25 January 2024
ESMA12-437499640-2360

Study on data formats and transmission protocols

Assessment study of the suitable data formats and transmission protocols for the purpose of the CTPs and other reporting regimes

Table of Contents

1 Executive Summary

Disclaimer

This document has been prepared for the European Securities and Markets Authority (ESMA) by Accenture NV/SA. It reflects the views only of its authors, and the European Securities and Markets Authority is not liable for any consequence stemming from the reuse of this publication.

Reasons for publication and contents

This final report presents the outcome of the preliminary assessment study conducted in the context of MIFIR review, which aims at relaunching the setup of Consolidated Tape Providers (CTPs).

The study aims at analysing the most suitable data formats and transmission protocols for the purpose of market data collection by the CTPs. In parallel, it intends to provide ESMA with a basis for potential revision of ESMA choice for formats across other datasets already being reported now or in the future. The overall analysis is combining in-house technical expertise with external input provided by CTP ecosystem's stakeholders, like market data contributors, prospective CTP candidates and messaging standard setters. While the regulatory provisions are not yet finalized, the reflection is built on a certain number of working assumptions.

In the course of the investigation, it appears necessary to take some distance from the initially envisioned ISO 20022 framework, which is largely used at ESMA, acknowledging that the trading industry is mostly relying on FIX framework for real time market data feeds. Additionally, it comes clear that the scope of the study is covering two distinct uses cases, which have their own constraints (e.g performance, volume management). In this context, the contemplated reusability criteria is leading to a common ground for both uses cases, which can be seen as not optimal when approaching them individually, yet providing the advantage of harmonization.

With the latter in mind, the report concludes on the recommendation of a single set of solutions driven by the imperative to accommodate the twofold objectives and resulting into generically applicable choices. The two steps approach followed for the assessment culminates in the selection of the following solutions:

- Regarding data format, JSON emerges as the ideal candidate because of its strict adherence to ISO 20022 standard, its well-established flexibility, which both demonstrate its reusability potential and its real time capability. Nonetheless, the latter may be impeded by its higher network overhead caused by its text-based format and may require higher bandwidth capability to cope with increased network traffic. Secondarily, JSON has a lower level of adoption in the context of market data feeds, while its usage is largely spread in an industry agnostic context.

- For transmission protocols, REST is recommended as it leverages the strengths of HTTP/HTTPS while offers a well-structured and standardized approach for efficient and future-proof data reporting. Notably, REST stands out by its simplicity and ease of use as well its high interoperability built upon the usage of HTTPS transport protocol. However, REST APIs operate on a request-response model, which makes it less suitable for instant communication.

At the end, the report paves the way towards a couple of next steps for the pursuit of the reflection initiated in this preliminary assessment. The suggested actions aim at overcoming some of the challenges faced during this study, such as the limited number of ISO 20022 compliant data formats; the absence of clear consensus on the most suitable solutions for CTP purposes; and the difficulty to accommodate the distinct nature of the two uses cases at stake along with asset classes specificities while embracing a single recommendation approach.

2 Introduction

This first chapter is depicting the contextual background in which this preliminary assessment study is taking place. Moreover, it aims at documenting the approach and methodology used for the purpose of the assessment.

2.1 Context

The European Commission legislative proposal for the MIFIR Review aims at relaunching the establishment of real-time Consolidated tape providers (CTP) for shares, ETFs, bonds, and derivatives.

The concept of CTP was formally introduced by MIFID 2 in 2014. CTPs are responsible for collecting from trading venues and approved publication arrangements ('APAs'), market data about financial instruments and consolidating those data into a continuous electronic live data stream, which provides market data per financial instrument. The idea behind the introduction of a CTP was that market data from trading venues and APAs would be made available to the public in a consolidated manner, including all of the Union's trading markets, using identical data tags, formats and user interfaces[1].

To date, no supervised entities has been registered to take over CTP's expected services. Among the various obstacles preventing the emergence of a CTP, ESMA identified the insufficient quality in terms of harmonisation of the data reported by market data contributors to allow for a cost-efficient consolidation[2].

Consequently, the EU regulator intends to remove these obstacles and more precisely ensure the provision of market data in a harmonised format, through a high-quality transmission protocol, and as close to real-time as it is technically possible[3].

2.2 Objective of the study

In the light of above considerations, and as per MIFIR review provision, ESMA will be required to advice on the content and format of data to be submitted to the CTP as well as the quality of the transmission protocol. The latter will be then specified in more detailed legislative measures.

[1] Rec. 3, MIFIR Review proposal
[2] ESMA MiFID II/MiFIR Review Report No. 1 on the development in prices for pre- and post-trade data and on the consolidated tape for equity instruments
[3] Art. 22 a §1, MIFIR Review proposal

For this purpose, it is deemed necessary to thoroughly analyse the various technical solutions available on the market. Therefore, the primary objective is to conduct a preliminary assessment study, which may serve future policy-development on CTP input data. In parallel, the study intends to analyse the possibility to leverage the recommended solution for others reporting obligations arising from EU financial markets regulations.

Consequently, the primary objective of the study is the assessment of the most suitable data formats and transmission protocols (hereafter "the solution") for the purpose of the CTP. Secondly, the outcome of the study should provide a basis for potential revision/upgrade of the regulatory choices for formats across other datasets to the extent of potential converging needs.

2.3 Approach and methodology

The study is taking place between April and June 2023 while the MIFIR review is still in discussion by the co-legislators. The exercise relies on the documentation available at this date, and especially on the MIFIR Review proposal from the European Commission. Trialogue discussions took place at the same time as the study and they are therefore not reflected.

2.3.1 Definition of the scope

The scope is defined upon the combination of two lenses: the "functional scope" describes the business components of the overall CTP functioning to be analysed; and the "technical scope" specifies the observed technical layers.

2.3.1.1 Definition of the functional scope

For the definition of the functional scope, the report refers to the three main data flows envisioned for CTP functioning:

i. Data flows between market data contributors and CTP for consolidating market data

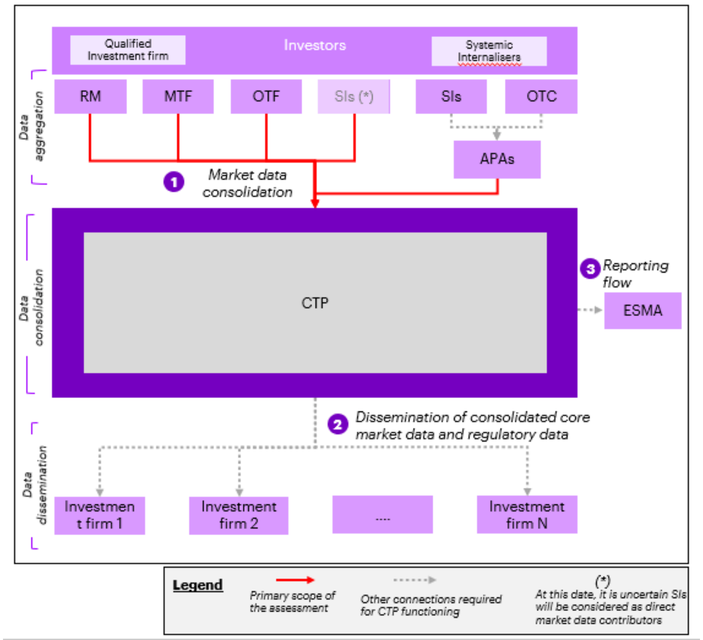
ii. Data flows between CTP and data consumer for disseminating consolidated market data (core and regulatory)

iii. Data flows between CTP and ESMA for MIFID2 transparency reporting obligations

The definition of the scope is based on the specific need of the regulator to define detailed rules on the provision of market data and for ESMA to advise specifically on this matter, as highlighted above.

Regarding the dissemination of consolidated core and regulatory market data, it is understood the selection of the required technical components (e.g. data formats and transmission protocols) belongs to the CTP as long as it meets the regulatory requirements.

FIGURE 1 - CTP PROCESS


Qualified Investment firm
Investors
Systemic Internalisers
Data aggregation
RM
MTF
OTF
SIs (*)
SIs
OTC
APAs
1 Market data consolidation
Data consolidation
CTP
3 Reporting flow
ESMA
2 Dissemination of consolidated core market data and regulatory data
Data dissemination
Investmen t firm 1
Investment firm 2
....
Investment firm N
Legend
Primary scope of the assessment
Other connections required for CTP functioning
(*) At this date, it is uncertain SIs will be considered as direct market data contributors

Lastly with regards to the data reporting flows between CTP and ESMA, the MIFIR review proposal is making no specific changes (Article 22 of MiFIR and RTS 3). It is also assumed that the proven provision of MIFID2 will be applicable without further changes.

Nonetheless, interdependencies should be acknowledged between these three data flows for ensuring the good functioning of the CTP and guarantee an overall consistency.

In scope of the assessment:

- The study will focus on the data flows between the market data contributors and the CTP (cf. connectors in red in figure 1) while managing interdependencies with the two other data flows.

Out of scope of the assessment:

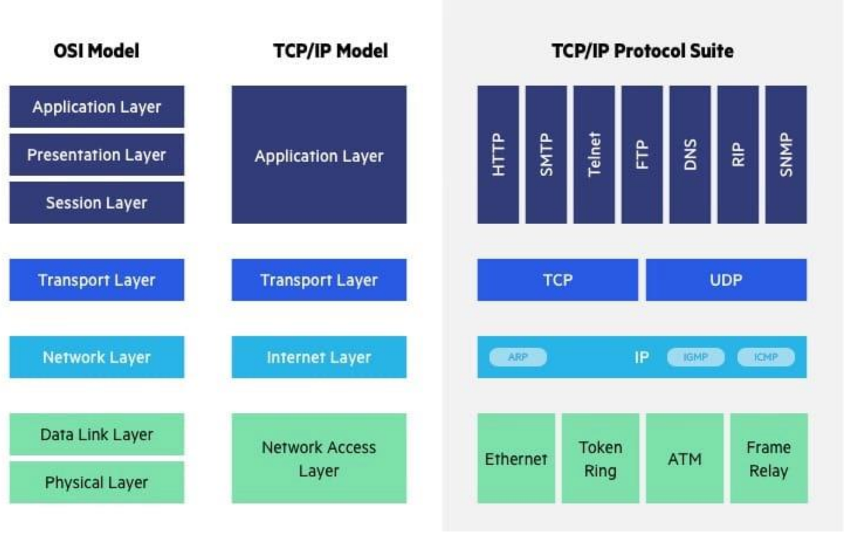
- Neither the data flows from the CTP to data consumers, nor the one from the CTP to ESMA are considered as part of the study.

2.3.1.2 Definition of the technical scope

For the definition of the technical scope, reference is made for data formats to the various standard structures available at the level of the physical layer which supports the logical layer (i.e semantics).

With regards to transmission protocols, it is referred to the TCP/IP model, which divides network communication into four distinct layers.

FIGURE 4 - TCP/IP MODEL



In scope of the assessment:

- **Transmission Protocols:** the scope of the study is limited to the protocols at the Transport Layer (TCP, UDP) and Application Layer (like but not limited to HTTP, FTP, SMTP, SNMP) in the TCP/IP model as these layers are more directly involved in the transmission and processing of application-level data.

- **Message Formats:** The analysis of the data formats, understood as the messaging physical layers, is focusing on the ones that are commonly used and those specific to the financial services industry.

Working assumption – ISO 20022 compatibility

- *It was initially intended to focus on data formats in principle compatible with ISO 20022. Swift presents ISO 20022 method in the light of 3 different and independent layers*[4] *(i.e components) which are:*

- *Conceptual layer: ISO 20022 Business processes & concepts provides a standardized definition of business activities and processes – for example a business process could be "the notification of a date".*
- *Logical layer: ISO 20022 logical messages repository gives standardized descriptions of the information and the required components for performing a specific business activity – in the example of notifying a date, the components would be: a day, a month, and a year.*
- *Physical layer: ISO 20022 describes as physical syntax the standardized format in which the message is represented and structured – in the example, the date would be represented with numerical values and structured in a specific order - e.g 1900-01-01 and then embedded in XML.*

- *ISO 20022 standard natively used XML for physical representation, which was then extended to ASN.1 and JSON. This means that ISO 20022 provides and maintains the specification for transforming the logical messages using the syntax of these data formats. Consequently, XML, ASN.1 and JSON are considered as "ISO 20022 compliant".*

- *Under certain conditions, it is also possible to transform the ISO 20022 message models in another syntax than the XML and ASN.1 described in the ISO 20022 standard. The resulting message formats are not strictly compliant with the current ISO 20022 recipe, but they are granted the label of "ISO 20022 compliant using a domain specific syntax"*[5]*. This label is provided upon the evaluation of a change request by a Standard Evaluation Group (SEG) and approval of the ISO 20022 Registration Management Group (RMG).*

- *On the other hand, the external consultations conducted during the study have demonstrated that the trading industry is largely relying on data formats other than the three ISO 20022 compliant ones. Also, the ISO 20022 investment roadmap clearly states the ambition to expand ISO 20022 compliance to other domain specific formats*[6]*. Acknowledging this*

[4] pp 11-15 Swift, ISO 200022 for dummies, 6th limited edition, John Wiley & Sons Inc., 2022 (link)
[5] S.45, Introduction to ISO 20022 presentation (link)
[6] P.9 Investment Roadmap Frequently Asked Questions (iso20022.org)

is a work in progress, it appears that very little or no information is publicly available on the current status.

- *In the light of above considerations, it has been taken as working assumption that the assessment pool should be kept open to potentially ISO 20022 compatible formats while the evaluation of this criteria will rely on existing information and be limited to ISO 20022 compliant status.*

Out of scope of the assessment:

- **Transmission Protocols:** The protocols in the Internet Layer (like ARP, ICMP) and Network Access Layer (like ethernet, token ring, FDDI, X.25, frame relay) will not be part of the study as these layers are primarily concerned with the routing and delivery of network packets between devices on a network and they are not directly responsible for the handling and processing of application-level data.

- **Message formats:** The semantical and logical layers of messages are not taken in scope of the study. Also data formats that are not commonly used nor specific to the financial services industry are not considered in this study.

2.3.2 Methodology

The overall study is resulting from desk research and external consultations, which were conducted following a structured sequence in four phases.

2.3.2.1 Capture of the assessment input

This first phase aims at apprehending the context of the study and the overall functioning of the CTPs including the main stakeholders involved in the envisioned CTP processes. Concretely, this work package is leading to the definition of the functional and non-functional requirements applicable to CTP market data consolidation, and their translation into assessment criteria (cf. sub-section 2.3.2.3). The outcome of this phase is a jointly reviewed list of both CTP specific requirements and technical standard requirements.

2.3.2.2 Mapping of available solutions

Initiated in parallel with the first phase, this second activity is meant to constitute the assessment pool by establishing an initial list of the most common transmission protocols and data formats for the later assessment.

Next to the desk research, this exercise is completed with external workshops for collecting market practices. The workshops also provided insights on the challenges and opportunities arising with the typical set up of a CTP. For these workshops, three different categories of stakeholders have been mobilized, namely “market data contributors” “prospective CTP candidates” and “messaging standard setting bodies”. The details of external contributors and their input have been anonymized for confidentiality reasons.

2.3.2.3 Assessment of the shortlisted solutions

Based on the outcome of phase 2, this activity aims at evaluating the identified data formats and transmission protocols against the assessment criteria.

This activity is based on available expertise, desk research and external consultation through the release of a questionnaire. This questionnaire is intended to capture the feedback from a larger audience of stakeholders and provide qualitative and quantitative input for the assessment. The audience foreseen for the questionnaire aims on one hand at extending the initial workshop’s audience and on the other hand, at maintaining the focus on CTP’s ecosystem. The details of external contributors and their input have been anonymized for confidentiality reasons.

The assessment follows the weighted sum methodology, which is widely used for multi-criteria analysis. Practically, a list of criteria is established based on standard technical requirements and CTP specific requirements. For the sake of the assessment, the CTP specific requirements have been embedded either in the evaluation criteria applicable to data formats or in those applicable to transmission protocols, or in both of them.

These criteria are weighted based on their level of importance; and numerical values are attributed to the qualitative grade. The assessment consists then at grading the performance of each solution against each criteria translating the requirements.

The following reference tables serve the latter assessment of both data formats and transmission protocols.

FIGURE 6 - REFERENCE TABLE FOR WEIGHTING

Value	Score
Mandatory	10
Recommended	6
Nice to have	2

FIGURE 7 - REFERENCE TABLE FOR SCORING

Value	Score
High	10
Medium	5
Low	1
Yes	10
No	0

2.3.2.4 Recommendations on the most suitable solutions

The fourth and final phase aims at drawing conclusions and at formalizing them into recommendations on the most suitable data formats and transmission protocols. The assumption has been taken that “one size may not fit all” given the different nature of the uses cases at stake (i.e. CTP and ESMA basis for potential upgrade), the varying expectations and technological maturity per asset classes, among others.

Before reaching the final and common recommendation for both CTP’s purpose and other regulatory reporting purposes, it is then important to observe independently how the shortlisted solutions performed in the two distinct contexts.

Ultimately, the conclusion aims at reconciling the two objectives and at providing a common denominator solution. The drafting of the recommendations is not solely relying on the total score provided by the assessment but leverages the findings from the multiple sources being the desk research, working sessions with ESMA CTP project team, external workshops, and questionnaires.

3 Technical assessment

This chapter aims at presenting the outcome of the technical analysis conducted on both data formats and transmission protocols and based on desk research. Precisely, the objective is here to elaborate on the justification of the scores captured in the assessment grid.

3.1 Data formats

This section provides firstly a description of the data formats and secondly an analysis per criteria of the observed data format.

3.1.1 Description of the data formats

The assessment of the data formats has been conducted over 10 different items, which are considered as commonly used especially in the financial services industry.

XML (Extensible Markup Language): XML is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It uses tags to define elements and their hierarchical relationships, allowing the representation of structured data. XML is widely used for data storage, document exchange, and configuration files.

XML has been normalized by the World Wide Web consortium (W3C) in 1998. Originally the W3C was hosted by the Massachusetts Institute of Technology (MIT) in collaboration with CERN and the support of the European Commission. Till recently, the W3C has been administered accordingly to a hosted model involving four hosts, namely: the MIT, the French Institute for research in informatic & automation (INRIA) replaced then by European Research Consortium in Informatics and Mathematics (ERCIM); Keio University of Japan; and Beihang University of China[7]. In January 2023, the W3C registered as a public interest non-profit organization governed by U.S laws[8].

[7] https://www.w3.org/about/history/

[8] Art. 18 Governing law https://www.w3.org/2023/01/Member-Agreement

JSON (JavaScript Object Notation): JSON is a lightweight data interchange format that represents data as key-value pairs, which is both human and machine readable. It is inspired by JavaScript object syntax but is language independent. JSON is often used to transmit data between a server and a web application as an alternative to XML. It is easy to parse, generate, and manipulate, making it popular in web services and APIs.

JSON has been subject to a first normalization in 2013 by the European Computer Manufacturers Association (ECMA) under standard ECMA-404, which has then been reedited in 2017[9] and published by the IETF under RFC 8259[10]. The same year, JSON was also standardized under ISO/IEC standard 21778:2017. Founded in 1960, ECMA recently renamed European association for standardizing information and communication systems, is a Swiss-based non-profit organization in charge of developing international standards for the information and communication industry[11].

CSV (Comma-Separated Values): CSV is a simple file format used to store tabular data, such as spreadsheets or databases. Each line in a CSV file represents a row, and values within each row are separated by commas. CSV files are human-readable and widely supported, but they lack standardized data types and do not support nested structures.

CSV has been documented to a limited extent under RFC 4180[12]. Both W3C and the Internet Engineering Task Force (IETF) attempted to enhance CSV readability and format semantics. The IETF is an international standard setting organization[13] registered as a single member limited liability company of the Internet Society, which is an international non-profit organization registered in the U.S (Washington D.C)[14].

FIXML (Financial Information eXchange Markup Language): FIXML is an XML-based dialect of the FIX protocol, which is widely used in the financial industry for the exchange of electronic messages related to securities trading and market data. FIXML provides a standardized schema and rules for message formatting, allowing reliable and interoperable communication between financial systems.

[9] ECMA-404 - Ecma International (ecma-international.org)
[10] RFC 8259: The JavaScript Object Notation (JSON) Data Interchange Format (rfc-editor.org)
[11] History - Ecma International (ecma-international.org)
[12] RFC 4180 - Common Format and MIME Type for Comma-Separated Values (CSV) Files (ietf.org)
[13] IETF | Internet Engineering Task Force
[14] p.11, IETF 2022 audited financial statements

FIXML originally authored in 1992 is maintained by the FIX Trading Community. The latter is defined as a non-profit, independent and neutral industry-driven standards body at the heart of global trading. FIX Trading Community™ is a brand of FIX Protocol Ltd, which is registered in UK as a non-profit organization[15].

ASN.1 (Abstract Syntax Notation One): ASN.1 is a formal language and notation used to describe data structures and protocol messages. It provides a platform-independent and extensible approach to define complex data structures, allowing interoperability between different systems. ASN.1 is commonly used in telecommunications and network protocols.

ASN.1 is a standard developed by the Telecommunication standardization sector of the International Telecommunication Union (ITU-T). The ITU is an agency of the United Nations which is heardquartered in Switzerland[16].

Protocol Buffer: Protocol Buffers, also known as Protobuf, is a language-agnostic binary serialization format developed by Google. It enables efficient and extensible communication between different processes or systems. Protocol Buffers use a language-specific schema definition to define the structure of data, which is then compiled into code for various programming languages.

Initially elaborated for Google internal use, Protocol Buffer is open sourced since 2008. Google supports the open source community with regular technical updates[17]. Headquartered in California, Google is a limited liability company and subsidiary of Alphabet conglomerate, a U.S listed company[18].

BSON (Binary JSON): BSON is a binary representation of JSON-like documents used primarily in MongoDB, a popular NoSQL database. BSON extends JSON with additional data types and features, such as support for binary data, dates, and efficient indexing. It provides a more compact representation and faster encoding/decoding compared to plain text JSON.

[15] Corporate Governance • FIX Trading Community
[16] ITU Telecommunication Standardization Sector
[17] https://protobuf.dev/overview/
[18] 2022-alphabet-annual-report.pdf (abc.xyz)

Originally developed by MongoDB, BSON can be used independently outside of MongoDB relying on a variety of languages. Established in 2007, MongoDB Inc. is a US registered public company listed on Nasdaq Global Market[19].

FAST (FIX Adapted for Streaming): FAST is a binary messaging protocol that optimizes the transmission of financial data in high-performance systems. It is an adaptation of the FIX protocol designed for low-latency and high-throughput applications, such as high-frequency trading. FAST uses binary encoding and various compression techniques to achieve efficient message transmission.

FAST has been initially developed by FIX trading community's market data optimization working group in 2004 for addressing the challenge of the ever-increasing volumes of market data causing delays in trading operations. FIX Trading community oversees the standardization and maintenance of the format[20].

SBE (Simple Binary Encoding): SBE is a high-performance binary encoding format designed for low-latency messaging systems. It offers a compact and efficient representation of structured data, reducing processing overhead and improving message throughput. SBE provides a schema definition language to describe message structures, allowing for standardized and reliable communication in high-throughput environments.

SBE has been designed by FIX trading community's High performance working group and has been firstly released in 2016. It aims at completing FAST format capabilities (developed in 2005) by providing compact encoding[21].

FIX Tag Value: FIX Tag Value is a text-based format used in the FIX protocol for the exchange of financial messages. FIX is a widely adopted protocol in the financial industry, and Tag Value is one of its encoding variants. It represents messages as a sequence of tags and values, where each tag uniquely identifies a specific data field. FIX Tag Value messages have a strict structure and are used for reliable communication between financial systems.

[19] MongoDB: The Developer Data Platform | MongoDB
[20] FAST Protocol • FIX Trading Community
[21] Simple Binary Encoding (SBE) • FIX Trading Community

FIX Tag value is the original encoding used for FIX messages[22]. In April 2022, ISO published the normative specifications under ISO 3531-1-2022 standard[23].

3.1.2 Assessment of the data formats

The assessment of the data formats has been performed against 19 criteria featuring both standard technical and CTP specific requirements.

3.1.2.1 Reliability

The reliability of a data format refers to its ability to ensure that data is accurately transmitted and received without corruption or loss. This includes features such as error correction and detection mechanisms, as well as recovery mechanisms in case of transmission errors or failures.

To which extent does the solution provide reliability?

Data Format	Grade	Justification
XML	Medium	XML provides error-checking and validation capabilities through tools like Document Type Definitions (DTD) and XML Schema. However, XML is relatively verbose, which increases the chances of human error in data entry or transmission. Additionally, XML lacks standardized data types, making it prone to inconsistencies and requiring additional effort for data integrity checks.
JSON	High	JSON provides high reliability due to its simplicity and widespread support across programming languages. JSON syntax is easy to read and write, reducing the chances of human error. It also offers built-in data types and a straightforward key-value structure, enabling effective data representation and validation. JSON's popularity and

[22] FIX TagValue Encoding • FIX Trading Community

[23] ISO 3531-1:2022(en), Financial services — Financial information eXchange session layer — Part 1: FIX tagvalue encoding

		extensive usage contribute to the availability of reliable parsing and error-handling libraries, enhancing its overall reliability.
CSV	Low	CSV has relatively low reliability compared to other formats. While it is widely supported and easy to generate, CSV lacks standardized data types and does not support nested structures or complex relationships. These limitations make CSV susceptible to data integrity issues, such as incorrect data types or inconsistent values. CSV's reliance on manual handling and interpretation can introduce errors, reducing its overall reliability.
FIXML	High	FIXML is designed specifically for financial messaging and offers high reliability. It provides a standardized schema and strict validation rules, ensuring consistency and accuracy in financial systems. FIXML's focus on financial applications and its adherence to the FIX protocol enhance its reliability.
ASN.1	High	ASN.1 is known for its high reliability in telecommunications and network protocols. It offers a flexible and extensible way to represent complex data structures. ASN.1's strong type checking and error detection mechanisms contribute to its reliability, ensuring data consistency and integrity. Additionally, ASN.1's widespread usage and extensive standards enhance its reliability through well-established implementations and interoperability.
Protocol Buffer	High	Protocol Buffers provide high reliability due to their efficient binary encoding and strong data validation capabilities. Protocol Buffers offer type safety and schema evolution, ensuring consistent and reliable data representation. The binary encoding reduces the risk of errors during serialization and deserialization.
BSON	High	BSON as a binary representation of JSON-like documents improves upon JSON by providing more compact storage and faster encoding/decoding. The binary format reduces the chance of data corruption during transmission or storage compared to text-based formats. BSON also includes features like support for different data types, object references, and queries, which can enhance data reliability in specific use cases.
FAST	High	FAST is designed for high-performance financial systems and its binary messaging protocol provides efficient encoding/decoding,

		ensuring reliable and low-latency message transmission. FAST's focus on speed and efficiency, along with its usage in high-frequency trading, enhances its reliability.
SBE	High	SBE is optimized for high-performance messaging systems, offering high reliability. It provides efficient binary encoding and decoding of structured data with a focus on low latency. SBE's strict schema definition and compact binary representation contribute to its reliability.
FIX Tag Value	Medium	FIX Tag Value offers moderate reliability as a text-based format used in the FIX protocol for financial messaging. It provides a standardized message structure and syntax, contributing to reliability in financial systems. However, FIX Tag Value is still prone to human error in data entry and lacks the efficiency of binary formats.

3.1.2.2 Ease of use

Ease of use refers to how user-friendly and easy to understand the data format is for both developers and end-users. This includes features such as clear and concise syntax, well-defined data structures, and ease of implementation.

To which extent is the solution providing ease of use?

Data Format	**Grade**	**Justification**
XML	Medium	XML uses tags and a hierarchical structure, which can be intuitive for representing structured data. However, XML can be verbose, requiring a larger amount of code or configuration compared to other formats. Additionally, XML's flexibility allows for different schema definitions, which may introduce complexity. XML parsers and libraries are widely available, making it relatively easy to parse and manipulate XML data.
JSON	High	JSON is simple and concise syntax makes it easy to read, write, and understand. JSON's key-value structure resembles popular programming language data structures, making it intuitive for

		developers. The availability of JSON libraries and tools for parsing and serialization further contributes to its ease of use.
CSV	High	CSV provides high ease of use due to its simplicity. Its plain text format is human-readable and easy to generate or edit using common spreadsheet software. CSV lacks complex syntax and schema definitions, reducing the learning curve for users.
FIXML	Low	FIXML follows the XML syntax, which can introduce complexity and verbosity. FIXML is designed for the specific purpose of financial messaging, which requires a deep understanding of the FIX protocol and its intricacies. Working with FIXML may involve managing a large number of tags and adhering to strict message standards. The learning curve for understanding FIXML and implementing it correctly can be steep, making it less user-friendly for beginners.
ASN.1	Low	ASN.1 has a low ease of use due to its complexity and formal nature. Defining data structures and messages in ASN.1 requires a thorough understanding of the ASN.1 specification and syntax. ASN.1 uses a notation that is less familiar to developers compared to other formats, making it more challenging to work with. Implementations and tools for working with ASN.1 may require specialized knowledge, which can increase the learning curve and make it less accessible to novice users.
Protocol Buffer	Medium	While Protocol Buffers require a schema definition, they offer automatic code generation for various programming languages, which simplifies data manipulation. Protocol Buffers' strict typing and versioning support contribute to ease of use by ensuring data consistency and evolution. However, using Protocol Buffers effectively may still require understanding the schema definition language and the associated tooling.
BSON	Medium	BSON extends the JSON format and retains its simplicity, making it familiar to developers already familiar with JSON. BSON adds support for additional data types and features, but these enhancements can introduce some complexity. BSON libraries and tools are available in various programming languages, facilitating parsing and serialization tasks.

FAST	Low	FAST has a relatively low ease of use due to its specialization in high-performance financial messaging. Implementing FAST requires understanding its binary encoding, compression techniques, and optimization strategies. Working with FAST messages may involve handling low-level binary data, which can be complex and error prone. Developing or integrating FAST-based solutions may require specialized knowledge and the use of dedicated libraries or frameworks.
SBE	Low	SBE requires the definition of message schemas using its DSL (Domain-Specific Language) or XML. Working with SBE involves generating code based on these schemas, which may require familiarity with the SBE toolchain and its concepts. While SBE provides efficient binary encoding and decoding, using it effectively requires understanding its intricacies, making it less user-friendly for those unfamiliar with SBE's domain.
FIX Tag Value	Medium	FIX Tag Value utilizes a textual representation for financial messages and follows a specific syntax and message structure defined by the FIX protocol. Working with FIX Tag Value requires knowledge of the FIX protocol's standards and conventions. While FIX Tag Value is relatively straightforward, understanding and properly constructing FIX messages may require referring to the FIX specification documentation and managing a large set of FIX tags. The availability of libraries and tooling can simplify parsing and validation tasks.

3.1.2.3 Encryption

Encryption refers to the ability of the data format to protect data by encrypting it to prevent unauthorized access or tampering. This includes features such as data encryption algorithms and key management mechanisms.

Does the solution support encryption mechanism?

Data Format	Grade	Justification
XML	Yes	XML messages can be encrypted using XML Encryption, which is a standard specification for encrypting XML data. It provides mechanisms

		for encrypting specific elements or entire XML documents, supporting both symmetric and asymmetric encryption algorithms.
JSON	Yes	While JSON does not have native encryption capabilities, it is possible to encrypt JSON messages. External encryption libraries or frameworks can be used for this purpose.
CSV	Yes	Although CSV is a plain-text format, it is possible to encrypt CSV messages. Though encryption is not natively supported, it can be achieved using a third-party program.
FIXML	Yes	FIXML messages can be encrypted using standard encryption techniques. By leveraging XML Encryption or other encryption mechanisms, FIXML data can be encrypted at the XML level, providing confidentiality and security for the messages during transmission or storage.
ASN.1	Yes	ASN.1 messages can be encrypted by applying encryption techniques at a higher level, such as within the application or transport layer. By utilizing appropriate encryption algorithms and protocols, the confidentiality and integrity of ASN.1 data can be ensured.
Protocol Buffer	Yes	While Protocol Buffers do not have native encryption capabilities, it is possible to encrypt Protocol Buffer messages. Encryption can be applied at a higher level, such as encrypting the serialized Protocol Buffer data using symmetric or asymmetric encryption algorithms or by integrating with external encryption libraries or frameworks.
BSON	Yes	BSON messages can be encrypted by applying encryption techniques to the BSON data. External encryption libraries or frameworks can be used to achieve encryption.
FAST	Yes	FAST messages can be encrypted by applying encryption techniques to the BSON data. External encryption libraries or frameworks can be used to achieve encryption.
SBE	Yes	SBE messages can be encrypted by applying encryption techniques to the BSON data. External encryption libraries or frameworks can be used to achieve encryption.

FIX Tag Value	Yes	FIX Tag Value messages can be encrypted by applying encryption techniques to the BSON data. External encryption libraries or frameworks can be used to achieve encryption.

3.1.2.4 Digital signature

Digital signature refers to the ability of the data format to provide authentication and integrity of the data by using cryptographic signatures. This includes features such as digital signature algorithms and key management mechanisms.

Does the solution support digital signature?

Data Format	Grade	Justification
XML	Yes	XML provides native support for digital signatures through XML Signature Syntax and Processing. XML signatures enable the verification of the authenticity, integrity, and non-repudiation of XML data. The XML Signature standard defines the structure and syntax for including digital signatures within XML documents.
JSON	No	JSON does not provide built-in support for digital signatures. JSON is primarily a data interchange format and does not have native mechanisms or standards specifically designed for digital signatures. Implementing digital signatures in JSON requires external libraries or custom solutions.
CSV	No	CSV does not provide built-in support for digital signatures. CSV is a plain text format without native mechanisms for incorporating digital signatures.
FIXML	Yes	FIXML supports digital signatures as part of the broader FIX protocol's security measures. FIXML messages can be digitally signed to ensure message integrity, authentication, and non-repudiation.
ASN.1	Yes	ASN.1-based systems can incorporate digital signatures by defining specific data structures within the schema to accommodate signature data. Digital signature mechanisms, such as RSA or ECDSA, can be

		integrated into ASN.1-based systems to provide authentication, integrity, and non-repudiation.
Protocol Buffer	No	Protocol Buffer does not include specific features or standards for digital signatures. Including digital signatures in Protocol Buffer data requires custom implementation, such as adding signature fields or integrating external libraries for generating and verifying signatures.
BSON	No	BSON does not include built-in mechanisms for incorporating digital signatures. To include digital signatures in BSON data, it requires custom implementations using external libraries or frameworks for generating and validating signatures.
FAST	No	FAST does not include specific features or standards for digital signatures. If digital signatures are required in FAST messages, it requires custom implementation by adding signature fields and integrating with external libraries or frameworks for generating and validating signatures.
SBE	No	SBE does not include specific mechanisms or standards for digital signatures. If digital signatures are required in SBE-encoded data, it requires custom implementation by adding signature fields and integrating with external libraries or frameworks for generating and validating signatures.
FIX Tag Value	Yes	FIX Tag Value supports digital signatures as part of the broader FIX protocol's security measures. FIX Tag Value messages can be digitally signed to ensure message integrity, authentication, and non-repudiation. The FIX protocol includes specifications for incorporating digital signatures, providing support for secure message transmission.

3.1.2.5 Technical data validation

Technical data validation refers to the ability of the data format to validate data before it is transmitted to ensure that it meets certain criteria or conforms to a specified schema. This includes features such as data validation rules, data type validation, and schema validation.

Does the solution support data validation?

Data Format	Grade	Justification
XML	Yes	XML provides support for data validation through Document Type Definitions (DTD) or XML Schema Definition (XSD). These validation mechanisms allow defining rules and constraints for the structure, data types, and relationships within an XML document. The availability of XML validation tools and libraries further enhances the validation support for XML data.
JSON	Yes	JSON has support for data validation through JSON Schema. JSON Schema is a specification that defines the structure, format, and constraints for JSON documents. JSON Schema validators and libraries are available in various programming languages, enabling the validation of JSON data for data integrity and consistency.
CSV	No	CSV does not have inherent support for data validation. CSV files primarily represent tabular data without predefined rules or structure.
FIXML	Yes	FIXML supports data validation through the Financial Information eXchange (FIX) protocol. The FIX protocol defines a standardized set of message types and fields used in the financial industry. FIXML messages adhere to the rules and specifications of the FIX protocol, which includes data validation as part of the protocol's requirements.
ASN.1	Yes	ASN.1 provides robust support for data validation through its powerful and comprehensive schema language. ASN.1 compilers generate code that can perform automatic data validation against the specified schema, ensuring the integrity and correctness of the transmitted or stored data.
Protocol Buffer	Yes	Protocol Buffers support data validation through protocol buffer message definitions. Protocol Buffers allow developers to define message schemas using a language-agnostic interface definition language (IDL). The schema definitions can include validation rules, data types, and constraints. Generated protocol buffer code provides built-in validation methods that validate the integrity and conformance of data against the defined schema, ensuring data consistency and correctness in protocol buffer-based systems.

BSON	No	BSON does not provide native support for data validation. BSON primarily extends the JSON data model with additional data types and is used in MongoDB for efficient storage and retrieval of JSON-like documents. While MongoDB can enforce some level of validation through its collection schema and document validation features, BSON itself does not include built-in mechanisms for defining or enforcing validation rules or constraints. Data validation in BSON is typically handled at the application or database layer.
FAST	No	FAST does not include native support for data validation. While FAST focuses on efficient encoding and decoding of messages, it does not provide inherent mechanisms for data validation. Data validation in FAST-based systems typically relies on external validation logic implemented within the application or business domain.
SBE	Yes	SBE (Simple Binary Encoding) provides support for data validation through its schema definition language. SBE allows developers to define message schemas using a concise and expressive syntax. The SBE schema includes rules, data types, and constraints that can be used for data validation. Generated code based on the SBE schema provides validation methods that ensure the integrity and compliance of data against the defined schema, enhancing the reliability and correctness of data processing in SBE-based systems.
FIX Tag Value	Yes	FIX Tag Value supports data validation as defined by the FIX protocol. FIX Tag Value messages adhere to the specifications and rules of the FIX protocol, which includes data validation as an essential aspect. FIX Tag Value parsers and libraries often provide validation capabilities to ensure compliance with the FIX protocol, ensuring the accuracy and integrity of data in FIX Tag Value messages.

3.1.2.6 Encoding

Encoding refers to how the data is represented in binary form, and the ability of the data format to efficiently encode and decode data for transmission. This includes features such as variable-length encoding, fixed-length encoding, and character encoding.

Does the solution support encoding?

Data Format	Grade	Justification
XML	Yes	XML supports various encoding schemes, including UTF-8, UTF-16, and others. These encoding schemes allow the representation of characters from different languages and character sets, ensuring compatibility and interoperability across systems. XML parsers and processors are designed to handle different encodings, allowing data to be correctly interpreted and processed regardless of the encoding used.
JSON	Yes	JSON supports Unicode encoding, primarily using UTF-8, which allows the representation of characters from different languages and character sets. UTF-8 is the recommended encoding for JSON data and is widely supported by JSON libraries and parsers.
CSV	No	CSV itself does not define an encoding scheme. The encoding of the actual CSV data depends on the encoding used for the text file, such as UTF-8, UTF-16, or others. However, CSV as a format does not enforce or provide specific support for encoding.
FIXML	Yes	FIXML, being based on XML, inherits the encoding support provided by XML. It supports various encoding schemes, including UTF-8, UTF-16, and others. FIXML data can be encoded in different character encodings to handle different language requirements and character sets.
ASN.1	Yes	ASN.1 supports encoding using various encoding rules such as Basic Encoding Rules (BER), Canonical Encoding Rules (CER), Distinguished Encoding Rules (DER), and others.
Protocol Buffer	Yes	Protocol Buffers support a binary encoding format that is compact and efficient for serialization and transmission. The binary encoding of Protocol Buffers allows for fast serialization and deserialization, making it well-suited for high-performance and resource-constrained environments.
BSON	Yes	BSON supports binary encoding for representing JSON-like documents. BSON's binary encoding ensures that data can be

		serialized and deserialized efficiently, contributing to the performance and scalability of MongoDB databases.
FAST	Yes	FAST, as a binary protocol, inherently supports binary encoding. It provides efficient binary representations for various data types, enabling fast encoding and decoding.
SBE	Yes	SBE (Simple Binary Encoding) is specifically designed for binary encoding of structured data. SBE encoding optimizes the use of bits and bytes, resulting in highly efficient data serialization and transmission.
FIX Tag Value	No	FIX Tag Value does not specify or enforce a particular encoding scheme. The encoding of FIX Tag Value messages depends on the transport layer or wire protocol used to transmit the data. Commonly, FIX Tag Value messages are encoded using character-based encodings such as ASCII or UTF-8, but the choice of encoding is not mandated by the FIX Tag Value format itself.

3.1.2.7 Support complex data structures

Support for complex data structures refers to the ability of the data format to represent and transmit complex data structures such as nested arrays, objects, and graphs.

Does the solution support complex data structures?

Data Format	Grade	Justification
XML	Yes	XML provides support for complex data structures through its hierarchical tree-like structure. XML allows the nesting of elements and the creation of parent-child relationships, enabling the representation of complex data models. The use of XML namespaces and attributes further enhances the flexibility and expressiveness of XML for representing complex structures.

JSON	Yes	JSON supports complex data structures through its hierarchical and nested object notation. JSON allows the representation of complex data models by nesting objects, arrays, and key-value pairs.
CSV	No	CSV is a simple tabular format that does not inherently support complex data structures. CSV primarily represents flat, two-dimensional data with rows and columns. While CSV can store data with some level of hierarchy or relationships through additional fields or delimiters, it does not provide explicit support for complex data structures.
FIXML	Yes	FIXML supports complex data structures through its XML-based format. FIXML defines various message types and fields that allow the representation of complex financial transactions and instruments. The FIXML format supports nesting of elements, attributes, and namespaces, enabling the representation of complex relationships and structures.
ASN.1	Yes	ASN.1 provides extensive support for complex data structures. ASN.1 supports the creation of nested structures, sequences, choice types, and recursive data structures. The versatility of ASN.1 enables the representation of complex data relationships and dependencies.
Protocol Buffer	Yes	Protocol Buffers support complex data structures through their message definitions. The schema definitions can include nested message types, repeated fields, and custom data types, enabling the representation of complex data structures.
BSON	Yes	BSON supports complex data structures by extending the JSON data model. BSON allows the nesting of documents, arrays, and key-value pairs, providing a hierarchical structure similar to JSON. BSON's support for additional data types and features, such as dates, binary data, and references, further enhances its capability to handle complex data structures.
FAST	Yes	FAST supports complex data structures through its data modelling capabilities. FAST allows to define complex data models using field templates, groups, and repeating groups. These constructs enable the representation of complex relationships and hierarchical structures.

SBE	Yes	SBE provides support for complex data structures through its schema definition language. SBE schema language supports complex field types, groups, and repeating groups, enabling the representation of complex data structures.
FIX Tag Value	Yes	FIX Tag Value supports complex data structures through its use within the FIX (Financial Information eXchange) protocol. FIX Tag Value messages consist of a sequence of tagged fields, which can be repeated and nested to represent complex relationships. The well-defined FIX protocol specification defines the structure and semantics of complex data within FIX Tag Value messages.

3.1.2.8 Support metadata

Support for metadata refers to the ability of the data format to include additional information about the data being transmitted, such as data type, version, author, etc.

Does the solution support metadata?

Data Format	Grade	Justification
XML	Yes	XML supports metadata through the use of attributes, namespaces, and additional elements. Attributes in XML provide a way to add metadata to XML elements, allowing for the inclusion of additional information about the data. Namespaces enable the organization and categorization of XML elements and attributes, providing metadata about the XML structure. Additionally, XML allows the inclusion of dedicated metadata elements within the document to describe the content, structure, or purpose of the data.
JSON	Yes	JSON supports metadata through the inclusion of additional key-value pairs within JSON objects. Developers can include custom key-value pairs to represent metadata associated with the JSON data. These metadata entries can provide information about the structure, semantics, or any other relevant details of the JSON document.
CSV	No	CSV is a simple, flat file format that does not have native support for metadata. CSV primarily represents tabular data without built-in provisions for including metadata. However, developers can apply

		conventions or incorporate additional columns or rows to represent metadata within a CSV file.
FIXML	Yes	FIXML provides support for metadata through the use of attributes, namespaces, and additional elements within the XML-based format. Attributes can be used to add metadata to FIXML elements, providing additional information about the data. Namespaces enable the categorization and organization of FIXML elements and attributes, allowing for the inclusion of metadata related to the XML structure.
ASN.1	Yes	ASN.1 supports metadata through the use of annotations and constraints within the schema definition. Developers can add annotations to ASN.1 types, values, and modules to include metadata about the data model or specific elements. These annotations provide additional information, such as semantic descriptions, usage guidelines, or cross-references. Constraints defined in ASN.1 also serve as metadata by specifying limitations, conditions, or requirements on the data.
Protocol Buffer	Yes	Protocol Buffers support metadata through the use of custom options within the schema definition. Developers can define custom options to include metadata about the data model or specific elements. These options provide additional information, such as field descriptions, semantic meanings, or versioning details.
BSON	Yes	BSON supports metadata through the inclusion of additional key-value pairs within BSON documents. Similar to JSON, BSON allows developers to include custom key-value pairs to represent metadata associated with the data. These metadata entries can provide information about the structure, semantics, or any other relevant details of the BSON document.
FAST	Yes	FAST supports metadata through the use of template attributes and context fields. Template attributes can be used to define and include metadata within the FAST message templates. These attributes provide additional information about the data fields, such as semantic meanings, descriptions, or data formatting rules. Context fields in FAST messages can also be used to carry metadata that provides contextual information about the data.

SBE	Yes	SBE supports metadata through the use of the presence of optional fields within the schema definition. By defining optional fields, developers can include metadata within the SBE messages. These optional fields can carry additional information about the data, such as versioning details, semantic meanings, or encoding options.
FIX Tag Value	Yes	FIX Tag Value supports metadata through the use of FIX protocol-defined fields and attributes. The FIX protocol specifies various fields that can carry metadata about the financial data being transmitted. These fields include information such as message types, security identifiers, timestamps, and more.

3.1.2.9 Support nested data

Support for nested data refers to the ability of the data format to represent and transmit data that contains hierarchical or nested structures.

Does the solution support nested data?

Data Format	Grade	Justification
XML	Yes	XML supports nested data structures through its hierarchical nature. XML allows the nesting of elements within elements, enabling the representation of complex and nested data structures. This hierarchical structure makes XML well-suited for representing and exchanging data with multiple levels of nesting, such as tree-like structures or parent-child relationships.
JSON	Yes	JSON supports nested data structures natively. JSON allows the nesting of objects within objects and arrays within arrays, enabling the representation of complex and hierarchical data structures. This nesting capability makes JSON well-suited for representing and exchanging nested data, such as hierarchical relationships or nested collections of values.
CSV	No	CSV does not have native support for representing nested data structures. CSV is a flat file format that primarily represents tabular

		data. There is no built-in mechanism to represent nested or hierarchical data structures.
FIXML	Yes	FIXML provides support for nested data structures through its XML-based format. XML's inherent support for nesting elements allows FIXML to represent and exchange complex and hierarchical financial data. FIXML allows the nesting of repeating groups and components within the message structure, enabling the representation of nested relationships and complex financial instruments.
ASN.1	Yes	ASN.1 supports nested data structures through its definition language. ASN.1 allows to define complex data types, including nested structures, using modules and type definitions. The hierarchical nature of ASN.1 notation enables the representation of nested data structures and complex relationships between data elements
Protocol Buffer	Yes	Protocol Buffers support nested data structures through their message definition language. Developers can define message types that include nested fields, allowing for the representation of hierarchical and nested data structures. Protocol Buffers provide a concise and expressive syntax to define complex data models, including nested relationships and repeated fields.
BSON	Yes	BSON supports nested data structures natively. BSON allows the nesting of documents within documents, arrays within arrays, and a combination of both, allowing for the representation of complex and hierarchical data structures. This support for nesting enables BSON to handle nested data relationships, such as parent-child or nested collections of values.
FAST	Yes	FAST supports nested data structures through the use of templates and template references. Templates in FAST define the structure and data types of messages, including the support for nesting. Developers can define complex templates with nested fields and repeating groups, allowing for the representation of hierarchical and nested data structures.
SBE	Yes	SBE supports nested data structures through its message schema definition. SBE allows developers to define composite data types and nested structures using its schema language. This includes the

		support for nesting fields within other fields, enabling the representation of hierarchical and nested data structures.
FIX Tag Value	Yes	FIX provides mechanisms to represent nested data structures through the use of repeating groups and components. FIX allows for the definition of repeating groups, which enable the representation of nested data structures within a single tag. FIX also supports the use of components, which are reusable blocks of fields that can be included within messages. Components provide a way to define and reuse nested structures across multiple messages.

3.1.2.10 Support inline documentation

Support for inline documentation refers to the ability of the data format to include comments or documentation within the data itself, to make it easier to understand and maintain.

Does the solution support inline documentation?

Data Format	Grade	Justification
XML	Yes	XML supports inline documentation through XML comments. Comments can be included within the XML tags to provide explanations or additional information about the data elements and their usage.
JSON	No	JSON does not have built-in support for inline documentation. However, you can include documentation as a separate field within the JSON structure, but it is not inherent to the format itself.
CSV	No	CSV does not have native support for inline documentation. It is a simple data format without any provision for including comments or documentation within the data itself.
FIXML	Yes	FIXML supports inline documentation through XML comments. It uses XML syntax, allowing you to include comments within the XML tags.

ASN.1	No	ASN.1 does not inherently support inline documentation. It is a binary encoding format used to describe data structures, but it lacks a dedicated mechanism for documentation.
Protocol Buffer	No	Protocol Buffers do not provide native support for inline documentation. However, Protocol Buffer schema files (in .proto format) allow adding comments for documentation purposes, but they are separate from the encoded data.
BSON	No	BSON does not have inherent support for inline documentation. It is a binary representation of JSON-like documents and does not provide a specific mechanism for including comments or documentation.
FAST	No	FAST is a binary protocol and does not have native support for inline documentation. It focuses on efficiency and speed, sacrificing additional features such as inline documentation.
SBE	No	SBE is a binary protocol and does not natively support inline documentation. It prioritizes compactness and performance, thus lacking built-in mechanisms for including documentation.
FIX Tag Value	Yes	FIX Tag Value format supports inline documentation through FIX comments. Comments can be included within the FIX messages using a specific tag (e.g., 58=). This allows adding explanations or notes about the data.

3.1.2.11 Parsing speed

Parsing speed refers to how quickly the data format can be parsed and converted to a usable format, and how efficiently it can be processed by the receiving application.

To which extent does the solution provide parsing speed?

Data Format	Grade	Justification
XML	Medium	XML parsing speed is generally slower compared to other formats due to its complex and verbose nature. XML parsing requires traversing the document tree and handling various XML features like namespaces, attributes, and nested elements.

JSON	High	JSON parsing is generally fast due to its simple and straightforward syntax. JSON can be parsed efficiently using libraries that are optimized for JSON processing. JSON's lack of complex features, such as namespaces, contributes to its parsing speed.
CSV	High	CSV parsing is usually fast because of its simplistic structure. CSV files consist of plain text with a simple delimiter (e.g., comma or tab), allowing for efficient parsing by reading the data row by row without the need for extensive processing.
FIXML	Medium	FIXML parsing speed can vary depending on the complexity of the FIXML schema and the implementation. FIXML is based on XML, so it shares similar characteristics and potential parsing challenges, such as traversing the document structure and handling XML features.
ASN.1	High	ASN.1 parsing speed can be high, thanks to its compact binary encoding. ASN.1 uses a predefined schema, allowing for efficient parsing by directly decoding the binary data according to the schema definition, without the need for complex tree traversal.
Protocol Buffer	High	Protocol Buffer parsing is generally fast due to its efficient binary encoding. Protocol Buffers use a compact binary format, enabling direct decoding based on the defined schema. This eliminates the need for extensive parsing and results in high parsing speed.
BSON	High	BSON is designed for efficient parsing and processing. Its binary structure allows for direct access to data elements, resulting in faster parsing compared to JSON. BSON's compact size and optimized encoding further contribute to its superior parsing speed.
FAST	High	FAST parsing speed is typically high due to its optimized binary format. FAST is designed for high-performance, low-latency scenarios, prioritizing parsing speed. Its binary encoding and lack of complex features contribute to its fast-parsing capabilities.
SBE	High	SBE parsing speed is generally high because of its binary encoding and focus on performance. SBE uses a compact binary format, allowing for direct decoding based on the predefined schema. This results in efficient and fast parsing operations.

FIX Tag Value	Medium	FIX Tag Value parsing speed can vary depending on the implementation. While it uses a simple text-based format, parsing efficiency can be influenced by factors such as the number of tags and the complexity of handling tag/value pairs and repeating groups.

3.1.2.12 Serialization speed

Serialization speed refers to how quickly the data format can be converted to binary form for transmission, and how efficiently it can be transmitted over a network.

To which extent does the solution provide serialization speed?

Data Format	Grade	Justification
XML	Medium	XML serialization speed is typically slower compared to other formats due to its verbose nature and the need to convert structured data into text-based XML markup. The process involves converting data elements into XML tags, handling namespaces, attributes, and other XML features, which adds complexity and impacts serialization speed.
JSON	High	JSON serialization is generally fast due to its simple and straightforward structure. JSON can be serialized efficiently using libraries that are optimized for JSON processing. The serialization process involves converting data into a JSON string representation, which is relatively straightforward and results in high serialization speed.
CSV	High	CSV serialization speed is typically high due to its simplistic structure. CSV files consist of plain text with a simple delimiter (e.g., comma or tab), allowing for efficient serialization by concatenating data elements with the delimiter and writing them to the file. CSV serialization does not require complex encoding or conversion processes, resulting in high serialization speed.
FIXML	Medium	FIXML serialization speed can vary depending on the complexity of the FIXML schema and the implementation. FIXML is based on XML, so the serialization process involves converting structured data into XML markup. This includes adding XML tags, handling namespaces,

		attributes, and other XML features, which can impact serialization speed compared to simpler formats.
ASN.1	High	ASN.1 serialization speed can be high due to its compact binary encoding. ASN.1 uses a predefined schema and efficient binary encoding, allowing for direct serialization of data according to the schema definition. This eliminates the need for complex encoding processes, resulting in high serialization speed.
Protocol Buffer	High	Protocol Buffer serialization is generally fast due to its efficient binary encoding. Protocol Buffers use a compact binary format, enabling direct serialization based on the defined schema. This eliminates the need for extensive processing or conversion steps, resulting in high serialization speed.
BSON	High	BSON binary format allows for efficient serialization of data, as it directly maps to the internal representation of the data. The binary structure and optimized encoding of BSON contribute to its high serialization speed.
FAST	High	FAST serialization speed is typically high due to its optimized binary format. FAST is designed for high-performance, low-latency scenarios, prioritizing serialization speed. Its binary encoding and lack of complex features contribute to efficient and fast serialization operations.
SBE	High	SBE serialization speed is generally high due to its binary encoding and focus on performance. SBE uses a compact binary format, allowing for direct serialization based on the predefined schema. This results in efficient and fast serialization operations.
FIX Tag Value	Medium	FIX Tag Value serialization speed can vary depending on the implementation. While it uses a simple text-based format, serialization efficiency can be influenced by factors such as the number of tags and the complexity of handling tag/value pairs and repeating groups. Overall, it tends to have a medium serialization speed.

3.1.2.13 Network overhead optimization

Network overhead optimization refers to the process of minimizing the additional data that is added to the transmitted data due to the data format, including header information, metadata, and error correction mechanisms. The lower the network overhead, the more efficient the data format is in terms of network bandwidth usage.

To which extent does the solution provide network overhead optimization?

Data Format	Grade	Justification
XML	Low	XML has a low network overhead optimization due to its verbose and text-based nature. XML includes additional tags, attribute names, and human-readable content, resulting in larger data size when transmitted over the network. While XML compression techniques exist, such as XML compression libraries or gzip compression, they can improve transmission efficiency but still may not match the optimization levels of more compact formats.
JSON	Medium	JSON has a moderate network overhead optimization. While it is more compact than XML, JSON still includes attribute names and human-readable content. JSON data can benefit from compression techniques like gzip, which can reduce the data size during transmission. However, since JSON is still text-based, the optimization level may not be as high as that of binary formats.
CSV	High	CSV has high network overhead optimization. CSV files consist of plain text with minimal formatting, resulting in smaller data size when transmitted over the network. CSV data can be easily compressed using standard compression techniques like gzip, achieving a significant reduction in data size and network overhead. CSV's simplicity and lack of complex features make it highly optimized for network transmission.
FIXML	Medium	FIXML has a medium network overhead optimization due to its XML-based structure. While it is more compact than plain XML, FIXML still includes XML tags, attributes, and human-readable content. XML compression techniques, such as gzip compression, can be applied

		to reduce the data size during transmission. However, the optimization level may not be as high as that of binary formats.
ASN.1	High	ASN.1 has high network overhead optimization due to its compact binary encoding. ASN.1 encoding eliminates unnecessary data and reduces the overall data size when transmitted over the network. The binary format allows for efficient transmission and enables the use of compression techniques, further enhancing optimization levels. ASN.1's compactness and efficiency make it highly optimized for network transmission.
Protocol Buffer	High	Protocol Buffers have high network overhead optimization due to their efficient binary encoding. Protocol Buffer messages are compact and do not include unnecessary data or human-readable content. The binary format allows for efficient transmission and facilitates the use of compression techniques. Protocol Buffers can achieve significant network overhead optimization, resulting in reduced data size and improved network efficiency.
BSON	Medium	BSON has a medium network overhead optimization due to its binary encoding. While BSON is more compact than JSON, it still incurs additional overhead for the binary representation. BSON-encoded data can benefit from compression techniques, such as gzip compression, to reduce the data size during transmission. However, the optimization level may not be as high as that of fully optimized binary formats.
FAST	High	FAST has high network overhead optimization due to its optimized binary format. FAST is specifically designed for high-performance, low-latency scenarios, prioritizing efficient network transmission. The compact binary encoding and lack of complex features contribute to high optimization levels. Additionally, compression techniques can be applied to further enhance network efficiency.
SBE	High	SBE has high network overhead optimization due to its binary encoding and focus on performance. SBE uses a compact binary format that eliminates unnecessary data and reduces the overall data size during network transmission. The binary encoding allows for efficient transmission, and compression techniques can be applied to achieve even higher optimization levels, improving network efficiency.

FIX Tag Value	High	FIX Tag Value has high network overhead optimization due to its simple text-based format. FIX Tag Value data consists of tag/value pairs with minimal formatting, resulting in smaller data size when transmitted over the network. The lack of complex features allows for efficient transmission, and compression techniques like gzip can be applied to further optimize network overhead. FIX Tag Value is highly optimized for network transmission.

3.1.2.14 Flexibility

Flexibility of a data format refers to its ability to accommodate diverse data structures, adapt to varying data requirements, and support customization and extensibility.

To which extent does the solution provide flexibility?

Data Format	Grade	Justification
XML	High	XML offers high flexibility due to its extensible nature and support for complex data structures. XML allows for the use of namespaces, custom tags, and attributes, providing flexibility in defining and representing data. It allows hierarchical structuring and nesting of elements, making it suitable for representing diverse and complex data models. XML also supports transformations using technologies like XSLT, making it adaptable to different requirements and enabling data integration across systems.
JSON	High	JSON provides high flexibility due to its flexible and lightweight structure. JSON supports key-value pairs, arrays, and nested structures, allowing for the representation of a wide range of data formats. Its simplicity and ease of use make it suitable for various applications. JSON also supports dynamic typing, enabling flexibility in representing different data types within a single JSON object. JSON's flexibility is further enhanced by its compatibility with modern programming languages, making it a popular choice for data exchange and storage in web-based applications and APIs.
CSV	Low	CSV has low flexibility compared to other formats due to its simplistic structure. CSV represents data as plain text with a simple delimiter, typically a comma or tab. It lacks inherent support for complex data

		structures, nested data, or data types other than plain text. CSV is primarily designed for representing tabular data, making it less flexible for representing hierarchical or relational data models. While CSV can be extended with additional metadata or conventions, it requires custom implementations or external documentation to interpret and handle such extensions, limiting its flexibility in comparison to more specialized formats.
FIXML	Medium	FIXML offers medium flexibility due to its XML-based structure. It inherits some flexibility from XML, including support for namespaces, custom tags, and attributes. FIXML allows for the representation of financial messages and structures, providing a level of flexibility for defining specific message formats. However, FIXML's flexibility is constrained by the FIX protocol's standard message definitions and the need to adhere to industry standards and message schemas.
ASN.1	High	ASN.1 provides high flexibility due to its extensible and modular nature. ASN.1 allows the definition of complex data structures, including choice types, sequences, and extensibility mechanisms. It supports the definition of custom data types, allowing precise specification of data formats. ASN.1's flexibility enables the representation of a wide range of data models, making it suitable for various industries and domains. ASN.1 also supports versioning and evolution of data formats, enabling compatibility and adaptability over time.
Protocol Buffer	Medium	Protocol Buffers offer medium flexibility. While they provide support for defining custom data structures and data types, the flexibility is somewhat limited compared to text-based formats like XML or JSON. Protocol Buffers emphasize performance and efficiency, prioritizing a compact binary representation. This focus on efficiency restricts some of the flexibility in data modelling and schema evolution. While Protocol Buffers allow optional fields and extensions, modifications to existing schemas require careful consideration to maintain compatibility.
BSON	Medium	BSON provides medium flexibility due to its binary format and support for additional data types compared to JSON. BSON allows for the representation of more complex data structures than plain JSON, including nested arrays and objects. It also supports additional data types such as dates, binary data, and regular expressions. However,

		BSON's flexibility is still limited compared to formats like XML or ASN.1. The binary nature of BSON can make it less human-readable and less compatible with some programming languages, reducing its flexibility in certain contexts.
FAST	Low	FAST has low flexibility compared to other formats. It is designed for high-performance and efficiency in financial trading scenarios, focusing on optimizing message encoding and decoding. FAST employs predefined message templates and a fixed set of data types, limiting the flexibility for defining custom data structures or extending the format. While FAST allows some customization through templates and the use of optional fields, its primary goal is to achieve high throughput and low latency rather than offering extensive flexibility in data modelling or schema evolution.
SBE	Medium	SBE provides medium flexibility, balancing performance with some degree of customization. SBE allows the definition of message schemas with custom data types, fields, and groups. It supports fixed-size fields, variable-length data, and composite data structures. While SBE provides flexibility in defining data formats, its focus on performance and efficient binary encoding imposes some limitations compared to more expressive formats.
FIX Tag Value	Low	FIX Tag Value has low flexibility compared to other formats. It represents data as a series of tag/value pairs with minimal formatting. FIX Tag Value lacks inherent support for complex data structures or nested data. While FIX Tag Value allows customization through the definition of custom tags and values, the format's simplicity limits its flexibility compared to formats like XML or JSON.

3.1.2.15 Open solution

Data format must be available as an open solution. The latter is understood as a solution, which allows the users to benefit from the right to use, change and distribute the solution without any restrictions and free of charge. Further information is provided in the appendix on the governance model surrounding the standardisation of the shortlisted solutions.

Can the solution be considered as an open solution?

Data Format	Grade	Justification
XML	Yes	XML is considered an open solution. It is a widely adopted and well-documented standard that is platform independent. XML specifications are publicly available, and various libraries and tools exist to parse, generate, and manipulate XML data in multiple programming languages. XML's openness promotes interoperability, allowing different systems to exchange data using a common format.
JSON	Yes	JSON is considered an open solution. It has become a widely adopted standard for data exchange, particularly in web-based applications and APIs. JSON specifications are publicly available, and there are numerous libraries and tools available in various programming languages for parsing, generating, and manipulating JSON data. JSON's simplicity and openness have contributed to its popularity and interoperability across different systems and platforms.
CSV	Yes	CSV is considered an open solution. It is a simple and widely supported format for tabular data representation. CSV files can be easily read and written using a variety of tools and programming languages. While CSV does not have an official specification, its basic structure and conventions are well-known and widely adopted. CSV's openness and simplicity make it easy to work with and promote interoperability between different systems and applications.
FIXML	Yes	FIXML is considered an open solution. It is an XML-based standard for representing financial messages within the FIX protocol. FIXML specifications are publicly available, allowing users to implement and interact with the standard. There are libraries and tools available for parsing, generating, and processing FIXML data in different programming languages. FIXML's openness enables interoperability in the financial industry, facilitating communication between various systems and counterparties.
ASN.1	Yes	ASN.1 is considered an open solution. It is a widely used standard for specifying data structures and encoding rules. ASN.1 specifications are publicly available, and multiple tools and libraries exist for working with ASN.1 data in various programming languages. ASN.1's openness promotes interoperability and allows different systems to exchange data using a standardized format. Additionally, ASN.1

		supports extensibility and customization, allowing users to define their own data types and structures within the framework of the standard.
Protocol Buffer	Yes	Protocol Buffers are considered an open solution. Google, the creator of Protocol Buffers, has released the protocol specifications publicly, allowing anyone to implement or use them. The Protocol Buffers compiler and libraries are also open-source and available for multiple programming languages. The open nature of Protocol Buffers promotes interoperability and encourages the development of various tools, libraries, and frameworks that support the format.
BSON	Yes	BSON is considered an open solution. The BSON specification is publicly available, providing details on its structure and encoding rules. BSON libraries and tools are available as open-source projects for multiple programming languages, enabling developers to parse, generate, and manipulate BSON data. BSON's openness promotes interoperability, allowing different systems to exchange data using a common binary format.
FAST	Yes	FAST is considered as an open solution developed by the FIX Trading Community for high-performance financial trading. The FAST specifications and implementations are publicly available. The usage of FAST is facilitated by the libraries and tools available
SBE	Yes	SBE was developed as an open-source project by the OpenFAST community. As an open-source project, SBE is freely available and can be implemented by any organization without proprietary restrictions. The openness of SBE allows for transparency, collaboration, and innovation within the community, making it an open format.
FIX Tag Value	Yes	FIX Tag Value is considered an open solution. While FIX Tag Value is part of the FIX protocol, which is a proprietary industry standard, the FIX specification itself is publicly available, and numerous libraries and tools exist for working with FIX messages in different programming languages. FIX Tag Value's openness enables interoperability among various financial systems and promotes the exchange of financial messages using a standardized format.

3.1.2.16 Level of adoption

Data formats must show a significant level of adoption in other regulatory framework, in Europe, and in other jurisdictions.

The level of adoption has been determined based on the responses to the questionnaires from the 11 responding entities, including 9 market data contributors and 2 prospective CTP candidates. The grading is resulting from the following approach:

- For each possible level of usage of a data format (i.e High, Medium, Low), one point is added depending on the individual questionnaire's responses. This results into an intermediate score by data format and level of usage.
- Then for each data format, a total score is computed by summing up the weighted intermediate scores (i.e the intermediate score under "high" level of usage are multiplied by 10; the one under "medium" by 5; and the one under "low" by 1).
- Finally, the total score is converted into the grade "High" when above 60; into the grade "Medium" when between 30 and 60; and into the grade "Low", when below 30.

For the sake of consistency with the study's scope, proprietary binary formats were not considered while largely used by the responding entities.

What is the level of adoption of the data format?

Data Format	Grade	Justification
XML	Medium	From the questionnaire responses received from 11 entities, 9 have said to use XML with varying levels of adoption (4 High, 1 Medium and 4 Low) thus making the overall level of adoption as medium.
JSON	Medium	From the questionnaire responses received, 9 respondents have said to use JSON with varying levels of adoption (4 High, 2 Medium and 3 Low) thus making the overall level of adoption as medium.
CSV	Low	From the questionnaire responses received, 2 respondents have said to use CSV with varying levels of adoption (1 High, 1 Medium) thus making the overall level of adoption as low.
FIXML	Low	From the questionnaire responses received though many have stated using FIX format, but none has stated FIXML.

ASN.1	Low	No market data contributor has said to use ASN.1 format for market data distribution.
Protocol Buffer	Low	Out of 11 respondents, only 3 have said to use protocol buffer with 1 high and 2 medium level of adoption.
BSON	Low	Only one market data contributor has said to use BSON but historically.
FAST	High	High level of adoption is seen for FAST format with 11 market data contributors using it, of which 8 has rated high and 1 medium.
SBE	Medium	3 out of 11 respondents are using SBE format with high level of adoption for real-time market data distribution thus making the overall level of adoption medium.
FIX Tag Value	High	High level of adoption is seen for FIX format with 8 market data contributors using it, of which 7 has rated high and 1 medium.

3.1.2.17 Implementation feasibility

Implementation feasibility factor is the investment required for the setup of the solution. The solution should be implemented at a reasonable cost.

What is the level of implementation feasibility of the solution?

Data Format	Grade	Justification
XML	High	XML is a well-established and widely supported format with extensive documentation, libraries, and tools available for parsing, generating, and manipulating XML data in various programming languages. XML's text-based nature makes it accessible for developers and easy to work with. XML also benefits from its compatibility with existing web and enterprise technologies, making it relatively straightforward to integrate and implement in different systems and applications.
JSON	High	JSON become a standard format for data exchange in web-based applications and APIs. JSON libraries and tools are widely available for parsing, generating, and manipulating JSON data in multiple

		programming languages. JSON's simplicity and human-readable format contribute to its ease of implementation. JSON is also natively supported by modern programming languages, making it a popular choice for developers. JSON's wide adoption and tooling support enhance its feasibility for implementation across various platforms and systems.
CSV	High	CSV is a simple and widely used format for tabular data representation. Virtually all programming languages have built-in or third-party libraries and tools for reading and writing CSV data. CSV's straightforward structure, consisting of rows and columns with a delimiter, makes it easy to process and manipulate. The wide adoption and availability of CSV-related tooling, combined with its simplicity, contribute to its high feasibility for implementation in various applications, particularly those involving tabular data processing.
FIXML	Medium	FIXML is an XML-based format specific to the FIX protocol, widely used in the financial industry. Implementing FIXML requires familiarity with the FIX protocol and its messaging standards. While FIXML specifications are publicly available, understanding and adhering to the complex FIX messaging rules and schemas may require additional effort. However, several FIX-related libraries and tools exist to facilitate FIXML implementation, making it feasible to work with FIXML in financial systems and applications with the necessary domain knowledge.
ASN.1	Medium	ASN.1 provides a rich framework for defining data structures and encoding rules. Implementing ASN.1 requires knowledge of the ASN.1 notation and the chosen encoding rules (e.g., BER, DER). ASN.1 compilers and libraries are available for various programming languages to automate the process of encoding and decoding ASN.1 data. However, the complexity of ASN.1 and its various encoding options may require additional effort and expertise compared to simpler formats.
Protocol Buffer	High	Google provides an open-source Protocol Buffers compiler and libraries for multiple programming languages, making it easier to generate code for encoding and decoding Protocol Buffer messages. The well-defined schema definition language simplifies the development process. Protocol Buffers have good interoperability and support across different platforms and programming languages,

		enhancing their feasibility for implementation in diverse systems. The availability of community support and extensive documentation further contributes to the feasibility of implementing Protocol Buffers.
BSON	Medium	BSON libraries and tools are available as open-source projects for multiple programming languages, enabling developers to parse, generate, and manipulate BSON data. While BSON's binary structure may require more specialized handling compared to text-based formats, the availability of libraries simplifies the implementation process. BSON's feasibility for implementation depends on the programming language ecosystem and the specific requirements of the project.
FAST	Medium	Implementing FAST requires adherence to the FAST protocol specification, which may involve a learning curve and understanding of the messaging rules specific to the financial industry. While there are libraries and tools available for working with FAST, their availability and maturity may vary compared to more widely used formats. The feasibility of implementing FAST depends on the specific requirements of the financial trading system and the availability of suitable tooling and expertise.
SBE	Medium	Implementing SBE involves understanding the SBE specification and its binary encoding format. SBE compilers and libraries are available for various programming languages, facilitating the encoding and decoding of SBE messages. However, the specific tooling and documentation support for SBE may be less extensive compared to more widely adopted formats. The feasibility of implementing SBE depends on the availability of suitable tooling, expertise, and the specific needs of the high-performance messaging system.
FIX Tag Value	High	FIX Tag Value messages are represented as a series of tag/value pairs with minimal formatting. Several libraries and tools exist for parsing, generating, and manipulating FIX Tag Value messages in different programming languages. The FIX protocol itself has been widely adopted in the financial industry, and its related tooling and documentation contribute to the feasibility of implementing FIX Tag Value messages.

3.1.2.18 ISO 20022 compliance

Data format is expected to be compliant with ISO 20022 standard.

Does the solution provide compatibility with ISO 20022?

Data Format	Grade	Justification
XML	Yes	XML is compatible with ISO20022. ISO20022, an international standard for financial messaging, supports XML as one of the recommended syntaxes for message representation. XML's flexibility and extensibility make it suitable for defining complex message structures and supporting rich metadata.
JSON	Yes	JSON is compatible with ISO20022. While ISO20022 does not mandate a specific syntax, JSON is widely used and supported for representing ISO20022 messages. Many financial systems and APIs provide JSON-based representations for ISO20022 messages, and libraries and tools exist to handle JSON-based ISO20022 messages
CSV	No	CSV is not directly compatible with ISO20022. ISO20022 primarily focuses on defining a message structure and semantic rules for financial messaging, and CSV lacks the necessary structural elements and metadata support to meet the requirements of ISO20022. CSV represents tabular data without a standardized schema, making it challenging to define complex ISO20022 messages and maintain semantic consistency. While it is possible to map ISO20022 data to CSV for specific use cases, CSV is not a recommended format for ISO20022 messages due to its limitations in supporting the standard's full range of features and requirements.
FIXML	No	While FIXML cannot be considered as strictly ISO20022 compliant, mapping mechanisms exist to convert between FIXML and ISO20022 formats. These mappings facilitate interoperability between systems that use ISO20022 and those that utilize FIXML. The availability of mapping tools and the common usage of FIXML in financial systems contribute to its compatibility with ISO20022, allowing for the exchange of information between FIXML-based systems and ISO20022-based systems.

ASN.1	Yes	ISO20022 allows for the use of ASN.1 (Abstract Syntax Notation One) as a syntax for representing messages. ASN.1 provides a rich framework for defining data structures and encoding rules, which aligns with the requirements of ISO20022 message modelling. Many financial systems and protocols use ASN.1 for message serialization and transmission. Libraries and tools are available for handling ISO20022 messages encoded in ASN.1, ensuring compatibility with the standard and supporting interoperability in the financial industry.
Protocol Buffer	No	Protocol Buffers are not directly compatible with ISO20022. ISO20022 does not specify Protocol Buffers as a recommended or supported syntax for message representation. While it is possible to define a mapping between Protocol Buffers and ISO20022, Protocol Buffers lack the built-in structural features and semantic support required by ISO20022. Implementing ISO20022 compatibility with Protocol Buffers would require additional mapping and transformation layers to bridge the semantic differences between the two formats.
BSON	No	BSON is not directly compatible with ISO20022. BSON, a binary representation of JSON-like documents, does not have built-in support for ISO20022 message modelling or semantics. While it is possible to convert ISO20022 messages to BSON for specific use cases, BSON does not provide the necessary structural features and metadata support required by ISO20022, limiting its compatibility with the standard.
FAST	No	FAST is not directly compatible with ISO20022. FAST does not align with the structural requirements and metadata support of ISO20022. While it is possible to define mappings between ISO20022 and FAST, implementing compatibility between the two formats would require additional transformation and interpretation layers.
SBE	No	SBE is not directly compatible with ISO20022. SBE does not align with the structural requirements and metadata support of ISO20022. While it is possible to define mappings between ISO20022 and SBE, implementing compatibility between the two formats would require additional transformation and interpretation layers.

FIX Tag Value	No	FIX Tag Value is not directly compatible with ISO20022. FIX Tag Value is a specific representation format for the FIX protocol, which is distinct from the ISO20022 standard. While it is possible to define mappings between ISO20022 and FIX Tag Value, they are separate messaging standards with different structural requirements and semantics. Implementing ISO20022 compatibility with FIX Tag Value would require additional transformation and interpretation layers to bridge the semantic differences between the two formats.

3.1.2.19 Protocol compatibility

The data format must be compatible with the different protocols to facilitate message exchange.

To which extent is the solution providing protocol compatibility?

Data Format	Grade	Justification
XML	High	XML has high protocol compatibility. XML is a text-based format and can be easily transmitted over HTTP, HTTPS, and other web protocols. It can be included as the payload in RESTful API requests or SOAP messages. However, XML is not inherently optimized for FTP or SFTP, which primarily deal with file transfer rather than structured data. Websockets can also transmit XML data, but the compatibility may be medium due to the specific implementation requirements. Overall, XML has high compatibility with HTTP, HTTPS, REST, and SOAP, and lower compatibility with FTP, SFTP, MQ, and Websockets.
JSON	High	JSON has high protocol compatibility. It is widely used in web-based applications and APIs, making it a natural fit for HTTP, HTTPS, and RESTful services. JSON data can be easily included in request and response bodies. However, like XML, JSON is not optimized for FTP or SFTP, which primarily deal with file transfer. Websockets can transmit JSON data, but the compatibility may be medium due to the specific implementation requirements. JSON has lower compatibility with SOAP, as SOAP messages typically use XML as the primary data format. Overall, JSON has high compatibility with HTTP, HTTPS,

		REST, and lower compatibility with FTP, SFTP, MQ, Websockets, and SOAP.
CSV	Low	CSV has high protocol compatibility for HTTP, HTTPS, and FTP. CSV files can be easily transferred using these protocols, as they are commonly used for file transfer. However, CSV lacks native support for structured data and metadata, making it less compatible with protocols like MQ, Websockets, REST, and SOAP, which typically require well-defined message formats. While it is possible to include CSV data in HTTP requests or responses, additional handling and transformation may be necessary to integrate it with the requirements of the specific protocol or service. Overall, CSV has high compatibility with HTTP, HTTPS, FTP, and lower compatibility with SFTP, MQ, Websockets, REST, and SOAP.
FIXML	Medium	FIXML has high compatibility with HTTP, HTTPS, and RESTful services. FIXML messages can be transmitted using these protocols, often encapsulated in HTTP or HTTPS requests. However, FIXML is not optimized for FTP, SFTP, MQ, Websockets, or SOAP, as it is primarily designed for financial messaging using the FIX protocol. While it is possible to transfer FIXML files using FTP or SFTP, additional handling and mapping may be required. Websockets and MQ may require additional implementation efforts to support FIXML. Overall, FIXML has high compatibility with HTTP, HTTPS, REST, and SOAP, and lower compatibility with FTP, SFTP, MQ, and Websockets.
ASN.1	Medium	ASN.1 messages can be transmitted over HTTP, HTTPS, and other web protocols. However, ASN.1 is a binary format and requires specialized handling and encoding/decoding mechanisms. While it is possible to include ASN.1 data in HTTP or HTTPS requests/responses, additional handling and transformation may be necessary. ASN.1 is not optimized for FTP, SFTP, MQ, Websockets, REST, or SOAP, as these protocols typically rely on text-based or structured data formats. Overall, ASN.1 has medium compatibility with HTTP, HTTPS, and lower compatibility with FTP, SFTP, MQ, Websockets, REST, and SOAP.
Protocol Buffer	Medium	Protocol Buffers messages can be transmitted over HTTP, HTTPS, and RESTful services by including them as binary payloads. However, Protocol Buffers is not natively optimized for FTP, SFTP, MQ, Websockets, or SOAP. While it is possible to transfer Protocol Buffers

		files using FTP or SFTP, additional handling and encoding/decoding mechanisms may be required. Websockets can transmit Protocol Buffers data, but the compatibility may be medium due to the specific implementation requirements. Overall, Protocol Buffers have medium compatibility with HTTP, HTTPS, and lower compatibility with FTP, SFTP, MQ, Websockets, REST, and SOAP.
BSON	Medium	BSON has high protocol compatibility with HTTP, HTTPS, and RESTful services. BSON data can be included in HTTP or HTTPS requests and responses. However, BSON is not optimized for FTP, SFTP, MQ, Websockets, or SOAP. While it is possible to transfer BSON files using FTP or SFTP, additional handling and transformation may be necessary. BSON lacks specific features for MQ, Websockets, REST, and SOAP, which typically require structured or text-based message formats. Overall, BSON has high compatibility with HTTP, HTTPS, REST, and lower compatibility with FTP, SFTP, MQ, Websockets, and SOAP.
FAST	Medium	FAST has medium protocol compatibility. While FAST is a binary format optimized for high-performance messaging, it is not specifically tailored for protocols like FTP, SFTP, MQ, Websockets, REST, or SOAP. FAST messages can be encapsulated in HTTP or HTTPS requests/responses, but additional handling and encoding/decoding mechanisms are required. Websockets can transmit FAST-encoded messages, but the compatibility may be medium due to the specific implementation requirements. Overall, FAST has medium compatibility with HTTP, HTTPS, Websockets, and lower compatibility with FTP, SFTP, MQ, REST, and SOAP.
SBE	Medium	SBE has medium protocol compatibility. SBE is a binary encoding format optimized for high-performance messaging, but it is not specifically tailored for protocols like FTP, SFTP, MQ, Websockets, REST, or SOAP. SBE messages can be included in HTTP or HTTPS requests/responses, but additional handling and encoding/decoding mechanisms are required. Websockets can transmit SBE-encoded messages, but the compatibility may be medium due to the specific implementation requirements. Overall, SBE has medium compatibility with HTTP, HTTPS, Websockets, and lower compatibility with FTP, SFTP, MQ, REST, and SOAP.

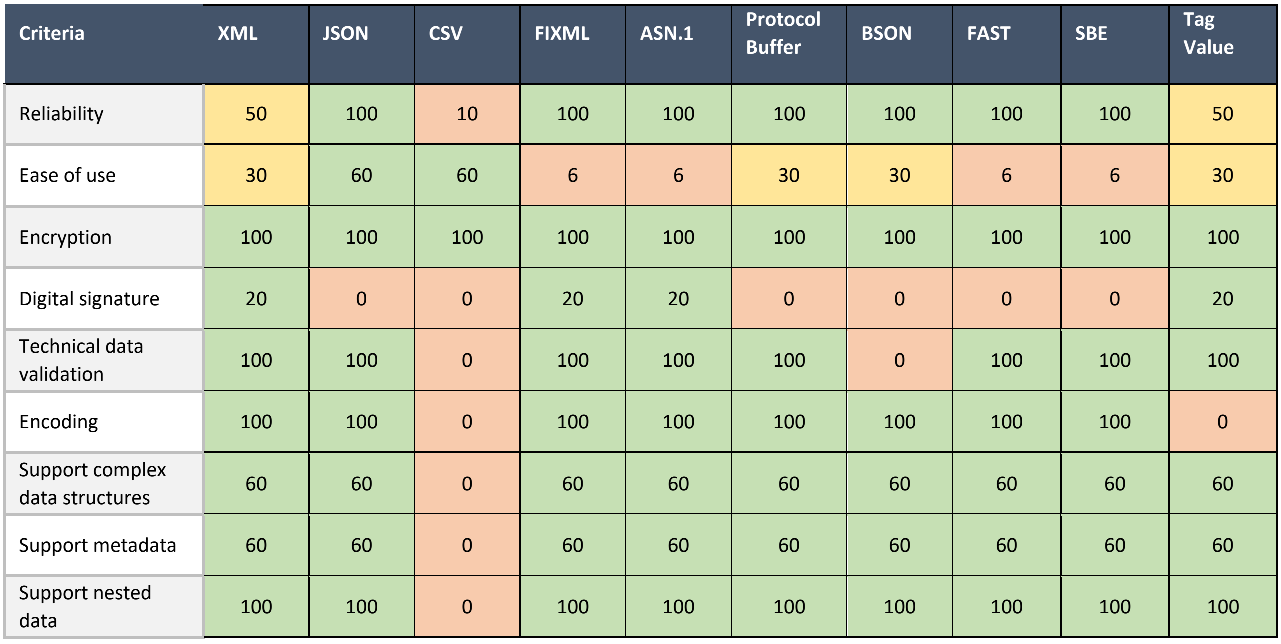
FIX Tag Value	High	FIX Tag Value has high protocol compatibility with HTTP, HTTPS, and RESTful services. FIX Tag Value messages can be included in HTTP or HTTPS requests/responses, often encapsulated as textual data. However, FIX Tag Value is not optimized for FTP, SFTP, MQ, Websockets, or SOAP. While it is possible to transfer FIX Tag Value files using FTP or SFTP, additional handling and mapping may be required. Websockets and MQ may require additional implementation efforts to support FIX Tag Value. Overall, FIX Tag Value has high compatibility with HTTP, HTTPS, REST, and SOAP, and lower compatibility with FTP, SFTP, MQ, and Websockets.

3.1.3 Outcome of the technical assessment

In order to evaluate the technical capabilities of each data format for the CT implementation, a weighted score was assigned based on the performance against the identified criteria. The assessment grid provides further information and details regarding the scores and criteria used during the assessment process.

The table below provides a summary of the scoring for each data format per criteria, as well as the overall score.

FIGURE 8 - OUTCOME OF THE DATA FORMATS' ASSESSMENT



Criteria	XML	JSON	CSV	FIXML	ASN.1	Protocol Buffer	BSON	FAST	SBE	Tag Value
Reliability	50	100	10	100	100	100	100	100	100	50
Ease of use	30	60	60	6	6	30	30	6	6	30
Encryption	100	100	100	100	100	100	100	100	100	100
Digital signature	20	0	0	20	20	0	0	0	0	20
Technical data validation	100	100	0	100	100	100	0	100	100	100
Encoding	100	100	0	100	100	100	100	100	100	0
Support complex data structures	60	60	0	60	60	60	60	60	60	60
Support metadata	60	60	0	60	60	60	60	60	60	60
Support nested data	100	100	0	100	100	100	100	100	100	100

Support inline documentation	20	0	0	20	0	0	0	0	0	20
Parsing speed	50	100	100	50	100	100	100	100	100	50
Serialization speed	50	100	100	50	100	100	100	100	100	50
Network overhead opt.	10	50	100	50	100	100	50	100	100	100
Flexibility	60	60	6	30	60	30	30	6	30	6
Open solution	60	60	60	60	60	60	60	60	60	60
Level of adoption	30	30	6	60	6	6	6	60	30	60
Implementation Feasibility	60	60	60	30	30	60	30	30	30	60
Protocol Compatibility	60	60	6	30	30	30	30	30	30	60
ISO 20022 Compatibility	60	60	0	0	60	0	0	0	0	0
Total Score	1080	1260	608	1026	1192	1136	956	1112	1106	986

Legend:

🟧 Scores equivalent to “Low” or “No”

🟨 Scores equivalent to “Medium”

🟩 Scores equivalent to “High” or “Yes”

Based on the scoring provided in the table above, the data formats are ranked as follows (from highest to lowest total score)

Figure 9 - Ranking of the data formats

Rank	*Data Format*	*Total Score*
1	JSON	1290
2	ASN.1	1192
3	Protocol Buffer	1136
4	FAST	1112
5	SBE	1106
6	XML	1080
7	FIXML	1026
8	Tag Value	986
9	BSON	956
10	CSV	608

JSON emerges as the highest-scoring data format, achieving a score of 1290 out of 1380, thus securing the top position. It is followed by binary formats such as ASN.1, Protocol Buffer, FAST, and SBE.

The text-based formats like XML, FIXML and Tag Value lags on the parsing speed and serialization speed as they are very verbose in nature. CSV format lacks support for technical data validation, encoding and nested data structures. BSON on the other hand scored low on the level of adoption and ISO compatibility.

The top 5 data formats JSON, ASN.1, Protocol Buffer, FAST and SBE secured the same score on the following 10 out of 19 criteria.

Figure 10 – Equally performing criteria for the top 5 data formats

Criteria	Weighting Classification	JSON	ASN.1	Protocol Buffer	FAST	SBE
Reliability	Mandatory	100	100	100	100	100
Encryption	Mandatory	100	100	100	100	100
Technical data validation	Mandatory	100	100	100	100	100
Encoding	Mandatory	100	100	100	100	100
Support complex data structures	Recommended	60	60	60	60	60
Support metadata	Recommended	60	60	60	60	60
Support nested data	Mandatory	100	100	100	100	100
Parsing speed	Mandatory	100	100	100	100	100
Serialization speed	Mandatory	100	100	100	100	100
Open solution	Recommended	60	60	60	60	60

The following criteria showcases the difference in performance of these data formats.

FIGURE 11 - DIFFERENTIATING CRITERIA FOR THE TOP 5 DATA FORMATS

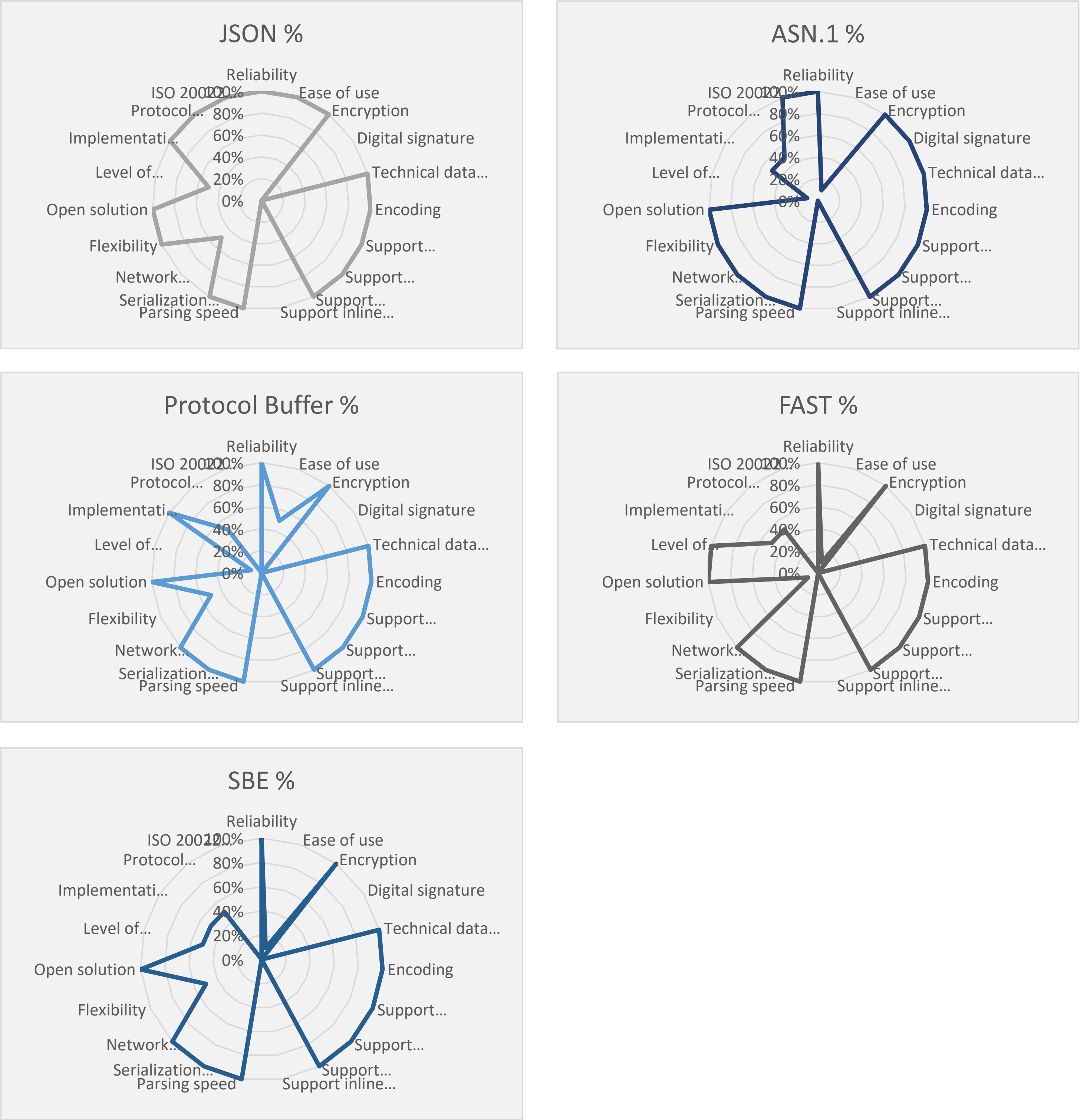
Criteria	Weighting Classification	JSON	ASN.1	Protocol Buffer	FAST	SBE
Ease of use	Recommended	60	6	30	6	6
Digital signature	Nice to have	0	20	0	0	0
Network overhead optimization	Mandatory	50	100	100	100	100
Flexibility	Recommended	60	60	30	6	30
Level of adoption	Recommended	30	6	6	60	30
Implementation Feasibility	Recommended	60	30	60	30	30
Protocol Compatibility	Recommended	60	30	30	30	30
ISO 20022 Compatibility	Recommended	60	60	0	0	0

JSON's success can be attributed to its compatibility with ISO 20022 standard, widespread adoption, and ease of use. However, the other binary formats, namely Protocol Buffer, ASN.1, FAST, and SBE, demonstrate superior network overhead optimization compared to JSON, which is suited for high-volume, low-latency transactions.

Based on this initial assessment, the following top five data formats will proceed to the next level of evaluation, incorporating feedback received from stakeholders involved in the CT project to determine its suitability for implementation of a consolidated tape.

1. JSON
2. ASN.1
3. Protocol Buffer
4. FAST
5. SBE

Figure 12 - Top 5 Data formats unweighted score performance (%)



3.2 Transmission protocols

This section provides firstly a description of the transmission protocols, which have been assessed and secondly an analysis per criteria.

3.2.1 Description of the transmission protocols

The assessment of the transmission protocols has been conducted over 11 different items, which are considered as commonly used especially in the financial services industry.

HTTP (Hypertext Transfer Protocol): HTTP is a protocol used for communication between web browsers and servers. It allows for the exchange of hypertext, which includes text, images, links, and other media. HTTP is based on a client-server model and operates over TCP/IP.

HTTP has been developed at the initiative of the European Organisation for Nuclear Research (CERN) in 1989. The initial standardization work happened with the support of IETF and W3C. The latest release of the protocol, HTTP/3, has been specificized by IETF under RFC 9114[24] in 2022.

HTTPS (Hypertext Transfer Protocol Secure): HTTPS is the secure version of HTTP. It adds encryption and authentication mechanisms to ensure secure communication between clients and servers. HTTPS uses SSL/TLS protocols to protect data transmitted over the network.

HTTPS was created in 1994 by the Californian company, Netscape Communications corporation. The latter has been acquired in 1998 by America Online, known as AOL which then became part of Yahoo, itself becoming a brand of Apollo Global Management, a private US based equity firm. HTTPS standard has been specified in 2000 with RFC 2818[25].

MQ (Message Queue): MQ refers to various message queuing systems that facilitate asynchronous communication between applications. Messages are sent and stored in queues, enabling decoupling of components and ensuring reliable message delivery. MQ systems often provide features like message persistence and priority-based processing.

[24] RFC 9114 - HTTP/3 (ietf.org)
[25] RFC 2818: HTTP Over TLS (rfc-editor.org)

Message queuing systems can either be proprietary or open sources. The proprietary versions have the longest history starting with IBM MQ[26] developed in 1993.The open sources versions are widely known under Apache ActiveMQ and Apache Kafka among others. Both are maintained by the Apache Software Foundation[27], a non-profit organization founded in 1999 in the U.S.

FTP (File Transfer Protocol): FTP is a protocol designed for file transfer between systems over a network. It allows users to upload, download, and manage files on a remote server. FTP operates on the client-server model and typically uses separate control and data connections for communication.

The original specifications of FTP are dated from 1971. Next to further evolution and standardization, the core of FTP standard has been published by the IETF under RFC 959[28]. The IETF is an international standard setting organization[29] registered as a single member limited liability company of the Internet Society, which is an international non-profit organization registered in the U.S (Washington D.C)[30].

FTPS (File Transfer Protocol Secure): FTPS is an extension of FTP that adds encryption to secure file transfers. It combines FTP with SSL/TLS protocols, providing data confidentiality and integrity during transmission. FTPS requires an SSL/TLS certificate for authentication.

While standardization of FTPS was initiated in 1996, it has been finalized in 2005 with the publication of RFC 4217[31] by the Internet Society. The latter is an international non-profit organization registered in the U.S.

SFTP (SSH File Transfer Protocol): SFTP is an SSH-based protocol for secure file transfer. It enables secure file operations like upload, download, and directory listing over an SSH connection. SFTP provides strong encryption and authentication mechanisms, using SSH for secure communication.

[26] IBM MQ | IBM
[27] Apache Software Foundation!
[28] RFC 959: File Transfer Protocol (rfc-editor.org)
[29] IETF | Internet Engineering Task Force
[30] p.11, IETF 2022 audited financial statements
[31] RFC 4217: Securing FTP with TLS (rfc-editor.org)

SSH FTP has been designed by the IETF as an extension of the Secure Shell protocol version 2[32]. The IETF is an international standard setting organization[33] registered as a single member limited liability company of the Internet Society, which is an international non-profit organization registered in the U.S (Washington D.C)[34].

AS2 (Applicability Statement 2): AS2 is a specification for secure data exchange between trading partners. It defines standards and protocols for secure and reliable transfer of structured business documents over the internet. AS2 often incorporates encryption, digital signatures, and message integrity checks.

AS2 has been developed in 2002 by the IETF for replacing AS1 created in the 1990's. AS2 specifications have been formalized under RFC 4130[35]. The IETF is an international standard setting organization[36] registered as a single member limited liability company of the Internet Society, which is an international non-profit organization registered in the U.S (Washington D.C)[37].

Websocket: Websocket is a communication protocol that provides full-duplex communication channels over a single TCP connection. It allows for real-time, bi-directional communication between clients and servers. Websocket is commonly used in web applications that require real-time updates or interactive features. For more details on websockets please refer to the following

1. https://web.dev/websockets-basics/
2. https://www.kodeco.com/13209594-an-introduction-to-websockets/

The development of websockect started in 2008 under the initiative of the W3C. Later on, the normative specification work of the protocol has been moved to the IETF, which published RFC 6455 in 2011[38]. The IETF is an international standard setting organization[39] registered as a single member limited liability company of the Internet Society, which is an international non-profit organization registered in the U.S (Washington D.C)[40].

[32] RFC 4251 - The Secure Shell (SSH) Protocol Architecture (ietf.org)
[33] IETF | Internet Engineering Task Force
[34] p.11, IETF 2022 audited financial statements
[35] RFC 4130 - MIME-Based Secure Peer-to-Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2) (ietf.org)
[36] IETF | Internet Engineering Task Force
[37] p.11, IETF 2022 audited financial statements
[38] RFC 6455 - The WebSocket Protocol (ietf.org)
[39] IETF | Internet Engineering Task Force
[40] p.11, IETF 2022 audited financial statements

SOAP (Simple Object Access Protocol): SOAP is a messaging protocol used for exchanging structured information between networked applications. It relies on XML for message formatting and often uses HTTP or other protocols for message transmission. SOAP allows for remote procedure calls and supports extensive standards for message routing and security.

SOAP was firstly elaborated and released as XML-RPC in June 1998 by collaborators of Microsoft, a U.S registered public multinational company. The specification work has then been undertaken by the W3C, which published the recommendations in 2003[41]. The W3C is an international standard setting organization, which registered in January 2023 a public interest non-profit organization governed by U.S laws[42].

REST (Representational State Transfer): REST is an architectural style for designing networked applications. It emphasizes simplicity, scalability, and a stateless client-server model. RESTful APIs use standard HTTP methods (GET, POST, PUT, DELETE) for communication and often exchange data in JSON or XML format.

The history of REST development is closely linked to the three Web's primary standards (URI, HTTP and HTML) elaborated by the W3C and IETF. REST has been defined by the American computer scientist, Roy Fielding, in his PhD dissertation defended in 2000 at the University of California[43]. Yet REST is a set of architectural constraints, and has not been subject to formal standardization other than the one applicable to the transport protocols like HTTPS.

Multicast UDP (User Datagram Protocol): Multicast UDP enables efficient one-to-many communication over a network. It allows a sender to transmit data to multiple recipients simultaneously by sending a single packet that is received by all members of a multicast group. Multicast UDP is commonly used for streaming media and other multicast applications.

Multicast UDP also known as Multicast IP was firstly developed by Stephen Deering, an American researcher working at Stanford university at that time. His work on multicast IP has been awarded the IEEE internet award in 2010 by the New York headquartered Institute of

[41] SOAP Specifications (w3.org)
[42] Art. 18 Governing law https://www.w3.org/2023/01/Member-Agreement
[43] Chap.5 in Architectural Styles and the Design of Network-based Software Architectures (uci.edu)

Electric and Electronics Engineers (IEEE). Multicast IP has been standardized with RFC 1112 published in 1989[44];

3.2.2 Assessment of the transmission protocols

The assessment of the transmission protocols has been performed against 18 criteria featuring both standard technical and CTP specific requirements.

3.2.2.1 Data confidentiality

Data confidentiality refers to the ability of a transmission protocol to keep data secure and protect it from unauthorized access. This includes features such as encryption and secure data transfer mechanisms.

Does the solution provide data confidentiality?

Protocol	Grade	Justification
HTTP	No	HTTP is not secure by default and does not provide data confidentiality. The data transmitted over HTTP is sent in plain text, making it vulnerable to eavesdropping and interception.
HTTPS	Yes	HTTPS provides data confidentiality by encrypting the data using SSL/TLS protocols. It establishes a secure connection between the client and server, ensuring that the data transmitted is encrypted and protected from unauthorized access or interception.
MQ	Yes	MQ (Message Queuing) protocols, such as IBM MQ or RabbitMQ, can provide data confidentiality by utilizing TLS (Transport Layer Security) encryption. By enabling TLS, the data transmitted between the message queue components is encrypted, ensuring confidentiality and protecting against eavesdropping.
FTP	No	FTP (File Transfer Protocol) does not provide data confidentiality. The data transferred over FTP is sent in plain text, leaving it susceptible to interception and unauthorized access.
FTPS	Yes	FTPS (FTP over SSL/TLS) provides data confidentiality by encrypting the data during transmission. It adds a layer of security to

[44] RFC 1112: Host extensions for IP multicasting (rfc-editor.org)

		FTP by using SSL/TLS protocols, ensuring that the data remains confidential and protected from eavesdropping.
SFTP	Yes	SFTP (SSH File Transfer Protocol) ensures data confidentiality by encrypting the data using Secure Shell (SSH) protocols. It establishes a secure connection between the client and server, preventing unauthorized access and protecting the data in transit.
AS2	Yes	AS2 (Applicability Statement 2) protocol incorporates encryption and digital signatures to provide data confidentiality. It ensures secure and reliable data exchange by using cryptographic mechanisms, protecting the integrity and privacy of the transmitted information.
Websocket	Yes	When Websocket is used over HTTPS (WSS - WebSockets over SSL/TLS), it provides data confidentiality. By leveraging the secure transport layer of HTTPS, the data transmitted via Websocket is encrypted and remains confidential, preventing unauthorized access or interception.
SOAP	Yes	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS provide data confidentiality. HTTPS encrypts the message payload, ensuring that the data remains secure during transmission and preventing unauthorized access or eavesdropping.
REST	Yes	REST (Representational State Transfer) messages transmitted over HTTPS provide data confidentiality. HTTPS encrypts the data transmitted between the client and server, ensuring its security and preventing unauthorized interception or access.
Multicast UDP	Yes	Multicast transmissions using IPsec (Internet Protocol Security) can provide data confidentiality. IPsec encrypts the multicast data packets, ensuring that they remain secure and preventing unauthorized access or eavesdropping. By using IPsec, data confidentiality can be achieved in Multicast UDP communications.

3.2.2.2 Authentication

Authentication refers to the ability of a transmission protocol to verify the identity of the sender or receiver of data. This includes features such as user credentials, digital certificates, or biometric authentication.

Does the solution provide authentication?

Protocol	Grade	Justification
HTTP	No	HTTP does not provide built-in authentication mechanisms. It does not have any inherent method to verify the identity of the client or server.
HTTPS	Yes	HTTPS supports authentication through SSL/TLS protocols. It allows the client and server to verify each other's identity using digital certificates, ensuring secure and authenticated communication.
MQ	Yes	MQ protocols, such as IBM MQ or RabbitMQ, can utilize TLS (Transport Layer Security) for authentication. TLS provides secure and authenticated communication by verifying the identities of the message queue components using digital certificates.
FTP	No	FTP does not have native authentication mechanisms. The protocol itself does not include built-in features for verifying the identity of the client or server.
FTPS	Yes	FTPS (FTP over SSL/TLS) supports authentication by leveraging SSL/TLS protocols. It allows the client and server to authenticate each other using digital certificates, ensuring secure and authenticated FTP communication.
SFTP	Yes	SFTP (SSH File Transfer Protocol) utilizes Secure Shell (SSH) protocols, which include authentication mechanisms. It allows clients to authenticate with the server using SSH keys or passwords, ensuring secure and authenticated file transfers.
AS2	Yes	AS2 (Applicability Statement 2) incorporates authentication mechanisms using digital signatures and certificates. It allows trading partners to authenticate each other's identities and ensure the integrity and authenticity of the transmitted data.

Websocket	Yes	When Websocket is used over HTTPS (WSS - WebSockets over SSL/TLS), it benefits from the authentication capabilities provided by HTTPS. This includes the ability to verify the identity of the client and server using digital certificates.
SOAP	Yes	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS can leverage the authentication mechanisms of HTTPS. It allows clients and servers to authenticate each other using digital certificates, ensuring secure and authenticated SOAP communication.
REST	Yes	REST (Representational State Transfer) messages transmitted over HTTPS can utilize the authentication mechanisms provided by HTTPS. It allows clients and servers to authenticate each other using digital certificates, ensuring secure and authenticated REST communication.
Multicast UDP	Yes	Multicast transmissions using IPsec (Internet Protocol Security) can include authentication mechanisms. IPsec allows for the use of authentication protocols, such as IKE (Internet Key Exchange), to verify the identities of participating hosts in a multicast group. This ensures secure and authenticated communication within the multicast network.

3.2.2.3 Authorization

Authorization refers to the ability of a transmission protocol to ensure that users or systems only have access to the data that they are authorized to access. This includes features such as access control mechanisms and role-based access controls.

Does the solution provide authorization?

Protocol	Grade	Justification
HTTP	No	HTTP does not provide built-in authorization mechanisms. It does not have inherent features to control access to resources or validate user permissions.

HTTPS	Yes	HTTPS supports authorization through various mechanisms implemented at the application layer. Authorization can be enforced through user authentication, session management, and access control mechanisms implemented by the web application or server.
MQ	Yes	MQ protocols, such as IBM MQ or RabbitMQ, can implement authorization mechanisms to control access to message queues and resources. This can include user-based access controls, role-based access controls (RBAC), or other authorization mechanisms defined by the specific MQ implementation.
FTP	No	FTP does not have native authorization mechanisms. The protocol itself does not include built-in features for access control or user permissions.
FTPS	Yes	FTPS (FTP over SSL/TLS) supports authorization through various mechanisms implemented by the FTP server software. This can include user authentication, access control lists (ACLs), or other methods to control access to FTP resources.
SFTP	Yes	SFTP (SSH File Transfer Protocol) leverages Secure Shell (SSH) protocols, which include authentication and authorization mechanisms. SFTP servers can enforce access controls based on user identities, group memberships, or other authorization mechanisms supported by the SSH server software.
AS2	Yes	AS2 (Applicability Statement 2) can implement authorization mechanisms to control access to AS2 endpoints and resources. This can include authentication and access control mechanisms defined by the AS2 implementation or through integration with existing security frameworks.
Websocket	Yes	Websocket applications running over HTTPS can implement authorization mechanisms similar to traditional web applications. This can include session management, user authentication, and access control mechanisms enforced at the application layer.
SOAP	Yes	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS can implement authorization mechanisms at the application layer. This can include authentication of SOAP requests,

		enforcement of user roles and permissions, or other access control mechanisms defined by the SOAP service implementation.
REST	Yes	REST (Representational State Transfer) services running over HTTPS can implement authorization mechanisms at the application layer. This can include token-based authentication, OAuth, or other methods to enforce access control and permissions on REST resources.
Multicast UDP	Yes	Multicast transmissions using IPsec (Internet Protocol Security) can include authorization mechanisms. Access controls can be enforced at the network layer or through higher-level protocols. IPsec can help secure multicast group membership and control access to multicast communication within the network.

3.2.2.4 Non-repudiation

Non-repudiation refers to the ability of a transmission protocol to ensure that the sender of data cannot deny that they sent it, and the receiver cannot deny that they received it. This includes features such as digital signatures and audit trails.

Does the solution provide no-repudiation?

Protocol	Grade	Justification
HTTP	No	HTTP does not provide built-in mechanisms for non-repudiation. It does not include features that can guarantee the authenticity of the sender or the integrity of the transmitted data, making it challenging to prove non-repudiation.
HTTPS	Yes	HTTPS, when combined with digital certificates and SSL/TLS protocols, can provide non-repudiation to some extent. The use of digital certificates enables the verification of the identity of the sender, ensuring that the sender cannot deny their involvement in the communication. SSL/TLS also helps protect the integrity of the data, preventing tampering and enhancing non-repudiation. However, it is important to note that non-repudiation is not an

		inherent feature of HTTPS and may require additional measures for stronger non-repudiation assurance.
MQ	Yes	MQ (Message Queuing) protocols, such as IBM MQ or RabbitMQ, can leverage TLS (Transport Layer Security) to provide non-repudiation. TLS ensures the integrity and authenticity of the messages exchanged between the message queue components, making it difficult for the sender to deny their participation in the communication. The use of digital certificates and cryptographic mechanisms strengthens the non-repudiation assurance.
FTP	No	FTP (File Transfer Protocol) does not provide native support for non-repudiation. It lacks built-in mechanisms to ensure the integrity of the data or guarantee the authenticity of the sender.
FTPS	Yes	FTPS (FTP over SSL/TLS) can provide non-repudiation by leveraging SSL/TLS protocols. The use of SSL/TLS ensures the integrity and authenticity of the data transmitted over FTPS, making it harder for the sender to deny their involvement in the communication. However, non-repudiation may still require additional measures, such as digital signatures or other mechanisms, depending on the specific implementation.
SFTP	Yes	SFTP (SSH File Transfer Protocol) can offer non-repudiation to some extent. By leveraging SSH protocols and authentication mechanisms, SFTP provides a secure and authenticated channel for data transfer, making it challenging for the sender to repudiate their actions. However, non-repudiation may also depend on the specific implementation and additional measures applied, such as the use of digital signatures.
AS2	Yes	AS2 (Applicability Statement 2) incorporates digital signatures and encryption to provide non-repudiation. Digital signatures help verify the authenticity of the sender, ensuring that the sender cannot deny their participation in the communication. The use of encryption enhances data integrity and confidentiality, further strengthening non-repudiation.
Websocket	Yes	When Websocket is used over HTTPS (WSS - WebSockets over SSL/TLS), it can offer non-repudiation to some extent. The use of HTTPS and SSL/TLS protocols ensures the integrity of the

		communication and the authenticity of the sender. However, stronger non-repudiation assurance may require additional measures such as digital signatures or other mechanisms at the application layer.
SOAP	Yes	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS can provide non-repudiation to some extent. HTTPS, along with digital certificates, ensures the integrity and authenticity of the communication, making it challenging for the sender to deny their involvement. However, stronger non-repudiation assurance may require additional measures such as digital signatures or other mechanisms implemented at the application layer.
REST	Yes	REST (Representational State Transfer) services running over HTTPS can provide non-repudiation to some extent. HTTPS, combined with digital certificates, helps verify the authenticity of the sender and ensures the integrity of the transmitted data. However, achieving stronger non-repudiation assurance may require additional measures such as digital signatures or other mechanisms implemented at the application layer.
Multicast UDP	Yes	Multicast transmissions using IPsec (Internet Protocol Security) can incorporate non-repudiation mechanisms. IPsec can provide data integrity, authenticity, and confidentiality, making it difficult for the sender to deny their participation in the multicast communication. By using digital signatures and cryptographic mechanisms, non-repudiation can be strengthened within the multicast network.

3.2.2.5 Encryption

Encryption refers to the ability of a transmission protocol to secure data by encrypting it to prevent unauthorized access or tampering.

Does the solution support encryption?

Protocol	Grade	Justification
HTTP	No	HTTP does not provide native encryption capabilities. It transmits data in plain text, making it susceptible to eavesdropping and unauthorized access.
HTTPS	Yes	HTTPS encrypts the communication between the client and server using SSL/TLS protocols. It ensures that the data transmitted over the network is encrypted, making it difficult for unauthorized parties to intercept and understand the content. HTTPS provides a secure and encrypted channel for data transmission.
MQ	Yes	MQ (Message Queuing) protocols, such as IBM MQ or RabbitMQ, can utilize TLS (Transport Layer Security) for encryption. TLS ensures that the messages exchanged between the message queue components are encrypted, protecting the confidentiality of the data during transmission.
FTP	No	FTP (File Transfer Protocol) does not provide native encryption capabilities. It transmits data in plain text, leaving it vulnerable to interception and unauthorized access.
FTPS	Yes	FTPS (FTP over SSL/TLS) supports encryption by leveraging SSL/TLS protocols. It encrypts the FTP communication, protecting the confidentiality of the data transferred between the client and server. FTPS provides a secure and encrypted channel for FTP transmissions.
SFTP	Yes	SFTP (SSH File Transfer Protocol) utilizes Secure Shell (SSH) protocols, which inherently provide encryption. SFTP encrypts the data during transfer, ensuring the confidentiality of the files being transferred.
AS2	Yes	AS2 (Applicability Statement 2) incorporates encryption mechanisms to protect the confidentiality of the transmitted data. It can utilize encryption algorithms like AES (Advanced Encryption Standard) to secure the AS2 messages during transmission.
Websocket	Yes	When Websocket is used over HTTPS (WSS - WebSockets over SSL/TLS), it benefits from the encryption capabilities provided by HTTPS. The communication between the client and server is

		encrypted, ensuring the confidentiality of the data exchanged over the Websocket connection.
SOAP	Yes	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS can utilize the encryption capabilities of HTTPS. The use of HTTPS ensures that the SOAP requests and responses are encrypted, protecting the confidentiality of the SOAP message content during transmission.
REST	Yes	REST (Representational State Transfer) services running over HTTPS benefit from the encryption capabilities provided by HTTPS. The communication between the client and server is encrypted, ensuring the confidentiality of the data exchanged over the REST API.
Multicast UDP	Yes	Multicast transmissions using IPsec (Internet Protocol Security) can incorporate encryption. IPsec provides encryption mechanisms to protect the confidentiality of multicast data during transmission. It ensures that only authorized recipients can decrypt and access the multicast messages.

3.2.2.6 Secure file transfer

Secure file transfer refers to the ability of a transmission protocol to transfer files securely and reliably over a network. This includes features such as secure file transfer protocols, encryption, and checksums.

Does the solution support secure file transfer?

Protocol	Grade	Justification
HTTP	No	HTTP does not provide built-in mechanisms for secure file transfer. It transmits data in plain text, making it susceptible to eavesdropping and unauthorized access.
HTTPS	Yes	HTTPS provides a secure and encrypted channel for file transfer. By utilizing SSL/TLS protocols, it ensures the confidentiality, integrity,

		and authenticity of the transferred files. HTTPS is commonly used for secure file transfer over the web.
MQ	Yes	MQ (Message Queuing) protocols, such as IBM MQ or RabbitMQ, can leverage TLS (Transport Layer Security) to enable secure file transfer. TLS ensures the encryption and authentication of the messages exchanged between the message queue components, providing a secure channel for file transfer.
FTP	No	FTP (File Transfer Protocol) does not inherently provide secure file transfer capabilities. It transmits data in plain text, leaving it vulnerable to interception and unauthorized access.
FTPS	Yes	FTPS (FTP over SSL/TLS) offers secure file transfer by incorporating SSL/TLS protocols. It encrypts the FTP communication and provides authentication mechanisms, ensuring the confidentiality, integrity, and authenticity of the transferred files.
SFTP	Yes	SFTP (SSH File Transfer Protocol) inherently provides secure file transfer capabilities. It utilizes SSH protocols, including encryption and authentication mechanisms, to protect the confidentiality and integrity of the files during transmission.
AS2	Yes	AS2 (Applicability Statement 2) is specifically designed for secure file transfer. It incorporates encryption, digital signatures, and secure protocols to ensure the confidentiality, integrity, authenticity, and non-repudiation of the transmitted files. AS2 is widely used for secure B2B file exchange.
Websocket	Yes	When Websocket is used over HTTPS (WSS - WebSockets over SSL/TLS), it can provide secure file transfer capabilities. The combination of Websocket and HTTPS ensures the encryption and authentication of the file transfer, offering a secure channel for transmitting files.
SOAP	Yes	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS can support secure file transfer. The use of HTTPS ensures the encryption, authentication, and integrity of the SOAP requests and responses, enabling secure file exchange through SOAP-based services.

REST	Yes	REST (Representational State Transfer) services running over HTTPS can facilitate secure file transfer. The combination of REST and HTTPS provides encryption, authentication, and integrity of the transferred files, ensuring secure file exchange through RESTful APIs.
Multicast UDP	Yes	Multicast transmissions using IPsec (Internet Protocol Security) can enable secure file transfer. IPsec provides encryption and authentication mechanisms to protect the confidentiality, integrity, and authenticity of multicast files during transmission.

3.2.2.7 Low Latency

Latency refers to the time delay between sending and receiving data over a network. Low latency is important for applications that require real-time processing.

What is the level of latency optimization provided by the solution?

Protocol	Grade	Justification
HTTP	Medium	HTTP is a request-response protocol that operates over TCP/IP. While it does not provide inherent low-latency characteristics, optimizations such as persistent connections and HTTP/2 can help reduce latency to a medium level. However, it may not be suitable for applications requiring extremely low latency due to factors like connection establishment and data serialization.
HTTPS	Medium	HTTPS, being an extension of HTTP, inherits similar latency characteristics. The additional encryption and security provided by SSL/TLS can introduce additional latency compared to plain HTTP. However, with optimizations like TLS session resumption and efficient SSL/TLS configurations, the latency impact can be minimized to a medium level.
MQ	High	MQ systems, such as IBM MQ or Apache Kafka, are designed to optimize message delivery and minimize latency. They use lightweight messaging protocols that are specifically engineered for efficient message passing and low-latency communication. MQ systems support asynchronous messaging, which reduces latency

		as senders can proceed without waiting for immediate responses, allowing for faster message delivery.
FTP	Medium	FTP (File Transfer Protocol) operates over TCP/IP and does not inherently provide low-latency communication. It involves multiple round trips for command-response interactions, leading to medium latency. FTP may not be suitable for applications requiring very low latency, but optimizations like TCP tuning and FTP-specific accelerators can help improve performance.
FTPS	Medium	FTPS (FTP over SSL/TLS) adds SSL/TLS encryption to FTP, which can introduce additional latency compared to plain FTP. The encryption and authentication processes increase the computational overhead, resulting in medium latency. While optimizations can help mitigate the impact, FTPS is generally rated as medium latency.
SFTP	High	SFTP operates over SSH and benefits from the low-latency characteristics of SSH. Compared to FTP which requires 2 connections (control and data), SFTP operated with a single connection with efficient encryption and authentication mechanisms, allowing for low-latency transfers. SFTP is designed with performance optimizations in mind, resulting in faster file transfers and low latency.
AS2	Medium	AS2 uses HTTP or HTTPS for secure file transfer. The latency for AS2 depends on the underlying HTTP/HTTPS protocol. While HTTP/HTTPS generally has medium latency, optimizations like persistent connections and content delivery networks (CDNs) can help reduce latency in AS2 transfers.
Websocket	High	The Websocket protocol is designed to provide low-latency, bidirectional communication between a client and a server over a single, long-lived connection. Compared to traditional HTTP, which is a request-response protocol, Websocket allows for real-time, interactive communication.
SOAP	Medium	SOAP (Simple Object Access Protocol) messages transmitted over HTTPS inherit the latency characteristics of HTTPS. While HTTPS introduces additional latency compared to plain HTTP, optimizations like connection reuse and efficient SSL/TLS configurations can help

		mitigate the impact, resulting in medium latency for SOAP over HTTPS communication.
REST	Medium	REST (Representational State Transfer) services running over HTTPS inherit the latency characteristics of HTTPS. The additional encryption and authentication overhead of HTTPS can introduce some latency. However, optimizations like connection reuse and efficient SSL/TLS configurations can help mitigate the impact, resulting in medium latency for REST over HTTPS.
Multicast UDP	High	Multicast transmissions using IPsec (Internet Protocol Security) operate at the network layer and can achieve low latency. IPsec implementations are designed to be efficient and can deliver packets with minimal additional latency, making multicast UDP with IPsec suitable for low-latency communication.

3.2.2.8 Throughput

Throughput refers to the amount of data that can be transferred over a network in a given amount of time. High throughput is important for applications that require large amounts of data to be transferred quickly.

What is the level of throughput provided by the solution?

Protocol	Grade	Justification
HTTP	High	HTTP is a widely used protocol that supports high throughput for data transmission. It is optimized for efficient delivery of web content and can handle concurrent connections, allowing for parallel processing and high data transfer rates.
HTTPS	Medium	HTTPS maintains similar throughput capabilities as HTTP. However, the encryption and decryption process adds some overhead, which might impact the throughput comparable to HTTP.
MQ	High	The throughput of MQ depends on the specific messaging system and its configuration. MQ systems, such as IBM MQ or Apache

		Kafka, are designed to handle high message volumes and provide efficient message delivery. With proper tuning and optimization, they can achieve medium to high throughput levels.
FTP	High	FTP is specifically designed for high-speed file transfer. It supports large file transfers and can achieve high throughput rates, especially in scenarios where bandwidth is not a limiting factor.
FTPS	High	FTPS maintains similar throughput capabilities as FTP while adding secure encryption. The overhead introduced by SSL/TLS encryption is minimized with optimized implementations, resulting in high throughput rates.
SFTP	Medium	SFTP can achieve medium to high throughput rates, but the encryption and decryption process may introduce some overhead compared to unencrypted protocols like FTP.
AS2	Medium	AS2 has medium throughput due to its larger message size and processing overhead such as digital signing and encryption. This overhead can impact the processing time and, consequently, the overall throughput.
Websocket	High	Websocket is designed for real-time, bidirectional communication. It provides high throughput capabilities, enabling efficient data exchange between clients and servers. The persistent connection and optimized messaging format contribute to its high throughput.
SOAP	Medium	SOAP (Simple Object Access Protocol) is an XML-based protocol used for exchanging structured information. Its throughput can vary depending on factors such as payload size and network conditions. With proper optimization, SOAP can achieve medium to high throughput levels.
REST	Medium	REST (Representational State Transfer) is an architectural style for building APIs. Its throughput depends on the implementation and the underlying protocols used (e.g., HTTP/HTTPS). With proper design and optimization, REST APIs can achieve medium to high throughput levels.

Multicast UDP	High	Multicast UDP (User Datagram Protocol) is designed for efficient one-to-many or many-to-many communication. It provides high throughput as data is transmitted simultaneously to multiple recipients. However, the network infrastructure and support for IP multicast can impact its effectiveness.

3.2.2.9 Connection setup time optimization

Connection setup time refers to the amount of time it takes to establish a connection between two systems over a network. Low connection setup time is important for applications that require fast and frequent connections, such as real-time communication applications.

To which extent does the solution optimize connection set up time?

Protocol	Grade	Justification
HTTP	High	HTTP is a stateless protocol that does not require a persistent connection. The connection setup time is generally low as it follows a request-response model, where a client establishes a connection with the server, sends a request, receives a response, and then closes the connection. The connection setup overhead is minimal.
HTTPS	Medium	HTTPS involves an additional handshake process for establishing a secure connection. This handshake includes multiple steps, such as negotiating encryption algorithms and exchanging digital certificates. These additional steps increase the connection setup time compared to plain HTTP.
MQ	Medium	The connection setup time for MQ depends on the specific messaging system and its configuration. Establishing a connection with an MQ system often involves handshakes, authentication, and potentially negotiating security protocols, which can increase the connection setup time compared to simpler protocols.
FTP	High	FTP requires a control connection for command exchange and a separate data connection for file transfer. The control connection setup time is generally low, but the data connection setup time can

		be affected by network latency and firewall configurations. Overall, FTP has a relatively low connection setup time.
FTPS	Medium	FTPS involves the additional steps of establishing a secure connection, including the SSL/TLS handshake process. This adds some overhead to the connection setup time compared to plain FTP.
SFTP	Medium	SFTP requires the establishment of an SSH connection for secure communication. The SSH connection setup involves handshakes, key exchange, and authentication, which can increase the connection setup time compared to non-encrypted protocols like FTP.
AS2	Medium	AS2 involves establishing a secure and reliable connection between the sender and receiver. The connection setup time can be influenced by factors such as authentication, digital certificate exchange, and negotiation of encryption algorithms. Overall, AS2 has a medium connection setup time.
Websocket	Medium	Websocket protocol uses an initial handshake to establish a connection between the client and server. This handshake is done over HTTP/HTTPS and includes upgrade requests and responses. The connection setup time is thus generally medium.
SOAP	Medium	SOAP typically relies on HTTP/HTTPS as the underlying transport protocol. The connection setup time for SOAP depends on the HTTP/HTTPS connection establishment process, which is generally medium due to the additional steps involved in establishing an HTTP-based connection.
REST	Medium	REST leverages HTTP/HTTPS as the underlying protocol. The connection setup time for REST is similar to that of HTTP/HTTPS, which is generally medium.
Multicast UDP	High	Multicast UDP (User Datagram Protocol) operates at the transport layer and does not require connection establishment like TCP-based protocols. The absence of a connection setup process makes the connection setup time very low, allowing for quick and efficient multicast communication.

3.2.2.10 Reliability

Reliability refers to the ability of a transmission protocol to ensure that data is accurately transmitted and received without corruption or loss. This includes features such as error correction and detection mechanisms, as well as recovery mechanisms in case of transmission errors or failures.

To which extent does the solution provide reliability?

Protocol	Grade	Justification
HTTP	Medium	HTTP is a reliable protocol at the transport level, as it uses TCP (Transmission Control Protocol) for data transmission. TCP provides reliable, in-order delivery of data, automatic error detection, and retransmission of lost packets. However, the reliability of HTTP is impacted due to the fact that it is less secure and data is transmitted as plain text.
HTTPS	High	HTTPS (HTTP over SSL/TLS) inherits the reliability of HTTP at the transport level. Additionally, SSL/TLS encryption adds an extra layer of security and integrity to the data transmission, enhancing overall reliability. HTTPS ensures that the data sent between the client and server remains encrypted and tamper-proof, further increasing reliability.
MQ	High	MQ (Message Queue) systems are designed to provide high reliability. They often incorporate features such as message persistence, acknowledgments, and transactional support. These mechanisms ensure that messages are reliably delivered, even in the event of network failures or system outages, making MQ a highly reliable choice for message-oriented communication.
FTP	Medium	FTP offers a medium level of reliability. It uses TCP as the underlying transport protocol, which provides reliable data transfer. However, FTP does not have built-in mechanisms for automatic error recovery or guaranteed message delivery. The reliability of FTP can also be affected by network conditions and interruptions during file transfers.
FTPS	High	FTPS inherits the reliability of FTP at the transport level. Additionally, it adds the security and integrity benefits of SSL/TLS

		encryption, further enhancing reliability. FTPS ensures that the file transfers remain encrypted and protected, reducing the risk of data corruption or unauthorized access, and increasing overall reliability.
SFTP	High	SFTP relies on the SSH protocol for secure communication. SSH provides encryption, authentication, and integrity checks, ensuring a high level of reliability. SFTP ensures that file transfers are secure and protected from unauthorized access or data corruption, making it a reliable choice for secure file transfers.
AS2	High	AS2 is designed to offer high reliability in exchanging business documents. It incorporates various mechanisms such as message acknowledgments, digital signatures, and MDNs (Message Disposition Notifications) to ensure reliable and secure message delivery. AS2 provides strong guarantees of message integrity and non-repudiation, making it highly reliable for business transactions.
Websocket	High	Websocket protocol is built on top of TCP, inheriting its reliability. It establishes a long-lived connection between the client and server, allowing for reliable bidirectional communication. Websockets include mechanisms for detecting connection failures and providing error notifications, ensuring reliable real-time communication between the client and server.
SOAP	High	SOAP (Simple Object Access Protocol) relies on HTTP/HTTPS as the underlying transport protocol, providing the reliability characteristics of HTTP. Additionally, SOAP can incorporate WS-ReliableMessaging, which introduces reliable message delivery features to ensure message integrity and delivery guarantees.
REST	High	REST (Representational State Transfer) APIs use HTTP/HTTPS, which offers reliable data transmission at the transport level. RESTful architectures can also incorporate mechanisms such as idempotency, error handling, and retries to enhance reliability.
Multicast UDP	Low	Multicast UDP offers a low level of reliability. UDP is a connectionless protocol that does not provide guaranteed delivery or error recovery mechanisms. While multicast UDP enables efficient one-to-many or many-to-many communication, its reliability

		heavily relies on higher-level protocols or application-layer mechanisms for error detection and recovery.

3.2.2.11 Scalability

Scalability refers to the ability of a transmission protocol to handle increasing amounts of traffic and users without sacrificing performance or reliability. This includes features such as load balancing and distributed systems.

To which extent does the solution support scalability?

Protocol	Grade	Justification
HTTP	High	HTTP is highly scalable due to its stateless nature and widespread adoption. It is designed to handle large numbers of concurrent connections, making it suitable for web-based applications and distributed systems. With load balancing techniques and horizontal scaling, HTTP-based systems can handle high traffic loads and scale horizontally by adding more servers.
HTTPS	High	HTTPS inherits the scalability of HTTP. SSL/TLS encryption adds some overhead, but it does not significantly impact the scalability of the underlying HTTP protocol. With proper hardware and network infrastructure, HTTPS can handle large-scale deployments and accommodate high traffic volumes efficiently.
MQ	High	MQ systems are designed for high scalability. They provide messaging infrastructure that can scale horizontally by adding more message brokers or queue managers to handle increased message traffic. MQ systems often incorporate features like clustering and load balancing, enabling distributed and scalable messaging architectures.
FTP	Medium	FTP can be scaled to some extent by implementing load balancing techniques and employing multiple FTP servers. However, the scalability of FTP may be limited by its stateful nature and the need for maintaining control connections. With proper setup and load distribution, FTP can handle a moderate level of scalability, but it

		may not scale as efficiently as some other protocols designed for high scalability.
FTPS	Medium	FTPS inherits the scalability characteristics of FTP. While SSL/TLS encryption adds some overhead, it does not significantly impact the scalability of the underlying FTP protocol. By employing load balancing and distributed server configurations, FTPS can handle a moderate level of scalability, similar to FTP.
SFTP	Medium	SFTP can scale to some extent by deploying multiple SFTP servers and load balancing incoming connections. The scalability of SFTP depends on factors such as server infrastructure, network bandwidth, and system resources. With proper configuration and load distribution, SFTP can handle a moderate level of scalability for secure file transfers.
AS2	Medium	AS2 can be scaled by deploying multiple AS2 servers and employing load balancing techniques. The scalability of AS2 depends on factors such as message volume, processing capacity, and network infrastructure. AS2 systems can handle a moderate to high level of scalability by distributing message traffic across multiple nodes and optimizing resource utilization.
Websocket	High	Websocket protocol is designed for high scalability. It supports persistent connections and enables real-time bidirectional communication. Websocket-based applications can scale horizontally by distributing client connections across multiple servers and using load balancing techniques. With proper architecture and infrastructure, Websocket applications can handle high levels of concurrent connections and scale efficiently.
SOAP	High	SOAP operates over HTTP/HTTPS and can be scaled similarly to HTTP-based systems. By employing load balancing and distributed server configurations, SOAP-based systems can handle a moderate to high level of scalability. Additionally, SOAP services can be designed using SOA (Service-Oriented Architecture) principles, allowing for modular and scalable service compositions.
REST	High	REST APIs can be highly scalable due to their stateless nature and the use of HTTP as the underlying protocol. RESTful architectures can take advantage of horizontal scaling, load balancing, and

		caching mechanisms to handle high levels of concurrent requests. With proper design and implementation, RESTful systems can achieve high scalability and accommodate large user bases.
Multicast UDP	High	Multicast UDP offers high scalability for multicast communication. It allows for efficient one-to-many or many-to-many communication by sending a single packet to multiple recipients. Multicast UDP can scale to a large number of recipients without significant performance degradation.

3.2.2.12 Interoperability

Interoperability refers to the ability of a transmission protocol to work seamlessly with other systems and technologies, regardless of their platform or language. This includes features such as standardized protocols and APIs.

To which extent does the solution support interoperability?

Protocol	Interoperability	Justification
HTTP	High	HTTP has widespread adoption and support across various platforms and programming languages. It is a standard protocol used for web communication and is supported by most web servers, web browsers, and programming frameworks. This wide adoption and compatibility enable seamless communication between different systems and facilitate the interoperability of HTTP-based applications.
HTTPS	High	HTTPS inherits the interoperability of HTTP. It is widely supported by web servers, web browsers, and programming libraries, making it compatible with various systems and platforms. The use of SSL/TLS encryption adds an extra layer of security but does not significantly impact the interoperability of the underlying HTTP protocol. As a result, HTTPS-based applications can communicate effectively with other systems that support HTTPS.

MQ	High	MQ systems are designed for interoperability, allowing different applications and systems to exchange messages reliably and asynchronously. MQ implementations conform to messaging standards such as JMS (Java Message Service) or AMQP (Advanced Message Queuing Protocol), ensuring interoperability between applications developed in different programming languages and running on various platforms. This enables seamless integration and communication between heterogeneous systems.
FTP	Medium	FTP though supported by a wide range of client and server software, making it compatible with many systems, its interoperability can be affected by issues such as firewall restrictions, security configurations, and differences in server implementations. While FTP is widely adopted, its interoperability may require additional configuration or compatibility checks to ensure seamless communication between different systems.
FTPS	Medium	FTPS inherits the interoperability characteristics of FTP. It is supported by various FTP client and server software, making it compatible with many systems. However, like FTP, FTPS interoperability can be influenced by firewall restrictions, security configurations, and differences in server implementations. While widely supported, FTPS may require additional configuration or compatibility checks for seamless communication between different systems.
SFTP	High	SFTP is widely supported by SSH servers and client software, ensuring compatibility across different platforms and systems. The standardization of SSH ensures that SFTP implementations across different platforms and systems adhere to the same set of protocols and specifications, enabling seamless interoperability.
AS2	High	AS2 is based on widely adopted internet standards such as HTTP, SSL/TLS, and MIME (Multipurpose Internet Mail Extensions). AS2 supports the secure and reliable exchange of business documents between different trading partners, regardless of their platform, programming

		language, or system. The use of standard protocols and encryption ensures seamless interoperability.
Websocket	High	Websocket protocol provides high interoperability as it is supported by major web browsers, servers, and programming languages. Websocket is based on a standard protocol and follows a well-defined handshake process, allowing for bidirectional communication between clients and servers. This widespread adoption and compatibility make Websocket highly interoperable.
SOAP	High	SOAP follows a standardized XML-based messaging format and can be implemented in various programming languages and platforms. SOAP interoperability is achieved through well-defined protocols and standards, enabling communication between different systems regardless of the underlying technology stack. This makes SOAP highly interoperable and suitable for building distributed applications that can exchange data across diverse platforms.
REST	High	REST is designed for high interoperability. It leverages widely adopted web standards such as HTTP, URI (Uniform Resource Identifier), and JSON (JavaScript Object Notation). RESTful APIs can be consumed by any client that understands these standard protocols, making them highly interoperable. REST's simplicity, scalability, and reliance on standard protocols contribute to its widespread adoption and compatibility across different systems and platforms.
Multicast UDP	Medium	While UDP itself is a widely supported transport protocol, multicast functionality may vary across network devices and configurations. Multicast communication requires network infrastructure support, and not all systems and networks are configured to handle multicast traffic. Achieving interoperability with Multicast UDP may require additional configuration and compatibility checks to ensure that all participating systems and networks can effectively communicate using multicast.

3.2.2.13 Manageability

Manageability refers to the ease with which a transmission protocol can be managed and configured, including features such as monitoring, logging, and troubleshooting. The protocol should be easy to use and configure, with comprehensive documentation and support resources available.

To which extent does the solution afford manageability?

Protocol	Grade	Justification
HTTP	High	HTTP has high manageability due to its widespread adoption and well-established tools and technologies for managing web servers, such as Apache HTTP Server, Nginx, and Microsoft IIS. These tools provide extensive configuration options, monitoring capabilities, and logging features, making it easier to manage and troubleshoot HTTP-based applications.
HTTPS	High	HTTPS inherits the manageability characteristics of HTTP. The same tools and technologies used for managing HTTP servers can be applied to HTTPS servers. The added security layer of SSL/TLS encryption does not significantly impact the manageability aspects of the underlying HTTP protocol. Therefore, HTTPS-based applications can benefit from the same management tools and techniques available for HTTP, ensuring high manageability.
MQ	High	MQ systems typically come with management tools that allow administrators to monitor and manage message queues, configure messaging rules, and track message flows. These tools provide comprehensive control and visibility over the messaging infrastructure, enabling efficient management and troubleshooting of message-based applications.
FTP	Medium	FTP servers often come with management interfaces or command-line tools that allow administrators to configure user accounts, set access permissions, and monitor file transfers. However, managing FTP servers may require some technical expertise and familiarity with FTP server configurations. Third-party FTP server management tools and GUI-based FTP clients can simplify the

		management tasks and provide additional features for monitoring and administration.
FTPS	Medium	FTPS inherits the manageability characteristics of FTP. The same management tools and techniques used for FTP can be applied to FTPS servers. However, additional considerations may be required for managing SSL/TLS certificates, encryption settings, and security configurations. While manageable, FTPS may involve more complexity in terms of managing security aspects compared to standard FTP.
SFTP	High	SFTP servers can be managed using SSH server management tools, which provide configuration options, user management, logging capabilities, and security settings. These tools allow administrators to manage SFTP server settings and monitor file transfers. Additionally, SFTP clients often include user-friendly interfaces for managing remote file transfers, enhancing the overall manageability of SFTP-based systems.
AS2	High	AS2 provides high manageability through its support for various message transfer options and extensive configuration settings. AS2 implementations typically offer management interfaces or administrative tools that allow users to configure trading partner profiles, encryption settings, digital certificates, and message tracking. These management features provide control and visibility over AS2 message exchanges, ensuring efficient management and monitoring of data transfer processes.
Websocket	Medium	Websocket servers often provide management interfaces or APIs for monitoring and controlling WebSocket connections, such as tracking active connections, managing events, and handling resource allocation. While there are some management tools available, they may not be as extensive or mature as those for HTTP or other established protocols. However, with the growing adoption of Websocket, the availability of management tools and frameworks is increasing, enhancing its manageability.
SOAP	Medium	SOAP-based applications can be managed using SOAP toolkits and development frameworks that provide management interfaces, debugging capabilities, and monitoring features. These tools enable administrators to manage SOAP endpoints, configure web services,

		and monitor message exchanges. Additionally, SOAP-based services can integrate with existing enterprise management systems, such as service registries and governance frameworks, to enhance their manageability.
REST	Medium	Implementation of REST architecture added complexity over HTTPS. REST provides guidelines for structuring APIs and interactions between clients and servers. Implementing RESTful APIs requires careful design and adherence to these principles, which can introduce some complexity compared to HTTPS.
Multicast UDP	Medium	Setting up and managing multicast groups, addressing, and routing can be more complex compared to unicast protocols. The manageability of Multicast UDP protocol is considered as medium as the advancements in network management tools and protocols have improved the overall manageability of multicast networks. With proper configuration, monitoring, and troubleshooting practices, network administrators can effectively manage and maintain multicast deployments.

3.2.2.14 Data integrity

Data integrity refers to the assurance that data is not altered or corrupted during transmission, storage or processing.

To which extent does the solution ensure data integrity?

Protocol	Grade	Justification
HTTP	Low	HTTP is a stateless protocol primarily focused on data transfer and does not include built-in mechanisms for ensuring data integrity during transmission. However, additional measures such as checksums or digital signatures can be implemented at the application layer to verify data integrity.
HTTPS	High	HTTPS provides high data integrity by encrypting the data transmission and using cryptographic mechanisms to ensure data integrity. SSL/TLS protocols use digital certificates, encryption algorithms, and integrity checks (e.g., message digests) to protect

		data from unauthorized modification or tampering during transmission. The use of SSL/TLS in HTTPS ensures that data integrity is maintained.
MQ	High	MQ with TLS (Transport Layer Security) provides high data integrity. TLS is a cryptographic protocol that ensures secure communication and protects data integrity. It uses encryption and integrity checks to guarantee that the messages exchanged through the MQ system remain intact and unaltered during transmission.
FTP	Medium	FTP does not inherently provide data integrity features. However, FTP supports the use of checksums or integrity checks as an optional measure to verify file integrity during transfer. By calculating and comparing checksums at the source and destination, the integrity of the transferred file can be verified.
FTPS	High	FTPS inherits the data integrity features of FTP, combined with the security provided by SSL/TLS encryption. By using SSL/TLS protocols, FTPS ensures the confidentiality and integrity of data during transmission. The encryption and cryptographic mechanisms used in SSL/TLS provide robust protection against unauthorized modification or tampering of data.
SFTP	High	SFTP encrypts the data transmission and includes built-in mechanisms for verifying the integrity of the transferred files. Through the use of cryptographic measures and integrity checks, SFTP ensures that data remains unchanged during transit. The combination of encryption and integrity checks results in a high level of data integrity in SFTP.
AS2	High	AS2 provides high data integrity by incorporating digital signatures and encryption in the message exchange process. AS2 messages can be digitally signed to ensure the integrity of the data during transmission, and encryption can be applied to protect the confidentiality and integrity of the message content. These cryptographic measures guarantee the integrity of the data exchanged using AS2, warranting a high rating in this category.
Websocket	High	Websocket with HTTPS inherits the data integrity features of HTTPS. By using HTTPS as the underlying transport layer for Websocket, data transmitted through Websocket is encrypted and

		protected by SSL/TLS protocols. This ensures the integrity of the data during transmission, preventing unauthorized modification or tampering.
SOAP	High	By leveraging HTTPS as the underlying transport layer, SOAP messages are encrypted and protected by SSL/TLS protocols. This guarantees the integrity of the message content during transmission, ensuring that it remains unchanged and protected from unauthorized modification.
REST	High	REST with HTTPS inherits the data integrity features of HTTPS. By utilizing HTTPS as the transport layer, RESTful APIs benefit from encryption and cryptographic measures provided by SSL/TLS protocols. This ensures the integrity of data during transmission, protecting it from unauthorized modification or tampering.
Multicast UDP	High	Multicast UDP does not inherently provide data integrity features. However, when Multicast UDP is used in conjunction with the IPsec, the data integrity can be considered high. IPsec is a suite of protocols used to provide security services for IP network communications, including data integrity, confidentiality, and authentication.

3.2.2.15 Open solutions

Transmission protocols must be available as an open solution. The latter is understood as a solution, which allows the users to benefit from the right to use, change and distribute the solution without any restrictions and free of charge. Further information is provided in the appendix on the governance model surrounding the standardisation of the shortlisted solutions.

Can the solution be considered as an open solution?

Protocol	Grade	Justification
HTTP	Yes	HTTP is an open and widely adopted protocol. Its specifications are openly available and well-documented, allowing developers to implement HTTP-based solutions using a variety of programming languages and frameworks. There are numerous open-source

		libraries and tools available to work with HTTP, making it highly accessible and customizable.
HTTPS	Yes	HTTPS is built upon the open nature of HTTP and extends it with SSL/TLS encryption. While the SSL/TLS protocols themselves are not open, the implementation of HTTPS, including SSL/TLS libraries and tools, is widely available as open-source solutions. Developers can access and utilize these open-source implementations to incorporate HTTPS into their applications and systems.
MQ	No	MQ is typically associated with proprietary solutions provided by messaging middleware vendors. While there might be open-source implementations or alternative messaging protocols available, the majority of MQ implementations are vendor-specific and may require proprietary software or licenses, resulting in a lack of open solution availability in this context.
FTP	Yes	FTP has been around for a long time and is widely supported by various open-source solutions. There are numerous open-source FTP server and client implementations available, making it easy to set up FTP servers and develop FTP-based applications using open-source tools and libraries.
FTPS	Yes	FTPS builds upon the open nature of FTP and incorporates SSL/TLS encryption. Like FTP, there are several open-source FTPS server and client implementations available. These open solutions provide the ability to secure FTP connections using SSL/TLS encryption while utilizing open-source tools and libraries.
SFTP	Yes	SFTP is primarily associated with the SSH (Secure Shell) suite of protocols, which provides secure file transfer functionality. While SSH implementations and libraries are open-source, the specific support and availability of open-source solutions tailored for SFTP may vary. However, there are open-source SFTP server and client implementations available, making it possible to work with SFTP in an open solution environment.
AS2	Yes	AS2 is a widely adopted standard for secure and reliable data exchange. While there are proprietary AS2 solutions, there are also open-source implementations and tools available. These open

		solutions enable developers to work with AS2 in an open and customizable manner, making it an open solution.
Websocket	Yes	Websocket, when used over HTTPS, leverages the open nature of both HTTP and SSL/TLS. There are numerous open-source libraries and frameworks available for developing Websocket-based applications, and the integration with HTTPS allows for secure and encrypted communication. The availability of open-source implementations for Websocket over HTTPS makes it an open solution.
SOAP	Yes	SOAP is a messaging protocol commonly used in web services. When used over HTTPS, SOAP benefits from the open nature of HTTP and the security provided by SSL/TLS. There are open-source SOAP libraries and frameworks available that enable developers to implement SOAP-based solutions in an open and customizable manner. The availability of open-source SOAP implementations over HTTPS makes it an open solution.
REST	Yes	When used over HTTPS, REST benefits from the open nature of HTTP/HTTPS and the security provided by SSL/TLS. There are numerous open-source frameworks and libraries available for developing RESTful APIs and services. The availability of these open-source solutions makes REST over HTTPS an open solution.
Multicast UDP	Yes	Multicast UDP is an open protocol that allows efficient one-to-many communication, commonly used for streaming media.

3.2.2.16 Level of adoption

Transmission protocols must show a significant level of adoption in other regulatory framework, in Europe, and in other jurisdictions.

The level of adoption has been determined based on the responses to the questionnaires from the 11 responding entities, including 9 market data contributors and 2 prospective CTP candidates. The grading is resulting from the following approach:

- For each possible level of usage of a transmission protocols (i.e High, Medium, Low), one point is added depending on the individual questionnaire’s responses. This provides an intermediate score by transmission protocol and level of usage.

- Then for each transmission protocols, a total score is computed by summing up the weighted intermediate scores (i.e the intermediate score under “high” level of usage are multiplied by 10; the one under “medium” by 5; and the one under “low” by 1).
- Finally, the total score is converted into the grade “High” when above 60; into the grade “Medium” when between 30 and 60; and into the grade “Low”, when below 30.

What is the level of adoption of the transmission protocol?

Protocol	Grade	Justification
HTTP	Low	None of the data contributors is using plain HTTP for market data distribution for the very obvious reason that it is unsecure.
HTTPS	Medium	6 out of 11 respondents have said to use HTTPS mostly for internal communications and for websites. The level of adoption has been set as medium based on the overall score of 52 out of 100.
MQ	Medium	5 out of 11 respondents have said to use MQ for internal messaging and to integrate with investment firms. Its overall score is 26 thus setting the level of adoption as medium.
FTP	Low	FTP is not used by any data contributors.
FTPS	Low	FTPS is not used by any data contributors.
SFTP	Medium	8 out of 11 respondents have said to use SFTP for file transfer and reporting though it is not suitable for real-time market data distribution. It has an overall score of 55 out of 100 thus the level of adoption is medium.
AS2	Low	AS2 is not used by any data contributors.
Websocket	Medium	6 out of 11 respondents have said to use websocket for web-based APIs and cloud-based market data distribution. It has an overall score of 41 out of 100 thus the level of adoption is medium.
SOAP	Low	SOAP is being used only at 2 of the market data contributors thus leaving the level of adoption as low.

REST	Medium	5 out of 11 respondents have said to use REST for internal communications and web APIs. Its overall score is 31 out of 100 thus the level of adoption is medium.
Multicast UDP	High	Multicast UDP has the highest level of adoption as it is being used by 11 out of 11 respondents for low latency market data feeds and internal communications and especially for dissemination. It has an overall score of 100.

3.2.2.17 Implementation feasibility

Impact in terms of costs to implement & run as well as implementation timing need to be assessed.

What is the level of implementation feasibility of the solution?

Protocol	Grade	Justification
HTTP	High	HTTP is a widely adopted protocol with extensive documentation, well-supported libraries, and frameworks available for various programming languages and platforms. It follows a straightforward request-response model, making it relatively easy to implement.
HTTPS	High	HTTPS, while it requires additional security measures compared to HTTP, the process is well-documented and widely supported. Many programming languages provide libraries and tools for handling secure communication, making it relatively straightforward to implement.
MQ	Medium	Implementation feasibility for MQ is medium. The specific implementation of message queuing systems may vary, and there are multiple options available, each with its own set of APIs and libraries. Implementing MQ requires familiarity with the chosen system, but various libraries and frameworks exist to facilitate integration.
FTP	Medium	FTP has been in use for a long time, and there are well-established libraries and tools available for handling FTP transfers. However, implementing FTP requires understanding and handling of the FTP

		protocol intricacies and the compatibility, which can add some complexity to the implementation process.
FTPS	Medium	Implementation feasibility for FTPS (File Transfer Protocol Secure) is similar to FTP. It requires the same understanding of the FTP protocol, but with additional security measures. Libraries and tools exist that provide support for FTPS, but configuring the security aspects and ensuring compatibility with different FTPS servers may require some additional effort.
SFTP	High	SFTP provides secure file transfer capabilities over SSH, and various libraries and implementations are available that make it relatively easy to integrate SFTP functionality into applications. Familiarity with SSH server configuration and SSH libraries is necessary for successful implementation.
AS2	Low	Implementation feasibility for AS2 is relatively low. AS2 involves specific software and integration requirements, such as digital certificates and specific message formats. Implementing AS2 requires working with specialized AS2 software and configuring it to integrate with existing systems, which can be complex and require specific expertise.
Websocket	Medium	While Websockets are supported by most modern web browsers and server-side frameworks, implementing them requires support on both the server and client sides. Libraries and frameworks are available to facilitate Websocket implementation, but some platform-specific considerations may be involved.
SOAP	Medium	SOAP relies on XML-based specifications and requires working with XML parsing and generation libraries. While SOAP frameworks and tools are available for various programming languages, implementing SOAP involves additional complexity compared to REST due to the XML-related aspects and adherence to specific standards.
REST	High	REST is based on standard HTTP methods and commonly uses JSON or XML for data exchange, which are well-supported in most programming languages. There are numerous libraries and frameworks available that simplify RESTful API development and

		make implementation relatively straightforward, as long as the underlying HTTP infrastructure is available.
Multicast UDP	Medium	Multicast UDP allows efficient one-to-many communication but requires network-level multicast support. While Multicast UDP is a standard feature of most modern networks, configuring routers and switches to support multicast can be involved. Implementing Multicast UDP also requires handling the UDP protocol and managing packet delivery and synchronization in a multicast group. Libraries and frameworks are available to facilitate the implementation process.

3.2.3 Outcome of the technical assessment

In order to evaluate the technical capabilities of each transmission protocol for the CT implementation, a weighted score was assigned based on the performance against the identified criteria. The assessment grid provides further information and details regarding the scores and criteria used during the assessment process.

The table below provides a summary of the scoring for each transmission protocol per criteria, as well as the overall score.

FIGURE 13 - OUTCOME OF THE TRANSMISSION PROTOCOLS' ASSESSMENT

Criteria	Web-socket	SFTP	MQ	HTTPS	Multi-cast	REST	SOAP	AS2	FTPS	HTTP	FTP
Data confidentiality	100	100	100	100	100	100	100	100	100	0	0
Authentication	100	100	100	100	100	100	100	100	100	0	0
Authorization	100	100	100	100	100	100	100	100	100	0	0
Non-repudiation	60	60	60	60	60	60	60	60	60	0	0
Encryption	100	100	100	100	100	100	100	100	100	0	0
Secure file transfer	20	20	20	20	20	20	20	20	20	0	0
Latency Optimization	100	100	100	50	100	50	50	50	50	50	50
Throughput	100	50	100	50	100	50	50	50	100	100	100

Connection setup time opt.	50	50	50	50	100	50	50	50	50	100	100
Reliability	100	100	100	100	10	100	100	100	100	50	50
Scalability	60	30	60	60	60	60	60	30	30	60	30
Interoperability	100	100	100	100	50	100	100	100	50	100	50
Manageability	30	60	60	60	30	30	30	60	30	60	30
Data Integrity	100	100	100	100	100	100	100	100	100	10	50
Open solution	60	60	0	60	60	60	60	60	60	60	60
Level of adoption	30	30	30	30	60	30	6	6	6	6	6
Implementation Feasibility	30	60	30	60	30	60	30	6	30	60	30
Total Score	1240	1220	1210	1200	1180	1170	1116	1092	1086	656	556

Legend:

🟧 Scores equivalent to "Low" or "No"

🟨 Scores equivalent to "Medium"

🟩 Scores equivalent to "High" or "Yes"

Based on the scoring provided in the table above, the transmission protocols are ranked as follows (from highest to lowest total score)

Figure 14 - Ranking of the transmission protocols

Rank	*Protocol*	*Total Score*
1	Websocket	1240
2	SFTP	1220
3	MQ	1210
4	HTTPS	1200
5	Multicast	1180
6	REST	1170
7	SOAP	1116
8	AS2	1092
9	FTPS	1086
10	HTTP	656
11	FTP	556

It can be observed that Websocket has topped the list with a score of 1240 out of 1380. Notably, SFTP, MQ, HTTPS, Multicast, and REST protocols closely trail behind. It is worth highlighting that MQ and Websocket outperformed other TCP-based protocols, such as HTTPS, SFTP, and REST, in terms of latency optimization and throughput. Additionally, their performance in these aspects is on par with that of the Multicast UDP protocol.

The top 6 transmission protocols namely HTTPS, MQ, SFTP, Websocket, REST and Multicast has secured same score on the following 7 criteria out of 17.

Figure 15 – Equally performing criteria for the top 6 transmission protocols

Criteria	Weighting Classification	HTTPS	MQ	SFTP	Websocket	REST	Multicast
Data confidentiality	Mandatory	100	100	100	100	100	100
Authentication	Mandatory	100	100	100	100	100	100
Authorization	Mandatory	100	100	100	100	100	100
Non-repudiation	Recommended	60	60	60	60	60	60
Encryption	Mandatory	100	100	100	100	100	100
Secure file transfer	Nice to have	20	20	20	20	20	20
Data Integrity	Mandatory	100	100	100	100	100	100

The following criteria showcases the difference in performance of these data formats.

Figure 16 - Differentiating criteria for the top 6 transmission protocols

Criteria	Weighting Classification	HTTPS	MQ	SFTP	Websocket	REST	Multicast
Latency Optimization	Mandatory	50	100	100	100	50	100
Throughput	Mandatory	50	100	50	100	50	100
Connection setup time Optimization	Mandatory	50	50	50	50	50	100
Reliability	Mandatory	100	100	100	100	100	10
Scalability	Recommended	60	60	30	60	60	60
Interoperability	Mandatory	100	100	100	100	100	50

Manageability	Recommended	60	60	60	30	30	30
Open solution	Recommended	60	0	60	60	60	60
Level of adoption	Recommended	30	30	30	30	30	60
Implementation Feasibility	Recommended	60	30	60	30	60	30

Nevertheless, to measure the real-time capability of transmission protocols which is one of the critical and key requirements of CT, the following 3 criteria can be considered.

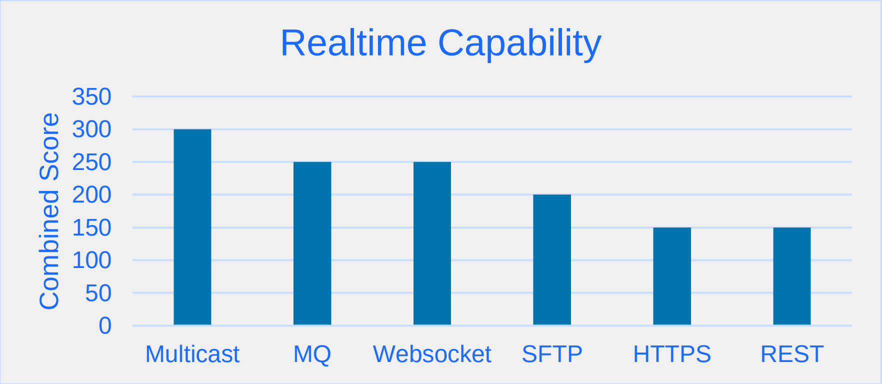
1. Latency optimization
2. Throughput
3. Connection setup time optimization

Here is the score of the top 6 transmission protocols on these 3 criteria to assess their real-time capability.

FIGURE 17 - REAL TIME SCORING OF THE TOP 6 TRANSMISSION PROTOCOLS

Criteria	HTTPS	MQ	SFTP	Websocket	REST	Multicast
Latency Optimization	50	100	100	100	50	100
Throughput	50	100	50	100	50	100
Connection setup time Optimization	50	50	50	50	50	100
Total score	150	250	200	250	150	300

FIGURE 18 - RANKING OF THE TOP 6 TRANSMISSION PROTOCOLS BASED ON REAL-TIME CRITERIA

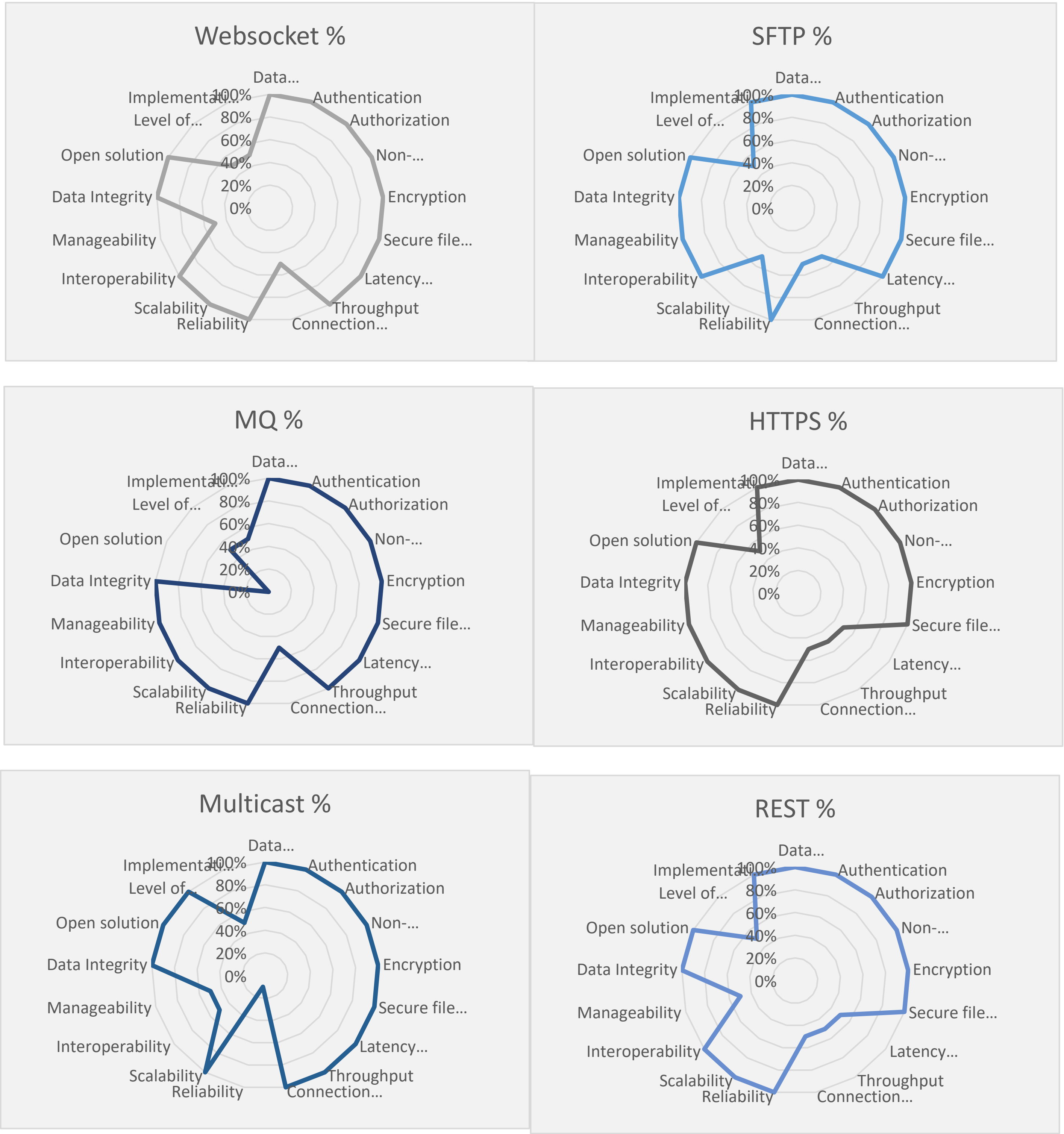


Multicast protocol holds the leading position in terms of real-time capability when compared to other TCP-based protocols. This can primarily be attributed to the fact that Multicast is a connectionless protocol, which confers advantages in terms of speed and efficiency for real-time transmission. In contrast, the other TCP-based protocols are connection-oriented, which introduces additional overhead and complexity in establishing and maintaining connections.

Based on this initial assessment, the following six transmission protocols will proceed to the next level of evaluation for determining their suitability.

1. Websocket
2. SFTP
3. MQ
4. HTTPS
5. Multicast
6. REST

Figure 19 - Top 6 transmission protocols unweighted score performance (%)



4 Evaluation of the suitability

This chapter aims at bringing an outside-in perspective to the technical assessment of the most suitable solution for the purpose of CTP market data collection and for a potential revision of the choices for formats across other regulatory datasets.

4.1 Introduction

Suitability of a solution can be defined as the ability of being appropriate for a particular purpose. In the information technology industry, solution's adequacy is approached through "the context-problem-solution triplet" which aims at the resolution of an occurring problem within a given context[45].

Consequently, it appears important to contextualize the outcome of the technical assessment and therefore to scrutinize the particularities of the use cases underpinning the study's objectives. This exercise is nonetheless constrained on one hand by the unpredictability of future regulatory data reporting needs and on the other hand, by the uncertainty of CTP fundamental choices on the technical architecture. For mitigating the latter, the study intends to leverage the insights collected from CTP ecosystem's stakeholders (duly flagged below).

In the light of the above, it is proposed to examine the two main different use cases which can be derived from the initial goals of the study. The first use case is the market data collection for the purpose of the CTP, where a notable distinction should apply whether it is for equities or for non-equities. The second use case is the regulatory data collection for generic purposes.

4.2 Market data collection for the purpose of the CTP

This section aims at highlighting the key principles identified for the functioning of the CTP and evaluates the suitability of the data formats and transmission protocols shortlisted in the technical assessment.

4.2.1 Contextualization

The contextualisation of market data collection for the purpose of the CTP can be conducted in the light of the CTP specific requirements, which have been identified as following:

[45] Avgeriou, Paris; Zdun, Uwe (2005), Architectural patterns revisited:a pattern language, UVK Verlagsgesellschaft

- Market data collection by CTP must cover the financials instruments in scope of MIFIR review, precisely the 4 asset classes in scope, namely shares, ETFs, bonds and derivatives. At this point, it is imperative to stress the specificities under which the trading industry operates either equities or non-equities transactions. To this respect, it is well established that equities market is more sensitive to latency than non-equities trading.

FIGURE 20 - EXTERNAL FEEDBACK: HOW TO APPROACH ASSET CLASSES SPECIFICITIES

Some of the consulted stakeholders highlighted that many trading venues are operating both on equities and non-equities markets. With regards to CTP technical set up, they expressed a clear preference for a common solution across asset classes for cost efficiency reasons.

Interestingly, one stakeholder reported a common acknowledgement across the trading industry that CTP is not intended to serve very latency sensitive use case, which will very likely continue relying on direct data feeds.

- Market data collection by CTP must ensure high quality data. In the absence of a formal definition, reference is made to the commonly agreed meaning, which describes data quality in terms of data completeness, consistency, availability, accuracy, validity, and uniqueness.
- The CTP and consequently market data collection must be performed as close to real-time as technically feasible. As a benchmark, the current regulatory technical standards set as a maximum timing threshold, real-time publication, within 1 minute for shares and ETFs[46]; and within 5 minutes for bonds & derivatives[47].

[46] Art. 14, RTS 1

[47] Art. 7 §4 b), RTS 2

Figure 21 - External feedback: Latency

> *The level of latency is perceived by the involved external stakeholders as a critical factor which determines the business value of market data.*
>
> *One of the key driving factors for overall latency is the network latency and the geographical distance between the publisher and data consumer. To achieve a low latency, it is important to have a large network bandwidth that would range between 1 Gbps to 10 Gbps.*
>
> *The level of latency that is currently achieved at the side of the various data contributors ranges* ***from less than 1 millisecond to about 1 second****. According to external feedbacks, the above-mentioned benchmark regulatory values of 1 minute and 5 minutes cannot be considered as close to real time as it is technically feasible.*

- Market data collection by CTP must enable the collection of all market data (currently defined in MIFIR RTS) and regulatory data from all market data contributors.
- Market data collection by CTP is required to meet non-functional requirement on volumes, which are here defined from two perspectives:
    - The CTP will be required to enable connectivity for a large volume of data contributing entities - estimated at 313 for bonds, and 195 for equities.
    - Market data collection by CTP must provide the sufficient messages payload for managing transactions spikes.

- Market data collection by CTP must meet non-functional requirement on security, which is defined as per common industry understanding.

FIGURE 22 - EXTERNAL FEEDBACK: SECURITY

Authentication:

Most of the stakeholders have shared to use the following authentication mechanisms widely within their organization: User credentials and Digital certificates.

For the CTP use case, as the connection between the contributors and CT would be a system-to-system integration, digital certificate is considered to be appropriate over the user based or biometric credentials.

Encryption:

Encryption of data in transit and at rest is crucial to protect private and sensitive information. Transport Layer Security (TLS) is widely adopted as the encryption mechanism with most of the stakeholders for publishing data over public network (internet).

However, some of the stakeholders have expressed not using encryption at least for the market data publication rather publishing data on completely private networks.

Digital signature and non-repudiation:

Digital signatures are expected to add complexity and introduce certain overhead and performance cost relative to the additional protection it would offer. Most of the stakeholders have responded as not using digital signatures with only one has said to use digital signature for transference of data.

Most of the stakeholders have responded to maintain audit trails of all data received from contributors to server the purpose of non-repudiation. Any additional non-repudiation mechanism is considered to be unwarranted and add additional complexity to the solution.

- Market data collection by CTP is required to happen with an open solution, especially data format & transmission protocols must afford change control and be free of charge.
- The CTP is required to meet CTP organizational requirements – at the time of the study, potential interdependencies with the selection of the data format and transmission protocols are considered with regards to CTP transparency reporting obligation and the requirement for revenues redistribution – yet to be confirmed.

4.2.2 Evaluation of the suitability of the shortlisted data formats for CTP purpose

The technical assessment concluded on the shortlisting of the following five data formats ranked in the following order: JSON; ASN.1; Protocol buffer; FAST; and SBE. Yet the final scores are tight and there is no outstanding solution with the given weighting. At this stage of the evaluation, it is then suggested to refer to the differentiating criteria and factor in the advantages and disadvantages of each solution in the specific context of the CTP's use case.

Figure 23 (11 bis) - Differentiating criteria for the top 5 data formats

Criteria	Weighting Classification	JSON	ASN.1	Protocol Buffer	FAST	SBE
Ease of use	Recommended	60	6	30	6	6
Digital signature	Nice to have	0	20	0	0	0
Network overhead optimization	Mandatory	50	100	100	100	100
Flexibility	Recommended	60	60	30	6	30
Level of adoption	Recommended	30	6	6	60	30
Implementation Feasibility	Recommended	60	30	60	30	30
Protocol Compatibility	Recommended	60	30	30	30	30
ISO 20022 Compatibility	Recommended	60	60	0	0	0

4.2.2.1 JSON

Following are the advantages and disadvantages of choosing JSON as the data format for CTP use case.

Advantages	Disadvantages
✓ High ease of use for developers due to its simple and intuitive syntax. ✓ Flexible data format that can represent complex data structures. ✓ Compatible with a wide range of protocols. ✓ Feasible to implement in applications and widely supported by libraries and frameworks. ✓ Compliant with ISO 20022 standard ✓ Adopted across different data contributors mainly for internal messaging and web APIs	× Lack of built-in support for digital signatures. 1. × JSON's textual format result in high network overhead compared to binary formats, leading to larger data sizes and increased transmission times, especially for large datasets.

JSON's high network overhead makes it less suitable choice for Consolidated Tape which requires efficient data transmission to handle the vast volume of information in real-time. Given network overheard optimization is flagged as a mandatory criterion, JSON is considered as **partially suitable** for CTP's purpose.

4.2.2.2 ASN.1

Following are the advantages and disadvantages of choosing ASN.1 as the data format for CTP use case.

Advantages	Disadvantages
✓ ISO 20022 compliant, ensuring conformity with international standards for data interchange. ✓ Provides support for digital signatures, enhancing data security and authenticity. ✓ Low network overhead due to its compact binary encoding, leading to efficient data transmission and reduced bandwidth usage.	× Low ease of use for developers due to its complex encoding and decoding process. 2. × Very low level of adoption across data contributors and financial industry, potentially limiting its widespread compatibility and ease of data exchange.

While ASN.1 is ISO 20022 compliant and offers advantages like support for digital signatures and low network overhead, its low ease of use and limited adoption across data contributors and financial industry are important factors with an intermediate weight to consider it as **partially suitable** for CTP's purpose.

4.2.2.3 Protocol buffer

Following are the advantages and disadvantages of choosing Protocol buffer as the data format for CTP use case.

Advantages	Disadvantages
✓ Medium ease of use for developers due to its simple and concise message definition syntax. ✓ Offers flexibility in representing complex data structures. ✓ Low network overhead due to its compact binary representation, leading to efficient data transmission and reduced bandwidth usage. ✓ High implementation feasibility with code generation capabilities for various programming languages.	× Low level of adoption across data contributors. × Not ISO 20022 compliant. × No built-in support for digital signatures. × Uncertainty on the long term maintenance given it was initially developed for Google internal usage

Protocol buffer has satisfied most of the critical requirements of CTP like Scope of data fields to ensure high quality data, high parsing speed, serialization and low network overhead for high volume and low latency transmission. However, it is not widely adopted across the data contributors for trade data transmission though a few of the data contributors use it for market data distribution and high-performance use case. Consequently, Protocol buffer is considered as **partially suitable** for CTP's purposes.

4.2.2.4 FAST

Following are the advantages and disadvantages of choosing FAST as the data format for CTP use case.

Advantages	Disadvantages
✓ High level of adoption, indicating wide acceptance and usage in financial industry for distribution of trade data. ✓ Low network overhead, making it suitable for high volume and low latency data transmission. ✓ Moderate implementation feasibility, meaning it can be implemented with reasonable effort. ✓ Protocol compatibility, as it can work well with various communication protocols.	✗ Low ease of use for developers due to its complex encoding and decoding process. ✗ Low flexibility in representing data structures compared to other formats. ✗ No built-in support for digital signatures ✗ Not ISO 20022 compliant

While FAST has certain disadvantages being not ISO compliant, offers low ease of use and low flexibility, they are not considered as critical requirement for CTP. Nevertheless, it is most suitable for high volume, low latency transmission of trade data with high quality and is widely used across the data contributors for disseminating trade data. While balancing the advantages and disadvantages, FAST should be considered as **mostly suitable** for CTP's purpose.

4.2.2.5 SBE

Following are the advantages and disadvantages of choosing SBE as the data format for CTP use case.

Advantages	Disadvantages
✓ Low network overhead, making it suitable for high volume and low latency data transmission. ✓ Moderate level of adoption for disseminating real-time trade data. ✓ Protocol compatibility, as it can work well with various communication protocols.	✗ Low ease of use for developers due to its complex encoding and decoding process. ✗ No built-in support for digital signatures. ✗ Not ISO 20022 compliant.

SBE has satisfied all the mandatory and critical requirements of CTP. Despite it has some disadvantages like no built-in support for digital signatures, not ISO compliant and low ease of use which are considered not critical for CTP, it is highly suitable for a high volume, low latency transmission of high quality trade data. Therefore, SBE should be considered as **mostly suitable** for CTP purpose.

4.2.3 Evaluation of the suitability of the shortlisted protocols for CTP purpose

The technical assessment concluded on the shortlisting of the following six transmission protocols ranked in the following order: Websocket; SFTP; MQ; HTTPS; Multicast and REST. Yet the final scores are tight and there is no outstanding solution with the given weighting. At this stage of the evaluation, it is then suggested to refer to the differentiating criteria and factor in the advantages and disadvantage of each solution in the specific context of the CTP's use case.

FIGURE 24 (16 BIS) - DIFFERENTIATING CRITERIA FOR THE TOP 6 TRANSMISSION PROTOCOLS

Criteria	Weighting Classification	HTTPS	MQ	SFTP	Websocket	REST	Multicast
Latency Optimization	Mandatory	50	100	100	100	50	100
Throughput	Mandatory	50	100	50	100	50	100
Connection setup time Optimization	Mandatory	50	50	50	50	50	100
Reliability	Mandatory	100	100	100	100	100	10
Scalability	Recommended	60	60	30	60	60	60
Interoperability	Mandatory	100	100	100	100	100	50
Manageability	Recommended	60	60	60	30	30	30
Open solution	Recommended	60	0	60	60	60	60
Level of adoption	Recommended	30	30	30	30	30	60
Implementation Feasibility	Recommended	60	30	60	30	60	30

4.2.3.1 Websocket

Following are the advantages and disadvantages of choosing Websocket as the transmission protocol for CTP use case.

Advantages	Disadvantages
✓ High Latency Optimization, ensuring minimal/no delays in data transmission. ✓ High Throughput, allowing for efficient handling of a large volume of trade data. ✓ High Reliability, offering dependable and consistent data delivery.	✗ Not very high on Connection setup time optimization. ✗ Not high level of adoption for trade data dissemination ✗ Relatively low on manageability compared to other protocols like HTTPS and MQ.

✓ High Scalability, enabling the system to handle increasing data loads and user demands. ✓ High Interoperability, as WebSocket is supported by most modern web browsers and widely used across platforms. ✓ Open Solution, as WebSocket is an open standard, ensuring wide accessibility and compatibility.	

Websocket due to its persistent connection is highly reliable and scalable. It scores well in latency optimization and throughput as the persistent connection eliminates the need for repeated connection setup reducing overhead and latency makes it suitable for real-time, high volume transactions. Nevertheless, it is not widely adopted for trade data feeds across market data contributors. Consequently, websocket should be considered as **partially suitable** for CTP's purposes.

4.2.3.2 SFTP

Following are the advantages and disadvantages of choosing SFTP as the transmission protocol for CTP use case.

Advantages	Disadvantages
✓ High on Latency Optimization, ensuring minimal delays in data transmission. ✓ High Reliability, offering dependable and secure data delivery. ✓ High Implementation Feasibility, meaning it can be implemented with relative ease. ✓ Open Solution, as SFTP is an open standard, ensuring wide accessibility and compatibility.	✗ Medium Throughput Limitation: While SFTP provides reasonable throughput, it may not match the high throughput capabilities of other protocols like WebSocket, MQ and multicast. ✗ Limited Scalability: SFTP may have limitations in scaling to handle extremely high data volumes or concurrent connections compared to more specialized protocols designed for high-frequency trading. ✗ Low Level of Adoption: SFTP is not as widely adopted for high volume and low latency trade data transmission, potentially leading to fewer available resources and support in this specific use case.

SFTP is highly reliable, interoperable, and manageable and is primarily designed for secure file transfer. Though the protocol is optimized for low latency, it is connection oriented and thus reduces the throughput making it less suitable for real-time, high volume trading data dissemination. It also has a low level of adoption across the market data contributors for trading data feeds. When compared to its peers, SFTP should be seen as **partially suitable** for CTP purposes.

4.2.3.3 MQ

Following are the advantages and disadvantages of choosing MQ as the transmission protocol for CTP use case.

Advantages	Disadvantages
✓ High Latency Optimization, ensuring minimal delays in data transmission. ✓ High Throughput, allowing for efficient handling of a large volume of trade data. ✓ High Reliability, offering dependable and guaranteed message delivery. ✓ High Scalability, capable of handling substantial data loads and accommodating increased demand. ✓ High Interoperability, as MQ solutions are available for various platforms and programming languages. ✓ Medium Manageability, providing a manageable level of complexity in managing message queues and configurations.	× Relatively low Level of Adoption: MQ solutions are not widely adopted for high volume and low latency trade data transmission with the data contributors. × Medium Implementation Feasibility, meaning that implementing an MQ-based solution may require moderate effort and expertise. × Moderate Connection Setup Time: Connection setup time is relatively higher compared to other protocols like multicast. × Many MQ solutions are proprietary, leading to potential vendor lock-in and limited flexibility in choosing the MQ implementation.

MQ has performed exceptionally well in latency optimization, throughput, reliability, and interoperability making it suitable for high volume, low latency distribution of high quality trade data. MQ solutions are mostly proprietary from software vendors and makes it less preferable for CTP. Nevertheless, MQ should be considered as **mostly suitable.**

4.2.3.4 HTTPS

Following are the advantages and disadvantages of choosing HTTPS as the transmission protocol for CTP use case.

Advantages	Disadvantages
✓ High Reliability, offering secure and reliable data transmission. ✓ High Scalability, capable of handling substantial data loads and accommodating increased demand. ✓ High Interoperability, as HTTPS is widely supported across different platforms and devices. ✓ High Implementation Feasibility, meaning that implementing HTTPS-based solutions is relatively straightforward. ✓ Open Solution, as HTTPS is a standardized and open protocol, ensuring compatibility and accessibility.	✗ Medium Manageability, providing a level of complexity in managing HTTPS connections and configurations. ✗ HTTPS is not widely adopted across data contributors for trade data distribution. ✗ Relatively low Latency Optimization, meaning HTTPS has some additional overhead compared to more lightweight protocols designed specifically for low latency data transmission. ✗ Relatively low Throughput, HTTPS may not match the high throughput capabilities of specialized protocols like WebSocket or multicast. ✗ Relatively low Connection Setup Time Optimization, establishing HTTPS connections may introduce some additional overhead.

While HTTPS is highly reliable, scalable and interoperable, it is not well suited for high volume, low latency trade data distribution as it is high in network latency, connection setup time and low throughput. For the same reason, it is not widely adopted across the market data contributors for trade data distribution. Consequently, HTTPS is evaluated as **partially suitable** for CTP's purposes.

4.2.3.5 Multicast

Following are the advantages and disadvantages of choosing Multicast as the transmission protocol for CTP use case.

Advantages	Disadvantages
✓ High Latency Optimization, ensuring minimal delays in data transmission. ✓ High Throughput, allowing for efficient handling of a large volume of trade data.	✗ Low Reliability: Multicast does not guarantee reliable data delivery. Packets can be lost without detection or retransmission. ✗ Medium Interoperability, as using Multicast would require additional configuration and

✓ No Connection Setup Time: Multicast does not require individual connections to be established, reducing initial overhead. ✓ High Scalability, capable of efficiently distributing data to multiple recipients simultaneously. ✓ Very High Level of Adoption: Multicast is widely used and well-established for high volume and low latency data distribution. ✓ Open Solution, as Multicast is a standardized and open network protocol, ensuring compatibility and accessibility.	compatibility check across participating systems. ✗ Medium Manageability, providing a moderate level of complexity in managing multicast configurations. ✗ Medium Implementation Feasibility, meaning that implementing multicast-based solutions may require moderate effort and network expertise.

Multicast is very well suited for high volume, low latency trade data distribution as it has very low latency and very high throughput. It is connectionless and therefore eliminates the extra overhead to setup a connection. Nevertheless, it is not reliable and consequently may jeopardize high data quality requirement although this can be overcome by use of audit trails/log capture and snapshots to trigger recovery mechanisms such as retransmission or error correction. For reference purpose, multicast is widely used across data contributors for market data dissemination, as well as by the U.S market data consolidation bodies (cf. appendix). Given Multicast is significantly lagging behind data reliability, which is a mandatory criterion, it is considered as **partially unsuitable** for CTP's data collection purposes.

4.2.3.6 REST

Following are the advantages and disadvantages of choosing REST as the transmission protocol for CTP use case.

Advantages	Disadvantages
✓ High Reliability, offering reliable data transmission over standard HTTP protocols. ✓ High Scalability, capable of handling substantial data loads and accommodating increased demand. ✓ High Interoperability, as REST is widely supported and can work with various platforms and programming languages.	✗ Relatively low Latency Optimization, meaning REST may introduce some additional overhead compared to more lightweight protocols designed specifically for low latency data transmission. ✗ Moderate Throughput: REST's reliance on HTTP can introduce some limitations on throughput compared to other specialized protocols for high-frequency trading.

✓ High Implementation Feasibility, meaning that implementing REST-based solutions is relatively straightforward. ✓ Open Solution, as REST is a standardized and open protocol, ensuring compatibility and accessibility. **3.**	× Relatively High Connection Setup Time: REST's request-response nature may lead to higher connection setup times compared to other protocols like WebSocket or Multicast. × Medium Manageability, providing a manageable level of complexity in managing RESTful APIs and configurations. × Relatively low Level of Adoption: REST is not widely for high volume, low latency trade data distribution.

While REST is highly reliable and scalable, interoperable with various platforms and easy to implement, it is relatively low on latency optimization, throughput and it involves extra connection overhead as it is connection-oriented making is less suitable for high volume, low latency trade data distribution. It is also not widely adopted across the market data distributors for trade data distribution. Acknowledging its intermediate scores against mandatory criteria, REST should be considered as **partially suitable** for CTP’s purposes.

4.3 Regulatory data collection for generic purposes

4.3.1 Contextualization

In order to satisfy other reporting obligations and provide a basis for a potential revision of ESMA technical choices, the following requirements are deemed relevant:

Data Format:

- **Ease of use –** Ease of use refers to how user-friendly and easy to understand the data format is for both developers and end-users. This includes features such as clear and concise syntax, well-defined data structures, and ease of implementation.
- **Flexibility –** The data format should be able to accommodate diverse data structures, adapt to varying data requirements, and support customization and extensibility.
- **Open solution –** Data format must afford change control and must be free of charge.
- **ISO 20022 compliancy –** The data format must be compliant with ISO 20022 messaging standard.

Transmission Protocol:

- **Scalability –** The transmission protocol should be able to handle increasing amounts of traffic and users without sacrificing performance or reliability. This includes features such as load balancing and distributed systems.
- **Open solution –** Transmission protocol must afford change control and must be free of charge.
- **Implementation Feasibility** - It assesses the level of effort, resources, and expertise required to integrate the transmission protocol effectively into the existing infrastructure or to develop a new system using the protocol.

4.3.2 Evaluation of the suitability of the data formats for generic purposes

Criteria	XML	JSON	CSV	FIXML	ASN.1	Protobuf	BSON	FAST	SBE	Tag Value
Ease of use	30	60	60	6	6	30	30	6	6	30
Flexibility	60	60	6	30	60	30	30	6	30	6
Open solution	60	60	60	60	60	60	60	60	60	60
ISO 20022 Compliancy	60	60	0	0	60	0	0	0	0	0

In the analysis of various data formats (XML, JSON, CSV, FIXML, ASN.1, Protocol Buffers, BSON, FAST, SBE, and Tag Value) for generic purpose reporting for ESMA, JSON stands out as the most suitable choice due to its excellent performance across all the identified key criteria.

Ease of Use: JSON excels in ease of use with its simple and human-readable syntax. Its lightweight and intuitive structure make it developer-friendly, allowing for quick and straightforward data manipulation and parsing. This ease of use translates to reduced development time and increased productivity.

Flexibility: JSON provides high flexibility in representing complex data structures, making it suitable for a wide range of reporting needs. It accommodates nested objects and arrays, enabling the representation of hierarchical and multi-level data without constraints. This flexibility is crucial for handling diverse financial data requirements.

Open Solution: JSON is an open standard, meaning it is not tied to any specific vendor or proprietary system. Its openness ensures broad compatibility across various platforms, systems, and programming languages, allowing for seamless integration and data exchange in a diverse ecosystem.

ISO 20022 Compliancy: JSON is ISO 20022 compliant which makes it a suitable choice for generic purpose reporting to ESMA.

4.3.3 Evaluation of the suitability of the protocols for generic purposes

Criteria	HTTP	HTTPS	MQ	FTP	FTPS	SFTP	AS2	Websocket	SOAP	REST	Multicast
Scalability	60	60	60	30	30	30	30	60	60	60	60
Open solution	60	60	0	60	60	60	60	60	60	60	60
Implementation Feasibility	60	60	30	30	30	60	6	30	30	60	30

In the analysis of various transmission protocols (HTTP, HTTPS, MQ, FTP, FTPS, SFTP, AS2, WebSocket, SOAP, REST, and Multicast) for generic purpose reporting to ESMA (European Securities and Markets Authority), three protocols stand out in terms of scalability, open solution, and implementation feasibility: HTTP, HTTPS, and REST.

Scalability: HTTP, HTTPS, and REST have demonstrated high scalability, enabling them to handle large volumes of data efficiently. These protocols are well-suited for generic purpose reporting, accommodating the data transmission needs of diverse financial reporting requirements without compromising on performance.

Open Solution: HTTP, HTTPS, and REST are open and standardized protocols, ensuring compatibility and accessibility across various systems, platforms, and programming languages. As open solutions, they do not require vendor-specific dependencies, making them ideal choices for ESMA's reporting ecosystem.

Implementation Feasibility: HTTP, HTTPS, and REST have proven to be highly feasible for implementation. They are widely used and well-documented, with numerous libraries and tools available for easy integration. The straightforward nature of these protocols simplifies the development process, reducing implementation time and effort.

5 Conclusions

This chapter aims at providing recommendation on the suitable data format and transmission protocol for the purpose of the CTP and on a potential basis for the revision of ESMA choice for other purposes (opportunity to reuse). Therefore, it is bringing together the outcomes of the overall study, which combines the outcome of the technical assessment with an evaluation of the suitability in both contexts.

5.1 Recommendations on the suitable data formats and transmission protocols

The reflection on the selection of the suitable data format and transmission protocol for CTP needs and other regulatory reporting regimes is heavily influenced by the following considerations, which are the requirements for high quality data, real time transmission and reusability.

As far as the data format is concerned, out of the 5 shortlisted formats JSON is recommended to be suitable to meet simultaneously both the CTP needs and other regulatory reporting requirements. It need to be taken with clear consideration that individually different formats could be more suitable for each use case separately. For instance, binary format like FAST better suit CTP needs. Nevertheless, considering the reusability factor for other ESMA reporting needs and to have a same solution for CTP, JSON is therefore recommended as the most suitable format.

Advantages of using JSON as data format for CTP’s and other regulatory reporting’s purposes

1. **Human-Readable and Lightweight:** JSON is a human-readable data format, making it easy to read and understand by developers and analysts. It is also lightweight, resulting in minimal overhead during data transmission and storage.
2. **Widely Supported:** JSON is supported by a vast array of programming languages, making it a highly versatile choice for data exchange in various applications and platforms.
3. **Easy to Integrate with Web Technologies:** JSON's native compatibility with JavaScript makes it an excellent choice for web-based applications, as it can be seamlessly integrated with front-end frameworks and libraries.
4. **High level of reusability:** JSON's simple and flexible structure allows for easy reusability of data across different systems and use cases, facilitating efficient data integration.
5. **Well-suited for real-time data transmission:** JSON's lightweight nature and ease of parsing make it well-suited for near real-time data transmission, providing low latency and quick response times.

6. **ISO Compliancy:** Compliant with ISO 20022 standard

Disadvantages of using JSON as data format for CTP's and other regulatory reporting's purposes

1. **Overhead in size:** JSON is a text-based format, which can lead to larger payload sizes compared to binary formats like ASN.1 or Protocol Buffers. This may result in increased network traffic and higher bandwidth requirements.
2. **Relatively low level of adoption:** JSON is not as extensively embraced by the market data contributors for trade data distribution, which may result in a less favorable response among some data contributors.

In consideration of the transmission protocol, among the six shortlisted options, REST is identified as the most suitable solution, addressing both the specific needs of the Consolidated Tape Platform (CTP) and meeting the requirements of other regulatory reporting purposes. However, it is worth noting that distinct protocols may be optimally suited for the specific needs of the Consolidated Tape Platform (CTP) and the unique requirements of other regulatory reporting purposes, respectively.

Advantages of using REST as transmission protocol for CTP's and other regulatory reporting purposes:

1. **Simplicity and ease of use:** RESTful APIs are simple to understand and use, making it easier for developers to integrate and communicate with the systems. This simplicity reduces the learning curve for new team members and fosters faster development.
2. **Protocol independence:** REST uses standard HTTP methods and can work with various transport protocols, including HTTP and HTTPS. This flexibility allows it to adapt to different network environments and infrastructures.
3. **Standardization and interoperability:** RESTful APIs adhere to well-established standards and conventions, promoting interoperability and ease of integration with various systems and tools. This standardization fosters consistency and compatibility across different platforms.
4. **Performance and caching:** REST leverages HTTP caching mechanisms, reducing redundant data transfers and improving performance. This can be beneficial in real-time data transmission and high-throughput scenarios.
5. **Security measures:** REST can be used securely by implementing SSL/TLS encryption and authentication mechanisms. These security measures help safeguard sensitive data transmitted through the RESTful APIs.

Disadvantages of using REST as transmission protocol for CTP's and other regulatory reporting's purposes:

1. **Overhead in size:** REST relies on text-based formats like JSON or XML, which can result in larger payload sizes compared to binary formats. This may lead to increased network traffic and higher bandwidth requirements.
2. **Complexity in versioning:** RESTful APIs may require careful versioning strategies to maintain backward compatibility and manage changes in the API over time. Improper versioning can lead to compatibility issues with existing clients.
3. **Lack of real-time communication:** REST APIs operate on a request-response model, which may not be suitable for scenarios requiring instant communication between clients and servers.
4. **Relatively low level of adoption:** REST is not as extensively embraced by the market data contributors for trade data distribution, which may result in a less favorable response among some data contributors.

5.2 Recommendations on relevant next steps

For concluding this preliminary study, this section aims at paving the way for pursuing the reflection on the selection of the most suitable solutions. Without pretending to exhaustivity, the assessment activities gave the opportunity to identify a few actions to be taken in three domains.

- **Suggested actions about ISO 20022 compatibility**

During this project, the adherence of CTP solutioning to ISO 20022 appeared as a sensitive topic. As a matter of fact, it should be acknowledged on one hand the strategic importance of data harmonization and on the other hand the impacts of a strict adherence to ISO 20022 (i.e in its current state) on the performance and in terms of disruption against current market practices. A potential way out to this difficulty may lie in the possibility to leverage ISO 20022 conceptual and logical layers for the sake of data harmonization while using a physical layer (i.e data format) meeting performance's expectations. For clearing this topic, the following action is recommended:

⇨ With MIFIR review schedule in mind, engage with financial services industry standard setting bodies on the current status of the work around ISO 20022 interoperability and the extension of ISO 20022 compliant data format leveraging the possibility to recognize other data format as "ISO 20022 compliant using a domain specific syntax".

- **Suggested actions for proofing the outcome of the study**

The delivery of this study is relying on available expertise, external consultation, and desk research. Moreover, the external consultations demonstrated a diversity of sights on the most suitable solutions for the purpose of the CTP. Consequently, it seems necessary to proof the outcome of the assessment against practical expertise and edify an alignment for ensuring later a smooth adoption. Hence, a few logical next steps would be to:

- ⇨ With regards to CTP purpose, appoint an industry working group to build a common understanding on the suitable solutions. This would also give the opportunity to validate the outcomes of the study with trading industry experts.
- ⇨ With regards to the revision of ESMA technical choices, qualify in greater details the future needs and consider the current technical infrastructure's constraints.
- ⇨ In both cases, conduct a proof of concept for validating the suitability and viability of the recommended solutions.

- **Suggested actions for the issuance of CTP related specifications**

While EC MIFIR review proposal envisions the set-up of four CTP for each of the asset classes, it is understood at the time of closing this study that this provision may further evolve. In parallel, ambiguity remains on the regulatory format under which ESMA will be required to intervene on the suitable solution for CTP purposes (e.g report to the Commission, RTS,…). Moreover, it emerges clearly from the study that the selection of the suitable solution for the CTP is resulting from a trade-off between the various identified criteria in the absence of a perfectly matching solution. Without interfering in the policy making process, it is suggested in this context to:

- ⇨ For the purpose of the specification's issuance, examine whether it is appropriate to name specific solutions or rather leave it open through requirements-oriented approach leaving to the CTP entity the possibility to propose an optimal set up.
- ⇨ Arbitrate whether a single solution should apply to all CTPs across all asset classes acknowledging the advantage of implementation ease but a potential disadvantage in terms of functional value.

6 Appendix

6.1 Assessment grid

6.1.1 Assessment grid for data formats

Criteria	Description	Weighting	Weighting justification	Qualitative grading
Reliability	The reliability of a data format refers to its ability to ensure that data is accurately transmitted and received without corruption or loss. This includes features such as error correction and detection mechanisms, as well as recovery mechanisms in case of transmission errors or failures.	Mandatory	Reliability is considered as one of the factors contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "mandatory"	High/ Medium/ Low
Ease of use	Ease of use refers to how user-friendly and easy to understand the data format is for both developers and end-users. This includes features such as clear and concise syntax, well-defined data structures, and ease of implementation.	Recommended	Ease of use is considered as one of the factors expected to ease overall CTP set-up, hence "recommended"	High/ Medium/ Low
Encryption	Encryption refers to the ability of the data format to protect data by encrypting it to prevent unauthorized access or tampering. This includes features such as data encryption algorithms and	Mandatory	Encryption is considered as a critical factor contributing to the overarching security requirement (see Art 27h §3), hence "mandatory"	Yes/No

	key management mechanisms.			
Digital signature	Digital signature refers to the ability of the data format to provide authentication and integrity of the data by using cryptographic signatures. This includes features such as digital signature algorithms and key management mechanisms.	Nice to have	Digital signature is considered as a complementary yet non-critical factor next to Encryption and Authentication contributing to the overarching security requirement (see Art 27h §3), hence "nice to have"	Yes/No
Technical data validation	Technical data validation refers to the ability of the data format to validate technically data before it is transmitted to ensure that it meets certain criteria or conforms to a specified schema. This includes features such as data validation rules, data type validation, and schema validation.	Mandatory	Technical data validation is considered as one of the factors contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "mandatory"	Yes/No
Encoding	Encoding refers to how the data is represented in binary form, and the ability of the data format to efficiently encode and decode data for transmission. This includes features such as variable-length encoding, fixed-length encoding, and character encoding.	Mandatory	Encoding is considered as a critical factor for supporting all character sets contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "mandatory"	Yes/No

Support complex data structures	Support for complex data structures refers to the ability of the data format to represent and transmit complex data structures such as nested arrays, objects, and graphs.	Recommended	Support complex data structures is considered as a relevant yet non-critical factor for supporting all character sets contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "recommended"	Yes/No
Support metadata	Support for metadata refers to the ability of the data format to include additional information about the data being transmitted, such as data type, version, author, etc.	Recommended	Support metadata is considered as a relevant yet non-critical factor for supporting all character sets contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "recommended"	Yes/No
Support nested data	Support for nested data refers to the ability of the data format to represent and transmit data that contains hierarchical or nested structures.	Mandatory	Support nested data is considered as a critical factor for supporting all data formats contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "mandatory"	Yes/No
Support inline documentation	Support for inline documentation refers to the ability of the data format to include comments or documentation within the data itself, to make it easier to understand and maintain.	Nice to have	Support inline documentation is considered as a complementary yet non-critical factor next to the actual documentation and specifications of the data format itself contributing to the overarching requirement	Yes/No

			for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "nice to have"	
Parsing speed	Parsing speed refers to how quickly the data format can be parsed and converted to a usable format, and how efficiently it can be processed by the receiving application.	Mandatory	Parsing speed is considered as one of the factors contributing to the overarching real time requirement (see Art. 22a §1), hence "mandatory"	High/ Medium/ Low
Serialization speed	Serialization speed refers to how quickly the data format can be converted to binary form for transmission, and how efficiently it can be transmitted over a network.	Mandatory	Serialization speed is considered as one of the factors contributing to the overarching real time requirement (see Art. 22a §1), hence "mandatory"	High/ Medium/ Low
Network overhead optimization	Network overhead refers to the amount of additional data that is added to the transmitted data due to the data format, including header information, metadata, and error correction mechanisms. The lower the network overhead, the more efficient the data format is in terms of network bandwidth usage.	Mandatory	Network overhead optimization is considered as a critical factor contributing to the overarching real time requirement (see Art. 22a §1), hence "mandatory"	High/ Medium/ Low
Flexibility	Flexibility of a data format refers to its ability to accommodate diverse data structures, adapt to varying data	Recommended	Flexibility is considered as a non-critical factor for CTP functioning yet important as contributing to the overarching	High/ Medium/ Low

	requirements, and support customization and extensibility.		"Reusability" requirement set by ESMA, hence "recommended"	
Open solution	Data format protocols must be available as an open solution understood as a solution, which allows the users to benefit from the right to use, change and distribute the solution without any restrictions and free of charge (i.e it is not meant to analyse the governance model (e.g ownership) surrounding the solution)	Recommended	Open solution is considered as a non-critical factor for CTP functioning yet important as contributing to the overarching "Reusability" requirement set by ESMA, hence "recommended"	Yes/No
Level of adoption	Data formats & transmission protocols must show a significant level of adoption in other regulatory framework, in Europe, and in other jurisdictions	Recommended	Level of adoption is considered as a necessary yet non-critical factor contributing to the overall CTP set-up, hence "recommended"	High/ Medium/ Low
Implementation Feasibility	Impact in terms of costs to implement & run as well as implementation timing need to be assessed	Recommended	Implementation feasibility is considered as a necessary yet non-critical factor contributing to the overall CTP set-up, hence "recommended"	High/ Medium/ Low
Protocol Compatibility	The data format must be compatible with the different protocols to facilitate message exchange.	Recommended	Protocol compatibility is considered as a necessary yet non-critical factor contributing to the overall CTP functioning, hence "recommended"	High/ Medium/ Low
ISO 20022 Compatibility	The data format is compatible with ISO 20022 messaging standard.	Recommended	ISO 20022 compatibility is considered as a necessary yet non-critical factor contributing to the overarching "Reusability"	Yes/No

			requirement set by ESMA, hence "Recommended"	

6.1.2 Assessment grid for transmission protocols

Criteria	Description	Weighting	Weighting justification	Qualitative grading
Data confidentiality	Data confidentiality refers to the ability of a transmission protocol to keep data secure and protect it from unauthorized access. This includes features such as encryption and secure data transfer mechanisms.	Mandatory	Data confidentiality is considered as a critical factor contributing to the overarching security requirement (see Art 27h §3), hence "mandatory"	Yes/No
Authentication	Authentication refers to the ability of a transmission protocol to verify the identity of the sender or receiver of data. This includes features such as user credentials, or digital certificates.	Mandatory	Authentication is considered as a critical factor contributing to the overarching security requirement (see Art 27h §3), hence "mandatory"	Yes/No
Authorization	Authorization refers to the ability of a transmission protocol to ensure that users or systems only have access to the data that they are authorized to access. This includes features such as access control mechanisms and role-based access controls.	Mandatory	Authorization is considered as a critical factor contributing to the overarching security requirement (see Art 27h §3), hence "mandatory"	Yes/No
Non-repudiation	Non-repudiation refers to the ability of a transmission protocol to ensure that the sender of data cannot deny that they sent it, and the receiver cannot deny that they received it. This includes features such as	Recommended	Non-repudiation is considered as a necessary yet non-critical factor contributing to the overarching security requirement (see Art 27h §3), hence "recommended"	Yes/No

	digital signatures and audit trails.			
Encryption	Encryption refers to the ability of a transmission protocol to secure data by encrypting it to prevent unauthorized access or tampering.	Mandatory	Encryption is considered as a critical factor contributing to the overarching security requirement (see Art 27h §3), hence "mandatory"	Yes/No
Secure file transfer	Secure file transfer refers to the ability of a transmission protocol to transfer files securely and reliably over a network. This includes features such as secure file transfer protocols, encryption, and checksums.	Nice to have	Secure file transfer is not necessary for the envisioned CTP implementation, yet potentially relevant for future needs, hence "nice to have"	Yes/No
Latency Optimization	Latency refers to the time delay between sending and receiving data over a network. Low latency is important for applications that require real-time processing, such as expected for the CTP.	Mandatory	Latency optimization is considered as a critical factor contributing to the overarching real time requirement (see Art. 22a §1), hence "mandatory"	High/ Medium/ Low
Throughput	Throughput refers to the amount of data that can be transferred over a network in a given amount of time. High throughput is important for applications that require large amounts of data to be transferred quickly, such as video streaming or file transfers.	Mandatory	Throughput is considered as critical factor contributing to the overarching real time requirement (see Art. 22a §1), hence "mandatory"	High/ Medium/ Low
Connection setup time Optimization	Connection setup time refers to the amount of time it takes to establish a connection between two systems over a network.	Mandatory	Connection setup time optimization is considered as a critical factor contributing to the overarching real time	High/ Medium/ Low

	Low connection setup time is important for applications that require fast and frequent connections, such as real-time communication applications.		requirement (see Art. 22a §1), hence "mandatory"	
Reliability	Reliability refers to the ability of a transmission protocol to ensure that data is accurately transmitted and received without corruption or loss. This includes features such as error correction and detection mechanisms, as well as recovery mechanisms in case of transmission errors or failures.	Mandatory	Reliability is considered as a critical factor contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "mandatory"	High/ Medium/ Low
Scalability	Scalability refers to the ability of a transmission protocol to handle increasing amounts of traffic and users without sacrificing performance or reliability. This includes features such as load balancing and distributed systems.	Recommended	Scalability is considered as a factor contributing to the overarching "Reusability" requirement set by ESMA, which is itself relevant yet not critical for the overall CTP functioning, hence "recommended"	High/ Medium/ Low
Interoperability	Interoperability refers to the ability of a transmission protocol to work seamlessly with other systems and technologies, regardless of their platform or language. This includes features such as standardized protocols and APIs.	Mandatory	Interoperability is considered as a critical factor for integration contributing to the overall CTP setup, hence "mandatory"	High/ Medium/ Low

Manageability	Manageability refers to the ease with which a transmission protocol can be managed and configured, including features such as monitoring, logging, and troubleshooting. The protocol should be easy to use and configure, with comprehensive documentation and support resources available.	Recommended	Manageability is considered as a necessary yet non-critical factor contributing to the overall CTP functioning, hence "recommended"	High/ Medium/ Low
Data Integrity	Data integrity refers to the assurance that data is not altered or corrupted during transmission, storage or processing.	Mandatory	Data integrity is considered as a critical factor contributing to the overarching requirement for "market data quality" and "harmonised format" (see Art. 22a §1 and Art. 22b), hence "mandatory"	High/ Medium/ Low
Open solution	Transmission protocols must be available as an open solution understood as a solution, which allows the users to benefit from the right to use, change and distribute the solution without any restrictions and free of charge (i.e it is not meant to analyse the governance model (e.g ownership) surrounding the solution).	Recommended	Open solution is considered as a non-critical factor for CTP functioning yet important as contributing to the overarching "Reusability" requirement set by ESMA, hence "recommended"	Yes/No
Level of adoption	Data formats & transmission protocols must show a significant level of adoption in other regulatory framework, in Europe, and in other jurisdictions	Recommended	Level of adoption is considered as a necessary yet non-critical factor contributing to the overall CTP set-up, hence "recommended"	High/Medium/Low

Implementation Feasibility	Impact in terms of costs to implement & run as well as implementation timing need to be assessed	Recommended	Implementation feasibility is considered as a necessary yet non-critical factor contributing to the overall CTP set-up, hence "recommended"	High/Medium/Low

6.2 Governance and change control process

This appendix provides an analysis of the governance structure and change procedure governing the standardisation of the solutions shortlisted in the assessment. In many cases, the standardization of the analysed solutions is overseen by a same governing body as illustrated in the table below. Interestingly, all of the assessed solutions are available through open-source license as documented in the assessment sections dedicated to the “open solution” criteria.

Governance body	Shortlisted solutions	Category
Internet Engineering Task Force (IETF)	JSON	Data format
	Protocol Buffer	Data format
	Websocket	Transmission protocol
	SFTP	Transmission protocol
	HTTPS	Transmission protocol
	Multicast	Transmission protocol
FIX Trading Community	FAST	Data format
	SBE	Data format
International Telecommunication Union (ITU-T)	ASN.1	Data format
Apache Software Foundation (Non-exhaustive)*	MQ	Transmission protocol
Not applicable (cf. IETF for HTTPS transport protocol)	REST	Transmission protocol

** There is no single governance body overseeing the standardization of MQ – yet this section provides further information on one of the organisations managing one of the open source version available for MQ.*

6.2.1 Internet Engineering Task Force

The Internet Engineering Task Force (IETF) is an international standard setting organization[48] registered as a single member limited liability company of the Internet Society, which is an international non-profit organization registered in the U.S (Washington D.C)[49].

6.2.1.1 Governance

There is no membership in the IETF. Anyone can participate by signing up to a mailing list, or registering for an IETF meeting. All IETF participants are considered volunteers and expected to participate as individuals, including those paid to participate. Over 6000 people actively participate in the IETF either by authoring a document, engaging in a mailing list discussion, or attending a meeting.

IETF is managed by the following groups:

- The Internet Architecture Board (IAB) provides long-range technical direction for Internet standards, ensuring the Internet continues to grow and evolve as a platform for global communication and innovation. Among others, the IAB provides oversight of the process used to create Internet Standards. The IAB serves as an appeal board for complaints of improper execution of the standards process through acting as an appeal body in respect of an IESG standards decision.
- The Internet Engineering Steering Group (IESG) is responsible for technical management of IETF activities, and the Internet standards process. The IESG consists of the Area Directors (ADs) who are appointed for two years.
- Area Directorates comprised of experienced IETF participants, serve as advisory groups for IETF work.
- IETF working groups (WGs) are the primary mechanism for development of IETF specifications and guidelines, many of which are intended to be standards or recommendations. A working group may be established at the initiative of an Area Director or it may be initiated by an individual or group of individuals. Anyone interested in creating an IETF working group must obtain the advice and consent of the IETF Area Director(s) in whose area the working group would fall.
- The Internet Research Task Force (IRTF) focuses on longer term research issues related to the Internet while the parallel organization, the Internet Engineering Task Force (IETF), focuses on the short-term issues of engineering and standards making.

[48] IETF | Internet Engineering Task Force
[49] p.11, IETF 2022 audited financial statements

6.2.1.2 Change control process of IETF standards

New work in the IETF begins with one or more participants producing a document called an Internet-Draft (I-D) and then working to get that I-D adopted for further work. Anyone can write an Internet-draft on any topic they believe is relevant to the IETF. There are different routes that an I-D can follow to be adopted, worked on and eventually become an RFC.

The vast majority of the IETF's work is done in its many Working Groups. A Working Group (WG) has its own mailing list with most of its interaction, and all of it official work, conducted via email. A WG also has a charter that states the scope of discussion for the WG and its goals. The WG's mailing list and any WG meetings are expected to focus only on what is in the charter. A WG is headed by one or two (occasionally three)"WG chairs".

Working Groups are organized into one of seven areas, Application and Real Time (art), General (gen), Internet (int), Operations and Management (ops), Routing (rtg), Security (sec), and Transport (tsv), with each area overseen by one to three "Area Directors" (AD).

The day to day work of WGs revolves around Internet-Drafts, those that have been proposed for adoption and those that have been adopted, and over time the WG shapes the latter into RFCs. Decisions within WGs, as with the broader IETF, are taken by 'rough consensus' and not by voting. It is the role of the WG chair(s) to determine when rough consensus has been reached. When a Working Group has finished with an I-D and is ready for it to become an RFC, the I-D goes through a process to ensure that it has approval from the appointed technical leadership and the consensus support of the IETF as a whole.

The other routes for an I-D to become an RFC are as the output of some of the leadership bodies, Area Directors can sponsor an I-D, and there is an independent submissions process.

For an RFC to become a Proposed Standard or Internet Standard there must be at least two independent and inter-operable implementations, and the RFC must have full IETF consensus.

The IETF has policies about Intellectual Property Rights (IPR) that are designed to ensure that Working Groups and participants have as much information as possible about any IPR constraints on a technical proposal as early as possible in the development process.

When an I-D has cleared all the hurdles to become an RFC it goes through a professional editorial process[50] and is then assigned a number, published in a range of formats, both human- and machine-readable, and deposited in libraries and archives.

[50] Publication Process » RFC Editor (rfc-editor.org)

6.2.2 FIX Trading Community

The FIX Trading Community is defined as a non-profit, independent, and neutral industry-driven standards body at the heart of global trading. FIX Trading Community™ is a brand of FIX Protocol Ltd, which is registered in UK as a non-profit organization[51].

Besides the maintenance of FIX protocol (i.e messaging standard), FIX Trading Community works on the continuous development of the FIX standards, such as the physical layers analysed in this study (FIXML, FAST, SBE). All initiatives are pursued in response to market participant requests.

6.2.2.1 Governance

FIX Trading Community is a membership-based organisation, which brings together peers and competitors to work together collaboratively within the various committees and working groups. It involves more than 270 financial service companies across the globe. With regards to the production of standards and specifications, the governance of the FIX Trading Community consists of three layers:

- The Global Steering Committee (GSC) is responsible for overseeing FIX Trading Community as a brand of FIX Protocol Ltd. Its primary role is to effectively manage the Fix Trading Community's ongoing needs at both strategic and operational level[52]. The GSC includes representation from the leaders of each of the FIX Trading Community's committees.
- The Global Technical Committee (GTC) Governance Board is the one in charge of reviewing and ratifying proposals for the enhancement of the FIX Protocol as well as overseeing initiatives such as the development of technical specifications, the FIX implementation guide, etc.[53]
- Working groups, product committees/sub-committees and regional committees typically initiate technical and gap analysis initiatives and are originating the standardization work following a project delivery mode.

6.2.2.2 Change control process of FIX managed solutions

The standard process of the FIX application layer by means of Gap Analysis proposals is made of five steps, which takes approximately 40 working days from the start of the process to the end time.

[51] Corporate Governance • FIX Trading Community
[52] http://www.fixtradingcommunity.org/pg/committees/29905/global-steering-committee/
[53] http://www.fixtradingcommunity.org/pg/committees/29906/global-technical-committee/

- Step 1 – Technical and gap analysis (‘’proposal’’) are initiated and formulated by working groups or product committees/sub-committee.
- Step 2 – The GTC co-chairs will accept the proposal for formal review and introduce the proposal to the wider GTC membership. Reviews are open for participation to all GTC members and other interested FPL members. Groups who submit proposals for which there are external deadlines (for instance, MiFIR technical format) should communicate this information to the GTC co-chairs at the beginning of the process so that proposals can be prioritised.
- Step 3 – Once the formal review is completed (Step 2), the GTC co-chairs will put forth the finalised proposal to a vote by the GTC members.
- Step 4 – In case where the proposal has been approved by the GTC Governance Board vote, the proposal will then be published to the Technical Specifications page of the FIX Trading Community website as ‘’Extension Release’’ or ‘’Errata/Sevrice Release’’.
- Step 5 – In case of ‘’rejected’’ proposal, the GTC Governance Board will review with the working group the reasons for rejection and what needs to be addressed. Therefore, the working group will need to resubmit the proposal and expedited review process will be initiated once the revised proposal is submitted.

6.2.3 International Telecommunication Union (ITU-T), an agency of the United Nations

The Telecommunication standardization sector of the International Telecommunication Union (ITU-T) is an agency of the United Nations which is headquartered in Switzerland[54]. The Study Groups of the ITU-T assemble experts from around the world to develop international standards known as ITU-T recommendations which act as defining elements in the global infrastructure of information and communication technologies (ICTs).

6.2.3.1 Governance

The ITU-T is driven by a contribution-led, consensus-based approach to standards development in which all countries and companies, no matter how large or small, are afforded equal rights to influence the development of ITU-T Recommendations. ITU-T Membership is open to all commercial companies and non-profit organizations. There are two types of ITU-T members: Member States and Sector Members. ITU Member States are represented by national administrations, while Sector Members consist of companies from the private sector (e.g. service providers, scientific institutions), and other regional and international organizations.

[54] ITU Telecommunication Standardization Sector

ITU-T standardization work is governed by the following framework[55]:

World Telecommunication Standardization Assembly (WTSA) sets the overall direction and structure for ITU-T. It meets every four years and defines the general policy for the Sector, establishes the study groups, approves their expected work programme for the next four-year period, and appoints their chairmen and vice-chairmen.

Telecommunication Standardization Advisory Group (TSAG) provides ITU-T with flexibility between WTSAs by reviewing priorities, programmes, operations, financial matters and strategies for the Sector. It also follows up on the accomplishments of the work programme, restructures and establishes ITU-T study groups, provides guidelines to the study groups, advises the Director of the Telecommunication Standardization Bureau (TSB), and produces organization and working procedures in the shape of A series Recommendations.

Study Groups (SG) are at the heart of ITU-T and drive their work primarily in the form of study questions addressing technical challenges in a particular area of telecommunication standardization. Each SG has a chairman and a number of vice-chairmen appointed by the World Telecommunication Standardization Assembly (WTSA).

Telecommunication Standardization Bureau (TSB) provides secretariat support to ITU-T Study Groups through sophisticated electronic working methods and state-of-the-art facilities.

Public workshops and webinars are organized by the ITU-T to progress existing work areas and explore new ones. The events cover a wide array of topics in the ICT field and speakers and attendees include engineering, strategy and policy experts from a range of industry sectors. Organized events are free of charge and open to the public.

6.2.3.2 Change control process of ITU-T standards

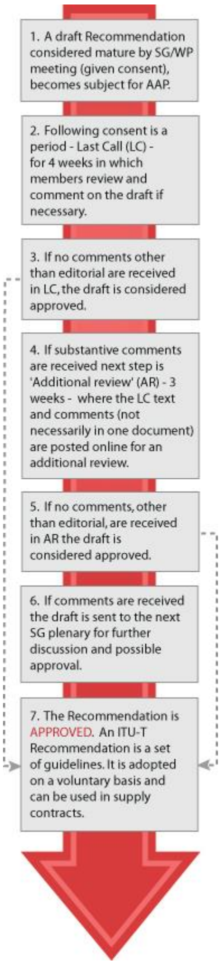
The development of standards at ITU-T is sequenced in two main phases –called “standards development” and “standards approval”.

Standards development:

[55] The framework of ITU-T

A question is the basic project unit within ITU-T. The area of study of the project is defined by the text of the question, and this is generally approved by the study group itself. For a new question to be established, it is necessary that a number of Members commit to support the work. A question is normally terminated once the defined work has been completed, or the task is revised in the light of developments, which can be technical, market-oriented, network or service driven.

FIGURE 25 - ITU-T STANDARDS DEVELOPMENT WORKFLOW



1. A draft Recommendation considered mature by SG/WP meeting (given consent), becomes subject for AAP.

2. Following consent is a period - Last Call (LC) - for 4 weeks in which members review and comment on the draft if necessary.

3. If no comments other than editorial are received in LC, the draft is considered approved.

4. If substantive comments are received next step is 'Additional review' (AR) - 3 weeks - where the LC text and comments (not necessarily in one document) are posted online for an additional review.

5. If no comments, other than editorial, are received in AR the draft is considered approved.

6. If comments are received the draft is sent to the next SG plenary for further discussion and possible approval.

7. The Recommendation is APPROVED. An ITU-T Recommendation is a set of guidelines. It is adopted on a voluntary basis and can be used in supply contracts.

To assist in the organization of the work, the SG may be organized into a number of working parties. The working party is the next organizational unit down within the SG. It coordinates a number of study questions on related themes.

The team of experts working on a specific question is known as the rapporteur group. Their meetings are chaired by the relevant rapporteur. Considering the text of the Question and guidance from the SG, the participants determine what Recommendations are required and develop text for these Recommendations taking all relevant inputs into account and consulting other relevant parts of ITU-T.

Once the text of a draft Recommendation prepared by SG's experts is considered mature, it is submitted for review to a SG or WP meeting.

Standard approval:

Upon review of a draft Recommendation prepared by SG, "consent" is given. This means that the SG or WP has given its consent that the text is sufficiently mature to initiate a final review process leading to approval of the draft Recommendation.

After this Consent has been achieved, the Telecommunication Standardization Bureau (TSB), announces the start of the Alternative Approval Procedure, an average 2 months long fast tracked process used in most of the cases. Precisely, the TSB is posting the draft text and calling for comments. This gives the opportunity for all members to review the text.

This phase, called Last Call, is a four-week period in which comments can be submitted by Member States and Sector Members. If no comments other than editorial corrections are received, the Recommendation is considered approved since no issues were identified that might need any further work. However, if there are any comments, the SG chairman, in consultation with TSB, sets up a comment resolution process by the concerned experts. The revised text is then posted on the web for an Additional Review period of three weeks. Like the Last Call phase, in Additional Review the Recommendation is considered as approved if no comments are received. If comments are received, it is apparent that there are some issues that still need more work, and the draft text and all comments are sent to the next SG meeting for further discussion and possible approval.

6.2.4 Apache Software Foundation

The Apache Software Foundation (ASF)[56], a non-profit organization founded in 1999 in the U.S is maintaining two of the well-known versions of message queuing systems (MQ), Apache ActiveMQ and Apache Kafka, which are available among others open sources versions and commercial versions.

6.2.4.1 Governance

The Apache Software Foundation is a decentralized open source community of developers. Next to the corporate governing bodies, the development of specifications at ASF is managed by the technical governing bodies involving non-governing bodies following the Apache Way. The latter is driven by the following principles:

- Collaboration: The Apache Way emphasizes collaborative development, where individuals work together to achieve common goals. Collaboration is fostered through open discussions, consensus decision-making, and encouraging community involvement.
- Meritocracy: The Apache Way operates on the principle of meritocracy, where individuals earn influence and decision-making authority based on their contributions, skills, and expertise. Those who contribute significantly and show a deep understanding of the project's goals are given more responsibility. In the Apache Way, contributors earn privileges and responsibilities over time based on their demonstrated commitment and contributions. Active contributors may be invited to join the PMC or become Apache Members.
- Openness and Transparency: The Apache Way promotes transparency in decision-making, development processes, and project governance. Discussions, decisions, and project activities are conducted openly on public mailing lists and other communication channels.
- Consensus Decision Making: Major decisions within Apache projects are typically made through consensus. This involves open discussion among community members until agreement is reached. The goal is to involve as many perspectives as possible in decision-making.

[56] Apache Software Foundation!

- Independence of Projects: Each Apache project operates relatively independently, with its own Project Management Committee (PMC) responsible for decision-making. Projects can choose their own technology stacks, release schedules, and development processes.
- Licensing and Intellectual Property: Proper licensing and intellectual property management are paramount in the Apache Way. All code contributions must be properly licensed and granted to the ASF, and contributors must sign a Contributor License Agreement (CLA).

- **Technical governing bodies**

Within the ASF, the board delegates the technical direction of each project to its Project Management Committees.

Project Management Committees (PMCs) are expected to manage their projects independently. They work to produce software for the public good by voting on releases of their project's software products. Especially, the main role of the PMC is to ensure that its community addresses all legal issues and follows stated procedures, and that each and every release is the product of the community as a whole. The second role of the PMC is to work on the long-term development and health of the community as a whole, and to ensure that balanced and wide scale peer review and collaboration takes place.

Committers are members of a project development community who have been granted write access to an Apache project. Each project's PMC invites people who have shown merit within their project to become committers. Committers must sign an Individual Contributor License Agreement (ICLA), which clearly defines the terms under which the committer contributes intellectual property to the ASF. This allows the projects to ensure that they can safely release the products they publish under the Apache License.Committers are elected separately for every project; merit within one project is not necessarily transferable to other projects. Committers also have access to a one Foundation-wide committer repository, where a few extra services and tools useful for doing Apache project work are available.

- **Non-Governing Bodies**

As a community-based organization, many other groups of individuals and organizations provide work and services to the ASF and Apache projects but are not directly part of the governance.

Contributors are individuals who contribute source code patches, documentation, and help on mailing lists to Apache projects. Contributors do not have a specific governance role, till they are nominating as new committers.

Users use, and often ask for help about the software. Many users do not code, but still spend the time to submit bug reports and answer questions on the project's mailing lists.

6.2.4.2 Change control process of Apache Software foundation’s standards

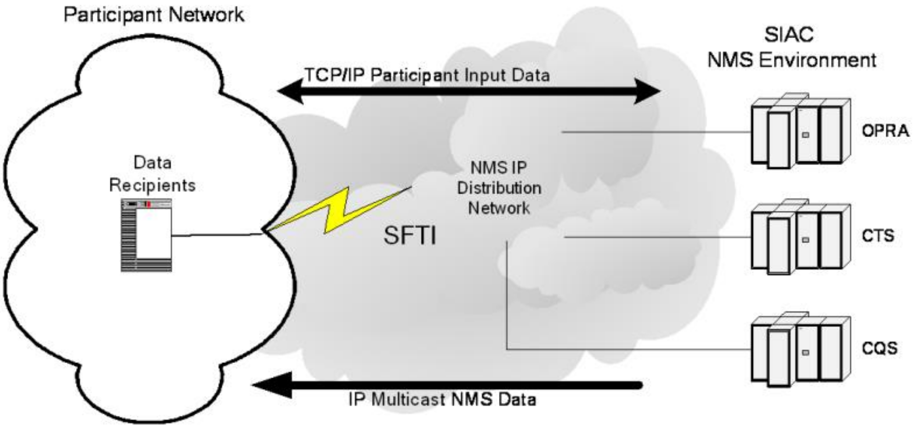
Besides the collaborative and consensus-based approach, there is no formal change procedure and projects are normally auto governing and driven by the people who volunteer. When coordination is required, projects make decisions with a consensus approach materialized by vote. The rules require that a PMC member registering a negative vote must include an alternative proposal or a detailed explanation of the reasons for the negative vote. The community then tries to gather consensus on an alternative proposal that can resolve the issue. Specific cases have some more detailed voting rules.

6.3 Research on market data consolidation in the United States

In the United States, market data consolidation for bonds and equities are operated independently while managed by the Securities Information Automation Corporation (SIAC). SIAC is itself overseen by the Consolidated Tape Association (CTA). For reference, the CTA supervises two separate plans, namely the Consolidated Tape System (CTS) for trades and the Consolidated Quotation System (CQS) for quotes[57].

With regards to equities, the figure below illustrates market data consolidation happens through a TCP/IP point to point transmission protocol, called National Market System (NMS) hosted on ICE Global Network[58]. Market participants (i.e maket data contributors) provide the CTS with market data using a binary format[59].

FIGURE 26 - OVERVIEW OF U.S MARKET DATA CONSOLIDATION ARCHITECTURE[60]



Regarding fixed incomes and specifically bonds, market data consolidation is organized by FINRA on the TRACE platform[61], which enables the mandatory reporting for eligible fixed income securities. Brokers-dealers (i.e market data contributors) are required to report market data on a web-based application. FINRA also supports computer-to-computer connectivity for

[57] CTS/CQS background Appendix K.PDF (sec.gov)
[58] P.5, TCP/IP for NMS Participant Input (website-files.com)
[59] CTS_Pillar_Input_Specification.pdf (ctaplan.com)
[60] TCP/IP for NMS Participant Input (website-files.com)
[61] P.65, The study on the creation of an EU consolidated tape - Publications Office of the EU (europa.eu)

trade reporting through FIX interface. The latter is using FIX protocol formats, specifically tagvalues and FIXML[62] and a web-based API.

[62] CA FIX Spec (finra.org)

6.4 References

6.4.1 Legislative sources

SHORT NAME	TITLE
MIFID 2	Directive 2014/65/EU of the European Parliament and of the Council of 15 May 2014 on markets in financial instruments and amending Directive 2002/92/EC and Directive 2011/61/EUText with EEA relevance
MIFIR	Regulation (EU) No 600/2014 of the European Parliament and of the Council of 15 May 2014 on markets in financial instruments and amending Regulation (EU) No 648/2012Text with EEA relevance
RTS1	Commission Delegated Regulation (EU) 2017/587 of 14 July 2016 supplementing Regulation (EU) No 600/2014 of the European Parliament and of the Council on markets in financial instruments with regard to regulatory technical standards on transparency requirements for trading venues and investment firms in respect of shares, depositary receipts, exchange-traded funds, certificates and other similar financial instruments and on transaction execution obligations in respect of certain shares on a trading venue or by a systematic internaliser
RTS2	Commission Delegated Regulation (EU) 2017/583 of 14 July 2016 supplementing Regulation (EU) No 600/2014 of the European Parliament and of the Council on markets in financial instruments with regard to regulatory technical standards on transparency requirements for trading venues and investment firms in respect of bonds, structured finance products, emission allowances and derivatives
RTS3	Commission Delegated Regulation (EU) 2017/577 of 13 June 2016 supplementing Regulation (EU) No 600/2014 of the European Parliament and of the Council on markets in financial instruments with regard to regulatory technical standards on the volume cap mechanism and the provision of information for the purposes of transparency and other calculations
MIFIR review proposal	Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL amending Regulation (EU) No 600/2014 as regards enhancing market data transparency, removing obstacles to the emergence of a consolidated tape, optimising the trading obligations and prohibiting receiving payments for forwarding client orders

6.4.2 Official publications

SHORT NAME	TITLE
EC CTP study	The study on the creation of an EU consolidated tape - Publications Office of the EU (europa.eu)
EP documentation portal	Procedure File: 2021/0385(COD) | Legislative Observatory | European Parliament (europa.eu)
AFM principles for a corporate bond CT	AFM and industry conclude high-level technical principles for a corporate bond consolidated tape
ESMA registers	ESMA Registers (europa.eu)
CTS/CQS background	Appendix K.PDF (sec.gov)
US CTS	CTS_Pillar_Input_Specification.pdf (ctaplan.com)
SIAC interface specifications	TCP/IP for NMS Participant Input (website-files.com)
FINRA FIX specifications	CA FIX Spec (finra.org)
ESMA MIFID2/MIFIR report	mifid_ii_mifir_review_report_no_1_on_prices_for_market_data_and_the_equity_ct.pdf (europa.eu)

6.4.3 Research papers & publications

SHORT NAME	TITLE
Data exchange study	Data Exchange Mechanisms and Considerations | Enterprise Architecture (harvard.edu)
Secure Data exchange	Secure Data Exchange Protocols (cleo.com)
SWIFT FAQ	Investment Roadmap Frequently Asked Questions (iso20022.org)
Architectural patterns	Avgeriou, Paris; Zdun, Uwe (2005), "Architectural patterns revisited:a pattern language", UVK Verlagsgesellschaft

6.4.4 Standard setting bodies & technology providers

SHORT NAME	TITLE
ISO 20022	About ISO 20022 | ISO20022
ISO 20022 introduction	Swift, ISO 200022 for dummies, 6th limited edition, John Wiley & Sons Inc., 2022 (link)
ISO 20022 presentation	Introduction to ISO 20022 presentation (link)
ISO20022 collaboration FAQ	Investment Roadmap Frequently Asked Questions (iso20022.org)
W3C presentation	History (w3.org)

W3C Membership draft contract	W3C Member Agreement
JSON standard	ECMA-404 - Ecma International (ecma-international.org)
ECMA history	History - Ecma International (ecma-international.org)
CSV RFC	RFC 4180 - Common Format and MIME Type for Comma-Separated Values (CSV) Files (ietf.org)
FIX	Corporate Governance • FIX Trading Community
ITU-T	ITU Telecommunication Standardization Sector
Protocol Buffer	Overview | Protocol Buffers Documentation (protobuf.dev)
MongoDB	MongoDB: The Developer Data Platform | MongoDB
IETF presentation	IETF | Internet Engineering Task Force
IETF 2022 financial statement	https://www.ietf.org/media/documents/2022_Audited_Financials.pdf
Alphabet 2022 financial statement	2022-alphabet-annual-report.pdf (abc.xyz)
FAST	FAST Protocol • FIX Trading Community
SBE	Simple Binary Encoding (SBE) • FIX Trading Community
FIX Tag value	FIX TagValue Encoding • FIX Trading Community
ISO FIX Tag Value	ISO 3531-1:2022(en), Financial services — Financial information eXchange session layer — Part 1: FIX tagvalue encoding

HTTP standard	RFC 9114 - HTTP/3 (ietf.org)
HTTPS standard	RFC 2818: HTTP Over TLS (rfc-editor.org)
IBM MQ	IBM MQ | IBM
Apache foundation	Apache Corporate Governance
FTP standard	RFC 959: File Transfer Protocol (rfc-editor.org)
FTPS standard	RFC 4217: Securing FTP with TLS (rfc-editor.org)
SSH protocol standard	RFC 4251 - The Secure Shell (SSH) Protocol Architecture (ietf.org)
AS2 standard	RFC 4130 - MIME-Based Secure Peer-to-Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2) (ietf.org)
Websocket standard	RFC 6455 - The WebSocket Protocol (ietf.org)
SOAP recommendation	SOAP Specifications (w3.org)
REST definition	Architectural Styles and the Design of Network-based Software Architectures (uci.edu)
Multicast IP standard	RFC 1112: Host extensions for IP multicasting (rfc-editor.org)