~EI defines an end-to-end interaction model for transactive services and for demand response. CTS uses the EI transactive services, and draws definitions of parties and transactive interactions primarily from the [EI] TEMIX profile.~~

~~This specification can be viewed as a minimal transactive profile of [EI]~~

~~1.6.21.1.1 Conformance with EMIX~~

~~This specification uses a simplified profile of the models and artifacts defined in OASIS Energy Market Information Exchange **[EMIX]** to communicate Product definitions, quantities, and prices. EMIX provides a succinct way to indicate how prices, quantities, or both vary over time.~~

~~The EMIX Product definition is the Transactive Resource in CTS 1.0.~~

~~EMIX also defines Market Context, a URI used as the identifier of the Market. EMIX further defines Standard Terms as retrievable information about the Marketplace that an actor can use to configure itself for interoperation with a given Marketplace. We extend and clarify those terms, provide an extension mechanism, and discuss the relationship of markets, Marketplaces, and products.~~

~~1.6.31.1.1 Conformance with WS-Calendar Streams~~

~~WS-Calendar expresses events and sequences to support machine-to-machine (M2M) negotiation of schedules while being semantically compatible with human schedules as standardized in [**iCalendar**]. Schemas in **[WS-Calendar]** support messages that are nearly identical to those used in human schedules. We use a conformant but simpler and more abstract Platform Independent Model **[CAL-PIM]** and the [**Streams**] compact expression[3], to support telemetry (Delivery Facet) and series of Tenders while not extending the semantics of [**Streams**].[4]~~

~~By design and intent, the **[WS-Calendar]** schemas provide the capability of mapping between human and M2M schedules.~~

~~WS-Calendar conveys domain specific information in a per-event payload. An essential concept of WS-Calendar is inheritance, by which a starting time can be applied to an existing message, or by which all events in a sequence share common information such as duration. Inheritance is used to "complete" a~~

[3] Simplified as CTS Streams in this specification.

[4] Some specifications (e.g. [FSGIM]) have extended the basic [Streams] capabilities, but this brings additional complexity which does not benefit our use cases.

~~partial message during negotiation. CTS makes use of this to apply a common market Product across a sequence, or to convey a specific starting time to a market Product.~~

~~CTS messages conform to **[Streams]** format. See also Section .~~

~~1.6.4 Compatibility with Facilities Smart Grid Information Model~~

~~The Facilities Smart Grid Information Model [**FSGIM**] was developed to define the power capabilities and requirements of building systems over time. FSGIM addresses the so-called *built environment* and uses the semantics of WS-Calendar and EMIX to construct its information models for power or other Resource use over time. These sequences of [power] requirements are referred to as load curves. Load curves can potentially be relocated in time, perhaps delaying or accelerating the start time to get a more advantageous price for [power].~~

~~Because FSGIM load curves use the information models of EMIX and WS-Calendar, conforming load curves submitted by a facility could be the basis upon which a TE Agent would base its market decisions.~~

~~The Architecture of CTS is premised on distinct physical systems being able to interoperate by coordinating their production and consumption of energy irrespective of their ownership, motivations, or internal mechanisms. This specification defines messages and interactions of that interoperation.~~

~~FSGIM load requests can be expressed using CTS tenders. CTS 1.0 uses single-interval **[Streams]** to express single-interval tenders in anticipation of possible future use of Streams in FSGIM-conformant communications.~~

~~2~~1 ~~Overview of Common Transactive Services~~

~~2.1 Scope of Common Transactive Services~~

~~CTS provides for the exchange of resources among parties which represent any provider or consumer of energy (e.g., a distributed energy resource). CTS makes no assumptions as to their internal processes or technology.~~

~~This specification supports agreements and transactional obligations, while offering flexibility of implementation to support specific approaches and goals of the various participants.~~

~~No particular agreements are endorsed, proposed or required in order to implement this specification. Energy market operations are beyond the scope of this specification although interactions that enable management of the actual delivery and acceptance are within scope but not included in CTS 1.0.~~

~~As shown in [CTS2016] the Common Transactive Services with suitable Product definitions can be used to communicate with essentially any market.~~

~~2.2~~1.8 Applicability to Microgrids (Informative)

As an extended example, using the Common Transactive Services terminology, a microgrid is comprised of interacting nodes each represented by an actor (interacting as CTS parties). Those actors interact in a micromarket co-extensive in scope with the microgrid. No actor reveals any internal mechanisms, but only its interest in buying and selling power.

An actor can represent a microgrid within a larger micromarket; the actor would in effect aggregate the resources in the microgrid. As above, such an actor would not reveal any internal mechanisms, but only its interest in buying and selling power. There is no explicit bound on repeating this interoperation pattern.

An actor representing a microgrid may interoperate with markets in a regional grid, which may or may not be using CTS. ~~The regional grid may use transactive energy expressed in non-CTS messages, or~~In addition, infrastructure capacity may limit delivery to the microgrid. ~~In either case, the An actor~~The Actor representing a microgrid must translate and enforce constraints and share information with the other nodes in the microgrid solely by means of CTS. Any translations or calculations performed are out of scope.

See informative references [StructuredEnergy] and [SmartGridBusiness] for a discussion. [Fractal Microgrids] is an early reference that describes hierarchies of microgrids. [Transactive Microgrids] describes transactive energy in microgrids.

~~2.3~~1.9 Specific scope statements

This specification interprets Energy Interoperation from the perspective of a Trader interacting with a Market. CTS defines Pre-Trade, Trade, and Post-Trade information exchanges. Trading refers to the specific interactions that buy or sell a resource. A Trader uses pre-trade information to discern the operation of the Market and the actions of the other Traders. Post-Trade information informs the participants of the Trade, tracks whether the resource is delivered, and any resulting changes to the Trader's ability to participate in the Market.

Interaction patterns and facet definitions to support the following are in scope for Common Transactive Services:

- Interaction patterns to support transactive energy, including tenders, transactions, and supporting information.
- Information models for price and Product communication
- Information models for ~~Marketplace~~Market and Market Segment characteristics
- Payload definitions for Common Transactive Services

The following are out of scope for Common Transactive Services:

- Requirements specifying the type of agreement, contract, Product definition, or tariff used by a particular market.

- Computations or agreements that describe how power is sold into or sold out of a ~~Marketplace~~market.
- Communication protocols, although semantic interaction patterns are in scope.

This specification describes standard messages, the set of which may be extended.

~~2.3.1 Resources, Products, and Instruments~~

~~Systems use the common transactive services to operate transactive resource markets. A transactive resource market balances the supply of a resource over time and the demand for that resource by using a market specifying the time of delivery.~~

~~In [EI], the fundamental resource definition was the [EMIX] Item, composed of: a resource name, a unit of measure, a scale factor, and a quantity. For example, a specific EMIX item may define a Market denominated in 25 MW-hour bids. [EI] defined how to buy and sell items during specific intervals defined by a duration and a start time. The Quotes, Tenders, and Transactions that are the subject of [EI] added specific prices and quantities to the item and interval. EMIX optionally included a location, i.e., a point of delivery for each [EI] service.~~

~~In CTS, we group and name these elements as a Resource, Product, and Instrument. A Resource is the name and the unit of measure, as in the EMIX Item. A Product, i.e., what can be bought and sold in a Market, in addition specifies “how much” and “for how long” as well as optional elements such as location and Warrants. The term Instrument, as in financial markets, adds a specific start time to a Product.~~

~~We define a Resource as any commodity whose value is determined by a fine-grained time of delivery. Transactable resources include, but are not limited to, energy, heat, natural gas, water, and transport as a support service for these. The ancillary services reactive power, voltage control, and frequency control are also transactable.~~

~~A Product names a transactive Resource that has been “chunked” for Market. These chunks define the Market’s granularity in quantity and in time. For example, the Product may be 1 MW of power delivered over an hour. Similarly, another Product may be 1 kW of power over a 5-minute period. Some transactive energy markets in North America today have durations as brief as two seconds. Temporal granularity is equally important as quantity for Product definition.~~

~~An Instrument is a Product at a specific time, following common usage in financial markets where an instrument names the thing that is bought or sold. For example, the 1 MW of Power delivered over an hour beginning at 3:00 PM is a different Instrument than the same Product delivered starting at 11:00 PM.~~

~~A Market considers all the tenders it has received offering to buy or sell an Instrument, using a Matching Engine to decide which can be cleared (*satisfied*) in full or in part. The 3:00pm Instrument is traded independently from the 4:00pm Instrument. This specification does not assume or require an Order Book, a Double Auction, or any other mechanism in the Matching Engine.~~

~~The Resource definition is extensible using standard UML techniques (subclassing); however CTS 1.0 uses only this base definition.~~

~~These terms are summarized in .~~

~~Table -: Definitions of CTS Market terms~~

~~Transactive Entity~~	~~Definition~~
~~Resource~~	~~A measurable commodity, substance, service, or force, whose value is determined by time of delivery.~~
~~Product~~	~~A Resource defined by size/granularity of the Resource and by the granularity of time. A Market is defined by its Product. Example 1: electric power in 10 kW units delivered over an hour of time. Example 2: electric energy in 1 kWh units delivered over a quarter hour.~~

~~Transactive Entity~~	~~Definition~~
~~Instrument~~	~~A Product instantiated by a particular begin time. Example: the Product beginning at 9:00 AM on April 3. An Instrument is Tendered to a Market with specific quantity and price.~~
~~Party~~	~~An Actor or a set of coordinating Actors that buys or sells Instruments in a CTS Marketplace. A Party may be described by a specific role in a specific interaction, such as Party or Counter Party. For semantic and privacy issues, see Section 2.2.3 below.~~
~~Market~~	~~A Facet where Parties trade a Product using tenders submitted to buy or sell an Instrument.~~
~~Marketplace~~	~~A Marketplace names a set of Markets accessible to a Party. The Marketplace Facet supplies limited information common to all Markets in the Marketplace. The Facet also enumerates all Resources and Products available in the Marketplace, as well as a directory of the Markets for each Resource.~~ ~~CTS differs from EMIX in adding a distinction between Market and Marketplace, while making no assumption about how the distinction is implimented or even whether Markets and Marketplaces share common ownership. The [EMIX] Market Context, identified by a URI, is akin to the CTS Marketplace.~~
~~Market Context~~	~~A URI identifying an individual Market, as defined in EMIX.~~
~~Marketplace Context~~	~~A URI identifying a Marketplace, as defined in the EMIX Market Context.~~
~~Matching Engine~~	~~There are many Market processes to exchange offers and reach agreements on transactions. Different parts of the same Marketplace MAY employ different Market processes. We term each of these processes a Matching Engine. This specification uses the term Matching Engine only to support narrative description. The specific processes, structures, and algorithms of Matching Engines are out of scope.~~

~~2.41.1 Common Transactive Services Roles~~

~~Actors interact through Facets. The specification makes no assertions about the behaviors, processes, or motives within each Actor. A particular Actor may use all Facets, a subset of Facets, or even a single Facet. This specification defines Facet messages and interactions.~~

~~**[EI]** defines contracts between Actors as services with defined messages and interactions. All **[EI]** services map to CTS Facets. Nearly all Facets defined in CTS are services as defined in **[EI]**. CTS defines two additional Facets for market operations not derived from the Services in [EI], namely Position and Delivery, as well as two facets for Market discovery, the Marketplace and Market Facets. CTS does not require a conforming transactive energy market to use every Facet.~~

~~2.4.11.1.1 Parties as Market Participants~~

~~The Common Transactive Services (CTS) defines interactions in a Resource Market. This Resource Market is a means to make collaborative decisions that allocate power or other Resource over time. We follow [EI] and financial markets by terming market participants as Parties.~~

~~A Party can take one of two Sides in Transaction:~~

- ~~Buy, or~~
- ~~Sell~~

~~A Party selling an Instrument takes the Sell Side of the Transaction. A Party buying [an Instrument] takes the Buy Side of the Transaction. The offering Party is called the Party in a Transaction; the other Party is called the Counterparty~~

~~From the perspective of the Market, there is no distinction between a Party selling additional power and Party selling from its previously acquired position. An Actor representing a generator would generally take the Sell side of a transaction. An Actor representing a consumer generally takes the Buy side of a transaction. However, a generator may take the Buy Side of a Transaction to reduce its own generation, in response either to changes in physical or market conditions or to reflect other commitments made by the actor. A consumer may choose to sell from its current position if its plans change, or if it receives an attractive price. A power storage system actor may choose to buy or sell from Interval to Interval, consistent with its operating and financial goals.~~

~~A Party may represent a single actor, or the roles (see Facets, below) of a single Party may distributed across multiple Actors.~~

~~We do not specify how to manage delivery of the Resource.~~

~~2.4.2 Facets in CTS~~

~~This specification refers to coherent se of interactions, that is, closely related requests, responses, as Facets. A Facet sends and receives defined messages to interact with other Actors that expose the same Facet. An Actor in a CTS-based system of systems may expose all Facets, a single Facet, or any collection of Facets. A particular Market may use some or all named Facets. A participant in a Market must include Actors supporting each Facet required in that Market; there is no requirement that each Actor supports all these Facets.~~

~~Each Facet named below groups a mandatory set of related messages and interactions. Detailed descriptions of each facet begin in Section .~~

~~Table -: Facets Defined in CTS~~

~~Facet~~	~~Definition~~
~~Marketplace~~	~~The Marketplace Facet exchanges information about the Marketplace and its Products and Markets. A Party registers with a Marketplace through this Facet. A Party may query the Marketplace to discover the Resources and Products traded in a Marketplace. When a Marketplace includes multiple Products, the Party needs to know where to find the Market for each Product. While a Marketplace may change slowly over time, the Marketplace facet can generally be viewed AS conveying static information.~~
~~Market~~	~~A Market Facet exchanges information for trading a particular Product in a particular Market. Parties submit Tenders to a Market, and the Market notifies the Parties of Transactions. A Market Facet contains a Matching Engine that matches Tenders to buy and Tenders to sell. The Market~~

~~Facet~~	~~Definition~~
	~~Facet conveys information as to how the Market matches orders, which may change the strategies used by a Market participant. Some Markets MAY register transactions privately agreed to among Parties.~~ ~~See Section 8 Market Facet~~
~~Registration~~	~~A Party must Register with a Marketplace to participate in the Markets in that Marketplace.~~ ~~See Section 6, Party Registration Facet.~~
~~Tender~~	~~Tenders are actionable offers to buy or to sell an Instrument at a given price. Tenders go to the Market and are generally private. It is possible to request that a Tender be advertised to all Parties in the Market. Note: a Tender for one side MAY match more than one Tender on the other side, which could generate multiple Transactions.~~ ~~See Section 9, Tender Facet.~~
~~Transaction~~	~~A Transaction records a contract when a Tender to buy and a Tender to sell are matched. Each Party is notified of the creation of the Transaction. Note: a Tender for one side MAY match more than one Tender on the other side, which would generate multiple Transactions.~~ ~~See Section 10, Transaction Facet.~~
~~Position~~	~~At any moment, a Party has a position which represents the cumulative amount of an Instrument that an actor has previously transacted for within a bounding time interval. A Position for an Instrument reflects the algebraic sum of all quantities previously bought or sold.~~ ~~See Section 11, Position Facet.~~
~~Delivery~~	~~It is simplest to think of Delivery as a meter reading, although that meter may be virtual or computed. Some implementations may compare what was purchased or sold with what was delivered. What a system does after this comparison is out of scope.~~ ~~See Section 12, Delivery Facet.~~
~~Quote~~	~~A Quote is a non-actionable indication of a potential price or availability of an Instrument.~~ ~~**[EI]**~~ ~~defines the EiQuote service. This specification extends the Quote to include forecasts, information about completed Transactions, and other Market information.~~ ~~See Section 13 Market Information – the Quote and Ticker Facets~~
~~Ticker~~	~~Named for the stock ticker, best known for its printed output the ticker tape. A ticker provides public information about transactions over time.~~ ~~See Section 13 Market Information – the Quote and Ticker Facets~~

~~Each of these facets includes multiple messages which are described starting in Section . Sometimes one facet precedes the use of another facet, as Tenders may initiate messages for the Transaction Facet.~~

~~2.4.3 Party and Counterparty in Tenders and Transactions~~

~~The Party in a Tender is offering to buy or sell. The PartyID in a Tender should always reference the Party that is tendering.~~

~~When the Market recognizes Tenders that match each other, however defined, the Market generates a Transaction that represents a contract between the buyer and the seller. This Transaction includes a Party and a Counterparty.~~

~~See Section , “” for a discussion of Market Information.~~

~~2.51.1 Responses~~

~~This section re-iterates terms and simplifies models from [EI]. That specification is normative. The form of the Response is common across all Facets.~~

~~Table 2 : Responses~~

~~Attribute~~	~~Meaning~~
~~Request ID~~	~~A reference ID which identifies the artifact or message element to which this is a response. The Request ID uniquely identifies this request and can serve as a messaging correlation ID~~[5]~~.~~
~~Response Code~~	~~The Response Code indicates success or failure of the operation requested. The Response Description is unconstrained text, perhaps for use in a user interface.~~ ~~The code ranges are those used for HTTP response codes,~~[6] ~~specifically~~ ~~1xx: Informational - Request received, continuing process~~ ~~2xx: Success - The action was successfully received, understood, and accepted~~ ~~3xx: Pending - Further action must be taken in order to complete the request~~ ~~4xx: Requester Error - The request contains bad syntax or cannot be fulfilled~~ ~~5xx: Responder Error - The responder failed to fulfill an apparently valid request~~

~~The Most operations have a response.~~

[5] As an example of the *Correlation Pattern* for messages

[6] See e.g. https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

~~3 Common Semantic Elements of CTS~~

~~The messages of CTS use a few common elements. These elements are derived from and compatible with definitions in **[WS-Calendar]**, **[EMIX]**, and in **[EI]**.~~

~~3.1 Semantic Elements from WS-Calendar~~

~~Time and Duration are the essential elements of defining an Instrument as well as for interacting with a Market. A Stream **[Streams]** is a series of back-to-back intervals each with its own associated information. Section defines the *CTS Stream* as a conformant specialization of **[Streams]**, integrating information that is outside of a Stream data structure but associated with a Stream.~~[7]

~~Table -: CTS Elements from WS-Calendar~~

~~**Attribute**~~	~~**Meaning**~~
~~Duration~~	~~Duration is used to define Products, as in "Power can be purchased and there is a one-hour (duration) market for Power".~~ ~~Duration is also used in Delivery to specify the period over which Delivery is measured, as in "How much Power was delivered in the 4 hours beginning with the Begin Date-Time?"~~
~~Offset~~	~~An offset (expressed as a WS-Calendar Duration) that some markets MAY use to transfer trading away from hourly boundaries.~~ ~~A power distribution entity may experience disruption if there is a big price change on the hour. Offset enables a Market to trade, for example, 3 minutes after the hour. See also Market Facet~~
~~Begin Date-Time~~	~~Begin Date-Time fully binds a Duration into an Interval. When applied to a Product, the Begin Date-Time defines an Instrument., i.e., something that is directly traded in the Market.~~
~~Expiration Date-Time~~	~~Expiration is used to limit the time a Tender is on the Market. There is an implicit expiration for every Tender equal to the Begin Date-Time of the Instrument. Expiration Date-Time is needed only if the requested Expiration is prior to the Begin Date-Time of the Instrument.~~
~~Interval~~	~~An Interval in CTS is a Duration with a Begin Date-Time. This maps to what WS-Calendar names a "Scheduled Interval".~~

~~3.2 Semantic Elements from EMIX~~

~~EMIX defines the specification of commodity goods and services whose value is determined by time and location of delivery. EMIX defines an "Item" by what is sold in a Market, when it is sold, what the units are, and what the standard trade size is. EMIX further defines how to communicate the date and time of delivery for that commodity to define a unique Product that can be bought and sold in a Market.~~

~~In CTS, we maintain the semantics of EMIX while giving name to each refinement of the information. These names are the Resource (what is sold), the Product (how the Resource is packaged into a size~~

[7] Including Resource Designator, Stream Start, and Decimal Fraction

~~and Duration for sale), and the Instrument (a Product sold at a specific time). Instruments are what are bought and sold in CTS markets.~~

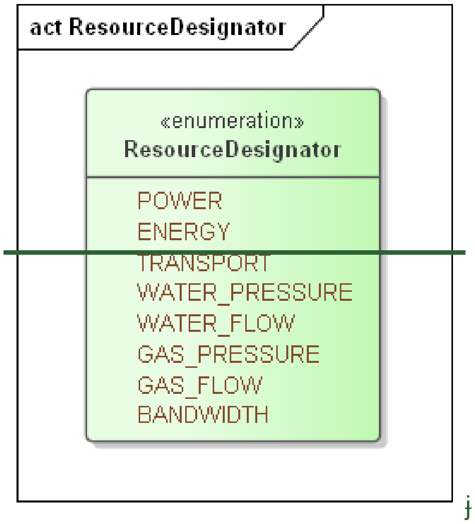
~~3.2.1 Defining Resource~~

~~Here we define a Resource as a commodity that is bought or sold in a CTS Marketplace. A Party can query a Marketplace to discover the Resources that can traded in each of the Markets in the Marketplace.~~

~~Table -: Defining the Resource~~

~~Attribute~~	~~Meaning~~
~~Resource~~	~~A Resource consists of a Resource Designator, a Resource Name and a Resource [Item] Description.~~
~~Item Description~~	~~The Item Description is a common name, as defined in EMIX~~
~~Item Unit~~	~~Item Unit is the unit of measure for the Resource.~~
~~Attributes~~	~~Optional elements that further describe the Resource, as in hertz and voltage~~

~~A Resource Designator is an extensible enumeration. The standard enumeration is defined in :~~



~~Figure - Resource Designator Extensible Enumeration~~

~~3.2.2 Defining Product~~

~~The Product is a Resource packaged for Market. The size and duration of the Resource define what is, in effect, the "package size" for the commodity. A Marketplace may offer multiple Products for the same Resource.~~

~~Table -: Defining the Product~~

~~Attribute~~	~~Meaning~~
~~Product~~	~~Abstract Base for all defining all Products. The core of each Product is the Resource, as referenced by the Resource Designator.~~

~~Attribute~~	~~Meaning~~
~~Scale~~	~~Exponent that specifies the size of the Resource Unit. For example, a Product denominated in Megawatts has a Scale of 6.~~
~~Size~~	~~An integer "chunking" the Product, i.e., the Product could be traded in units of 5 kW, a size of 5 and a scale of 3.~~
~~Warrant~~	~~Undefined element of a Product that restricts the Product beyond the Resource definition. For example, it is possible to trade in power designated to be Neighborhood Solar Power. In CTS, Products that are identical other than the Warrant are traded in different Markets within the same Marketplace.~~

~~Products with differing Warrants are different Products and therefore traded in different Markets.~~

~~As non-normative examples, if an Actor wishes to buy energy with a *Green Warrant* (however defined) then the Actor, not the Market, is responsible for defining its trading strategies if the warranted Product is not available. Similarly, an Actor that wishes to buy or sell Neighborhood Solar Power is responsible for submitting Tenders that expire in time to make alternate arrangements, or in time to cancel Tenders before fulfillment. This specification establishes no expectation that the Market engine address these issues automatically.~~

~~Warrants are defined in **[EMIX]**, and are permitted in CTS to support this complexity if desired, but not described in this specification.~~

~~3.2.3 Market Semantics from EMIX~~

~~EMIX defines vocabulary used in market messages and interactions.~~

~~Table -: Market-related elements from EMIX~~

~~Attribute~~	~~Meaning~~
~~PartyID~~	~~The Marketplace-based ID of an actor participating in Markets, particularly the actor originating a Tender, Quote, or Contract.~~
~~Counter PartyID~~	~~The Marketplace-based ID of an actor participating in Markets, particularly the actor taking the other side of a contract from the Party. See Section 2.2.3.~~
~~Side~~	~~An indication of what a Party offers in a Tender or other message, i.e., "Buy" or "Sell".~~
~~Expiration Date-Time~~	~~Expiration is used to limit the time a Tender is on the Market. There is an implicit expiration for every Tender equal to the Begin Date-Time of the Instrument. Expiration Date-Time is needed only if the requested Expiration is prior or subsequent to the Begin Date-Time of the Instrument.~~
~~Market Context~~	~~In EMIX, the Market Context is simply a URI to name a Market, and need not be resolvable. CTS distinguishes between a Marketplace, where many Products may be sold and the Market, where a specific Product is sold. See Section 6. "Marketplace Facet".~~

~~Standard Terms~~	~~Standard Terms are the machine-readable information about a Marketplace or Market, and the interactions it supports. In CTS, the Standard Terms include an enumeration of the Products and their respective Markets tradable in this Marketplace. See Section 6, "Market Facet".~~

~~EMIX does not define how an Actor discovers the Standard Terms in a Marketplace. CTS defines the Marketplace Facet and the Market Facet to discover and expose Products and Standard Terms.~~

~~4 Basic Interaction and Terminology~~

~~4.1~~1.10 ~~Structure of Common Transactive Services~~Naming of Messages and Operations

~~The Common Transactive Services presented in this specification are described in the following sections, and are~~

- ~~Marketplace Facet—characteristics and to know what Products and Instruments can be traded~~
- ~~Party Registration Facet—identification of actors within a Market or Marketplace~~
- ~~Tender Facet—make offers to buy and sell Instruments~~
- ~~Transaction Facet—for expressing transactions.~~
- ~~Position Facet—Describe what has been previously bought or sold~~
- ~~Delivery Facet—Request data on actual deliveries~~
- ~~Market Information—the Quote and Ticker Facets~~

~~We include UML definitions for the standard payloads for service requests, rather than the service, communication, or other characteristics. In Section we describe standard serialization for the CTS standard payloads; additional bindings may be used by conforming implementations.~~

~~Transactive Services in EI define and support the lifecycle of transactions from preparation (registration) to initial Tender to final settlement. The phases described in EI are in the following table with the CTS Facets in Column 2.~~

~~Table -: Mapping CTS Facets to EI Phases~~

~~EI Phase~~	~~CTS Facet(s)~~
~~Registration (and Market discovery)~~	~~Party Registration Facet~~ ~~Marketplace Facet~~ ~~Market Facet~~
~~Pre-Transaction~~	~~Quote Facet~~ ~~Tender Facet~~
~~Transaction~~	~~Transaction Facet~~
~~Post-Transaction~~	~~Position Facet~~ ~~Delivery Facet~~ ~~Ticker Facet~~

~~4.2 Naming of Services and Operations~~

The naming of ~~services~~messages and operations and ~~service operation~~message payloads follows the pattern defined in [EI]. Services are named starting with the letters ***Ei*** following the Upper Camel Case convention. Operations in each service use one or more of the following patterns. The first listed is a fragment of the name of the initial service operation; the second is a fragment of the name of the response message which acknowledges receipt, describes errors, and may pass information back to the invoker of the first operation.

Create—Created	An object is created and sent to the other Party.
Cancel—Canceled	A previously created request is canceled.

For example, to construct an operation name for the Tender ~~facet~~Facet, "Ei" is concatenated with the name fragment (verb) as listed. An operation to cancel an outstanding Tender is called *EiCancelTender*.[8]

Facets describe what would be called services in a full Service-Oriented Architecture implementation, as we do not define SOA services, but only imply and follow a service structure from [EI].

[8] This pattern was developed and is used by IEC Technical Committee 57 (Power Systems).

2 Overview of Common Transactive Services

CTS provides for the exchange of resources among parties which represent any provider or consumer of energy (e.g., a distributed energy resource). CTS makes no assumptions as to their internal processes or technology.

Systems use the common transactive services to interoperate in transactive resource markets. A transactive resource market balances the supply of a resource over time and the demand for that resource by using a market specifying the time of delivery.

Although the Common Transactive Services are a profile of Energy Interoperation, the CTS focus is markets and trading. The language used in the Energy Interoperation specification was developed with extensive input from economists, regulators, and participants in highly regulated markets. This profile strives instead for the language of markets and traders.

This specification supports agreements and transactional obligations, while offering flexibility of implementation to support specific approaches and goals of the various participants.

~~4.3 Payloads and Messages~~

~~We define only the payloads; the particular networking technique and message structure is determined by the applications sending and receiving CTS payloads.~~

2.1 ~~While the payloads are logically complete with respect to the SOA~~Parties

The Common Transactive Services (CTS) defines interactions in ~~[EI],~~ a resource market. This Resource Market is a means to make collaborative decisions that allocate power or other resources over time. We follow [EI] and financial markets by calling market participants "Parties".

A Party may represent a single actor, or the ~~payloads~~roles (see Facets, below) of a single Party may be ~~exchanged by any means; such exchanges are below the semantic level of~~ distributed across multiple Actors. When the market recognizes tenders that match each other, however decided, the market generates a transaction that represents a contract ("Trade") between the buyer and the seller. This transaction includes a party and a counterparty.

2.2 Trading semantics from FIX

The FIX Community divides messages into Pre-Trade, Trade, and Post-Trade Messages.

Pre-Trade messages convey information that traders need to discover how to use the market and to develop a strategy to buy and sell successfully. Pre-trade messages include advertisements and announcements ("Tickers") of offers and contracts in the market, and negotiations between parties ("Quotes"). Other pre-Trade messages describe how the market itself works and what a Party can expect when interacting with the market.

Trade messages include submitting and cancelling orders ("Tenders") to the market and executing contracts ("Transactions") when orders to sell match (however defined) orders to buy.

Post-trade messages include settlement and position management.

For narrative purposes, this specification~~.~~ begins with the Trade facets, Tenders and Contracts. It then discusses the post-trade facets of Delivery and Position. This covers all the functions in some transactive resource markets. This specification then describes Negotiation, an optional pre-Trade facet. It next describes the pre-trade market data reports ("Tickers") that inform an Actor about the activities of other participants. The pre-trade Market Instrument Report facet provides summary information about Tenders currently held in the market. Finally, the pre-trade Market Structure facet conveys how a Trader may interact with the market, which includes how to find each Facet and which messages this market supports.

An Actor interacting with the market would first discover the market structure, subscribe to Tickers relevant to its interest, and then use the facets and messages that are permitted in this market to Trade.

A Party MAY not understand negotiation, or MAY skip subscribing to Tickers, but any party MUST be able to Trade.

2.2.1 Parties and Orders

In Energy Interop as in FIX, a trade is executed between two parties. While Energy Interoperation acknowledges only a Party and a Counterparty, FIX is more semantically rich.

What Energy Interoperation (and this specification) terms Tenders, FIX terms orders. An order that is on the book in the market is a Resting or Passive order. An order that enters the market to match a Resting order is the Initiating or Aggressive order. Passive orders increase market liquidity. Aggressive orders decrease market liquidity. Regulators of financial markets are often interested in liquidity and in the ratios of Aggressive to Passive orders.

When it makes the discussion clearer, this specification uses the terms Resting, Passive, Initiating, and Aggressive as they are used in financial markets.

2.2.2 Instruments

4.4 Financial Markets trade financial instruments. CTS borrows this language from FIX. See Section 3, Market Semantics: Resource, Product, Instrument~~Description of the Facets and Payloads~~

~~The sections below provide the following for each service:~~

- ~~Facet description~~
- ~~Table of Payloads~~

~~Interaction patterns~~, for ~~payload~~a discussion.

2.2.3 Market Crossing

Market Crossing refers to either the opening or to the closing of a market or market segment. A traditional exchange ~~in graphic form,~~opens in the morning and closes in the afternoon. Tenders are not matched prior to market opening or after the market close.

In many markets, parties wishing to trade pay close attention to prices and volumes in the period around closing. Many traders prefer not to trade close to a crossing because it is a period of high price volatility on a market. Many markets announce a “closing price” and an anticipated “opening price”, even as no transaction may have occurred at either of those prices.

Transactive resource markets may have regulatory time limits on trading. Some electricity markets have banned transactions more than a day prior to delivery. CTS traders must be able to understand the local rules and adjust their trading tactics without human intervention. A Market MAY accept Tenders prior to the opening of the Market Segment or Instrument. Transactive market researchers have used tenders submitted prior to opening to generate opening prices in black-start scenarios. Others have used trade residue, which is the tenders left in the market after closing to seed real-time prices for unplanned energy use.

Consider a utility that provides day-ahead hourly pricing from the local bulk power market to retail customers. The prices for tomorrow may be posted at 9:00 AM each day. These prices may be good until 3:00 PM each day. The market may begin accepting tenders an hour before opening. This describes a market segment with two crosses each day, opening at 9:00 AM and closing at 3:00 PM. After 3:00 PM orders may be processed as a normal forward market using an order book to process tenders into transactions.

As transactive resource markets are in essence markets in time of delivery, individual instruments can be considered to open and close as well. In a continuously open market segment, a rule might prevent trading more than 24 hours in advance. In that same market, an instrument for delivery of a resource between 10:00 AM and 11:00 AM can no longer be traded at noon.

2.2.4 Markets and Venues

Systems use the common transactive services to interoperate in transactive resource markets. A transactive resource market balances the supply of a resource over time and the demand for that resource by using a market specifying the time of delivery.

A Market MAY be divided up into different venues wherein different products are traded, perhaps with different rules. Following the FIX Protocol, we term these Market Segments, and we use the FIX classification Venue Type to name the market activities of each Segment. A Market may have one or many Market Segments.

2.3 Common Transactive Services Roles

Actors interact through messages submitted to Facets. The specification makes no assertions about the behaviors, processes, or motives within each Actor. A particular Actor may use all Facets, a subset of Facets, or even a single Facet. ~~EI normative~~This specification groups similar messages by Facet messages and interactions~~ and UML Sequence Diagrams [UML].~~.

2.3.1 Parties as Market Participants

The Common Transactive Services (CTS) defines interactions in a Resource Market. This Resource Market is a means to make collaborative decisions that allocate power or other Resource over time. We follow [EI] and financial markets by calling market participants “Parties”.

A Party can take one of two Sides in Transaction:

- Buy, or
- Sell

A Party selling an Instrument takes the Sell Side of the Transaction. A Party buying an Instrument takes the Buy Side of the Transaction. The initiating Party is called the Party in a Transaction; the other Party is called the Counterparty.

From the perspective of the Market, there is no distinction between a Party selling additional power and Party selling from its previously acquired position. An Actor representing a generator would generally take the Sell side of a transaction. An Actor representing a consumer generally takes the Buy side of a transaction. However, a generator may take the Buy Side of a Transaction to reduce its own generation, in response either to changes in physical or market conditions or to reflect other commitments made by the actor. A consumer may choose to sell from its current position if its plans change, or if it receives an attractive price. A power storage system actor may choose to buy or sell from Interval to Interval, consistent with its operating and financial goals.

We do not specify how to manage delivery of the Resource.

2.3.2 Party and Counterparty and Transactions

The party in a tender is offering to buy or sell. The PartyID in a Tender should always reference the Party that is tendering.

When the Market recognizes tenders that *match* each other (however defined), the market generates a Transaction that represents an agreement between the buyer and the seller. This transaction includes a Party and a Counterparty.

2.3.3 Facets in the CTS Specification

This specification refers to a coherent set of interactions, that is, closely related requests and responses, as Facets. An Actor sends and receives defined messages through its Facet to interact with other Actors that expose a complementary Facet. An Actor in a CTS-based system of systems may expose all Facets, a single Facet, or any collection of Facets. A particular Market may use some or all named Facets. A participant in a Market must include Actors supporting each Facet required in that Market; there is no requirement that each Actor supports all these Facets.

- ~~Normative information model using [UML] for key artifacts used by the facet~~
- ~~Normative operation payloads using [UML] for each interaction~~

~~4.5 Responses~~

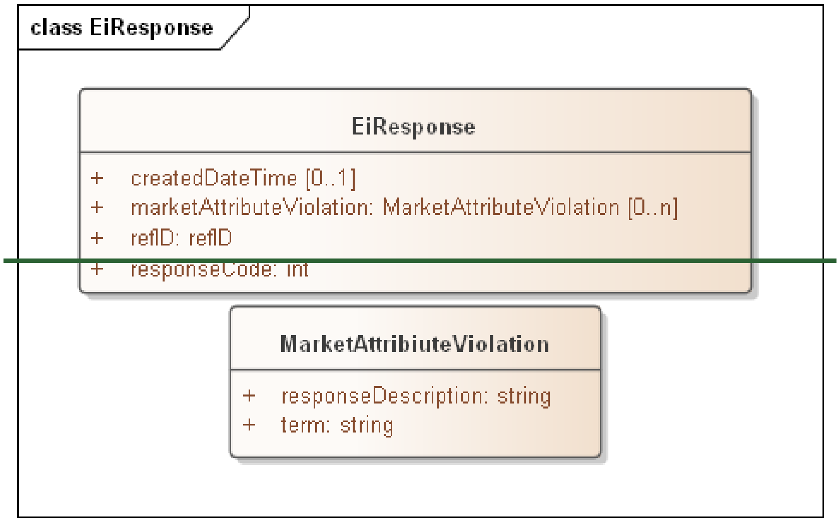
~~Responses may need to be tracked to determine whether an operation succeeds or not. This may be complicated by the fact that any given Transaction may involve the transmission of one or more information objects.~~

~~An EiResponse returns the success or failure of the entire operation, with possible detail included in responseTermsViolated (see Section).~~

~~It is MANDATORY to return responses[9] indicating partial or complete success or failure.~~

~~The class diagram in shows the generic CTS response.~~

~~CTS uses a simplified version of EiResponseType from EI, deleting ArrayOfResponseTermsViolated and responseDescription (to zero, that is, not passed). Response Terms Violated is renamed *Market Attribute Violation.*~~


class EiResponse
EiResponse
+ createdDateTime [0..1]
+ marketAttributeViolation: MarketAttributeViolation [0..n]
+ refID: refID
+ responseCode: int
MarketAttribiuteViolation
+ responseDescription: string
+ term: string

~~Figure -: Example of generic response object~~

~~There is no exhaustive list of all possible Response Codes. More detail on Response Codes is in Section .~~

~~The Response Codes are intended to enable even the smallest device to interpret Response. This specification uses a pattern consisting of a 3-digit code, with the most significant digit sufficient to interpret success or failure. This pattern is intended to support that smallest device, while still supporting more nuanced messages that may be developed.[10]~~

~~The only defined value in EI after the leading digit of the Response Code is 00. Conforming specifications may extend these codes to define more fine-grained response codes. These should extend the pattern above; for example, a response code of 403 should always indicate Requester Error. Response codes not of the form x00 MAY be treated as the parallel x00 response.~~

~~As an example, consider a request to quote 13.5 kW at three minutes offset for 17 minutes where the market characteristics and its product include 10kW granularity, zero offset, and five minute duration. The terms in the Market Attribute Violation therefore include at least these violations:~~

[9] This contrasts with EI, where it is not mandatory to return any responses if the entire EiCancelTender service operation was completed successfully. The pattern in EI is to return those that have failed (required) and those that succeeded (optional).

[10] This is parallel to HTTP response codes.

- ~~T_GRAIN, 5m~~
- ~~Q_GRAIN, 10kW~~
- ~~OFFSET, 0~~

~~The definition of the respective terms is in *Section* , .~~

~~5 CTS Streams~~

~~Aside from registration and market information, Payloads in CTS are derived from and conformant with WS-Calendar Streams. The essence of Streams is that for a series of consecutive Durations over time, called Intervals, invariant information is in the header or preface to the stream, and only the varying information is expressed in each Interval.~~

~~For CTS, this means that a Product is fully described in the header, and only the elements that vary, such as the Price or the Quantity, are expressed in the intervals.~~

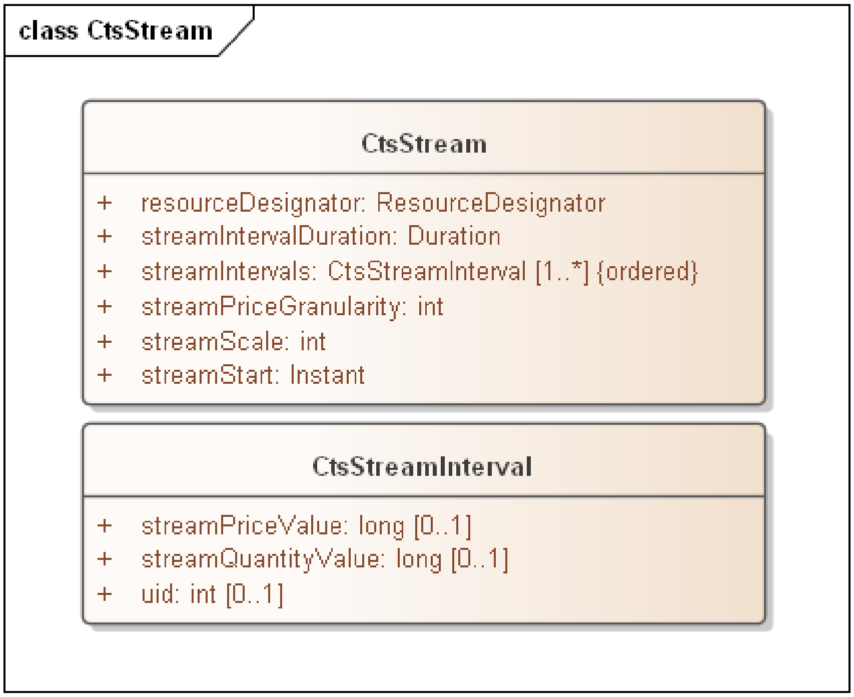
~~CTS Streams use this same format even when the Intervals contain only a single Interval.~~

~~In addition, CTS Streams include energy-market elements that are outside the Streams standard but follow the pattern of referrals as defined in [Steams] conformance.~~

~~CTS Streams have neither interaction patterns nor payloads, as they are a common abstract information model used to define the messages used in Facet messages.~~

~~5.1 Information Model for CTS Streams~~

~~The CTS Stream is defined as follows. The elements from **[Streams]** have been flattened into the CTS Stream, and the Stream Interval and payload flattened into a streamPayloadValue and the internal local UID for the stream element.~~


class CtsStream
CtsStream
+ resourceDesignator: ResourceDesignator
+ streamIntervalDuration: Duration
+ streamIntervals: CtsStreamInterval [1..*] {ordered}
+ streamPriceGranularity: int
+ streamScale: int
+ streamStart: Instant
CtsStreamInterval
+ streamPriceValue: long [0..1]
+ streamQuantityValue: long [0..1]
+ uid: int [0..1]

~~Figure -: CTS Stream Definition~~

~~As with [Streams], CTS Stream Intervals are ordered, that is the sequence of intervals is essential. Some serialization specifications, notably XML, do not require that order be preserved when deserializing a list. The UID enables proper ordering of the Stream Intervals if order is not preserved. Since conformant CTS implementations need not be owned by the same implementer, and may pass through multiple translations, the UID property is required.~~

Detailed descriptions of each facet begin in Section 4~~The following tables describe the attributes for CTS Streams and Stream Intervals.~~

.

Table 1-1: Facets Defined in CTS ~~Stream Attributes~~

Facet	Description
Registration	A Party must Register with a Market to participate in the Market Segments in that Market. See Section 4, "Party Registration Facet".
Tender	Tenders are actionable offers to buy or to sell an Instrument at a given price. Tenders are sent to the Market Segment and are generally private. It is possible to request that a Tender be advertised to all Parties in the Market. See Section 5, "The Tender Facet".
Transaction	A Transaction records the trade when a Tender to buy and a Tender to sell are matched. Each Party is notified of the creation of the Transaction. Note: a Tender for one side MAY match more than one Tender on the other side, and could generate multiple Transactions, potentially at different prices. See Section 6, "The Transaction Facet".
Position	At any moment, a Party has a position which represents the cumulative amount of Instruments that the Party has previously transacted for within a bounding time interval across all Segments in the Market. A Position for an Instrument reflects the algebraic sum of all quantities previously bought or sold. See Section 7, "The Position Facet".
Delivery	It is simplest to think of Delivery as a meter reading, although that meter may be virtual or computed. Some implementations may compare what was purchased or sold with what was delivered. What a system does after this comparison is out of scope. See Section 8, "The Delivery Facet".
Negotiation	Negotiation covers messages that may lead to a Tender that will be accepted. Negotiation includes Requests for Quotes (RFQs), Indications of Interest (IOI), and Quotes. See Section 9, "The Negotiation Facet".
Tickers	A Ticker is a continuous live view of market interactions-consider a ticker tape. A Ticker is one form of Market Subscriptions as defined by FIX. See Section 11, "Ticker "
Market Instrument Summaries	A Market Instrument Summary is a compressed or summarized variant of Market Data as defined by FIX. See Section 12, "Instrument Market Data".
Market Structure	The Market Facet exchanges information about the Market and its Products and Market Segments. An Actor may query the Market to discover the Resource and Products traded in a Market. While a Market trades a single Resource, it may consist of multiple Market Segments trading multiple Products. See Section 13, Market Structure Subscription

Each of these facets includes multiple messages which are described starting in Section 3.2 below. Sometimes the use of one facet precedes the use of another facet, as Tenders may initiate messages that result in messages for the Transaction Facet.

2.4 Responses

This section re-iterates terms and simplifies models from [EI]. That specification is normative. The form of the Response is common across all Facets.

714 Table 1-2: Responses

Attribute	Meaning
Created Date Time	Timestamp for creation of this response
Market Attribute Violation	Market and Segment attributes violated in the referenced request. See Section 13 Market Structure Subscription and Table 13-2 and Table 13-3.
In Response To	A reference ID which identifies the artifact or message element to which this is a response. The Request ID uniquely identifies this request and can serve as a messaging correlation ID[11].
Response Code	The Response Code indicates success or failure of the operation requested. The Response Description is unconstrained text, perhaps for use in a user interface. The code ranges are those used for HTTP response codes,[12] specifically: 1xx: Informational - Request received, continuing process. 2xx: Success - The action was successfully received, understood, and accepted 3xx: Pending - Further action must be taken in order to complete the request 4xx: Requester Error - The request contains bad syntax or cannot be fulfilled 5xx: Responder Error - The responder failed to fulfill an apparently valid request
Response Description	A string describing the response, e.g. “Duration doesn’t match Segment configured Duration”

715 Most messages elicit a response. Information-only messages, as in Tickers, do not.

[11] As an example of the *Correlation Pattern* for messages

[12] See e.g. https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

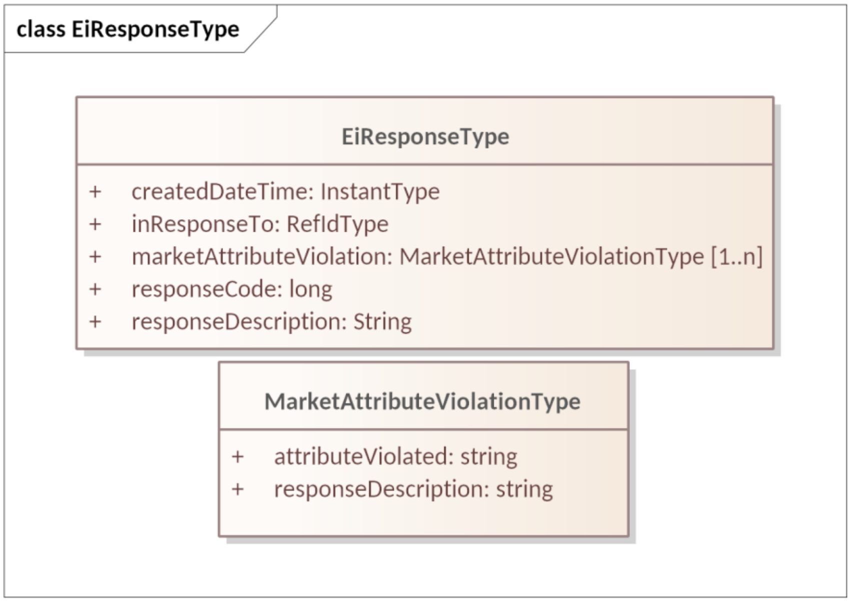
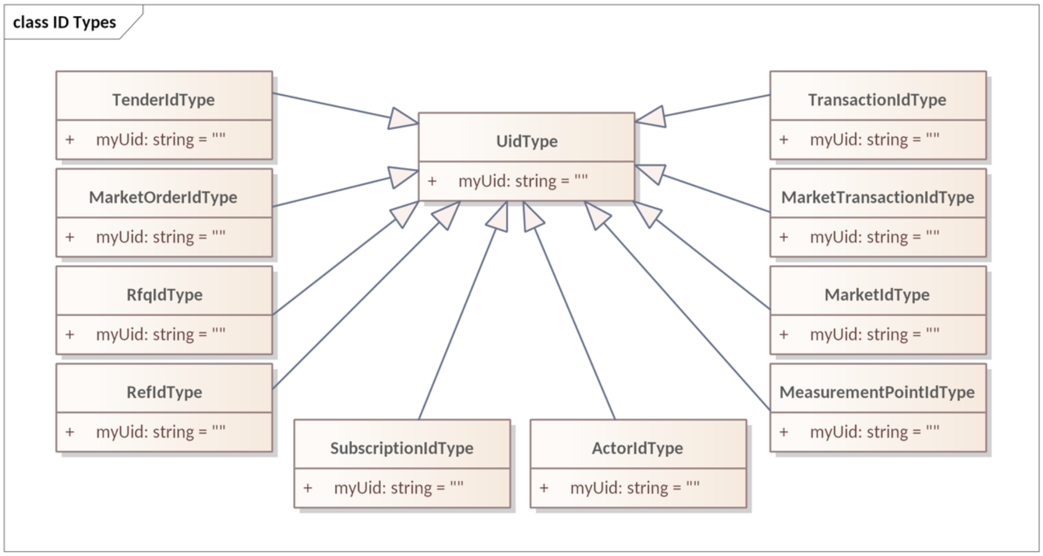

class EiResponseType
EiResponseType
+ createdDateTime: InstantType
+ inResponseTo: RefIdType
+ marketAttributeViolation: MarketAttributeViolationType [1..n]
+ responseCode: long
+ responseDescription: String
MarketAttributeViolationType
+ attributeViolated: string
+ responseDescription: string

Figure 1-1~~Attribute~~	~~Meaning~~	~~Notes~~
~~Resource Designator~~	~~An item from an enumeration that indicates the Resource for the Product and Market~~	~~The Resource Designator in a Market should match the Resource Designator indicated in the Marketplace~~
~~Stream Scale~~	~~The Scale is the exponent that determines the size of the Resource.~~	~~For example, if Scale is 3 and the Resource is Watts, then the value is in kW. If the Scale is 6, then the value is in MW.~~
~~Stream Interval Duration~~	~~The duration for each of the contiguous Stream Intervals~~	~~This completes the Product definition of a Resource at a Scale and Size delivered over a Duration.~~
~~Stream Price Granularity~~	~~Price granularity expressed as an exponent. Applies to all Intervals in the Stream. Not required for all Facets.~~	~~For example, if the price granularity is -3, and the value is 1500, the price is 1.500 currency units.~~
~~Stream Start~~	~~The Start Date and Time for a bound CTS Stream~~	~~See WS-Calendar Date-Time in Section 3.1.~~
~~Stream Intervals~~	~~The ordered set of Stream Intervals~~	~~The set of Intervals is ordered by means of a local UID which is concatenated with the Stream UID as described in [Streams] and in [EI]~~

UML Class Diagram of EiResponseType and MarketAttributeViolationType

2.5 Identities

Table 1-3 ID UML Class Diagram of ID Types in CTS



3 Market Semantics: Resource, Product, Instrument, and Streams

The messages of CTS use a few common elements. These elements derive from and are compatible with definitions in [WS-Calendar], [EMIX], and in [EI].

Every CTS-based market offers the exchange of a specific resource. Each CTS market segment is a venue for trading a single product, which is a resource packaged for sale. All tenders and transactions are for instruments, which are products scheduled for delivery at a specific time.

3.1 Resource, Product, & Instrument

We define a Resource as a commodity whose value depends on time of delivery. A Party subscribes (see Section 10) to a Market to discover the Resource that is traded in the market, and the Products available in different Market Segments. (See Section 13.2, “Market Definition”) A Party can then trade Instruments, a Product at a specific time, in a Market Segment. This specification leaves Market Definition until the end of the specification, as the meaning and import of the terms used to define each Segment are first described in the trading process.

Figure 3-1 illustrates the relationship between Resource, Product and Instrument.

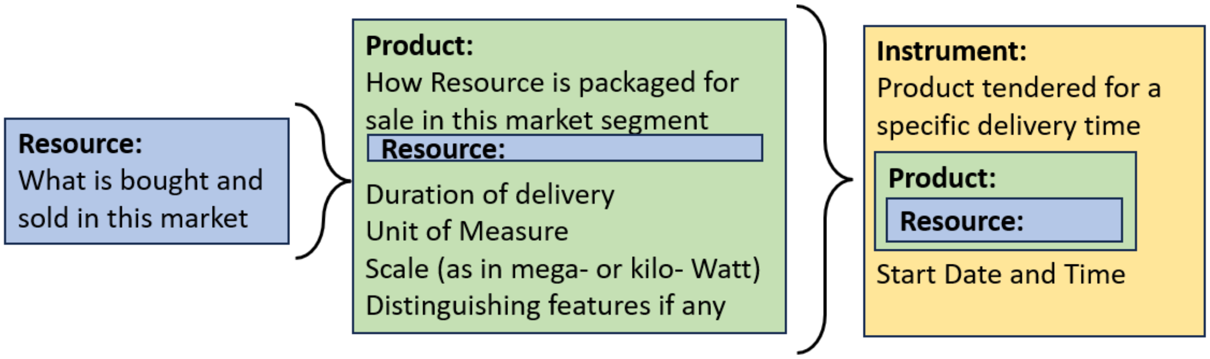

Resource:
What is bought and sold in this market
Product:
How Resource is packaged for sale in this market segment
Resource:
Duration of delivery
Unit of Measure
Scale (as in mega- or kilo- Watt)
Distinguishing features if any
Instrument:
Product tendered for a specific delivery time
Product:
Resource:
Start Date and Time

Figure 3-1 showing the relationship between Resource, Product, and Instrument

The Product incorporates the Resource, defining how the Resource is “packaged” for market. Adding a start date-time to a Product defines an Instrument.

A Market Segment trades Instruments, as a financial market trades financial instruments. CTS trades Instruments to deliver Product at a specific time.

The UML in Figure 3-2 shows the relationship between Resource, Product, and Instrument.

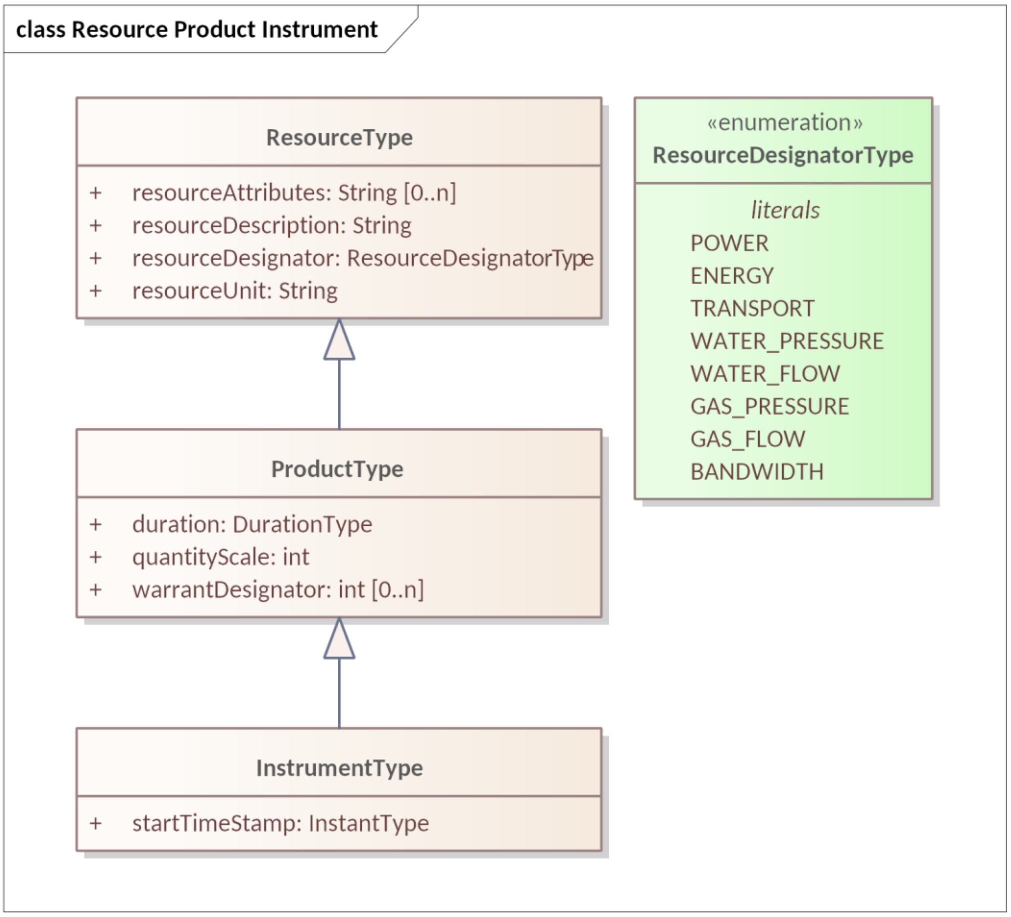


Figure 3-2 UML Class Diagram for Resource, Product, and Instrument

3.1.1 Defining Resource

We define a Resource as a commodity whose value depends on time of delivery. A developer may extend the Resource enumeration using standard UML techniques (subclassing); however, CTS 1.0 uses only the limited list in the Resource Designator Type (Figure 3-2).

A Market typically includes some information that further specifies the Resource, for example voltage and frequency for Power.

Table 3-1: Defining the Resource

Attribute	**Type**	**FIX Field Name**	**Meaning**	**Notes**
Resource Designator	String	*Not in FIX*	POWER ENERGY TRANSPORT WATER_PRESSURE WATER_FLOW GAS_PRESSURE GAS_FLOW BANDWIDTH	The Resource Designator serves a purpose similar to that of the FIX AssetSubClass (1939) The list is extensible

Attribute	Type	FIX Field Name	Meaning	Notes
Resource Unit	String	*Not in FIX*	The unit of measure for the Resource	Item Unit in [EMIX]
Resource Attributes	String	*Not in FIX*	Optional elements that further describe the Resource	e.g. Hertz and Voltage

The Resource is named in the Market. Each Market deals in a single Resource. Segments of a Market restrict trading into profiles of the Resource. Position and Delivery (see Sections 7, 8 below) itemize Resource quantities.

3.1.2 Defining Product

The Product is a Resource packaged for Market. The size and duration of the Product define what is, in effect, the "package size" for the commodity. A Market may offer multiple Products for the same Resource in different Market Segments.

Note that the Product is derived from the[EMIX] ItemBase.

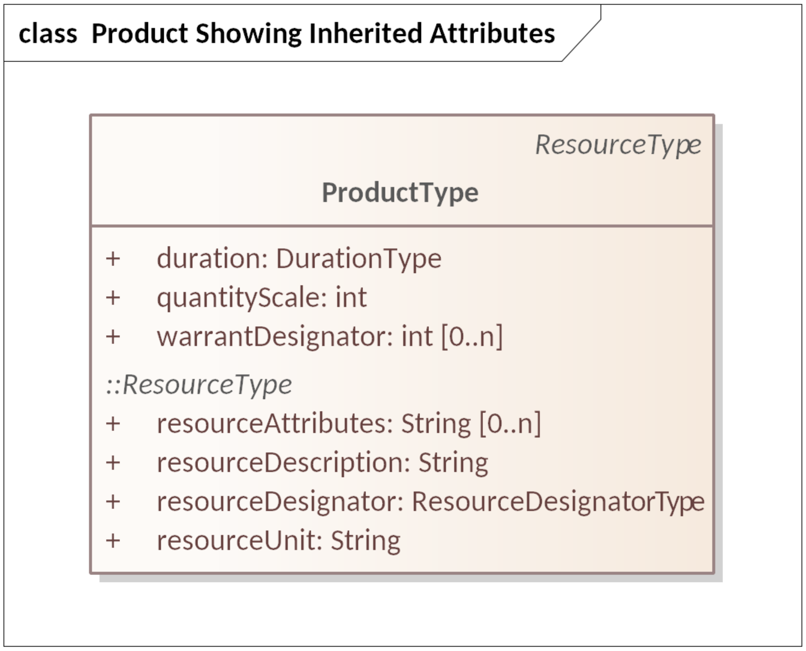

class Product Showing Inherited Attributes
ResourceType
ProductType
+ duration: DurationType
+ quantityScale: int
+ warrantDesignator: int [0..n]
::ResourceType
+ resourceAttributes: String [0..n]
+ resourceDescription: String
+ resourceDesignator: ResourceDesignatorType
+ resourceUnit: String

Figure 3-3 UML Class Diagram for Product showing Inheritance from Resource

Table 3-2, below, defines each of the fields in the Product.

Table 3-2: Defining the Product

Attribute	Type	FIX Field Name	Meaning	Notes
Duration	String	*Not in FIX*	The interval Duration for the specific Product definition.	As defined in **[WS-Calendar]**

Attribute	Type	FIX Field Name	Meaning	Notes
Quantity Scale	Int	*Not in FIX*	The exponent of the Quantity	For example, a Product denominated in kilowatts has a QuantityScale of 3.
Resource Designator	String	*Not in FIX*	Reference to the Resource as defined in this Market.	Used to support Tender validation and auditing
Resource Unit	String	*Not in FIX*	The unit of measure for the Resource	Item Unit in [EMIX]
Warrant Designator (Optional)	Int	*Not in FIX*	Optional further specificity of Product.	Warrants are itemized in the Market. This specification does not define Warrants.

Products with differing Warrants are different Products and therefore traded in different Market Segments.

As non-normative examples, if an Actor wishes to buy energy with a *Green Warrant* (however defined) then the Actor, not the Market, is responsible for defining its trading strategies if the warranted Product is not available. Similarly, an Actor that wishes to buy or sell Neighborhood Solar Power is responsible for submitting Tenders that expire in time to make alternate arrangements, or in time to cancel Tenders before fulfillment. This specification establishes no expectation that the Market engine will address these issues automatically.

Warrants are defined in [EMIX], and CTS permits Warrants to support this complexity if desired, but not described in this specification. A Market MAY define a list of Warrants and Warrant Designators. Warrants were defined in [EI].as additional non-essential characteristics of a Resource such as how it was produced, or an attribute of regulatory interest. Warrants are defined in the Market but are offered per Market Segment.

3.1.3 Defining Instrument

A Market Segment trades Instruments for a single Product. In CTS, an Instrument is a Product delivered for a specific duration beginning at a certain time. CTS includes Duration explicitly in both the Tender and the Quote. The Instrument follows the pattern defined in WS-Calendar—a Resource bound to a Duration ("Product") and the Product bound to a Starting DateTime.

Table 3-3: Specifying the Instrument

Attribute	Type	FIX Field Name	Meaning	Notes
The fields below as defined in the Product, Table 3-2				
Duration	String	*Not in FIX*	The interval Duration for the specific Product definition.	As defined in **[WS-Calendar]**
Quantity Scale	Int	*Not in FIX*	The exponent of the Quantity	For example, a Product denominated in kilowatts has a QuantityScale of 3.
Resource Designator	String	*Not in FIX*	Reference to the Resource as defined in this Market.	Used to support Tender validation and auditing

Attribute	Type	FIX Field Name	Meaning	Notes
Resource Unit	String	*Not in FIX*	The unit of measure for the Resource	Item Unit in [EMIX]
Warrant Designator (Optional)	Int	*Not in FIX*	Optional further specificity of Product.	This specification does not define Warrants.
A start time completes the Product (above) into a tradable Instrument				
Start Time	DateTime (UTC)	*Not in FIX*	Starting Date & Time	A start time completes the specification of Product into a tradable Instrument

Every Tender, Transaction, Quote is to buy or sell a quantity of an Instrument.

Within a market Segment, the Start Date and Time uniquely identifies an Instrument. Because an off-market Segment, sometimes known as over-the-counter (OTC) Segment can transact products of any Duration, Tenders, Quotes, and Transactions all use the Segment identifier, the Start Time, and the Duration to identify the Product.

3.1.4 Summary of Instrument Specification

A UML model for the Instrument showing all inheritance is in Figure 3-4 below:

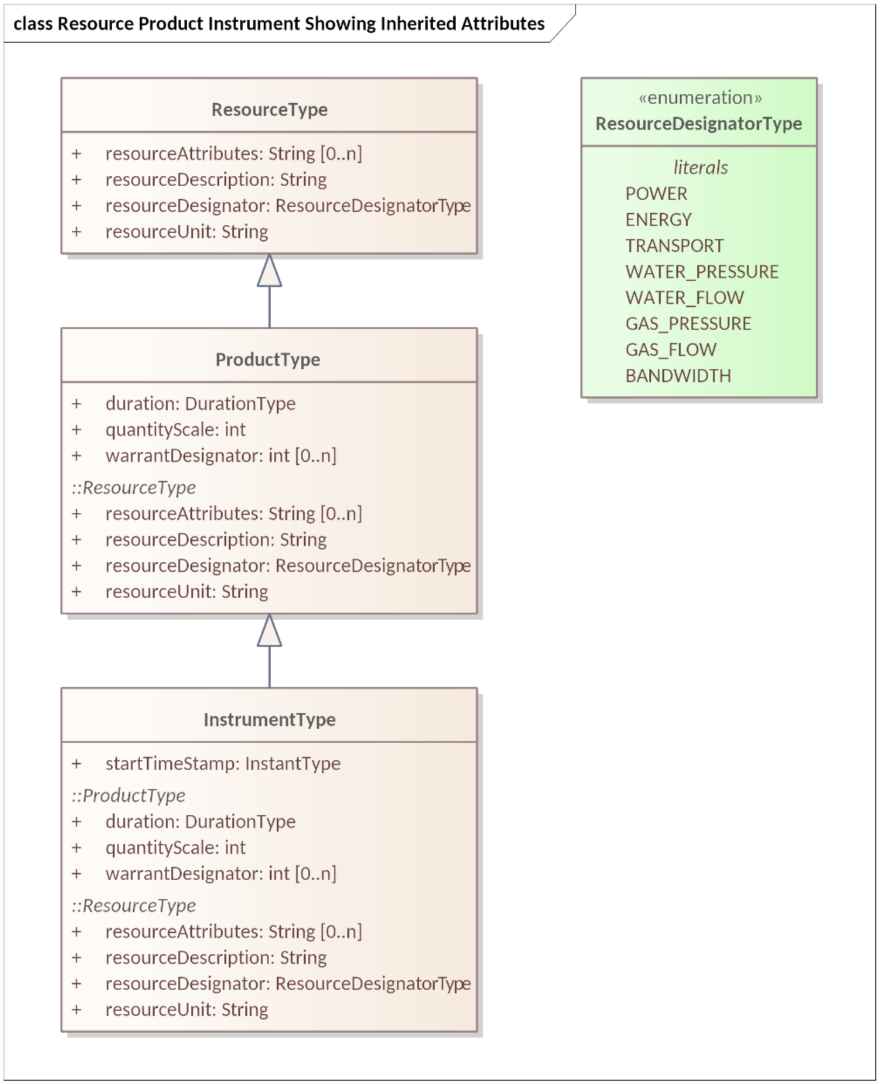

class Resource Product Instrument Showing Inherited Attributes
ResourceType
+ resourceAttributes: String [0..n]
+ resourceDescription: String
+ resourceDesignator: ResourceDesignatorType
+ resourceUnit: String
ProductType
+ duration: DurationType
+ quantityScale: int
+ warrantDesignator: int [0..n]
::ResourceType
+ resourceAttributes: String [0..n]
+ resourceDescription: String
+ resourceDesignator: ResourceDesignatorType
+ resourceUnit: String
InstrumentType
+ startTimeStamp: InstantType
::ProductType
+ duration: DurationType
+ quantityScale: int
+ warrantDesignator: int [0..n]
::ResourceType
+ resourceAttributes: String [0..n]
+ resourceDescription: String
+ resourceDesignator: ResourceDesignatorType
+ resourceUnit: String
«enumeration»
ResourceDesignatorType
literals
POWER
ENERGY
TRANSPORT
WATER_PRESSURE
WATER_FLOW
GAS_PRESSURE
GAS_FLOW
BANDWIDTH

Figure 3-4 UML Class Diagram for Instrument showing Inheritance from Resource & Product

3.2 CTS Streams: Expressing Time Series

Resource Markets are based on time-of-delivery. It is often useful to convey requests and information about consecutive durations. This specification uses the simplified pattern described in WS-Calendar [Streams], that is, common information followed by a repeating set of information for each consecutive Interval. Each Interval uses a common Duration. All Intervals in a Stream are consecutive.

Streams are a response to a request for a Stream. A Request for Quotes may specify Intervals for an entire day. A Request for a Position or for Delivery may also include several Intervals. In either case, the Request includes the common information (as in what is requested, a Duration, and a bounding interval, expressed as a Start DateTime, and an End DateTime).

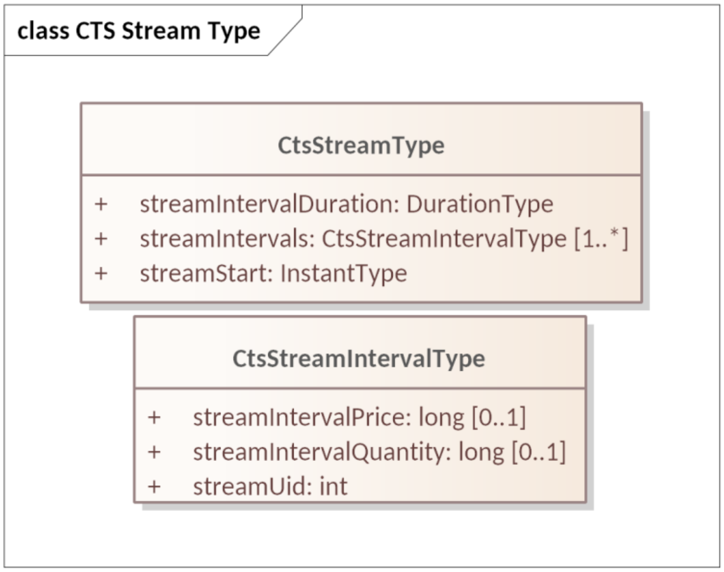

class CTS Stream Type
CtsStreamType
+ streamIntervalDuration: DurationType
+ streamIntervals: CtsStreamIntervalType [1..*]
+ streamStart: InstantType
CtsStreamIntervalType
+ streamIntervalPrice: long [0..1]
+ streamIntervalQuantity: long [0..1]
+ streamUid: int

Figure 3-5~~: Stream~~: UML Class Model for CtsStream and the Stream Intervals

The response to a request for a stream is a stream.

The common information in CTS Streams is the Product and the Start DateTime. The Product specifies Resource and Duration. The consecutive intervals in the CtsStream begin with the Start DateTime for the specified Duration. The second Interval ~~Attributes~~has an implied start of the end of the first Interval. The third Interval has an implied start of the end of the second Interval…and so on.

Each interval carries what can be considered a *local UID.*[13]

Several Facets request a CtsStream in the response. They are:

- Position Facet
- Delivery Facet

Certain payloads may include a CtsStream, including:

- Tender Facet (see *"Interval Tenders and Stream Tenders"*, Section 5.3.1)
- Quote and Negotiation Facet (*see Stream Quote)*

Figure 3-6 shows payloads for the Position and Delivery Facets as an example of the pattern for requesting and responding with streams.

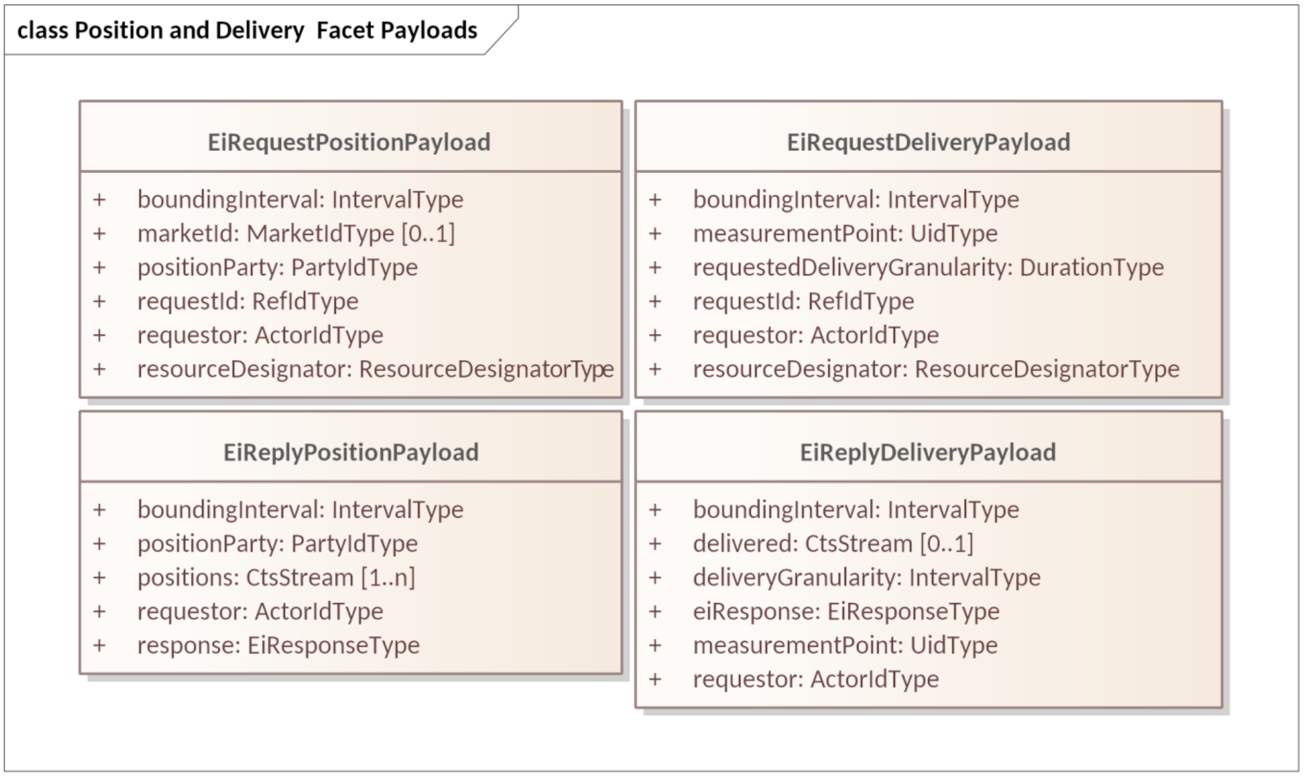
Table 3-4: Specifying the Stream

Attribute	**Type**	**FIX Field Name**	**Meaning**	**Notes**
Stream Interval Duration	String	*Not in FIX*	The interval Duration for each Stream element.	As defined in **[WS-Calendar]** Optional if inherited from message containing Stream
Stream Start	DateTime	*Not in FIX*	Starting Date & Time for the first element in the series of Intervals.	After the first Interval, each Interval starts when the proceeding Interval finishes

[13] Certain serializations for payloads do not guarantee order, so a small integer serves as a unique identifier for each interval.

Attribute	**Type**	**FIX Field Name**	**Meaning**	**Notes**
Stream Interval Price Value	Long	Price (44)	Price per Unit during Interval	Optional depending upon purpose of message including Stream
Stream Interval Quantity Value	Long	OrderQty (38)	The Quantity of the Product during the Interval	Optional depending upon purpose of message including Stream
StreamUID	Int	*Not in FIX*	Unique identifier for each interval.	Certain serializations for do not guarantee order- the UID enables it to be reconstructed

The CTS pattern includes a Bounding Interval, and the response requests is all CtsStream Intervals that are contained within the Bounding Interval including those which align with the ends of the Bounding Interval.



class Position and Delivery Facet Payloads
EiRequestPositionPayload
+ boundingInterval: IntervalType
+ marketId: MarketIdType [0..1]
+ positionParty: PartyIdType
+ requestId: RefIdType
+ requestor: ActorIdType
+ resourceDesignator: ResourceDesignatorType
EiRequestDeliveryPayload
+ boundingInterval: IntervalType
+ measurementPoint: UidType
+ requestedDeliveryGranularity: DurationType
+ requestId: RefIdType
+ requestor: ActorIdType
+ resourceDesignator: ResourceDesignatorType
EiReplyPositionPayload
+ boundingInterval: IntervalType
+ positionParty: PartyIdType
+ positions: CtsStream [1..n]
+ requestor: ActorIdType
+ response: EiResponseType
EiReplyDeliveryPayload
+ boundingInterval: IntervalType
+ delivered: CtsStream [0..1]
+ deliveryGranularity: IntervalType
+ eiResponse: EiResponseType
+ measurementPoint: UidType
+ requestor: ActorIdType

Figure 3-6~~Attribute~~	~~Meaning~~	~~Notes~~
~~Stream Price Value~~	~~The Price value for this specific Stream Interval, subject to indicated Scale/Granularity~~	~~At least one of (Stream Price Value, Stream Quantity Value) MUST be present.~~
~~Stream Quantity Value~~	~~The Quantity value for this specific Stream Interval, subject to indicated Scale/Granularity~~	~~At least one of (Stream Price Value, Stream Quantity Value) MUST be present.~~
~~UID~~	~~A "Local UID" used to order the Interval within the Stream~~	~~As conformant CTS implementations need not be owned by the same implementer, intermarket gateways (however defined) may deserialize and re-serialize to different specifications~~

: UML Examples of Payloads incorporating Streams

The information within each Interval varies per message type. For example, a StreamQuote or a StreamTender will put the Price and Quantity in each interval. A Delivery (metering) payload will put only the Quantity in each Interval.

~~6~~4 Party Registration Facet

~~Background (adapted from **[EI]**)~~

A valid Party ID is required to interact with a Market and is included in most payloads.

Party Registration is described in EI. This facet describes the messages necessary for an actor to register and obtain a Party ID ~~in order~~ to participate in a Market.

EiCreateParty associates an actor with a Party ID and informs the ~~Marketplace~~Market of that ID. CTS makes no representation on whether that ID is an immutable characteristic, such as a MAC address, a stable network address, such as an IP, or assigned during registration,

EiRegisterParty names the exchange of information about an actor that enables full participation in a CTS ~~Marketplace~~Market. It may exchange information needed for financial transfers including, perhaps, reference to an existing customer or vendor ID, or proof of financial bond for large participants, or issuance of crypto-tokens, or any other local market requirements. A Registered Party is ready to be a full participant in the local Market.

Cancel Party Registration removes a Party from the Market. It may include final settlement, cancellation of outstanding Tenders, backing out of future contracts, or other activities as defined in a particular CTS ~~Marketplace~~Market.

Aside from the business services as described, Party Registration may have additional low-level requirements tied to the protocol itself used in a particular implementation based on CTS.

This specification does not attempt to standardize these interactions and messages beyond naming the Register Party facet. A more complete discussion can be found in the [EI] specification.

Some ~~Marketplaces~~Markets MAY wish to associate one or more measurement points with a Party. Such measurement points could be used to audit Transaction completion, to assess charges for using uncontracted-for-energy, etc. Measurement points are referenced in *Section 8 “The Delivery Facet”*, Markets that require this functionality may want to include an enumeration of Measurement Points in Party Registration.

~~7 Marketplace Facet~~

~~The Marketplace facet is an extension of [EI]. In CTS the Marketplace includes all the Markets wherein a Party can trade, and are associated with all the Products a Party can trade for.~~

~~For example, where all trading is in a single microgrid, the Marketplace is implicitly tied to that microgrid. Where trading is across a city or across a traditional utility or across a region the Marketplace hosts all Market interactions for that utility or region. Nothing in this specification prohibits multiple Marketplaces, such as a Wholesale or a Retail, or a sourced Marketplace such as Solar.~~

~~Using the Resource / Product / Instrument terminology, each Product has its own market, and these different markets may have different rules, or different matching engines. All are in the same Marketplace.~~

~~The Marketplace Facet defines characteristics common to all the local Markets, and catalogs how to participate in each Market.~~

~~7.1 The [EI] Market Context and the Marketplace~~

~~Market Contexts in [EMIX] and [EI] are URIs and are used to request information about the Market or Marketplace that rarely changes, so it is not necessary to communicate it with each message.~~

~~Note that a Market Context is associated with and identifies a collection of values and behaviors; while an [EI]~~ An implementation ~~MAY use operations such as POST to a Market Context URI, that behavior~~ is not required~~.~~

~~For~~ to use the Party Registration Facet. For example, if uniqueness and universality are satisfied, any ~~Marketplace, there are standing terms and expectations about Product offerings. If these standing terms and expectations are not known, many exchanges may need to occur before finding Products and Tenders that meet those expectations. If all information about the Marketplace were to be transmitted in every information exchange, messages would be overly repetitive.~~assignment of Party IDs should work.

~~7.2 Registering in a Marketplace~~

~~The scope of a Party ID is a Marketplace. The Party ID MUST be unique within a Marketplace.~~

~~Only the acquiring of Party ID is in scope in the following list:~~

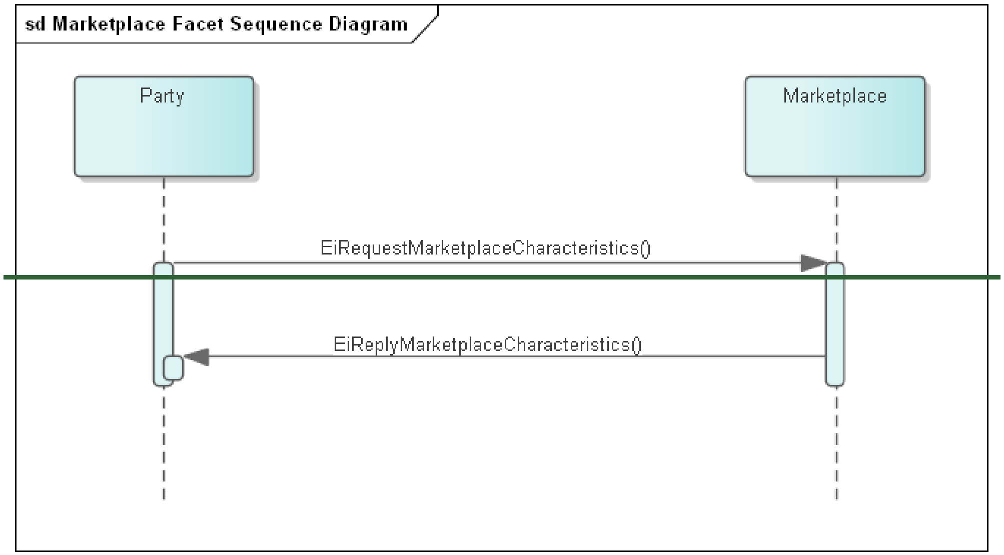
- ~~Obtain Party ID~~
- ~~Establish billing terms, if any~~
- ~~Exchange Location, if required~~

~~See for more information.~~

~~7.3 Interaction Pattern for the Marketplace Facet~~

~~An Actor MUST interact with a specific Market to trade a specific Product. A Market matches Tenders for all Instruments based on a given Product. The matching engine is contained within the Market and different matching engines have no visibility past the Market Facet.~~

~~The Marketplace Facet enables a Party to request the details of a Marketplace and the Markets contained in the Marketplace. Using the Marketplace Facet for discovery, Parties MAY request and compare Market Characteristics to select which markets to participate in.~~


sd Marketplace Facet Sequence Diagram
Party
Marketplace
EiRequestMarketplaceCharacteristics()
EiReplyMarketplaceCharacteristics()

~~Figure -: UML Sequence Diagram for the Markeplacet Facet~~

~~Once Markets are identified as a candidate, the Market Facet can retrieve the standard terms associated with those Markets. See .~~

5 The Tender Facet (Trade Messages)

A party wishing to buy or sell submits

~~7.4 Information Model for the Marketplace Facet~~

~~Table - Information Model for the Marketplace Facet~~

~~Attribute~~	~~Attribute Name~~	~~Attribute Type~~	~~Meaning~~
~~Marketplace Name~~	~~NAME~~	~~String~~	~~Text providing a descriptive name for a Marketplace. While the Name MAY be displayed in a user interface; it is not meaningful to the Actors.~~
~~Currency~~	~~CURRENCY~~	~~String~~	~~String indicating how value is denominated in a market. If fiat currency, should be selected from current codes maintained by UN CEFACT. May also be cryptocurrencies or local currency.~~
~~Time Offset~~	~~T_OFFSET~~	~~Long~~	~~A Duration that some Marketplaces MAY use to describe trading where a first interval is not on an hourly boundary.~~[14]
~~Time Zone~~	~~TZ~~	~~String~~	~~A Time Zone indicates how all Times and Dates are expressed.~~
~~Price Decimal Fraction Digits~~	~~PRICE_FRAC~~	~~Long~~	~~Some market implementations use a Marketplace-wide indication of how many decimal fraction digits are used.~~
~~Resource Descriptor~~	~~RESOURCE~~	~~String~~	~~The Resource traded in this Market.~~
~~Markets~~	~~MARKETS~~	~~Market Description~~	~~A list of Market Descriptions for each Market contained in the Marketplace.~~

~~Market Descriptions in a Marketplace~~

~~A marketplace itemizes each of the Markets in the marketplace. This is indicated by a set of Market Descriptors with the following attributes, one for each contained Market:~~

[14] A power distribution entity may experience disruption if there is a big price change on the hour. For example, a distribution system operator (DSO) that operates multiple CTS Marketplaces could opt to set a different offset on each Marketplace operated out of a given substation. In this model, a Market could use an offset duration of 3 minutes to indicate that all tenders are based on three minutes after the hour.

~~Table - Market Description~~

~~Attribute~~	~~Attribute Name~~	~~Attribute Type~~	~~Meaning~~
~~Market Name~~	~~MARKET~~	~~String~~	~~Optional text providing a descriptive name for a Market. While the Name MAY be displayed in a user interface; it is not meaningful to the Actors.~~
~~Resource~~	~~Resource~~	~~Resource Designator~~	~~[Extensible] enumeration indicating what is sold in each Market~~
~~URI~~	~~MARKET-URI~~	~~String~~	~~URI to access the Market.~~

~~Descriptions of the Product found in each Market are found in each Market and are not replicated into the Marketplace.~~

~~8 Market Facet~~

~~All interactions in a Market are subject to Standard Terms which are discovered through the Market Facet.~~

1. ~~A Party interacts with the Marketplace to discover all Markets in which the Resources the Party is interested in are traded.~~
2. ~~A Party queries each of the Markets trading that Resource discover the Products in each Market, and the Standard Terms for each.~~
3. ~~Resources with Warrants are in their own Markets, which may have their own Standard Terms. The Warrant is a Market Term.~~
4. ~~The Party uses the Party ID determined during Marketplace Registration for all Tenders.~~
5. ~~The Party determines which Products it wants by submitting Tenders to the Market is chooses.~~
6. ~~Each Tender is for a specific Instrument, which is the Market Product plus a Starting Time.~~

~~A Market matches Tenders to create Transactions using the Tender and Transaction Facets.~~

~~While there is no standard matching algorithm defined in CTS, the Standard Terms include indicators of how the Market matches Tenders. For example, different bidding strategies may be used when submitting to a double auction market than for~~ an order ~~book market.~~

~~Interactions with the Market are through~~ (“Tender”) to the Tender ~~(see Section) and Transaction (see Section) Facets.~~

~~8.1 Market Context History~~

~~Market Contexts in [EMIX] and [EI] are URIs and express Standard Terms that rarely changes, so it is not necessary to communicate it with each message.~~

~~In CTS, this is refined to the Marketplace Facet (Section) and the Market Facet (Section).~~

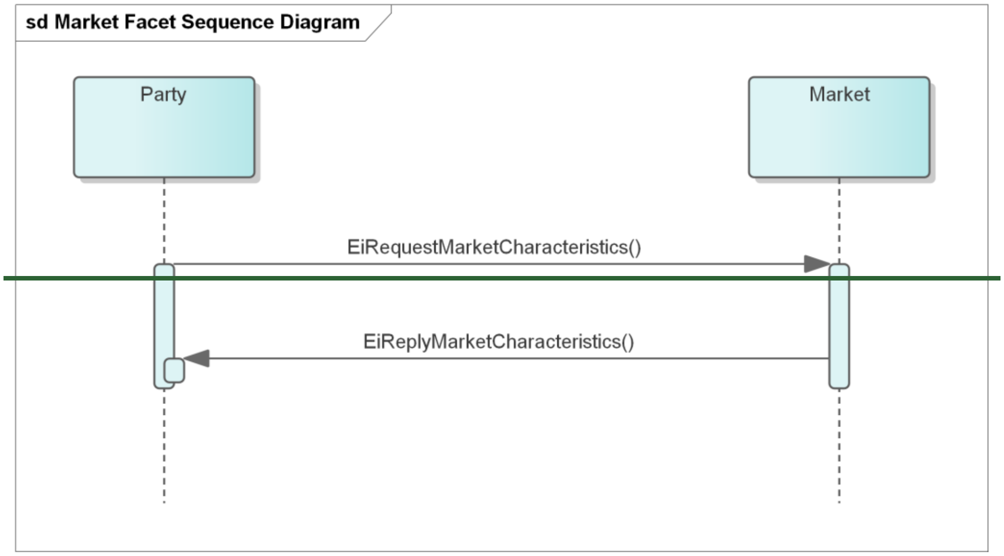
~~8.2 Payloads for the Market Facet~~

~~Facet~~	~~Request Payload~~	~~Response Payload~~	~~Notes~~
~~Market~~	~~EiRequestMarketCharacteristcs~~	~~EiReplyMarketCharacteristcs~~	~~Request specific Market Characteristics~~

~~8.3 Interaction Pattern for the Market Facet~~

~~An Actor interacts with a specific Market to trade a specific Product. A Market matches Tenders for all Instruments based on a given Product. The matching engine is contained within the Market and different matching engines have no visibility past the Market Facet.~~

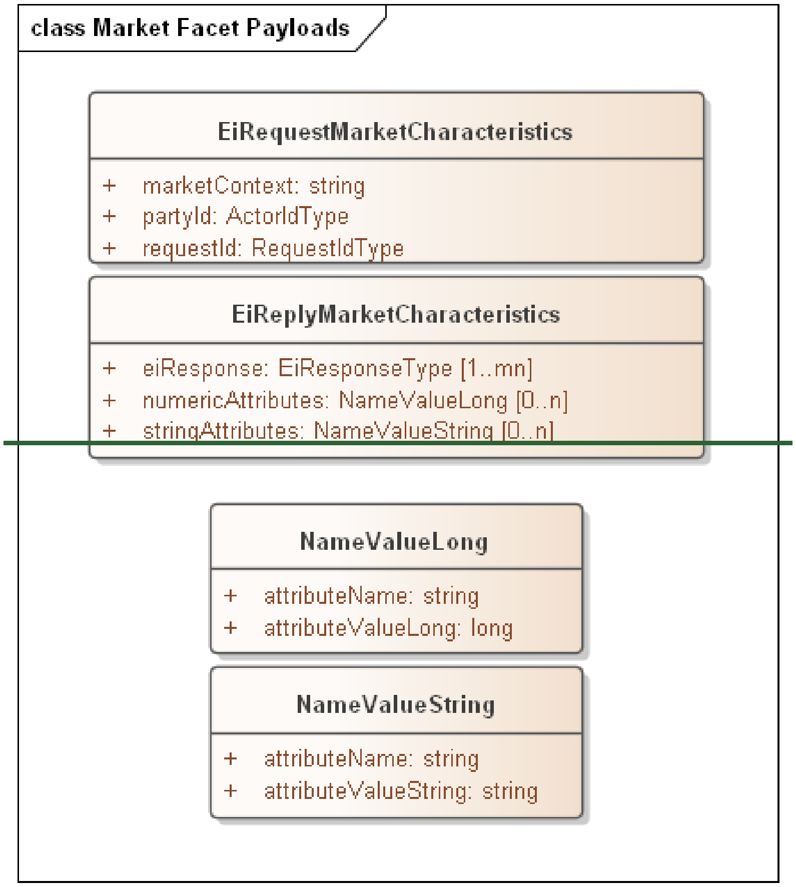
~~The Market Facet enables a Party to request the details of a Marketplace. Using the Market Facet, Parties MAY be able to request and compare Market Contexts to select which markets to participate in.~~


sd Market Facet Sequence Diagram
Party
Market
EiRequestMarketCharacteristics()
EiReplyMarketCharacteristics()

~~Figure -: UML Sequence Diagram for the Market Facet. TODO: Update for Market Facet~~

~~The Market Facet can be used to retrieve the standard terms associated with a Market.~~

~~An EiRequestMarketCharacteristics payload requests the standard terms for a Market; the reply payload EiReplyMarketCharacteristics returns those terms as name-value pairs.~~


class Market Facet Payloads
EiRequestMarketCharacteristics
+ marketContext: string
+ partyId: ActorIdType
+ requestId: RequestIdType
EiReplyMarketCharacteristics
+ eiResponse: EiResponseType [1..mn]
+ numericAttributes: NameValueLong [0..n]
+ stringAttributes: NameValueString [0..n]
NameValueLong
+ attributeName: string
+ attributeValueLong: long
NameValueString
+ attributeName: string
+ attributeValueString: string

~~Figure -: UML of Market Facet payloads~~Facet.

~~8.4 Information Model for the Market Facet~~

~~Sending an EiRequestMarketCharacteristics payload referencing a Market requests standard terms as given in .~~

~~These are derived and extended from EMIX Terms; those are extrinsic to the Product delivery but effect how each Party interacts with others. Terms may be tied to basic operational needs, or schedules of availability, or limits on bids and prices acceptability.~~

~~The CTS Standard Terms MAY be extended to reflect additional capabilities and description.~~

~~Strings returned for attribute values MUST be no longer than 256 bytes.~~

~~TODO Consider requiring resolvable URIs for contexts, or pairing with potential transport endpoints.~~

~~Table -: Standard Terms returned by Market Facet~~

~~Attribute~~	~~Attribute Name~~	~~Attribute Type~~	~~Meaning~~
~~Market Name~~	~~NAME~~	~~String~~	~~Text providing a descriptive name for a Market. While the Name MAY be displayed in a user interface; it is not meaningful to the Actors.~~
~~Currency~~	~~CURRENCY~~	~~String~~	~~String indicating how value is denominated in a market. If fiat currency, should be selected from current codes maintained by UN CEFACT. May also be cryptocurrencies or local currency.~~
~~Time Offset~~	~~T_OFFSET~~	~~Long~~	~~A Duration that some Marketplaces MAY use to describe trading where a first interval is not on an hourly boundary.~~[15]
~~Time Zone~~	~~TZ~~	~~String~~	~~A Time Zone indicates how all Times and Dates are expressed.~~
~~Price Decimal Fraction Digits~~	~~PRICE_FRAC~~	~~Long~~	~~Some market implementations use a Marketplace-wide indication of how many decimal fraction digits are used.~~[16]
~~Market Party ID~~	~~MPARTYID~~	~~String~~	~~The PartyID to use in a Tender (reference 2.2.3)~~

[15] A power distribution entity may experience disruption if there is a big price change on the hour. For example, a distribution system operator (DSO) that operates multiple CTS Marketplaces could opt to set a different offset on each Marketplace operated out of a given substation. In this model, a Market could use an offset duration of 3 minutes to indicate that all tenders are based on three minutes after the hour.

[16] Integer operations are typically much more efficient than fixed or floating point, so it is likely to be much faster to apply decimal shift on input and output rather than for more frequent comparison operations in the matching engine implementation

~~Attribute~~	~~Attribute Name~~	~~Attribute Type~~	~~Meaning~~
~~Bilateral OK~~	~~BILATERAL OK~~	~~Long~~	~~Boolean expressed as an integer:~~ ~~0 – False – bilateral Tenders or Transactions not permitted, only Market Tenders~~ ~~1 – True – bilateral Tenders or Transactions with identified parties are permitted.~~
~~Resource Designator~~	~~R_ID~~	~~Resource Designator~~	~~[Extensible] enumeration indicating the Resource traded in this Market. This establishes the Resource Designator used in Product definitions and in messages~~
~~Containing Marketplace~~	~~MPLACE~~	~~String~~	~~URI for Marketplace Context~~
~~Product~~	~~PRODUCT~~	~~Array of Ordered Pairs~~	~~See Product Definition, Table 8-2: Elements that define Products in a Market. It SHALL match the Product Definition indicated in the Marketplace for this Market.~~
~~Tender Grouping~~	~~TGROUP~~	~~Long~~	~~Enumeration expressed as an integer for treatment of multiple tenders either in a single EiCreateTender payload or across all Tenders for the same Instrument:~~ ~~0 – Tenders are independent (*JBOT*)~~ ~~1 – All Tenders within a single EiCreateTenderPayload SHALL BE treated by the market as points on a supply or demand curve as indicated by the Side of the Tenders (ALLINPAYLOAD)~~
~~Clearing Approach~~	~~CLEAR~~	~~Long~~	~~Enumeration expressed as an integer to describe market clearing approach:~~ ~~0 – CONTINUOUS – continuous clearing~~ ~~1 – PERIODIC – not continuous, typically with periodicity related to Product T_GRAIN.~~
~~Clearing Duration~~	~~CLEAR-DURATION~~	~~String~~	~~Duration before Instrument start time used for market matches. Only valid if Clearing Approach is 1 – PERIODIC Not valid if Periodic Time is specified. *"All matches on the hourly market are made 30 minutes before the hour"*~~

~~Attribute~~	~~Attribute Name~~	~~Attribute Type~~	~~Meaning~~
~~Clearing Time~~	~~CLEAR-TIME~~	~~String~~	~~Time for market matches. Only valid if Clearing Approach is 1 – PERIODIC Not valid if Periodic Duration is specified.~~ ~~*"All matches on the Day-Ahead hourly market are made at 3:00 PM".Table 9-1*~~
~~Clearing Days Ahead~~	~~CLEAR-DA~~	~~Long~~	~~Number of days prior to the Instruments for the CLEAR-TIME. For example if a Two-day ahead market clears at 3pm, CLEAR-DA = 2 and CLEAR-TIME = "3:00PM"~~
~~All transactions for an Instrument at the same clearing price~~	~~ALL_AT_CLEAR~~	~~Long~~	~~Boolean expressed as integer~~ ~~0 – False – Tenders for a specific Instrument MAY clear at different prices.~~[17] ~~1 – True – As in Double Auction, all participants clear at the same price.~~
~~Maximum~~	~~MAX~~	~~Integer~~	~~Maximum Transaction size the Market will accept.~~
~~Quote Reference~~	~~QUOTE-REF~~	~~String~~	~~A string indicating the Quote Reference for this Market to which an actor may subscribe or unsubscribe.~~
~~Ticker Reference~~	~~TICKER-REF~~	~~String~~	~~A string indicating the Ticker Reference for this Market to which an actor may subscribe or unsubscribe.~~

~~Each Product in a Marketplace is defined using attributes as below~~

~~Table -: Elements that define Products in a Market~~

~~Attribute~~	~~Attribute Name~~	~~Meaning~~
~~Resource Designator~~	~~R_ID~~	~~[Extensible] enumeration indicating the required Resource~~
~~Time Granularity~~	~~T_GRAIN~~	~~The interval duration in seconds for the specific Product definition~~

[17] A sophisticated Party may change its Bidding strategy based on this Market Characteristic. For example, in a Double Auction all negative tenders clear at a market price that may be positive, so a negative bid in a market with positive prices ensures that the bidder is "in the money", so negative bids are a realistic strategy.

~~Attribute~~	~~Attribute Name~~	~~Meaning~~
~~Quantity Scale~~	~~Q_SCALE~~	~~The exponent of the Quantity. For example, a Product denominated in kilowatts has a Q_SCALE of 3.~~
~~Quantity Granularity~~	~~Q_GRAIN~~	~~The allowed quantity unit size, e.g. Q_GRAIN == 10 means that a Tender for 9 units will be rejected but any multiple of 10 will be accepted.~~
~~Price Granularity~~	~~PRICE_GRAIN~~	~~The allowed price unit, e.g. Price Granularity == 10 means that that any multiple of 10 CURRENCY units is acceptable, but any price not matching, say a price of 9 CURRENCY units, is rejected. May be negative as in -3, Prices are multiples of .001.~~
~~Market~~	~~MARKET~~	~~The message endpoint to access the market where this Product is traded.~~
~~Warrants~~	~~WARRANT~~	~~Optional further specificity of Product~~

960

~~9 Tender Facet~~

~~The terminology of this section is that of business agreements: Tender and Transaction.~~ The Service descriptions and payloads are simplified and updated from those defined in EI. The FIX Protocol classifies Tenders as one of the Trade messages.

~~9.1 Tenders as a Pre-Transaction Payloads~~

5.1 ~~Pre-transaction interactions~~Messages for the Tender Facet

Trade messages are ~~those~~exchanged between parties ~~that may prepare for a transaction.~~to find or create a Transaction. The ~~pre-transaction facet in CTS is the Tender Facet (and including EiDistributeTender), with~~Tender Facet payloads are shown in Table 1-1.

Tenders and transactions are artifacts based on **[EMIX]** artifacts, suitably flattened and simplified, and which contain schedules and prices in varying degrees of specificity or concreteness.

Table 1-1: ~~Pre-Transaction~~ Tender ~~Facets~~Facet Payloads

Facet	~~Request Payload~~CTS Initial Message	CTS Response ~~Payload~~Message	~~Notes~~Meaning
EiTender	EiCreateTender	EiCreatedTender	~~Send~~Create sends a ~~CTS Stream of~~message containing one or more Tenders. ~~Create~~ Created returns errors, and ~~emit Request Payload~~when successful returns the Market-assigned ID for the submitted Tender
EiTender	EiCancelTender	EiCanceledTender	Cancel one or more Tenders

In the FIX specification, a Tender is “completed” when it is satisfied, when it is cancelled, or when it is replaced. CTS does not permit replacing tenders, instead requiring that a Party cancel a tender and submit a new one. If a Tender is already partially filled, cancellation cancels only the unfilled portion.[18]

5.1.1 Illustrative Narrative on Tenders [Non-Normative]

For example, Party A submits a Tender 1 to buy 100 kWh over an hour. A Tender	~~EiDistributeTender~~	~~None~~	~~Describe a list of Tenders to be notified to a set of parties~~

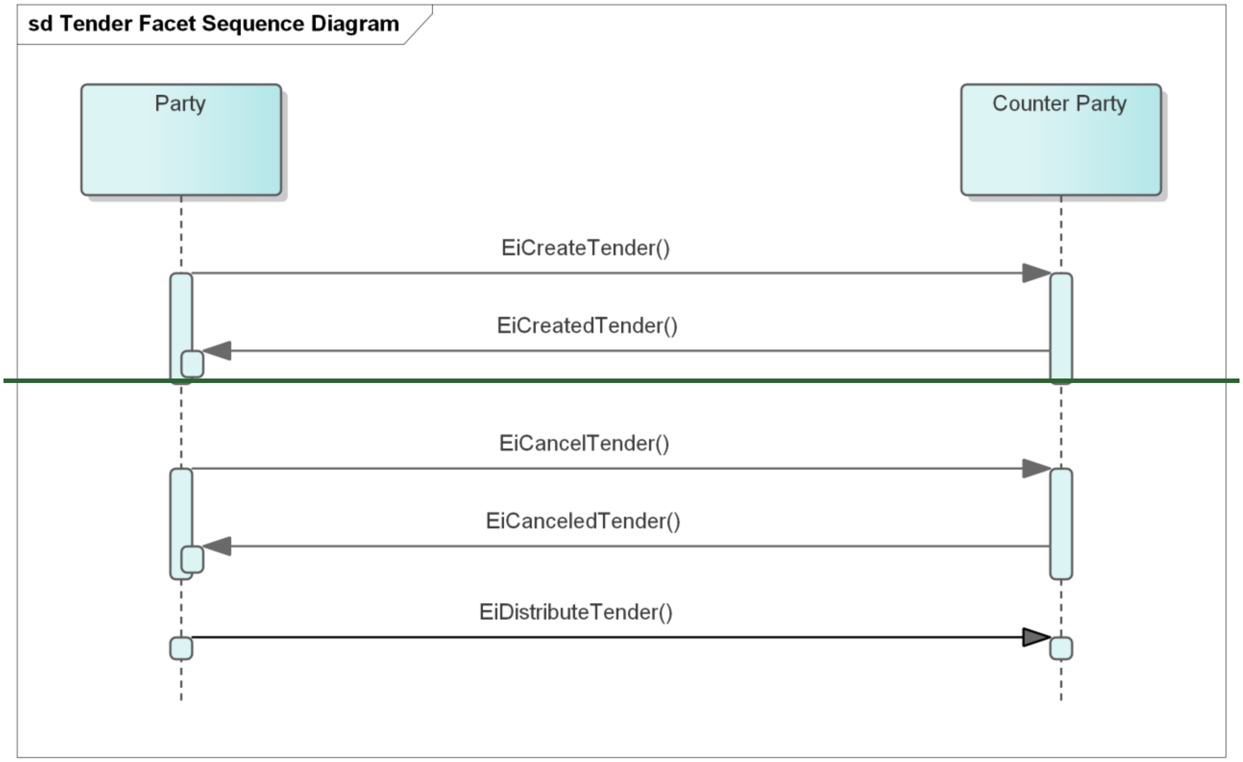
[18] This avoids a potential race condition in variable latency distributed systems.

from Party B for 45 kWh matches Party A’s Tender and the Market creates a Transaction (see Section **6**, *“The Transaction Facet*~~Ei Tender~~			

” for a discussion of Transactions). A Tender from Party C for 35 kWh matches Party A’s Tender and the Market creates a Transaction. Party A’s Tender 1 remains on the market with 20 kWh remaining. If Party A wishes to increase the price offered to get the 20 kWh for a critical operation, Party A must cancel Tender 1, with 20 kWh remaining, and submit a Tender 2 offering a new price. Cancelling Tender 1 does not invalidate either of the two completed Transactions.

~~9.2~~5.2 Interaction Patterns for the Tender Facet

Figure 1-1 presents the ~~[~~UML~~]~~ sequence diagram for the EiTender Facet. Note that ~~EiDistributeTender is not part of~~while [EI] defines a message EIDistributeTender, CTS uses the Negotiation Facet (Section 9, *“The Negotiation Facet”*) and Ticker Subscriptions (Section 11, *“Ticker* ~~1.0 at present, but is being considered for a future release~~”) to accomplish similar purposes.


sd Tender Facet Sequence Diagram
Party
Counter Party
EiCreateTender()
EiCreatedTender()
EiCancelTender()
EiCanceledTender()
EiDistributeTender()

988

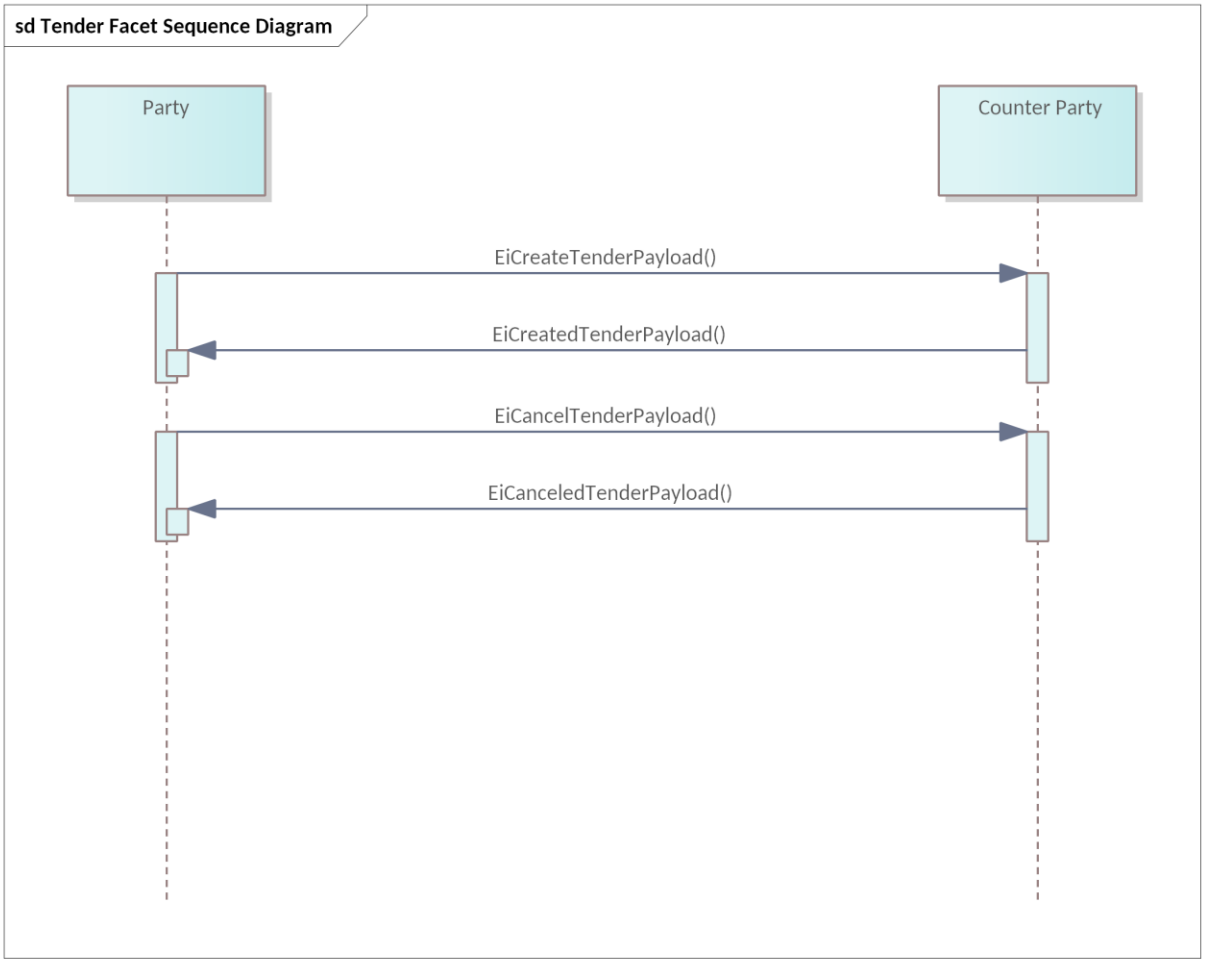
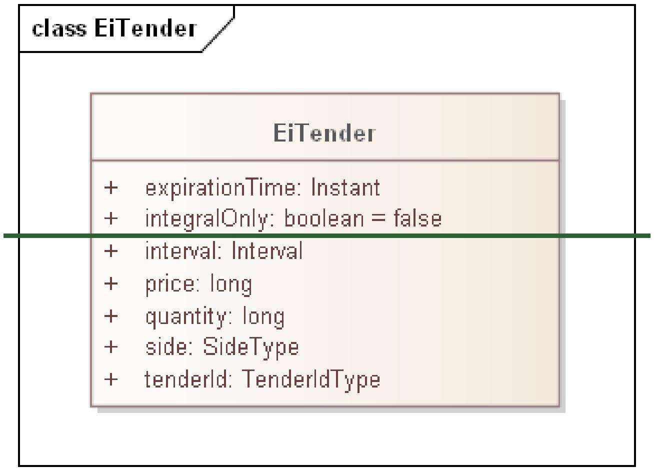


Figure 1-1: UML Sequence Diagram for the Tender Facet

~~9.3~~5.3 Information Model for the Tender Facet

The information model for the ~~EiTender~~Tender Facet artifacts follows that of [EMIX~~]~~,] but flattened and with Product definition implied by the implementation. See Section 5.5 Message Payloads for the Tender Facet below.

The EiTender and EiQuote classes share most attributes in common. Accordingly, a superclass *TenderBase* holds those common attributes as shown in Figure 1-2~~Start time, price, and quantity are key elements for an Instrument offering to buy or sell a Product. The other aspects of Product definition (e.g. Resource, units, and duration) are described in Section .~~



. *TenderBase* is an abstract class, so no object can be of that class.

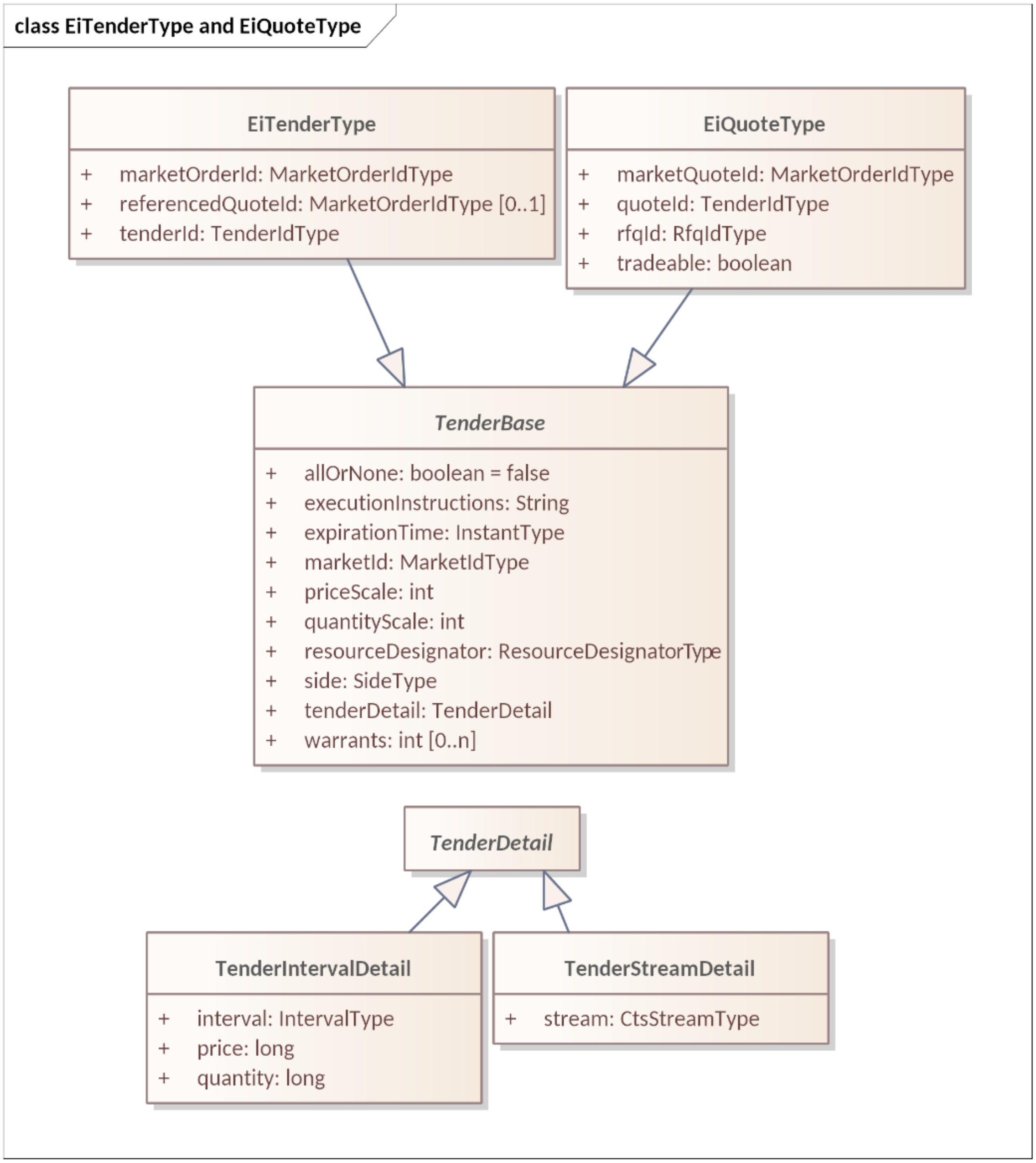


Figure 1-2~~:~~ UML Class ~~EiTender~~Diagram showing common attributes between EiTenderType and EiQuoteType

Attributes used in Tenders, are shown in Table 1-2

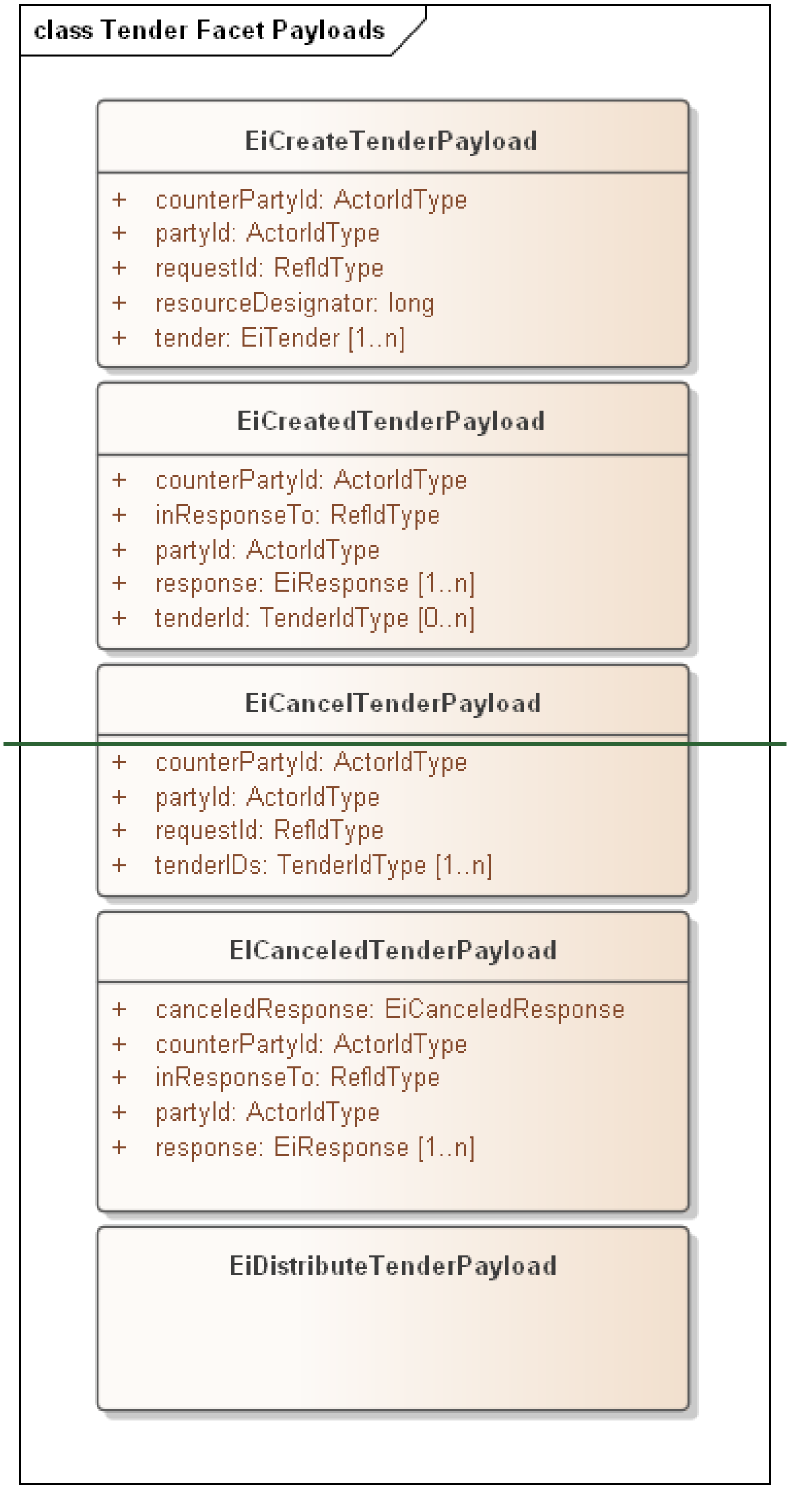
Table 1-2~~: EiTender~~ Tender Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
~~Expiration Time~~			~~The date and time after which this Tender is no longer valid.~~	
~~Integral Only~~All or None	Boolean	In Execution Instructions	All or none of the ~~Tender~~tendered or quoted amount must be ~~bought or sold at once; no partial sale or purchase~~traded.	~~In CTS set to False. Partial sale or purchase is always allowed. The attribute is present for possible future evolution.~~In Energy Interoperation 1.0 this was called *IntegralOnly*
Execution Instructions	String	ExecInst (18)	FIX Supports many instructions for how to execute a tender (or Tradable Quote)	See Table 1-1 below. Modeled as a String in CTS.
Expiration Time	Instant Type	ExpireTime (126)	The Tender or Quote expires at the specific time.	Always expressed in UTC
Interval	Start Time and Duration	*Not in FIX*	~~The time interval for the Product being offered~~Start Date and Time for Product delivery together with Duration of delivery. Part of Instrument	While a Market Segment only accepts Tenders and Quotes of a single configured duration, the complete description is required to ensure validity and for off-market interactions.
Market ID	UID	MarketID (1301)	Identifies the Market	
Market Segment ID	UID	MarketSegmentID (1300)	Identifies the Segment processing the Tender, Transaction, or Quote	This should be a unique combination paired with the Market Order ID
Order ID	UID	ClOrdID (11)	ID assigned by originating Party	
Market Order ID	UID	ORDERID (37)	ID assigned by the Segment or Market.	Used in acknowledgment and in all future market messages
Price	Long	Price (44)	The unit price for the Product being Tendered	~~Total price~~Amount is the product of ~~price~~Price and ~~quantity~~Quantity. Note that ~~price~~Price is subject to the Price Decimal Fraction value. ~~See Scale and Granularity constraints in Section , “”~~
Price Scale	Int	Not Defined in FIX	A multiplier for the Price	A Market Segment may be denominated in, for example dollars or 10ths of a cent.

Attribute	Type	FIX Field Name	Meaning	Notes
~~Expiration Time~~			~~The date and time after which this Tender is no longer valid.~~	
Quantity	Long	OrderQty (38)	The quantity of the Product being Tendered	~~Total price is~~Must meet the ~~product of price~~SCALE and ~~quantity suitably scaled~~Round Lot requirements of the Segment
Quantity Scale	Int		A scale factor on the Resource unit for this Market	For example, “mega” vs “kilo” vs “femto-”
Referenced Quote ID	UID	QuoteMsgID (1166)	Tradable Quote reference in Tender	Reference to Market Quote ID assigned by Market. See Negotiation Facet (Section 9) for use.
Resource Designator	Resource Designator	Not defined in FIX	Identifier of the Resource being offered (Optional in many markets)	While a Market only accepts Tenders and Quotes for a single Resource, the complete description is required to ensure validity and for off-market interactions.
Tender Detail	Tender Detail	Not defined in FIX	Unit price and quantity for this tender	May be Interval or Stream as permitted
Tender ID	UID		ID as submitted to Market	Identifies Tender until Market Order ID is assigned by Market
Tender Interval	Tender Interval Detail	Not defined in FIX	Interval, price and quantity for this tender	Used in Interval Tender
Tender Stream	Tender Stream Detail	Not defined in FIX	Stream of consecutive Intervals with Prices and Quantities	Sometime referred to as a Load Curve in Power Markets.
Side	Side Type	Side (54)	Whether ~~The~~the Tender is to buy or to sell the Product	Buy or Sell
~~Tender ID~~Warrant Ref	~~An ID for this Tender~~Int	*Not in Fix*	Reference to Warrants as defined in the Market	If used, see comments Warrants in Tenders, Section 5.3.3.

~~9.4 Payloads for the Tender Facet~~

The ~~[UML] class~~following diagram ~~describes the payloads for the Tender Facet operations~~shows EiTenderType showing inherited and included attributes.


class Tender Facet Payloads
EiCreateTenderPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ resourceDesignator: long
+ tender: EiTender [1..n]
EiCreatedTenderPayload
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ partyId: ActorIdType
+ response: EiResponse [1..n]
+ tenderId: TenderIdType [0..n]
EiCancelTenderPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ tenderIDs: TenderIdType [1..n]
EICanceledTenderPayload
+ canceledResponse: EiCanceledResponse
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ partyId: ActorIdType
+ response: EiResponse [1..n]
EiDistributeTenderPayload

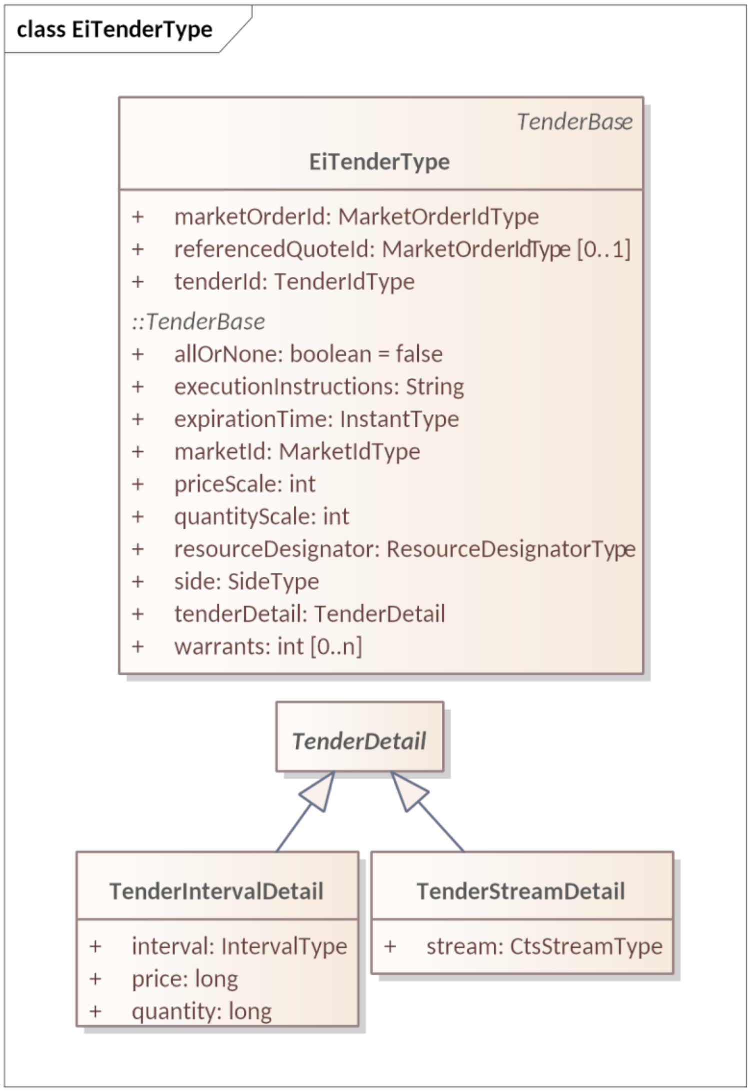


Figure 1-3: UML Class Diagram for EiTenderType showing attributes inherited from Tender Base

Of the attributes in Table 1-3~~the Operation Payloads for the~~ Tender ID and Referenced Quote ID (Referenced Quote Id) are unique to EiTender Type; the others are inherited from Tender Base and shared with EiQuote Type. See Section 9, *“The Negotiation Facet*~~Facet~~”, for a discussion of Quotes.

~~The following table describes the attributes for EiCreateTenderPayload~~

Table 1-3 ~~EiCreateTenderPayload~~: EiTender Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
Tender ID	UID		An ID for this Tender generated by the submitting Party	

Attribute	Type	FIX Field Name	Meaning	Notes
~~Counter Party~~Referenced Quote ID	UID	QuoteMsgID (1166)	~~The Actor~~ ID ~~for~~of the ~~CounterParty for~~Tradable Quote to which ~~the Tender~~this is ~~created~~a response.	~~This~~Optional. If Quote ID is ~~most frequently the PartyID for~~not known to the Market~~. To indicate a bilateral exchange, i.e., a~~ Segment, or if the referenced Quote has expired, then the Tender ~~between two specific parties, the PartyID of a specific Party~~ is ~~used~~rejected.
The fields below are as defined in the Tender, Table 1-2				
All or None	Boolean	In Execution Instructions	All or none of the tendered or quoted amount must be traded.	In Energy Interoperation 1.0 this was called *IntegralOnly*
Execution Instructions	String	ExecInst (18)	FIX Supports many instructions for how to execute a tender (or Tradable Quote)	See Table 1-1 below. Modeled as a String in CTS.
Expiration Time	Instant Type	ExpireTime (126)	The Tender or Quote expires at the specific time.	Always expressed in UTC
Interval	Start Time and Duration	*Not in FIX*	Start Date and Time for Product delivery together with Duration of delivery. Part of Instrument	While a Market Segment only accepts Tenders and Quotes of a single configured duration, the complete description is required to ensure validity and for off-market interactions.
Market ID	UID	MarketID (1301)	Identifies the Market	
~~Party~~Market Segment ID	Int	MarketSegmentID (1300)	~~The Actor ID for~~Identifies the ~~Party on whose behalf this~~Segment processing the Tender~~ is made.~~, Transaction, or Quote	~~Indicates which Actor proposes the buy or sell side EiCreateTender.~~This should be a unique combination paired with the Market Order ID
Order ID	UID	ClOrdID (11)	ID assigned by originating Party	
Market Order ID	UID	ORDERID (37)	ID assigned by the Segment or Market.	Used in acknowledgment and in all future market messages
~~EiTender~~Price	~~One or more EiTenders to be created.~~Long	Price (44)	The unit price for the Product being Tendered	~~In CTS an object describing a Tender is instantiated then sent; the latter is a consequence of processing an EiCreateTender payload.~~Amount is the product of Price and Quantity. Note that Price is subject to the Price Decimal Fraction value.

Attribute	Type	FIX Field Name	Meaning	Notes
Price Scale	Int	Not Defined in FIX	A multiplier for the Price	A Market Segment may be denominated in, for example dollars or 10ths of a cent.
Quantity	Long	OrderQty (38)	The quantity of the Product being Tendered	Must meet the SCALE and Round Lot requirements of the Segment
Quantity Scale	Int		A scale factor on the Resource unit for this Market	For example, “mega” vs “kilo” vs “femto-”
Resource Designator	Text	Not defined in FIX	~~The~~Identifier of the Resource being ~~tendered~~offered (Optional in many markets)	~~Must match the Market Resource Designator on receipt at the Market~~While a Market only accepts Tenders and Quotes for a single Resource, the complete description is required to ensure validity and for off-market interactions.
Tender Detail	Tender Detail	Not defined in FIX	Unit price and quantity for this tender	May be Interval or Stream as permitted
~~Request~~Tender ID	UID		~~A reference ID which identifies the artifact or message element. The Request ID uniquely identifies this request, and can serve as a messaging correlation ID[19].~~ID as submitted to Market	Identifies Tender until Market Order ID is assigned by Market
~~Responses~~ Tender Interval	Tender Interval Detail	Not defined in FIX	~~Responses for each attempted EiTender creation~~Interval, price and quantity for this tender	~~Array Of Responses [EI]~~Used in Interval Tender
Tender Stream	Tender Stream Detail	Not defined in FIX	Stream of consecutive Intervals with Prices and Quantities	Sometime referred to as a Load Curve in Power Markets.
Side	Side Type	Side (54)	Whether the Tender is to buy or to sell the Product	Buy or Sell
Warrant Ref	Int	*Not in Fix*	Reference to Warrants as defined in the Market	If used, see comments Warrants in Tenders, Section 5.3.3.

5.3.1 Interval Tenders and Stream Tenders

The most common Tender is the simple or Interval Tender, that is, an offer for a Product beginning at a specific date and time.

[19] As an example of the *Correlation Pattern* for messages

In financial markets, a *multi-leg order* submits simultaneous offers to buy or sell as a single order all of which must be accepted or rejected together. This specification describes a specialized type of multi-leg order for use in in some Market Segments which we term a Stream Tender. A Stream Tender defines a consecutive series of Intervals of identical Duration. The price and quantity tendered must be specified for each Interval.

For example, an industrial customer in a power market may intend to buy power to support a long running process. In power markets, such a sequence of power use is sometimes referred to as a *load curve*.

Such multi-leg orders are expressed using a CtsStream(see 3.2, *"**CTS Streams: Expressing Time Series**"*). While the information contained in a Stream Tender can be mapped precisely to a group of Interval Tenders, multi-leg semantics and processing of the related tenders leads to a Stream Tender. For example, All-or-None would refer to the entire set of Intervals in the Stream Tender.

Not all Market Segments permit Stream Tenders; some may require them. A Party submits a Stream Tender, when permitted or required, just as a Party submits an Interval Tender. A Market responds to the submission of a Stream Tender, when permitted or required, just as it responds to an Interval Tender.

The submission of a Stream Tender is restricted to Market Segments that specifically permit or require them and forbidden in all other Segments. See Section 13, "Market Structure Subscription".

Market Segments that support Stream Tenders SHALL also support Stream Quotes (if they support Quotes) and Stream Transactions. See Section 9, "The Negotiation Facet", for a discussion of Quotes.

5.3.2 Execution Instructions

FIX supports many Execution Instructions, only a profile of which are addressed in this specification.

Additional Instructions as defined by FIX are conforming but may not be implemented in all Venues. Since more than one may be present (albeit with semantic constraints) these are modeled as a string using single letters for each FIX Execution Instruction Code.

For example, the string *HKA* indicates the following:

- Cross is forbidden.
- Reinstate on system failure.
- Cancel on trading halt.

Table 1-1~~EiCreateTenderPayload with more than one EiTender SHALL be treated as a shorthand for sending each EiTender in a separate payload.~~

~~See . Note that if more than one EiTender is included in a single EiCreateTenderPayload, and the Market Characteristics include TGROUP = JBOT , there is no implication that there be an all or none meaning. This avoids the complexity of database-style transaction processing consistency, and simplifies implementations.~~

~~If TGROUP is ALLINPAYLOAD then all Tenders in the EiCreateTenderPayload SHALL be taken by the market to represent a supply or demand curve~~

~~TODO: Add example of how different components in a building can create their own bid curves independently.~~

~~10 Transaction Facet~~

~~10.1 Transaction Services~~

~~This section presents the Transaction Facet payloads, used by Actors in the role of creating and responding to Transactions.~~

~~This section makes them explicit, consistent with the definitions in Section .~~

~~here is no CTS payload that permits Canceling or modifying Transactions; given disparate ownership techniques from Distributed Agreement Protocols SHOULD be applied is not permitted.~~ presents a subset of the FIX Execution Instructions permitted for use in CTS.

Table 1-1~~: Transaction Management Service~~: Trading Instructions

~~Service~~Instruction	~~Request Payload~~FIX Code	~~Response Payload~~Abbreviation	Notes
No cross	A	[NoCross]	Cross is Forbidden
OK to cross	B	[OKToCross]	Cross is Permitted. In Segments that have Successors, this MAY permit conversion.
All or none – AON	G	[AllOrNone]	Ignored in deference to the AllOrNone attribute.
~~EiTransaction~~Reinstate on system failure	~~EiCreateTransaction~~H	~~EiCreatedTransaction~~[ReinstateOnSystemFailure]	~~Create and acknowledge creation of a Transaction~~Mutually exclusive with Q and l (lower case L).
Reinstate on trading halt	J	[ReinstateOnTradingHalt]	Mutually exclusive with K and m.
Cancel on trading halt	K	[CancelOnTradingHalt]	Mutually exclusive with J and m.
Cancel on system failure	Q	[CancelOnSystemFailure]	Mutually exclusive with H and l (lower case L).
Cancel if not best	Z	[CancelIfNotBest]	Cancel if order is not immediately matchable
Ignore price validity checks	c	[IgnorePriceValidityChecks]	
Suspend on system failure	l	[SuspendOnSystemFailure]	Mutually exclusive with H and Q.
Suspend on trading halt	m	[SuspendOnTradingHalt]	Mutually exclusive with J and K.

5.3.3 Use of Warrants in Tenders

Warrants increase the specificity of Product (and Instrument). A Buyer who does not specify a Warrant will be satisfied Delivery of a Product whether or not it has a Warrant. A Buyer who requests Product with a Warrant will only be satisfied by Delivery of a Product that has that Warrant.

Consider a buyer who wishes to buy a package of coffee beans and a buyer who wishes to buy a package of organic coffee beans. The word “Organic” on the label serves as a Warrant. The first buyer will buy solely on price, and is indifferent to seeing the word “Organic” on the label. The second buyer will choose only from among those packages with the warrant “Organic” on the label.

When a Tender on the Buy side specifies a Warrant, it must be rejected by any Market Segment that does not include that Warrant. A Tender on the Sell side that specifies a Warrant may be accepted by any Segment where the same Resource and Duration are traded. Conversely, a Tender on the Sell side without a Warrant must be rejected by any Segment that specifies a Warrant.

5.4 Contingent Tenders

Contingent Tenders are multiple Tenders submitted in a single message. The FIX List Order bundles multiple Tenders (Orders) with a common instruction (Contingency Type, 1385) that influences how fulfilling each Tender affects the other Tenders. A Market Segment either forbids or requires the use of Contingent Tenders. Tender Contingency Types are defined in the FIX Order List Contingency codeset (FIX 1385)

5.4.1 Illustrative Narrative on Contingent Tenders [Non-Normative]

The Contingency Type describes how the other Tenders in the List are affected by the acceptance of any one Tender in the Market. A common usage for Contingent Tenders is to submit a List of Tenders or StreamTenders (Load Curves), to support a long-running industrial process. The submitter wishes a contract for only one of the Tenders. In CTS Version 1, the only in Contingency is OCO or “One Cancels the Other” and it is expressed as a Boolean *atMostOne* in EiCreateTenderPayload and EiCreateStreamTenderPayload.

A Party submitting a List with the atMostOne = True is willing to accept whatever schedule matches the Transaction that returns from the Market.

A Party MAY wish to probe the market to make a more nuanced decision. This may include choosing one of several options. A decision to schedule a long-running process may depend upon being able to acquire a specific load curve over the entire schedule. A party that requires such complex contingent behavior should use the Negotiation Facet (section 9) to obtain Tradable Quotes, and then make its own choices based on those Quotes.

5.4.2 Rejecting a Tender

A Market may reject a Tender that violates market rules or which if transacted would violate the market’s integrity and other constraints (e.g. liquidity goals). Rejection Reasons include but are not limited to:

- Tender exceeds price limits on the potential transaction.
- Tender exceeds total value limits on the potential transaction.
- Tender violates total quantity limits for this Market Segment.
- Party is not in good standing with the Market.
- Tender violates lot size requirements of the Market Segment.
- Tender violates starting time requirements for instruments in the Market Segment.
- Market Segment is not open.
- Instrument is prior to temporal trading limits for this Market Segment.
- Instrument is past to temporal trading limits for this Market Segment.
- Tender is incomplete or corrupt.
- Referenced Quote not found.
- Referenced Quote has expired.

Details for rejection MAY be included in the EiResponse included in the EiCreatedTenderPayload.

5.5 Message Payloads for the Tender Facet

Figure 1-1 is a **[UML]** class diagram for the payloads for the Tender Facet operations. Note that each operation supports a Tender Set, and any set may consist of any number of Tenders, Interval or Stream.

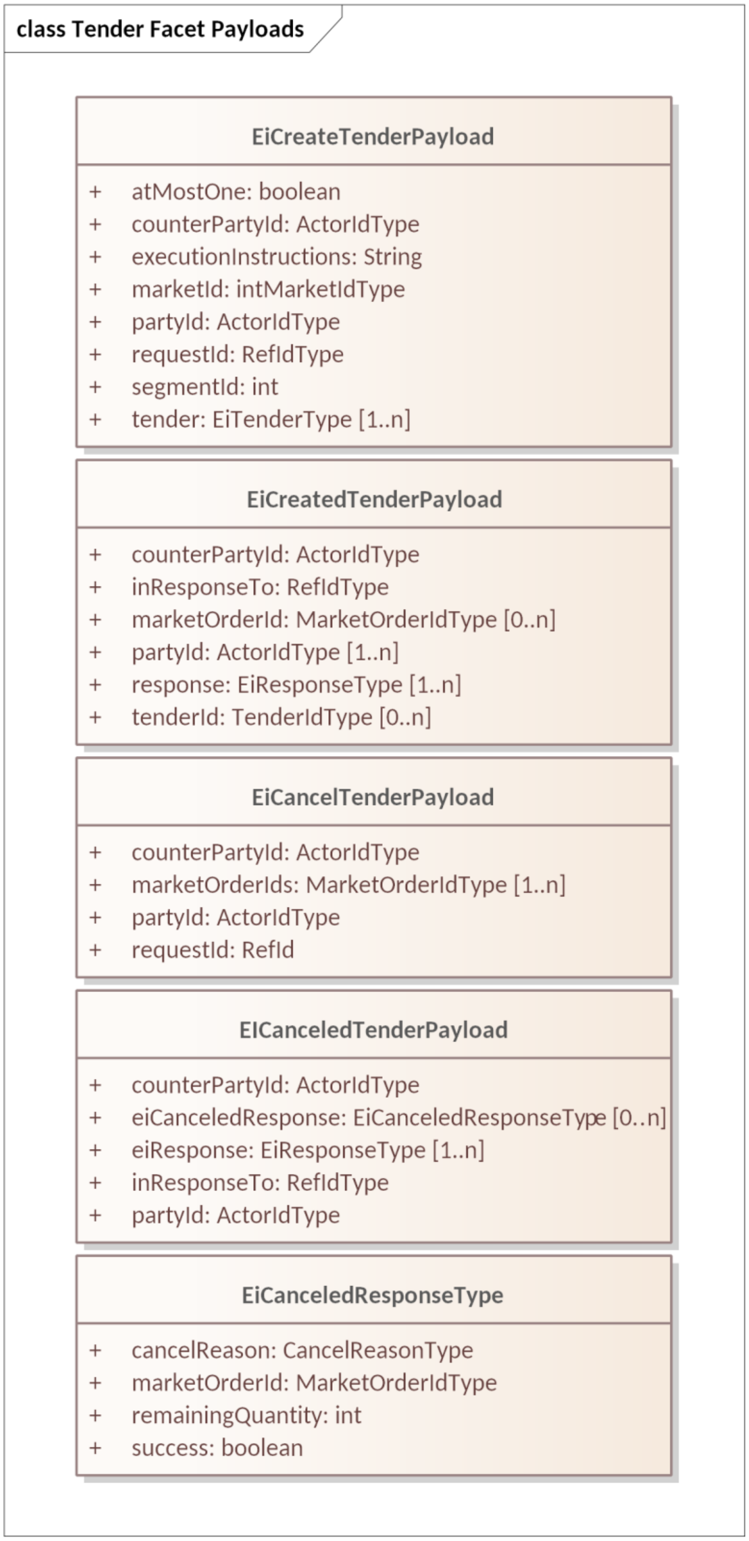

class Tender Facet Payloads
EiCreateTenderPayload
+ atMostOne: boolean
+ counterPartyId: ActorIdType
+ executionInstructions: String
+ marketId: intMarketIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ segmentId: int
+ tender: EiTenderType [1..n]
EiCreatedTenderPayload
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ marketOrderId: MarketOrderIdType [0..n]
+ partyId: ActorIdType [1..n]
+ response: EiResponseType [1..n]
+ tenderId: TenderIdType [0..n]
EiCancelTenderPayload
+ counterPartyId: ActorIdType
+ marketOrderIds: MarketOrderIdType [1..n]
+ partyId: ActorIdType
+ requestId: RefId
EICanceledTenderPayload
+ counterPartyId: ActorIdType
+ eiCanceledResponse: EiCanceledResponseType [0..n]
+ eiResponse: EiResponseType [1..n]
+ inResponseTo: RefIdType
+ partyId: ActorIdType
EiCanceledResponseType
+ cancelReason: CancelReasonType
+ marketOrderId: MarketOrderIdType
+ remainingQuantity: int
+ success: boolean

Figure 1-1 UML Class Diagram for Tender Facet Payloads

The Market Order ID is assigned by the Market on receipt of a Tender. The Market makes no assumption that the Tender ID (Order ID) submitted as part of the Tender is unique across all Parties in the Market. The Market responds with a Market Order ID for each Tender ID submitted. The submitting Party should record this Market Order ID, as it will be used in any Transactions awarded by the Market, and is required to cancel any Tender.

Specific Market Segments may limit all Tender submissions to either Interval Tenders or to Stream Tenders or may accept both. Specific Market Segments may restrict each Tender Set to all Interval Tenders or all Stream Tenders. Specific Market Segments may limit the cardinality of a Tender Set to any count. In the absence of such Segment specification, to support minimal interoperability, Interval Tenders are permitted, Stream Tenders, and the cardinality of each Tender Set is limited to one.

See Section 13.3 "*Segment Definition*" for details.

The following tables describe the attributes for the Tender Facet Payloads.

Table 1-2 EiCreateTenderPayload Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
At Most One	Boolean	See ExecInst (18)	Used to express alternatives, only one of which is to be effective	See Trading Instructions in Table 1-1. First match cancels other Tenders.
Counter Party ID	Actor ID		The Actor ID for the Counterparty for which the Tender is created.	In CTS, generally the PartyID for the Market. To indicate a bilateral exchange, i.e., a Tender between two specific parties, the PartyID of a specific counterparty is used.
Execution Instructions	String	ExecInst (18)	Execution Instruction.	Used only for multi-leg, and appliesto for all tenders in multi-leg. Execution instructions apply to each Tender in the List.
Market ID	UID	MarketID (1301)	Identifies the Market	
Market Segment ID	Int	MarketSegmentID (1300)	Identifies the Segment processing the Tender, Transaction, or Quote	This should be a unique combination paired with the Market Order ID
Party ID	Actor ID		The Actor ID for the Party on whose behalf this Tender is made.	Indicates which Actor proposes the buy or sell side EiCreateTender.
Tender	Ei Tender Type	OrderID (37)	Tenders requested to be created	One or more Tenders per Table 1-3: EiTender Attributes.
Request ID	Ref ID		An identifier for this Create Tender Payload	

Table 1-3 EICreatedTenderPayload Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
Counter Party ID	Actor ID		The Actor ID for the CounterParty for which the Tender is created.	In CTS, generally the PartyID for the Market. To indicate a bilateral exchange, i.e., a Tender between two specific parties, the PartyID of a specific counterparty is used.
In Response To	Ref ID		An identifier for Create Tender Payload to which this is a response	
Response	EiResponse Type		Specific error responses	See Section 2.4
The fields below are as defined in the Tender, Table 1-2				
Market Order ID	UID	ORDERID (37)	ID assigned by the Segment or Market.	Used in acknowledgment and in all future market messages

Attribute	Type	FIX Field Name	Meaning	Notes
Party ID	Actor ID		The Actor ID for the Party on whose behalf this Tender is made.	Indicates which Actor proposes the buy or sell side EiCreateTender.
Market Order ID	Market Order ID Type	OrderID (37)	The market-assigned identifier for the order in the Create Tender Payload to which this is a response	
Tender ID	Tender ID Type		Identifies the tenders in the Create Tender Payload to which this is a response	

Table 1-4 EiCancelTender Payload Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
Counter Party ID	Actor ID		The Actor ID for the CounterParty for which the Tender is created.	In CTS, generally the PartyID for the Market. To indicate a bilateral exchange, i.e., a Tender between two specific parties, the PartyID of a specific counterparty is used.
Party ID	Actor ID		Actor ID for the Party that created the Tender	
Request ID	Ref ID		An identifier for this Cancel Tender Payload	
The fields below are as defined in the Tender, Table 1-2				
Market Order ID	UID	ORDERID (37)	ID assigned by the Segment or Market.	Used in acknowledgment and in all future market messages

Table 1-5 EiCanceledTenderPayload Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
Counter Party ID	Actor ID		The Actor ID for the CounterParty for which the Tender is created.	In CTS, generally the PartyID for the Market
Party ID	Actor ID		The Actor ID for the Party on whose behalf this Tender was made.	Indicates which Actor proposes the buy or sell side EiCreateTender.
In Response To	Ref ID		An identifier for the Cancel Tender Payload to which this is a response	

Attribute	Type	FIX Field Name	Meaning	Notes
Ei Canceled Response	Canceled Response Type		Detailed response for each tender that was included in the EiCancelTender Payload	
EiResponse	EiResponse Type		Specific error responses	See Section 2.4

6 The Transaction Facet

This section presents the Transaction Facet, used by the Market to notify of the creation of Transactions. FIX terms the matching of a Buyer and a Seller as a "Trade". CTS follows EI (and the term transactive energy) in naming it a Transaction.

In the general case, the Market notifies each Party of the creation of a Transaction when two Tenders match as discovered by the Market's internal execution engine. To protect participant privacy, the market MAY use the MarketID as the counterparty to each Party receiving the Notification.

Unlike in financial markets, the market operator must still enforce limits imposed by physical infrastructure limits. For example, a substation or distribution cable will have physical limits for Power transferred during a given Interval. The reasons and mechanisms for such an enforcement are out of scope for CTS.

See Section 9, *"The Negotiation Facet"* for a discussion of Transactions based upon a Tradable Quote.

All Transactions are committed, that is, they cannot be cancelled or modified under normal market operations. Transactions in aggregate make up the Position. (See Section 7, The Position Facet for a discussion of Position.) A Party may thereafter choose to sell any portion or all of its Position in any instrument.

6.1 Messages for the Transaction Facet

A Transaction is created by a Market or Segment (See Section 13) based on some mechanism internal to the Market.[20] When a Market recognizes a potential Transaction, it creates a Transaction ID, and notifies the participating Parties.

Table 6-1: Transaction Management Service

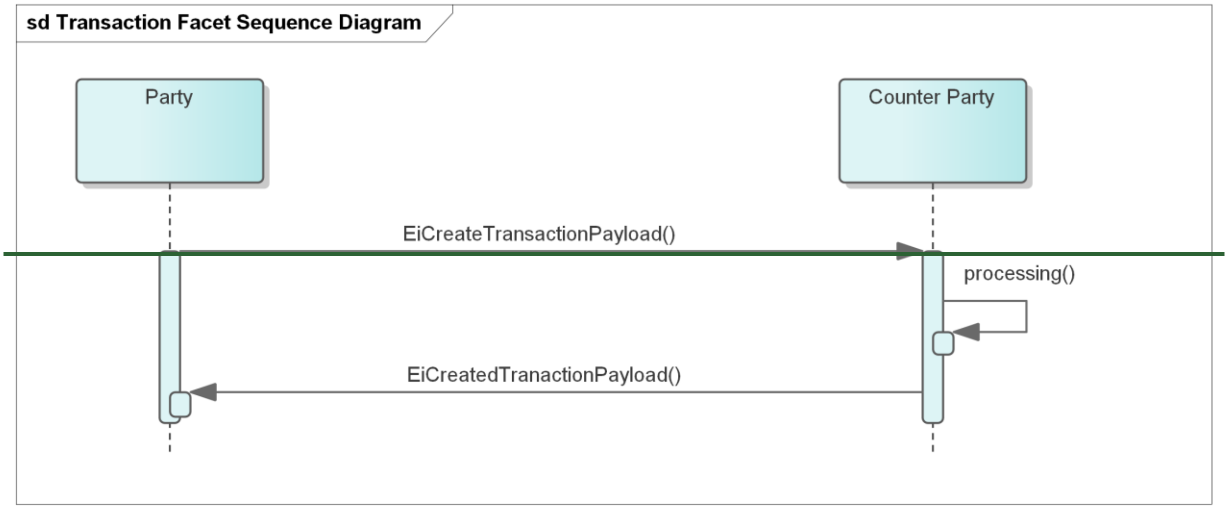
Facet	CTS Initial Message	CTS Response Message	Meaning
EiTransaction	EiCreateTransaction	EiCreatedTransaction	Create and acknowledge creation of a Transaction; typically initiated by the Market Segment engine
EiTransaction	EiCreateStreamTransaction	EiCreatedStreamTransaction	Create and acknowledge creation of a Stream Transaction; typically initiated by the Market Segment engine

~~10.2~~6.2 Interaction Pattern for the Transaction Facet

In Off-Market venues (see Section 13.1.1,"Venue Types~~This is~~"), the Parties match Tenders themselves, and inform the Market of their agreement. Even in Off-Market venues, the market operator must still enforce limits that affect physical integrity.

Figure 6-1~~[~~ shows the UML~~]~~ sequence diagram or the EiTransaction Facet:

[20] Some aspects of the market's mechanism(s) are visible to actors who are trading, generally where the mechanism affects rational bidding strategies. For example, bidding very low in a double auction market is reasonable (as you get the clearing price), but bidding very low in an order book market is not (as you get something similar to what you offered). See Section 13.1.1, *"Venue Types"*.


sd Transaction Facet Sequence Diagram
Party
Counter Party
EiCreateTransactionPayload()
processing()
EiCreatedTranactionPayload()

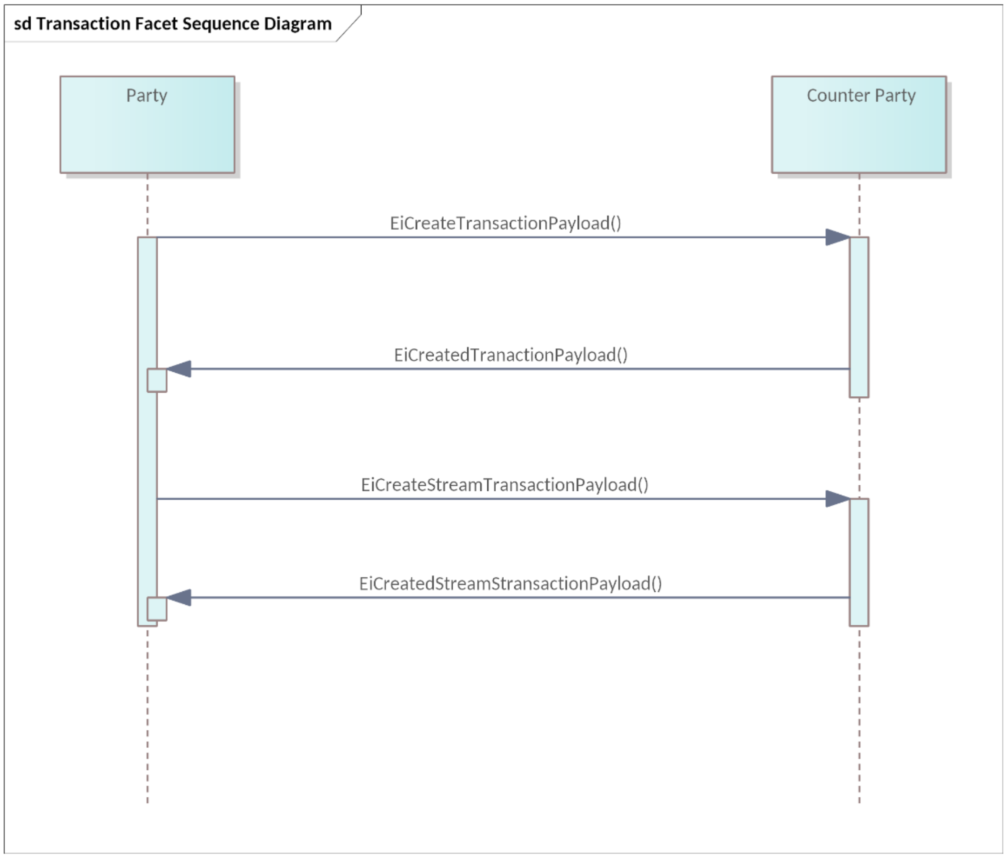

sd Transaction Facet Sequence Diagram
Party
Counter Party
EiCreateTransactionPayload()
EiCreatedTranactionPayload()
EiCreateStreamTransactionPayload()
EiCreatedStreamStransactionPayload()

Figure 6-1: UML Sequence Diagram for the EiTransaction Facet

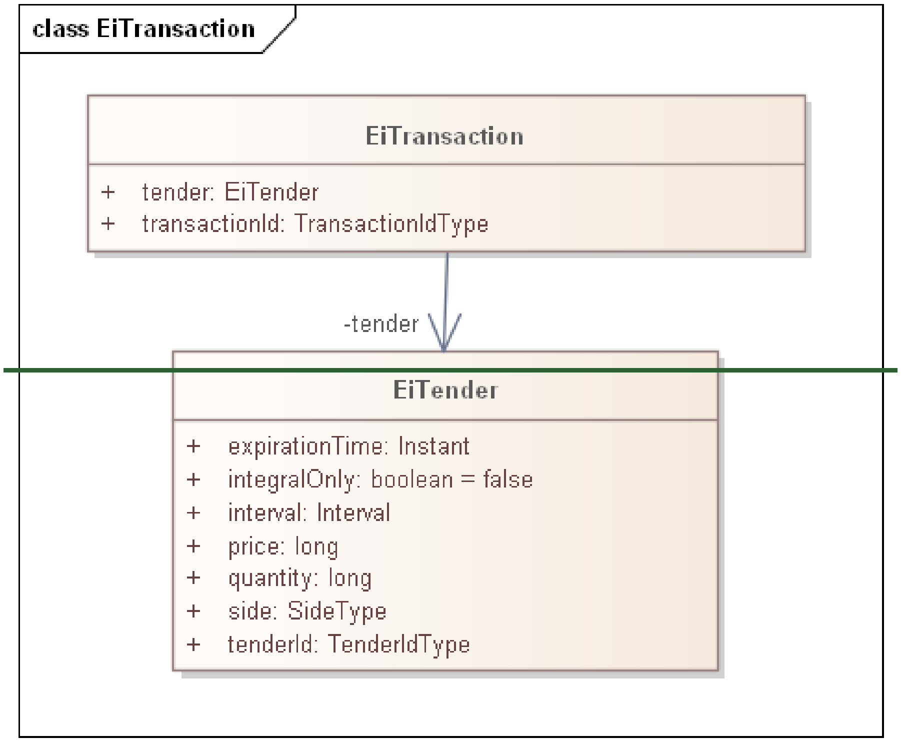
~~A transaction may be~~ Most Transactions are mediated by a market~~, in which case an EiCreateTransactionPayload is sent to each of~~. The Market matches Tenders, creates a Transaction, and notifies the ~~matched~~submitting Parties.

In Off-Market venues (see Table 13-1, *“Venue Types in CTS”*), the Parties match Quote and Tender, and inform the Market. Even in Off-Market venues, the market operator must still enforce physical or other limitations.

~~10.3~~6.3 Information Model for the Transaction Facet

~~Transactions are a CTS artifact evolved from EMIX including a Stream with time, quantity, and price. Flattening similar to that in the Tender Facet) is used.~~

The EiTransaction object includes the information in the original EiTender, possibly ~~rewritten~~updated to reflect the ~~clearing~~actual price and quantity rather than the requested price and quantity.


class EiTransaction
EiTransaction
+ tender: EiTender
+ transactionId: TransactionIdType
-tender
EiTender
+ expirationTime: Instant
+ integralOnly: boolean = false
+ interval: Interval
+ price: long
+ quantity: long
+ side: SideType
+ tenderId: TenderIdType

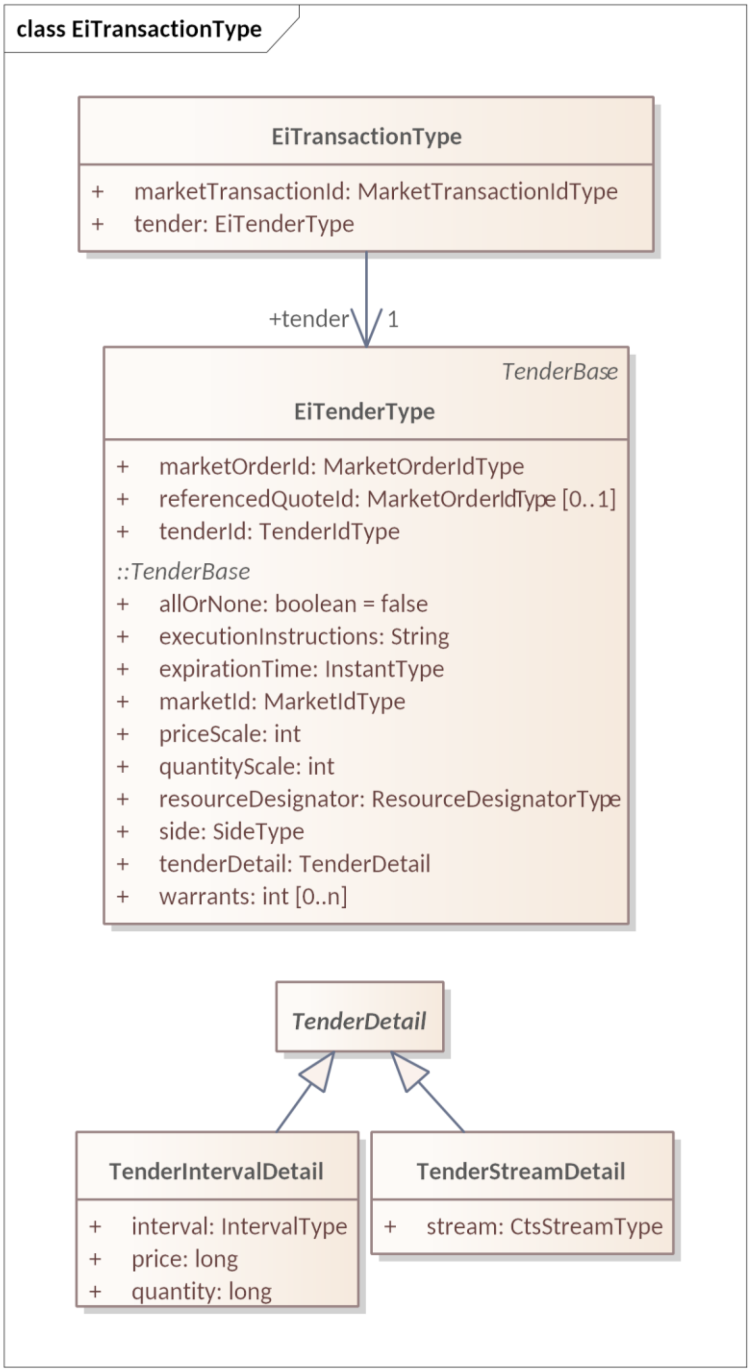


Figure 6-2: UML Class Diagram of ~~EiTransaction~~ EiTransactionType

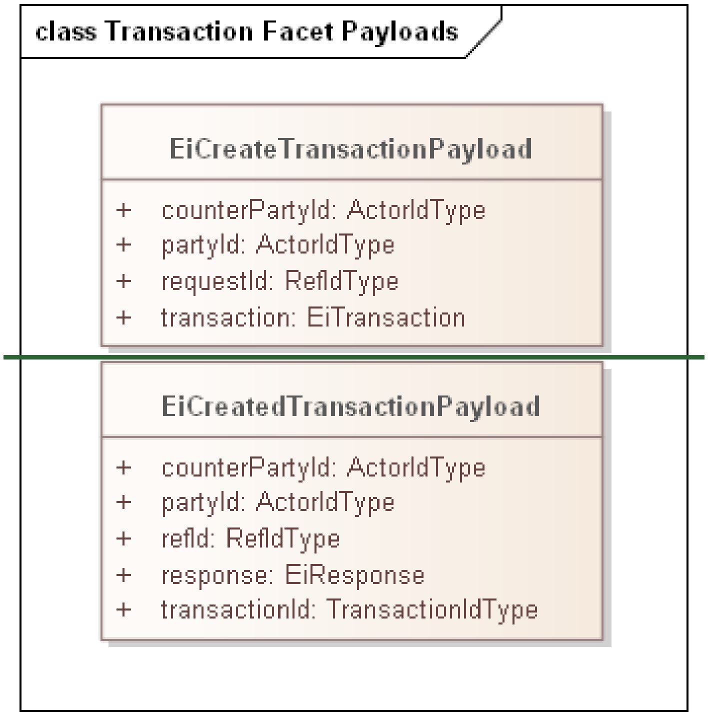
The attributes of ~~EiTransaction~~EiTransactionType are shown in Table 6-2~~the following table.~~.

Table 6-2: EiTransaction Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
~~Tender~~ Market Transaction ID	Market Transaction ID Type	TradeID (1003)	~~The Tender (Fig. 4-2) that led to~~ID Assigned this Transaction~~.~~ (Trade) by the Market (Segment)	~~The ID, quantity and price may differ from that originally tendered due to market actions.~~Note that Energy Interoperation defines Transaction ID differently than does FIX. This is assigned by the actor that performed the match, typically a market segment.
~~Transaction ID~~		~~An ID for this Transaction~~	~~The contained~~All other fields are as defined in the Tender~~ has its own Tender Id~~, Table 1-2	

~~10.4~~6.4 Payloads for the Transaction Facet

The **[UML]** class diagram in Figure 6-3 describes the payloads for the EiTransaction facet operations.


class Transaction Facet Payloads
EiCreateTransactionPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ transaction: EiTransaction
EiCreatedTransactionPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ refId: RefIdType
+ response: EiResponse
+ transactionId: TransactionIdType

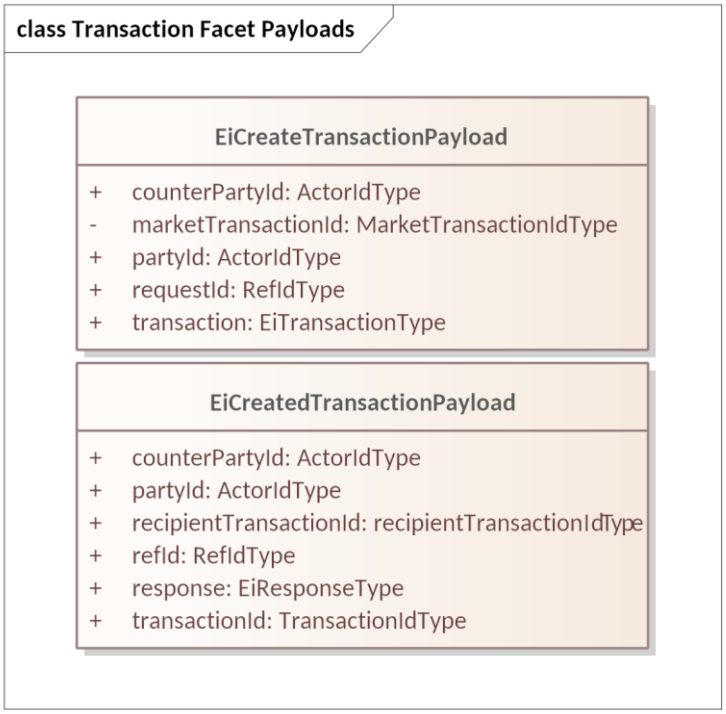

class Transaction Facet Payloads
EiCreateTransactionPayload
+ counterPartyId: ActorIdType
- marketTransactionId: MarketTransactionIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ transaction: EiTransactionType
EiCreatedTransactionPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ recipientTransactionId: recipientTransactionIdType
+ refId: RefIdType
+ response: EiResponseType
+ transactionId: TransactionIdType

Figure 6-3: UML Class Diagram of EiTransaction Facet ~~Operation~~ Payloads

The following tables list the attributes of the Transaction Facet Payloads.

Transactions are produced by a market or actor that performs matches; the resulting Transaction information is sent to the Parties whose Tender(s) are matched. Note that there is not a one-to-one relationship of Tender to Tender, or Tender to Contract. A Tender to buy one- hundred might match multiple Tenders to sell ten; this results in one Tender resulting in multiple Transactions. Each

Transaction is created by an interaction between a Tender to buy and a Tender to Sell. The Transaction payloads “echo” to each Tender to the Party that submitted it to become part of the Transaction.

The Tender included as part of a Transaction payload indicates a buy side or a sell side. When the Transaction indicates “buy”, then the PartyID is that of the Buyer. When the Transaction indicates “sell”, then the PartyID is that of the Seller. The CounterpartyID is the other participant in the Transaction.

As in financial markets often designate a “clearing” or “central” counterparty. Privacy concerns, particularly for transactions involving homes, are one reason for using the PartyID of the central counterparty. Under some rules, certain Parties must be revealed. For example, the PartyID of a dominant participant such as a distribution serving operator MAY be deemed public information; transactions involving such a designated participant would use the participant’s PartyID in the payload.

When use of a PartyID clearing counterparty is required, CTS uses the PartyID of the Market.

Table 6-3 EiCreateTransactionPayload Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
Counter Party ID	Actor ID	PartyID (448)	PartyID of the Party on the other “side” from the Tender in the payload.	May be the PartyID of the clearing counterparty.
Market Order ID	UID	ORDERID (37)	ID assigned by the Market when processing Tender	
Party ID	Actor ID	PartyID (448)	Party ID of the Party on the same “side” of the Tender in the Payload.	Side of the included transaction determines the Party.
Trade ID	String	TrdId (1003)	ID assigned to the trade entity once it is received or matched	Assigned by the Market
Reference ID	String	ExecId (17)	An identifier for this message	
Tender	TenderBase		Price and Quantity for Interval[s] in Transaction	

Table 6-4 EiCreatedTransactionPayload Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
Counter Party ID	Actor ID	PartyID (448)	PartyID of the Party on the other “side” from the Tender in the payload.	May be the PartyID of the clearing counterparty.
Party ID	Actor ID	PartyID (448)	Party ID of the Party on the same “side” of the Tender in the Payload.	Side of the included transaction determines the Party.
Trade ID	String	TrdId (1003)	Identifier for the Market’s ID for the received Transaction	

Attribute	Type	FIX Field Name	Meaning	Notes
Recipient Transaction ID	Recipient Transaction ID Type	XID	The ID assigned to the received Transaction by the recipient of the associated EiCreateTransaction	
Reference ID	String	ExecId (17)	The Ref ID for the message payload indicating the cleared Transaction	
Response	EiResponse Type		Specific error responses	See Section 2.4

~~10.5~~6.5 Comparison of Transactive Payloads

In this section we show the payloads for the Tender and Transactive Facets

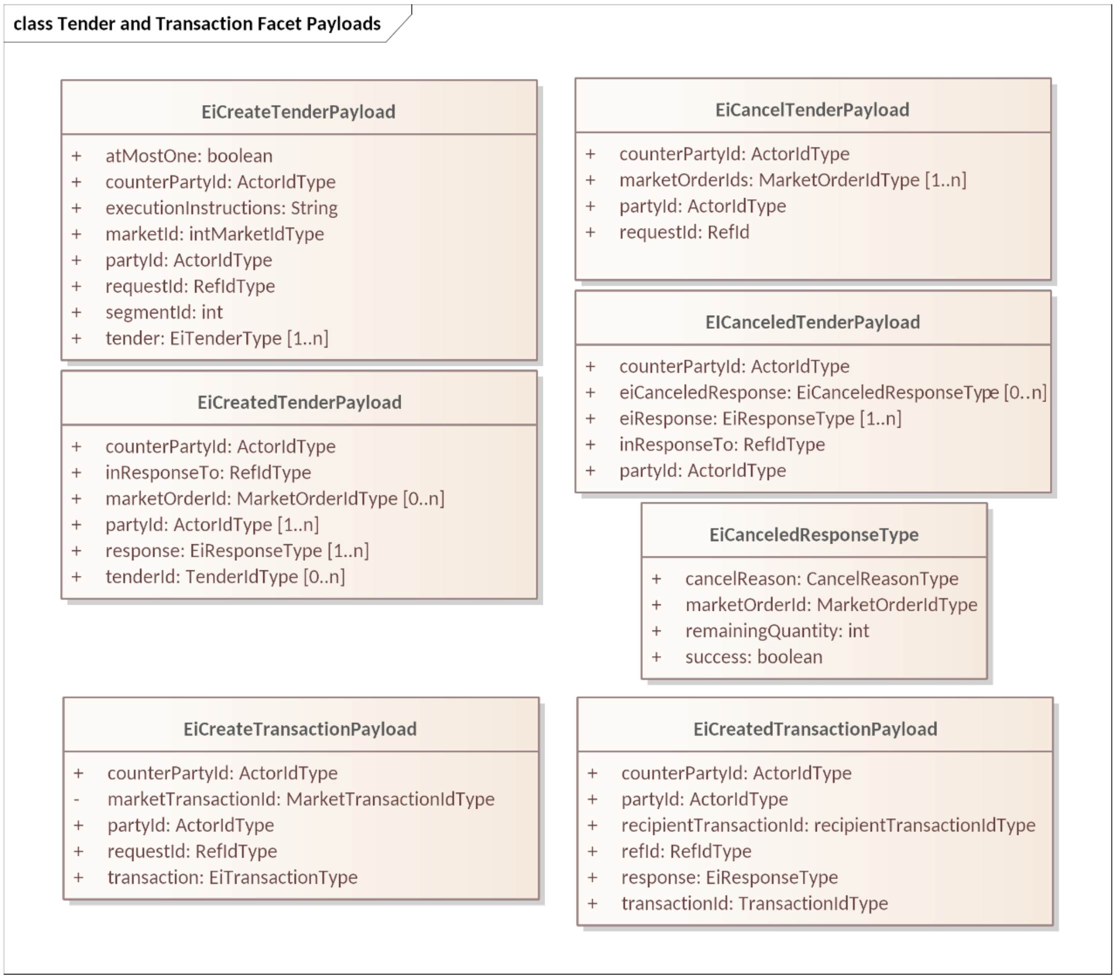
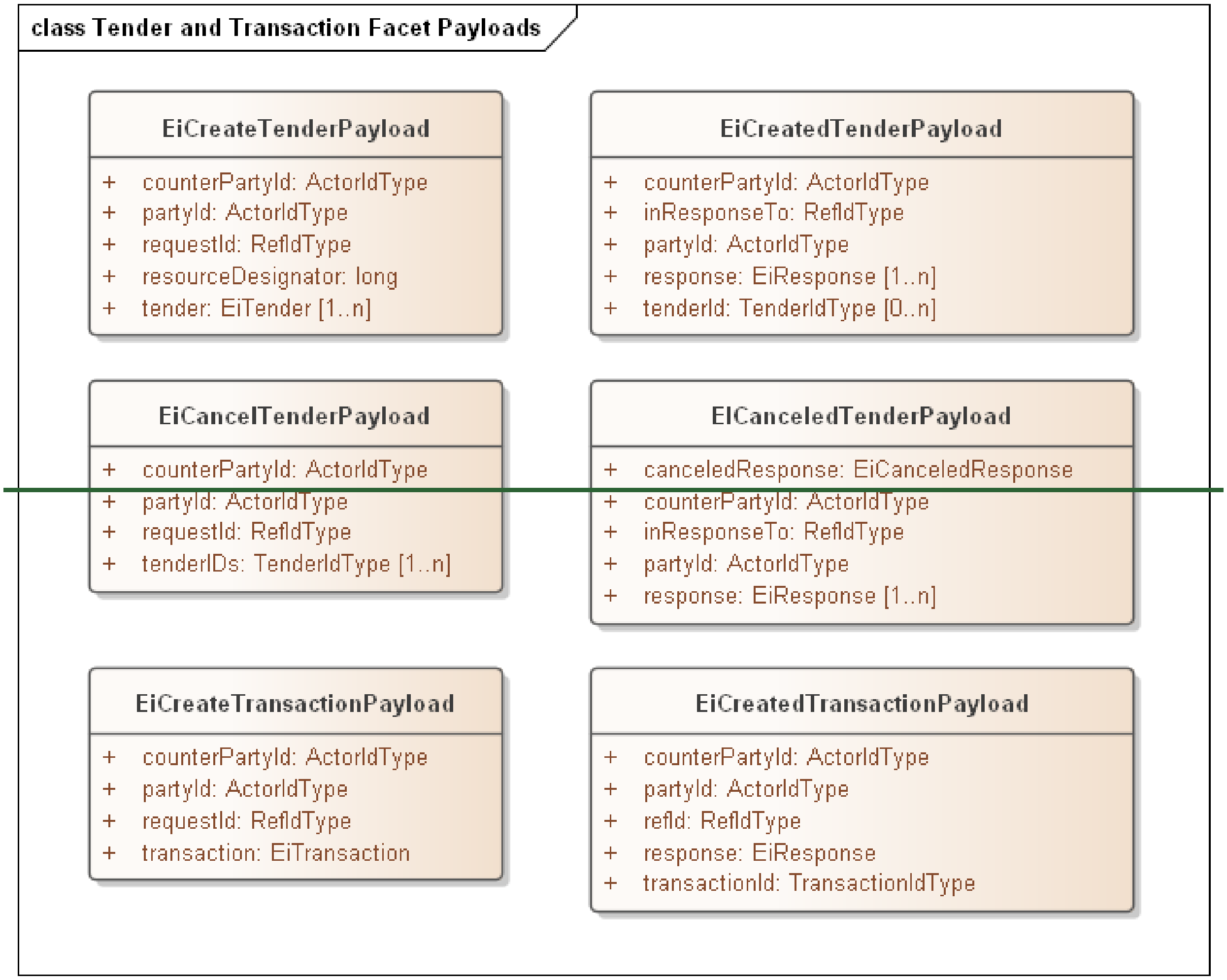

class Tender and Transaction Facet Payloads
EiCreateTenderPayload
+ atMostOne: boolean
+ counterPartyId: ActorIdType
+ executionInstructions: String
+ marketId: intMarketIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ segmentId: int
+ tender: EiTenderType [1..n]
EiCancelTenderPayload
+ counterPartyId: ActorIdType
+ marketOrderIds: MarketOrderIdType [1..n]
+ partyId: ActorIdType
+ requestId: RefId
EiCanceledTenderPayload
+ counterPartyId: ActorIdType
+ eiCanceledResponse: EiCanceledResponseType [0..n]
+ eiResponse: EiResponseType [1..n]
+ inResponseTo: RefIdType
+ partyId: ActorIdType
EiCreatedTenderPayload
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ marketOrderId: MarketOrderIdType [0..n]
+ partyId: ActorIdType [1..n]
+ response: EiResponseType [1..n]
+ tenderId: TenderIdType [0..n]
EiCanceledResponseType
+ cancelReason: CancelReasonType
+ marketOrderId: MarketOrderIdType
+ remainingQuantity: int
+ success: boolean
EiCreateTransactionPayload
+ counterPartyId: ActorIdType
- marketTransactionId: MarketTransactionIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ transaction: EiTransactionType
EiCreatedTransactionPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ recipientTransactionId: recipientTransactionIdType
+ refId: RefIdType
+ response: EiResponseType
+ transactionId: TransactionIdType

Figure 6-4: UML Diagram comparing Tender and Transaction Facet Payloads


class Tender and Transaction Facet Payloads
EiCreateTenderPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ resourceDesignator: long
+ tender: EiTender [1..n]
EiCreatedTenderPayload
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ partyId: ActorIdType
+ response: EiResponse [1..n]
+ tenderId: TenderIdType [0..n]
EiCancelTenderPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ tenderIDs: TenderIdType [1..n]
EICanceledTenderPayload
+ canceledResponse: EiCanceledResponse
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ partyId: ActorIdType
+ response: EiResponse [1..n]
EiCreateTransactionPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ transaction: EiTransaction
EiCreatedTransactionPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ refId: RefIdType
+ response: EiResponse
+ transactionId: TransactionIdType

6.6 Off-Market Transactions

While most transactions originate as Tenders submitted to the Market, are then matched by the Market, and ~~the resule~~result in a Transaction created by the Market, there are use cases for bilateral actions that generate a Transaction that did not come through the market.

For example, two parties within a market may choose to transact directly. A party may opt to buy directly from his neighbor's solar power. Another market may permit ~~Charity~~charity, that is~~ an anonymous~~, a donation to the Position of a neighbor. In either case, the Transaction is sent to the Market so that each Party's Position is maintained and so that the Buyer does not get double billed. These transactions ~~are~~may also be referred to as over-the-counter (OTC) agreements.

Off-Market agreements require both parties to report to the Market. The originating Party sends a Tradable Quote to the Market, including the ID of the counterparty. The simplest means is for one Party to publish a targeted Quote (see Section 9, *"The Negotiation Facet"*, below) naming the CounterParty in the Quote. The Counterparty then submits a Tender referencing that QuoteID and the terms on the Quote.

Some Markets will have specific Market Segments for Off-Market Transactions with specific message patterns. An OTC Market is notable for permitting violations of the Lot Size constraint and of the start time and duration constraints of other market segments. For example, in a Market with a Market Segment with a product of Lot Size 20 kWh and a Duration of one hour, an Off-Market execution could register a transaction of 23 kWh delivered over 27 minutes beginning at 2:48.

See Sections 13.2, *"Market Definition"* and 13.3 *"Segment Definition*~~OTC Agreements can span multiple instruments.~~

~~The parties must wait for the Market's acknowledgment and approval before they proceed with the delivery. This acknowledgment is by an EiTransactionCreated message to each Party.~~

~~[The Committee is most interested in comments on OTC Transactions, and the potential errors that would prevent them being registered. It is possible that EiRegisterTransaction and EiTransactionAcknowledged is a correct interaction.]~~

”.

~~11~~7 The Position Facet

The Position Facet provides the sum of a Resource transacted for by a Party, positive and negative, for each interval and for each Segment, within a possibly larger bounding Interval. For example, a Position sum all transactions over the course of a day. It is typically requested by an auditor or settlement agent (See Section 8 The Delivery Facet) or by a Party to get information about its own position.

For example, a Party may buy and sell from several Market Segments, perhaps with different Durations. A Party may also transact with specific counterparties in an Over-The-Counter (OTC) market. All of these are part of the Party's position. In most Resource markets, a Party may also take delivery (see Section 8, The Delivery Facet) which is measured by a meter. But what is the Quantity for this "self-executed" Transaction? This amount can be calculated by the difference between Position and Delivery and thereby creates Transactions for the used-but-never-bought Resource.

There may be other reasons to track Position. A market rule may require a Party designated as a Market Maker to maintain a Position of a certain quantity. A Party representing a Storage System may have specific rules for Position before a weather event. This specification does not catalog all the uses for Position that a Market or Party may require.

~~11.1~~7.1 Introduction

The purpose of the Position Facet is to allow access to the accumulated position for actors supporting specific Roles. A Party's ***Position*** for a time period is the algebraic sum of committed supply or sales for instruments overlapping that time period. A Party's position for an Instrument is evolved from an accumulation of trades for that Instrument.

~~Roles in using the Position Facet include~~

- ~~The Actor whose position is being requested—the *position Party*~~
- ~~An Actor who is authorized to request position information for other actors—including but not limited to an auditor—the *requestor*~~
- ~~The Market and Product for which the Position is being requested.~~

~~11.2 Position Definition~~

~~A Party's ***Position*** for a time period is the algebraic sum of committed supply or sale typically represented as purchases and sales expressed by means of EiCreateTransaction payloads for that instrument and Party.[21]~~

~~The time period for position intervals SHOULD be the same as for the underlying market used to buy and sell, but need not be; conversion of differing time granularity is programmatic and not required by this specification.~~

~~A Party needs to know both~~

- ~~The Party's projected needs for a time interval (not in scope)~~
- ~~The Party's committed net inflow and outflow for the interval~~

~~Note that committed inflow and outflow may be outside a market, e.g. local generation or battery interaction.~~

An Actor may, with appropriate authorization, request positions for other parties. This permits the specification and implementation of an auditor Actor. Roles using the Position Facet include:

- ~~An Actor sees its own Tenders and Transactions, and can maintain its own~~The Actor whose position~~. This facet allows~~ is being requested—the ~~offloading of that data management, but could in fact be a~~ *position Party.*

[21] One may say that a Party's position for an Instrument is evolved from an accumulation of trades for that Instrument.

- An Actor who is authorized to request ~~to a local~~ position information for other actors—including but not limited to an auditor—the *requestor.*

Position ~~manager~~Interactions follow the Streams pattern. A request for position includes a bounding interval The response reports, at least, the Position for each Interval included within the bounded Interval of the Request.

~~11.3 Interaction Pattern for the Position Facet~~

Table 7-1: Position Facet

Facet	Request Payload	Response Payload	Notes
Position	EiRequestPosition	EiReplyPosition	Request an Actor's Position(s) for a specific time interval, and reply with those Position(s) if access is authorized.

This is the ~~[~~UML~~]~~ sequence diagram for the Position Facet:

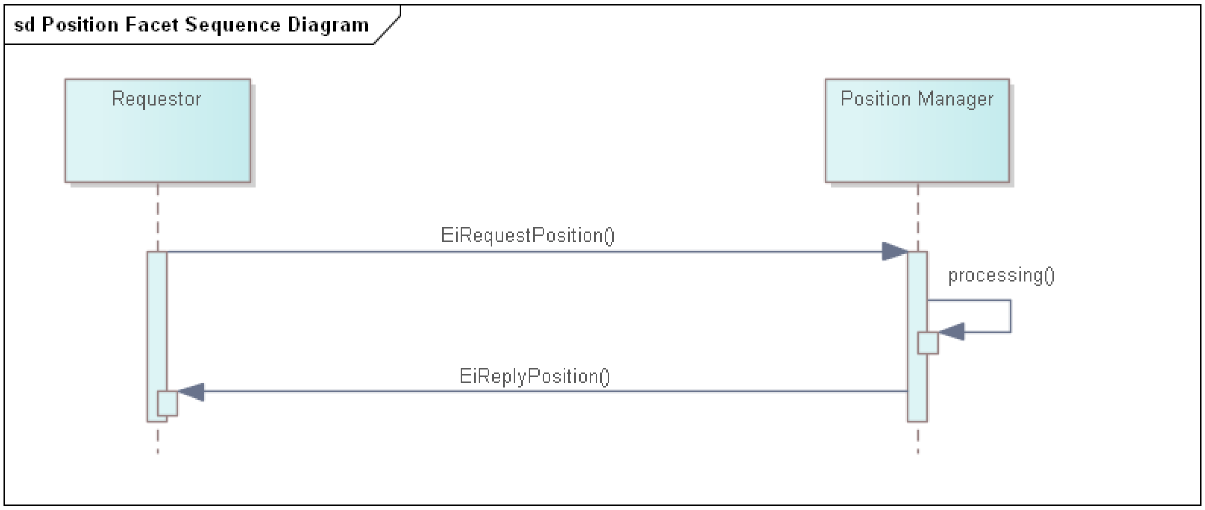


Figure 7-1: UML Sequence Diagram for the Position Facet

~~11.4~~7.2 Information Model for the Position Facet

For Position, a bounding interval is specified and the position in each interval contained in the closed bounding interval is returned. ~~An Actor has a position in a Product, and a Product specifies a temporal granularity or Interval duration. This Product duration defines the Interval duration for the returned CTS Stream. All elements of the stream share the duration and the stream has an explicitly stated start time~~A Request for Position specifies either a Product or a Resource.

When the Position Request is for a Resource, then the Position is assembled from all Transactions for that Resource. For example, a Transaction for Green Power, however defined, may only exist between 1:00 PM and 4:00 PM. The Position for Power for the rest of the day may be assembled from several sources, perhaps with different Warrants.

A ~~position~~Position is concerned with the total amount under contract, not the prices. If an Actor has positions in more than one Product, say, in a one-hour Product and in a one-minute Product, then ~~that requires two requests for position, and the two replies have different interval durations. The integration of these two Positions into a single combined Position is the responsibility of the Requestor~~the returned Position SHALL use the shorter Duration.

The attributes are shown in the following section.

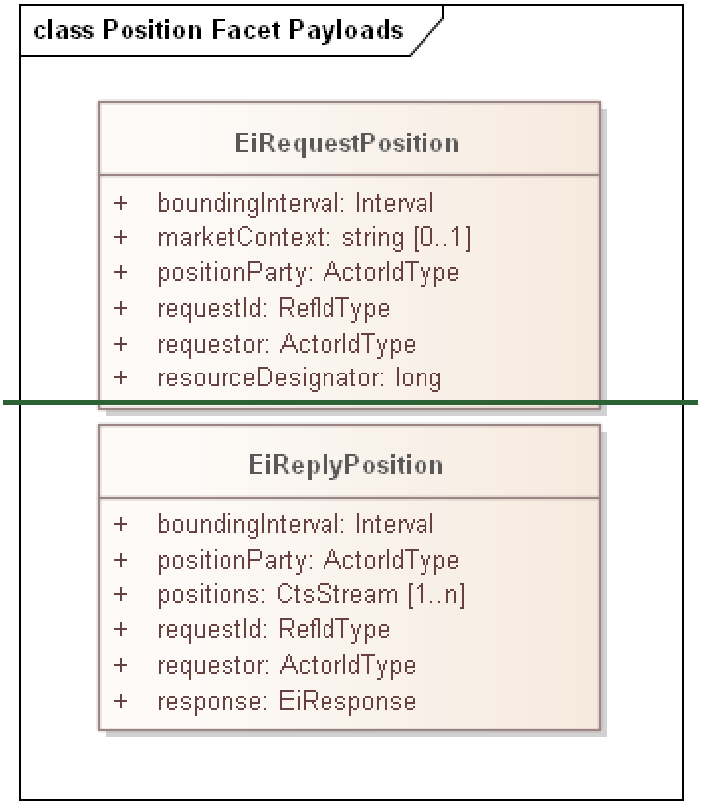
~~11.5~~7.3 Payloads for the Position Facet

The Position payload is in the format of a CTS Stream, with only a Quantity in the Interval Payload.

~~TODO: discuss overlapping positions, as in 1 Hour position overlaid with a 5 minute position.~~

Position stated against the sum of Transactions in all Segments.

The **[UML]** class diagram describes the payloads for the Position facet.


class Position Facet Payloads
EiRequestPosition
+ boundingInterval: Interval
+ marketContext: string [0..1]
+ positionParty: ActorIdType
+ requestId: RefIdType
+ requestor: ActorIdType
+ resourceDesignator: long
EiReplyPosition
+ boundingInterval: Interval
+ positionParty: ActorIdType
+ positions: CtsStream [1..n]
+ requestId: RefIdType
+ requestor: ActorIdType
+ response: EiResponse

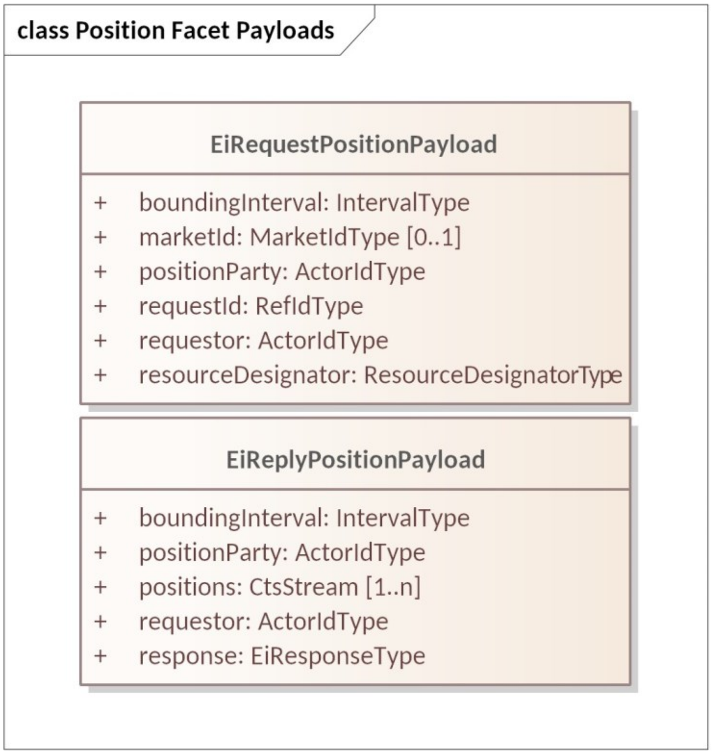

class Position Facet Payloads
EiRequestPositionPayload
+ boundingInterval: IntervalType
+ marketId: MarketIdType [0..1]
+ positionParty: ActorIdType
+ requestId: RefIdType
+ requestor: ActorIdType
+ resourceDesignator: ResourceDesignatorType
EiReplyPositionPayload
+ boundingInterval: IntervalType
+ positionParty: ActorIdType
+ positions: CtsStream [1..n]
+ requestor: ActorIdType
+ response: EiResponseType

Figure 7-2: UML Class Diagram of Payloads for the Position Facet

Table 7-2: Attributes of Position Facet Payloads

Attribute	Meaning	Notes
Bounding Interval	The [closed] time interval for which position information is requested. The first Positions Stream element starts at or after the start of the Bounding Interval. The last Stream element ends at or before the start of the Bounding Interval.	
Position Party	The Party whose position is being requested.	Allows a request for another Party's position, with appropriate privacy and security constraints
~~Market Context~~Resource Designator	~~The market context of interest~~The Resource for which Position is being requested	~~Used to determine the Resource for position. If not present, any Resource of which the responder is aware, with no claim to completeness, will be used~~Should match the identified Market's Resource Designator
Market ID	Identifier of the market of interest	A Party MAY be able to participate in more than one Market See Section 13.
Request ID	A reference to this payload	May be used as a correlation ID
Requestor	The Party requesting the position.	A failure indication will be returned if the Requestor is not authorized to access position information for Position Party. ~~Addresses the auditor use case.~~
Positions	CTS Streams containing the positions for Position Party for each Resource. Positions are signed ~~or~~and may be zero.	Each ~~CTS Stream~~CtsStream interval that is contained within the Bounding Interval will have a value associated (signed integer~~, zero permitted~~). Note that a ~~CTS Stream~~CtsStream contains a Resource Designator
Response	An EiResponse~~. Will~~ will indicate failure if Requestor is not authorized to access position information for Position Party for any of the requested intervals.	

The following system-specific requirements are out of scope:

- Different systems may support Position requests for different purposes. An Actor MAY request its own position(s) to recover from failure.
- ~~Positions MAY be used to compute Actor reliability.~~
- A supplier of last resort MAY compare Positions to Delivery to impute transactions for unpurchased power delivered. (See 8 The Delivery Facet)

~~12~~8 The Delivery Facet

The CTS Delivery Facet can be considered as the meter telemetry facet. We ~~term~~name it “Delivery” to align with the market focus of this specification, ~~i.e.,~~that is, a building takes delivery of power, or a distributed energy Resource (DER) delivers power. A CTS Delivery payload contains a ~~CTS Stream~~CtsStream that conveys the measured or computed flow of a specific Resource through a particular point on the ~~Product’s~~Resource’s delivery network ~~between particular times~~during a specific Interval.

CTS Delivery is typically derived from reading one or more meters, but it may be computed, implied or derived from some other method. Every ~~contract~~Transaction is between a Party that promises to buy and a Party that promises to sell. Consider an actor that performs temporal arbitrage, i.e., buys one-hour Products and sells one-minute Products during the same hour. The Actor MAY report that it took delivery in each minute of that Interval, and the sales to other Actors MAY be visible only as reductions as recorded in Delivery.

In most ~~TE markets~~cases, a node that takes delivery of more power or other Resource during an Interval than contracted for must eventually pay for that delivery. For example, An *auditor*~~,~~ (however defined~~,~~) could sum all positions (See section 7, *The Position* Facet) and compare the result to total Delivery. The Auditor can then impute a transaction for the over-delivery. This may not be a simple “spot price”; if multiple Actors are taking over-delivery, then the last ~~small~~ transaction is likely underpriced. Systems that track “actor reputation” may lower the reputation score. These examples explain the potential use of the information delivered by this facet~~,~~ and are not meant ~~not~~ to suggest or dictate any particular business process or system model.

A CTS Delivery payload reports on the flow of a Resource ~~because~~and the duration of that report stream may not match the temporal granularity ~~MAY not match that~~ of any particular Product. The payload may (e.g.) ~~report~~include the sum of a one-hour market and of a one-minute market for the same Resource.

A CTS ~~Marketplace~~Market MAY have ~~expectations about levelized load~~assess penalties for Delivery outside certain bounds from the Position—as do many of today’s tariffed markets. ~~Exceeding the limiting bounds for Delivery may result in a market penalty. It~~Such bounds and penalties are out of scope for CTS. Computation and notification of Penalties is outside ~~the scope of this specification to define the bounds or the nature of the penalty~~of scope.

A request for delivery specifies a Resource, ~~physical granularity~~unit of measure, and a temporal granularity~~.~~ [Duration]. While the ~~physical granularity~~unit of measure and temporal granularity need to be within the capabilities of the telemetry node, they need not match any particular Product~~.]~~.

~~12.1~~8.1 Interaction Pattern for the Delivery Facet

Table 8-1: Delivery Facet

Facet	Request Payload	Response Payload	Notes
Delivery	EiRequestDelivery	EiReplyDelivery	Request Delivery through a specific Measurement Point

Figure 8-1 is the ~~[~~UML~~]~~ sequence diagram for the Delivery Facet~~:~~.

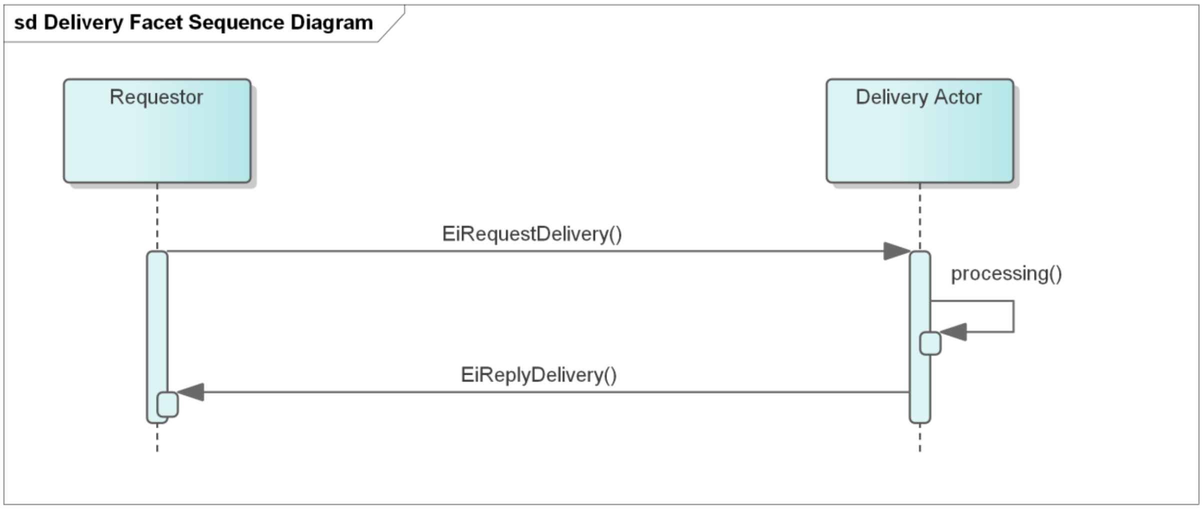


Figure 8-1: UML Sequence Diagram for the Delivery Facet

~~12.2~~8.2 Information Model for the Delivery Facet

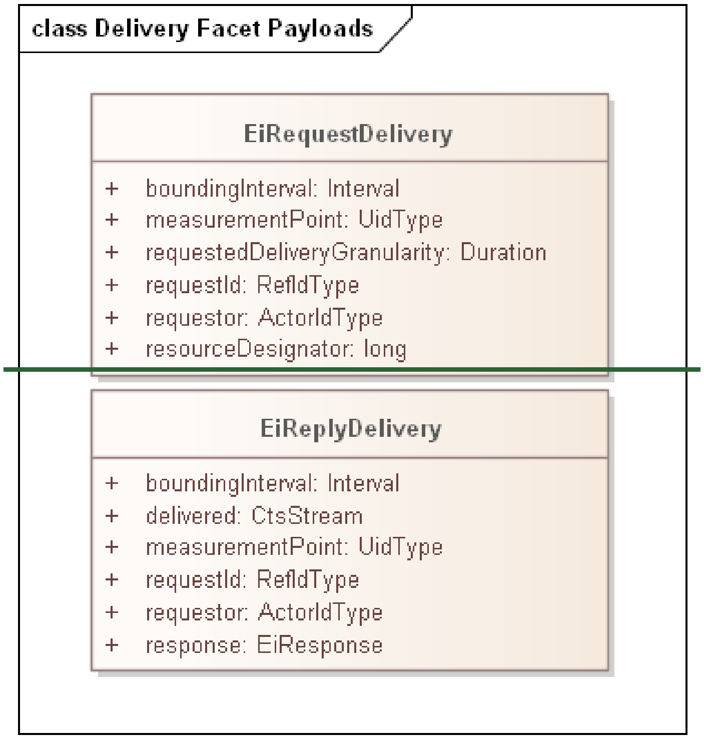
A Delivery response returns a single ~~CTS Stream~~CtsStream of intervals of the requested Duration, with a quantity in each.

As with the Position Facet a bounding interval is specified and the delivery in each interval contained in the closed bounding interval is returned. The temporal granularity as requested MAY not be available, or the Delivery Actor may convert and combine—for example a request for one hour delivery intervals could be responded to using information from 1 minute or 5-minute measurement cycles.

The attributes are shown in the following section.

~~12.3~~8.3 Payloads for the Delivery Facet

The **[UML]** class diagram describes the payloads for the Delivery facet.


class Delivery Facet Payloads
EiRequestDelivery
+ boundingInterval: Interval
+ measurementPoint: UidType
+ requestedDeliveryGranularity: Duration
+ requestId: RefIdType
+ requestor: ActorIdType
+ resourceDesignator: long
EiReplyDelivery
+ boundingInterval: Interval
+ delivered: CtsStream
+ measurementPoint: UidType
+ requestId: RefIdType
+ requestor: ActorIdType
+ response: EiResponse

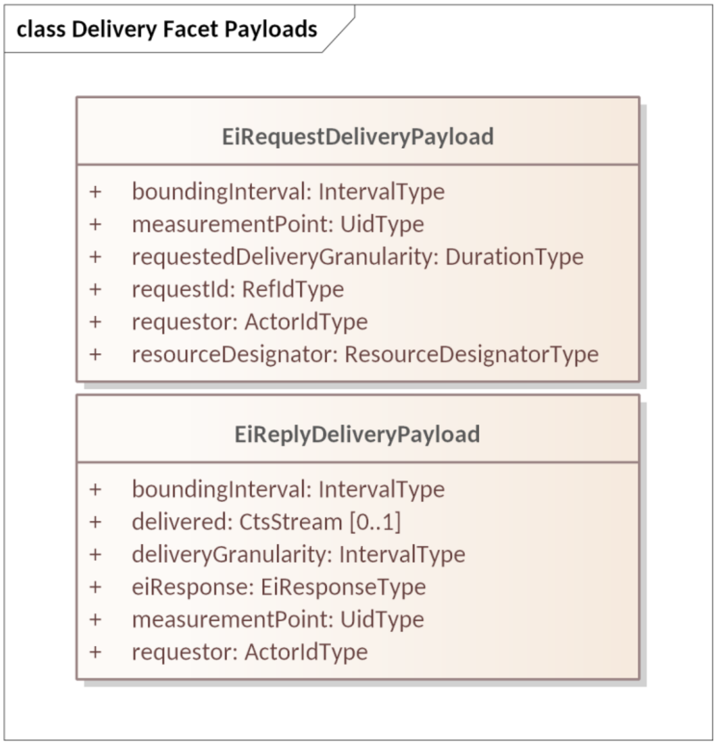

class Delivery Facet Payloads
EiRequestDeliveryPayload
+ boundingInterval: IntervalType
+ measurementPoint: UidType
+ requestedDeliveryGranularity: DurationType
+ requestId: RefIdType
+ requestor: ActorIdType
+ resourceDesignator: ResourceDesignatorType
EiReplyDeliveryPayload
+ boundingInterval: IntervalType
+ delivered: CtsStream [0..1]
+ deliveryGranularity: IntervalType
+ eiResponse: EiResponseType
+ measurementPoint: UidType
+ requestor: ActorIdType

Figure 8-2: UML Class Diagram of Payloads for the ~~Position~~Delivery Facet

Table 8-2: Attributes of Delivery Facet Payloads

Attribute	Type	Meaning	Notes
Bounding Interval	Interval	The [closed] time interval for which position information is requested.	~~The [closed] time interval for which position information is requested.~~ The first Positions Stream element starts at or after the start of the Bounding Interval. The last Stream element ends at or before the start of the Bounding Interval~~.~~
Measurement Point	ID	~~The~~ An identification of the Point ~~for which telemetry is provided about the~~ where measurements are made of the flow of the ~~resources..~~resource.	~~Allows a request to any Measurement Point for information on Resource flow at that point over time.~~ Information should be secured in conformance with appropriate privacy and security constraints
Request ID		A reference to this payload	May be used as a correlation ID
Requestor	PartyID	The Party requesting the position.	A failure indication will be returned if the Requestor is not authorized to access position information for Position Party. Addresses the auditor use case.
Delivered	~~A CTS Stream containing the Delivery information for the Resource. Delivery value is signed or zero.~~CtsStream	~~Each CTS Stream interval that is contained within the Bounding Interval will have a value associated (signed integer, zero permitted). Note that a CTS Stream contains a Resource Designator which SHOULD match that in the requested Resource Designator~~A CtsStream containing the Quantity delivered in each Interval.	

Attribute	Type	Meaning	Notes	
Response		An EiResponse. Will indicate failure if Requestor is not authorized to access position information for	~~An EiResponse. Will indicate failure if Requestor is not authorized to access position information for Position Party for any of the requested intervals.~~ If the Requested Delivery Granularity cannot be used, MAY indicate what granularity can be used.	

9 The Negotiation Facet

Negotiations are part of what FIX terms Pre-Trade Information. So far, this specification has discussed interactions between a Party and a Market in which some internal process decides how a Transaction is created. This section describes instead how Parties come to an agreement to create a Transaction through direct communication. This conversation is conducted using various types of quotations. The Market facilitates the quote process but does not intervene—it acts as a neutral party.

Any Segment may support Negotiation as indicated in the Market Structure (see Section 13, *"Market Structure Subscriptions")*.

9.1 The Negotiation Process (non-normative)

The Negotiation process is inherently flexible. A Transaction may come after many rounds of negotiation, or directly from a response to the first tradable quote. This section describes some potential interactions to clarify the concepts before we define the message types in the following sections.

A Party that wishes to transact some amount of a resource, to find a potential counterparty, or to arrive at an agreement with a specific known counterparty begins a Negotiation by sending a Request for Quotation (RFQ), an Indication of Interest (IOI), or perhaps a unsolicited Quote. In CTS, the distinction between an IOI and Quote is Boolean variable "tradable" which is false for an IOI and True for a Quote.

The initial message may be general, advertised to all participants in a Segment, or targeted, sent to one or more chosen Parties. The IOI is non-tradable and sent to elicit a counteroffer which could be a tradable Quote, another IOI, or an RFQ. A tradable Quote, whether solicited or unsolicited, invites a Tender which will result in a Transaction.

Financial markets assume that the same party, called the Issuer, initiates all quotes in a specific negotiation. The recipient of a quote can accept the quote, if it is tradable and the terms are agreeable, or reject the quote. When a Party accepts ("hits" or "lifts") a tradable quote, the Market executes the Transaction—the issuer of the quote cannot back out. The market facilitates the quote process but does not intervene—it acts as a neutral party.

CTS negotiations differ from financial practice in that in financial negotiation, the instrument never changes. Over the course of a CTS negotiation, the time of delivery may change, which is a change of Instrument.

And RFQ uses a Bounded Interval to indicate what an acceptable IOI or Quote would be.

- Consider Party A that wishes to buy 15 kW of power over a two-hour period, sometime within an 8-hour window. This would take the form of an RFQ with an eight hour bounding interval with a specific start time, but with a Stream of two Intervals with a Duration of 1 hour but with no starting time specified.

- Consider instead that Party A further wishes to buy 10 kW of energy over an hour at $0.05/kWh sometime during the work day. Party A create an RFQ, with a bounding duration of the workday, containing a single unscheduled Interval of one hour containing the Price and the Quantity.

Party A and B can send these RFQs directly to one or more potential counterparties or advertise it to the entire market. Because it is not tradable, the Market does not need to know about or register the RFQ. The response is a quote, either indicative (IOI) or tradable (Quote).

Party A may receive one or more offers in response. These become more specific, perhaps two Quotes issued by the same counterparty with different prices at different times. A quote issuer may make a counteroffer by sending a quote proposing different quantities and/or prices. Perhaps the responding Party considers that it will turn on a generator, but only if it can operate the generator at an economic rate for an economic duration. A quote MAY be for only one of the two hours indicated in the original RFQ, leaving the requesting Party to find an acceptable match from among all the offers (quotes) it receives. The prices may be higher or lower than requested in the original RFQ. Until one party issues a tradable Quote, than all responses are RFQs or IOIs, issued to continue the negotiation.

When the respondent thinks that there is an essential meeting of requirements, that Party submits a tradable Quote, that is a quote that a matching Tender will turn into a Transaction. For a CTS quote to be tradable, the Party informs the Segment, even if it is a private quote and not generally advertised.

To accept a tradable Quote, a Party submits a Tender to the Segment, referencing the Quote ID, and matching the details of the quote. The Segment compares the Tender to the quote, and, if they match, it awards the Transaction without going through the Segment's matching engine. All tradable quotes are treated as if they are marked All-or-None (AON).

The issuer MUST accept a Tender received in response to a tradable Quote.

~~13~~ Negotiations may include Interval Quotes or Stream Quotes, a pattern that matches that of Tenders (See Section 5.3.1, *“Interval Tenders and Stream Tenders* ~~Market Information—the Quote and Ticker Facets~~

~~Tenders are typically private in a market, whether the market matches tenders using an order book, a double auction, or some other means to match buyer and seller to award contracts. Markets generate order by enabling price knowledge to emerge from the tenders of independent actors. If all tenders are public, then this price cannot emerge. No seller would ever offer a price less than the highest outstanding tender to buy; no buyer would ever offer a price higher than the lowest outstanding tender to sell. Moreover, analysis of tenders can reveal detailed information about the market participant beyond that necessary to balance supply and demand. (See *Appendix* , .)~~

~~Even so, some Actors may wish to advertise specific Tenders. In a transitional environment, a utility may wish to publish day ahead prices for each hour of the day. An Actor may wish to draw others into the market quickly in response to a system failure or unplanned-for need—and may offer an unusually high or low price to attract sellers or buyers. Others may wish to quickly dispose of a previous position. A distribution operator in TE markets may wish to advertise short term deals temporal price boundaries to protect grid components by smoothly ramping power delivery requirements. Whatever the reason, [EI] specifies the EiQuote service for advertising Tenders.~~

~~Transaction prices are public information. Consider a financial market, which lists the current stock price, or more precisely, the price of the last transaction. Parties use this public information to plan whether to submit new Tenders, or perhaps to cancel old Tenders. An old technology for broadcasting financial transactions was the ticker producing ticker tape, so named for the sound it made with each transmission. CTS uses the term “Ticker” for the completed transaction information.~~

~~The information payloads for Quotes and Tickers are nearly identical.~~

~~For each Market, there MAY be URI for the Ticker service and for the Quote service. The Marketplace Characteristics MAY include a URI for the Ticker service and a URI for the Quote service.~~

~~This specification has no position on the whether a common Ticker for all Markets for a given Resource is better than a Ticker for each Market. Similarly, a Marketplace MAY choose to put all “Green” (however defined) Products in one Ticker stream and “conventional” in another. Any number of Markets within a Marketplace MAY use the same URI for the ticker service, and other Markets share another. Such decisions are left up to the developers of CTS-based systems.~~

~~Quotes use the same mechanisms, and like Tickers may have many Markets or a single Market in a single Quote service.~~

~~13.1 Quotes~~

~~[EI] defines a quotation as a market price or possible price, which does not replace the Tender and acceptance to reach a Transaction. The Quote message looks very much like a Tender.~~

~~As noted above in Section , the , a Party may wish to advertise certain of its Tenders to the market. An advertisement of an attractive price for limited amount of power might only be available to the first to respond Such a public Tender is distributed to other Parties by the Quote Facet by including a Tender ID for the parallel Tender.~~

~~Publish-Subscribe semantics are a likely communication paradigm.~~

~~13.1.1 Interaction Pattern for the Quote Facet~~

.”) A Stream Quote must be matched to a Stream Tender to create a Transaction. A requester that wishes to get a Stream Quote back indicates it in the RFQ or IOI. The stream in an RFQ need not fill the

Bounding Interval; an overnight bounding interval of fifteen hours may be seeking any proposal three-hour stream during that interval.

There are three scenarios for negotiation below to illustrate the flexibility of Negotiation: (1) Single-provider, (2) over-the-counter, and (3) system recovery.

1) In certain Market Segments, a single Provider may operate entirely by general advertisement, available to the entire Market. Consider a local distribution electric utility that provides hourly prices for the next day. The providing Party announces day-ahead prices at 9:00 AM each day through 24 indications of interest (IOI) (or a single Stream Indication) to the Segment. Each indication expires at 11:30 AM. Each IOI includes the maximum Power that the Party is willing to sell during each hour of the next day. Each Consumer that wishes to buy on these terms submits RFQs to the issuer for the power they want. The Issuer prepares an actionable quote for each Consumer; each Consumer accepts that Quote by means of a Tender. The Issuer MAY repeat the process, perhaps at different prices, until Transactions for all the power that the Issuer intends to sell are executed.
2) Parties may choose to use an over-the-counter (OTC) trade because they wish to bypass certain market restrictions. A Segment with an Off-market Venue Type would permit, for example, a Party to buy 87 kWh of Power for a period over 1 hour and 5 minutes beginning 15 minutes after the hour. Parties negotiate as above, come to terms, as above, and the Segment records the Transaction.
   Order Book Segments impose restrictions on Round Lots and Intervals to improve Market Liquidity, that is, the likelihood of a match between a Tender to Buy and a Tender to Sell. If OTC Parties already have made an agreement, then there is no need to improve liquidity. This makes the Durations and Round-Lots in Off-market venues indicative rather than prescriptive. 87 kWh is a rough match for Off-Market Segment with a 20 kWh round lot—a gWh is not. In a comparable way, the quote's Duration is a rough match for Segment with an Hour Duration while 3 minutes is not.
3) Markets commonly project opening prices for Instruments before they open. If a system recovery requires a market re-start, there may be no good information to set opening prices. Prior to a re-start, a Segment may advertise RFQs to buy and to sell. The Operator may use a Segment to probe the potential market in this way several times, perhaps with different prices and quantities, to discover an opening price in which the Resource will be in rough balance. When the market has enough information, then the market opens a Segment for trading, announcing the opening prices.

This specification does not require that a Market include any of the scenarios described above. We include them to illustrate how the essential components of Negotiation might fit together in a specific system.

9.2 Negotiation Vocabulary

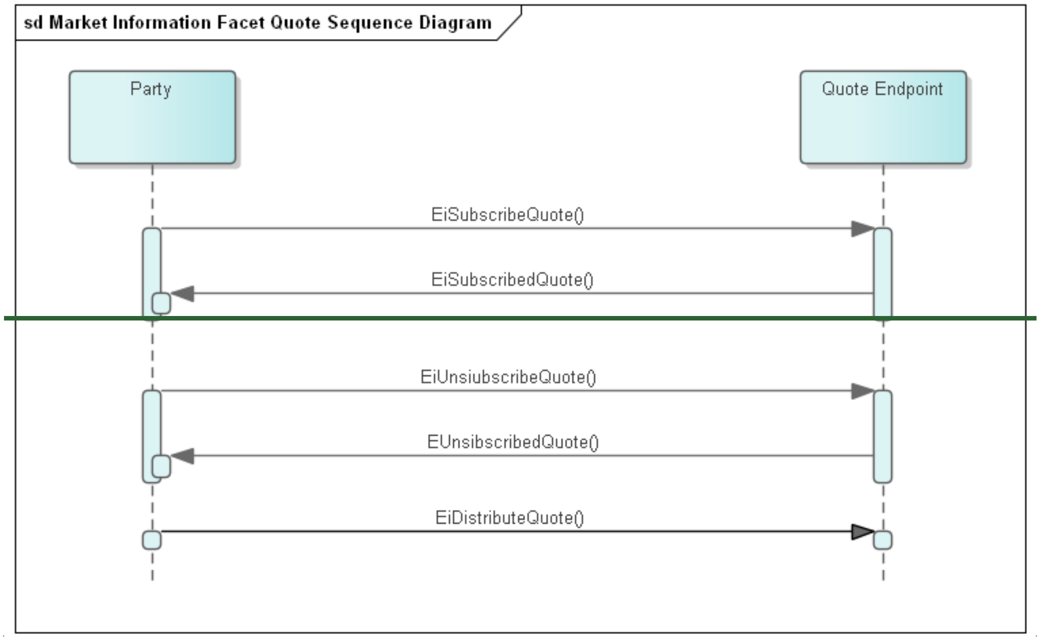
The Messages use advertisement and negotiation are essentially identical to Tenders. Note that the lowercase quote includes both the IOI and the actionable Quote.

Table 1-1: ~~Quote Facet~~Negotiation Message Types

Term	Purpose	Comment	
~~Facet~~Request for Quote (RFQ)	~~Request Payload~~A Party submits a Request for Quote to try to find a market in an Instrument or Instruments. A Request for a Quote may be for a time range of Instruments	~~Response Payload~~May be used pre-opening to elicit tenders, both buy and sell, to determine market opening prices.	~~Notes~~

Term	Purpose	Comment
Quote	~~EiSubscribeQuote~~Indicates the price and quantity at which an instrument can be bought or sold. A quote may be issued in response to an RFQ or to a prior quote, or it may start a negotiation. ~~EiSubscribedQuote~~	~~As multiple Markets may use same Quote service, must tolerate multiple subscriptions.~~The CTS Quote may be either a Bid Quote or an Ask Quote. Any Quote or IOI may be either advertised. *Note CTS does not support two-sided quotes*
Interval Quote ~~EiUnsubscribeQuote~~ ~~EiUnsubscribedQuote~~	~~Unsubscribe for all Markets on this facet.~~A Quote provided for only a Specific Interval.	Some Segments MAY limit negotiations to Intervals only.
Stream Quote	Prices and Quantities for a Product in a series of consecutive Instruments submitted as a single Quote	In energy markets, a stream curve may be referred to as a “Load Curve.”
Tradable Quote	~~EiDistributeQuote~~ An offer to buy or sell up to a specific quantity of an Instrument for a specific price. ~~None~~	~~Post to a Quote endpoint.~~A Tradable Quote is registered by the Segment and can be referenced (“lifted”) to initiate a Trade as if it were a Tender.
Indication of Interest (IOI)	A type of quote that does not create a commitment to accept a Transaction.	As part of a Negotiation, an IOI may encourage further Negotiation. May be issued in response to an RFQ.
Private Quote Private RFQ	A quote sent only to potential Counterparties during a Negotiation	An implementation may use the Segment to distribute Quotes to Counterparties or it may expect Parties to message Counterparties directly.
Advertised Quote Advertised RFQ	Setting the Boolean requestPublication when creating a quote requests that the quote be advertised to all Segment participants.	A Segment advertises quotes on the Indication Ticker.(See 11.3) It is undefined what a private Party (other than the Segment) does after receiving a requestPublication
Issuer	The Issuer is the Party that originates a Quote, whether in response to an RFQ or unsolicited.	The Issuer must accept a matching Tender sent in response to a Tradable Quote.

~~13.2~~9.3 ~~See~~ Messages for the ~~UML sequence diagram for the Quote~~ Negotiation Facet~~:~~

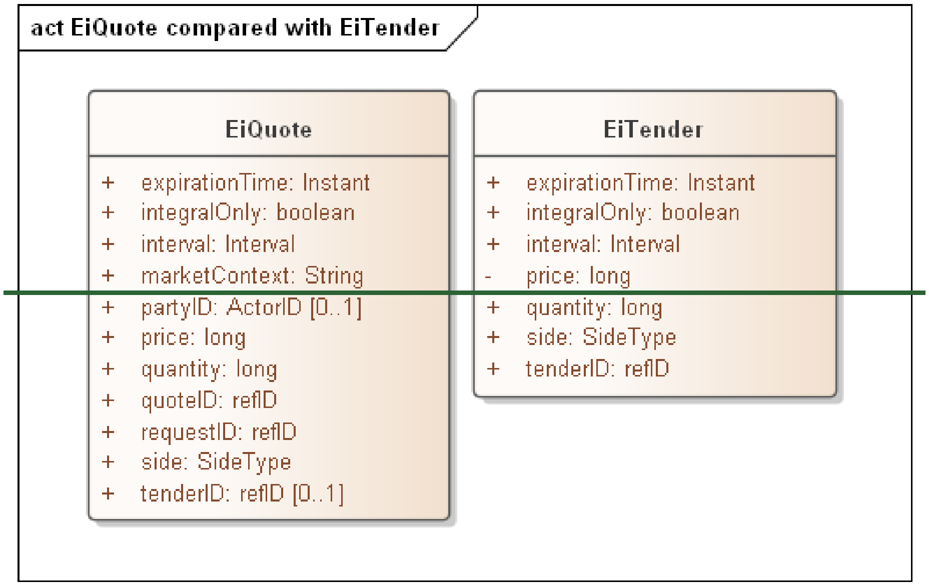


~~Figure -: UML Sequence Diagram for the Quote Facet~~

~~The Quote Service does not wait for or expect acknowledgements of distributed Quotes.~~

~~13.2.1 Information Model for the Quote Facet~~

~~The **[UML]** class diagram describes the information model for EiQuote for the Quote Facet. The diagram includes an informative class diagram of EiTender for comparison.~~



~~Figure -: UML Class Diagram of EiQuote~~

~~The following table details the attributes of the EiQuote class.~~

~~Table -:EiQuote Attributes~~

~~**Attribute**~~	~~**Meaning**~~	~~Notes~~
~~Expiration Time~~	~~The date and time after which this Quote is no longer valid.~~	~~If an advertised Tender, the expiration time for the Tender.~~
~~Integral Only~~	~~All of the Quote must be bought or sold at once; no partial sale or purchase~~	~~Useful for advertisement of Tenders. In CTS Integral Only is conformed to False.~~
~~Interval~~	~~The time interval for the Product being offered~~	~~The Resource Designator is that from the Market.~~
~~Party ID~~	~~Identifies the Party making the Quote~~	~~See Appendix B.2, CTS and Privacy Considerations. Optional.~~
~~Price~~	~~The unit price for the Product being Quoted~~	~~Total price is the product of price and quantity. Note that price is subject to the Price Decimal Fraction value for the Market. See Scale and Granularity Constraints in Section 8, "Market Facet"~~
~~Quantity~~	~~The quantity of the Product being Tendered~~	~~Total price is the product of price and quantity suitably scaled~~
~~Side~~	~~Whether the Quote is to buy or to sell the Product~~	
~~Quote ID~~	~~An ID for this Quote~~	
~~Tender ID~~	~~ID for the Tender being advertised, if any.~~	~~Optional. If present MAY claim that a Tender has been submitted to the Market, in effect advertising that Tender.~~
~~Market Context~~	~~The market context for which this is a quote~~	~~The Quote Reference is a Market Characteristic.~~

~~An Actor may submit quotes for several consecutive Intervals, a set of Instruments for an identical Product. An example is a load serving entity quoting 24 prices for the next day. All elements of the stream share the duration and the stream has the explicitly stated start time.~~[22]

[22] Integration of CtsStreams is pending. TODO

~~13.2.2 Payloads for the Quote Facet~~

~~Conceptually the Quote streams may be considered implemented as Publish-Subscribe streams (Pub-Sub).~~

~~The Market Characteristics provide a Quote Reference for a Market by requesting the QUOTE-REF characteristic.~~

~~The mechanism and setup for subscribing is out of scope, as is the mechanism for publishing. The payloads are as follows:~~

- ~~EiSubscribeQuote – PartyID for the sender and the Quote Reference for the market as found in Market Characteristics~~
- ~~EiSubscribedQuote – EiResponse indicating success or failure. Implementations MAY send information related to the Subscribed stream.~~
- ~~EiUnsubscribeQuote – Removes the subscription to the indicated Quote Reference~~
- ~~EiUnsubscribedQuote – EiResponse indicating success or failure. Implementations MAY send information related to the now un-Subscribed stream.~~
- ~~EiDistributeQuote – Publish an EiQuote to the Quote Reference.~~

~~13.3 Tickers~~

~~Ticker interactions and payloads are nearly identical to those for Quotes. Tickers are anonymized public information about Transactions submitted to the Ticker service by the Markets. The mechanism and interactions of this submission are out of scope.~~

~~13.3.1 Interaction Pattern for the Ticker Facet~~

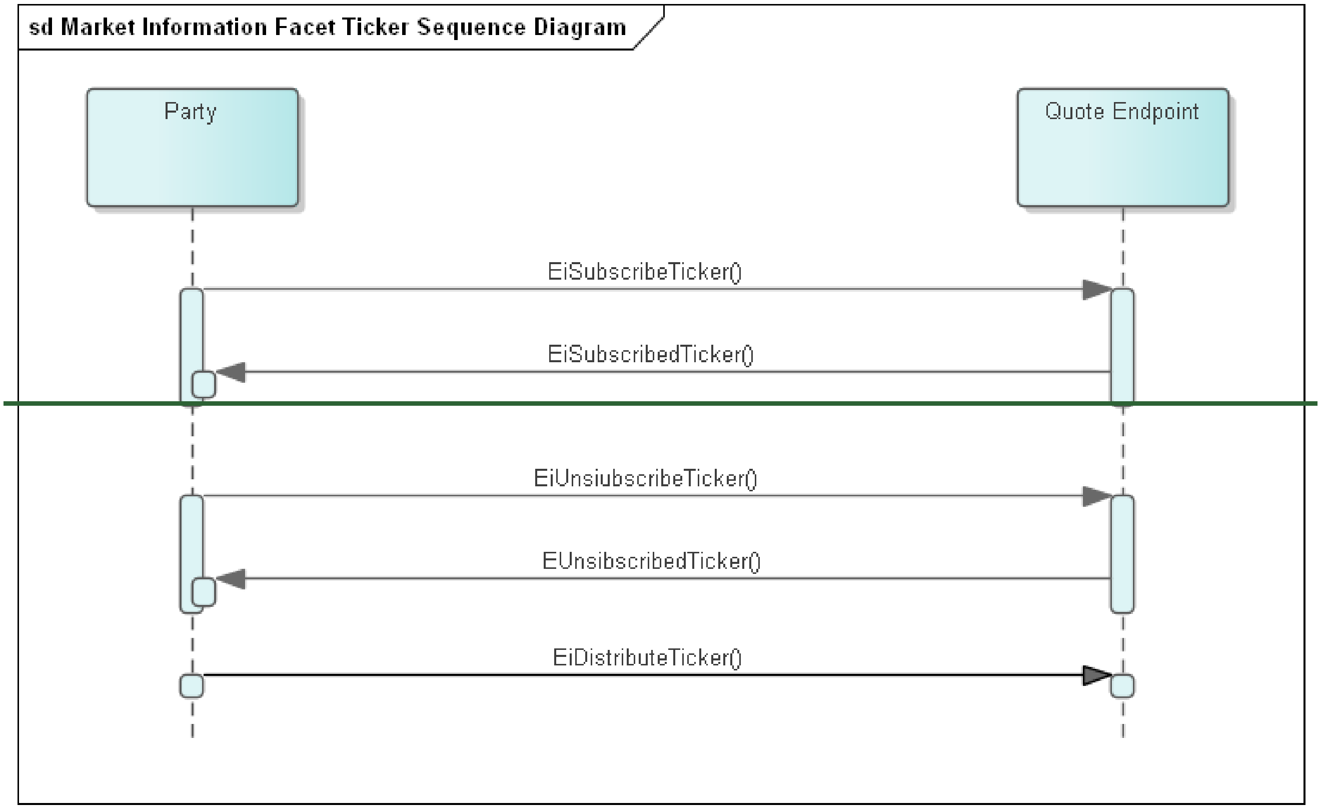
~~Table -: Ticker Facet~~

A Request For Quotes (RFQ) is a message describing what is to be quoted, and may be sent to the Segment or to one or more intended counterparties.

A Quote is either spontaneous or in response to an RFQ or prior IOI. Quotes may be Tradable, in which case a Counterparty may respond with a Tender acknowledged by a Transaction, avoiding the market clearing process.

Facet	Request Payload	Response Payload	Notes
~~Ticker~~Negotiation	~~EiSubscribeTicker~~EiCreateRfq	~~EiSubscribedTicker~~ EiCreatedRfq	~~As multiple Markets may use same Ticker service, must tolerate multiple subscriptions.~~Create and send an RFQ. If the RFQ is to be advertised, the Counterparty is the ID of the Market. Otherwise, it goes to the intended Counterparty. The sender of EiCreateRfq may request publication, but has no guarantee that publication is performed.
~~Ticker~~Negotiation	~~EiUnsubscribeTicker~~EiCreateQuote	~~EiUnsubscribedTicker~~ EiCreatedQuote	~~Unsubscribe for all Markets on this facet.~~Create and send a Quote. If the RFQ is to be advertised, the Counterparty is the ID of the Market. Otherwise, it goes to the intended Counterparty.

Facet	Request Payload	Response Payload	Notes
			The sender of EiCreateQuote may request publication, but has no guarantee that publication is performed.
~~Ticker~~Negotiation	~~EiDistributeTicker~~EiCancelQuote	~~None~~EiCancelledQuote	~~Publish to the Ticker Reference~~Cancel RFQ or Quote or Quote



NOTE TO REVIEWERS: Quotes and RFQs could be merged into a single *Create an Indication of Interest*. Is that an improvement? This Draft goes part way there by including a *RequestPublication* flag in the respectivce Create message payloads. This would be subsumed by the implied IOI convergence, and may lead to elimination of the undefined *EiDistributeIOI* interactions. We request specific comments on this issue.

9.4 Interaction Pattern for the Negotiation Facet

This is the UML sequence diagram for the Negotiation Facet:

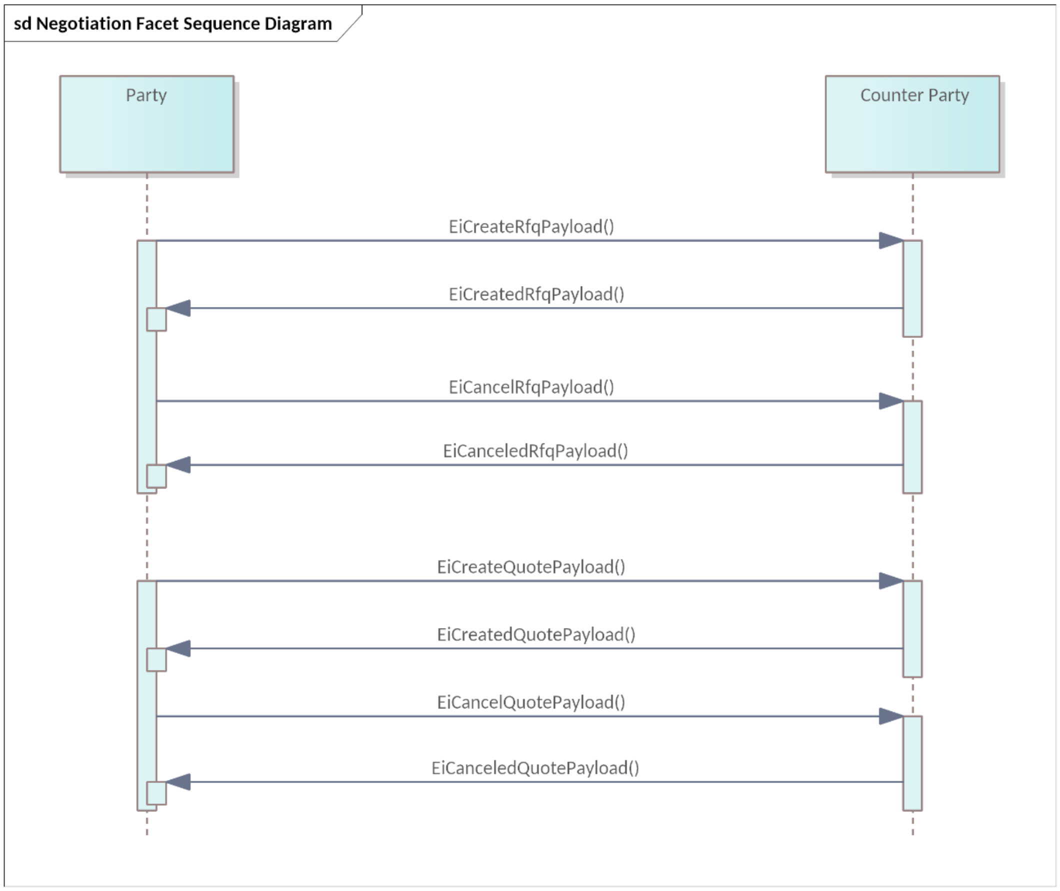
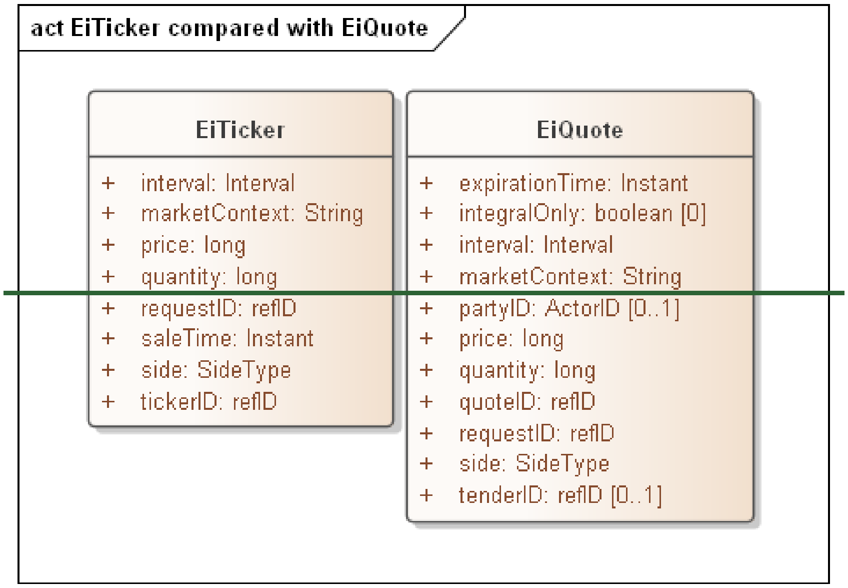


Figure 1-1~~:~~ UML Sequence Diagram for the ~~Ticker~~Negotiation Facet

~~13.4~~9.5 Information Model for ~~the Ticker Facet~~RFQ and Quote

The UML Class Diagram for the EiRfqType are shown in Figure 1-2~~The **[UML]** class diagram describes the information model for EiTicker for the Ticker Facet. The diagram includes an informative class diagram of EiQuote for comparison.~~


act EiTicker compared with EiQuote
EiTicker
+ interval: Interval
+ marketContext: String
+ price: long
+ quantity: long
+ requestID: refID
+ saleTime: Instant
+ side: SideType
+ tickerID: refID
EiQuote
+ expirationTime: Instant
+ integralOnly: boolean [0]
+ interval: Interval
+ marketContext: String
+ partyID: ActorID [0..1]
+ price: long
+ quantity: long
+ quoteID: refID
+ requestID: refID
+ side: SideType
+ tenderID: refID [0..1]

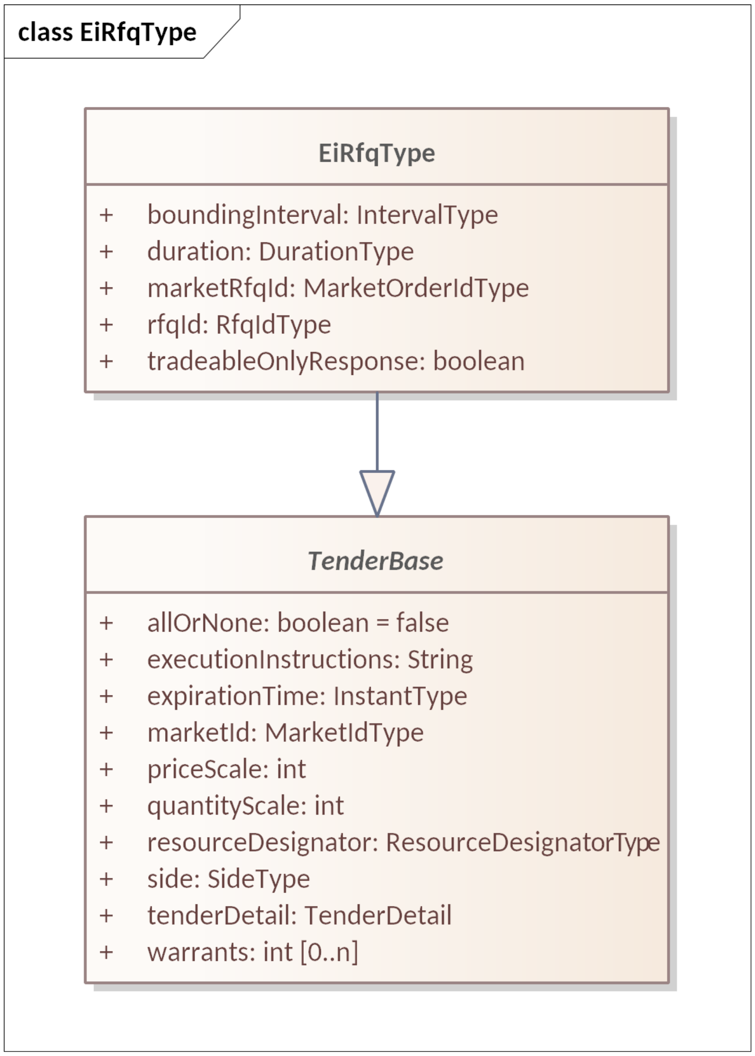
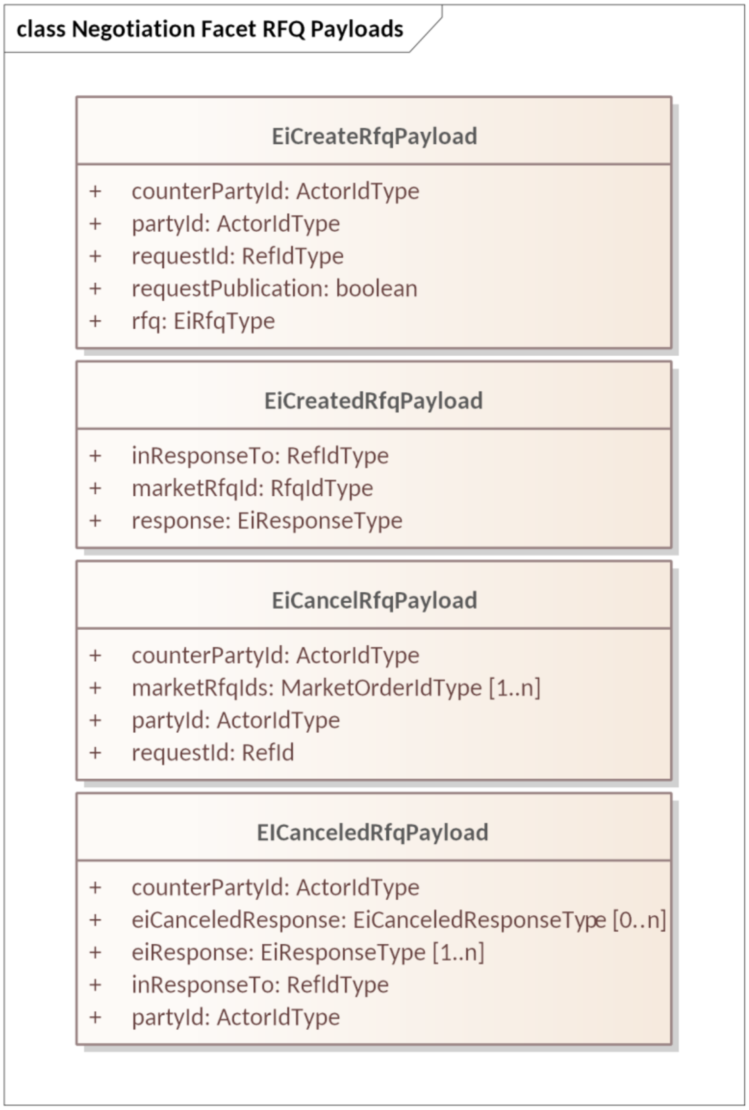

class EiRfqType
EiRfqType
+ boundingInterval: IntervalType
+ duration: DurationType
+ marketRfqId: MarketOrderIdType
+ rfqId: RfqIdType
+ tradeableOnlyResponse: boolean
TenderBase
+ allOrNone: boolean = false
+ executionInstructions: String
+ expirationTime: InstantType
+ marketId: MarketIdType
+ priceScale: int
+ quantityScale: int
+ resourceDesignator: ResourceDesignatorType
+ side: SideType
+ tenderDetail: TenderDetail
+ warrants: int [0..n]

Figure 1-2~~:~~ UML Class Diagram ~~of EiTicker compared with EiQiuote~~for EiRfqType

The UML Class Diagram for the RFQ payloads is shown below. Attributes are in Table 1-2~~The following table details the attributes of the EiTicker class.~~.


class Negotiation Facet RFQ Payloads
EiCreateRfqPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ requestPublication: boolean
+ rfq: EiRfqType
EiCreatedRfqPayload
+ inResponseTo: RefIdType
+ marketRfqId: RfqIdType
+ response: EiResponseType
EiCancelRfqPayload
+ counterPartyId: ActorIdType
+ marketRfqIds: MarketOrderIdType [1..n]
+ partyId: ActorIdType
+ requestId: RefId
EICanceledRfqPayload
+ counterPartyId: ActorIdType
+ eiCanceledResponse: EiCanceledResponseType [0..n]
+ eiResponse: EiResponseType [1..n]
+ inResponseTo: RefIdType
+ partyId: ActorIdType

Figure

~~Table~~ 1-3 UML Class Diagram Showing Negotiation Facet RFQ Payloads

Table 1-2~~-:EiTicker~~: EiCreateRFQ Payload Attributes

Attribute	Type	Fix Field Name	Meaning	Notes
~~Interval~~			~~The time interval for the Product sold~~	~~The Resource Designator is that from the Market.~~
~~Market Context~~			~~The market context for which this is a Ticker~~	~~The Ticker Reference is a Market Characteristic.~~
~~Price~~Counter Party IDs	Actor ID		The ~~unit price~~Party IDs for the ~~Product sold~~CounterParties for which the RFQ is created.	~~Total price is the product of price and quantity. Note that price is subject to the Price Decimal Fraction value for the Market. See Scale and Granularity Constraints in Section , “”~~In CTS, generally the PartyID for the Market. To indicate a bilateral exchange, i.e., a Tender between two specific parties, the PartyID of a specific counter-party is used.
~~Quantity~~			~~The quantity of the sold~~	~~Total price is the product of price and quantity suitably scaled~~
~~Sale Time~~			~~Timestamp indicating when the sale took place.~~	
~~Side~~Party ID	Actor ID		~~Whether~~The Actor ID for the ~~sale was to buy or to sell~~Party requesting the ~~Product~~Quote.	~~An implementation MAY deliver only Buy side Ticker elements~~Indicates which Actor proposes the buy or sell side EiCreateTender.
~~Ticker~~Request ID	RefIDType		~~An ID for~~Reference to this ~~Ticker~~message payload	
RFQ	EiRfqType		The RFQ transmitted.	Unlike EiCreateTender, only one RFQ is created per message payload.
Request Publication	Boolean			The sender of EiCreateRfq may request publication by setting Request Publication to true, but has no guarantee that publication is performed.
The fields below are as defined in EiRfqType				
RFQ Id	QuoteIdType		Assigned by the sender of EiCreateRfqPayload	Creator’s ID for the RFQ

Bounding Interval	Interval Type		A closed interval which encloses all Quotes requested.	See Sections 78 (Position and Delivery Facets). Alignment of instruments is a characteristic of the Segment.
Duration	Duration Type		Resource and Duration determine product.	An explicit duration rather than a database lookup simplifies consumption and generation planning.
All Other fields are as defined in the Tender, Table 1-2				

Table 1-3 EiCreatedRfq Payload Attributes

Attribute	**Type**	**Fix Field Name**	**Meaning**	**Notes**
Market Request for Quote ID	RfqIdType	RFQId	ID for this Request for Quote	Market Assigned as it in for Tenders. Used for Cancel request.
Response	EiResponse Type		Standard response object	
In Response To	RefIdType		ID for the corresponding EiCreateRfq Payload	

Table 1-4 EiCancelRfq and EiCanceledRfq Payload Attributes

Attribute	**Type**	**FIX Field Name**	**Meaning**	**Notes**
Counter Party ID	Actor ID		The Actor ID for the CounterParty for which the Tender is created.	In CTS, generally the PartyID for the Market. To indicate a bilateral exchange, i.e., a Tender between two specific parties, the PartyID of a specific counterparty is used.
Party ID	Actor ID		The Actor ID for the Party on whose behalf this Tender is made.	Indicates which Actor proposes the buy or sell side EiCreateTender.
Request ID	Ref ID		An identifier for this Cancel RFQ Payload	
Market Request for Quote ID	RfqIdType	OrderID (37)	Market-assigned ID for the subject Request for Quote	Market Assigned in parallel with Tenders.
Quote Id	Quote ID Type		The Quote ID included in the CreateQuoteTender Payload or the Tenders within an CreateQuotePayload	

Attribute	**Type**	**FIX Field Name**	**Meaning**	**Notes**
Canceled Response	EiCanceled Response Type		Optional Detailed response for each RFQ for which cancelation was requested	
In Response To	RefIdType		The EiCancelRfqPayload that is responded to in the Canceled Payload	

9.6 Information Model for Negotiation Facet

As described in Section 5.3 Information Model for the Tender Facet, EiQuote and EiTender are subclasses of abstract class TenderBase. In the following table, only the first three attributes are part of EiQuoteType; the rest are inherited.

See Table 1-3: EiTender *Attributes*” for details. See Figure 1-2 for the inheritance and relationship between EiTenderType and EiQuoteType.

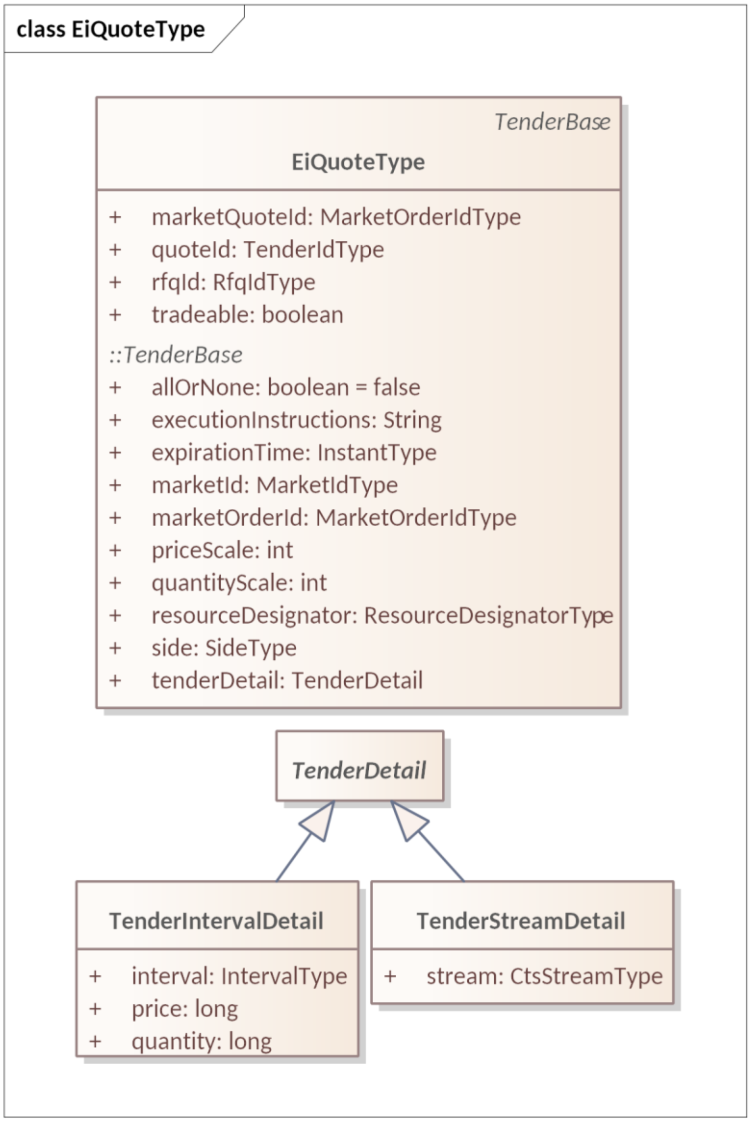


Figure 1-4 UML Diagram of EiQuoteType showing inherited attributes.

Table 1-5: Attributes of EiQuoteType

Attribute	**Type**	**FIX Field Name**	**Meaning**	**Notes**
Market Quote ID	Market Order Type		ID assigned by the Segment or Market	Used in acknowledgment of tradable quote in future market messages.
Private Quote	Bool	PrivateQuote (1171)	Specifies Quote is available to a specified counterparty or only.	
Quote ID	Quote Id Type	QuoteID (117)	ID as submitted by Quote originator	Used in off-market negotiation Also used in Tender to reference a Tradable Quote

Attribute	Type	FIX Field Name	Meaning	Notes
RFQ ID	RFQ Id Type	QuoteReqID (131)	ID as submitted by RFQ originator	Referenced by Quote responding to RFQ
Quote Type	Boolean	QuoteType (537)	IOI or Tradable	Tradable is a Boolean; if true, the quote is tradable. If false the quote is not tradable, or an Indication of Interest (IOI) consistent with FIX terminology.
All Other fields are as defined in the Tender, Table 1-2				

9.7 Negotiation Facet Payloads

Figure 1-5 shows UML Class Diagrams for the Negotiation Facet Payloads.

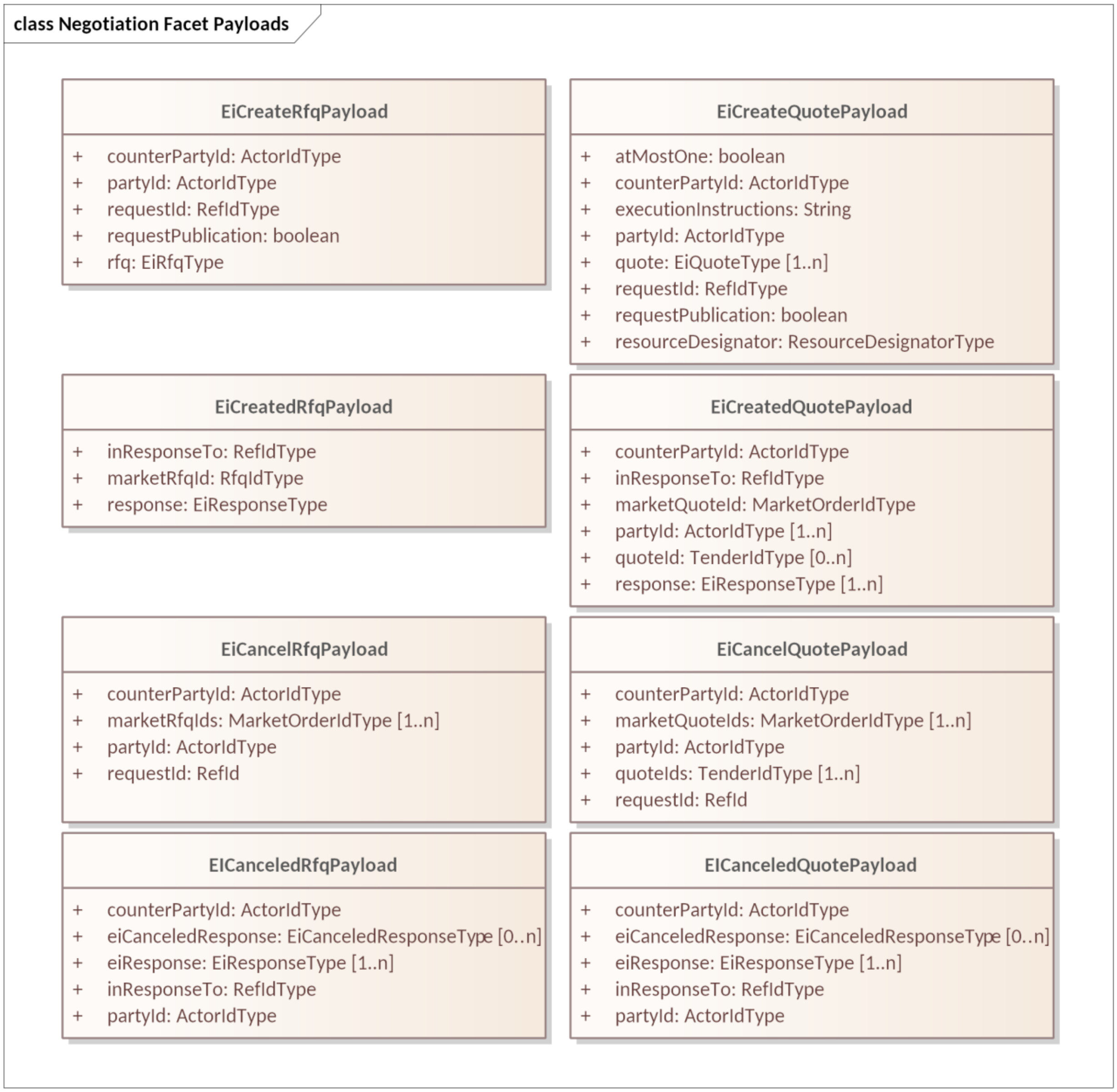

class Negotiation Facet Payloads
EiCreateRfqPayload
+ counterPartyId: ActorIdType
+ partyId: ActorIdType
+ requestId: RefIdType
+ requestPublication: boolean
+ rfq: EiRfqType
EiCreateQuotePayload
+ atMostOne: boolean
+ counterPartyId: ActorIdType
+ executionInstructions: String
+ partyId: ActorIdType
+ quote: EiQuoteType [1..n]
+ requestId: RefIdType
+ requestPublication: boolean
+ resourceDesignator: ResourceDesignatorType
EiCreatedRfqPayload
+ inResponseTo: RefIdType
+ marketRfqId: RfqIdType
+ response: EiResponseType
EiCreatedQuotePayload
+ counterPartyId: ActorIdType
+ inResponseTo: RefIdType
+ marketQuoteId: MarketOrderIdType
+ partyId: ActorIdType [1..n]
+ quoteId: TenderIdType [0..n]
+ response: EiResponseType [1..n]
EiCancelRfqPayload
+ counterPartyId: ActorIdType
+ marketRfqIds: MarketOrderIdType [1..n]
+ partyId: ActorIdType
+ requestId: RefId
EiCancelQuotePayload
+ counterPartyId: ActorIdType
+ marketQuoteIds: MarketOrderIdType [1..n]
+ partyId: ActorIdType
+ quoteIds: TenderIdType [1..n]
+ requestId: RefId
EICanceledRfqPayload
+ counterPartyId: ActorIdType
+ eiCanceledResponse: EiCanceledResponseType [0..n]
+ eiResponse: EiResponseType [1..n]
+ inResponseTo: RefIdType
+ partyId: ActorIdType
EICanceledQuotePayload
+ counterPartyId: ActorIdType
+ eiCanceledResponse: EiCanceledResponseType [0..n]
+ eiResponse: EiResponseType [1..n]
+ inResponseTo: RefIdType
+ partyId: ActorIdType

Figure 1-5 UML Class Diagram for Negotiation Facet Payloads

The following tables show attributes for the Quote Payloads.

Table 1-6 EiCreateQuotePayload

Attribute	Type	FIX Field Name	Meaning	Notes
Counter Party ID	Actor ID		The Actor ID for the CounterParty for which the Quote is created.	In CTS, generally the PartyID for the Market. To indicate a bilateral exchange, i.e., a Tender between two specific parties, the PartyID of a specific counter-party is used.
EiQuote			As above	
Party ID	Actor ID		The Actor ID for the Party requesting the Quote.	Indicates which Actor proposes the buy or sell side
Tradable	Boolean	(part of Codeset in FIX)	If false, this is an indicative quote.	Contains Boolean for whether the quote is Tradable.
Request Publication	Boolean			The sender of EiCreateQuote may request publication by setting Request Publication to true, but has no guarantee that publication is performed.

9.8 Information Model for Stream Quotes

Some Market Segments may permit Stream Quotes, that is, a single Quote for multiple consecutive Instruments. When responding to a Request for Quote, the range of quoted Intervals must be within the bounding Interval of the Request.

Stream quotes and quotes are the same class.

9.9 Rejecting a Quote

Quotes may be rejected if ill formed, violate granularity rules, or other requirements of the Segment or Market.

9.10 Creating Transactions from Quotes

A Party receiving a Tradable Quote MAY respond by submitting a Tender that references that Quote.

The Market registers a Tradable Quote it receives AS IF it were a Tender, that is, places them in a book until it expires. A Quote that is marked as Advertised by using an EiDistributeQuotePayload is included in the Indications Ticker (see Section 11.3~~An Actor~~).

The Market does not Advertise a Quote that is directed to a specific Party or Parties, though it MAY follow its anonymization and advertising practices.

To accept a Tradable Quote, whether on first notice or after negotiation, a Party submits a Tender matching the Price and Quantity of the Quote and referencing the QuoteID. The Market will then validate the match and create a Transaction if it fits. The Tender must match price, be within the acceptable quantity of the Quote, and so on. An All-or-None Execution Instruction on either Quote or Tender MUST be honored. If the Quote accepts partial fulfillment, the remaining balance on the Quote is decremented as it is for a Tender.

Notwithstanding any negotiation, the Market may reject the Tender if accepting it would interfere with resource operations or violate financial requirements on participants.

If a Tradable Quote is open when the Instrument closes, it is the responsibility of the Party that submitted the Quote to cancel it. If the Negotiator still wishes to accept an instrument scheduled for 11:00 at 11:30, that is up to the Parties; how that is accomplished is out of scope.

9.11 Negotiation Messages

The negotiation messages are variations on a common theme, but are distinguished by message type.

The Quote Service does not wait for or expect acknowledgements of advertised Quotes.

If permitted in the venue (Segment), a Party may submit ~~ticker instances~~quotes for several consecutive Intervals, a set of Instruments for an identical Product~~. An example is a load serving entity quoting 24 prices for the next day. ~~, as a single Quote. In Power, this is called a “Load Curve”. Stream Quotes are constructed and interpreted in the same manner as are Stream Tenders.

All elements of the stream share the duration and the stream has the explicitly stated start time.

9.12 Discussion of Negotiation (Non-Normative)

Any Market Segment may support Negotiation as indicated in the Market Structure information.

A Quote-Driven Market is one of the venue types (see Table 13-1: Venue Types in CTS) for a CTS Market Segment. In financial markets, a quote-driven market is a secondary market trading structure wherein buyers and sellers interact with dealers. It is not as transparent as an order-driven market in that the market orders and prices that traders are willing to buy or sell at are not available to the counterparty.

A Quote-Driven Market MAY be chosen to negotiate with a dominant supplier that represents many third parties, i.e., a Distribution System Operator acting as an intermediary to a bulk power market. Such a Quote-Driven Market can permit large buyers to plan significant resource use over time, for example, scheduling a long running industrial process which also requires labor planning. Such a buyer could submit multiple Requests for Quotes with different schedules, and then select from among the Quotes received in response.

The overarching Market tracks all Transactions and monitors all Delivery.

10 Subscriptions

A Party wishing to trade in a market naturally wants to be kept apprised of changing information about the market. This can be roughly divided into granular information about what other Parties are doing in the Market, and information about the Market as a whole. The FIX Protocol specification terms these as Market Data, that is, granular or aggregate information about activities in a Market, and Market Structure Reports, that is, information about how each Market Segment is operating.

In the FIX Protocol, a Party gets this information by means of Subscriptions. A Party subscribes to the information it needs and thereafter receives periodic updates relating to that subscription. The FIX interaction model defines a *subscription* as how an Actor requests one or more market reports.

A Market consists of multiple Market Segments, each trading a single Product based on the Resource traded in that market. Multiple Market Segments in a Market MAY trade the same Product, perhaps with different trading rules, or different schedules of operation. The Market Segments in a Market may support different Market Data Reports. Information about a Market and its Market Segments is conveyed in the Market Structure Subscription.

Subscriptions are how a Party requests specific Pre-Trade information. Not all Markets and Segments will support all Subscription types. The Subscriptions supported by each Segment are described in Section 13, *"Market Structure Subscriptions".*

Some markets will not support granular subscriptions. The Market Structure will instead indicate a multi-cast point or other source that a Party can choose to listen to.

10.1 Subscriptions

Several varieties of market and trade-related information are described in this specification. To request and acknowledge subscription changes objects of the EiSubscriptionRequest and Response Types are included in the appropriate subscription management messages.

10.1.1 Information Model for Subscription Requests and Responses

The UML Class Diagram for the Subscription Request and Response is shown in Figure 10-1. Specific requests, for example for tickers or market information, are defined in Section 11.

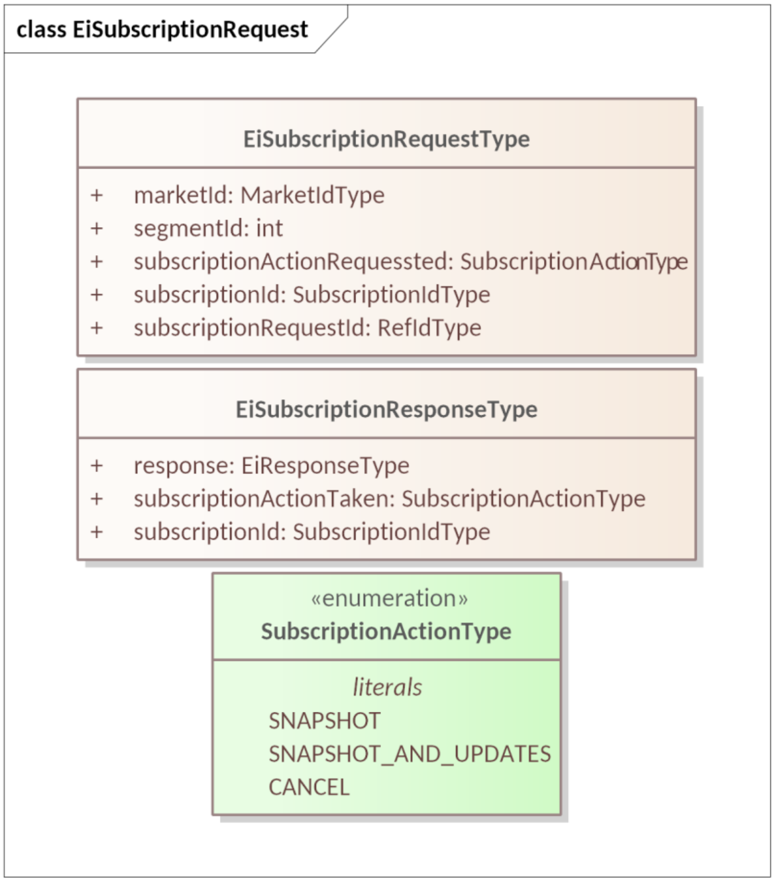

class EiSubscriptionRequest
EiSubscriptionRequestType
+ marketId: MarketIdType
+ segmentId: int
+ subscriptionActionRequessted: SubscriptionActionType
+ subscriptionId: SubscriptionIdType
+ subscriptionRequestId: RefIdType
EiSubscriptionResponseType
+ response: EiResponseType
+ subscriptionActionTaken: SubscriptionActionType
+ subscriptionId: SubscriptionIdType
«enumeration»
SubscriptionActionType
literals
SNAPSHOT
SNAPSHOT_AND_UPDATES
CANCEL

Figure 10-1 UML Class Diagram for Subscription Request and Response Types

Attributes for EiSubscriptionRequest are shown in Table 10-1.

Table 10-1 EiSubscriptionRequest Attributes

Attribute	Attribute Type	FIX Field Name	Meaning
Market Id	Market ID Type	MarketID (1301)	A UID referencing a Market
Segment ID	Natural Number	MarketSegmentID (1300)	If Segment ID is non-zero, the request is limited to reporting on the indicated single Segment. If zero, the subscription requests reporting on all Segments of the indicated Market
Subscription Action Requested	Enumeration	SubscriptionRequestType (263)	The type of subscription being requested. CTS uses an enumeration carrying the same information as the FIX numeric codes: 0 – Snapshot 1 – Snapshot + Updates (update frequency is determined by the supplier) 2 – End the indicated Subscription

Attribute	Attribute Type	FIX Field Name	Meaning
Subscription ID	Subscription ID Type		A UID indicating the created subscription, so that it may be canceled or modified. Should be null for initial request.
Subscription Request ID	Id Type	MDReqID (262)	Used to identify this request for managing a subscription. See ALSO FIX MarketDataRequest (35=DR).

Attributes for EiSubscriptionResponse are shown in Table 10-2.

Table 10-2 EiSubscriptionResponse Attributes

Attribute	Attribute Type	FIX Field Name	Meaning
Response	EiResponse Type		
Subscription Action Taken	Enumeration	SubscriptionRequestType (263)	The action taken on the referenced Subscription ID.
Subscription ID	Subscription ID Type		A UID indicating the newly created, modified, or canceled subscription.

11 Ticker Facet

A Party wishing to trade in a market naturally wants to be kept apprised of changing information about the market. FIX divides this information into two categories: Market Data and Market Structure. Market Data reports activity in the Market, Market Structure describes how the Market (or Segment in CTS) is organized.

This section describes mechanisms to access continuous Market Data on the activities of market participants. CTS calls these *Tickers*. Tickers update continuously, on a schedule determined by the provider, as Parties interact with a Segment.

There are four types of Tickers, represented as an enumeration. See Table 11-1 and Figure 11-1 below.

Table 11-1: Types of Tickers in CTS Facet

Ticker Type	Request Payload
Bids	Anonymized Tenders to Buy
Indications	Advertised Indications of Interest (IOI) and RFQs
Offers	Anonymized Tenders to Sell
Transactions	Anonymized Completed Transactions, whether from market matches or from Negotiation

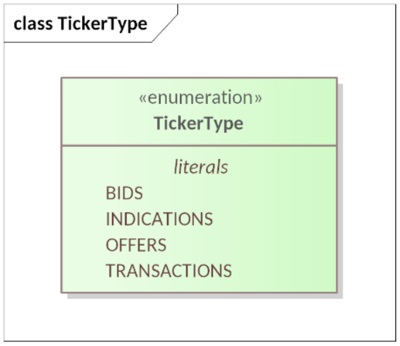


Figure 11-1 TickerType Enumeration

Not all Markets or Market Segments support Ticker subscriptions or all Ticker types. Actors can discover what Tickers a Segment supports and how to interact with them through the Market Structure reports. Market Structure is discussed in Section 13, *"Market Structure Subscription"*.

Private Quotes do not appear in Tickers.

In many markets, it is required for most participants to be anonymized, that is, the identity of the submitter kept private. In such markets, the Market Id is used as the Party ID in the Ticker. In Resource markets as in financial markets, Partys in specific and influential roles are not anonymized. For example, a Market may opt not to anonymize the Party Id of the distribution system operator (DSO). CTS makes no statement about what anonymization rules a resource market must use. This specification offers the general guidance that most participants be anonymized to preserve privacy, but that Ticker messages for significant participants can be submitted under their own identity.

11.1 Ticker Facet Subscriptions

A Party subscribes to a Ticker common FIX market subscription model (Section 10.1, *"Subscriptions"*). A Party can subscribe to a single Market Segment or all Market Segments. Each Ticker Type, if available, requires a separate Subscription.

Table 21-2: Ticker Facet

Facet	Request Payload	Response Payload	Notes
Ticker	EiManage Ticker Subscription. Payload	EiManaged Ticker Subscription. Payload	As multiple Markets may use same Ticker service, must allow multiple subscriptions.
~~Ticker~~	~~EiDistributeTicker~~	~~None~~	~~Publish Indication to the Ticker~~

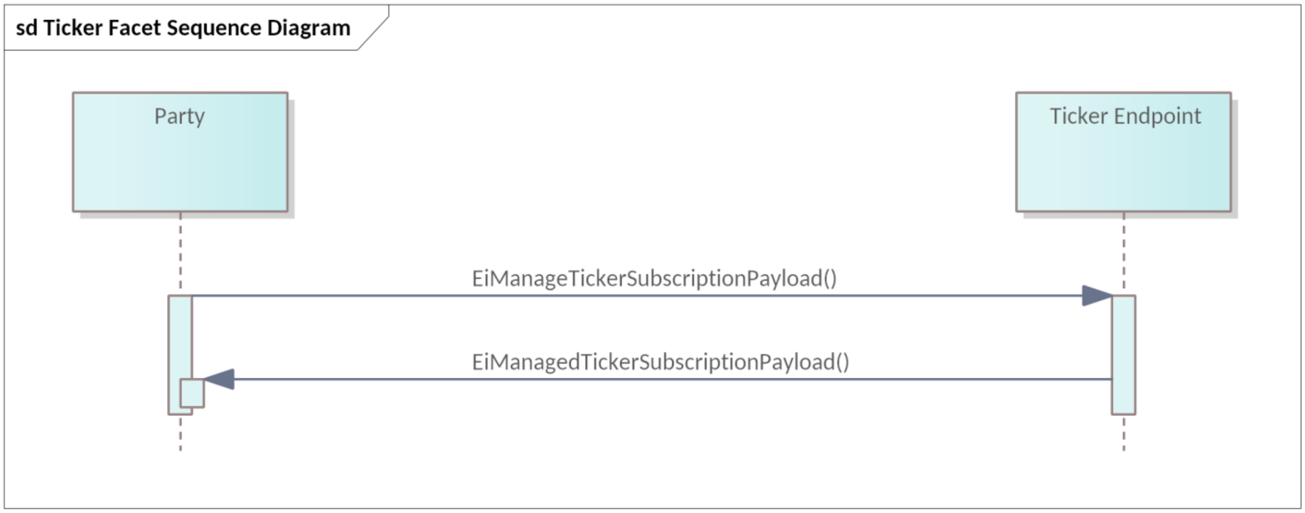
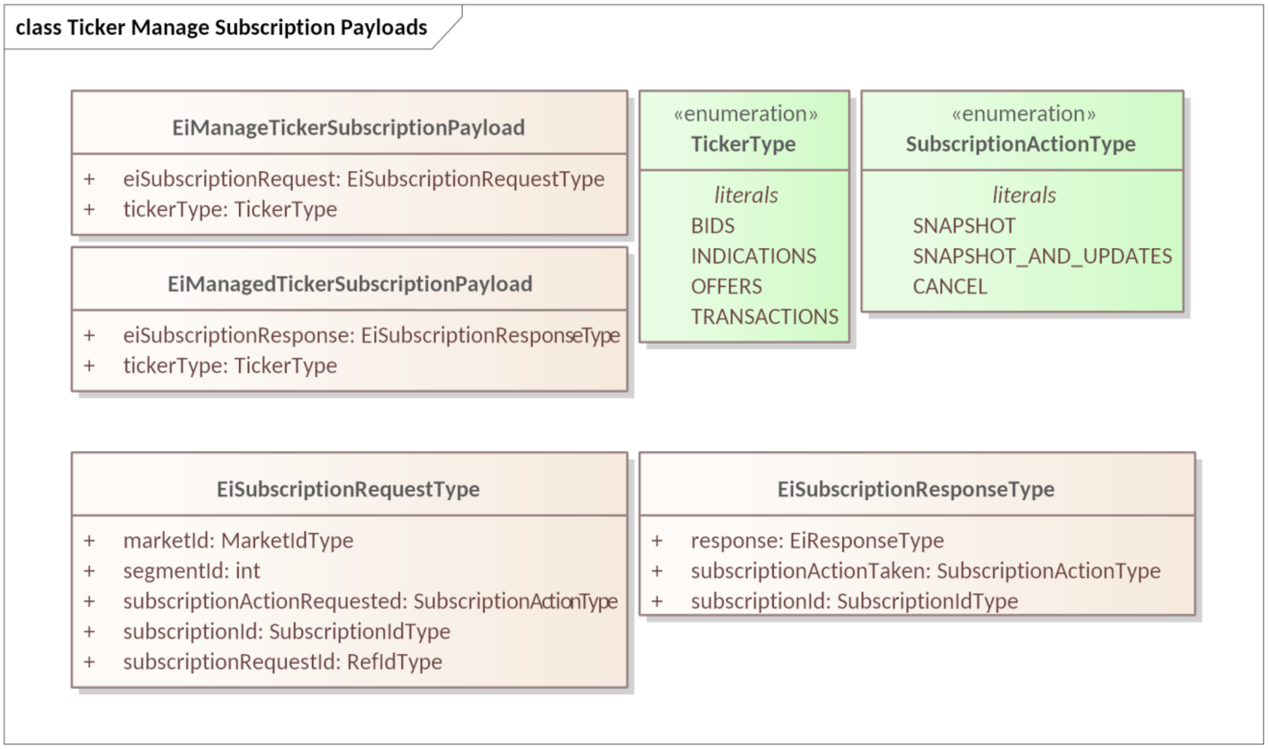

sd Ticker Facet Sequence Diagram
Party
Ticker Endpoint
EiManageTickerSubscriptionPayload()
EiManagedTickerSubscriptionPayload()

Figure 11-2: UML Sequence Diagram for the Ticker Facet

11.1.1 Information Model for Ticker Facet Subscriptions

The messages for adding, changing, or deleting a Ticker subscription contain only the ticker type and a subscription request or response. The UML Class Diagrams for the payloads are shown in Figure 11-3.


class Ticker Manage Subscription Payloads
EiManageTickerSubscriptionPayload
+ eiSubscriptionRequest: EiSubscriptionRequestType
+ tickerType: TickerType
EiManagedTickerSubscriptionPayload
+ eiSubscriptionResponse: EiSubscriptionResponseType
+ tickerType: TickerType
«enumeration»
TickerType
literals
BIDS
INDICATIONS
OFFERS
TRANSACTIONS
«enumeration»
SubscriptionActionType
literals
SNAPSHOT
SNAPSHOT_AND_UPDATES
CANCEL
EiSubscriptionRequestType
+ marketId: MarketIdType
+ segmentId: int
+ subscriptionActionRequested: SubscriptionActionType
+ subscriptionId: SubscriptionIdType
+ subscriptionRequestId: RefIdType
EiSubscriptionResponseType
+ response: EiResponseType
+ subscriptionActionTaken: SubscriptionActionType
+ subscriptionId: SubscriptionIdType

~~13.4.1 ~~Figure 11-3~~Payloads for the Ticker Facet~~

~~The **[UML]** class diagram describes the payloads for the Delivery facet.~~

~~Conceptually the Ticker streams may be considered implemented as Publish-Subscribe streams (Pub-Sub).~~

~~The Market Characteristics provide a Ticker Reference for a Market by requesting the TICKER-REF characteristic.~~

~~The~~ Ticker Manage Subscription Payloads

11.1.2 Exceptions to Subscription Interactions

A given Segment may support the same Ticker for any or all of the Ticker types. In that case, an Actor that subscribes one of these Tickers subscribes to all the Types included in that Ticker.

In larger markets, there may be a common broadcast channel for a Ticker. In such markets, there is no subscription; the Actor simply listed to that broadcast channel.

11.2 Ticker Patterns

The various types of tickers share a common approach:

- A subscription is created using EiManageTickerSubscription, passing the requested change and which ticker is being managed.
- The ticker payloads contain the subscription ID and the relevant object for the ticker type:
  - TenderTickerType is EiTenderType for Bid and Offer tickers.
  - TransactionTickerType is EiTransactionType
  - IndicationTickerType is either an EiQuoteType (not tradable) or EiRfqType.
- The ticker delivery is defined by the provider.

- ○ Large or complex markets might use a multicast for delivery, with the relevant ticker payloads.
- ○ Small or less complex markets might use a market-defined delivery mechanism (out of scope)

The UML Class Diagram for ticker types is shown in Figure 11-4.

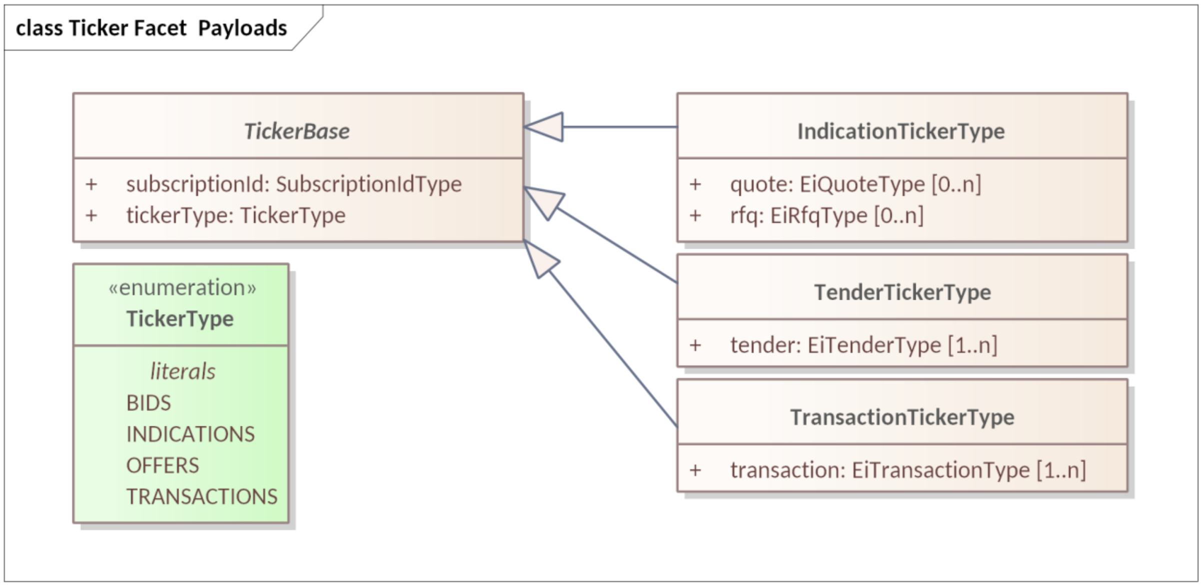


Figure 11-4 Ticker Facet Payloads

The attributes for the Ticker Types are shown in Table 11-2.

Table 11-2 Attributes for the Ticker Type

Attribute	Attribute Type	FIX Field Name	Meaning
Subscription ID	Subscription ID Type		A UID indicating the related subscription.
Ticker Type	TickerType		Enumeration consisting of literals BIDS, INDICATIONS, OFFERS, or TRANSACTIONS.
Tender	EiTenderType		For TenderTickerType
Transaction	EiTransactionType		For TransactionTickerType
Quote	EiQuoteType		Alternate for IndicationTickerType; either all Quotes or all RFQs.
RFQ	EiRfqType		Alternate for IndicationTickerType; either all Quotes or all RFQs.

11.3 The Bids and Offers Tickers

Bids and Offers are simply Buy or Sell side Tenders. When a Tender is submitted, the Segment announces the Tender on the Ticker. A Segment may use a single Ticker for both Bids and Offers as described in its Market Structure reports. Tenders are submitted to the entire market; There is no guarantee that an Offer or a Bid will still be there when a Party submits a matching Tender. [23]

Tradeable Quotes MAY be published in a ticker only. if it is delivered by means of an EiDistributeQuote payload. See Section 9 Negotiation Facet.

The payload for Bids and Offers Tickers includes one or more EiTenderType objects with attributes anonymized following market or segment rules. Attributes are shown in Table 1-3: EiTender Attributes~~setup for~~.

A Party that wishes to receive Bids or Offers from a Segment must subscribe to the Bid or to the Offer Ticker for that Segment. If a single Ticker only is provided Bids and Offers, then subscribing to either one subscribes to both.

11.4 The Indications Ticker

If a Segment supports Negotiations, then it supports an Indication Ticker. While the messages in Bids and Offers are essentially identical, except for BUY or SELL side, there is more diversity in the messages for Indications.

The payload of the Indications Ticker is just as defined for the IOI in Negotiations. Because the purpose of a public Offer is to initiate a Negotiation between Partys, the Indications Ticker is not anonymized.

The objects in the Indication Ticker payload are either all Quotes or all RFQs.

11.5 The Transactions Ticker

The Transactions Ticker is the continuous publication of information about Transactions executed in a Market Segment. Both Parties are listed on a Transaction, although either or both may be anonymized as specified in market rules.

In Negotiation-based Markets, the Transaction announcement is the first public appearance of the Negotiation.

Figure 11-4 Ticker Facet Payloads includes the payload for the TransactionTicker—an EiTransactionType object with the ticker type and subscription ID.

Table 11-3: EiCompletedTransaction Attributes

Attribute	Type	FIX Field Name	Meaning	Notes
BuyerId	Uid		ID of the Buyer in this Transaction	May be ExchangeID if anonymized
SellerId	Uid		ID of the Seller in this Transaction	May be ExchangeID if anonymized
The remaining fields are as defined in the Tender, Table 1-2				

[23] One edge case is that of a Tradable Quote—in some sense a tradable quote might be considered a Tender, and in another sense considered an Indication. CTS adopts the second approach, but a

12 Instrument Market Data

Instrument Summaries are Subscriptions (described in Section 10) that provide market data about the specific Instruments traded in the Segment. Like other Subscriptions, Instrument Summary Subscriptions provide part of what FIX terms Pre-Trade Data.

12.1 Instrument Summary Subscription

An Instrument Summary Subscription requests periodic data on a bounded range of Instruments. Within a Market Segment, trading is for a single Product, and Instruments are distinguished by Start Time and Date. The Subscription returns market data for all Instruments whose Start Time falls within the Bounding Interval of the Subscription.

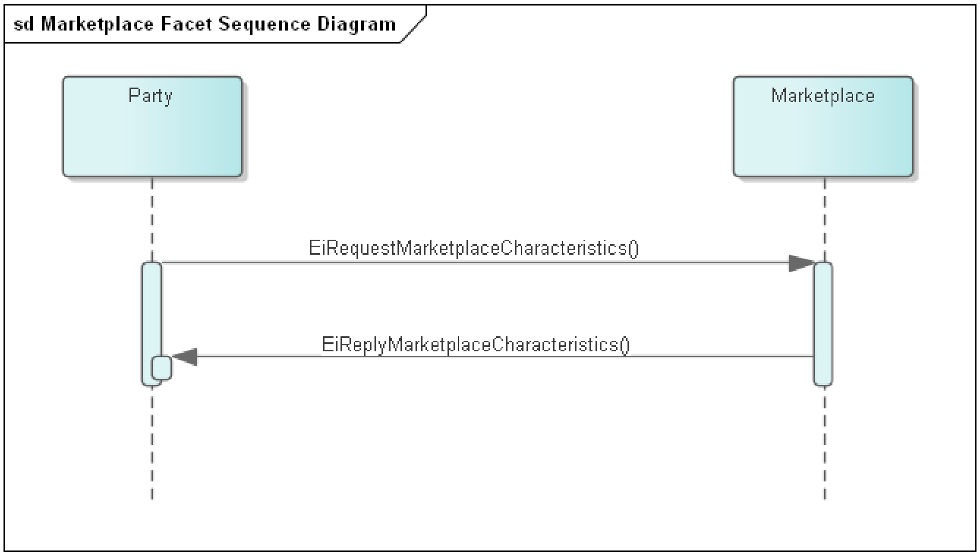

sd Marketplace Facet Sequence Diagram
Party
Marketplace
EiRequestMarketplaceCharacteristics()
EiReplyMarketplaceCharacteristics()

Figure 12-1 : Need new UML for the Instrument Summary Subscription Request

The Instrument Summary Subscription Request specifies the type of summary and the instruments requested.

Table 12-1: Information Model for Instrument Summary Subscriptions

Attribute	Attribute Type	FIX Field Name	Meaning	Notes
Market Data Request ID	String	MDReqId (262)	ID of this Market Data Request	Use the ID of previous Market Data Request to modify or cancel a previous request
Request Type	Char	Subscription Request Type (263)	Indicates to the other party what type of response is expected. .	0 = A snapshot request only asks for current information. 1 = A snapshot and updates. 2 = Unsubscribe, will cancel any future update messages from the counter party
Update Type	Char	MDUpdateType (265)	0 = Full Refresh 1 = Incremental Refresh	

Attribute	Attribute Type	FIX Field Name	Meaning	Notes
Market Segment ID	String	MarketSegmentId (1300)	Unique Identifier for Segment	Market Data Reports are for a single Segment
Bounding Interval	Start Date End Date	Bounding Interval	Bounding Interval?	Subscription covers all Instruments within Bounding Interval
Instrument Summary Type	Int		Type of Instrument Summary to subscribe to: 0 = Session Summary 1 = Top-Of-Book	
Market Depth	Int	MktDepth (264)	Depth of market for Book Snapshot / Incremental updates 0 = full book depth 1 = top of book 2 or greater = book depth (number of levels)	

12.2 The Instrument Summary Report

Instrument Summary Reports provide summary information about one or more instruments. Common information about the report is presented in a Market Data Instrument Header. Information about each instrument is presented in the Market Data Instrument Summary.

12.2.1 The Instrument Summary Header

Table 12-2: Information Model for the Market Instrument Summary Report

Attribute	Attribute Type	FIX Field Name	Meaning	Notes
Market Data Request ID	String	MDReqId (262)	ID of this Market Data Request	Use the ID of previous Market Data Request to modify or cancel a previous request
Request Type	Char	Subscription Request Type (263)	Indicates to the other party what type of response is expected.	0 = A snapshot request only asks for current information. 1 = A snapshot and updates. 2 = Unsubscribe, will cancel any future update messages from the counter party
Market Segment ID	String	Segment (1300)	Unique Identifier for Segment	

Attribute	Attribute Type	FIX Field Name	Meaning	Notes
Market Segment Status	Int	MarketSeg Stat (2542)	1 = Active: Market segment is active, i.e. trading is possible. 2 = Inactive: Market segment has previously been active and is now inactive. 3 = Published: Market segment information is provided prior to its first activation.	
Instrument Summary		Series	Repeating series for each Instrument in the Report. The information varies by the Summary Type requested.	

12.2.2 The Instrument Summary

The Instrument Summary is the information in an Instrument Summary Report that is repeated for each Instrument in the range.

The Instrument Summary are a repeating group of summaries, one for each instrument in the Market Data Instrument Summary Subscription.

The information conveyed varies with the Instrument Subscription Type.

12.3 The Instrument Summary Types

12.3.1 The Instrument Session Summary

A common change reported in an Instrument Session Summary is market crossing, announcing when a Segment opens and when a Segment closes. In transactive resources, each Instrument crosses as well A Segment may not permit trading in an Instrument more than forty-eight hours in the future. An Instrument in the past can no longer be traded. We term the union of Segment crossing and Instrument crossing the Instrument Session.

Session reports include opening prices and closing prices.

Table 12-3: Instrument Session Summary Detail

Attribute	FIX Field Name	Attribute Type	Meaning
Begin DateTime	BeginDateTime	DateTime	StartTime that identifies this Instrument, and thereby this Instrument Session Summary Detail
High Price	HighPx (332)	Price	The high end of the price range prior to the open or reopen

Attribute	FIX Field Name	Attribute Type	Meaning
Low Price	LowPx (333)	Price	The low end of the price range prior to the open or reopen
First Price	FirstPx (1025)	Price	Indicates the first price of a trading session; can be a bid, ask, or trade price.
Last Price	LastPx (31)	Price	Indicates the last price of a trading session; can be a bid, ask, or trade price.

12.3.2 The Instrument Book Summary

The Book is the set of all Tenders, including Tradable Quotes, in the Market Segment. In an active market, unless there are restrictions on matching, all Tenders to sell are priced higher than all Tenders to buy; if there were an overlap, they would already be matched, Transactions generated, and the Tenders already removed from the book.

All Tenders are sorted by price. Tenders to sell are sorted by ascending price—the top of the book is the tender offering the cheapest price. Tenders to buy are sorted by descending price—the top of the book is the tender offering the highest price.

The depth of the Book is the number of Tenders in each list. A Top of the Book request, subscription depth of 1 provides just the top entry in each list. anonymized. A subscription depth of 0 provides both entire sorted lists, anonymized. Any other subscription level (n) provides the first (n) entries in each level.

Table 12-4~~is out of scope, as is the~~: Instrument Book Subscription

Attribute	Attribute Type	FIX Field Name	Meaning
Begin DateTime	DateTime	BeginDateTime	StartTime that identifies this Instrument, and thereby this Instrument Session Summary Detail
Book Entry	Compound		Repeating element for each side and level of the Book

The Book Entry is the repeating information for each Side in the Book. The Book Entry is the same message format as a Quote, anonymized as required by market rules.

The Book entry is always summarized. That is, if there are five Tenders at the top of the offer side with the same price, the quantity returned in the Book Entry is the sum of the Quantities in each of those Tenders. The same rule applies to bid side of the Book.

13 Market Structure Subscriptions

For any Market, there are standing terms and expectations about Product offerings. If these standing terms and expectations are not known, many exchanges may need to occur before finding what Products and Tenders meet those expectations. The Market Structure Report tells the Party how to trade. Following the classification used by FIX, Market Structure is just one part of Pre-Trade Information.

Market Structure changes are limited to Opening, Closing, Settlement and to the currently Active Segments.

Segment Structure includes Opening, Closing, as well as Crossing information for specific Instruments. It also includes detailed information to guide Training, detailed rules and negotiations and trading.

13.1.1 Venue Types

One of the most important distinctions between Market Segments is the Venue Type.

FIX defines the Venue Type to describe the general mechanism ~~for publishing. The payloads are as follows:~~of trading. A Party participating in trading may change its behavior based on the Venue Type. The optimum trading strategy for a Party will change between an order book and an auction. If there is only a single seller, the Buyer will want to attend closely to the quotes from that buyer.

The Venue Type categorizes how a Transaction is generated by the Venue. A Party wishing to buy or to sell will change strategies based on the rules of the Venue. FIX defines many Venue Types (FIX tag 1430), only some of which are supported by CTS-conforming Markets. Table 13-1 lists the Venue Types supported by CTS.

Table 13-1: Venue Types in CTS

FIX Code	Name	Meaning
B	OrderBook	Participants submit their buy and sell orders, which are matched based on specific rules and executed accordingly. Order Books provide transparency, liquidity, and support for different types of orders, but they also have limitations and rely on the integrity and fairness of the exchange. Also referred to as a Centralized Limit Order Book. CTS specifies a "hard" order book, which executes orders immediately and automatically
O	OffMarket	Trades are conducted outside the market, but registered in the automated exchange. Sometimes referred to as an Over-the-counter (OTC) or Bilateral market. Off-Market trades may be for odd lots, for custom durations, and span the temporal boundaries of Products. The exchange may reject a trade that would cause violation of the physical limits of the Resource distribution.
Q	QuoteDrivenMarket	Quote Driven Markets are used for Markets with a single dominant supplier. Parties are notified of the Quoted price for each Instrument and submit Tenders to match those Quotes. Example: A System Operator provides 24 quotes for the next day's hourly pricing.
N	QuoteNegotiation	A Quote Negotiation Market is used for bilateral negotiations around price. Sellers may advertise round lots that they would like to buy or sell, and indicate an interest in buying or selling.

FIX Code	Name	Meaning
A	AuctionDrivenMarket	An Auction Driven Market matches Tenders only in scheduled auctions wherein all participants clear at the same price. In Resource Markets, also referred to as a “Double Auction”.
S	SpotMarket (not in FIX)	A Spot Market indicates the “instant” price in the Market Segment that will be used for purchases or sales.
X	Settlement (not in Fix)	A Settlement venue self-executes Transactions for Resource consumed without previously being bought.

13.1.2 Interaction Pattern for Market Structure

The Market Definition Facet enables a Party to request the details of a Market and its Market Segments. The initial request returns the Market and all Segments. Subsequent reports occur when there is a change to a Market Segment and include only the changed Market Segment(s). A request to refresh a subscription returns the entire Market Structure. A request to cancel the subscription suspends all further updates.

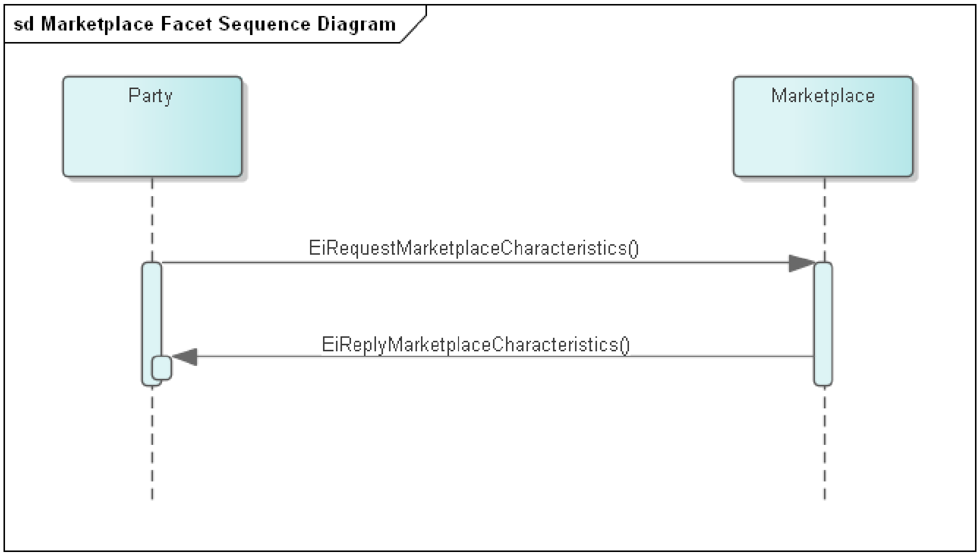

sd Marketplace Facet Sequence Diagram
Party
Marketplace
EiRequestMarketplaceCharacteristics()
EiReplyMarketplaceCharacteristics()

Figure 13-1: UML Sequence Diagram for the Market Definition Facet

A Party may watch changes to a single Market by naming that Segment in the subscription request. This will return only that Segment and updates to that Segment.

13.2 Market Definition

Table 13-2 Information Model for the Market Definition

Attribute	Attribute Type	FIX Field Name	Meaning
Market Name	String	NAME	Text providing a descriptive name for a Market. While the Name MAY be displayed in a user interface; it is not meaningful to the Actors.

Attribute	Attribute Type	FIX Field Name	Meaning
Currency	String	Currency (537)	String indicating how value is denominated in a market.
Currency Code Source	String	Currency Code Source (2897)	ISO – Fiat Currency per ISO 4217 DTI – Digital Token Identifier LOC – Locally defined Currency
Time Offset	Duration	T_OFFSET	A Duration that some Markets MAY use to describe trading where a first interval is not on an hourly boundary.[24]
Tick Size	Price	Tick Increment (1208)	Specifies the valid price increments at which a Party may quote or trade an Instrument.[25]. Use if a common Tick Increment required for all Market Segments. Tick Increments can increase market liquidity.
Resource Designator		Resource	The Resource traded in this Market and Segment
Party Id		Party Id	The PartyID used in Tenders to the Market and in Transactions with the Market.
Market Segments		Market Segment	A list of Market Segment descriptions for each Market Segment contained in the Market. See Section 13.3.
Maximum		MAX	Maximum Transaction size the Market will accept.

13.3 Segment Definition

A Party must interact with a specific Market Segment to trade a specific Product. A Market MAY contain two or more Market Segment trading the same Product; such segments may differ in venue type, or in trading window. For example, a regulated provider may offer a day-ahead hourly market based on an Auction between 9:00 AM and 3:00 PM. Thereafter, a forward Market in the same Product may move to another Market Segment using an Order Book. The Auction and the Order Book are different venues, matching buyer, and seller by different mechanisms.

A Party chooses the Market Segment that it anticipates will be to its greatest advantage. The Party will make this choice based on anticipated price, or on block size, or even on Warrant. Because Transactions are committed when created, a Party may buy on one Market Segment, and thereafter sell part of it on another.

[24] A power distribution entity may experience disruption if there is a big price change on the hour. For example, a distribution system operator (DSO) that operates multiple CTS Markets could opt to set a different offset on each Market Segment operated out of a given substation. In this model, a Market could use an offset duration of 3 minutes to indicate that all tenders are based on three minutes after the hour.

[25] Integer operations are typically much more efficient than fixed or floating point, so it is likely to be much faster to apply decimal shift on input and output rather than for more frequent comparison operations in the Execution Engine implementation.

A Party discovers Market Structure, including changes over time, by subscribing to that Market. Even without market activity, the information in a Subscription may change. For example, a Segment may open or close and the biddable Instruments change regularly.

- Table 13-3~~EiSubscribeTicker – PartyID for the sender and the Ticker Reference for the market as found in Market Characteristics~~
- ~~EiSubscribedTicker – EiResponse indicating success or failure. Implementations MAY send information related to the Subscribed stream.~~
- ~~EiUnsubscribeTicker – Removes the subscription to the indicated Ticker Reference~~
- ~~EiUnsubscribedTicker – EiResponse indicating success or failure. Implementations MAY send information related to the now un-Subscribed stream.~~
- ~~EiDistributeTicker – Publish an EiTicker to the Ticker Reference.~~

Market Segment Description

Attribute	**Attribute Type**	**FIX Field Name**	**Meaning**	**Comments**
Market Segment ID	String	SEGMENT (1300)	Unique Identifier for Segment	
Market Segment Name	String	SEG_NAME (1396)	Optional text providing a descriptive name for a Market Segment.	While the Name MAY be displayed in a user interface; it is not meaningful to the Actors.
Product	Compound		Product transactable this Segment. See Defining Product (Section 3.1.2) for details.	Each Product shares a Resource with the Market
End Point	String	EndPoint	End point to access the Market Segment.	May be a transport end point, mailbox address or other.
Segment Status		MktSegStat	Current trading status of the Market Segment.	1 = Active: Market segment is active, i.e. trading is possible. 2 = Inactive: Market segment has previously been active and is not currently Open. 3 = Published: Market segment information is provided prior to its first activation.
Tradable Interval	Interval		Instruments whose start is within the bounding of the Interval can be traded	
Session Start Time	DateTime	TradSes StartTime	Date and Time when Tenders may first be submitted for the current or next Session	
Session Open Time	DateTime	TradSes OpenTime	Date and Time Market Segment next opens (or when current or last session Opened)	

Attribute	Attribute Type	FIX Field Name	Meaning	Comments
Session Close Time	DateTime	TradSes CloseTime	Date and Time current Market Segment next Closes (or when last session Closed)	
Venue Type	String	VenuTyp	Description of Venue used to match and execute trades.	See Table 13-1, below.
Residue Disposition	String	RESIDUE (Optional)	How is Residue processed. Allowable values are CANCEL / CONVERT / TRANSFER	Residue is the unfulfilled Tenders on the books when the Segment or Instrument closes. While Resource markets are typically continuously open, Individual Instruments close. i.e., 11:00 AM power closes at 11:00 AM.
Successor Segment	String	SUCCESSOR (Optional)	The Successor is the Segment ID of the Market Successor that is the follow-on after the closing of this Segment.	If Residue is "TRANSFERred" then this is the segment it is transferred to.
Maximum	Integer	MaxTradeVol	The maximum order quantity (as round lots) that can be submitted.	
Execution Instructions	String		A list of FIX Execution Instructions that are accepted in this Segment (see Table 1-1),	
Stream Trading	Integer	MLegOK[26] (Optional)	Applies to both Tenders and Quotes	0 – Prohibited (default if missing) 1 – Permitted 2 - Required
Negotiations Permitted		(Optional except Mandatory for Venue Type "Q")	Segment supports Negotiation	0 – Prohibited (default if missing) 1 – Permitted 2 – Private Quotes Only 3 – Advertised Quotes Only
Private Negotiations through Market	Integer	(Optional. Prohibited if missing)	Private Quotes are sent to the Segment which then forwards them to Counterparties	0 – Prohibited – Private Quotes not forwarded by Segment 1 – Permitted – Segment forwards Private Quotes to listed CounterParties
Market Segment ID	Int	MarketSegmentID (1300)	Identifies the Segment processing the Tender, Transaction, or Quote	This should be a unique combination paired with the Market Order ID

[26] Fix refers to a trade of multiple instruments together as a Multi-Legged Trade

Attribute	Attribute Type	FIX Field Name	Meaning	Comments
Market Instrument Summary	String	(Optional)	If blank or absent, no Market Instrument Summary is available	
Max Summary Instruments	Int		0 – U Unlimited Instruments 1-N – Maximum Instruments in a Subscription	
Top of Book Depth	Int		0 – Unlimited 1-N – Up to N levels of Book can be requested	
Ticker Advertisement	Int		Advertisement is publication of Tenders and Advertised Quotes as they are entered into the Book	0 – Unavailable 1 – Available for this segment
Ticker Announcement	Int		Announcement of Transactions, that is, Matched Tenders and completed Negotiations in this Segment.	0 – Unavailable 1 – Available for this segment
The fields below are as defined in the Tender, Table 1-2				
Duration	Duration	*Not in FIX*	Duration of delivery.	Part of all Instruments
Price Scale	Int	Not Defined in FIX	A multiplier for the Price	A Market Segment may be denominated in, for example dollars or 10ths of a cent.
Quantity Scale	Int		A scale factor on the Resource unit for this Market	For example, "mega" vs "kilo" vs "femto-"
Resource Designator	Resource Designator Enumeration	FIX Instrument component	Identifier of the Resource being offered (Optional in many markets)	While a Market only accepts Tenders and Quotes for a single Resource, the complete description is required to ensure validity and for off-market interactions.
Warrant Ref	Int	*Not in Fix*	Reference to Warrants as defined in the Market	If used, see comments Warrants in Tenders, Section 5.3.3.

1903

1904

Bindings

Payloads and interaction patterns are described in **[UML]** in Sections 6 through 12 above. This section contains bindings for the payloads in three encoding schemes:

- JSON **[JSON]**
- XML Schema **[XSD]**
- FIX Simple Binary Encoding **[SBE]**

~~13.5~~13.4 JSON

PENDING—JSON Schema awaiting stable payload definitions

~~13.6~~13.5 XML Schema

PENDING—XML Schema awaiting stable payload definitions

~~13.6.1~~13.5.1 XML Namespaces

PENDING—XML Namespaces awaiting XML Schema

~~13.7~~13.6 Simple Binary Encoding

TODO—SBE Schema awaiting stable payload definitions.

14 Conformance

14.1 Introduction to Conformance

By design, CTS is a simplified and restricted subset profile of TeMIX. See Appendix

Portions of CTS conform to, and use updated and simplified versions of the specifications consumed by EI, specifically:

- OASIS WS-Calendar **[WS-Calendar]**
- A definition of Streams contained in **[EI]**

We normatively reference and apply the evolution of these specifications, in particular:

- OASIS WS-Calendar Schedule Streams and signals **[Streams]**, simplified as CTS Streams~~ (see ~~.
- The WS-Calendar **[CAL-MIN]** ~~interval~~Interval is used directly (as IntervalType).

This specification simplifies WS-Calendar Schedule Streams and Signals [Streams] as CTS Streams (see Section 3.2~~,~~) and refactors the TEMIX profile of **[EI].**

Conformance of the CTS evolved specification can be shown with the techniques of **[IEC62746-10-3]** is described in informative Appendix D.

14.2 Claiming Conformance to Common Transactive Services

Implementations claim conformance to Common Transactive Services 1.0 by asserting conformance statements on the numbered items below.

1. The conformance statement MUST list all Facets which it supports in full or and in part.
2. The conformance statement MUST describe all extensions to payloads described in this specification.
3. The conformance statement MUST describe the Binding(s) which it supports along with any extensions. If the implementation does not use a standard binding as defined in Section 13, the conformance statement MUST define the binding used, at a similar level to detail to Section 13.
4. The conformance statement MUST describe how each payload definition conforms to the UML and/or profiled definitions for each payload unless it uses only standard Bindings in Section 13.
5. The conformance statement MUST indicate cardinality for message payload attributes where there is flexibility in this specification.
6. The conformance statement MUST describe any facets it defines to extend this specification.

14.3 Annex: Conformance statements from Spec not yet incorporated into this section

14.3.1 Conforming with Use of Warrants in Tenders

Warrants increase the specificity of Product (and Instrument). A Buyer who does not specify a Warrant will be satisfied Delivery of a Product whether or not it has a Warrant. A Buyer who requests Product with a Warrant will only be satisfied by Delivery of a Product that has that Warrant.

Consider a buyer who wishes to buy a package of coffee beans and a buyer who wishes to buy a package of organic coffee beans. The word “Organic” on the label serves as a Warrant. The first buyer will buy solely on price, and is indifferent to seeing the word “Organic” on the label. The second buyer will choose only from among those packages with the warrant “Organic” on the label.

When a Tender on the Buy side specifies a Warrant, it must be rejected by any Market Segment that does not include that Warrant. A Tender on the Sell side that specifies a Warrant may be accepted by any Segment where the same Resource and Duration are traded. Conversely, a Tender on the Sell side without a Warrant must be rejected by any Segment that specifies a Warrant.

Appendix A. References

This appendix contains the normative and informative references that are used in this document. Normative references are specific (identified by date of publication and/or edition number or Version number) and Informative references may be either specific or non-specific.

While any hyperlinks included in this appendix were valid at the time of publication, OASIS cannot guarantee their long-term validity.

A.1 Normative References

The following documents are referenced in such a way that some or all of their content constitutes requirements of this document.

NOTE: INSERT AS FORMATTED REFERENCES. Consider [EI]

[CAL-MIN]
WS-Calendar Minimal PIM-Conformant Schema Version 1.0. Edited by William Cox and Toby Considine. 26 August 2016. OASIS Committee Specification. http://docs.oasis-open.org/ws-calendar/ws-calendar-min/v1.0/ws-calendar-min-v1.0.html

[CAL-PIM]
OASIS WS-Calendar Platform-Independent Model version 1.0, Committee Specification 02 Edited by William T. Cox and Toby Considine, 21 August 2015. http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/cs02/ws-calendar-pim-v1.0-cs02.html Latest version: http://docs.oasis-open.org/ws-calendar/ws-calendar-pim/v1.0/ws-calendar-pim-v1.0.html

[EI]
Energy Interoperation Version 1.0. Edited by Toby Considine, 11 June 2014. OASIS Standard. http://docs.oasis-open.org/energyinterop/ei/v1.0/os/energyinterop-v1.0-os.html Latest version: http://docs.oasis-open.org/energyinterop/ei/v1.0/energyinterop-v1.0.html. and its TeMIX Profile

[EMIX] OASIS Energy Market Information Exchange (EMIX) Version 1.0 Committee Specification 02 Edited by Toby Considine, 11 January 2012. http://docs.oasis-open.org/emix/emix/v1.0/cs02/emix-v1.0-cs02.html Latest version: http://docs.oasis-open.org/emix/emix/v1.0/emix-v1.0.html

[JSON]

JavaScript Object Notation and JSON Schema. https://cswr.github.io/JsonSchema/

[RFC8174]
Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <http://www.rfc-editor.org/info/rfc8174>.

[RFC2119]
Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2246]
T. Dierks, C. Allen *Transport Layer Security (TLS) Protocol Version 1.0*, http://www.ietf.org/rfc/rfc2246.txt, IETF RFC 2246, January 1999.

[SBE]
Simple Binary Encoding Technical Specification 1.0. FIX Trading Community, June 16, 2016. https://www.fixtrading.org/standards/sbe/

[Streams]
Schedule Signals and Streams Version 1.0. Edited by Toby Considine and William T. Cox. 18 September 2016. OASIS Committee Specification. http://docs.oasis-open.org/ws-calendar/streams/v1.0/streams-v1.0.html.

A.2 Informative References

The following referenced documents are not required for the application of this document but may assist the reader with regard to a particular subject area.

[Actor Model]
C. Hewitt, "Actor Model of Computation: Scalable Robust Information Systems," arxiv.org, 2010.

[Fractal Microgrids]
Art Villanueva et al, *Camp Pendleton Fractal Microgrid Demonstration,* California Energy Commission Report CEC-500-2016-013,j available at http://400.sydneyplus.com/CaliforniaEnergy_SydneyEnterprise/Download.aspx?template=Books&field=PublicURL&record=57483797-a40e-49e7-b675-2858a3ad0d91&showSave=False&repeat=d4e63b56-27d1-4476-9300-7ede86a533ca

[Framework]
National Institute of Standards and Technology, *NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 1.0*, January 2010, http://nist.gov/public_affairs/releases/upload/smartgrid_interoperability_final.pdf

[CTS2016]
W.T. Cox, E. Cazalet, E., A Krstulovic, W Miller, & W.Wijbrandi *Common Transactive Services*. TESC 2016. Available at http://coxsoftwarearchitects.com/Resources/TransactiveSystemsConf2016/Common%20Transactive%20Services%20Paper%2020160516.pdf

[EML-CTS]
Energy Mashup Lab Common Transactive Services (open-source software) https://github.com/EnergyMashupLab/eml-cts)

[FSGIM]
Facility smart grid information model. ISO 17800. https://www.iso.org/standard/71547.html 2017

[iCalendar]
B. Desruisseaux*, Internet Calendaring and Scheduling Core Object Specification (iCalendar)*, https://tools.ietf.org/html/rfc5545. 2009,
See also
C. Daboo & M. Douglas. *Calendar Availability*, https://tools.ietf.org/html/rfc7953, 2016

[GridFaultResilience]
W.T. Cox & T. Considine. *Grid Fault Recovery and Resilience: Applying Structured Energy and Microgrids*~~..~~. IEEE Innovative Smart Grid Technologies 2014. Available at http://coxsoftwarearchitects.com/Resources/ISGT_2014/ISGT2014_GridFaultRecoveryResilienceStructuredMicrogrids_Paper.pdf

[IEC62746-10-3] International Standard. Systems interface between customer energy management system and the power management system - Part 10-3: Open automated demand response - Adapting smart grid user interfaces to the IEC common information model, https://webstore.iec.ch/publication/59771 2018.

[Micromarkets]
W.T. Cox & T. Considine, *Energy, Micromarkets, and Microgrids.* GridInterop 2011, https://www.gridwiseac.org/pdfs/forum_papers11/cox_considine_paper_gi11.pdf

[RFC3552]
E Rescorla & B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[SmartGridBusiness]
T. Considine & W.T. Cox, *Smart Loads and Smart Grids—Creating the Smart Grid Business Case*. Grid-Interop 2009. Available at http://coxsoftwarearchitects.com/Resources/Grid-Interop2009/Smart%20Loads%20and%20Smart%20Grids.pdf

[StructuredEnergy]
Structured Energy: Microgrids and Autonomous Transactive Operation, http://coxsoftwarearchitects.com/Resources/ISGT_2013/ISGT-Cox_StructuredEnergyPaper518.pdf. Innovative Smart Grid Technologies 2013 (IEEE).

[TPB-EI] Transport Protocol Bindings for OASIS Energy Interoperation 1.0 Version 1.0. 01 October 2012. OASIS Committee Specification 01.
.

[TeMIX] TeMIX Transactive Energy Market Information Exchange [TeMIX] an approved Note of the EMIX TC. Ed Cazalet et al., 23 May 2010. http://www.oasis-open.org/committees/download.php/37954/TeMIX-20100523.pdf

[TransactiveMicrogrids]
Jennifer M. Worrall, Edward G. Cazalet, PhD, William T. Cox, PhD, Narayanan Rajagopal, Thomas R. Nudell, PhD, and Paul D. Heitman, *Energy Management in Microgrid Systems,*TESC 2016. Available at
http://coxsoftwarearchitects.com/Resources/TransactiveSystemsConf2016/tes2016_microgrids_paper_Final.pdf

[TRM] (Transactive Resource Management)
B. Huberman and S. H. Clearwater, *Thermal markets for controlling building environments*, Energy Engineering, vol. 91, no. 3, pp. 26- 56, January 1994.

[UML]
Object Management Group, *Unified Modeling Language (UML), V2.4.1*, August 2011.
http://www.omg.org/spec/UML/2.4.1/

[XSD]
W3C XML Schema Definition Language (XSD) 1.1. Part 1: Structures, S Gao, C. M. Sperberg-McQueen, H Thompson, N Mendelsohn, D Beech, M Maloney http://www.w3.org/TR/xmlschema11-1/, April 2012, Part 2: Datatypes, D Peterson, S Gao, A Malhotra, C. M. Sperberg-McQueen, H Thompson, P Biron, http://www.w3.org/TR/xmlschema11-2/ April 2012

[ZeroTrust]
Zero Trust Architecture, S Rose, O Borcher, S Mitchell, S Connelly. NIST Special Publication 800-207, https://doi.org/10.6028/NIST.SP.800-20 August 2020

Appendix B. Security and Privacy Considerations

This specification defines message payloads only. Security must be composed in. Privacy considerations must be decided when implementing specific systems for specific purposes.

B.1 CTS and Security Considerations

Procuring energy for local use and selling energy for remote use are each at the cusp of finance and operations.

- A price that is falsely low may cause the buyer to operate a system when there is inadequate power, potentially harming systems within a facility, or harming other facilities on the same circuit.
- A price that is falsely low may cause the seller to leave the market.
- A price that is falsely high may cause the buyer to shut down operation of systems or equipment.
- A price that is falsely high may cause the seller to increase operations when there is neither a ready consumer ~~or~~nor perhaps even grid capacity to take delivery.

For these reasons, it is important that each system guard the integrity of each message, assure the identities of the sender and of the receiver, and prove whether a message was received or not.

Messages should be encrypted to prevent eavesdropping. Any node should be able to detect replay, message insertion, deletion, and modification. A system must guard against and detect man-in-the-middle" attacks wherein an intermediary node passes of messages as originating from a known and trusted source.

The Technical Committee generally recommends that production implementations use Zero-Trust security **[ZeroTrust]**, especially because of the wide distribution and potentially diverse ownership of TRM Actors. Zero Trust security requires authentication and authorization of every device, person, and application. The best practice is to encrypt all messages, even those between the separate components of an application within the cloud.

This specification makes no attempt to describe methods or technologies to enable Zero Trust interactions between Actors.

B.2 CTS and Privacy Considerations

The United Nations has defined privacy as “the presumption that individuals should have an area of autonomous development, interaction and liberty, a ‘private sphere’ with or without interaction with others, free from state intervention and excessive unsolicited intervention by other uninvited individuals. The right to privacy is also the ability of individuals to determine who holds information about them and how that information is used” (UN General Assembly 2013:15).

Electrical usage data inherently creates a privacy risk. Published work has demonstrated that simple usage data can be used to reveal the inner operations and decisions in a home. Other research has demonstrated that anonymous electrical usage data can be “de-anonymized” to identify an individual electricity user. The more fine-grained the data, the more intimate the details that can be garnered from meter telemetry.

In an amicus brief in a case on smart metering, the Electronic Freedom Foundation testified that that aggregate smart meter data collected from someone’s home in 15-minute intervals could be used to infer, for example, whether they tend to cook meals in the microwave or on the stove; whether they make breakfast; whether and how often they use exercise equipment, such as a treadmill; whether they have an in-home alarm system; when they typically take a shower; if they have a washer and dryer, and how often they use them; and whether they switch on the lights at odd hours, such as in the middle of the night. And these inferences, in turn, can permit intimate deductions about a person's lifestyle, including their occupation, health, religion, sexuality, and financial circumstances. These privacy concerns are linked to increased security risks criminals may be able to access the data and use the information to enable inferences about what people are doing in their home or if they are away from home.

This specification describes how to share communications beyond mere electrical usage telemetry. Communications reveal what the user would like to buy, how much they would be willing to spend, and future intents and plans.

System developers using this specification should consider legal requirements under the Fair Practice Principles and the European Union's General Data Protection Regulation. These include:

1) The Collection Limitation Principle. There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject.
2) The Data Quality Principle. Personal data should be relevant to the purposes for which they are to be used and, to the extent necessary for those purposes, should be accurate, complete and kept up ~~-~~to ~~-~~date.
3) The Purpose Specification Principle. The purposes for which personal data are collected should be specified not later than at the time of data collection and the subsequent use limited to the fulfillment of those purposes or such others as are not incompatible with those purposes and as are specified on each occasion of change of purpose.
4) The Use Limitation Principle. Personal data should not be disclosed, made available or otherwise used for purposes other than those specified, except a) with the consent of the data subject, or b) by the authority of law.
5) The Security Safeguards Principle. Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorized access, destruction, use, modification or disclosure of data.
6) The Openness Principle. There should be a general policy of openness about developments, practices and policies with respect to personal data. Means should be readily available of establishing the existence and nature of personal data and the main purposes of their use, as well as the identity and usual residence of the data controller.
7) The Individual Participation Principle. An individual should have the right:

to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him~~;~~.

to have data relating to him communicated to him, within a reasonable time, at a charge, if any, that is not excessive; in a reasonable manner, and in a form that is readily intelligible to him~~;~~.

to be given reasons if a request made under subparagraphs (a) and (b) is denied and to be able to challenge such denial; and

to challenge data relating to him and, if the challenge is successful, to have the data erased, rectified, completed or amended~~;~~.

8) The Accountability Principle. A data controller should be accountable for complying with measures which give effect to the principles stated above.

In developing this specification, the Technical Committee has kept in mind the need to support a developer wishing to support privacy. Actors representing an up-stream electrical serving entity, say a distribution system operator or traditional utility, use the tame messages as anyone else—no actor is inherently privileged. Messages to provide market information or “~~ticker-tape~~tickertape” functions do not include Party IDs. General advertising of Tenders, while necessary to draw matching Tenders quickly to market, may be anonymous.

In some messages and some markets, it is necessary to use a proxy ID to protect privacy or to simply conveyance of a transaction from a complex matching mechanism. To protect privacy, a market may transmit such a proxy ID in place of a Party Id in Quotes, Tenders, Transactions, and Tickers. Markets that use cumulative matching algorithms such as double auction cannot identify a specific Counter Party to a transaction.

The system developer should keep the privacy principals in mind when making specific technology choices. For example, messages between an actor and the market MAY be encrypted to protect the privacy of people represented by individual actors. While the transactive energy market must know both buyers and sellers to support ~~transaction contracts~~transactions and settlements, the developer should take steps to guard that information. A developer may opt that each notice of contract sent to an actor always has a counterparty of the market, so as to protect the sources and uses of electricity.

It is beyond the scope of this specification to specify security practices and privacy design for markets built using this specification.

Appendix C. Semantic Composition from Energy Interoperation, EMIX, and WS-Calendar

The semantics and interactions of CTS are selected from and derived from [EI].

EI references two other standards, [EMIX] and [WS-Calendar], and uses an earlier Streams definition. We adapt, update, and simplify the use of the referenced standards, while maintaining conformance.

- EMIX describes price and product for electricity markets.
- WS-Calendar communicates schedules and sequences of operations. CTS uses the [Streams] optimization, which is a standalone specification, rather than part of EI 1.0.
- EI uses the vocabulary and information models defined by those specifications to describe the services that it provides. The payload for each EI service references a product defined using **[EMIX]**. EMIX schedules and sequences are defined using [WS-Calendar]. Any additional schedule-related information required by [EI] is expressed using [WS-Calendar].
- Since [EI] was published, a semantically equivalent but simpler [Streams] specification was developed in the OASIS WS-Calendar Technical Committee. CTS uses that simpler [Streams] specification.

All terms used in this specification are as defined in their respective specifications.

In [EI], the fundamental resource definition was the [EMIX] Item, composed of: a resource name, a unit of measure, a scale factor, and a quantity. For example, a specific EMIX Item may define a Market denominated in 25 MWh bids. [EI] defined how to buy and sell items during specific intervals defined by a duration and a start time. The Quotes, Tenders, and Transactions that are the subject of [EI] added specific prices and quantities to the item and interval. EMIX optionally included a location, i.e., a point of delivery for each [EI] service.

In CTS, we group and name these elements as a Resource, Product, and Instrument. These terms are defined in Section 2.2.4, “Markets and Venues”

Note that the informational elements in a fully defined tender or transaction are identical to those described in EMIX. The conceptual regrouping enables common behaviors including Market discovery and interoperation between Actors built on different code bases.

14.3.2 Conformance with Energy Interoperation

EI defines an end-to-end interaction model for transactive services and for demand response. CTS uses the EI transactive services and draws definitions of parties and transactive interactions primarily from the [EI] TEMIX profile.

This specification can be viewed as a minimal transactive profile of [EI].

14.3.3 Conformance with EMIX

This specification uses a simplified profile of the models and artifacts defined in OASIS Energy Market Information Exchange **[EMIX]** to communicate Product definitions, quantities, and prices. EMIX provides a succinct way to indicate how prices, quantities, or both vary over time.

The EMIX Product definition is the Transactive Resource in CTS 1.0.

EMIX defines Market Context, a URI used as the identifier of the Market. EMIX further defines Standard Terms as retrievable information about the Market that an actor can use to configure itself for interoperation with a given Market. We extend and clarify those terms, provide an extension mechanism, and discuss the relationship of markets, market segments, and products.

14.3.4 Conformance with WS-Calendar Streams

WS-Calendar expresses events and sequences to support machine-to-machine (M2M) negotiation of schedules while being semantically compatible with human schedules as standardized in [**iCalendar**].

Schemas in **[WS-Calendar]** support messages that are nearly identical to those used in human schedules. We use a conformant but simpler and more abstract Platform Independent Model **[CAL-PIM]** and the [**Streams**] compact expression[27], to support telemetry (Delivery Facet) and series of Tenders while not extending the semantics of [**Streams**].[28]

WS-Calendar conveys domain specific information in a per-event payload within a schedule-centric message; in CTS, the domain is the price, product, and quantity. An essential concept of WS-Calendar is inheritance, by which a starting time can be applied to an existing message, or by which all events in a sequence share common information such as duration. Inheritance is used to “complete” a partial message during negotiation. CTS makes use of this to apply a common product across a sequence, or to convey a specific starting time to a market product.

14.3.5 CTS and WS-Calendar Streams

The **[Streams]** specification describes how to handle repeating time series of similar data, applying repeating information to a series of schedulable intervals, expressing common information once for the series, overriding the common information only if needed within a specific interval, and potentially scheduling (“binding”) the entire series by adding a starting date and time to one of the Intervals.

For CTS, this means that a Product is fully described in the header, and only the elements that vary, such as the Price or the Quantity, are expressed in the intervals.

CTS Streams use this same format even when the Intervals contain only a single Interval.

In addition, CTS Streams include energy-market elements that are outside the Streams standard but follow the pattern of referrals as defined in [Steams] conformance.

CTS Streams have neither interaction patterns nor payloads, as they are a common abstract information model used to define the messages used in Facet messages.

The CtsStream follows this pattern. The elements from **[Streams]** have been flattened into the CTS Stream, and the Stream Interval and payload flattened into a streamPayloadValue and the internal local UID for the stream element.

[27] Simplified as CTS Streams in this specification.

[28] Some specifications (e.g. [FSGIM]) have extended the basic [Streams] capabilities, but this brings additional complexity which does not benefit our use cases.

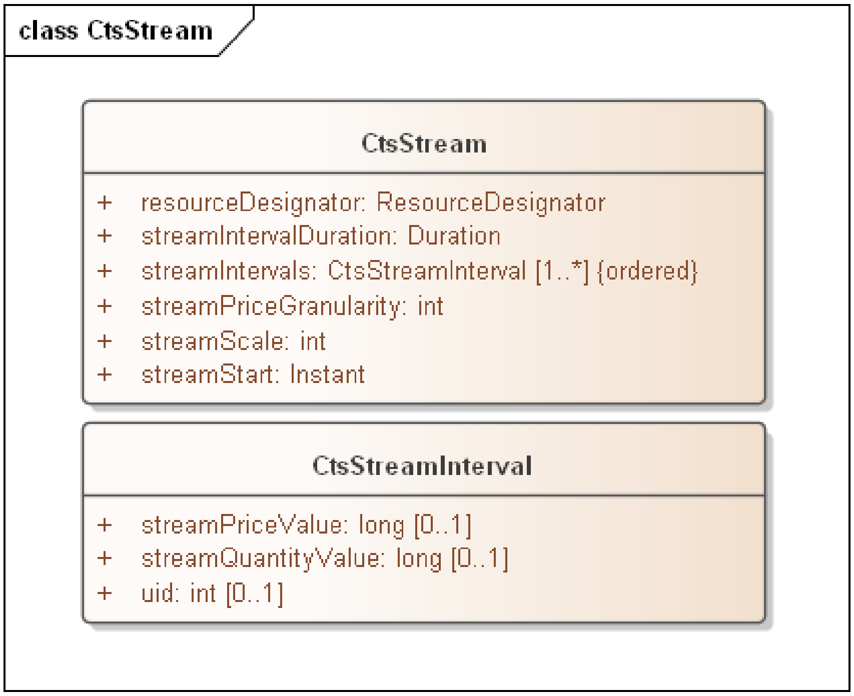

class CtsStream
CtsStream
+ resourceDesignator: ResourceDesignator
+ streamIntervalDuration: Duration
+ streamIntervals: CtsStreamInterval [1..*] {ordered}
+ streamPriceGranularity: int
+ streamScale: int
+ streamStart: Instant
CtsStreamInterval
+ streamPriceValue: long [0..1]
+ streamQuantityValue: long [0..1]
+ uid: int [0..1]

Figure C-14-1: CtsStreamDefinition

As with **[Streams]**, CtsStreamIntervals are ordered, that is the sequence of intervals is essential. Some serialization specifications, notably XML, do not require that order be preserved when deserializing a list. The UID enables proper ordering of the Stream Intervals if order is not preserved. Since conformant CTS implementations need not be owned by the same implementer, and may pass through multiple translations, the UID property is required.

~~Appendix C.~~Appendix D. Conformance to the TEMIX Profile of Energy Interoperation

TBD

~~Appendix D.~~Appendix E. Glossary of Terms and Abbreviations Used in this document

Throughout this document, abbreviations are used to improve clarity and brevity, especially to reference specifications with long titles.

Table C--14-1 Abbreviations and Terms used throughout this document

Attribute	Meaning
CTS	Common Transactive Services
EI	Energy Interoperation, an OASIS specification as per the normative references, CTS is a conforming profile of EI.
EMIX	Energy Market Information Exchange, an OASIS specification used to describe Products and markets for resources, particularly those traded in power grids.

~~Appendix E.~~Appendix F. Acknowledgments

This work is derived from the specification Common Transactive Services 1.0 , contributed by The Energy Mashup Lab, written by William T. Cox and Toby Considine.

Portions of models and text is derived from The Energy Mashup Lab open-source project, EML-CTS and is used under terms of the Apache 2.0 License for that project.[29]

~~E.1~~F.1 Participants

The following individuals were members of this Technical Committee during the creation of this document and their contributions are gratefully acknowledged:

Rolf Bienert, OpenADR Alliance
Toby Considine, University of North Carolina at Chapel Hill
William T. Cox, Individual Member
Pim van der Eijk, Sonnenglanz Consulting
David Holmberg, National Institute for Standards & Technology (NIST)
Elysa Jones, Individual
Chuck Thomas, Electric Power Research Institute (EPRI)

[29] https://github.com/EnergyMashupLab/eml-cts

~~Appendix F.~~Appendix G. Revision History

Revision	Date	Editor	Changes Made
WD01	2/15/2021	Toby Considine	Initial reformatting and conversion of the specification contributed by The Energy Mashup Lab to create a document for committee work.
WD02	3/3/2021	Toby Considine	Added prose definitions of Resource, Product, and Instrument
WD03	4/5/2021	Toby Considine	Simplified introductory material, raised message type to earlier in document. Removed some repetitive material. Revised UML required.
WD04	5/7/2021	Toby Considine David Holmberg William T Cox	Reordered intro material to reduce repetition, Reference Actor Model more consistently, Revise and re-factor Resource/Product/Instrument Add Section 3 to elevate common semantic elements
WD05	5/25/2021	Toby Considine David Holmberg William T Cox	Continues clean-up and condensation of sections 1, 2
WD06	6/7/2021	Toby Considine	Refines Item language into Resource and Products. Explains Message Groups as a conforming descendant of EI Services.
WD07	6/21/2021	Toby Considine William T Cox	Clarified terminology and relationship to implied Service-Oriented Architecture. Structured CTS facets for clearer explanation
WD08	8/5/2021	Toby Considine William T Cox David Holmberg	Clarify and simplify actor facets descriptions, including Tender, Transaction, and Configuration. Reduce redundant and less relevant content.
WD09	9/14/2021	William T Cox Toby Considine David Holmberg	Added Facet descriptions for Position, Market Characteristics, CTS Streams, and drafts of Privacy Consideration, Delivery and Party Registration Facets. Numerous edits for clarity and conciseness.
WD10	10/4/2021	Toby Considine William T Cox David Holmberg	Extended Market Facets. Defined Position and Delivery facets. Made references more consistent. Updated UML model and diagrams.
WD11	10/22/2021	David Holmberg William T Cox Toby Considine	Corrections for clarity. Improved UML diagrams. Flagged requests for comments in Public Review

CSD01	10/29/2021	OASIS TC Administration	Content as in WD11, formatted to include OASIS metadata and references to the published specification
WD12	1/10/2022	William T Cox Toby Considine	Simpler edits in response to comments from PR
WD13		William T Cox Toby Considine	Clarification of Resource/Product/Instrument Removal of references to “Architecture” Responses to “Clarity” tagged issues
WD14	2/22/2022	William T Cox Toby Considine	Clarification of front material Section 1/-2 compared to eliminate duplicative definitions Numerous issues resolutions applied as per Jira
WD15	3/20/2020	William T Cox Toby Considine	Clarity, responses to issues from Review
WD16	4/12/2022	William T Cox Toby Considine	Marketplace and Market characteristics responses to issues Expanded Quotes and Tickers Focus on capitalization
WD17	4/25/2022	William T Cox Toby Considine	Updated UML Market Information added OTC Transactions Edits for Clarity
WD18	9/19/2023	Toby Considine	First response to FIX meetings Changed to Market/Market Segment language Reference FIX Tags when known Closings and Crossings added First pass at FIX-conformant Market Data Reports

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website: [https://www.oasis-open.org/policies-guidelines/ipr/].

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OASIS AND ITS MEMBERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THIS DOCUMENT OR ANY PART THEREOF.

As stated in the OASIS IPR Policy, the following three paragraphs in brackets apply to OASIS Standards Final Deliverable documents (Committee Specifications, OASIS Standards, or Approved Errata).

[OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Standards Final Deliverable, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this deliverable.]

[OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this OASIS Standards Final Deliverable by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this OASIS Standards Final Deliverable. OASIS may include such claims on its website, but disclaims any obligation to do so.]

[OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this OASIS Standards Final Deliverable or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Standards Final Deliverable, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.]

The name "OASIS" is a trademark of OASIS, the owner and developer of this document, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, documents, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark/ for above guidance.