Purpose of this Reading Guide

This Reading Guide should allow readers of the Digital Container Shipping Association (DCSA) Interface Standard for Track and Trace to understand what the context of the DCSA Interface Standard for Track and Trace 1.0 is, what you can expect from it, and what you can achieve with it.

What is the context of the DCSA Interface Standard for Track and Trace 1.0?

What can you expect from the DCSA Interface Standard for Track and Trace 1.0?

What can you achieve with the DCSA Interface Standard for Track and Trace 1.0?
What is the context of the DCSA Interface Standard for Track and Trace 1.0?
Digital Container Shipping Association (DCSA)

The Interface Standard for Track and Trace results from a project of the DCSA.

VISION

The vision of the DCSA is to pave the way for interoperability in the container shipping industry through digitization and standardization. It is the DCSA’s mission to represent, lead and serve the container shipping industry for safer, more secure and efficient operations of container shipping companies. The project track of the DCSA Interface Standard for Track and Trace 1.0 in particular aims at increasing the level of common standards and at designing a common language for processes, events, and messages.

MEMBERS

The DCSA has the following members: CMA-CGM, Evergreen, Hapag-Lloyd, HMM, MSC, Maersk, ONE, Yang Ming and ZIM.
Purpose of the DCSA Interface Standard

A technology-agnostic interface standard was developed to set the stage for facilitation of standardization and digitization throughout the industry.

1. Standards support a common view across the industry in relation to processes, events and messages, facilitating industry standardization and digitization efforts. They are about the definition and alignment of terms, entities and attributes, and they are designed to support a common shared understanding of concepts, terms and rules of the business.

2. Additionally, a clearly defined DCSA Interface Standard for Track and Trace 1.0 is the foundation on which future DCSA standards will be defined and developed such as the project tracks of IoT, blockchain and cybersecurity. Per se, it is a first publication and thus subject to regular updates. Hence, it can serve as a baseline for industry participants to initiate the required steps towards the next level of container shipping.
What can you expect from the DSCA Interface Standard for Track and Trace 1.0?
Approach to the DCSA Interface Standard for Track and Trace 1.0

Any published standard will be made publicly available on the DCSA website. During the development of standards, it will likewise be possible for interested parties to obtain selected materials and to learn about upcoming standard publication(s) via the website DCSA.org.

Review of business perspectives

- **DCSA Industry Blueprint**
- **Existing Standards**
- **Use-case Description & Baseline, Requirements document**
- **Carrier SME Expertise**

DCSA Interface Standard for Track and Trace 1.0

- **Industry blueprint**
  - Alignment of processes and business terms
- **Existing standards**
  - Assessment around reuse, enhancement or rework of existing standards
- **Use-case & Baseline requirements**
  - Setting the scope for standardization
- **Carrier SME expertise**
  - Industry knowledge is key to understanding the domain

DCSA Industry Blueprint

Industry blueprint

Existing Standards

Use-case Description & Baseline, Requirements document

Carrier SME Expertise

Industry knowledge is key to understanding the domain
Scope of the DCSA Interface Standard for Track and Trace 1.0

The DCSA Interface Standard for Track and Trace 1.0 is complemented by this Reading Guide, the DCSA Information Model 1.0, the DCSA Glossary of Terms 1.1, OpenAPI definitions and the document on the DCSA Event Naming Convention 1.0 and DCSA Event Structure Definitions 1.0 are provided.

DCSA Interface Standard for Track and Trace 1.0
The objective of the DCSA Interface Standard for Track and Trace 1.0 is to standardize the fundamental information provided across the industry through track and trace interfaces. Following the DCSA Information Model, this DCSA Interface Standard for Track and Trace 1.0 assumes a limited scope with a one-to-one relationship between shipment and bill of lading.

DCSA Information Model 1.0
The DCSA Information Model 1.0 provides a holistic overview of the information standardizations defined in the DCSA Industry Blueprint 1.0. The DCSA Information Model 1.0 assumes a limited scope with a one-to-one relationship between shipment and bill of lading.

DCSA Glossary of Terms 1.1
The glossary is used to support the reader with definitions and explanations of the business terms used in the documents. The primary function of the glossary is to make sure that all readers are interpreting the terms in the same way.

OpenAPI definitions
OpenAPI definitions following the DCSA Information Model 1.0 and in particular the DCSA Interface Standard for Track and Trace 1.0 will be published on DCSA.org and DCSA-org SwaggerHub.

DCSA Event Naming Convention 1.0 and Event Structure Definitions 1.0
To align terminology across the industry, the DCSA has developed a naming convention, which sets the standard for naming as well as understanding of customer facing track & trace events.
Elements of the Interface for Track and Trace 1.0

The DCSA Interface Standard for Track and Trace 1.0 document provides standardized key UML diagrams and lists of inputs and outputs as depicted below. The symbols used are explained in detail in the appendix of this reading guide.

**Activity diagram**

The purpose of the activity diagram is to capture dynamic behavior in the system to demonstrate a message flow. The activity begins when a user requests track and trace details in relation to a shipment.

**Use-case diagram**

The use-case supports the requirement of an actor, who is in possession of a relevant identifier, to be able to request track and trace information for the shipment attached to the identifier and receive the available track and trace information for the shipment in return.

**List of inputs and outputs**

Booking reference, bill of lading number or equipment reference individually are identifiers that link to a shipment. At least one of these identifiers must be provided to constitute a valid request. The interface output is built around the events that occur for a shipment. This means that every shipment contains multiple events.

**Class diagram**

The class diagram provides an overview of all entities of the DCSA Information Model 1.0 and how they work together to support the functionality around the tracking and tracing of shipment. Following the DCSA Information Model 1.0, this DCSA Interface Standard for Track and Trace 1.0 assumes a limited scope for the first publication with a one-to-one relationship between shipment and bill of lading.
What can you achieve with the DCSA Interface Standard for Track and Trace 1.0?
What to achieve with DCSA Interface Standard for Track and Trace 1.0

The DCSA Interface Standard for Track and Trace 1.0 aims at standardizing the communication between data consumer, and data provider and it is agnostic towards the messaging media.

- Be consistent and aligned in the usage of DCSA terminology in calls and emails
- Update to latest UN/CEFACT EDI version, e.g. IFSTA D19A
- Refer to Appendix II for more details
- Mapping to existing standards
- Enhance UX on track and trace portals
- Align data elements to be exchanged through an API
- Build on top of OpenAPI definitions
Contribute

The DCSA Interface Standard for Track and Trace 1.0 will continue to be expanded with ever more data elements as we standardize the inter-operational aspects of the container shipping industry. This will be done based on ongoing collaboration with the industry.

Creation process

The DCSA Interface Standard for Track and Trace 1.0 has been made in collaboration with some of the world’s largest shipping companies. The collection and consolidation of interface documentation was carried out by the DSCA. The DCSA Interface Standard for Track and Trace 1.0 aims at creating a representation of processes across all carriers.

Suggested improvements

The DCSA Interface Standard for Track and Trace 1.0 will be a constantly evolving entity, which will change as processes and best practice across the industry change.

For this reason, the DCSA is always interested in feedback, which can increase the quality of published work and drive standardization and digitalization going forward.

If you have any feedback or inputs, go to our webpage under “Contact”.

Thomas Bagge
CEO, DCSA
thomas@dcsta.org

Henning Schleyerbach
COO, DCSA
henning@dcsta.org

Follow us on LinkedIn
www.dcsa.org
info@dcsta.org
@DCSA_ORG
Appendix I

Legend on the DCSA Interface Standard for Track and Trace 1.0

Use-case diagram

**Actor**
An actor is a user of the system. A user can refer to many entities, such as a human being but also a machine or another (sub-) system. In our example, the actors represent the stakeholders.

**Association**
An association is used to indicate a relationship between two elements.

**Use-case**
A use-case is an element in UML modeling used to describe how a user of a system interacts with the system to perform a task. In our example the use-case is “tracking and tracing of shipment”.

---

**Shipper**

**Consignee**

**Tracking and tracing of shipment**

**Carrier**
Legend on the DCSA Interface Standard for Track and Trace 1.0

Activity diagram

**Decision**
Decision element is used to highlight a condition: if a condition holds true, then processing continues one way. It is marked in green color in this example.

**Activity**
An Activity reflects the data flow of a process and specifies a sequence of behavior. An activity is shown as a round-cornered rectangle enclosing all the actions, control flows and other elements that make up the activity.

**Flow final**
The flow final node is depicted as a circle with a cross inside. The flow final node denotes the end of a single control flow.

**Initial**
An initial or start node is depicted by a large black spot.
Appendix I
Legend on the DCSA Interface Standard for Track and Trace 1.0

Class diagram

Class
A class is an element that defines the attributes and behaviors that an object can generate. Classes are represented by rectangles depicting the class’ name and – if applicable - the name of the operations and attributes.

Association
Association describes a relationship between two classes. It is denoted by a straight line where an arrow represents the direction in which the information is being used.

Composition
Composition is a special form of association where one class object owns the other class object and where the child object cannot exist on its own. It is denoted by a filled diamond-shaped arrowhead pointing towards the target or parent class.

Generalization
A generalization indicates inheritance. They are drawn from a specific classifier to a general classifier. This means that the source inherits the characteristics of the target.

Aggregation
Composition is a special form of association where one class object owns the other class object and where the child object can exist on its own. It is denoted by a hollow diamond-shaped arrowhead pointing towards the target or parent class.
Appendix II – Implementing an interface over EDI

EDI is currently the most dominant technology when it comes to the tracking and tracing of a shipment. Further details are provided below.

- EDI is implemented in an asynchronous manner, whereas the DCSA Interface Standard for Track and Trace 1.0 follows a synchronous model.
- EDI is still an ideal choice for existing message exchanges for tracking and tracing, that follow the asynchronous push model. Please refer to Appendix III for an explanation of the push and pull models.
- If the DCSA Interface Standard for Track and Trace 1.0 were to be implemented over EDI, it would require an exchange of 4 separate EDI messages, as shown in the image below.
- It is not considered optimal to implement a synchronous interface over EDI.
- For asynchronous communication or publishing of events, the latest UN/EDIFACT IFTSTA or ANSI standard is to be used.

Asynchronous communication over EDI

Requester

Tracking and tracing

Input interface

Tracking and tracing

Output interface

Request

Response

Acknowledge

Acknowledge
Appendix III - Push model (asynchronous)

What impact does pushing or publishing information have on the underlying technology and the manner of communication?

- Push model is event based
- Requester is responsible for subscribing to information once
- Provider is responsible for pushing information
- Requester never expects or waits for a response
- Response generally does not reflect the real-time state as it depends on how frequently events are published. As a result, current state might differ from the time that an event was published.
- Is typically implemented using an asynchronous technology like webhooks, callback APIs, Publish/Subscribe message queues, EDI
- Current DCSA Interface Standard for Track and Trace 1.0 does not support this model
- Current communication over EDI follows this model
- Future standardization could be needed for API callbacks/webhooks as this model offers better scalability than the pull model
Appendix III - Pull model (synchronous)

What impact does pulling information have on the underlying technology and the manner of communication?

- Pull model is request based
- Requester is responsible for pulling information
- Requester always expects a response
- Response provides the real-time state
- Pull model is typically implemented using a synchronous technology like web APIs
- The DCSA Interface Standard for Track and Trace 1.0 supports this model
Appendix IV - Sample responses for API endpoints

Please note: These are sample responses for possible API endpoints. The actual DCSA OpenAPI specifications can be found on DCSA SwaggerHub.

GET /shipments/{bookingReference}
Possible Sample Response in JSON

```
{
  bookingReference: "YCH698840",
  billOfLadingNumber: "855230148",
  bookingDateTime: "2019-11-12T07:41:00+08:30",
  carrierCode: "MAEU",
  shipperName: "Computer Corp.",
  consigneeName: "Computer Corp.",
  origin: "USNYC",
  destination: "GBMNC",
  actualDeliveryDateTime: "2019-11-12T07:41:00+08:30",
  eventsLink: "/shipments/YCH698840/events"
}
```

GET /shipments/{bookingReference}/events
Possible Sample Response in JSON

```
{
  bookingReference: "YCH698840",
  shipmentLink: "/shipments/YCH698840"
  events: [
    {
      eventID: 57254,
      bookingReference: "YCH698840",
      eventType: "ARI",
      eventStatus: "ACT",
      eventDateTime: "2019-11-12T07:41:00+08:30",
      transportReference: "5",
      transportLink: "/transports/5"
      equipmentReference: "ZCSU2785124",
      equipmentLink: "/equipments/ZCSU2785124",
      location: "USNYC",
      sublocationType: "DE",
      sublocation: "DEHAMCTA"
    }, ...
  ]
}
```
Appendix V - Approach to error messages for APIs

- REST APIs are typically designed around HTTP response codes for both valid responses and error responses.

- The most relevant response codes are considered to be:
  - 200 OK
  - 201 Created
  - 202 Accepted
  - 304 Not Modified (In relation to caching)

- The most relevant error codes are considered to be:
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 405 Method Not Allowed
  - 500 Internal Server Error
Appendix VI – Approach to versioning for APIs

- API specifications should follow the semantic versioning standard (http://semver.org)
- Semantic versioning standard defines a version in the format, MAJOR.MINOR.PATCH
- MAJOR version is incremented when incompatible API changes are made
- MINOR version is incremented when functionality is added in a backwards compatible manner
- PATCH version is incremented when backwards compatible bug fixes are made.
- API implementation should adhere to a certain version of the API specification
Appendix VII – Recommendations on API security

**Authorization**

- Determining what the user has access to.
- It is recommended to implement authorization by issuing an access token based on the OAuth 2.0 standard.
- Access token is then used to authorize individual API requests, typically attached in the authorization header of the HTTP request as a bearer token. More about bearer token usage can be found in RFC 6750.
- Read more about OAuth 2.0 [here](#) and in RFC 6749.

**Authentication**

- Verifying the identity of the user.
- It is recommended to implement authentication by issuing an ID token based on the OpenID Connect standard.
- ID token is issued as a JSON Web Token (JWT). More about the JSON structure of the token can be found in RFC 7519.
- OpenID Connect is part of the OAuth 2.0 framework.
- Read more about OpenID Connect [here](#).
Legal disclaimer

Copyright 2020 Digital Container Shipping Association (DCSA)

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License here: License

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.