TECHNICAL SPECIFICATION

ISO/TS 19468

First edition
2019-10

Intelligent transport systems — Data interfaces between centres for transport information and control systems — Platform independent model specifications for data exchange protocols for transport information and control systems

Systèmes de transport intelligents — Interface de données entre centres pour les systèmes de commande et d'information des transports — Spécification du modèle indépendant de plateforme pour les protocoles d'échange de données pour les systèmes de commande et d'information des transports

Reference number
ISO/TS 19468:2019(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*, in collaboration with the European Committee for Standardization (CEN) Technical Committee CEN/TC 278, *Intelligent transport systems (ITS)*, in accordance with the Agreement on technical cooperation between ISO and CEN (Vienna Agreement).

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

This document defines a common set of data exchange specifications to support the vision of a seamless interoperable exchange of traffic and travel information across boundaries, including national, urban, interurban, road administrations, infrastructure providers and service providers. Standardisation in this context is a vital constituent to ensure interoperability, reduction of risk, reduction of the cost base, promotion of open marketplaces and many social, economic and community benefits to be gained from more informed travellers, network managers and transport operators.

Especially in Europe, delivering transport policy in line with the White Paper issued by the European Commission requires co-ordination of traffic management and development of seamless pan European services. With the aim to support sustainable mobility in Europe, the European Commission has been supporting the development of information exchange mainly between the actors of the road traffic management domain for a number of years.

This document supports a methodology that is extensible.

To be able to successfully connect systems and start exchanging data, in an interoperable and easy way, there is a need to describe and agree on how the exchange should be done. This is set out in a data exchange specification. Data exchange in different scenarios can have different needs and requirements. Therefore, several data exchange specifications can be needed.

Data exchange specifications need to address two main issues. First, they model the stakeholders and actors involved in data exchange, each potentially in different roles, as well as abstract exchange patterns for their interactions. Second, they select a suitable implementation platform and clearly specify how the abstract scenarios and patterns are effectively implemented on this platform.

The following diagram in Figure 1 shows such an abstract communication scenario from the perspective of a road operator who requires data exchange interfaces between the different components of its own operational systems, either between centre side components or between centre and field devices, but also to exchange information with other road operators or service providers.

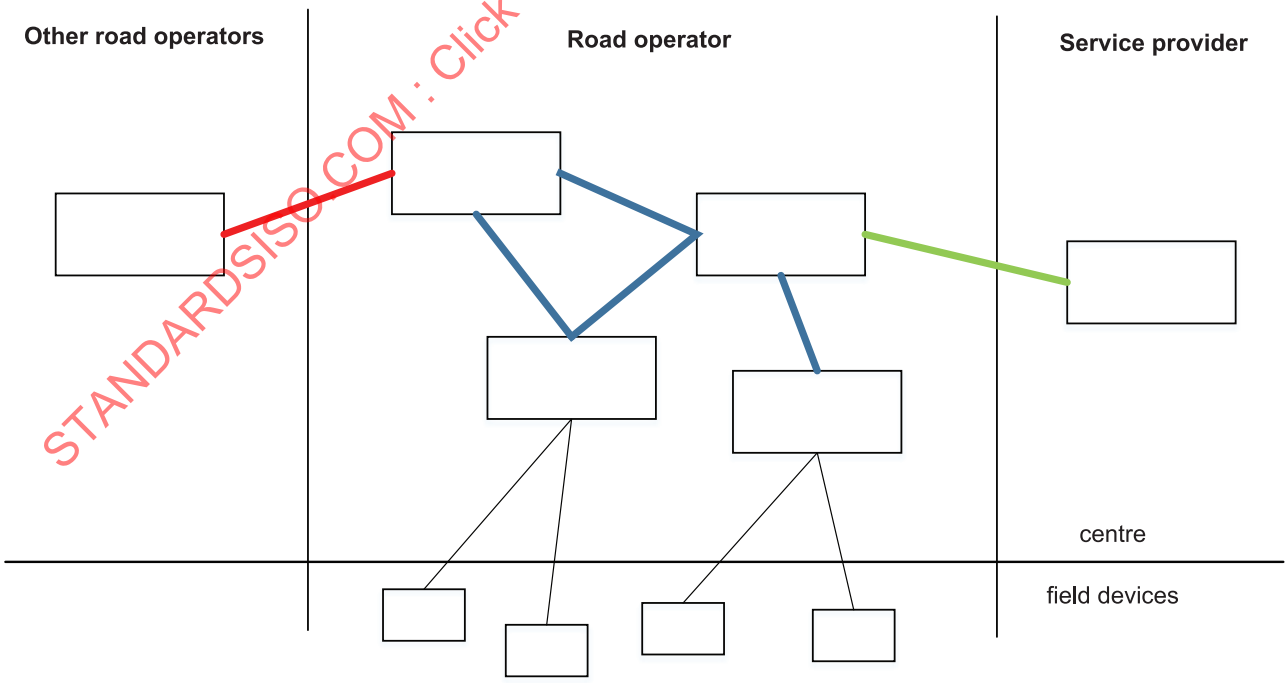

Figure 1 — Abstract communication scenario

While the black links between centre side components and field devices may use a variety of communication protocols, mostly depending on the physical link conditions, the vast majority

of other coloured links between centre-side components, internal to one organisation or external to others, is based on an IP network and mostly use the TCP transport layer protocol (UDP is also possible in a few cases).

Nevertheless, as the different colours indicate, they can very well have significantly different requirements. Internal links (blue) can reside in one domain of trust, hence do not require protocols compatible with security gateways. This can already be different for links to other road operators (red) and will certainly not hold for links to other types of organisations, like service providers, via the Internet (green).

While different security requirements offer the most striking and obvious example, there are more criteria that can lead to different preferences on different types of links, e.g. scalability, robustness, integration complexity.

In broad terms, the colours blue – red – green form a hierarchy from more internal, closely-coupled, well-integrated systems towards external, loosely-coupled, and non-integrated systems. The world of *information and communication technology* (ICT) offers a broad range of solutions for these different scenarios, offering different advantages and disadvantages. It is obvious that the *one-size-fits-all* principle will not provide the most efficient way of working here. Even on the highest level of abstraction and inside the ICT domain itself, we already find a well-known battle of paradigms between *remote-procedure-call* (RPC) type service specifications and *RESTful* architectures. The same clusters of options are found in the domain of ITS standards, where for example the European standard for the real-time information interface relating to public transport operations (SIRI – EN 15531 series) introduces both concepts as complementary options: *Publish-Subscribe* and *Request-Response*.

As well, the ITS station architecture is not in contradiction with this document but is complementary of what is defined in this document. According to the principles and the taxonomy defined in ISO 21217, this document defines a conceptual notion of:

— How 2 *central ITS (sub) stations* could communicate to:

  — deliver (*application data units*) information,

  — negotiate functional service behaviour for collaborating traffic management functions (even if this use case could not directly be matched to ISO 21217 as it is not about information delivery).

— How a *Central ITS (sub) station* could communicate to deliver information (*application data units*) to another *ITS station* with the characteristics of a central ITS station.

This document specifies the process of defining the exchange characteristics by use case-driven feature selection of relevant parameters for the relevant OSI layers as defined in ISO 21217. Two exchange schemas are considered: *Information delivery* and *Functional service negotiation* between central ITS stations.

The drafting of this document was guided by the following principles:

— Interoperability, such that different implementations can successfully engage in a data exchange process;

— Support legacy implementations which are based on existing (exchange) specification, in order to maximize investments already made by stakeholders;

— Address other user profiles, not only road operators, and thus make this document available to a broader audience;

— Reuse of existing (communications) standards, in order to reduce implementation complexity and take benefit of proven and already existent solutions for common ICT problems;

— Clear separation between the payload content and the exchange model.

The informative Annex A details the adopted methodology for defining this exchange Platform Independent Model.

Intelligent transport systems — Data interfaces between centres for transport information and control systems — Platform independent model specifications for data exchange protocols for transport information and control systems

1 Scope

This document defines and specifies component facets supporting the exchange and shared use of data and information in the field of traffic and travel.

The component facets include the framework and context for exchanges, the data content, structure and relationships necessary and the communications specification, in such a way that they are independent from any defined technical platform.

This document establishes specifications for data exchange between any two instances of the following actors:

— Traffic Information Centres (TIC);

— Traffic Control Centres/Traffic Management Centres (TCC/TMC);

— Service Providers (SP).

This document can be applied for use by other actors, e.g. car park operators.

This document includes the following types of information:

— the use cases and associated requirements, and features relative to different exchange situations;

— the different functional exchange profiles;

— the abstract elements for protocols;

— the data model for exchange (informational structures, relationships, roles, attributes and associated data types required).

In order to set up a new technical exchange framework, it is necessary to associate one functional exchange profile with a technical platform providing an interoperability domain where plug-and-play interoperability at technical level can be expected. The definition of such interoperability domains is not part of this document but can be found in other standards or technical specifications, e.g. ISO 14827-3. This document is restricted to data exchange. Definition of payload content models is beyond the scope of this document.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
business scenario
high level description of the interactions that can exist within a system being analysed or between the system and external entities (called actors) in terms of business functions

Note 1 to entry: See also *use case* (3.20).

3.2
client
entity that receives the information

Note 1 to entry: It is represented in the information delivery *business scenario* (3.1).

3.3
Functional Exchange Profile
FEP
selection of data exchange features for a particular *business scenario* (3.1)

3.4
HTTP server
software application that provides content to client applications by using the HTTP protocol

3.5
interoperability domain
pair of *Functional Exchange Profile (FEP)* (3.3) and platform selected for implementing a data exchange subsystem

Note 1 to entry: Each Platform Specific Model (PSM) document defines an interoperability domain, which ensures that two implementations of this PSM are interoperable and can successfully exchange payload.

3.6
payload content model
content model
UML definition of the data structures that can be used to describe travel and traffic information to be exchanged in an exchange system

3.7
payload publication
payload
bundle of information that is exchanged between two exchange systems containing an instance of the *content model* (3.6)

3.8
Platform Independent Model
PIM
document describing the abstract model of the standardised data exchange process in a platform-independent way

Note 1 to entry: This definition is specific to this document.

3.9
Platform Specific Model
PSM
document providing the implementation details of a *Functional Exchange Profile (FEP)* (3.3) described in a *Platform Independent Model (PIM)* (3.8) for a concrete platform

Note 1 to entry: This definition is specific to this document.

3.10
profile-to-platform mapping
act of defining an *interoperability domain* (3.5)

3.11
pub/sub
publish-subscribe pattern

3.12
pull exchange
exchange pattern where the exchange of information is originated by the client

3.13
push exchange
exchange pattern where the exchange of information is originated by the *supplier* (3.19)

3.14
simple push
push-based exchange pattern where that does not require state to be maintained

3.15
snapshot
set of data providing all of the last known state as opposed to providing partial changes

Note 1 to entry: This definition is specific to this document.

3.16
snapshot pull
pull-based exchange pattern where only the last snapshot version is exchanged

3.17
snapshot push
push-based exchange pattern where only the last snapshot version is exchanged

3.18
stateful push
push-based exchange pattern where data describing a communication session is maintained across successive communication within that session

3.19
supplier
entity that provides the information

Note 1 to entry: It is represented in the information delivery *business scenario* (3.1).

3.20
use case
UC
set of operational interactions between entities (called actors) and a system to ease understanding the main functions behind such interactions

4 Symbols and abbreviated terms

ASN.1	Abstract Syntax Notation One
BUC	Business use case
HTTP	Hypertext Transfer Protocol
ICT	Information and Communication Technology
IP	Internet Protocol
ITS	Intelligent Transport Systems
MDA	Model Driven Architecture
REST	Representational State Transfer
RPC	Remote Procedure Call
SOAP	Simple Object Access Protocol
TCP	Transmission Control Protocol
TMP	Traffic Management Plan
UDDI	Universal Description Discovery and Integration
UDP	User Datagram Protocol
UML	Unified Modeling Language
	NOTE Specified in ISO/IEC 19505.
VMS	Variable Message Sign
W3C	World Wide Web Consortium
WSDL	Web Service Definition Language
WSIL	Web Services Inspection Language
WSS	Web Services Security
XML	eXtensible Markup Language

5 Exchange modelling framework

5.1 Overview

The model driven approach is chosen to describe exchange: this leads to describe exchange systems by means of abstract models which are named Platform Independent Model (PIM), in which modelling of exchange features is done by describing interaction among systems and subsystems as exchange patterns. These interactions implement system capabilities as features which fulfil exchange requirements requested by specific business scenarios which are used to specify specific uses of exchange.

Since simple data exchange process is no longer sufficient for resolving all business needs, this document encompasses more business functions as stakeholders consolidate their systems and look at new ways to address the requests they encounter with the tools they already know and have in place.

5.2 Business scenarios and Functional Exchange Profile (FEP)

This document is based on business scenarios, i.e. a high-level description of the interactions that can exist within a system being analysed or between the system and external entities (called actors) in terms of business functions. It is derived from requirements a business scenario has on information as well as technical parameters. FEPs are identified to ensure interoperable services with the restriction of determining one FEP per business scenario for a specific technical platform. One FEP can support more than one business scenario (see Figure 2).

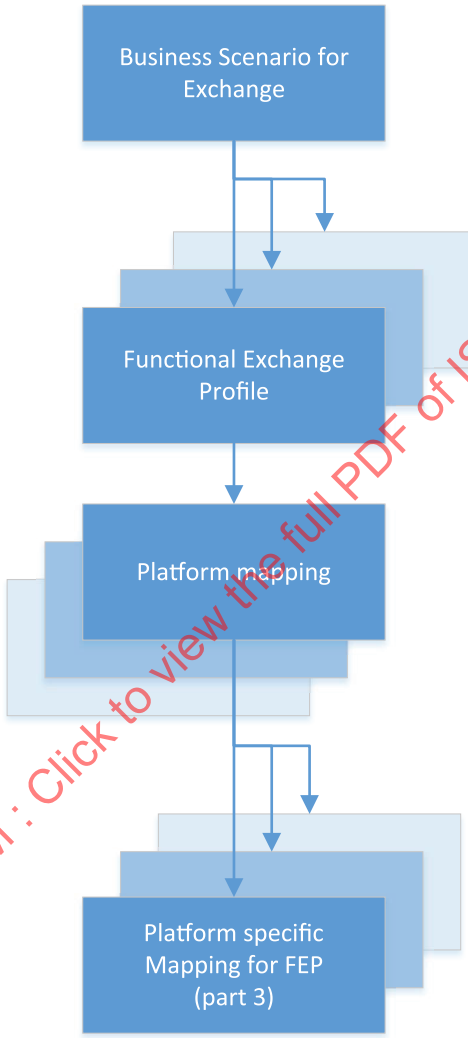

Figure 2 — Business scenario and Functional Exchange Profile (FEP)

This version of the document addresses the following business scenarios:

— Information delivery,

— Collaborative ITS services (partly).

Other business scenarios can be developed in future versions using the same methodology.

5.3 Requirements, feature and exchange patterns

Requirements can vary depending on data exchange applications, i.e. use cases to be fulfilled, so there are many reasons to consider or not any requirement based both on the gathering of data at supplier system and the usage of the delivered data in the client.

Requirements address the following items:

— Information provision,

— Communications,

— Security,

— Financial aspects.

Exchange is defined through **features** which implement the information exchange and fulfils data exchange requirements.

Depending on many possible exchange platforms considered for implementation, a subset of features and relative requirements is possible to be implemented. To fulfil a set of requirements many platforms are possible, but to be interoperable a client and a supplier shall implement the same platform with the same pattern. Allowing a wide variety of possible **exchange patterns**, the best suitable for an application is chosen, according to requirements the fulfilment of which is granted by the selected exchange pattern.

Based on the requirements of a specific business scenario, a set of appropriate exchange features shall be combined into a Functional Exchange Profile (FEP).

The following schema in Figure 3 represent the domains of PIM and PSM introducing exchange pattern (EP) and Functional Exchange Profile (FEP).

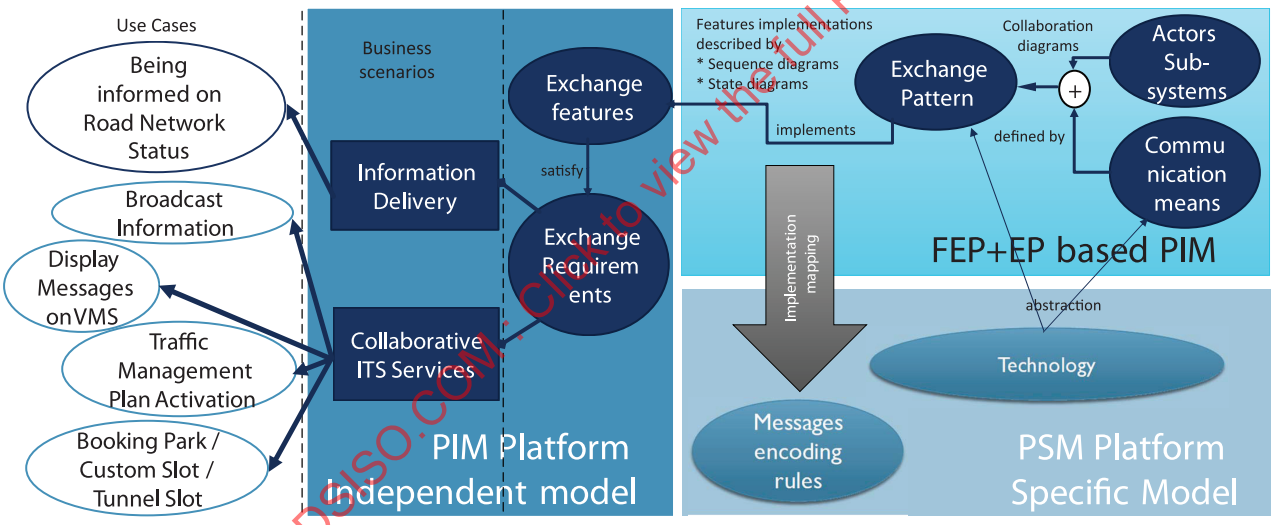

Figure 3 — Business scenario and Functional Exchange Profile (FEP)

5.4 Business scenario: Information delivery

5.4.1 Overview

One of the most common applications of a data exchange system is the exchange of traffic and travel information between two nodes. In such a scenario, one node acts as the supplier of the information while the other acts as the intended receiver of that information, i.e. the client.

EXAMPLE It is done by using the form of publications, e.g. in the European DATEX II.

The data delivery business scenario considers the actors as defined in the following Figure 4:

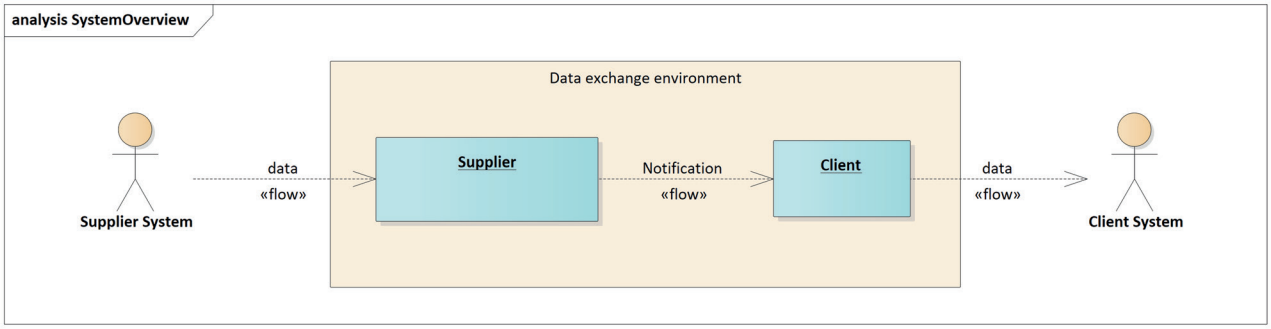

Figure 4 — General data delivery business scenario actors

The following table provides the basic definitions for exchange:

Table 1 — Main definitions in exchange

Name	Definition
Supplier system	A system which gathers information (road information) which needs to be conveyed to another system named client system. Examples of supplier systems are traffic control centres or traffic information centres or service provider systems, gathering road data from any available source they have.
Client system	A system which needs to update its internal information based on information which is available from another system named supplier. Examples of client systems are traffic control centres or traffic information centres or service provider systems.
Exchange environment	The set of components which enables information exchange among client systems and supplier systems via a data exchange protocol.
Supplier	The component of exchange environment which is devoted to providing data to client, retrieving them from supplier system.
Client	The component of exchange environment which is devoted to collecting data from supplier, delivering them to the client system.

Road and traffic information is gathered in a system which is named “Supplier System” in case the information it stores is needed to be transferred to another system, named “Client System”, for any purpose.

The data delivery business scenario describes the exchange pattern and messages which are needed to be exchanged among supplier and client systems besides the underneath technology and exchange pattern. The purpose for which information is exchanged is not considered in this use case description.

As explained in the information delivery background in Annex F, any update of information status at the supplier system shall be replicated to the client system via information delivery. The main objective of information delivery is that information on the client system is updated exactly in the same way as it is in the supplier system without any difference in information values and semantics.

“Exchange message” is defined as the data structure in which the information is coded to transfer information in the exchange system from the supplier to the client.

Assuming Sc = Client status, Ss = Supplier status, exchange is a mean to have Sc equivalent to Ss,

Formally:

Ss → Information → Supplier → Exchange Message → Client → Information → Sc

i.e. client system status is updated in equivalent mode as supplier system status by means of data delivery exchange messages between supplier and client.

Information delivery business scenario scope is implemented by selected exchange features which enable this scope and other secondary requirements which are based on available features on considered platforms and patterns.

1) Supplier system characterisation

   — Shall provide data as input to the data exchange environment

   — Is mandatory for the information data delivery process

2) Data exchange environment

   — Shall be an environment supporting the exchange of information and data by mean of messages

   — The supplier of data exchange system shall produce and transmit the messages (Notification)

   — The client of data exchange system should receive and process the messages

3) Client system characterisation

   — Shall receive traffic and travel related data from the data exchange environment

   — Can be either another system for further processing or a simple client for the content visualization, the purpose of information exchange is not considered to be relevant in information delivery business scenario

   — Is mandatory for the information data delivery process.

5.4.2 Requirements

The normative Annex B provides a description of all requirements that exist for specific business scenario including information delivery whether they are actually used in a particular FEP or not. All requirements are organised into groups based on their characteristics.

NOTE A FEP is a different concept from content profile. The description of the content profiles that can exist in an information delivery scenario is not in the scope of this document. Content profiles are handled in respect to which information is exchanged.

5.4.3 Data delivery exchange pattern

For data delivery the concept of "exchange pattern" is introduced, as well as PIM and FEP. It can be defined as follows:

Exchange pattern: the basic exchange architecture model which identifies actors in the exchange framework and messaging patterns as basic tools which are available to implement information delivery by exchange features.

Messaging patterns can be defined by means of UML sequence diagrams, collaboration diagrams and state diagrams, in a way that messages triggering conditions are well defined and any update of state is well defined and identified based on the subsequent message exchange or conditions.

Examples of exchange pattern are the GET method of HTTP, Pull, Push, PubSub, etc.

Once an exchange pattern and a selection of features (FEP) which can be implemented on this exchange pattern are set, a set of specification at PIM level for exchange pattern/FEP is well defined.

A PIM for exchange pattern/FEP that enables all mandatory requirements is a valid platform for information delivery business case.

5.4.4 Specific exchange pattern specification PIM included in this document

In the following clauses PIM for exchange pattern/FEP will be described for the following platform:

— “Snapshot Pull”;

— “Snapshot Push”;

— “Simple Push”;

— “Stateful Push”;

— “PubSub”, Publish Subscribe Pattern [based on OASIS WS Notification Specification (2006)].

The informative Annex E details the features of the main FEPs considered in this document for the corresponding exchange pattern.

5.5 Business scenario: Collaborative ITS Service (CIS)

5.5.1 Overview

In CIS business scenario the exchange of information among centres is considered as a base to trigger processes and services on this data, so the data exchange layer is to be considered as an “ITS services” enabler.

Information about CIS concept and its model description is found in Annex G.

5.5.2 Data exchange enabling service request and feedback paradigm

The involved systems which in data delivery had been considered as supplier and client can be seen in this paradigm respectively as service provider and service requester.

Note that at application level for implementing a business case such as management of TMP, some workflow modelling is normally requested. This could be simple workflow (accept request or reject request) or more complex: This workflow modelling at application level is out of the scope of this document, which only provides the essential tools to a raise service request and provide feedback: exchange of payload to be processed and processing results:

— **Collaborative** means 2 to many systems interoperating

— 1 to n Data exchange connection(s)

  — Modelling any single CIS usage invoking with

    — 1 Service Requester

    — many Service Providers

  — Reducing complex combination of single usages from many requesters

    — out of scope, not needed to be described

    — question on possible lock of resources to be managed at application or human operator level not described in the document

— ITS Services definition, just to give a reference, service description is out of the scope of this document

— Specific definition of ITS Service in the informative Annex G:
  — TIS / Traffic Information Services
    — Information delivery
      — Allowed channels
  — TMS Traffic Management Systems
    — Hard shoulder running
    — Dynamic speed
    — Dynamic lane management
    — etc.
  — F&L Freight & Logistic
    — Secure truck parking
    — etc.

5.5.3 Requirements

The normative Annex B provides a description of all requirements that exist for specific business scenarios including Collaborative Informative Services.

5.6 Exchange data model

To implement some exchange features, such as session management or link monitoring, or security features, extra information is to be added to the informational (payload) content. This extra information, named exchange information, enables messages to convey information and data which are related to client and supplier status, identity, authorisation, etc.

Exchange information could be different among exchange pattern/FEP selection, but some common general information models are considered, which can be specialised to manage specific exchange pattern and PIM.

A general UML model for managing a minimum set of information for exchange is provided in C.2.

5.7 Data exchange features

5.7.1 Context diagram

This document defines the features and rules that systems shall implement in order to be able to communicate in the traffic and travel data exchange world.

The context diagram below in Figure 5 identifies the entities and the features that are specified in this document and which need to be addressed by the technical implementations. The diagram presents the features in different layers for communication, message rules and application.

— The **Application** layer is used for defining how using and implementing different business and application needs. This document describes the features to deal with how and when the information is published and made available, how subscriptions are managed, how to handle services and transactions. This layer defines the semantics of the communication.

— The **Messages Rules** layer defines the features and the rules used for the transport of the messages. This is the layer where the rules (protocol) are defined that enable different systems (suppliers and

clients) to communicate and understand each other, i.e. for sending and receiving messages. This layer defines the syntax of the communication.

— The **Communication** layer deals with the support for communication between systems, and defines the protocols and services that are used by the data exchange applications, e.g. TCP/IP, security, Web Services. The communication layer deals with rules at the lowest level, i.e. the physical exchange. This layer provides solutions to the defined requirements and features although it is up to upper applications to define what protocols to use and how to use them. This layer defines the physical support for the communication.

Figure 5 shows the topics broken up into boxes representing the exchange features and the relation with each layer.

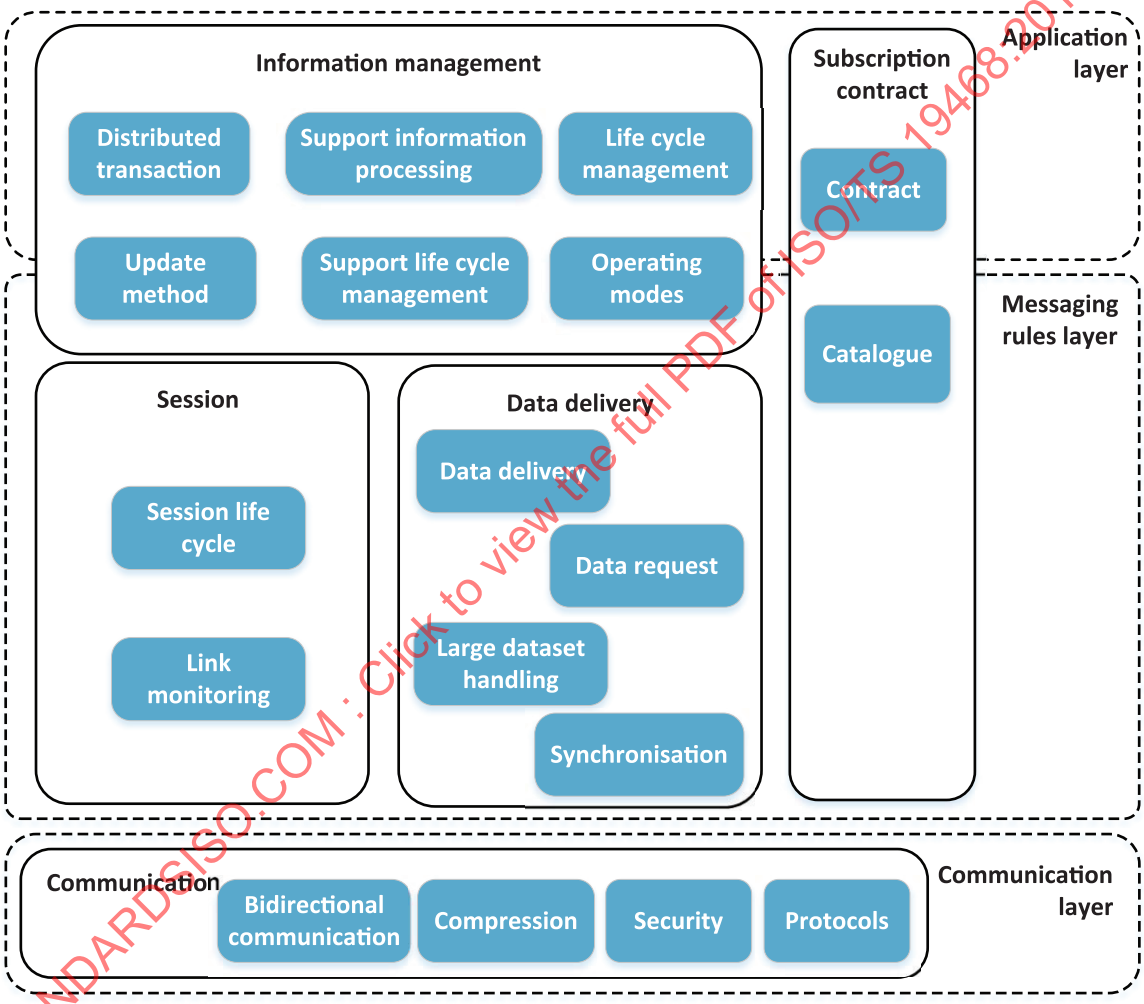

Figure 5 — Context diagram

5.7.2 Features

The features that support the requirements defined by use cases specified in this document are detailed below:

1) **Subscription contract** – Supports all features related to the contract or agreement, such as:

   i) Contract and contract life cycle: a model and features that can be used for the support of the information of a subscription contract;

   ii) Catalogue: a model for handling catalogues.

The subscription contract is optional and includes the following features:

Table 2 — Subscription contract: Features and requirements

Features	Requirement type	Requirement name
Contract	Information	Subscription
		Client profiles
		Filter handling
Catalogue	Information	Catalogue exchange

2) **Session** – The session feature group supports all features related to the establishment of a logical session.

   i) Session life cycle – features for managing the life cycle of a logical session (create, manage and terminate);

   ii) Link Monitoring – features for link monitoring and control.

The session feature group is optional and includes the following features:

Table 3 — Session: Features and requirements

Features	Requirement type	Requirement name
Session life cycle	Communication	Error handling
		Time-out management
		Session
Link monitoring	Communication	Error handling
		Time-out management
		Full reliability
		Link monitoring and control

3) **Information management -** Handles the features related to management of the information, includes features such as:

   i) Operating modes: features to specify what portion of the information shall be exchanged;

   ii) Update methods: features that let a data exchange system specify when the information should be exchanged;

   iii) Life cycle management: features for handling the life cycle management of exchanged payload information for payload for which life cycle is applicable;

      I) Situation life cycle management

      II) Filter handling

   iv) Support information processing: feature for handling directives to process exchanged data and send feedback on processing outcomes;

   v) Distributed transaction: features for handling a transaction on several systems consistently, i.e. an operation is capable to keep data consistency among several systems based on undertaken operator actions.

The information management group includes the following features:

Table 4 — Information management: Features and requirements

Features	Requirement type	Requirement name
Operating modes	Information	Reference datasets for different versions
Update methods	Information	Full audit trail data delivery (all state changes)
Life cycle management	Information	Support for life cycle management
Support information processing	Information	Support for information management
		Support for feedback on information management
Distributed transaction	Information	Distributed transaction
		Distributed atomic transaction

4) **Data delivery –** The data delivery feature group supports all features to exchange data between the supplier and client. In the publish-subscribe pattern this feature group will support all related interfaces between the producer and the consumer; it supports features such as:

   i) Data delivery: features to delivery information by the supplier to the client on a push mode (direct delivery and fetched delivery);

   ii) Data request: features to exchange information requested by the client;

   iii) Large datasets: support the exchange messages with large volumes;

   iv) Synchronisation: how to ensure data synchronisation between the systems that are communicating.

The data delivery group includes the following features:

Table 5 — Data delivery: Features and requirements

Features	Requirement type	Requirement name
Data delivery	Information	Extensibility
	Communication	Delivery/response
		Message sequence
		Snapshot data delivery (last known state)
		Exchange quality measures (ex. response times-tamp)
		On occurrence update
		Periodic update
	Security	Client identification
		Supplier identification
	Information	Incremental data delivery

Table 5 *(continued)*

Features	Requirement type	Requirement name
Data request	Information	Extensibility
	Communication	Request/response
		Message sequence
		Snapshot data delivery (last known state)
		Exchange quality measures (ex. response times-tamp)
		On occurrence update
		Periodic update
	Security	Client identification
		Supplier identification
Large datasets handling	Information	Data delivered as soon as possible
		Delayed delivery
		Multi-part data delivery
Synchronisation	Information	Synchronisation
	Communication	With state supplier
		Failed data recovery

5) **Communication –** Handles all features related to a protocol (close to the physical layer). It is used by the application for the massage transport:

   i) Communication – describe the communication protocols that can be used;

   ii) Security – describe how security features can be implemented;

   iii) Compression – how data compression can be used while transmitting data;

   iv) Bidirectional communication – enable a client to send back data to a supplier as feedback on data exchanged and processing status of data based on supplier's processing directive in Collaborative ITS Services.

The communication function is responsible for implementing the following features:

Table 6 — Communication: Features and requirements

Features	Requirement type	Requirement name
Security	Security	Security (data)
		Integrity
		Confidentiality
		State the intended recipient
		Client authentication
		Supplier authentication
		Client authorisation
		Non-repudiation
Compression	Communication	Compression
Communication	Communication	Timely responses
Bidirectional communication	Communication	Bidirectional communication

6 Snapshot Pull

6.1 Overview

The “Snapshot Pull” exchange pattern / FEP at PIM level is based on information retrieval by a client from a supplier which delivers a snapshot of information, i.e. all currently available information content, to the client. It can be implemented in several platforms: some examples are XML retrieval of generated XML files by http/get, or supplier exposing a SOAP Web Service method (e.g. named "PULL"), from which currently available data is retrieved by the client.

This “Snapshot Pull” does not manage session life cycle and link monitoring requirements, as well as synchronisation. This feature and related requirements are not considered in this pattern, synchronisation is not needed as implicit when delivering snapshots of currently available information content.

To describe the Snapshot Pull exchange pattern/FEP at PIM level all features are described in a general abstract format, independently from the specific technology platform this model will be implemented with (e.g. http/get XML, WebService).

Table 7 — Selection of features for Snapshot Pull

Features area	Feature	Snapshot Pull implemented
Subscription contract	Contract	N
	Catalogue	N
Session	Session life cycle	N
	Link monitoring	N
Information management	Operating modes	Periodic or On Occurrence (i.e. triggered by client conditions)
	Update methods	Snapshot
	Life cycle management	Y, based on snapshot: new and updated content delivered, outdated data not delivered.
Data delivery	Data delivery	Y
	Data request	N
	Large datasets handling	N
	Synchronisation	Y
Self-Description	Handshake	N
Communication	Security	To be defined at PSM level
	Compression	To be defined at PSM level
	Communication	To be defined at PSM level

6.2 Exchange pattern messages definition

6.2.1 Overall presentation

The information delivery business scenario description and definition states that data exchange is needed to align the information kept by the supplier system into the client system. For this purpose an exchange systems is used which provides tools enabling messages generation and their transfer between supplier and client.

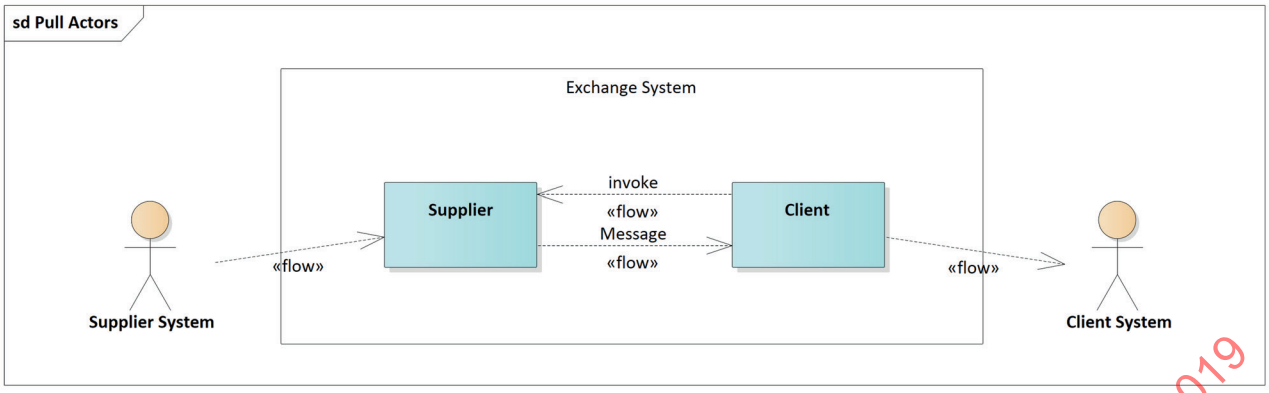

Figure 6 — Snapshot Pull exchange actors

The "Snapshot Pull" exchange pattern is described in the following subclauses.

6.2.2 Basic exchange pattern

The supplier provides a mechanism to retrieve current data, also called "snapshot" of information, normally for a specific payload, i.e. current information content of client system or last retrieved information for sampled data (see Figure 7).

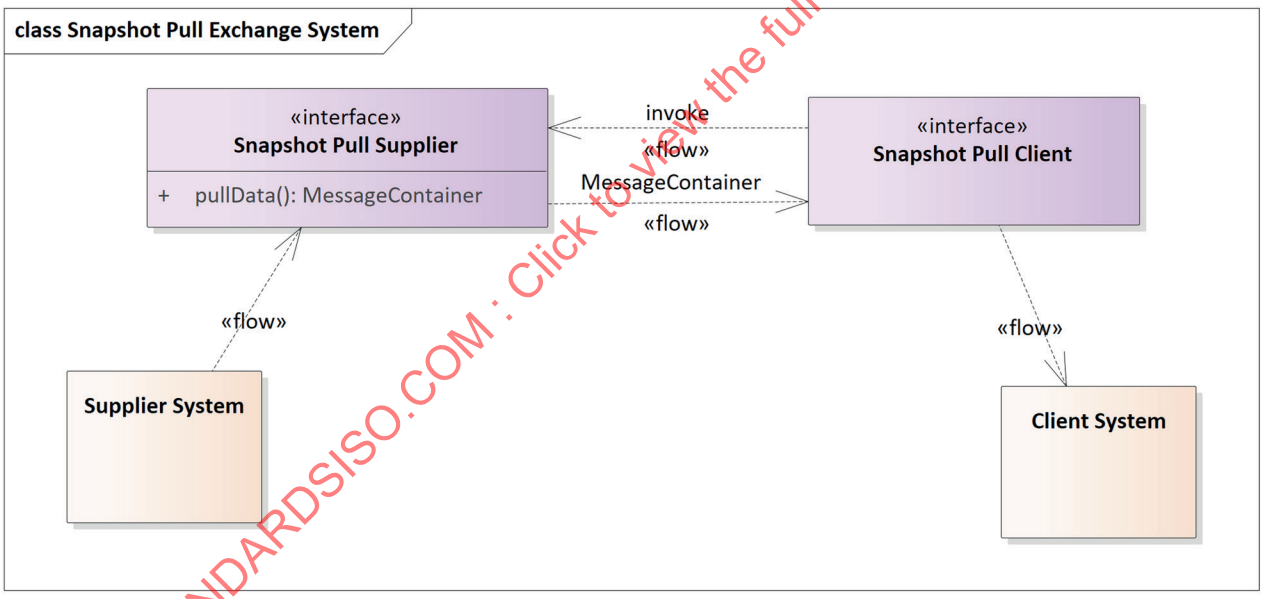

Figure 7 — Snapshot Pull exchange subsystems, interface interactions and methods

This mechanism is referred to in this document as "**pullData**".

The client takes the initiative to retrieve the data based on its needs, normally when some conditions defined at client side are triggered or in a periodic way; specifically:

— **Periodic Pull**, based on fixed or variable time cycle, depending on application logic at the client system;

— **On Occurrence** (Triggered Pull), on any condition stated by the client or by the client system.

NOTE Depending on implementation (e.g. http /get of XML static files generated at supplier side) the payload message is generated by the supplier based on condition on the supplier side (e.g. event update or data gathered at the supplier side). In these cases, extra information can be available to implement some optimisation such as bandwidth saving by not transferring the same data in case no update had been generated.

6.2.3 Relevant exchange information in exchange data model

No extra exchange information is needed in this pattern to implement any described features.

Basic exchange data model has been provided to allow the implementation to deliver more payload content on the same message and further information to allow managing extra features not required by the Snapshot Pull exchange. The usage of the exchange data model wrapping is for harmonisation of other exchange patterns such as "Simple Push" or "Stateful Push".

A “MessageContainer” instance should be retrieved using the basic exchange data model as reported in previous figure, alternatively a payload may be retrieved without any further information.

6.2.4 Exchange messages

— Payload Message.

  — Simple payload messages can be exchanged within this FEP+EP.

  — Payload messages should be delivered wrapped into a container (see basic exchange data model in the normative Annex C with exchange data).

  — Payload messages contain payload update timestamp which can be used to understand when payload has been created/updated for error management and processing saving.

6.3 Features implementation description

6.3.1 Overview

This subclause provides a description and the corresponding specification for each feature identified in the context diagram, according to the Snapshot Pull exchange architecture. The following features are specified:

— Subscription contract;

— Subscription (also known as session);

— Information management;

— Data delivery;

— Communication/protocol.

6.3.2 Subscription contract

6.3.2.1 Contract

Managed offline, not automated. It assumes information for controls to be implemented in client to assess the identity of supplier and authenticate the supplier request in message exchange.

6.3.2.2 Catalogue

Managed offline, not automated.

6.3.3 Session

6.3.3.1 Session life cycle

No session is managed for the current EP+FEP.

6.3.3.2 Link monitoring

Link monitoring is not managed for the current EP+FEP.

6.3.4 Information management

6.3.4.1 Operating modes

Available operating mode for client Pull is Periodic, or On Occurrence (i.e. a condition triggered based on client). Pull-exchange based on client-side conditions.

6.3.4.2 Update methods

Available update method is Snapshot, i.e. retrieval of only currently valid data.

6.3.4.3 Life cycle management

Currently available information is included in the payload at a supplier system to prepare message delivery. It can be done at time-out on a cycle basis or at specific triggering condition as "data updated" condition.

For life cycle management information, a snapshot includes all active information, outdated information is not delivered in content.

For sampled information the snapshot information contains last sampled data available at supplier site.

Whenever a condition is raised the supplier system triggers it to the supplier to manage the creation of a pull message.

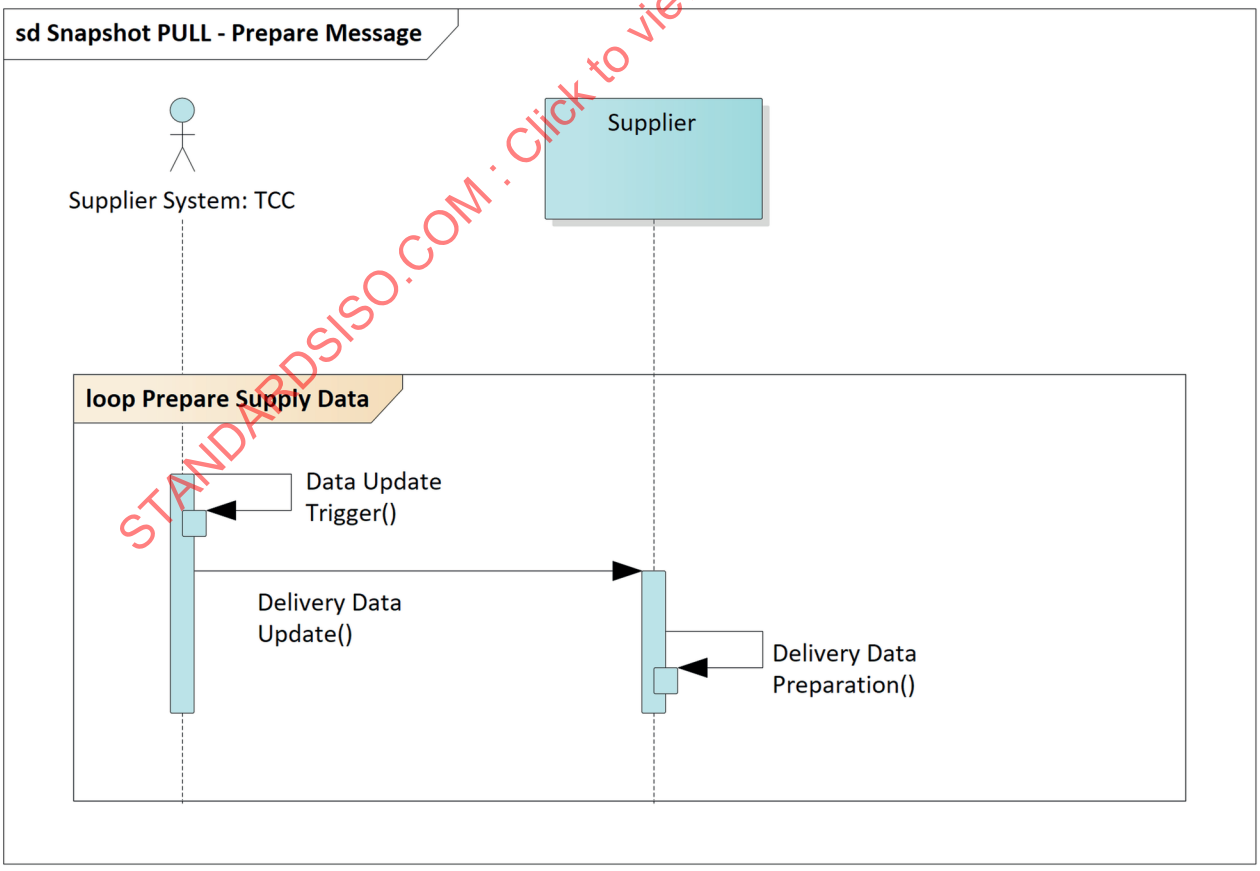

Figure 8 — Snapshot Pull sequence diagram: Information management at supplier side

Information management for Snapshot Pull is implemented as follows: when client gets a snapshot of the last updated/created items including all active items, it has to check for information which has been removed from payload to deduce it had been invalidated (i.e. closed or cancelled).

6.3.5 Data delivery

6.3.5.1 Data delivery

The sequence diagram for data delivery is as follows in Figure 9:

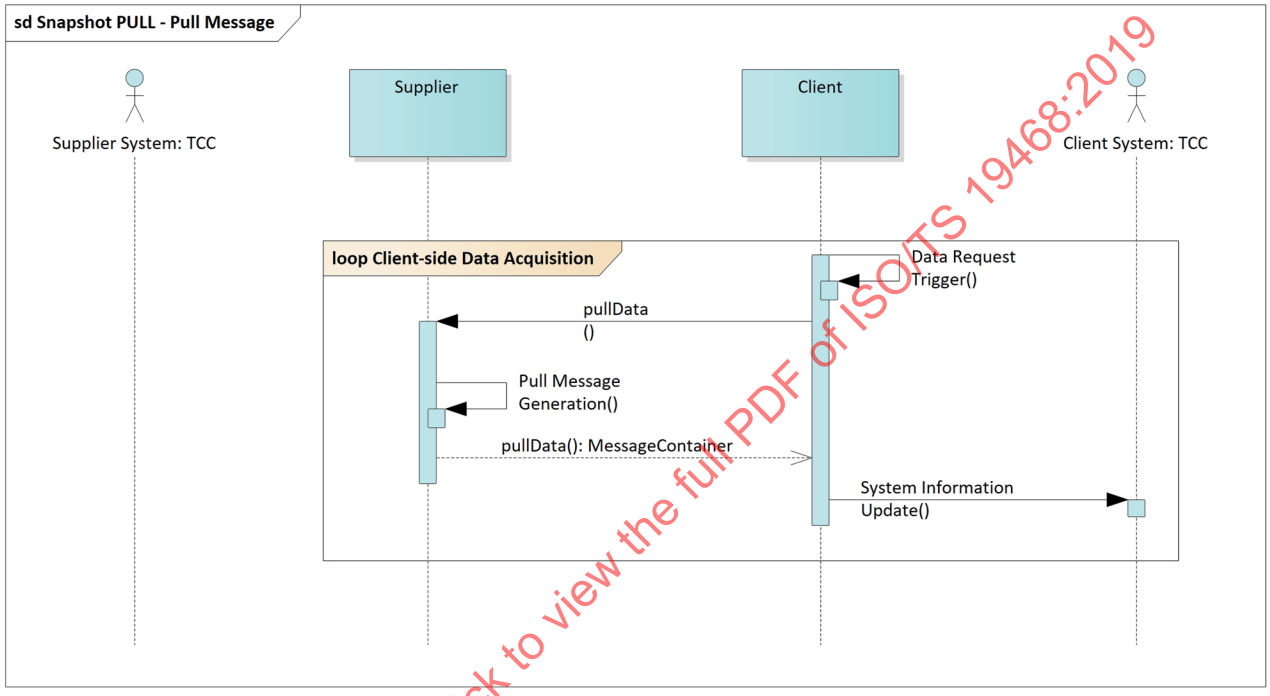

Figure 9 — Snapshot Pull sequence diagram for data retrieval: Implicit data delivery

Pull message generation processing is optional based on chosen platform technology.

6.3.5.2 Data request

Not implemented in this pattern.

6.3.5.3 Large datasets handling

Not described in this pattern at PIM level (it may be implemented at PSM level as optimisation – see 6.3.8).

6.3.5.4 Synchronisation

Implicit synchronisation is available as only currently available elements are retrieved by Snapshot Pull.

6.3.6 Self-description

Handshake is not available.

6.3.7 Communication

Communication feature are implemented at PSM level, they are relevant for the specific platform chosen on which the exchange pattern will be implemented (e.g. http/XML, Web Services SOAP, REST, etc.).

6.3.8 Optimisation issues

6.3.8.1 Bandwidth and processing savings

Payload timestamp information is available for client-side processing optimisation made at the application level.

Pull message may be generated for all clients reducing processing resources at the supplier-side.

No extra optimisation issues are considered in this EP+FEP.

6.3.8.2 Huge dataset handling

Not managed in this EP+FEP.

7 Snapshot Push

7.1 Overview

The “Snapshot Push” exchange pattern / FEP at PIM level is based on pushing information by the supplier to the client delivering a snapshot of information, i.e. all currently available information content, to the client. This exchange pattern can be implemented in several platforms: some examples are XML delivering of generated XML files by http/post, or a client exposing a SOAP web service method (e.g. named "PUSH") when currently available data can be sent by the supplier to the client.

This “Snapshot Push” does not manage session life cycle and link monitoring requirements, as well as synchronisation, these features and related requirements are not considered in this pattern. Synchronisation is not needed as implicit by snapshot delivering of currently available information content.

To describe Snapshot Push exchange pattern/FEP at PIM level all features are described in a general abstract format, independently from the specific technology platform in which this model will be implemented (e.g. http/get XML, WebService).

Table 8 — Selection of features for Snapshot Push

Features area	Feature	Snapshot Push implemented
Subscription contract	Contract	N
	Catalogue	N
Session	Session life cycle	N
	Link monitoring	N
Information management	Operating modes	Periodic or On Occurrence (i.e. Triggered by supplier condition)
	Update methods	Snapshot
	Life cycle management	Y, based on snapshot: new and updated content delivered, outdated data not delivered.
Data delivery	Data delivery	Y
	Data request	N
	Large datasets handling	N
	Synchronisation	Y
Self-Description	Handshake	N
Communication	Security	To be defined at PSM level

Table 8 *(continued)*

Features area	Feature	Snapshot Push implemented
	Compression	To be defined at PSM level
	Communication	To be defined at PSM level

7.2 Exchange pattern messages definition

7.2.1 Overview

The information delivery business scenario description and definition imply that data exchange is needed to align the information kept by the supplier system into the client system; for this purpose, an exchange system is used which provides tools enabling message generation and their transfer between supplier and client (see Figure 10).

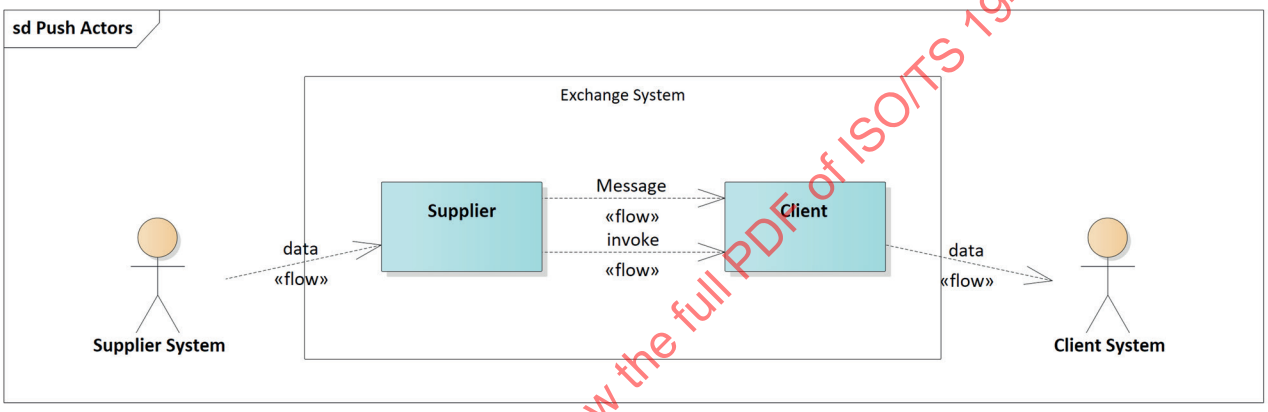

Figure 10 — Snapshot Push exchange actors

The "Snapshot Push" exchange pattern is described by the following subclauses.

7.2.2 Basic exchange pattern

The client provides a mechanism to receive data from action taken at a supplier site invoking specific resources / methods offered by the client. Then the supplier logically "pushes" messages to the client. The client shall acknowledge what is received by a return information to the supplier. This return message is available to bring some information back from the client to the supplier (Figure 11).

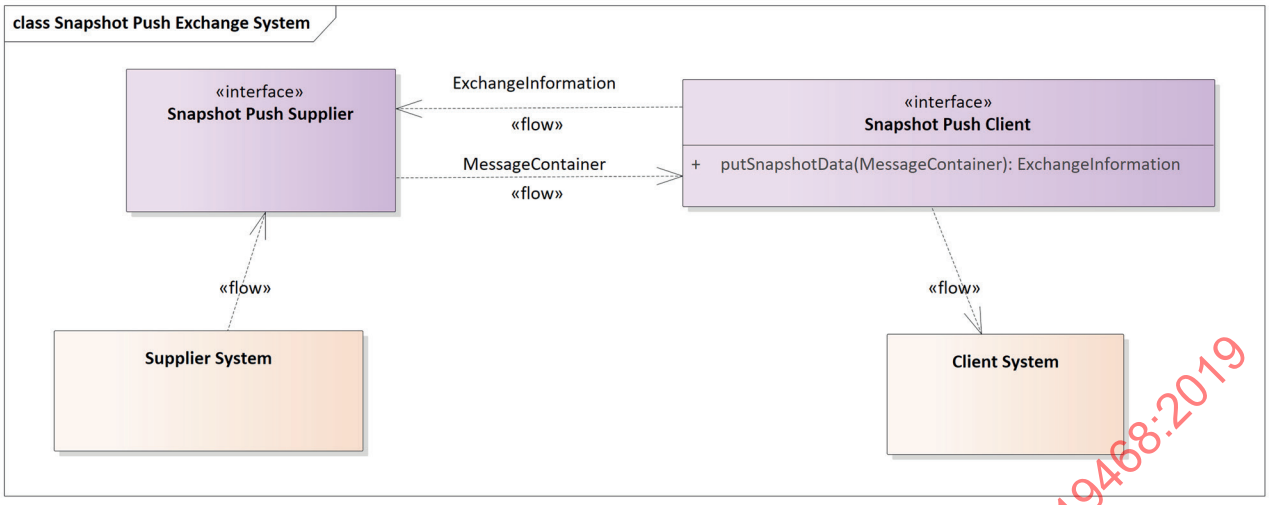

Figure 11 — Snapshot Push exchange subsystems, interfaces interactions and methods

The client provides a mechanism to the supplier to push currently available data, also called “snapshot” of information, i.e. current information content at supplier system or last retrieved information for sampled data (see Annex F).

This mechanism is referred to in this document as "**putSnapshotData**".

The supplier takes the initiative to deliver the data to the client based on some rules, normally when some conditions defined at supplier side are triggered or in a periodic way; specifically:

— **Periodic Push**, based on fixed or variable time cycle, depending on application logic at supplier system;

— **On Occurrence** (triggered push), on any condition stated by the supplier.

7.2.3 Relevant exchange information in exchange data model

No extra exchange information is needed in this pattern to implement any described features.

Basic exchange data model has been provided to allow the implementation to deliver more payload contents in the same message and further information to allow managing some extra features not required by the basic “Snapshot Push” exchange. The usage of the exchange data model wrapping is for harmonisation with other exchange patterns such as “Simple Push” or “Stateful Push”.

A container should be retrieved using basic exchange data model as reported in the previous figure, alternatively a payload may be delivered.

An ExchangeInformation is returned to convey information about exchange operation and connection status.

7.2.4 Exchange Messages

— Payload message.

  — Simple payload messages can be exchanged within this FEP+EP.

  — Payload messages should be delivered wrapped into a container (see basic exchange data model in Annex C) with exchange data.

  — Payload messages contain payload update timestamp which can be used to understand when payload has been updated for error management and processing saving.

7.3 Features implementation description

This subclause provides a description and the corresponding specification for each feature identified in the context diagram, according to the Snapshot Pull exchange architecture. The following features are specified:

— Subscription contract;

— Subscription (also known as session);

— Information management;

— Data delivery;

— Communication/protocol.

7.3.1 Subscription contract

7.3.1.1 Contract

Managed offline, not automated. It assumes information for controls to be implemented in client to assess the identity of supplier and authenticate the supplier request in messages exchange.

7.3.1.2 Catalogue

Managed offline, not automated.

7.3.2 Session

7.3.2.1 Session life cycle

No session is managed for the current EP+FEP.

7.3.2.2 Link monitoring

Link monitoring is not managed for the current EP+FEP.

7.3.3 Information management

7.3.3.1 Operating modes

Available operating mode for Snapshot Push is “Periodic”, or “On Occurrence” (i.e. a condition triggered based on supplier) Push-based on supplier-side conditions.

7.3.3.2 Update methods

Available updated method is Snapshot, i.e. retrieval of only currently valid data.

7.3.3.3 Life cycle management

Currently available information is included in the payload at supplier system to prepare message delivery; this can be done at time-out on a cycle basis or at specific triggering condition as “data updated” condition.

For life cycle management information, snapshots include all active information, outdated information is not delivered in content.

For sampled information the snapshot information contains last sampled data available at supplier site.

Whenever a condition is raised the supplier system manages the creation of a push message and delivers it (Figure 12).

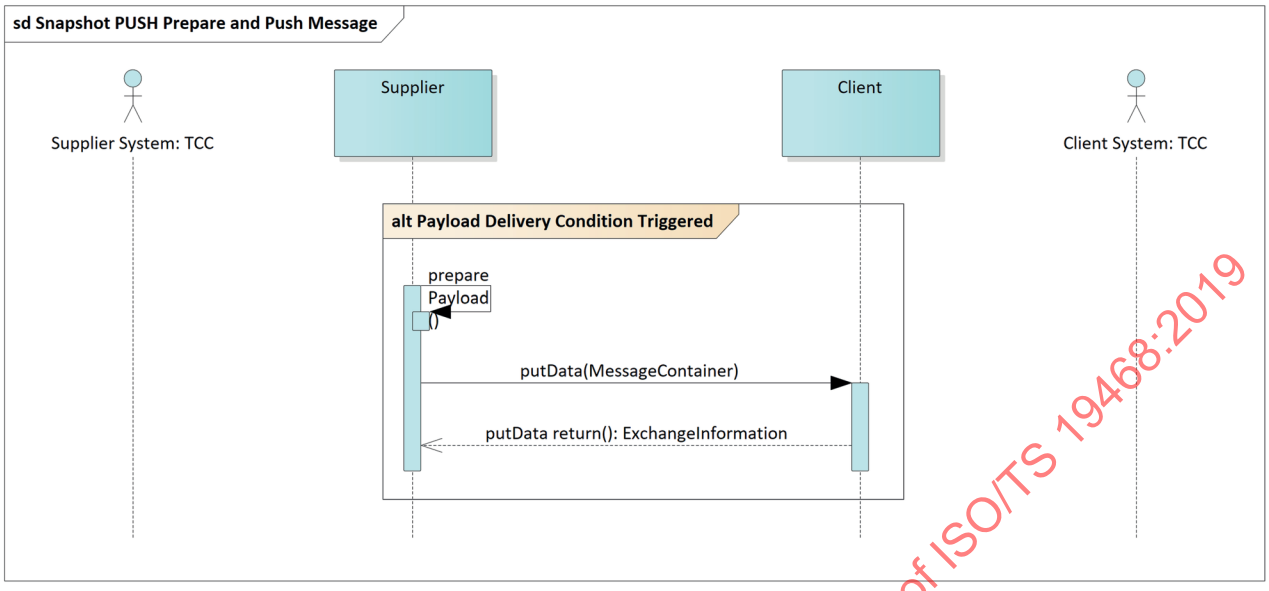

Figure 12 — Snapshot Push sequence diagram: Information management at supplier side

Information management for Snapshot Push is implemented as follows: when client gets snapshot of last updated/created items including all active items, it has to check for information which has been removed from payload to deduce it had been invalidated (i.e. closed or cancelled).

7.3.4 Data delivery

7.3.4.1 Data delivery

The sequence diagram for data delivery is the same as in the previous figure.

7.3.4.2 Data request

Not implemented in this pattern.

7.3.4.3 Large datasets handling

Not described in this pattern at PIM level. May be implemented at PSM level (see optimisation subclause).

7.3.4.4 Synchronisation

Implicit synchronisation is available as only currently available elements are retrieved by Snapshot Push.

7.3.5 Self-Description

Handshake is not available.

7.3.6 Communication

Communication features are implemented at PSM level, they are relevant for the specific platform chosen on which the exchange pattern will be implemented (e.g. http/XML, Web Services SOAP, REST, etc.).

7.3.7 Optimisation issues

7.3.7.1 Bandwidth and processing savings

Payload timestamp information is available for client-side processing optimisation made at application level.

Push message may be generated for all clients reducing processing resources at the supplier side.

No extra optimisation issues are considered in this EP+FEP.

7.3.7.2 Huge dataset handling

Not managed in this EP+FEP.

8 Simple Push

8.1 Overview

Simple Push exchange pattern / FEP at PIM level is based on information messages sent or pushed by the supplier to a client. It can be implemented in several platforms: some examples are push of generated XML content by http/post, or client providing a SOAP WebService method "push" by which data can be provided by the supplier to the client.

This "Simple Push" adds extra features to the basic Snapshot Push exchange pattern to manage link monitoring, as well as synchronisation/realignment in case of communication lacks or system maintenance. This mechanism will be detailed at PIM level in the following subclauses.

To describe the exchange pattern/FEP at PIM level all features are described in a general abstract format, independently from the specific technology platform in which this model will be implemented. (e.g. http/get XML, WebService).

Table 9 — Selection of features for Simple Push

Features Area	Feature	Simple Push available
Subscription contract	Contract	N
	Catalogue	N
Session	Session life cycle	N
	Link monitoring	Y
Information management	Operating modes	Periodic / On Occurrence based on triggering of supplier condition
	Update methods	Snapshot, Single Element Update, All Element Update
	Life cycle management	Y
Data delivery	Data delivery	Y
	Data request	N
	Large datasets handling	N
	Synchronisation	Y, optional
Self-Description	Handshake	N
Communication	Security	At PSM level
	Compression	At PSM level
	Communication	At PSM level

8.2 Exchange pattern messages definition

Information delivery business scenario description and definition state that data exchange is needed to align the information kept by the supplier system into the client system; for this purpose, an exchange system is used which provides tools enabling messages generation and their transfer between a supplier and a client (Figure 13).

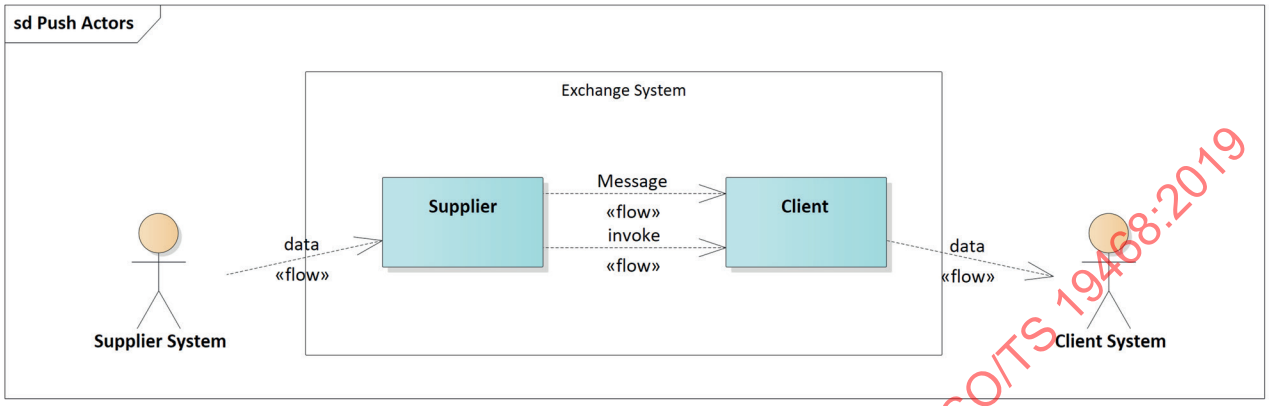

Figure 13 — Simple Push exchange actors

The “Simple Push” exchange pattern is described in the following subclauses.

8.2.1 Basic exchange pattern

The client provides a mechanism to receive data from action taken at the supplier site invoking specific resources / methods offered by the client.

Then the supplier logically “pushes” messages to the client. The client shall acknowledge what it received by a return information to the supplier. This return message is available to bring information back from the client to the supplier, such as failure, success, snapshot synchronisation request. Return message information is logically described in this PIM, while implementation will be defined at PSM, level (Figure 14).

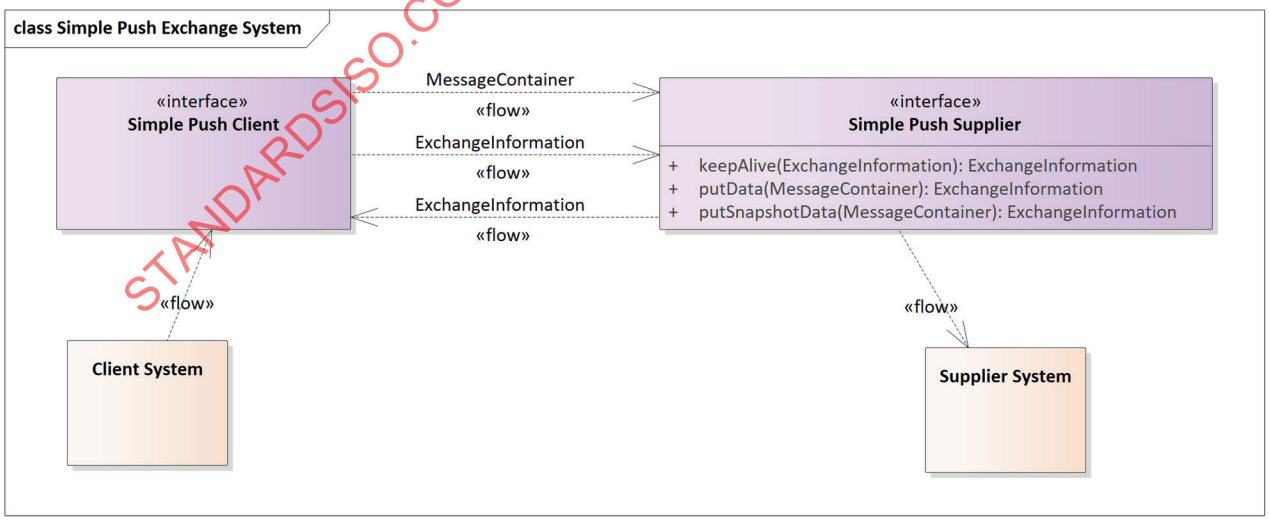

Figure 14 — Simple Push exchange subsystems, interface interactions and methods

The mechanisms available are referred to in this document as **"putData", "keepAlive"** and optional **"putSnapshotData".**

Simple Push actually pushes available information or available updated information to the clients, but does not keep track of what has been delivered to specific clients. For specific use cases such as delivery of stateful information (e.g. situation payload information), a first snapshot can be useful and is delivered to the client based on information to be exchanged or on explicit client request in return code. This feature is optional in this FEP+EP PIM.

Based on previous optional requirements, exchanged data can either be currently available data, also called “snapshot” of information, or, when a synchronisation is not needed such as for stateless information (e.g. road data), a set of information which had changed since the last push is exchanged to align the information status on the client system. These messages may include single updated element or all updated elements depending on update method chosen.

The supplier takes the initiative to push the data under the following conditions:

— **On Occurrence push**: as soon as information is updated at the supplier systems, this condition triggers the supplier to send push data to the client for updating the client system as soon as possible after this update.

— **Periodic push**: at a predefined time interval the supplier starts an exchange based on client and supplier agreement (subscription contract).

— A **snapshot synchronisation** with the whole currently available content snapshot in case a snapshot alignment is requested by client.

— **Response to a snapshot synchronisation request**: one snapshot alignment may also be transmitted to the client for internal system needs/maintenance/debug, it can be requested by the client via any return messages, i.e. wrapped in returned exchange information**.**

8.2.2 Relevant exchange information from exchange data model

8.2.2.1 Overview

Basic exchange data model has been provided to allow the implementation to deliver more payload contents on the same message and further information to allow managing extra features not required by the Simple Push exchange.

For interoperability convenience the exchange data model wrapping shall be managed in this exchange.

A payload shall be pushed to a client using basic exchange data model as reported in the previous figure.

An ExchangeInformation shall be returned from **putData** to convey information about exchange operation and connection status.

8.2.2.2 Exchange information

Some non-mandatory information which should be managed in the exchange information for easy application development are fully described in basic exchange data model:

— Supplier Identification (String)

  — Requirement: Supplier identification.

— Client Identification (String)

  — Requirement: Client identification.

— Exchange Information (provided both by the client and the supplier) wraps exchange and exchange status information "Success", "Fail", "Close Session Request", "Snapshot Synchronisation Request", “SessionId”

  — Requirement: Session management, Link monitoring.

8.2.2.3 Payload information

— Generation timestamp information

  — Requirement: timely, reliable Information, session management.

8.2.3 List of exchanged messages

Different messages or supplier / client interactions (invoked method) are exchanged in Simple Push which are needed to manage synchronisation, payload exchange, link monitoring. These are formally contained in pushed messages to a client from a supplier or in return messages from client to supplier.

Table 10 — List of messages types and detailed content

Interaction message	Direction supplier client	Designation	Description	Exchanged information	Optional
Payload Push	Direct	putData	Push delivery of payload, which has to be delivered from supplier to client. Exchange information such as client and supplier identification and exchange status may be provided to easy controls.	Payload + Exchange Information (MessageContainer)	N
Snapshot Payload Push	Direct	putSnapshotData	Push delivery of current available payload, i.e. snapshot after a first initialised session in case of first connection or after an explicit snapshot realignment request from the client. Exchange information such as client and supplier identification and exchange status may be provided to easy controls.	Snapshot Payload + Exchange Information (MessageContainer)	Y
Keep Alive	Direct	keepAlive	Test exchange link and confirm session validity when no payload push update is needed, exchange information such as client and supplier identification and exchange status may be provided to easy controls and supplier identification.	Exchange Information	N
Exchange Information Return	Return	Exchange Information	Exchange information is returned from client to supplier to provide return status i.e. Success, Fail, Snapshot Synchronisation Request and to easy controls such as supplier and client identification.	Exchange Information	N

8.3 Link monitoring and error management

The supplier initiates the communication and is made aware of the client status based on the client return response to the supplier.

The following diagram (Figure 15) describes the client status as monitored and managed by the supplier.

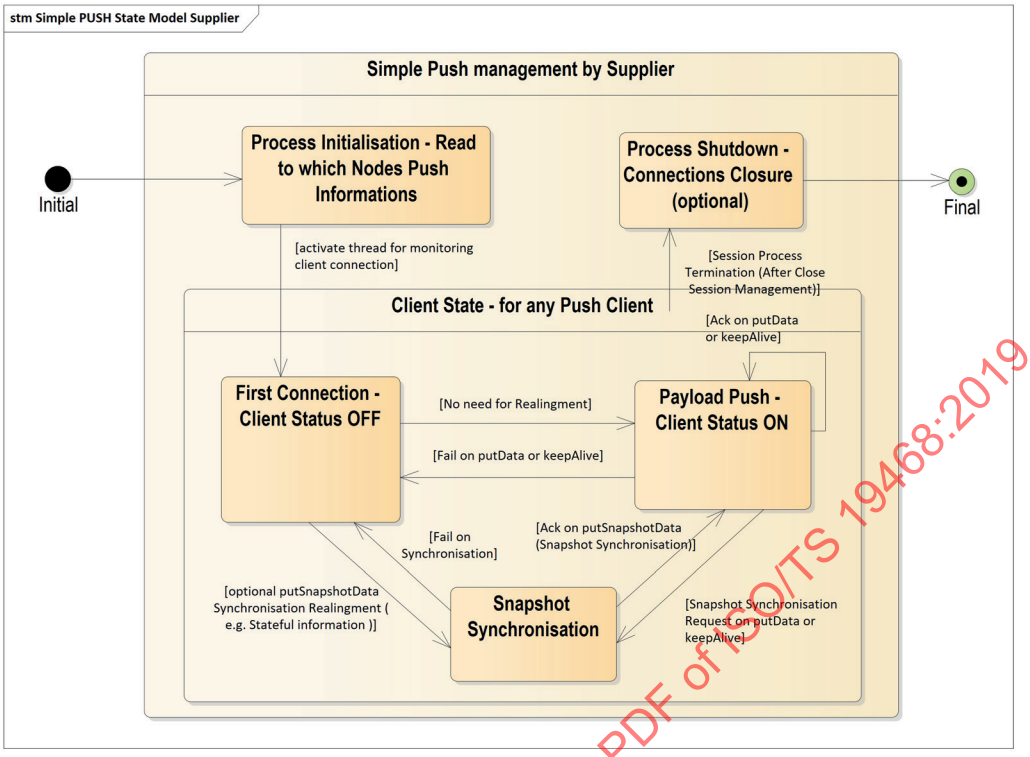

Figure 15 — Supplier-side Simple Push state diagram

The following diagram (Figure 16) describes the supplier status as monitored and managed by the client.

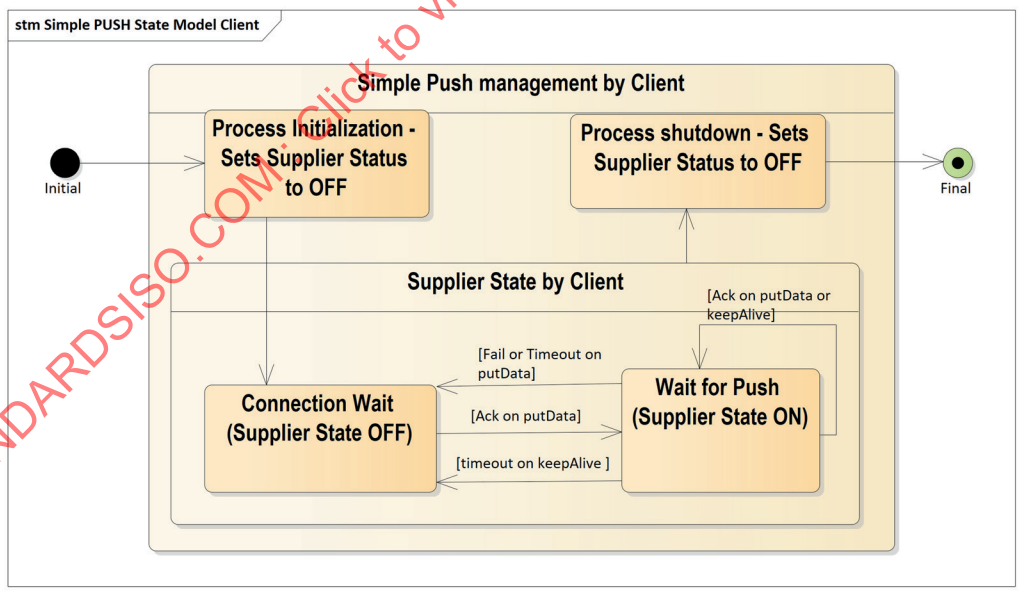

Figure 16 — Client-side Simple Push state diagram

Special management in initialisation and termination of Push process is to consider at the application level in supplier and client systems.

8.4 Features implementation description

8.4.1 Overview

This subclause provides a description and the corresponding specification for each feature identified in the context diagram, according to the Simple Push data exchange architecture. The following features are specified:

— Subscription contract;

— Subscription (also known as session);

— Information management;

— Data delivery;

— Communication/protocol.

8.4.2 Subscription contract

8.4.2.1 Contract

Managed offline, not automated. It assumes information for controls to be implemented in client to assess the identity of supplier and authenticate the supplier request in messages exchange.

8.4.2.2 Catalogue

Managed offline, not automated.

8.4.3 Session

8.4.3.1 Session life cycle

No session is managed for the current EP+FEP.

8.4.3.2 Link monitoring

Link monitoring is done by Payload Push and KeepAlive. When data is available a payload Push is exchanged which also informs client and supplier about the session and systems status: Push received from supplier on client side and return of push on payload push on supplier side guarantee the network is available and the systems are up and running.

When no data is available at supplier and a time-out has expired a KeepAlive message is exchanged to check network and system availability.

In case payload push or KeepAlive fails or times out, the supplier assumes the client is offline and keeps trace of its status for any management purposes at the supplier system side (for delivery retry mechanism is to be described at PSM level defining a logical push mapping iterated for a maximum number of times) (Figure 17).

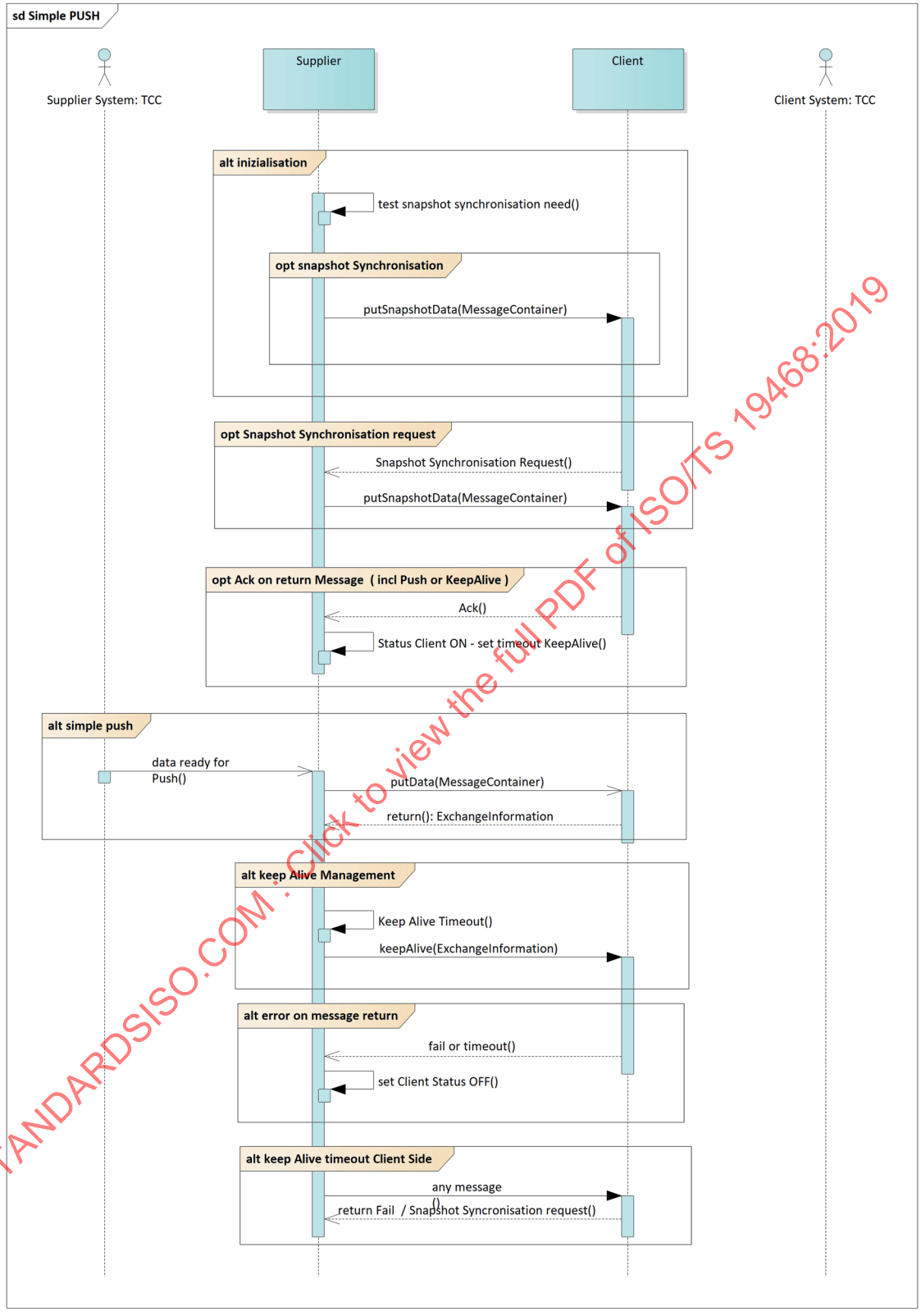

Figure 17 — Simple Push sequence diagram for link monitoring and data delivery

8.4.4 Information management

8.4.4.1 Operating modes

Available operating mode for Simple Push is “Periodic”, or “On Occurrence” (i.e. condition is triggered based on supplier) push based on supplier-side conditions.

Description of operating modes is done at a general PIM level; no extra details are needed in this subclause for PIM/Exchange Pattern / FEP.

A Payload Push condition is triggered based on the agreed operating mode defined at subscription among client and supplier.

8.4.4.2 Update methods

Available updated methods are: Snapshot, Single Element Update, All element update.

All updates available are conveyed in a payload publication push message.

Description of update methods is done at PIM level; no extra details are needed in this subclause for PIM/exchange pattern / FEP.

8.4.4.3 Life cycle management

Description of life cycle management is done at PIM level.

Life cycle management to exchange data between client and supplier is embedded with the operating mode and update method chosen at subscription contract.

Push delivery method allows conveying information from supplier to client as two different pieces of information.

Sampled data may be conveyed as Periodic or On Occurrence push of a snapshot payload containing all current active data or last collected data.

A Single/All Element updated push can be done for any operating mode as well with On Occurrence or Periodic Push.

8.4.5 Data delivery

8.4.5.1 Data delivery

Based on sequence diagram described, the supplier after initialisation starts sending push information to a client: in case of stateful information delivery and based on the possibly agreed conditions of the subscription contract, e.g. it manages a snapshot push whenever it has no historical information about client status, deriving it is the first connection ever and a Snapshot Push is needed to align the client system when it is not the first time the supplier sends to the client normal payload push data, but the client for internal system needs can also require for snapshot push data by its return status.

After initialisation ready data condition at the supplier system side triggers a payload push delivery.

A periodic push condition is also possible based on contract agreement between supplier and client.

See sequence diagram at link monitoring life cycle for all messaging details.

8.4.5.2 Data request

Not implemented in this pattern.

8.4.5.3 Large datasets handling

Not described at PIM level, may be defined at PSM level.

8.4.5.4 Synchronisation

Snapshot synchronisation and delta synchronisation are available in Simple Push.

Snapshot synchronisation is optionally managed at first connection with a client or under client request. In all other cases a Simple Push is delivered based on supplier site data available and conditions.

8.4.6 Self-Description

Handshake not available.

8.4.7 Communication

Communication features are implemented at PSM level, they are relevant to the specific Platform chosen on which the Exchange pattern will be implemented (e.g. http/XML, Web Services SOAP, REST).

8.4.8 Optimisation issues

8.4.8.1 Bandwidth and processing savings

Payload timestamp information is available for client-side processing optimisation made at application level.

Push message may be generated for all client reducing processing resources at supplier side.

No extra optimisation issues are considered in this EP+FEP.

8.4.8.2 Huge dataset handling

Not managed in this EP+FEP.

9 Stateful Push

9.1 Overview

Stateful Push exchange pattern / FEP at PIM level is based on information messages sent or pushed by a supplier to a client. This exchange pattern can be implemented in several platforms: some examples are pushing generated XML content by http/post, or client providing a SOAP Web service method “push” by which data can be provided to a client by a supplier.

This “Stateful Push” adds extra features to the basic Push exchange pattern in order to manage "Session life cycle" and "Link monitoring", as well as synchronisation/realignment in case of communication cut or system maintenance. This mechanism will be fully explained at the PIM level in the following subclauses.

To describe the exchange pattern/FEP at PIM level all the features are described in a general abstract format, independently from the specific technologic platform in which this model will be implemented (e.g. http/get XML or Web services).

Table 11 — Selection of features for Stateful Push

Features area	Feature	Stateful Push available
Subscription contract	Contract	N

Table 11 *(continued)*

Features area	Feature	Stateful Push available
	Catalogue	N
Session	Session life cycle	Y
	Link monitoring	Y
Information management	Operating modes	Periodic / On Occurrence based on supplier triggering conditions
	Update methods	Snapshot, Single Element Update, All Element Update
	Life cycle management	Y
Data delivery	Data delivery	Y
	Data request	Snapshot realignment
	Large datasets handling	N *(Can be implemented via exchange information defined and managed at PSM level via iterative implementation of partial delivery push)*
	Synchronisation	Y
Self-description	Handshake	N
Communication	Security	At PSM level
	Compression	At PSM level
	Communication	At PSM level

9.2 Exchange pattern messages definition

9.2.1 Overview

Information delivery business scenario description and definition state that data exchange is needed to align the information kept by the supplier system into the client system; for this purpose, an exchange system is used which provides tools enabling messages generation and their transfer between a supplier and a client (Figure 18).

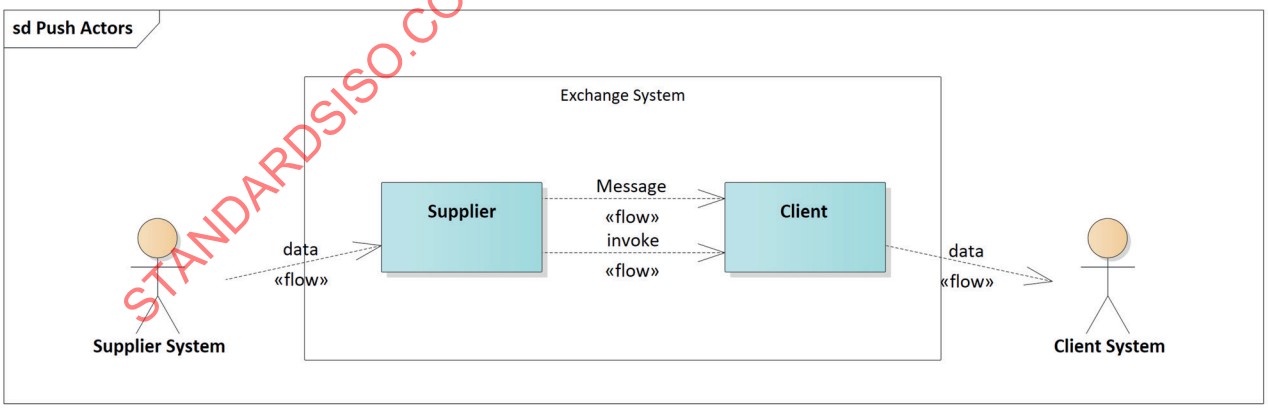

Figure 18 — Stateful Push exchange actors

The "Stateful Pull" exchange pattern is described in the following subclauses.

9.2.2 Basic exchange pattern

The client provides a mechanism to receive data from an action taken at the supplier site invoking specific resources / methods offered by the client.

Therefore, the supplier logically "pushes" messages to the client. The client shall acknowledge what is received by a return exchange information to the supplier. This exchange information return message is available to bring information back from the client to the supplier, such as SessionId, failure, success, snapshot synchronisation request. Return message information is logically described in this PIM, while implementation will be defined at PSM level (Figure 19.)

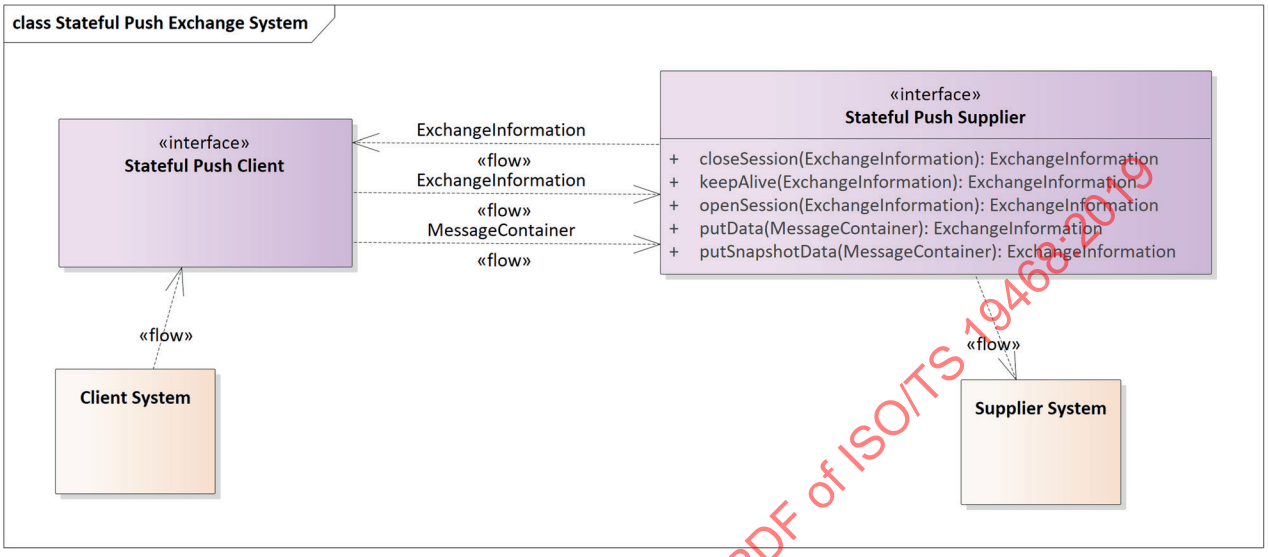

Figure 19 — Stateful Push exchange subsystems, interface interactions and methods

Exchanged data can either be the currently available data, also called "snapshot" of information, or, after a first transfer of currently valid data, i.e. a snapshot, a subset of information which had changed since the last push. This subset is then used to align the information status on the client system. These messages may include either the only updated elements in a set or the whole sets where one or several elements have been updated according to the chosen update method.

The supplier takes the initiative to push the data under the following conditions:

— **First session initialisation:** a global snapshot alignment is needed to convey all currently valid data at the first connection of exchange.

— **Session initialisation**: after a session had been created the client shall be aligned with an incremental delta synchronisation when a partial update feature is enabled, delivering all updated content since last exchange, or with a global synchronisation with all the current active content in case the snapshot alignment feature is enabled (this may also depend on specific payload depending on the agreement between client and supplier or contract).

— **Global synchronisation request**: a global snapshot alignment can also be transmitted to the client for internal system needs, maintenance or debug; it may be requested via any return message.

— **On occurrence push**: as soon as an information is updated at the supplier systems, this condition triggers the exchange supplier to manage an alignment to the exchange client to update the client system as soon as possible after this update.

— **Periodic push**: at predefined time interval the supplier starts an exchange based on client and supplier agreement (subscription contract).

9.2.3 Relevant exchange information from exchange data model

Basic exchange data model has been provided to allow an implementation that delivers more payload contents in the same message and further information to allows managing extra features which are not required by the basic Snapshot Push exchange.

In order to ensure interoperability, the exchange data model wrapping shall be used in this exchange pattern.

A container shall be pushed to client using basic exchange data model as stated in the previous figure.

An ExchangeInformation object shall be returned from **putData** to convey information about exchange operation and connection status.

9.2.3.1 Exchange information

Some non-mandatory information that should be managed in the exchange information to make application development easier are fully described in basic exchange data model:

— Supplier Identification (String)

  — Requirement: Supplier identification.

— Client Identification (String)

  — Requirement: Client identification.

— Exchange Information (provided both by the client and the supplier) wraps exchange and exchange status information "Success", "Fail", "Close Session Request", "Snapshot Synchronisation Request", SessionId

  — Requirement: Session management, Link monitoring.

9.2.3.2 Payload information

— Generation timestamp information

  — Requirement: timely, reliable Information, session management.

9.2.4 List of exchanged messages

Different messages or supplier/client interactions are exchanged in the Stateful Push which are needed to manage session, synchronisation, payload exchange, link monitoring. They are formally contained in pushed messages to the client from the supplier or in return messages from the client to the supplier.

Table 12 — List of message types and detailed content

Interaction Message	Direction Supplier Client	Designation	Description	Exchanged information	Optional
Open Session	Direct	openSession	Supplier initialise a push delivery session.	Exchange Information	N
Payload Push	Direct	putData	Push delivery of payload, which has not been yet delivered from supplier to client. It shall contain in Exchange information the Session ID, previously obtained with OpenSession, referring which session it is pushing data.	Payload + Exchange Information (MessageContainer)	N

Table 12 *(continued)*

Interaction Message	Direction Supplier Client	Designation	Description	Exchanged information	Optional
Snapshot Payload Push	Direct	putSnapshot-data	Push delivery of current available payload, i.e. snapshot after a first initialised session in caso of first connection or after an explicit snapshot realignment request from the client. It shall contain in exchange information the Session ID, previously obtained with OpenSession, referring which session it is pushing data.	Snapshot Payload + Exchange Information (MessageContainer)	N
Keep Alive	Direct	keepAlive	Test exchange link and confirm session validity when no payload push update is needed. It shall deliver the Session ID of a previously opened session, wrapped in Exchange Information.	Exchange Information	N
Close Session	Direct	closeSession	Message to gracefully close a delivery session, initiated by the supplier	Exchange Information	N
Exchange Information Return	Return	D2Exchange Infomation	Exchange information is returned from client to supplier to provide return status i.e. Success, Fail, Snapshot Synchronisation Request and to easy controls such as supplier and client identification.	Exchange Information	N

9.3 Session status management

The supplier initiates the communication and can be aware of the client status based on the client return response to the supplier.

The following diagram (Figure 20) describes the client status as monitored and managed by the supplier.

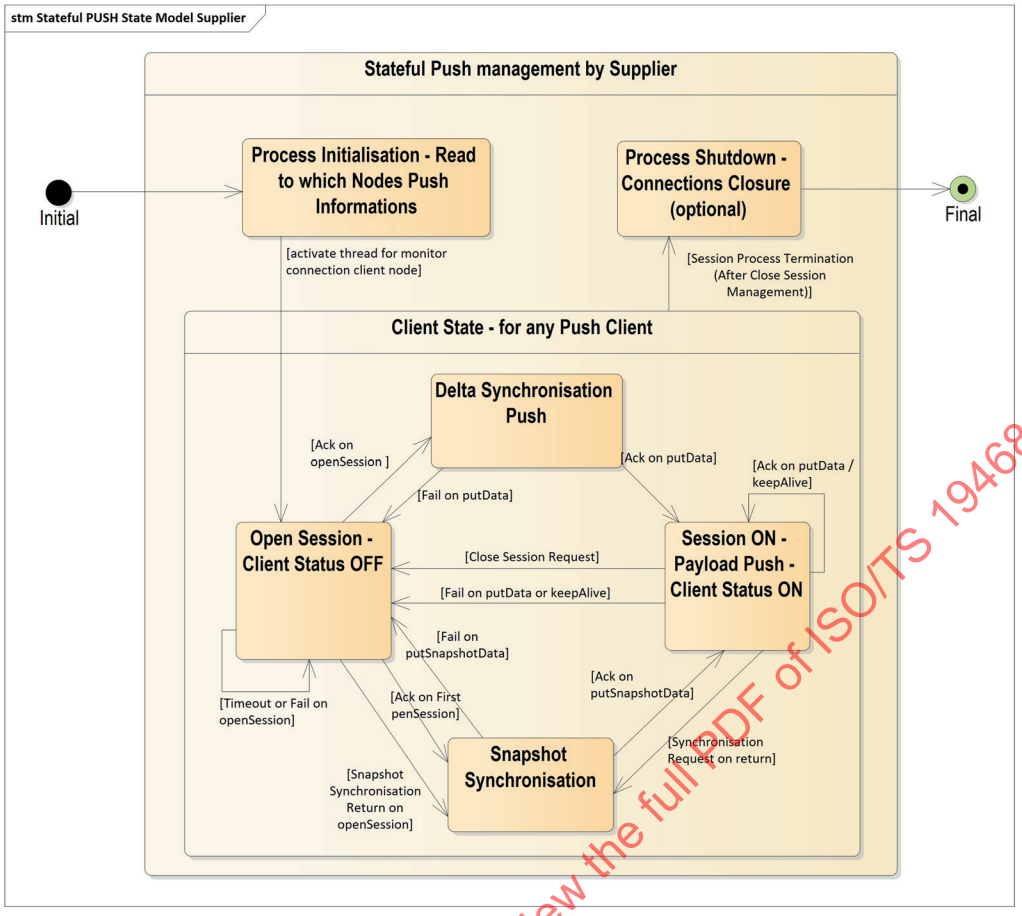

Figure 20 — Supplier-side Stateful Push state diagram

The following diagram (Figure 21) describes the supplier status as monitored and managed by the client.

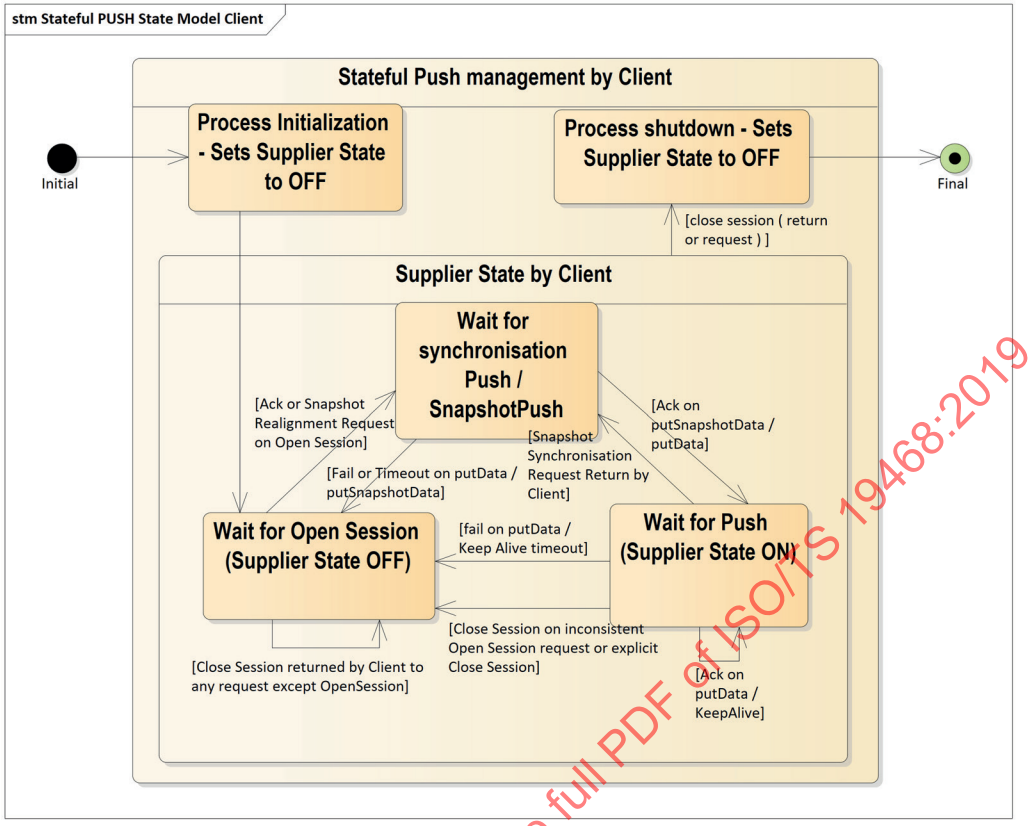

Figure 21 — Client-side Stateful Push state diagram

Specific management in the initialisation and termination of a push process should be considered at the application level in the supplier and client systems.

9.4 Features implementation description

9.4.1 Overview

This subclause provides a description and the corresponding specification for each feature identified in the context diagram, according to the Publish-Subscribe data exchange architecture. The following features are specified:

— Subscription contract;

— Subscription (also known as session);

— Information management;

— Data delivery;

— Communication/protocol.

The corresponding classes, attributes and relationships described in the class diagrams included in the next subclauses are described in C.4.

9.4.2 Subscription contract

9.4.2.1 Contract

Managed offline, not automated. It assumes information for controlling to be implemented in a client to assess the identity of supplier and authenticate the supplier request in messages exchange.

9.4.2.2 Catalogue

Managed offline, not automated.

9.4.3 Session

9.4.3.1 Session life cycle

After the session status management diagrams, the following sequence diagram illustrates the exchanged message and the expected return and behaviour.

When a session needs to be initiated, an “Open Session” is tried in loop until it succeeds.

A failure when opening a session includes cases of a client who has not subscribed or is not authorised. Checking this can be ensured at the PSM level (e.g. this could include VPN setting or IP firewall or signatures handling), logical information may be included in exchange data to be managed in subscription check at the client side.

When a session is "ON" the supplier pushes available payload data to the client in case one of the 2 conditions is fulfilled:

— payload available for On Occurrence operating mode, or

— payload delivery time-out for Periodic push operating mode.

In case no payload is available a" Keep Alive" message is used to check session status for the supplier and the client.

When no "Keep Alive" message or no payload is received, after a "Keep Alive" time-out the client invalidates the session on its side and returns a "Close Session" message to prevent any attempt of push from the supplier.

If any push or keep alive fails the supplier invalidates the session and starts a new loop to open session.

Realignment messages are managed when a session is opened, a global synchronisation request from the client is returned in opening session when needed. Further any global synchronisation may be asked by a client in any push return as well but this does not close the session.

Any message and return in the sequence diagram (Figure 22) will be mapped in PSM definition to real platform available implementation such as a web service "Service request and return" or any other available mechanism in a specific platform.

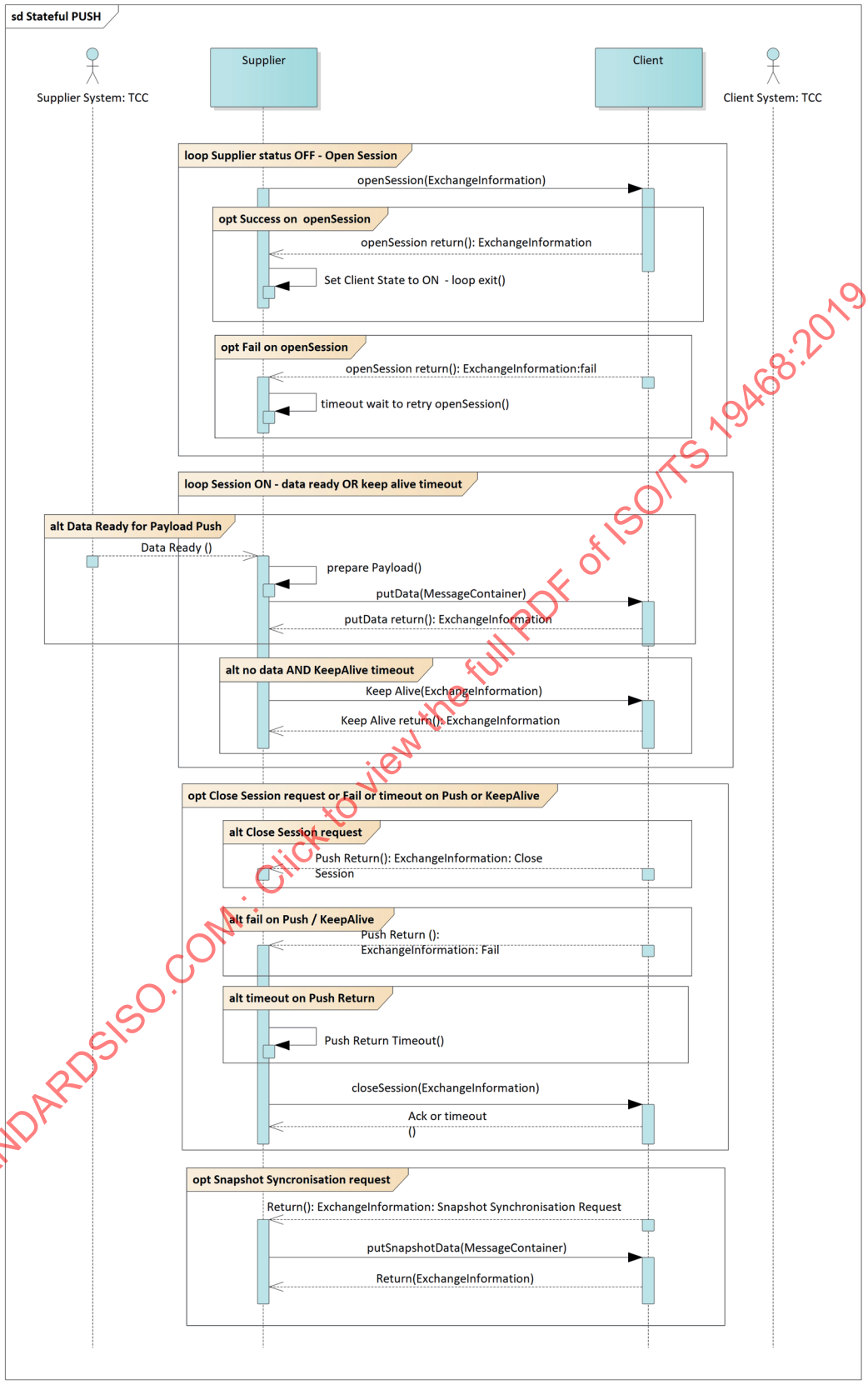

Figure 22 — Stateful Push sequence diagram for session life cycle and data delivery

9.4.3.2 Link monitoring

"Keep Alive" is based as described in the previous session life cycle.

When data is available a payload push is exchanged which informs the client and the supplier about the session and systems status: The push is received from the supplier on the client side and the return of Push on payload push on the supplier side guarantees the network is available and the systems are up and running.

When no data is available and a time-out has expired a "Keep Alive" message is exchanged to check the network and system availability.

In case a payload push or a "Keep Alive" fails the session shall be invalidated (the retry mechanism is to describe at the PSM level introducing a logical push mapping iterated for a maximum number of attempts).

9.4.4 Information management

9.4.4.1 Operating modes

The available operating modes for supplier Push are either periodic, or condition-triggered (based on the supplier-side conditions and the interchange agreement at the subscription).

The general description of the operating modes is done at the PIM level, no extra details are needed in this subclause for PIM/exchange pattern / FEP.

A payload Push is triggered based on the agreed operating mode defined at the subscription between the client and the supplier.

9.4.4.2 Update methods

Available updated methods are: Snapshot, Single element update, All elements update.

All updates available are conveyed in a payload publication push message.

The general description of the update methods is done at the PIM level, no extra details are needed in this subclause for PIM/exchange pattern / FEP.

9.4.4.3 Life cycle management

The description of life cycle management is done at the PIM level.

The life cycle management for exchanging data among a client and a supplier is embedded in the operating mode and the update method which have been chosen in the subscription contract.

The push delivery method allows conveying information from a supplier to a client as two different sets of information.

Sampled data can be conveyed (pushed) as “Periodic” or “On occurrence” of a global payload containing all currently active data or last collected data.

A "Single element" or "All elements" update push can be done for any operating mode, i.e. with "On occurrence" or "Periodic" push.

9.4.5 Data delivery

9.4.5.1 Data delivery

Session life cycle online section allows sending data when available at the supplier system by triggering a data ready condition.

A periodic push condition is also possible based on contract agreement among supplier and client.

See sequence diagram at the session life cycle for messaging details.

9.4.5.2 Data request

Not implemented in this pattern.

9.4.5.3 Large datasets handling

Not described at the PIM level, may be defined at the PSM level.

9.4.5.4 Synchronisation

Global synchronisation and delta synchronisation are available in Stateful Push.

The global synchronisation is managed at the first session with a client or under a client request.

The delta synchronisation is managed once a session had been created and closed due to a network error or any other condition, so that not-delivered payload data is stored at the supplier side and delivered to the client as soon as a session is established again.

9.4.6 Self-description

The handshake is not available.

9.4.7 Communication

The communication features are implemented at the PSM level; they are relevant to a specific platform chosen on which the exchange pattern will be implemented (e.g. http/XML, Web services with SOAP, REST, etc.).

10 Publish Subscribe

10.1 Exchange architecture

10.1.1 Pattern description

The described architecture is based on a messaging pattern which allows publishing information without care for delivery according to the identified requirements. This pattern allows suppliers of messages not sending these messages to specific clients directly; in contrary a dedicated dispatcher layer is used for notifying clients. It has the following characteristics:

— It allows for the asynchronous detection of real-time events (defined as new or updated information or data availability at the supplier system) by a producer service whose role is to generate and send notifications to one or more interested client systems.

— Published messages are characterised into categories by the supplier.

— The supplier does not have any knowledge about how many subscribers (clients) there can be. Subscribers express interest in one or more topics and only receive messages that are of interest, without any particular knowledge of the supplier.

NOTE 1 This decoupling of suppliers and subscribers allows for greater scalability and a more dynamic network topology.

The following Figure 23 depicts the several roles involved into a data exchange process based on a *Publish-Subscribe* paradigm and the main interactions between them.

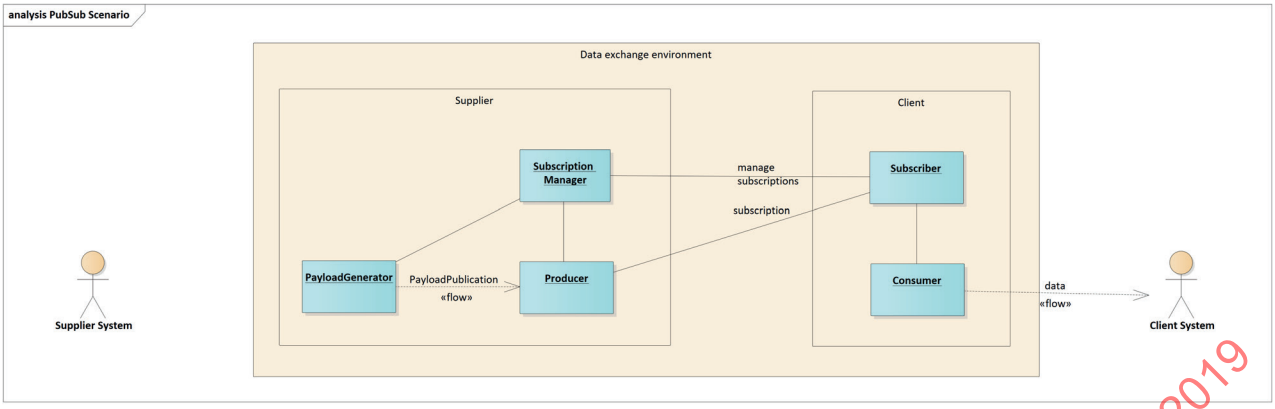

Figure 23 — Roles involved in data exchange

Besides providing a specification for the communications between supplier and client, this document also includes description of communications that are internal to a supplier or internal to a client. It is not required that they directly map to identically named and identically factored features in an implementation, but they are included in the PIM as a specification of the semantics of the data exchange communication.

NOTE 2 The adopted messaging pattern is derived from the Publish Subscribe pattern by extending it. Some roles have been renamed to avoid confusion when they have been changed regarding the original pattern.

10.1.2 The supplier

The supplier's role is broken down into the following items with the corresponding requirements:

— Payload generator

  — Shall generate notifications in accordance with the events that occur within the system it controls. This role is seen as being the publisher.

  — In data exchange such a notification is referred to a content message which is named ***PayloadPublication***.

    NOTE 1 Although represented as a separated role, this document does not mandate that such a role actually exists. It is just a logical separation that makes sense in most implementations.

— Producer

  — Shall execute the physical delivery of the information to clients.

  — Shall have the actual knowledge about any active clients waiting for the information, i.e. clients having a valid subscription for the data. For that, it shall interact with the "SubscriptionManager" role.

  — Shall create the final notification message to the client which shall contain:

    — Exchange specific information (with the "Exchange" class),

    — Payload content (through zero, one or more "PayloadPublication" messages),

  — The notification message between producer and client is named "Notification".

— Subscription Manager

  — Key element needed for this process, the "Subscription" class shall contain all the information a supplier system needs to know to allow the actual data exchange process to take place. It is

detailed later on. Nevertheless, once a subscription exists the subscription manager shall be the entity which provides means for the client to manage it, either by updating it or deleting it.

NOTE 2 New subscriptions are not created by the subscription manager but by the producer itself. The client gains knowledge of the subscription manager only after a successful subscription is created

10.1.3 Client

The supplier's role is broken down into the following items with the corresponding requirements:

— Subscriber

  — Shall request and manage subscriptions

— Consumer

  — Receives the information that arrives from the supplier's side.

10.2 Feature description

10.2.1 Overview

This subclause provides a description and the corresponding specification for each feature identified in the context diagram, according to the Publish-Subscribe data exchange architecture. The following features are specified:

— Subscription contract;

— Subscription (also known as session);

— Information management;

— Data delivery;

— Communication/protocol.

The corresponding classes, attributes and relationships described in the class diagrams included in the next subclauses are described in the normative Annex C.

10.2.2 Subscription contract

10.2.2.1 Description

A subscription contract is a function element in data exchange. When two entities agree to exchange data exchange information, they shall agree what information is exchanged between the systems and under what conditions, among other technical details. A Subscription Contract can be seen as a contract between a supplier and a client described in technical terms.

10.2.2.2 Features

10.2.2.2.1 Overall introduction

The subscription contract should have the following features:

— Contract: a model that can be used to the support of the information of a subscription contract;

— Contract life cycle: features for managing the life cycle of the subscription contract:

  — Create subscription contract;

  — Manage subscription contract;

— Terminate subscription contract;

— Catalogue: a model for handling catalogues;

10.2.2.2.2 Contract

The terms of the contract shall include at least the following details:

— Type of data (data that the supplier agrees to send to the client).

EXAMPLE 1 A supplier can agree with the client to publish only situation publications or from the situation publication only operator actions or even from the operator action only specific attributes, and at the same time provide this information using only point locations.

NOTE In order to define this information to be published, supplier and client can define a *Profile*. The profile consists of a data exchange sub-model that has been created out of the data exchange model by selecting those optional elements in data exchange that are useful with the type of information to be published.

— Operating modes (set of rules and conditions that regulate the physical transmission of data between the supplier and client). Different operating modes may be used according to the types of data that the supplier is delivering.

EXAMPLE 2 The operating mode **"onOccurrence"** can be used when publishing situations or events and the operating mode **"periodic"** for publishing measured data.

— Update methods (the way to deliver updated information globally or the modified elements).

— Period of validity (period of time in which the contract remains valid. After this time, the agreement may be extended or be considered as terminated).

— Technical details: server endpoint, security parameters, use of compression, update time, etc.

10.2.2.2.3 Contract life cycle

The following features manage the life cycle of the subscription contract:

— Create subscription contract (handles the creation of a subscription contract).

— Manage subscription contract (handles changes to an existing subscription contract).

— Terminate subscription contract (handles termination of an existing subscription contract).

The interface of these features is generally not implemented as a machine to machine exchange but in a configuration database.

10.2.2.2.4 Catalogue

The catalogue of a data exchange system consists of a list of available services.

The interface is generally not implemented as a machine to machine exchange but in a configuration database.

10.2.2.3 Data model

10.2.2.3.1 Presentation

Figure 24 describes the "Subscription contract and Catalogue" data model with the corresponding classes, attributes and relationships.

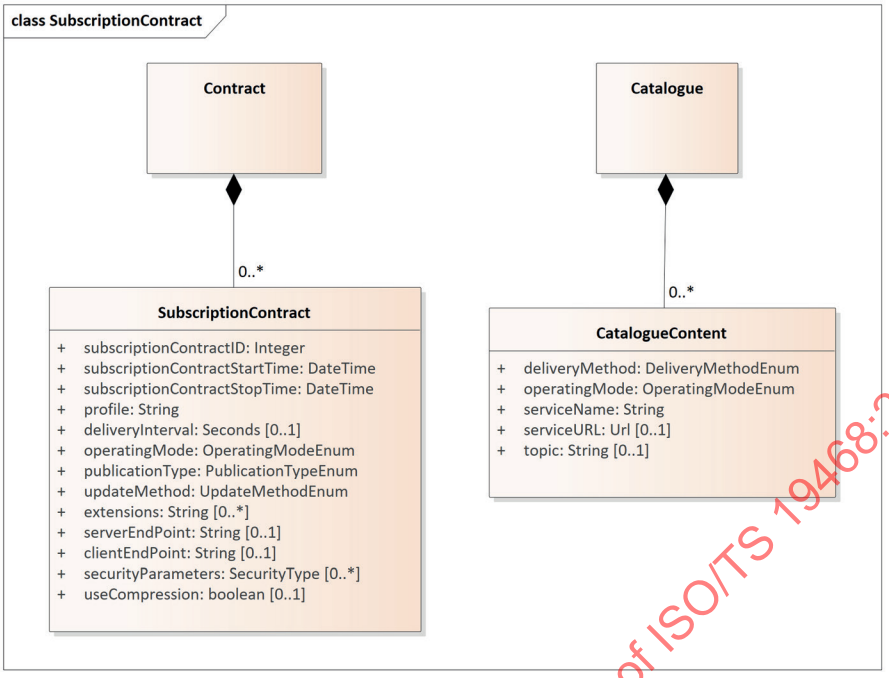

Figure 24 — Subscription contract data model

10.2.2.3.2 Semantics

A contract (class "Contract") is composed of zero, one or several subscription contract (class "SubscriptionContract"). Among the attributes defining it some of them are mandatory, e.g. the publication type, the contract start time, the operating mode and the update method.

A catalogue (class "Catalogue") is composed of zero, one or several catalogue contents (class "CatalogueContent"). Among the attributes defining it some of them are mandatory, e.g. the service name, the operating mode and the update method.

10.2.3 Subscription

10.2.3.1 Description

A Subscription is a continuous period of exchange between one stated supplier and one stated client, which are connected and ensure a fully reliable, timely and consistent delivery of all the information from the supplier to the client.

10.2.3.2 Features

10.2.3.2.1 Overall introduction

The subscription function elements shall have the following features:

— Subscription life cycle: features for managing the life cycle of a logical session:

  — Create subscription (Subscribe);

  — Manage subscription;

  — Terminate or destroy subscription;

— Link Monitoring: features for link monitoring and control;

— Subscription expired: management of an expired subscription;

— Error handling: how error occurrences are managed in the exchange process.

10.2.3.2.2 Subscription life cycle

The following sequence diagram in Figure 25 describes how to create a subscription between subscriber and producer:

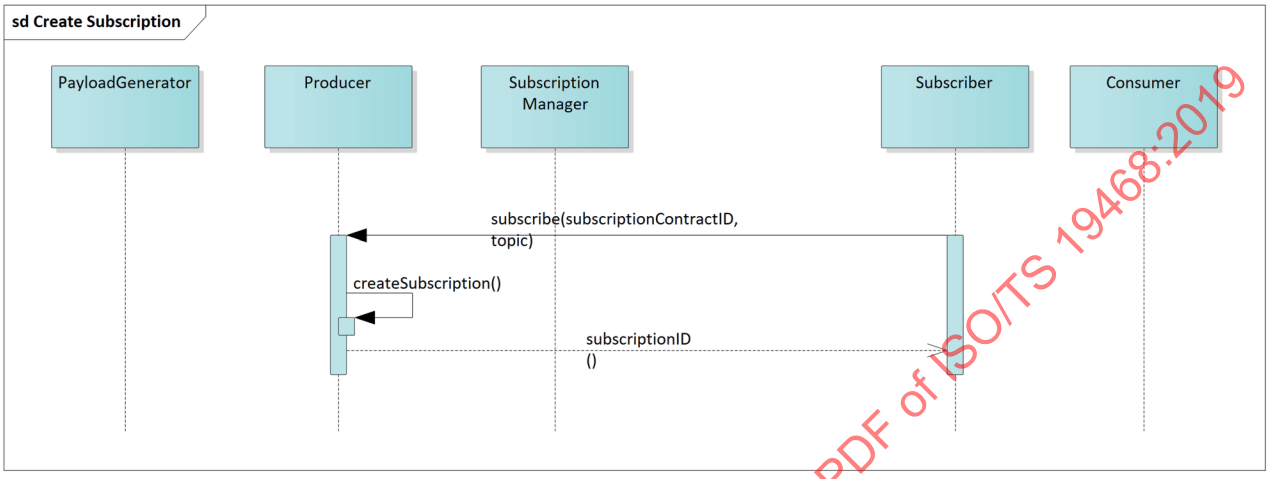

Figure 25 — CreateSubscription sequence diagram

Semantics: The client shall create a subscription in order to receive the information available at the supplier side.

The following sequence diagram in Figure 26 describes how a client manages after its creation.

NOTE In the following figures representing sequence diagrams the ":DataResponse" messages only correspond to acknowledgement (or rejection).

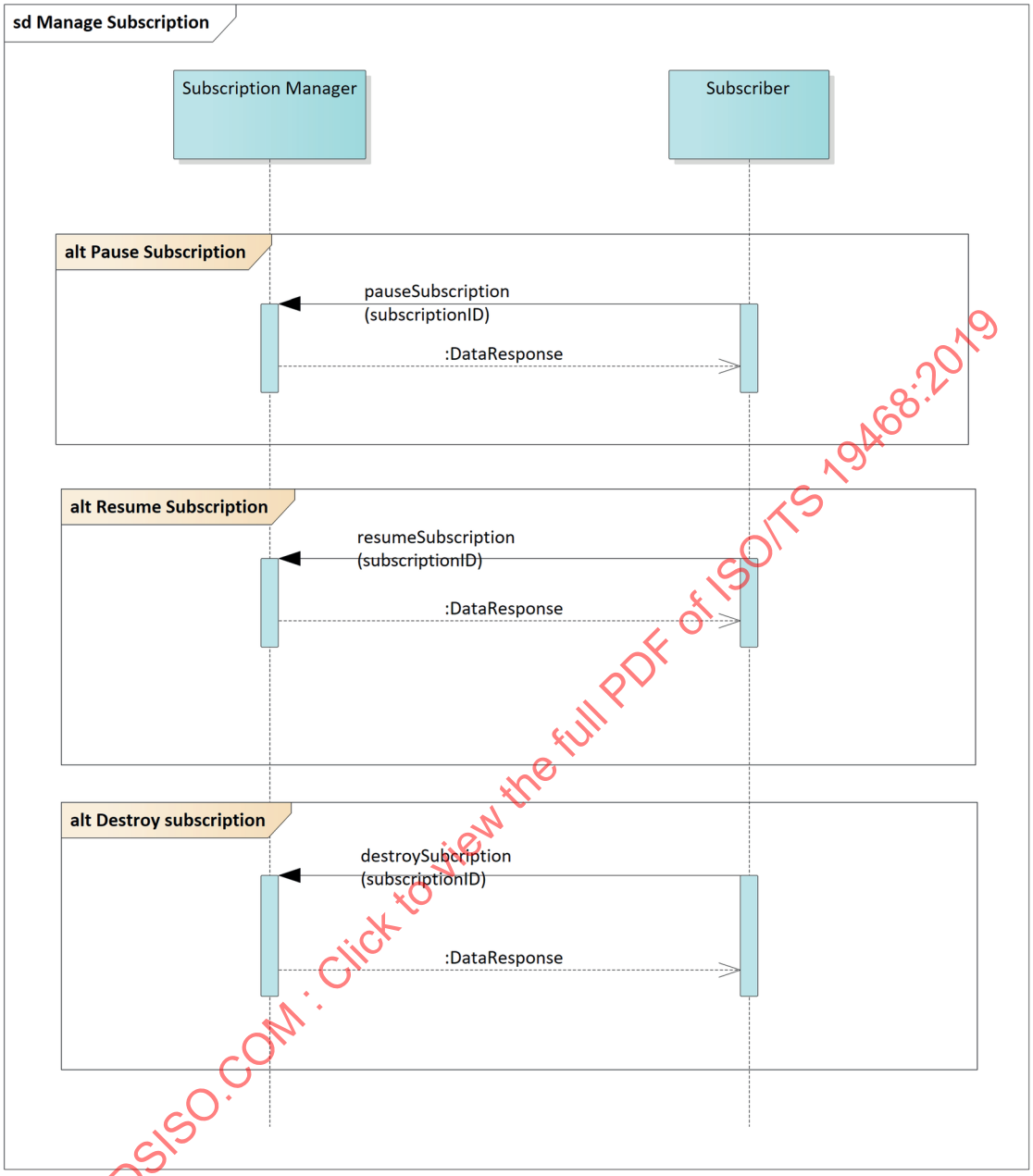

Figure 26 — "ManageSubscription" sequence diagram

Semantics: The subscriber should choose to ***pause***, ***resume*** or ***destroy*** a subscription by invoking associated methods on the Subscription Manager.

The subscription can be terminated on the following cases:

— If the client wants to terminate the subscription;

— When the client does not receive "keep alive" information and needs to terminate a data exchange;

— When an error occurs (in this case both client and supplier should ask for terminating the subscription).

10.2.3.2.3 Link monitoring

Link monitoring is a feature to assure the Subscription between the supplier and client is up and running. The "KeepAlive" message is a minimal exchange message that shall be sent to monitor the status of the link between supplier and client. The other messages defined in the blocks tagged as "opt" are recommended and should be implemented. The following sequence diagram in Figure 27 describes how a client manages a subscription after its creation:

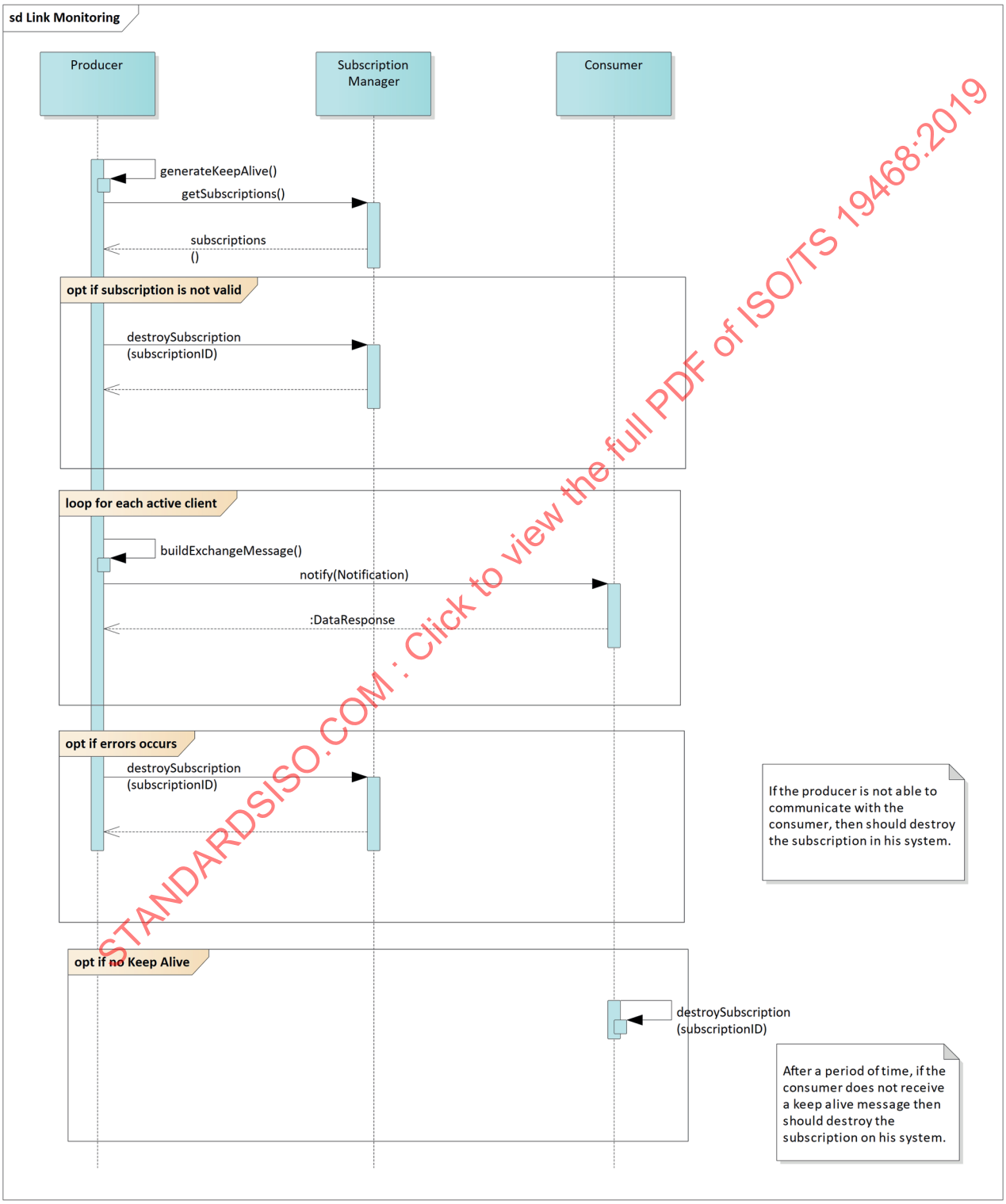

Figure 27 — "LinkMonitoring" sequence diagram

Semantics: The producer shall send a "KeepAlive" message regularly and check for active subscriptions.

NOTE The management of errors in this sequence diagram corresponds to cases of unrecoverable errors, which implies to have made several tries beforehand.

10.2.3.2.4 Subscription expired

In case of lost "KeepAlive" messages or unmanageable errors in data delivery for a given time, the connection is considered as lost and a new subscription should be created again when connection is available. Data recovery can be needed in case the subscription is restarted after such a situation.

10.2.3.2.5 Error handling

During the lifetime of a subscription there are cases where errors occur. In such cases, the producer or subscriber should retry a new connection in order to return to the normal scenario:

— **Communications errors:** Network or equipment problems, where communication fails between the supplier and the client;

— **Bad response:** A bad response is related to an unexpected response in the communication of the supplier and the client;

EXAMPLE 1 Case of http response error codes, unexpected information, XML mismatch or service unavailable.

— **Data exchange message errors:** When all the exchanged messages are not consistent.

EXAMPLE 2 In case sent information on a specific update method does not match with the payload, the answer is an error response.

The following sequence diagram in Figure 28 describes how errors are handled during a subscription:

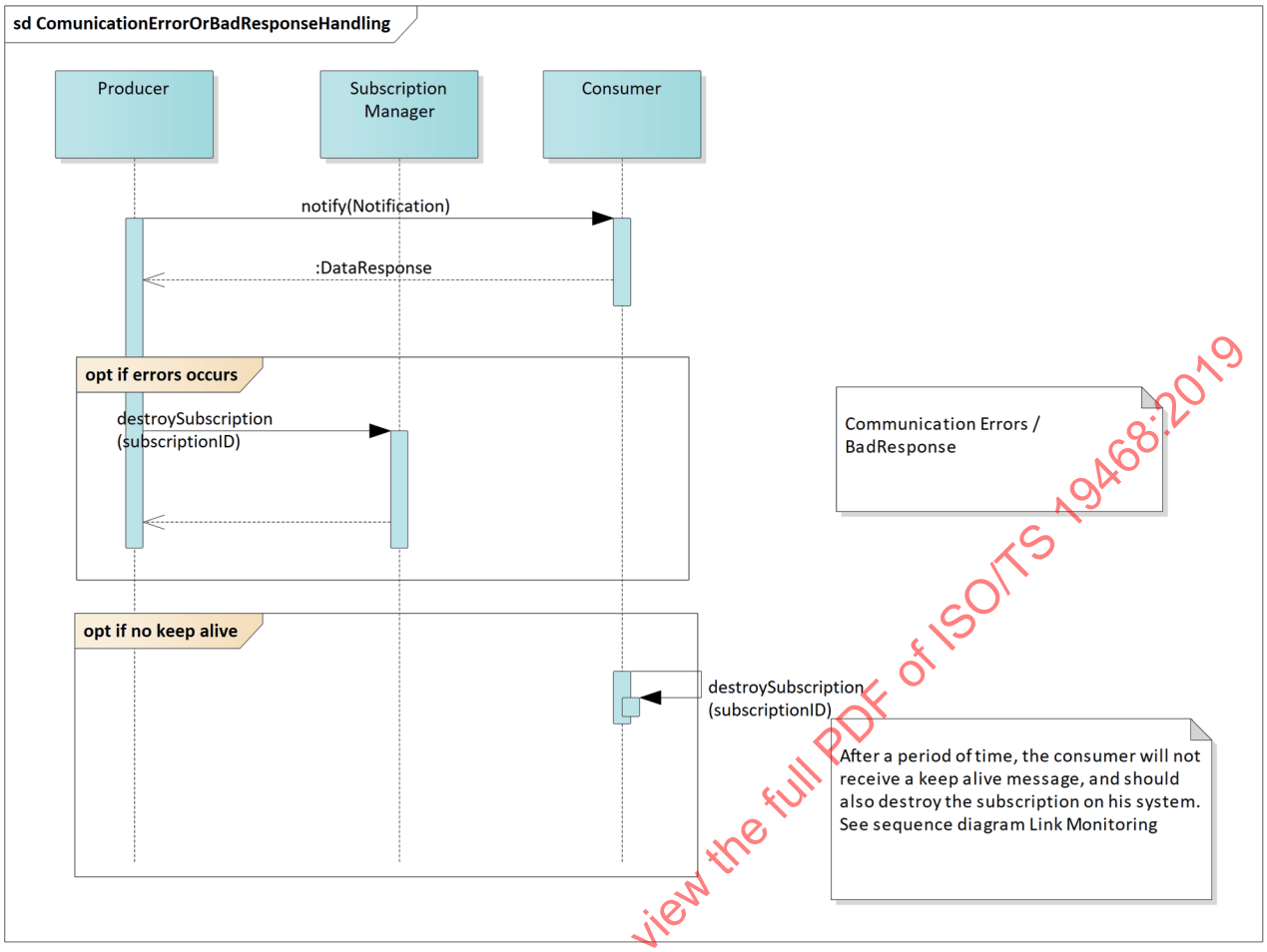

Figure 28 — "ErrorHandling" sequence diagram

Semantics: in case of communications error or bad responses the producer closes the subscription. In case "KeepAlive" messages are not received by the client this one closes the subscription.

10.2.3.3 Data model

10.2.3.3.1 Presentation

Figure 29 describes the "Subscription" data model with the corresponding classes, attributes and relationships.

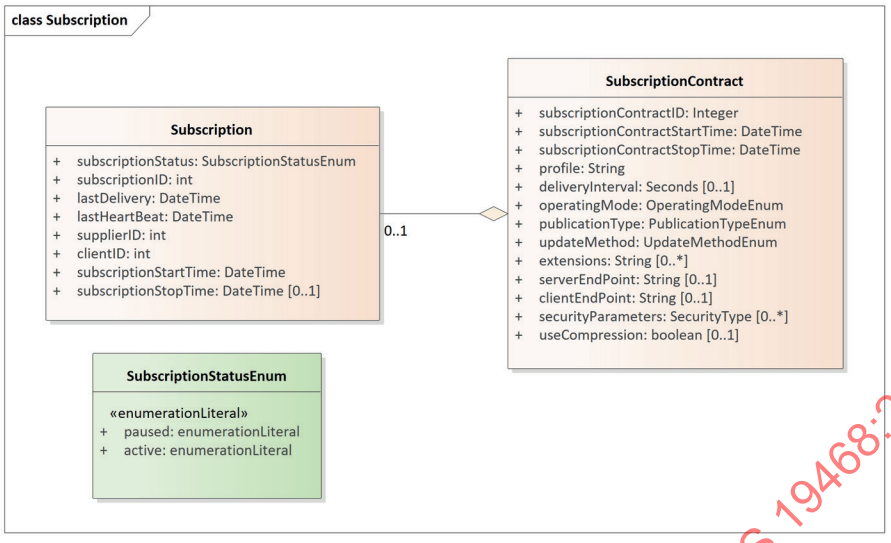

Figure 29 — The "Subscription" data model

10.2.3.3.2 Semantics

A subscription (class "Subscription") refers to an instance of subscription contract (class "SubscriptionContract"). Among the attributes defining it some of them are mandatory, e.g. the subscription status, the IDs of client, supplier and subscription and different time information.

10.2.4 Information management

10.2.4.1 Description

Information is not static and its changes are managed during the active subscription.

10.2.4.2 Features

The information management has the following features:

— Update methods: features to specify what portion of the information needs to be exchanged;

— Operating modes: features that let a data exchange system specify when the information should be exchanged;

— Life cycle Management: features for handling the life cycle management of publications;

  — Situation life cycle management;

  — Filter handling.

10.2.4.3 Exchange data model

Figure 30 below depicts the part of the Exchange data model reduced to only include classes and attributes that address the Information management features.

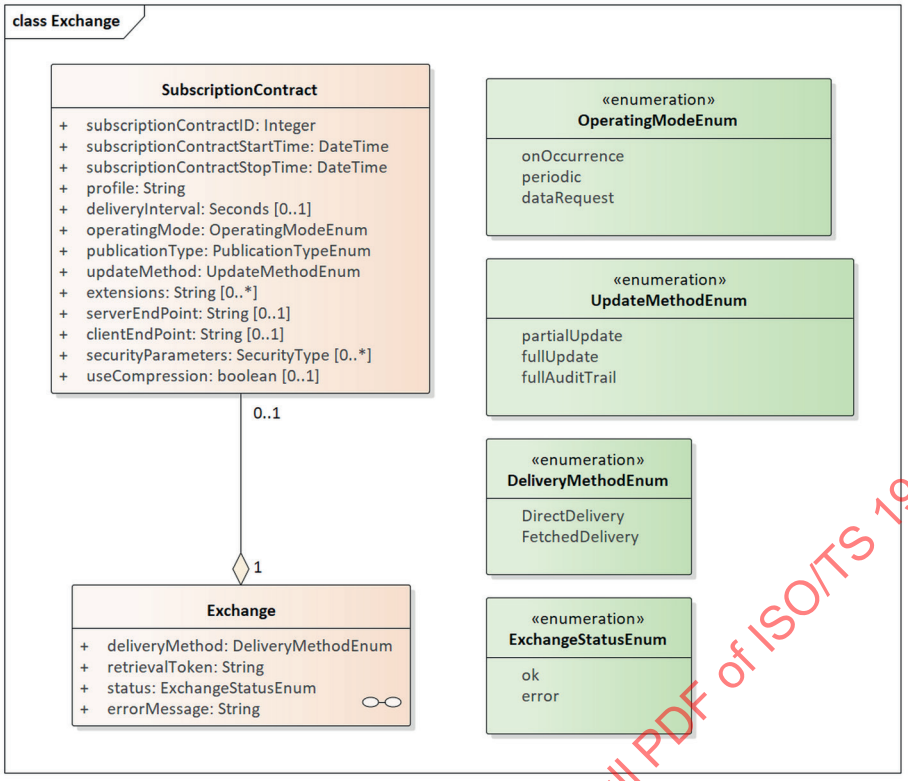

Figure 30 — Information management data model

Semantics: An instance of exchange (class "Exchange") refers to only one subscription contract (class "SubscriptionContract"). All the attributes defining it are mandatory: they are related to the delivery method, the corresponding exchange status, the error message (if any) and the retrieval token.

10.2.4.4 Management data model

Figure 31 below depicts the Management data model reduced to only include classes and attributes that address the life cycle management features.

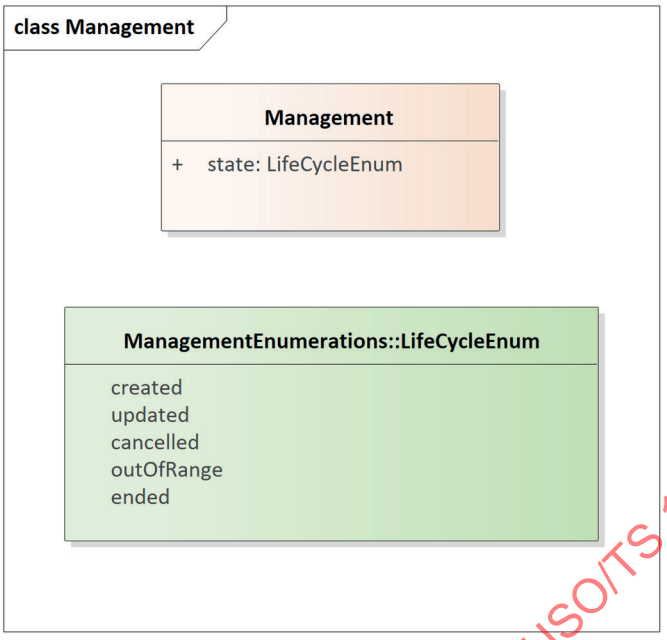

Figure 31 — "Management" data model

Semantics: The "state" attribute of the "Management" class is used to define the actual life cycle state.

10.2.5 Data delivery

10.2.5.1 Description

During a data exchange process there is a need to know how the information published by a supplier is reliably conveyed to its clients:

— The supplier shall know how to physically interact with its clients. With this publish-subscribe architecture the supplier shall use the same platform and the same mechanisms provided by this platform as his clients;

EXAMPLE 1 If the chosen platform is SOAP over HTTP this means that the supplier knows the service endpoint address of his clients.

— The supplier shall create every message with the relevant attributes for the exchange operation, allowing controlling the exchange process itself;

— The messages produced by a supplier should only be delivered to the clients who have valid subscriptions for that information;

— If the published information needs to be complemented with extra information in order to enable clients to properly process it, then the supplier should provide his clients with a mechanism for accessing this contextual information.

EXAMPLE 2 This can be made by using ancillary publications like e.g. in DATEX II the ***MeasurementSiteTablePublication*** that provides static information on traffic monitoring stations.

— If synchronisation is supported, then the supplier should provide his clients with a mechanism to request and receive information that will support the synchronisation process.

10.2.5.2 Features

10.2.5.2.1 Overall introduction

The features related to data delivery are:

— Data delivery: features to deliver information by the supplier to a client in a “push mode” (direct delivery and fetched delivery);

— Data request: features to delivery information requested by the client;

— Large datasets handling: support the exchange of messages with large data volumes;

— Synchronisation: how to ensure data synchronisation between the systems that are communicating;

10.2.5.2.2 Direct delivery

The direct delivery feature is used whenever the supplier sends the data in a single interaction with the client, i.e. during the notification action. This is the most common and preferred way to perform message exchange. It is a mandatory exchange feature.

The sequence of actions is shown in the following sequence diagram in Figure 32.

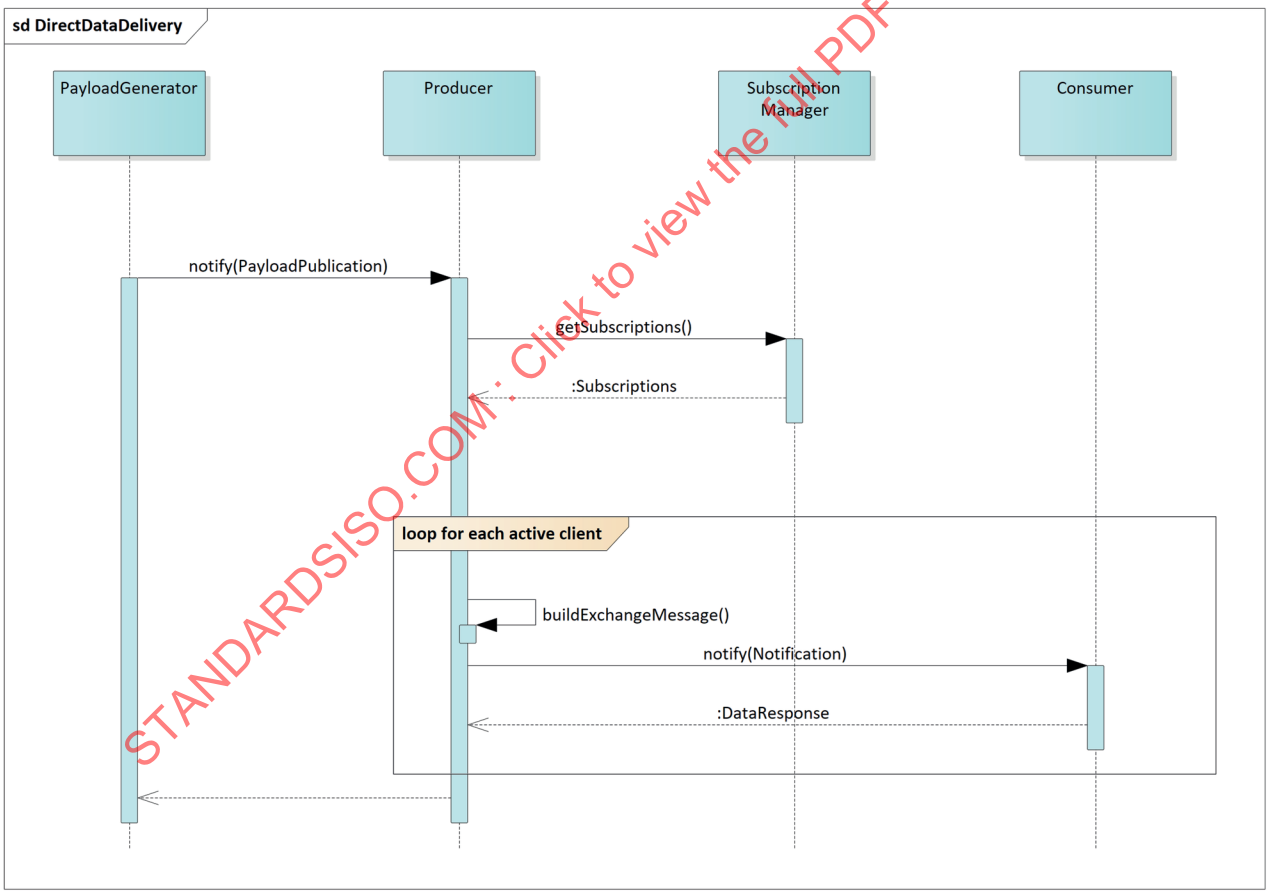

Figure 32 — “DirectDelivery” sequence diagram

10.2.5.2.3 Fetched delivery

A fetched delivery is used to let the supplier tell the client that the data it wants is ready to be fetched, instead of handling it over directly. Therefore, the notification that flows from the supplier to the

client has the single purpose to inform the client about the availability of the data along with a so-called "retrieval method". The retrieval method contains all the information the client needs in order to actually fetch the data. Although the specification provides a default retrieval operation (the "fetch operation"), the actual retrieval method may be agreed on between supplier and client.

The following sequence diagram in Figure 33 illustrates the scenario where the producer sends a message to consumer which indicates that the data is available via fetched delivery.

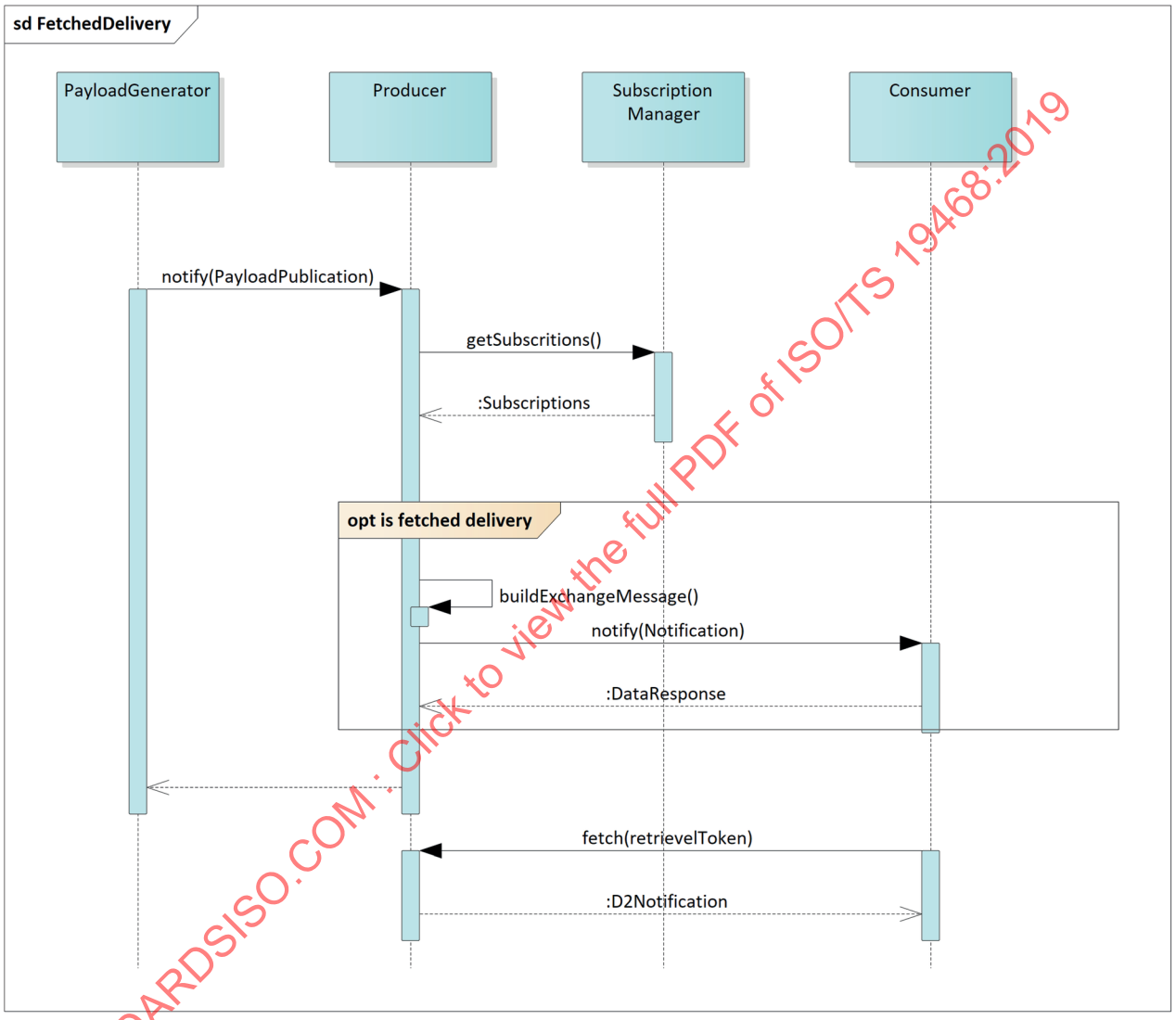

Figure 33 — "FetchedDelivery" sequence diagram

The "Notification" instance contains information that lets the client query the supplier about a specific data set. In that case, the original "Notification" will not carry any content data. This information is part of the "Notification" which is then obtained via the fetch operation. It is an optional Exchange feature.

10.2.5.2.4 Data request

The publish-subscribe pattern states that a publisher (i.e. supplier) sends information to all subscribers (i.e. clients) who have registered themselves for specific information.

Although this general statement does not address all business needs driving an exchange process, sometimes a client needs to receive information the lifespan of which has already elapsed (e.g. at the first synchronisation or after a link failure). The "Data Request" feature can be used by a client to request that specific information be delivered by a supplier.

The following constraints apply for the data request feature:

— Only clients having subscriptions including the “dataRequest” operating mode are allowed to perform data request operations;

— In case a client performs a data request for receiving messages that have already been sent, the update method is not taken into account;

— The supplier may choose not to support the data request operation. In this case, it should not allow creating subscriptions including the “dataRequest” operating mode.

It is an optional Exchange feature. The following sequence diagram in Figure 34 pictures how this feature is operated.

NOTE As explained above, the ":DataResponse" message only corresponds to acknowledgement (or rejection). The actual answer from producer to consumer's data request is realised through the "notify" message.

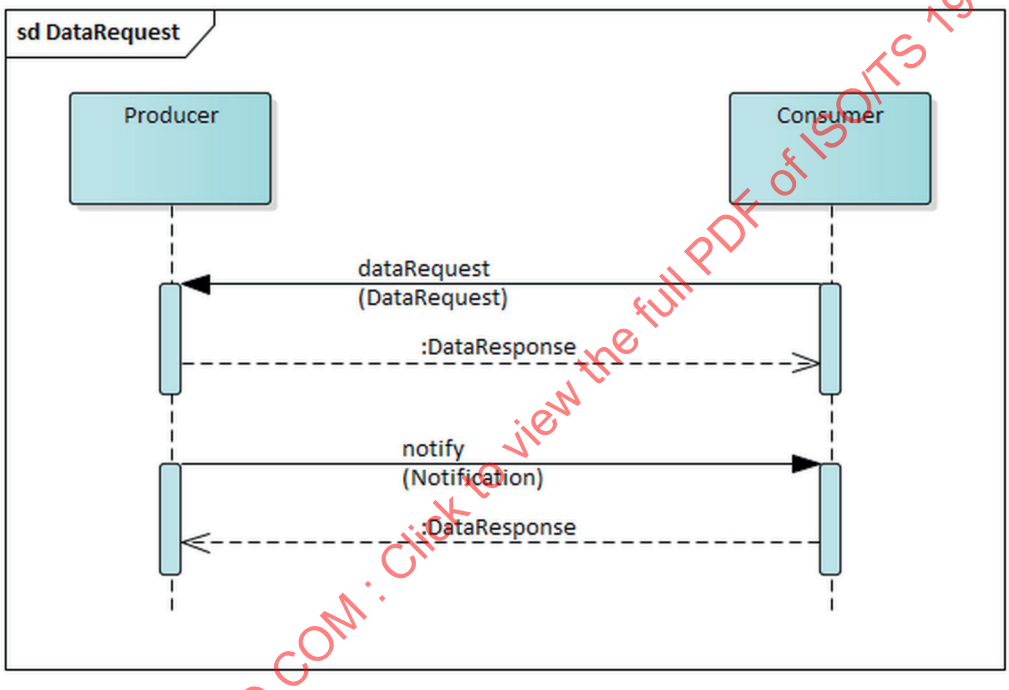

Figure 34 — “DataRequest” sequence diagram

10.2.5.2.5 Large datasets handling

When considering large amounts of information, a solution is to use compression mechanism in the data exchange process. This document does not define any mechanism that supports the compression capability. That is left to the communication protocol in use.

An alternative approach consists in using the fetched delivery mechanism when the payload content is too big to be exchanged using the regular event driven approach of the publish-subscribe architecture.

10.2.5.2.6 Synchronisation

When two exchange systems engage into a data exchange process they often need to reach a common baseline before the normal data exchange actually starts. This operation is known as synchronisation and happens due to several reasons, e.g. initialisation of client’s exchange system, restart after a system failure or a communication break.

The mechanism available to support the synchronisation process is based on data request. Two types of synchronisation are considered:

— "**Regular**": – used when the client asks for publication information based on the "fullAuditTrail" update method.

— "**Replay**": special type of synchronisation used to ask the supplier to re-send messages that were already sent in the course of a subscription. The session is either the current subscription or a previous subscription, provided that the supplier allows the replay of previous subscription messages.

The synchronisation mechanism depending on the availability of the data request feature is an optional feature.

10.2.5.3 Data model

The data model in Figure 35 below shows the structure of elements that are used to convey exchange information between exchange systems. It consists of a top level "Notification" class which contains a mandatory class named "Exchange" designed to accommodate exchange-specific information and an optional sub-class named "NotificationData" which holds the actual information being exchanged.

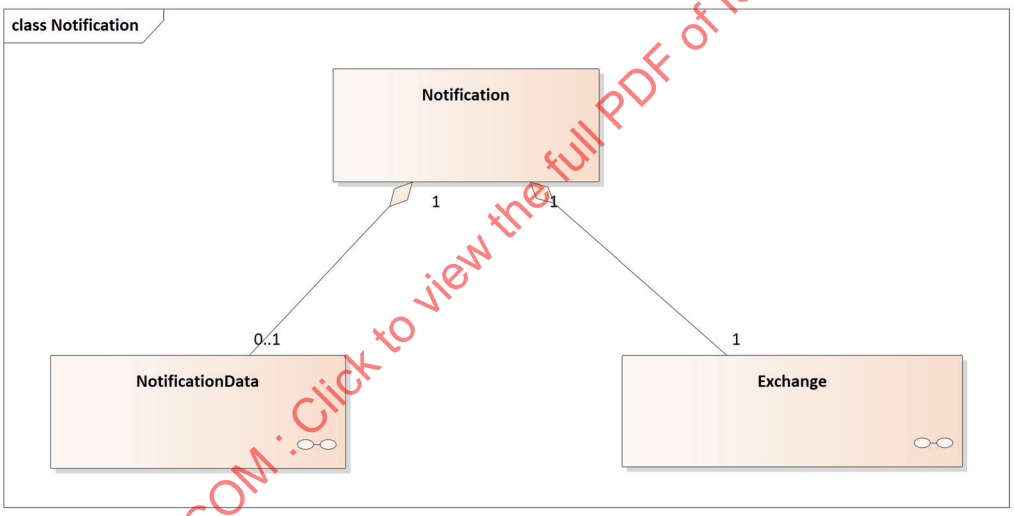

Figure 35 — The "Notification" data model

The "NotificationData" package is pictured according to the following Figure 36.

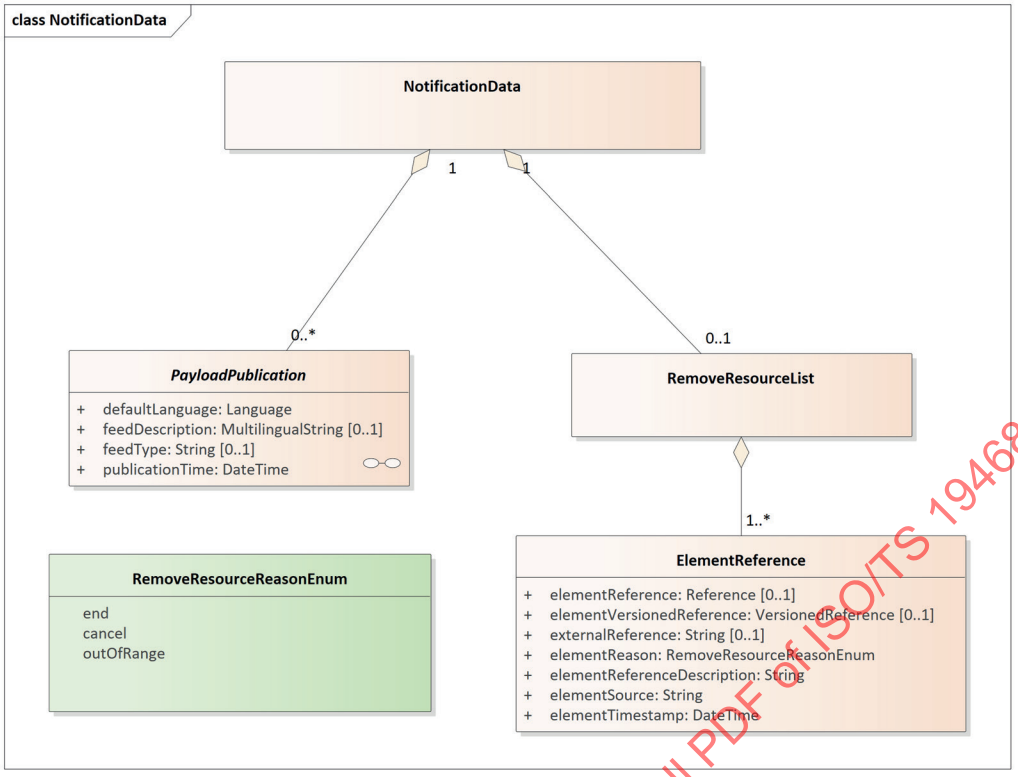

Figure 36 — The "NotificationData" class model

10.2.6 Communication and protocol

As described in the context diagram the communication layer deals with the support for communication between systems, and defines the protocols and services that can be used by data exchange, e.g. TCP/IP, HTTP, Web services, security.

At the communication layer level, the exchange specification defines the following features:

— Protocol of communication used for exchanging data;

— Security features that can be implemented in data exchange;

— Methods for data compression while transmitting data.

This document presents a model for the exchange specification to allow the implementation over multiple platforms, each described in the corresponding platform-specific model (PSM). The exchange specification is not limited to any protocol, security and compression solutions; the informative Annex D introduces some protocols available that can be used for PSMs.

10.3 Publish-Subscribe Functional Exchange Profiles

10.3.1 Overview

This FEP deals with Information currently available at the supplier system that needs to be delivered to the client system. Furthermore, in some business scenarios the client needs to receive all relevant information that has been available to the supplier even during disconnected periods, e.g. due to technical problems such as network disruption or due to system maintenance. Thus, there is also a need to deliver information that had been available during such disconnected periods in the past to synchronize the client and avoid loss of data.

It is important to note that some state information needs to be shared by the supplier and client in order to deliver past and/or incremental information. This shared state enables FEP to manage errors and achieve a full synchronisation of client and supplier. The state information itself can be stored either at the supplier or at the client system which is out of the scope of an exchange system environment.

This FEP focuses on the described requirements and presents a profile that is based on the PIM and the Publish-Subscribe pattern, which allows for a reliable implementation of data exchange among systems, and many specifications derived from this pattern are available.

A PSM description of this pattern on web services is to be provided in an accompanying document, implementing all data exchange requirements for reliable traffic and travel information exchange. These requirements will be analysed in the following subclauses.

10.3.2 Objectives

The simplest form of a data exchange scenario is providing information of a data supplier to a client, just for information or to allow further processing of the data. The client can select a part of the data available, e.g. providing filters, but is not able to modify the content of the information provided to him, neither semantics nor syntax. The business scenario Information Delivery provides requirements for this way of information exchange.

Based on the requirements defined in the business scenario for information delivery, this FEP addresses the following objectives:

— Allow the establishment of an agreement or a contract on-line or off-line, between the supplier and client;

— Manage the information life cycle and ensure that both supplier and client share the same status of the information that has been exchanged;

— Ensure the exchange of all information between supplier and client, in an efficient, robust and consistent way;

— Be able to establish a session between the supplier and client and deal with errors and failures of communication;

— Provide high levels of security;

— Use efficient and optimized mechanisms of communication.

A FEP contains the selection of exchange features for a specific business scenario. There are several levels of functional requirements for information delivery, and each can ask for a different FEP. This FEP focuses on an extensive requirement set for an Exchange specification, based on the Publish-Subscribe pattern, and is therefore is called "**Full Publish Subscribe"**; other FEPs for different information delivery scenarios are expected to have a subset of this feature set or contain fewer complex features due to less complex requirements. The Full Publish Subscribe FEP has a list of requirements for each objective that are retailed in the normative Annex B.

11 Other PIM definitions

This clause is reserved for future new models/patterns supporting new business scenarios, such as "Collaboration ITS Services".

The definition of the business scenario is mandatory prior to the PIM description because a PIM implementation is "connected" to a business scenario.

CIS can be implemented using bidirectional data delivery approach for service request and feedback exchanges as possible PIM for a non-pre-emptive CIS implementation as described in Annex H.

Annex A
(informative)

Methodology presentation

A.1 Introduction

This informative annex presents the approach followed for this document and the rationale behind the choices made.

A.2 Apply Model Driven Architecture

The original approach taken for modelling the full set of aspects covered by data exchange clearly identified the need to separate the exchange specifications, of what needs to be exchanged and the exchange of the data itself. This is reflected by creating two independent interoperability domains:

— The information interoperability domain as reflected in the PIM defining content (e.g. of DATEX II);

— The exchange interoperability domain as reflected by this document.

While the payload content model can be regarded as describing "what to exchange", the present exchange specification deals with the problems about "how to exchange".

A distinction has also been made in the modelling phase to separate the abstract model from its concrete implementation(s). In practice, this approach led to the creation of a Platform Independent Model for exchange (PIM), which described the concepts behind exchange, and one or more Platform Specific Models (PSMs) which defined how the abstract model would actually be implemented on specific technical platforms. As this basic principle of the Model Driven Architecture (MDA) has guided all the work that was already undertaken for creating the full set of specifications for content definition with notable success, it was chosen as well for the Exchange Specification definition.

Therefore, this exchange specification is based on a Platform Independent Model for exchange (the Exchange PIM), which is detailed in one or more documents containing specifications targeted to specific platform implementations (PSM).

A.3 Use case driven

One of the principles followed in this document is to clearly identify the business scenarios that are addressed by the specifications plus the full set of features that can be available for implementing actual systems based on this document for each of those Business Scenarios. This led to the identification of two main business scenarios:

— Information Delivery;

— Collaborative ITS Services.

The information delivery business scenario addresses the exchange of traffic and travel information between two data exchange nodes, whereas the Collaborative ITS Services business scenario broadens this approach by including the possibility of having one data exchange node stimulating actions within another data exchange node by requesting the execution of a particular service offered by this node.

These two business scenarios clearly show distinct characteristics, but in order to fully describe them, both need to be detailed further, leading to different possible options. Therefore, the approach was to

further refine these general business scenarios into particular, more detailed use cases, each of which address specific requirements.

The term use case is used here to describe a set of interactions between entities (called actors) and the system being analysed, providing a better understanding of the main functions behind such interactions.

In the case of the information delivery domain, the actors involved in the corresponding use cases are the supplier of exchanged information and the receiver of such information – the client. For the Collaborative ITS Services domain, the actors involved are the entity requesting the ITS services, or the service consumer, and the entity providing the service, i.e. the service provider.

A.4 Functional Exchange Profiles

When analysing the business scenarios and requirements for this document, it became apparent that different use cases within such a business scenario might contain disparate requirements, which could not be easily accommodated into a single, specific implementation. Even worse, some use case might even contain requirements that are, by nature, contradictory, depending on the business needs of the user community they originated from.

Having that in mind, this specification identifies the full set of features that can be available for a data exchange process to take place. Then – in a second step – the specification reflects the particular use cases and selects the best suited set of features for each of them. This selection is denoted a ***Functional Exchange Profile*** – FEP.

A.5 Profile-to-platform mapping

By performing a short survey on the capabilities that modern ICT platforms offer it became apparent that some of the features indicated for implementing data exchange systems can be realised differently depending on the platforms chosen. As such, the *one-size-fits-all* approach that was followed on the previous version of the Exchange Specification for creating concrete implementations simply does not fit into the new paradigm.

At the heart of the current specification sits the Platform Independent Model (PIM) for data exchange. Its purpose is to model the interactions that were identified for each use case in an abstract, platform independent way. The Exchange PIM is detailed in this document.

This platform independent specification then has to be mapped to a realisation of the indicated FEP on a specific technical platform, described in a Platform Specific Model (PSM) for data exchange, e.g. ISO 14823-3 defines such a mapping. The act of matching a FEP to a platform is also known as **profile-to-platform mapping**. Each one of these mappings form what is called an **interoperability domain**. It is only in the scope of such an Interoperability Domain that two data exchange nodes can expect to be interoperable, i.e. if two different implementers of data exchange systems need to be sure that their system will be able to communicate, they both need to follow the same mapping.

Annex B
(normative)

Definition of requirements

B.1 Information requirements

The primary intent of the information delivery scenario is the provision of information by a supplier to a client. This subclause provides requirements that are related to the data actually being transferred from supplier to client. For each requirement it is stated whether applicable to data delivery or Collaborative ITS Services (CIS) business scenarios, assuming for CIS supplier is intended as service requester and client as service provider and data are exchanged or processed as input to provision of services.

Table B.1 — Information requirements

Requirement	Description	Data delivery	Collaborative ITS services
Simple server	In a simple exchange the supplier keeps no track of previous interactions that have occurred between with any of its clients. Therefore, the client is responsible for providing the supplier with all information it can need in order to serve his request. Note that this requirement does not oblige the supplier to have a mechanism for receiving state information as that depends on the availability of other requirements described in this clause (e.g. filter handling).	Y	
Stateful server	Any supplier implementing this feature should provide some sort of state keeping mechanism that will allow it to know what information was sent to his client. Therefore, it would be possible for the supplier to send only information that has not been already sent before. A supplier implementing this mechanism can also support other requirements that allow clients to shape the actual data set being received (e.g. filter handling).	Y	Y
Subscription	The information that a supplier delivers to a client is defined by a subscription. A subscription results from an interchange/license agreement, where both parties agree on parameters like, e.g. the information type and periodicity that should be observed upon data delivered. The supplier can choose to reject requests without proper subscription, or it can deliver a standard set of information.	Y	Y
Catalogue exchange	The information a supplier provides is described in a catalogue. The catalogue is used to identify the information contained in a subscription.	Y	Y
Self-description	The ability of two nodes to exchange information that will allow them to negotiate the requirements supported by both of them (handshake). This feature deals with exchange parameters, which is different from a catalogue exchange that deals with content.	Y	Y
Filter handling	A supplier implementing this feature enables clients to provide specific filters that shall be applied to the information being exchanged.	Y	Y

Table B.1 *(continued)*

Requirement	Description	Data delivery	Collaborative ITS services
Client profiles	A client profile lets suppliers shape the information they send according to the requirements of the client requesting it.	Y	Y
Interchange/License agreement	Legal artefact where parties taking part in data exchange define the terms and conditions that govern the whole exchange process (e.g. Non-disclosure of information, service level agreements, etc.).	Y	Y
Integrity	This feature implies that the data that is prepared by the supplier should reach its intended recipient without being tampered in any way, semantic, structural or other.	Y	Y
Full audit trail data delivery (all state changes)	All data item versions are delivered to the client.	Y	Y
Snapshot data delivery (last known state)	Only the current version of the data items is delivered to the client.	Y	Y
Incremental Data Delivery	A mechanism where the server sends only the information that has changed since the previous exchange cycle	Y	Y
Reference datasets for different versions	Service a supplier should provide for the client to access referenced data in a versioned way. Even if new versions appear the old versions remain accessible.	Y	Y
Extensibility	Extensions to the data exchange protocol should be supported; therefore, any implementation of the data delivery use case shall take into account that the protocol can evolve. Thus, each message should state the protocol version it refers to.	Y	Y
Support for life cycle management	A set of function for updating information at client systems according to updated information gathered at supplier system	Y	Y
Support for information management	A set of functions that will let the supplier (as service requester) tell the client (as service provider) what to do with the information being exchanged (processing directive).		Y
Support for feedback on information management	A set of functions that will let the client (service provider) tell the supplier (service requester) the outcomes of processing the information by the stated directive which had been exchanged.		Y
Distributed transaction	A capability for the exchange system to coordinate among several involved service provider systems to collaborate in implementing a distributed service.		Y
Distributed atomic transaction	A capability for the exchange system to coordinate among several involved service provider systems to collaborate in implementing a distributed service keeping consistency in transaction.		Y
Synchronization	A mechanism that lets clients request the whole set of information currently known to the supplier to ensure that its internal data structures be in exactly the same state as those of the supplier. (intended for CIS the Service provider needs the exact information to provide the requested services)	Y	Y
Delayed delivery	In the case that preparing and sending a data set by the supplier would take too much time to complete, the supplier should inform the client about this fact. This mechanism should also define how and when the client would be able to access the information it needs.	Y	

Table B.1 *(continued)*

Requirement	Description	Data delivery	Collaborative ITS services
Data delivered as soon as possible	This feature is used to ensure that the supplier sends the information as soon as it becomes available in its system.	Y	Y
On demand request (query)	The ability of the client to ask the server for information it needs whenever it wants (Client pull).	Y	

B.2 Communication requirements

Communication requirements characterise the mechanism that data exchange nodes implement in order to address problems specific to the communication layer of the data exchange process. These requirements are completely unaware of both the security and information features that are in use. The idea is that given a supplier has prepared a particular dataset; the requirements described in this subclause can be used to successfully convey it to the client.

Table B.2 — Communication requirements

Requirement	Description	Data delivery	Collaborative ITS services
Sessionless	No session is used during the data exchange process.	Y	Y
Session	Client and supplier negotiate and establish a session before starting to exchange information. The session parameters constitute state information shared between both stations.	Y	Y
Request/response	A mechanism where the client requests the data and instantly receives the response.	Y	Y
Delivery/response	A mechanism where the supplier initiates the data delivery process, while the client patiently waits for it.	Y	Y
Error handling	A mechanism that allows both client and supplier to detect that an error has occurred during the exchange process and to decide what actions should be taken to handle it.	Y	Y
Timely responses	The communication layer should introduce a minimum delay on the data delivery process, ideally none.	Y	Y
Time-out management	The ability to handle time-out situations that happen during an exchange process.	Y	Y
Exchange quality measures (ex. response timestamp)	The messages exchanged should include extra information that allows both parties involved to measure the quality of service of the communication layer.	Y	Y
Logging	A mechanism for storing information about exchange activities, which could then be used to analyse the whole process.	Y	Y
Failed data recovery	When the supplier fails to deliver the information to the client, this feature ensures that the failed data messages will be successfully delivered to the client at a later time.	Y	Y
Message sequence	A mechanism that allows identifying each message exchanged between two entities by including a unique sequence number.	Y	Y

Table B.2 *(continued)*

Requirement	Description	Data delivery	Collaborative ITS services
Full reliability	A mechanism that ensures that data sent by a supplier is really received by the client provided that both client and supplier are active and have the ability to communicate. This does not include any semantic validation. Syntax validation is optional.	Y	Y
Link monitoring and control	This feature enables both parties to continually check whether the communication link works properly and act accordingly when it is broken.	Y	Y
On occurrence update	A supplier implementing this feature should send the information to the client as soon as it is available.	Y	
Periodic update	A supplier implementing this feature should buffer all the information to be sent to a particular client for a pre-defined time period send information only when this period has elapsed.	Y	
Multi-part data delivery	When the size of the information to be delivered is too large, the supplier can choose to deliver it in chunks. At the first request, the supplier returns the first data chunk. The response contains the message ID and the total number of parts (chunks) that comprise the dataset. The client then has to request each of the remaining parts of the message.	Y	Y
Compression	The ability to pack the same information in a smaller amount of data in order to decrease the transmission time.	Y	Y

B.3 Security requirements

The requirements described in this subclause deal with all aspects used to provide security services at any of the different communication levels, such as peer authentication, channel security and so on.

Table B.3 — Security requirements

Requirement	Description	Data delivery	Collaborative ITS services
Client authentication	The act of establishing or confirming a client as *authentic*.	Y	Y
Client authorization	The process of verifying if a client is allowed to access some resource (commonly referred to as *read access*) or execute some action (commonly referred to as *write access*).	Y	Y
Supplier authentication	The act of establishing or confirming a supplier as *authentic*.	Y	Y
Supplier authorization	The process of verifying if a supplier is allowed to access some resource (commonly referred to as *read access*) or execute some action (commonly referred to as *write access*).	Y	Y
State of the intended recipient	The act of indicating the destination peer of a message.	Y	Y
Confidentiality	Ensuring that information is accessible only to those authorized to have access (ISO 27002).	Y	Y
Client identification	A mechanism that allows a client to provide his identity.	Y	Y

Table B.3 *(continued)*

Requirement	Description	Data delivery	Collaborative ITS services
Supplier identification	A mechanism that allows a supplier to provide his identity.	Y	Y
Non-repudiation	A mechanism to guarantee that the sender (supplier of a client) of a message cannot later deny having sent the message.	Y	Y

B.4 Financial/economic requirements

Although being not at the same level the requirements described in this subclause have some economic/financial impact on the actual implementation of the data exchange sub-system.

Table B.4 — Financial/economic requirements

Requirement	Description	Data delivery	Collaborative ITS services
Reasonable TCO (Total Cost of Ownership)	The data exchange sub-system should have a reasonable TCO (Total Cost of Ownership).	Y	Y
Expandability at a reasonable cost (scalability)	The data exchange sub-system should be implemented in such a way that it can be possible to increase the capacity of the system at a reasonable cost. Capacity relates to any of the system resources, such as data volumes, computation power, parallel processing, etc.	Y	Y
Low processing resources	It shall be possible to implement a data exchange sub-system on systems with low processing resources.	Y	Y

Annex C
(normative)

Basic exchange data model and data dictionary

C.1 Overall presentation

The described data model is based on a UML methodology that is independent from any technical platform. It describes the whole data needed to implement in this version the information delivery business scenario besides the only Snapshot Pull and Snapshot Push FEP and exchange patterns.

Whenever exchange features as session management, link monitoring are needed some extra information need to be conveyed among supplier and client to enable control and reliable exchange management.

Furthermore, in specific contexts, more than one Information Payload occurence can be exchanged, so that this model further allows exchanging multiple payloads based on such extra exchange requirements.

As already explained this model is not required to implement FEPs like Snapshot Pull and Snapshot Push.

C.2 Basic exchange data model

C.2.1 Overview

Exchange data model is used to convey information which is needed to manage exchange features such as information management, session management, link monitoring.

Further information is needed to manage CIS.

C.2.2 The MessageContainer class diagram

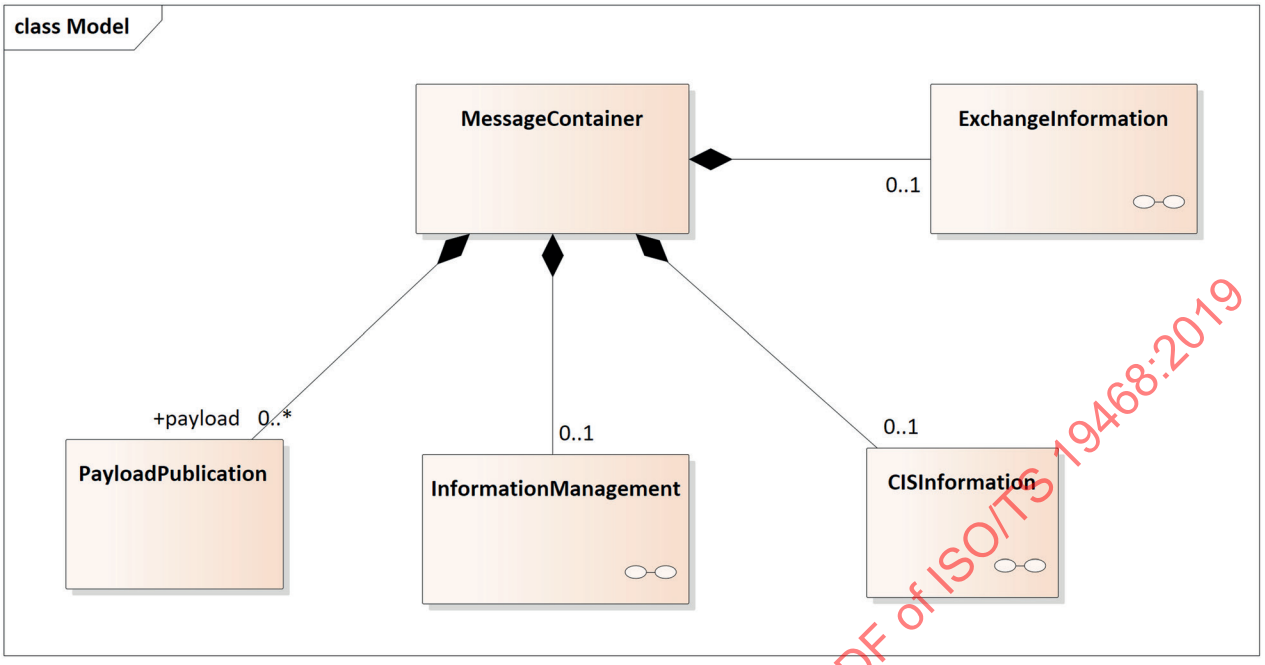

Figure C.1 — The MessageContainer class diagram

A MessageContainer class is introduced to allow delivery of further information besides payload such as:

— Exchange Information: which is needed to manage exchange feature such as session management and link monitoring and exchange error management.

— Information management related information.

— Collaborative ITS Services information.

All linked classes are optional, constraints on implementation depending on different FEP+EP which implement the exchange and its features, e.g. an Exchange class will be mandatory in a FEP+EP which needs to implement session management, such as Stateful Push FEP+EP.

The MessageContainer class also allows conveying multiple payloads within a single exchanged message.

C.2.3 The ExchangeInformation class diagram

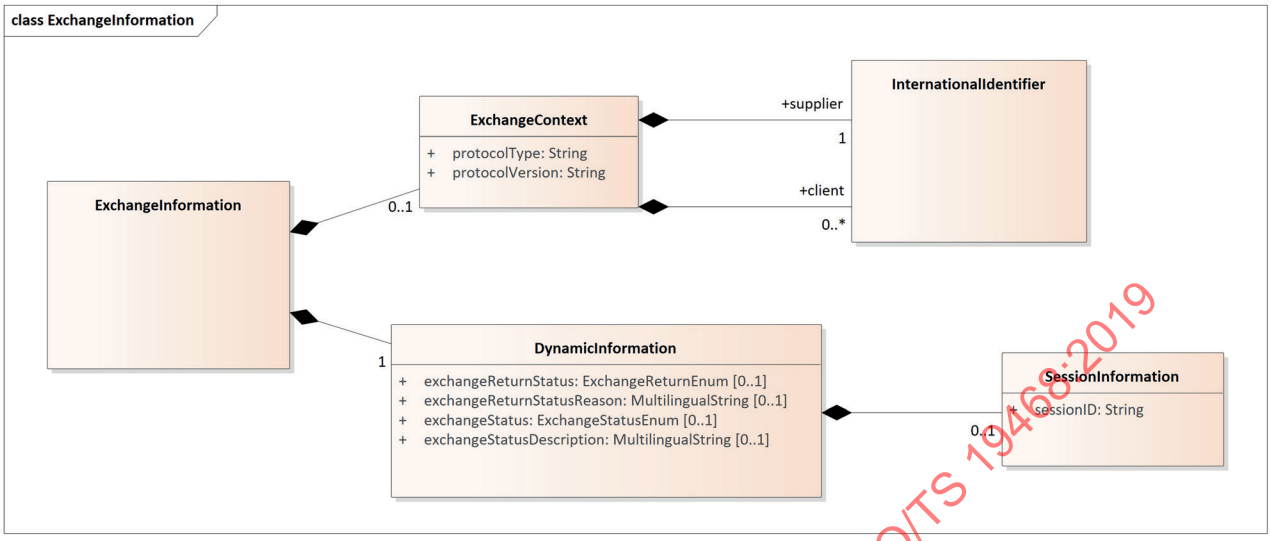

Figure C.2 — ExchangeInformation class diagram

Exchange Information is used to convey information about Exchange Environment such as:

— Exchange Context, which is implied by supplier and client(s) identification, the type of protocol which is defined to be used in exchanging data, the implemented version of this protocol.

— Dynamic Information: such as exchange status and return status (within a set of predefined exchanged status and return status enumeration lists and their reason) and SessionID for exchange patterns which manages sessions.

Exchange context is optional as mostly implicit in subscription contract and normally can be omitted so the link to ExchangeContext class is not mandatory. Otherwise such information could be used to simplify or double check some communication features such as supplier and client identification, authentication, authorisation, which can only be implemented in a reliable way based on communication layers.

Dynamic information is the only mandatory class in which the attribute exchangeStatus is the only mandatory information. Other information will be mandatory whenever needed to be managed in different FEP+EP, i.e. in case of session management feature implementation SessionID attribute in class SessionInformation will be needed to be managed.

C.2.4 The InformationManagement class diagram

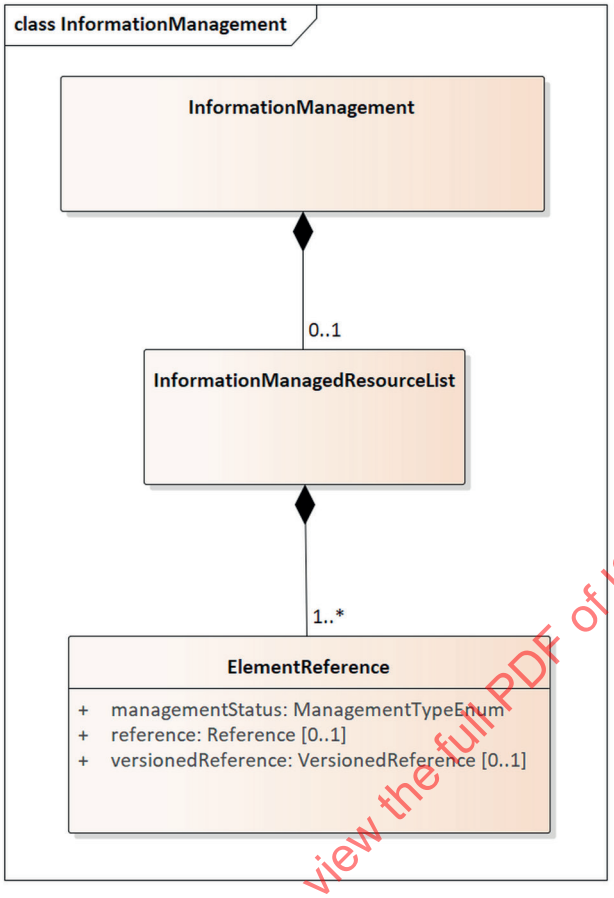

Figure C.3 — InformationManagement class diagram

InformationManagement class allows delivering information management data: as defined in Annex F for life cycle information only valid information is managed in the payload, whenever an element ends its life cycle, i.e. being closed or cancelled, it is no longer conveyed in the payload and information management is to be delivered to manage closures or cancellation of such elements.

These closed and cancelled elements are conveyed linked through InformationManagedResourceList class and are addressed by ElementReference class which allows identification of the element by its reference or versioned reference, specifying its managementStatus, i.e. closed or cancelled.

C.2.5 The CISInformation class diagram

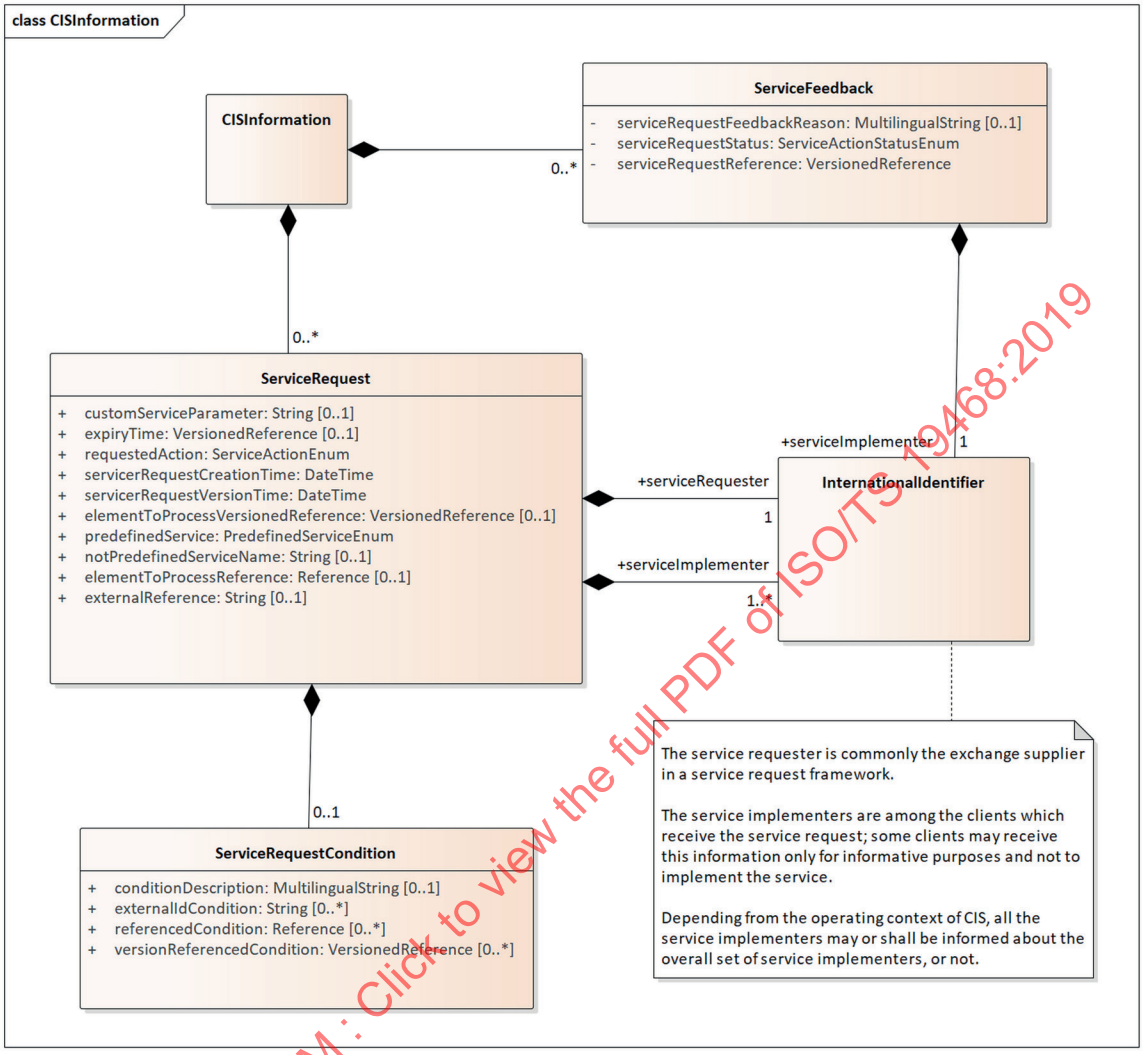

Figure C.4 — The CISInformation class diagram

With reference to Annex G Collaborative ITS Services transform the vision of supplier and client in service provider and service requester: CISInformation conveys in the data model the information needed to implement service request and service feedback as described to be exchanged between service provider and service requester.

CISInformation class: allows to convey any Service Request and Service Feedback which are requested to be exchanged among Service Requester and Service Providers through the respective named classes ServiceRequest and ServiceFeedback.

ServiceRequest: conveys information related to a single Service Request

— Service Request creation and update time,

— predefinedService: to be chosen among an enumerated list such as processing, broadcast delivery, VMS message process, Traffic Management Plan activation,

— not predefined service name,

— custom service parameter: custom parameter to raise for processing,

— requestedAction: Approve, implement, terminate, cancel,

— element to process reference or external reference,

— expiry time when a service is valid only implemented within a limited time amount after which it will be no longer needed.

Service Request can be related to a Service Condition through the ServiceCondition class identifying the needs for that kind of service through some payload reference for a predefined, coded or not predefined, i.e. unpredictable condition.

ServiceRequest class also allows addressing the identification of involved Service Providers needed to implement such request and the corresponding service.

ServiceFeedback: allows to convey information about the processing of a Service request previously exchanged through reference information to the Service Request itself and specifying its Status among a set (compliant, failed, not compliant, processing, rejected, scheduled, success, timedOut) plus a reason for that status.

C.3 Data dictionary overview

This data dictionary describes the characteristics of the different classes, attributes and roles appearing in the data model defined in C.2. The dictionary is specified as a set of tables grouping classes, attributes and roles for each package as they are defined in C.2.

For each package the following are successively provided:

a) The class table,

b) The association table,

c) The attribute table.

Each table of a given type has the same structure.

The data dictionary is categorised into sections following the different UML model packages as mentioned above. It defines for every package the entities and elements corresponding.

The table columns have the following meaning:

a) Column **Name**: they provide the symbolic name (either in Lower or Upper Camel Case) given to the corresponding class, attribute or association role.

b) Column **Designation**: it provides the corresponding name in natural language of the corresponding class, attribute or role.

c) Column **Definition**: it provides a comprehensive definition detailing this class, attribute or association role.

Some columns are specific for one or two tables. The class tables include the following two columns:

d) Column **Abstract**: it indicates if the corresponding class is abstract (value “yes”) or concrete (value “no”). Abstract classes are defined in ISO/IEC 19505-1.

The attribute tables and the association tables include the following column:

e) Column **Multiplicity**: it provides the number of occurrences a class may have when instantiating this association (resp. a class attribute may have when instantiating this class). The adopted syntax is the following: m..n where ‘m’ and ‘n’ respectively represent the minimum and the maximum value of multiplicity.

   For association roles, the possible value for ‘m’ are:

1) 0 in case of an optional participation of the corresponding class when instantiating the association;

2) 1 in case of a mandatory participation of the corresponding class when instantiating the association;

3) 2, 3, … in case a minimum number of participations of the corresponding class is explicitly defined when instantiating the association.

For association ends, the possible value for ‘n’ are:

4) 1 in case only one class instance is at most participating at the association instantiation;

5) * in case several instances are allowed participating at the association instantiation;

6) 2, 3, .. in case a maximum number of participations of the corresponding class is explicitly defined when instantiating the association.

For attributes, the possible value for 'm' are:

7) 0 in case of an optional attribute;

8) 1 in case of a mandatory association / attribute;

9) 2, 3, … in case a minimum number of occurrences is explicitly defined.

10) For attributes, the possible value for 'n' are:

11) 1 in case only one attribute instance is allowed;

12) * in case several instances are allowed for this attribute without being specified;

13) 2, 3, .. in case a maximum number of occurrences is explicitly defined.

For the attribute tables, the following column has been added:

f) Column **Type**: it provides the class name used as data type. It is only provided for elements corresponding to class attributes. When the type name ends with 'Enum' this means it corresponds to an enumeration of accepted values defined in C.6.

For the association table, the following column has been added:

g) Column **Target**: it provides the class name appearing at the second end of the relationship, i.e. linked through the corresponding association.

C.4 Data Dictionary for “ExchangeDataModel”

C.4.1 "Classes" package for “CISInformation”

MessageContainer/CISInformation/Classes

C.4.1.1 "Classes" package classes

Table C.1 — Classes of the "Classes" package

Class name	Designation	Definition	Abstract
CISInformation	CIS information	CIS information.	no
ServiceFeedback	Service feedback	Feedback about a specific service request from the service implementer to the requester.	no

Table C.1 *(continued)*

Class name	Designation	Definition	Abstract
ServiceRequest	Service request	Service request from the service implementer to the requester.	no
ServiceRequest-Condition	Service request condition	Specifies the condition which is behind the need for the service request, e.g. a specific situation or situation record, travel times status, specific road data or external conditions.	no

C.4.1.2 Associations of "Classes" package

Table C.2 — Associations of the "Classes" package

Class name	Association end	Designation	Definition	Multiplicity	Target
CISInformation	serviceFeedback	Service feedback		0..*	ServiceFeedback
	serviceRequest	Service request		0..*	ServiceRequest
ServiceFeedback	serviceImplementer	Service implementer	It identifies the international identifier requester feedback.	1..1	International-Identifier
ServiceRequest	serviceRequester	Service requester	It identifies the international identifier requester.	1..1	International-Identifier
	serviceImplementer	Service implementer	It identifies the list of international identifier implementers.	1..*	International-Identifier
	serviceRequestCondition	Service request condition		0..1	ServiceRequest-Condition

C.4.1.3 "Classes" package attributes

Table C.3 — Attributes of the "Classes" package

Class name	Attribute name	Designation	Definition	Multiplicity	Type
ServiceFeedback	serviceRequestFeedbackReason	Service request feedback reason	Additional text to feedback about the status of the service request.	0..1	MultilingualString
	serviceRequestReference	Service request reference	Reference to the service request to which refers the service feedback.	1..1	VersionedReference
	serviceRequestStatus	Service request status	Specifies the status of service request referenced.	1..1	ServiceActionStatusEnum
ServiceRequest	customServiceParameter	Custom service parameter	A string conveying information for custom parameter to service.	0..1	String
	elementToProcessReference	Element to process reference	Element reference to be processed.	0..1	Reference
	elementToProcessVersionedReference	Element to process versioned reference	Element versioned reference to be processed.	0..1	VersionedReference
	expiryTime	Expiry time	Date and time until which to implement the required action for service.	0..1	VersionedReference
	externalReference	External reference	External reference to be processed.	0..1	String
	notPredefinedServiceName	Not predefined service name	Name of service not predefined.	0..1	String
	predefinedService	Predefined service	Type of predefined service.	1..1	PredefinedServiceEnum
	requestedAction	Requested action	Identifies the action requested for the specified service.	1..1	ServiceActionEnum
	servicerRequestCreationTime	Servicer request creation time	Time of request creation.	1..1	DateTime
	servicerRequestVersionTime	Servicer request version time	Time of request version.	1..1	DateTime
ServiceRequestCondition	conditionDescription	Condition description	A multilingual description of the condition under which the service request is instantiated.	0..1	MultilingualString
	externalIdCondition	External id condition	An external reference ID to the condition for the service request.	0..*	String
	referencedCondition	Referenced condition	The list of condition information which are referenced by an identifier in payload publications.	0..*	Reference

Table C.3 *(continued)*

Class name	Attribute name	Designation	Definition	Multiplicity	Type
	versionReferencedCondition	Version referenced condition	The list of condition information which are version referenced by an identifier in payload publications.	0..*	VersionedReference

C.4.2 "Classes" package for “ExchangeInformation”

MessageContainer/ExchangeInformation/Classes

C.4.2.1 "Classes" package classes

Table C.4 — Classes of the "Classes" package

Class name	Designation	Definition	Abstract
ExchangeInformation	Exchange information	Exchange information.	no
DynamicInformation	Dynamic information	Dynamic exchange information.	no
ExchangeContext	Exchange context	Exchange context.	no
SessionInformation	Session information	Session information	no

C.4.2.2 Associations of "Classes" package

Table C.5 — Associations of the "Classes" package

Class name	Association end	Designation	Definition	Multiplicity	Target
ExchangeInformation	exchangeContext	Exchange context		0..1	ExchangeContext
	dynamicInformation	Dynamic information		1..1	DynamicInformation
DynamicInformation	sessionInformation	Session information		0..1	SessionInformation
ExchangeContext	supplier	Supplier	Defines the supplier of information of the exchange.	1..1	InternationalIdentifier
	client	Client	Client information, depending on exchange pattern that may be stated for single or multiple clients or not stated.	0..*	InternationalIdentifier

C.4.2.3 "Classes" package attributes

Table C.6 — Attributes of the "Classes" package

Class name	Attribute name	Designation	Definition	Multiplicity	Type
DynamicInformation	exchangeReturnStatus	Exchange return status	Exchange return status defined by protocol specification.	0..1	ExchangeStatusEnum
	exchangeStatusDescription	Exchange return status description	Multilingual textual description of exchange return status defined by protocol specification.	0..1	MultilingualString
	exchangeStatus	Exchange status	Status of exchange defined by protocol specification.	0..1	ExchangeStatusEnum
	exchangeStatusDescription	Exchange status description	Multilingual textual description of exchange status defined by protocol specification.	0..1	MultilingualString
ExchangeContext	protocolType	Protocol type	The exchange protocol type as referenced by any standard or agreement among a client and a supplier.	1..1	String
	protocolVersion	Protocol version	The version of the protocol used for the exchange as by standard or as agreed among a client and a supplier.	1..1	String
SessionInformation	sessionID	Session ID	The ID of the session established among a client and a supplier.	1..1	String

C.4.3 "Classes" package for “InformationManagement”

MessageContainer/InformationManagement/Classes

C.4.3.1 "Classes" package classes

Table C.7 — Classes of the "Classes" package

Class name	Designation	Definition	Abstract
InformationManagement	Information management	Information management hook data.	no
ElementReference	Element reference	Element reference.	no
InformationManagedResourceList	Information managed resource list	Managed resource list.	no

C.4.3.2 Associations of "Classes" package

Table C.8 — Associations of the "Classes" package

Class name	Association end	Designation	Definition	Multiplicity	Target
InformationManagement	informationManagedResourceList	Information managed resource list		0..1	InformationManagedResourceList
InformationManagedResourceList	elementReference	Element reference		1..*	ElementReference

C.4.3.3 "Classes" package attributes

Table C.9 — Attributes of the "Classes" package

Class name	Attribute name	Designation	Definition	Multiplicity	Type
ElementReference	managementStatus	Management status	It identifies the status of the element referenced.	1..1	Management-TypeEnum
	reference	Reference	It identifies the reference of the data exchange.	0..1	Reference
	versionedReference	Versioned reference	It identifies the versioned reference of the data exchange.	0..1	VersionedReference

C.4.4 "MessageContainer" package

MessageContainer

C.4.4.1 "MessageContainer" package classes

Table C.10 — Classes of the "Container" package

Class name	Designation	Definition	Stereotype	Abstract
MessageContainer	Message container	A Container class to manage further information classes as payload, information management, CIS information and exchange information	D2ModelRoot	no

C.4.4.2 Associations of "MessageContainer" package

Table C.11 — Associations of the "Container" package

Class name	Association end	Designation	Definition	Multiplicity	Target
MessageContainer	CISInformation	CIS information		0..1	CISInformation
	InformationManagement	Information management		0..1	InformationManagement
	ExchangeInformation	Exchange information		0..1	ExchangeInformation
	payload	Payload		0..*	PayloadPublication

C.4.4.3 "MessageContainer" package attributes

There are no defined attributes in the "MessageContainer" package.

C.4.5 External references

These classes are not part of the exchange packages as such but are generally provided through the payload content.

Table C.12 — External references

Class name	Designation	Definition
InternationalIdentifier	International identifier	
PayloadPublication	Payload publication	
Payload	Payload	

C.5 Data Dictionary of data types for "ExchangeDataModel"

C.5.1 Overview

There are no specific data types defined in the "ExchangeDataModel". However, different attributes are used and are on generic data types which are in principle defined in the content packages. They often map to intrinsic datatypes of PSM.

C.5.2 The data type "DateTime"

A combination of integer-valued year, month, day, hour, minute properties, a decimal-valued second property and a time zone property from which it is possible to determine the local time, the equivalent UTC time and the time zone offset from UTC.

C.5.3 The data type "MultilingualString"

A multilingual string, whereby the same text may be expressed in more than one language. If the data type is not implemented as such in the targeted PSM it may mapped to a String.

C.5.4 The data type "Reference"

A reference to an identifiable managed object where the identifier is unique. It comprises an identifier (e.g. GUID) and a string identifying the class of the referenced object.

C.5.5 The data type "String"

A character string whose value space is the set of finite-length sequences of characters. Every character has a corresponding Universal Character Set code point (as defined in ISO/IEC 10646), which is an integer.

C.5.6 The data type "VersionedReference"

A reference to an identifiable version managed object where the combination of the identifier and version is unique. It comprises an identifier (e.g. GUID), a version (NonNegativeInteger) and a string identifying the class of the referenced object.

C.6 Data Dictionary of enumerations for "ExchangeDataModel"

This clause contains the definitions of all enumerations used in the "ExchangeDataModel".

C.6.1 The "ExchangeReturnStatusEnum" enumeration

Enumeration that identifies the return status of exchange.

Table C.13 — Values contained in the enumeration "ExchangeStatusEnum"

Enumerated value name	Designation	Definition
ack	Ack	Request fulfilled.
closeSessionRequest	Close session request	Client request to close session.
fail	Fail	Request failed.
snapshotSynchronisationre-quest	Snapshot synchronisation request	Client request for snapshot syn-chronisation.

C.6.2 The "ExchangeStatusEnum" enumeration

Enumeration that identifies the status of exchange session.

Table C.14 — Values contained in the enumeration "ExchangeStatusEnum"

Enumerated value name	Designation	Definition
closingSession	Closing session	The closure of session is under negotiation for protocol with session management.
offline	Offline	No exchange is possible due to a lack of connectivity.
online	Online	Exchange connection is regular with no errors detected.
openingSession	Opening session	Session opening for protocols with session management.
resuming	Resuming	Some errors had happened and the session is resuming for protocol with session management.

C.6.3 The "ManagementTypeEnum" enumeration

Enumeration of status information.

Table C.15 — Values contained in the enumeration "ManagementTypeEnum"

Enumerated value name	Designation	Definition
cancelled	Cancelled	The information has been can-celled.
closed	Closed	The information has been closed.

C.6.4 The "PredefinedServiceEnum" enumeration

List of predefined service requests.

Table C.16 — Values contained in the enumeration "PredefinedServiceEnum"

Enumerated value name	Designation	Definition
broadcastDelivery	Broadcast delivery	Service delivery broadcast.
other	Other	Other service.
tmpActivation	TMP activation	Service TMP activation.

Table C.16 *(continued)*

Enumerated value name	Designation	Definition
vmsMessageProcessing	VMS message processing	Activation of VMS message processing on information.

C.6.5 The "ServiceActionEnum" enumeration

The current or requested status of TMP activation during request, implementation and termination phases.

Table C.17 — Values contained in the enumeration "ServiceActionEnum"

Enumerated value name	Designation	Definition
agreement	Agreement	Request for CIS agreement.
cancellation	Cancellation	Request for cancelling the CIS agreement or implementation request before agreement or implementation phase is completed.
implementation	Implementation	The request of CIS implementation, whenever no need of agreement proposal is needed, or a previous proposal have been agreed.
processing	Processing	Request to process information according to Requested Service (e.g. Processing of any information to generate messages to display on VMS).
statusInformation	Status information	Request on previously delivered service request processing status information, to be delivered in feedback.
termination	Termination	Request to terminate the CIS activation once implemented.

C.6.6 The "ServiceActionStatusEnum" enumeration

Defines values of service status.

Table C.18 — Values contained in the enumeration "ServiceActionStatusEnum"

Enumerated value name	Designation	Definition
compliant	Compliant	The service request has been found compliant to the required specification for the service.
failed	Failed	The service request failed during processing.
notCompliant	Not compliant	The service request has not been found compliant to the required specification for the service.
processing	Processing	The service request processing phase is not completed yet.
rejected	Rejected	The service request has been rejected by the service provider.

Table C.18 *(continued)*

Enumerated value name	Designation	Definition
scheduled	Scheduled	The service request had been scheduled for processing by the service provider.
success	Success	The service request had been successfully processed by the service provider.
timedOut	Timed out	The service request had not been fulfilled in the agreed time.