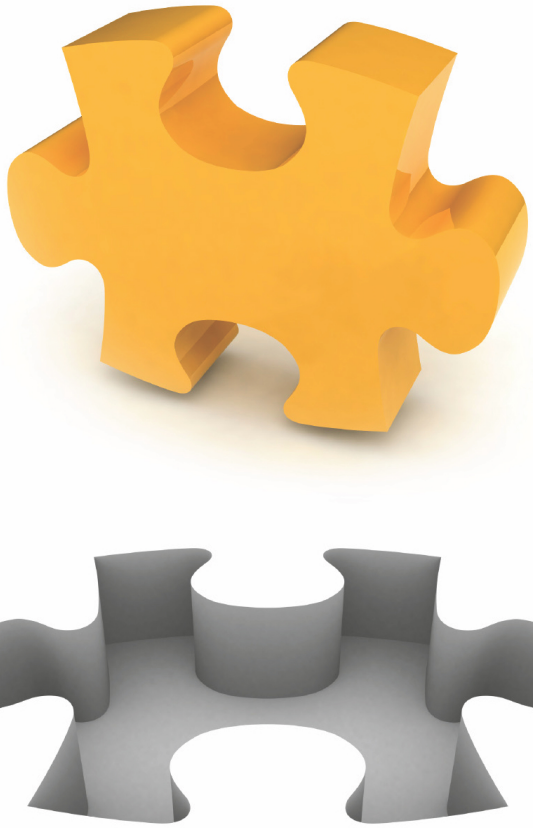


CERTIFICATION DATABASE INTERFACE V1.1

International Gaming Standards Association

S2S Technical Committee

Released: 2020/03/23

IGSA.ORG

Certification Database Interface v1.1

Released 2020/03/23 by International Gaming Standards Association® (IGSA®):

v1.0 released 2017/03/22.

v1.0.1 released 2017/12/14.

Patents and Intellectual Property

NOTE: The user's attention is called to the possibility that compliance with this [standard/specification] may require use of an invention covered by patent rights. By publication of this [standard/specification], IGSA takes no position with respect to the validity of any such patent rights or their impact on this [standard/specification].

Similarly, IGSA takes no position with respect to the terms or conditions under which such rights may be made available from the holder of any such rights. Contact IGSA for further information.

Trademarks and Copyright

IGSA Contact Information

E-mail: sec@igsa.org

WWW: http://www.igsa.org

Table of Contents

I About This Document

This document describes the Certification Database Interface. The Certification Database Interface is designed to provide an easy and straightforward method, which is based on HTTP and JSON, for systems to exchange information associated with the submission and approval of gaming equipment and software.

I.I Acknowledgements

The International Gaming Standards Association would like to express its appreciation to all members of the S2S committee, past and present, for their contribution and dedication to the creation of this document.

I.II Document Conventions and Organization

I.II.I Indicating Requirements, Recommendations, and Options

Terms and phrases in this document that indicate requirements, recommendations, and options in the CDI Protocol are used as defined in the IETF RFC 2119.

In summary:

Requirements:

To indicate requirements, this document uses "MUST", "MUST NOT", "REQUIRED", "SHALL", or "SHALL NOT".

Recommendations:

To indicate recommendations, this document uses "SHOULD", "SHOULD NOT", "RECOMMENDED".

Options:

To indicate **options**, this document uses "MAY" or "OPTIONAL".

I.II.II Other Formatting Conventions

- Blue text indicates an internal link or external hyperlink to a URL.
- **Bold** (other than in headings) or underlined text is used for emphasis, unless specifically indicated otherwise.
- *Italicized* text (other than in headings) is used for terms being defined, unless within an XML label (typically indicates a type of element appearing in multiple classes, such as `getClassState`) or unless specifically indicated otherwise.
- `Courier New` font is used to indicate XML components and their values, and other types of code or pseudo code.

Chapter 1

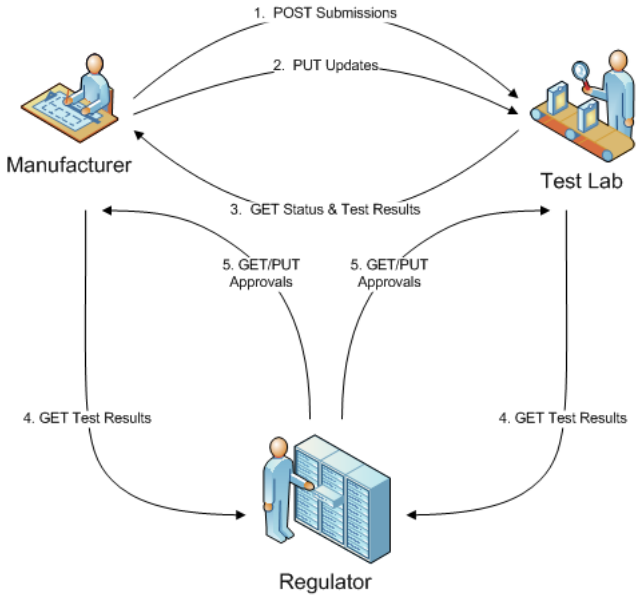
Introduction

1.1 Overview

This document describes the Certification Database Interface (CDI). The Certification Database Interface can be utilized to access the certification databases of testing laboratories, vendors, and regulatory agencies. It is intended to be used by testing laboratories, vendors, and regulators to exchange information related to the testing and approval of gaming products including EGM hardware, EGM software, and iGaming software.

With the Certification Database Interface, a Client System uses the HTTP protocol and HTTP verbs — such as GET, POST, PUT, and DELETE — to access resources on a Host System. The data exchanged between systems is encoded using JSON. The resources available through the Certification Database Interface are described in subsequent chapters of this specification. The following diagram illustrates the intended interactions.

Figure 1.1 System Interactions



The process begins when a vendor submits a product to a test laboratory. The information about the product and the certification request is conveyed to the test laboratory using an HTTP POST operation. Subsequently, additional certification requests can be added to the submission using an HTTP PUT operation. The vendor can request status updates about the submission from the test laboratory using an HTTP GET operation.

Once a submission has been accepted by the test laboratory, the vendor can request information about the testing and certification of the product using an HTTP GET operation. The vendor can request information about a specific product or the vendor can request information about all products that have had status changes during a specific time period.

Similarly, once a submission has been accepted by the test laboratory, the regulator can also request information about the testing and certification of the product using an HTTP GET operation. Like the vendor, the regulator can request information about a specific product or the regulator can request information about all products that have had status changes during a specific time period.

After a product has been approved by the regulator, the vendor or test laboratory can request information about the approval from the regulator using an HTTP GET operation. The approval information can be requested for a specific product or for all products with status changes during a specific time period.

Additionally, the regulator can send updates to the approval information to the vendor or test laboratory using an HTTP PUT operation.

1.1.1 Shipments

Extension in v1.1

The Certification Database Interface can also be utilized to exchange information about the shipment of gaming products. The following diagram illustrates the intended interactions.

The process begins when a vendor or operator submits a shipping request to a regulator for approval. The information about the shipment is conveyed to the regulator in an HTTP POST operation. Many types of shipments are supported including vendor-to-operator, operator-to-vendor, operator-to-operator, and disposals. The vendor or operator can amend the request and resubmit it to the regulator using an HTTP PUT operation.

If the shipment request is approved by the regulator, the process continues with the vendor or operator submitting product receiving information to the regulator. The receiving information is conveyed to the regulator in an HTTP PUT operation.

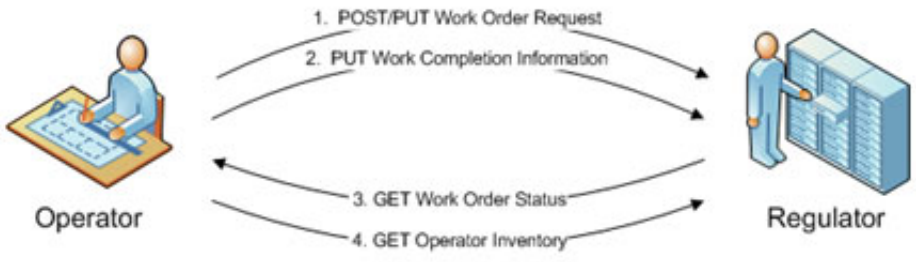
Subsequently, the vendor or operator can request a shipment status update from the regulator using an HTTP GET operation. The status indicates whether the shipping request was approved or denied by the regulator and whether the products have been received.

Additionally, a vendor or operator can request an operator's current inventory of products using an HTTP GET operation.

1.1.2 Work Orders

Extension in v1.1

The Certification Database Interface can also be utilized to exchange information about work orders which may be required when an operator perform certain actions with products – for example, moving a product to a new location, changing the configuration of a product, upgrading a product, etc. The following diagram illustrates the intended interactions.



The process begins when an operator submits a work order request to a regulator for approval. The information about the work order is conveyed to the regulator in an HTTP POST operation. Many types of work orders are supported including installations, removals, upgrades, reconfigurations, and relocations. The operator can amend the request and resubmit it to the regulator using an HTTP PUT operation.

If the work order request is approved by the regulator, the process continues with the operator submitting work completion information to the regulator. The work completion information is conveyed to the regulator in an HTTP PUT operation.

Subsequently, the operator can request a work order status update from the regulator using an HTTP GET operation. The status indicates whether the work order was approved or denied by the regulator and whether the work has been completed.

Additionally, an operator can request an operator's current inventory of products using an HTTP GET operation.

1.2 Terminology

1.2.1 Host & Client Systems

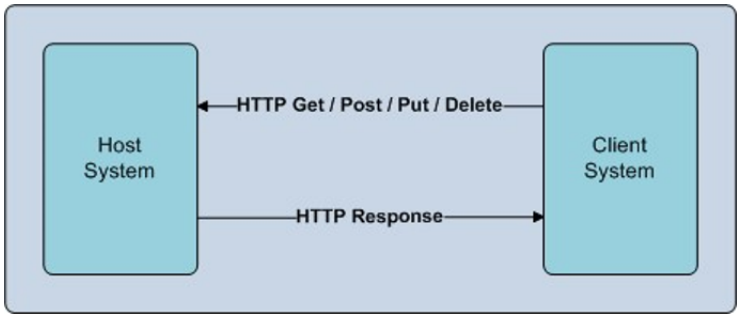
Within this specification, the following definitions are used when referring to various types of systems.

Table 1.1 Terminology

System	Description
Host System	A Host System is a system that provides resources to Client Systems.
Client System	A Client System is a system that accesses the resources of a Host System.

The following diagram illustrates the relationships between the systems.

Figure 1.2 System Relationships



Note that a particular endpoint – such as a testing laboratory, vendor, or regulator – may act as both a Host System and a Client System depending on which resources are being used. For example, a testing laboratory may act as a Host System when a vendor is delivering product submission information and retrieving test results. At the same time, the testing laboratory may act as a Client System when retrieving product approval information from regulators. The descriptions of individual resources identify the appropriate roles for each endpoint when using the resources.

1.2.2 Submissions

Within this specification, the term "submission" refers to the set of information exchanged between a vendor and a test laboratory to initiate the testing of a product. A submission consists of a number of elements: the submission record itself, one or more component records, one or more jurisdiction records for each component, zero or more software signature records for each component (software signatures may not be available for some components), and zero or more document records for each jurisdiction.

- Submission - The submission record (`certificationSub`) provides information about the overall product submission including the vendor's unique identifier for the submission, the unique identifier of the vendor, the unique identifier of the test laboratory, and descriptive information about the submission. The submission record may also include zero or more free-form notes (`note`) about the submission.
- Component - The component record (`componentSub`) describes a discreet unit of hardware or software that requires testing and approval. The record includes the vendor's unique identifier for the

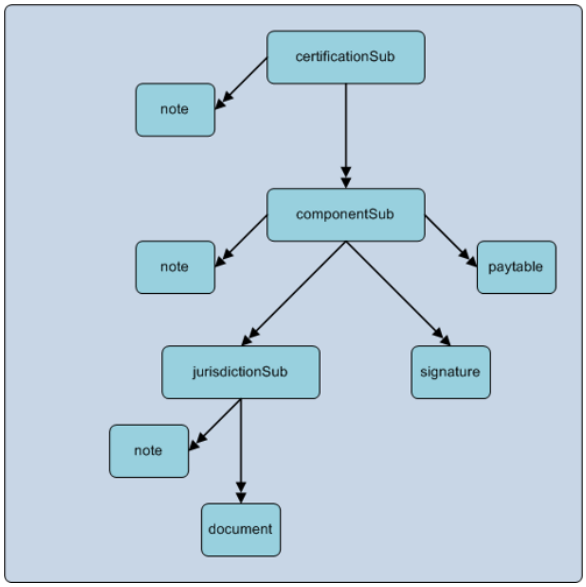
component, the common name of the component, the version number of the component, and other descriptive information about the component. The component record may also include zero or more free-form notes (`note`) about the component. In addition, for components that are games, information about zero or more paytables (`paytable`) may also be included.

- Jurisdiction - The jurisdiction record (`jurisdictionSub`) identifies a specific jurisdiction for which a component requires testing and approval. The record includes the jurisdiction's unique identifier, the submission package identifier for the jurisdiction, and the test laboratory's status for the submission. The status indicates whether the submission was accepted by the test laboratory, rejected, or is still pending. The jurisdiction record may also include zero or more notes (`note`) for the specific jurisdiction.
- Signature - The signature record (`signature`) contains a software signature for a component. The record identifies the algorithm used to generate the signature, any parameters – such as seed and salt values – used with the algorithm, and the software signature itself.
- Document - The document record (`document`) identifies a specific document relevant to the submission for a jurisdiction – for example, product documentation, par sheets, etc. The record includes the unique identifier for the document, a brief description of the document, the language in which the document is written, and a URL pointing to the document.

The following diagram illustrates the relationships between these elements.

Figure 1.3

Figure 1.4 Submission Record Relationships



The resources described in Chapter 3 Submission Resources are used to make submissions to test laboratories and, subsequently, to request status information about the submissions from test laboratories.

1.2.3 Certifications

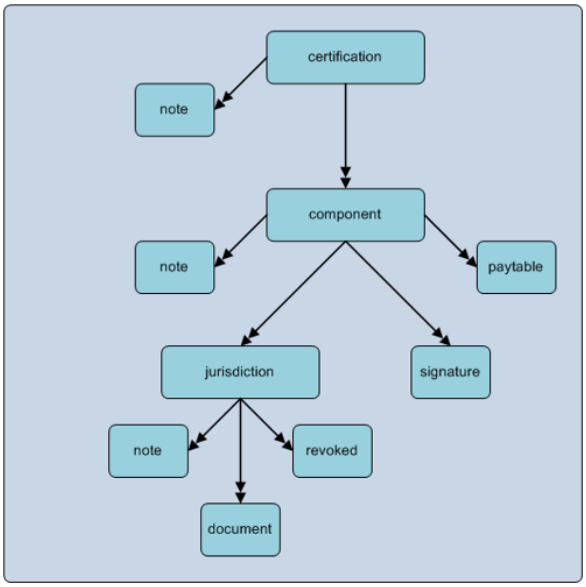
Within this specification, the term “certification” refers to the set of information reported by a test laboratory about the testing of the components within a submission. Only submissions that have been accepted by the test laboratory are visible as certifications. Like a submission, a certification consists of a number of elements:

the certification record itself, one or more component records, one or more jurisdiction records for each component, zero or more software signature records for each component, and zero or more document records for each jurisdiction.

- Certification - The certification record (`certification`) is very similar to the submission record described above for submissions but contains a slightly different set of data that is focused on the certification rather than the submission. It provides general information about the submission including the vendor's unique identifier for the submission, the unique identifier of the vendor, the unique identifier of the test laboratory, and descriptive information about the submission. It may also include free-form notes (`note`) about the certification.
- Component - The component record (`component`) for certifications is very similar to the component record described above for submissions but contains a slightly different set of data that is focused on the certification rather than the submission. It describes a discreet unit of hardware or software that requires testing and approval. The record includes the vendor's unique identifier for the component, the common name of the component, the version number of the component, and other descriptive information about the component. It may also include free-form notes (`note`) about the component and, when the component is a game, information about paytables (`paytable`) associated with the component.
- Jurisdiction - The jurisdiction record (`jurisdiction`) for certifications is very similar to the jurisdiction record described above for submissions but contains a slightly different set of data that is focused on the certification rather than the submission. It contains the testing and approval status of a component for a specific jurisdiction. It includes the jurisdiction's unique identifier, the test laboratory's status for the component, and the regulator's status for the component. The test laboratory's status identifies the state of the component in the testing and approval process from the test laboratory's perspective. The regulator's status identifies the state of the component from the regulator's perspective. The jurisdiction record may also include free-form notes (`note`) for the jurisdiction as well as a list of components revoked by the certified component (`revoked`).
- Signature - The signature record (signature) contains a software signature for a component. The record identifies the algorithm used to generate the signature, any parameters — such as seed and salt values — used with the algorithm, and the software signature itself. The list of signatures may or may not contain the same list of signatures that were included in the submission. The test laboratory may generate its own signatures using special salt or seed values supplied by a regulator. In such cases, the test laboratory may filter the list of signatures, providing one set of signatures to vendors and another set of signatures to regulators.
- Document - The document record (`document`) identifies a specific document associated with the certification for a jurisdiction – for example, product documentation, par sheets, certification reports, etc. The record includes a unique identifier for the document, a brief description of the document, the language in which the document is written, and a URL pointing to the document.

The following diagram illustrates the relationships between these elements.

Figure 1.5 Certification Record Relationships



The resources described in Chapter 4 Certification Resources are used to request information from test laboratories and vendors about certifications.

1.2.4 Approvals

Within this specification, the term "approval" refers to the set of information reported by a regulator about the suitability of a component for use within a jurisdiction. Like a submission or certification, an approval consists of a number of elements: the approval record itself, one or more component records, one or more jurisdiction records for each component, zero or more software signature records for each component, and zero or more document records for each jurisdiction.

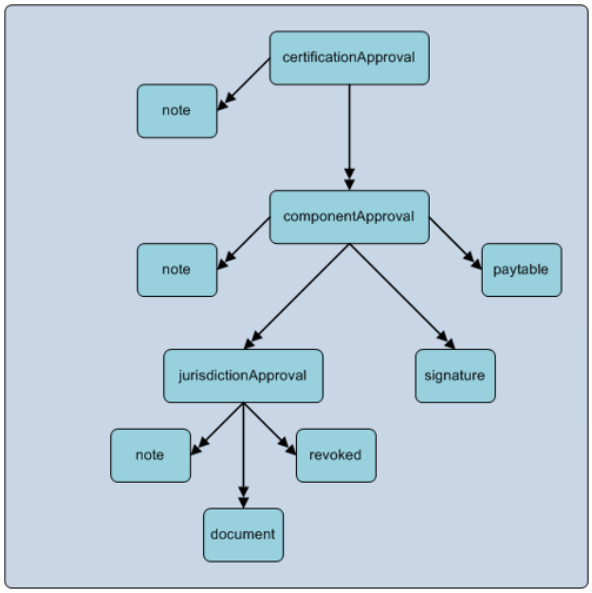
- Approval - The approval record (`certificationApproval`) is very similar to the submission and certification records described above for certifications but contains a slightly different set of data that is focused on the approval rather than the certification. It provides general information about the submission including the vendor's unique identifier for the submission, the unique identifier of the vendor, and the unique identifier of the test laboratory. It may also include free-form notes (`note`) about the approval.
- Component - The component record (`componentApproval`) is very similar to the component record described above for certifications but contains a slightly different set of data that is focused on the approval rather than the certification. It describes a discreet unit of hardware or software that requires testing and approval. The record includes the vendor's unique identifier for the component, the common name of the component, the version number of the component, and other descriptive information about the component. It may also include free-form notes (`note`) about the component and, when the component is a game, information about paytables (`paytable`) associated with the component.
- Jurisdiction - The jurisdiction record (`jurisdictionApproval`) is very similar to the jurisdiction record described above for certifications but contains a slightly different set of data that is focused on the approval rather than the certification. It contains the approval status of a component for a specific jurisdiction. It includes the jurisdiction's unique identifier and the regulator's status for the component. The regulator's status indicates whether the component has been approved for use with

the jurisdiction. The jurisdiction record may also include free-form notes (`note`) for the jurisdiction as well as a list of components revoked by the certified component (`revoked`).

- Signature - The signature record (`signature`) contains a software signature for a component. The record identifies the algorithm used to generate the signature, any parameters — such as seed and salt values — used with the algorithm, and the software signature itself. The list of signatures may or may not contain the same list of signatures that were included in the submission or certification. The regulator may generate its own signatures using special salt or seed values. In such cases, the regulator may filter the list of signatures, only providing the set of signatures that it has generated. Alternatively, the regulator may report back signatures generated by the vendors or the test lab.
- Document - The document record (`document`) identifies a specific document associated with the approval for a jurisdiction – for example, product documentation, par sheets, certification reports, approval letters, etc. The record includes the unique identifier for the document, a brief description of the document, the language in which the document is written, and a URL pointing to the document.

The following diagram illustrates the relationships between these elements.

Figure 1.6 Approval Record Relationships



The resources described in Chapter 5 Approval Resources are used to request information from regulators about approvals.

1.2.5 Shipments

Extension in v1.1

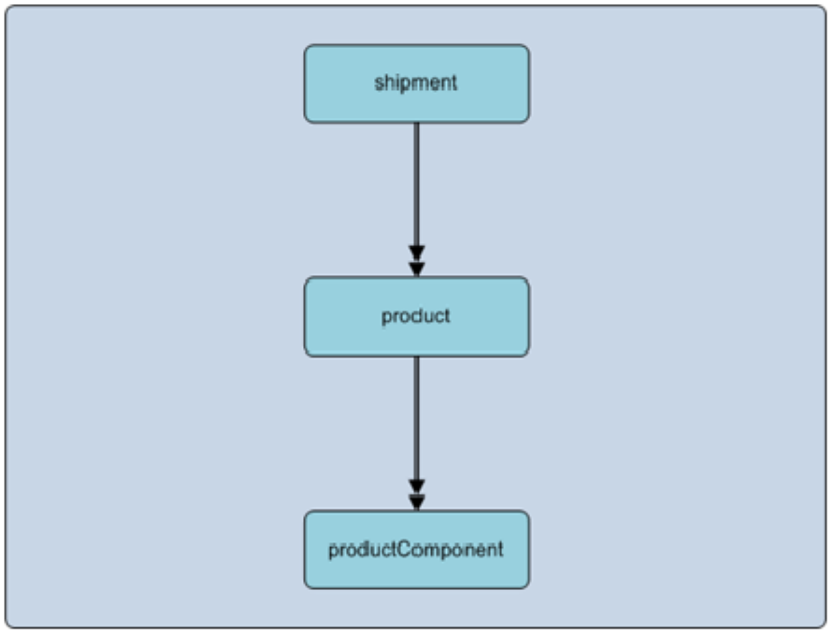
Within this specification, the term "shipment" refers to the set of information exchanged between vendors, operators, and regulators regarding the shipping of products to/from operators, as well as disposals of products by operators. A shipment consists of three elements: the shipment record, the product record, and the component record. There can be multiple product records per shipment and multiple component records per product.

- Shipment — The shipment record (`shipment`) provides information about the shipping request including the unique identifier for the shipping request, the unique identifier of the shipper, the unique

identifier of the receiver, and additional descriptive information about the shipping request. The shipment record may contain one or more product records.

- Product — The product record (`product`) identifies a specific product that is included in the shipment. It includes the unique identifier for the product, the serial number of the product, and other descriptive information about the product.
- Component — The component record (`productComponent`) identifies a specific component of the product. It includes the component identifier assigned during the submission and approval process, the common name of the component, the function of the component, and the version number of the component.

The following diagram illustrates the relationships between these elements.



The resources described in Chapter 7 Shipment Resources are used to submit shipping information to regulators, request approval information back from regulators, update the shipping information with product receiving information, and request current product inventory information for operators.

1.2.6 Work Orders

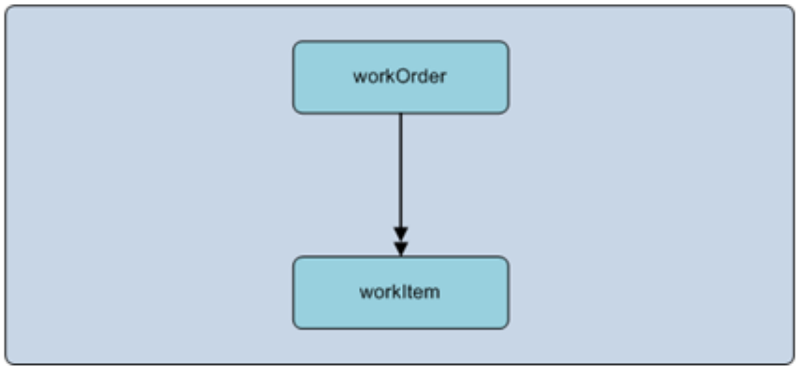
Extension in v1.1

Within this specification, the term "work order" refers to the set of information exchanged between operators and regulators regarding work performed on products by operators – for example, installations, removals, reconfigurations, upgrades, etc. A work order consists of two elements: the work order record and the work item record. There can be multiple work item records per work order.

- Work Order — The work order record (`workOrder`) provides information about the work order request including the unique identifier for the work order, the unique identifier of the operator, and additional descriptive information about the work order. The work order record may contain one or more work item records.

- Work Item — The work item record (`workItem`) identifies a specific product that is covered by the work order. It includes the unique identifier for the product, the type of work to be performed, and other descriptive information about the product and the work to be performed.

The following diagram illustrates the relationships between these elements.



The resources described in Chapter 8 Work Order Resources are used to submit work orders to regulators, request approval information back from regulators, and update the work order with completion information. The inventory resource described in Chapter 7 Shipment Resources is used to request current product inventory information.

1.3 Relevant Standards

The following standards are relevant to this specification. Host Systems and Client Systems MUST conform to these standards and their supporting standards.

Table 1.2 Relevant Standards

Standard	Description
HTTP	RFC 7230: Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing RFC 7231: Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content RFC 7235: Hypertext Transfer Protocol (HTTP/1.1): Authentication
JSON	RFC 7159: The JavaScript Object Notation (JSON) Data Interchange Format
TLS	RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2
URI	RFC 3986: Uniform Resource Identifier (URI): Generic Syntax
UUID	ISO/IEC 9834-8:2014: Generation of Universally Unique Identifiers

1.4 Base URI

The Base URI of the Host System is determined by the administrator of the Host System. The mechanism used by the Client System to discover the Base URI is beyond the scope of this specification. The Client System may use an administrative interface, a configuration file, or some other mechanism to determine the Base URI.

Host Systems and Client Systems MUST support the HTTP protocol using secure TLS communications. The "https" token is used to specify secure communications.

The following table contains an example of a fully-formed Base URI for a Host System.

Table 1.3 Base URI Example

Base URI Example
https://www.hostSystem.com:443

1.5 Resource URI

The Base URI is used to construct the Resource URIs of the individual resources offered by the Host System. Resource URIs are constructed by appending the pathname of the resource to the Base URI. The pathnames for individual resources are described later in this document.

The following table contains an example of a Resource URI for a Host System.

Table 1.4 Resource URI Examples

Resource URI Example
https://www.hostSystem.com:443/cdi/1.0/certification

If a Client System attempts to access a resource but the resource is not supported by the Host System, per the HTTP specification, the Host System should respond with HTTP status code 404 "Not Found".

Note that resource pathnames typically include a [ver] segment – for example, /cdi/[ver]/certification. The [ver] segment of the resource pathname MUST be replaced with the version number of the Certification Database Interface being used by the Client System. A dot-delimited format MUST be used for the version number. For example, version "1.0" must be represented as the character string "1.0". Superfluous leading and trailing characters MUST be removed. For example, version "v1.0" must be represented as "1.0".

1.6 HTTP Verbs

Client Systems MUST use the HTTP protocol to access the resources of the Host System. Different HTTP verbs – POST, GET, PUT, or DELETE – may be used to access different resources. The specific HTTP verbs that can be used with an individual resource are included in the description of the resource. The Client System SHOULD only use the HTTP verbs that are specified for a resource to access that resource.

If a Client System attempts to use an HTTP verb to access a resource and that HTTP verb is not supported by the Host System, per the HTTP specification, the Host System should respond with HTTP status code 405 "Method Not Allowed".

1.7 Persistent Connections

Host Systems and Client Systems MUST support HTTP persistent connections. Client Systems SHOULD specify persistent connections when establishing connections to Host systems. Client Systems and Host Systems SHOULD maintain persistent connections for at least five minutes unless the connection is no longer required at the application level.

The intent of this requirement is to avoid frequent use of "close" tokens within HTTP headers. The goal is to keep the HTTP session open between the Client System and the Host System so that TLS and TCP connections do not have to be recreated more often than necessary.

1.8 Minimum Message Size

Host Systems MUST be able to receive and process a 4-megabyte or smaller JSON-encoded message. Client Systems do not have a mandated maximum message size requirement but should be aware that messages, which are larger than 4 megabytes, may not be processed by Host Systems. This requirement is intended to promote interoperability by providing Host Systems and Client Systems with a known minimum limit on the size of messages.

If a Client System attempts to send a message that is too large for the Host System to process, per the HTTP specification, the Host System should respond with HTTP status code 413 "Payload Too Large".

1.9 HTTP Status Codes

Per the HTTP specification, each HTTP response from the Host System must include an HTTP status code. Only HTTP status codes, which are defined within the HTTP specification, should be used.

If the Host System successfully accepts a message, the Host System MUST include an HTTP status code from the 200 series in its response. This indicates that the Host System has successfully received, understood, and accepted the message and that the Client System does not have to report an exception, retry the message, etc.

If the Host System is unable to accept a message, the Host System MUST include an HTTP status code outside of the 200 series in its response. The HTTP status code should indicate as clearly as possible why the message was not processed.

Redirection — that is status codes from the 300 series — MAY be used by the Host System.

If one or more of the parameters of an HTTP GET operation are not supported by the Host System, the Host System MUST NOT simply ignore the parameters that are not supported. Instead, the Host System should respond with HTTP status code 501 "Not Implemented".

1.10 Case Sensitivity

Unless specified differently in an underlying specification, all protocol-defined constructs described within this specification are case-sensitive including object names, property names, enumeration values, code values, identifier values, data values, etc. Uppercase letters MUST NOT be considered equivalent to lowercase letters. For example, "ABC" must not be considered equivalent to "abc".

1.11 Resource Extensions

Implementations MAY extend the objects defined within this specification by adding new syntactically valid properties to the objects. The names of any such properties in an object MUST NOT conflict with the names of properties defined within this specification for that object. Implementations MUST expect to receive such properties within the objects defined within this specification. Implementations MUST NOT reject objects simply because they contain such properties.

1.12 Authentication

Authentication MUST be handled through Basic HTTP Authentication as defined by RFC 7235. In the token containing the username/password pair, the username MUST be in plain text; the password MUST be a SHA-1 hash of the plain-text password. The Authorization header MUST be included with each communication. The username/password pair MUST be hidden and secured within the required TLS connection. Per the HTTP specification, if any messages are received with a username or password that are not recognized a HTTP 401 Not Authorized status should be returned.

It is intended that the username provided by the Client System be used by the Host System to control access to information associated with specific testing laboratories, vendors, and jurisdictions. For example, a testing laboratory might use the username to limit a regulator's access to only product certifications for the regulator's jurisdiction. Similarly, a testing laboratory might use the username to limit a vendor's access to only product submissions from the vendor.

Basic HTTP Authentication MUST be used to access the resources described within this specification as well as the documents identified within document records (`document`) reported by those resources. The URLs specified in document records MUST include the "https" prefix, indicating that secure HTTP must be used to access the documents. The `documentSource` property of the document records indicates which end-point – supplier, test lab, or regulator – hosts the documents. The appropriate username/password pair for the end-point MUST be used when accessing the documents. That username/password pair may be different than the pair used to access the resource that reported the document records.

1.13 Metadata

The metadata resources described in Chapter 2 can be used by a Client System to determine the set of testing laboratories, vendors, and jurisdictions to which the username of the Client System has access. The metadata resources can also be used by the Client System to determine the set of status codes and software authentication algorithms used by the Host system.

Metadata is intended to be unique to a Host System. Thus, different Host Systems may use different identifiers to refer to the same entity within the metadata. For example, two different Host Systems may use two different identifiers to refer to the same testing laboratory. The use of metadata allows the different Host Systems to operate independently of one another, assigning unique identifiers to testing laboratories, vendors, jurisdictions, statuses, and algorithms as needed.

However, it also requires that Client Systems map the identifiers used by Host Systems to their own internal identifiers to provide semantic meaning to the Host System's metadata. For example, a Client System will have to map the identifier that the Host System uses to indicate that a product has been approved to the Client System's own internal identifier for the same status. Operator intervention may be required on an ongoing basis to create these mappings.

To help minimize the difficulty of managing the metadata mappings, Host Systems are required to assign universally unique identifiers (UUIDs) to metadata entities. Client Systems should maintain mappings of individual UUIDs to their own internal identifiers. UUIDs MUST be generated in a manner compliant with ISO/IEC 9834-8:2014 to guarantee uniqueness.

Client Systems SHOULD periodically refresh the metadata to assure that their metadata mappings are up to date and that they only make requests for data to which they have access.

1.13.1 Operator Metadata

Extension in v1.1

The metadata resources described in Chapter 2 can also be used by a Client System to determine the set of operators to which the Client System has access. Like other metadata, Client Systems SHOULD periodically refresh the operator metadata to assure that their mappings are up to date.

1.14 Record Identifiers

Each object within this specification contains a record identifier. For example, certification objects contain a `certificationId`, jurisdiction objects contain a `submissionId`, and note objects contain a `noteId`. These identifiers are intended to be globally unique allowing any end-entity to create an object and assign a unique identifier without concern for collisions with other end-entities.

Record identifiers MUST be UUIDs generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. Two records with the same record identifier are considered duplicates. Older instances of a record SHOULD be overwritten when newer instances are received.

When a record is retransmitted from one entity to another, the same record identifier that was received by the sender MUST be retransmitted to the other entity. For example, when submission information is retransmitted by a test laboratory, the same `certificationId` that was received by the test laboratory from the vendor must be retransmitted to other entities. Similarly, when a note is retransmitted by a regulator, the same `noteId` that was received by the regulator must be retransmitted to other entities. New record identifiers should only be assigned when a completely new object is created; not when an end-entity is simply making updates to an existing object.

Chapter 2

Metadata Resources

2.1 jurisdictions Resource

The jurisdictions resource can be used by a Client System to request a list of all available regulatory jurisdictions to which it has access. A regulatory jurisdiction is an entity to which a product is submitted for approval prior to deployment. It is intended that all types of Client Systems – testing laboratories, vendors, and regulators – use this resource and that all types of Host Systems support the resource.

Table 2.1 jurisdictions HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/jurisdictions	No	Yes	No	No

2.1.1 GET jurisdictions Resource

The following table contains information about the `jurisdictions` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `jurisdictions` resource using the HTTP GET verb, the specified values MUST be used.

Table 2.2 GET jurisdictions Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/jurisdictions`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`jurisdictions` Object. See Section 2.1.1.2, jurisdictions Object for details.

2.1.1.1 GET jurisdictions Parameters

The following table identifies the parameters of the `jurisdictions` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `jurisdictionId` parameter is included, only information regarding the specified `jurisdictionId` is included in the response; otherwise, the `jurisdictionId` parameter is ignored and information about all jurisdictions is included in the response.

If the included parameters result in no jurisdictions being selected, the Host System MUST simply return an empty list of jurisdictions to the Client System.

Table 2.3 GET jurisdictions Parameters

Parameter	Restrictions	Description
`jurisdictionId`	`type: t_jurisdictionId` `use: optional`	UUID representing the jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**

2.1.1.2 jurisdictions Object

The following table identifies the properties of the `jurisdictions` object. Additional properties MAY be included in the `jurisdictions` object.

Table 2.4 jurisdictions Properties

Property	Restrictions	Description

`jurisdictionArray`	`type: jurisdiction`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `jurisdiction` objects. See Section 2.1.1.3, jurisdiction Object for details.

2.1.1.3 jurisdiction Object

The following table identifies the properties of the `jurisdiction` object. Additional properties MAY be included in the `jurisdiction` object.

Table 2.5 jurisdiction Properties

Property	Restrictions	Description
`jurisdictionId`	`type: t_jurisdictionId` `use: required`	UUID representing the jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
`jurisdictionName`	`type: t_name` `use: required`	Human-readable name of the jurisdiction. **Example: Macau**
`jurisdictionCode`	`type: t_code` `use: optional`	Host-specific code representing the jurisdiction. **Example: 158**

2.1.1.4 GET jurisdictions Example

The following example demonstrates the construction of a GET jurisdictions request and a response containing a `jurisdictions` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/jurisdictions HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 251
Content-Type: application/json; charset=utf-8


{
    "jurisdictionArray": [
        {
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
            "jurisdictionName": "Macau",
            "jurisdictionCode": "158"
```

```
        },
        {
            "jurisdictionId": "E5D7330D-5BD1-4856-9566-6F6279115F1E",
            "jurisdictionName": "Arizona",
            "jurisdictionCode": "22"
        }
    ]
}
```

2.2 testLabs Resource

The `testLabs` resource can be used by a Client System to request a list of all available test laboratories to which it has access. A test laboratory is an entity to which a product is submitted for certification. It is intended that all types of Client Systems – testing laboratories, vendors, and regulators – use this resource and that all types of Host Systems support the resource.

Table 2.6 testLabs HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/testLabs	No	Yes	No	No

2.2.1 GET testLabs Resource

The following table contains information about the `testLabs` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `testLabs` resource using the HTTP GET verb, the specified values MUST be used.

Table 2.7 GET testLabs Resource Information

HTTP Method	GET
Pathname	/cdi/[ver]/testLabs
Request Content-Type	application/json; charset=utf-8
Request Content	None.
Response Content-Type	application/json; charset=utf-8
Response Content	`testLabs` Object. See Section 2.2.1.2, testLabs Object for details.

2.2.1.1 GET testLabs Parameters

The following table identifies the parameters of the `testLabs` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `testLabId` parameter is included, only information regarding the specified `testLabId` is included in the response; otherwise, the `testLabId` parameter is ignored and information about all test laboratories is included in the response.

If the included parameters result in no test laboratories being selected, the Host System MUST simply return an empty list of test laboratories to the Client System.

Table 2.8 GET testLabs Parameters

Parameter	Restrictions	Description
`testLabId`	`type: t_testLabId` `use: optional`	UUID representing the test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**

2.2.1.2 testLabs Object

The following table identifies the properties of the `testLabs` object. Additional properties MAY be included in the `testLabs` object.

Table 2.9 GET testLabs Parameters

Parameter	Restrictions	Description

`testLabArray`	`type: testLab`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `testLab` objects. See Section 2.2.1.3, testLab Object for details.

2.2.1.3 testLab Object

The following table identifies the properties of the `testLab` object. Additional properties MAY be included in the `testLab` object.

Table 2.10 testLab Properties

Property	Restrictions	Description
`testLabId`	`type:t_testLabId` `use: required`	UUID representing the test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**
`testLabName`	`type: t_name` `use: required`	The human-readable name of the test laboratory. **Example: A Better Laboratory**
`testLabCode`	`type: t_code` `use: optional`	Host-specific code representing the test laboratory. **Example: ABL**

2.2.1.4 GET testLabs Example

The following example demonstrates the construction of a GET testLabs request and a response containing a `testLabs` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/testLabs HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 240
Content-Type: application/json; charset=utf-8


{
    "testLabArray": [
        {
            "testLabId": "456A3411-78FA-3234-B341-0078CA123489",
            "testLabName": "Gaming Standards",
            "testLabCode": "GSA"
        },
```

```
        {
            "testLabId": "1230987D-8976-4321-CC11-098123457634",
            "testLabName": "A Better Laboratory",
            "testLabCode": "ABL"
        }
    ]
}
```

2.3 vendors Resource

The vendors resource can be used by a Client System to request a list of all available vendors to which it has access. A vendor is an entity that submits products for certification and approval. It is intended that all types of Client Systems – testing laboratories, vendors, and regulators – use this resource and that all types of Host Systems support the resource.

Table 2.11 vendors HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/vendors	No	Yes	No	No

2.3.1 GET vendors Resource

The following table contains information about the `vendors` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `vendors` resource using the HTTP GET verb, the specified values MUST be used.

Table 2.12 GET vendors Resource Information

HTTP Method	GET
Pathname	/cdi/[ver]/vendors
Request Content-Type	application/json; charset=utf-8
Request Content	None.
Response Content-Type	application/json; charset=utf-8
Response Content	`vendors` Object. See Section 2.3.1.2, vendors Object for details.

2.3.1.1 GET vendors Parameters

The following table identifies the parameters of the `vendors` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `vendorId` parameter is included, only information regarding the specified `vendorId` is included in the response; otherwise, the `vendorId` parameter is ignored and information about all vendors is included in the response.

If the included parameters result in no vendors being selected, the Host System MUST simply return an empty list of vendors to the Client System.

Table 2.13 GET vendors Parameters

Parameter	Restrictions	Description
`vendorId`	`type: t_vendorId` `use: optional`	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**

2.3.1.2 vendors Object

The following table identifies the properties of the `vendors` object. Additional properties MAY be included in the `vendors` object.

Table 2.14 vendors Properties

Property	Restrictions	Description

`vendorArray`	`type: vendor`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `vendor` objects. See Section 2.3.1.3, vendor Object for details.

2.3.1.3 vendor Object

The following table identifies the properties of the `vendor` object. Additional properties MAY be included in the `vendor` object.

Table 2.15 vendor Properties

Property	Restrictions	Description
`vendorId`	`type: t_vendorId` `use: required`	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`vendorName`	`type: t_name` `use: required`	The human-readable name of the vendor. **Example: A Better Vendor**
`vendorCode`	`type: t_code` `use: optional`	Host-specific code representing the vendor. **Example: ABV**

2.3.1.4 GET vendors Example

The following example demonstrates the construction of a GET vendors request and a response containing a `vendors` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/vendors HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 229
Content-Type: application/json; charset=utf-8


{
    "vendorArray": [
        {
            "vendorId": "2301987D-8976-4321-CC11-098123457634",
            "vendorName": "Gaming Standards",
            "vendorCode": "GSA"
        },
```

```
        {
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "vendorName": "A Better Vendor",
            "vendorCode": "ABV"
        }
    ]
}
```

2.4 algorithms Resource

The algorithms resource can be used by a Client System to request a list of all available algorithms to which it has access. Algorithms are methods for producing software signatures of components submitted for certification and approval. It is intended that all types of Client Systems – testing laboratories, vendors, and regulators – use this resource and that all types of Host Systems support the resource.

Table 2.16 algorithms HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/algorithms	No	Yes	No	No

2.4.1 GET algorithms Resource

The following table contains information about the algorithms resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the algorithms resource using the HTTP GET verb, the specified values MUST be used.

Table 2.17 GET vendors Resource Information

HTTP Method	GET
Pathname	/cdi/[ver]/algorithms
Request Content-Type	application/json; charset=utf-8
Request Content	None.
Response Content-Type	application/json; charset=utf-8
Response Content	`algorithms` Object. See Section 2.4.1.2, algorithms Object for details.

2.4.1.1 GET algorithms Parameters

The following table identifies the parameters of the `algorithms` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `algorithmId` parameter is included, only information regarding the specified `algorithmId` is included in the response; otherwise, the `algorithmId` parameter is ignored and information about all algorithms is included in the response.

If the included parameters result in no algorithms being selected, the Host System MUST simply return an empty list of algorithms to the Client System.

Table 2.18 GET algorithms Parameters

Parameter	Restrictions	Description
`algorithmId`	`type: t_algorithmId` `use: optional`	UUID Representing the algorithm for producing software signatures. **Example: 754580E4-2B24-4DF9-A583-469688405F39**

2.4.1.2 algorithms Object

The following table identifies the properties of the `algorithms` object. Additional properties MAY be included in the `algorithms` object.

Table 2.19 algorithms Properties

Property	Restrictions	Description

`algorithmArray`	`type: algorithm`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `algorithm` objects. See Section 2.4.1.3, algorithm Object for details.

2.4.1.3 algorithm Object

The following table identifies the properties of the `algorithm` object. Additional properties MAY be included in the `algorithm` object.

Table 2.20 algorithm Properties

Property	Restrictions	Description
`algorithmId`	`type: t_algorithmId` `use: required`	UUID representing the signature algorithm. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
`algorithmType`	`type: t_algorithmTypes` `use: required`	Human-readable name of the algorithm. **Example: SHA1**
`supportsSeed`	`type: boolean` `use: required`	Boolean indicating whether the Host System supports a seed when using the algorithm. **Example: false**
`supportsSalt`	`type: boolean` `use: required`	Boolean indicating whether the Host System supports a salt when using the algorithm. **Example: true**
`supportsHMAC`	`type: boolean` `use: required`	Boolean indicating whether the Host System supports an HMAC key when using the algorithm. **Example: true**
`supportsOffset`	`type: boolean` `use: required`	Boolean indicating whether the Host System supports offsets when using the algorithm. **Example: true**

2.4.1.4 GET algorithms Example

The following example demonstrates the construction of a GET algorithms request and a response containing an `algorithms` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/algorithms HTTP/1.1
Accept: application/json
```

```
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 385
Content-Type: application/json; charset=utf-8


{
    "algorithmArray": [
        {
            "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
            "algorithmType": "SHA1",
            "supportsSeed": "false",
            "supportsSalt": "true",
            "supportsHMAC": "true",
            "supportsOffset": "true"
        },
        {
            "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
            "algorithmType": "CRC-16",
            "supportsSeed": "true",
            "supportsSalt": " false ",
            "supportsHMAC": "false ",
            "supportsOffset": "false "
        }
    ]
}
```

2.5 statuses Resource

The `statuses` resource can be used by a Client System to request a list of all available status codes to which it has access. The status codes — such as Pending, Approved, Revoked, etc. — are used to indicate the current state of a product submission at a testing laboratory or regulator. It is intended that all types of Client Systems — testing laboratories, vendors, and regulators — use this resource and that all types of Host Systems support the resource.

Table 2.21 statuses HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/statuses	No	Yes	No	No

2.5.1 GET statuses Resource

The following table contains information about the `statuses` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `statuses` resource using the HTTP GET verb, the specified values MUST be used.

Table 2.22 GET statuses Resource Information

HTTP Method	GET
Pathname	/cdi/[ver]/statuses
Request Content-Type	application/json; charset=utf-8
Request Content	None.
Response Content-Type	application/json; charset=utf-8
Response Content	`statuses` Object. See Section 2.5.1.2, statuses Object for details.

2.5.1.1 GET statuses Parameters

The following table identifies the parameters of the `statuses` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `statusId` parameter is included, only information regarding the specified `statusId` is included in the response; otherwise, the `statusId` parameter is ignored and information about all statuses is included in the response.

If the included parameters result in no statuses being selected, the Host System MUST simply return an empty list of statuses to the Client System.

Table 2.23 GET statuses Parameters

Parameter	Restrictions	Description
`statusId`	`type: t_statusId` `use: optional`	UUID Representing the status. **Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593**

2.5.1.2 statuses Object

The following table identifies the properties of the `statuses` object. Additional properties MAY be included in the `statuses` object.

Table 2.24 statuses Properties

Property	Restrictions	Description

`statusArray`	`type: status`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `status` objects. See Section 2.5.1.3, status Object for details.

2.5.1.3 status Object

The following table identifies the properties of the `status` object. Additional properties MAY be included in the `status` object.

Table 2.25 status Properties

Property	Restrictions	Description
`statusId`	`type: t_statusId` `use: required`	UUID representing the status. **Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593**
`statusName`	`type: t_name` `use: required`	The human-readable name of the status. **Example: Approved**
`statusCode`	`type: t_code` `use: optional`	Host-specific code representing the status. **Example: AP**

2.5.1.4 GET statuses Example

The following example demonstrates the construction of a GET statuses request and a response containing a `statuses` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/statuses HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 404
Content-Type: application/json; charset=utf-8


{
    "statusArray": [
        {
            "statusId": "450FBA8B-8C2F-43D3-8FEF-66543D04B593",
            "statusName": "Approved",
            "statusCode": "AP "
        },
```

```
        {
            "statusId": "4EC10B07-C562-49DA-A31A-236C745AAE9B",
            "statusName": "Withdrawn",
            "statusCode": "WD"
        },
        {
            "statusId": "A255605E-2313-47B8-9FFF-42455EAFEB18",
            "statusName": "Revoked",
            "statusCode": "RV"
        },
        {
            "statusId": "59F6FDE8-39F8-439D-9CAF-22A64F283305",
            "statusName": "Draft",
            "statusCode": "DR"
        }
    ]
}
```

2.6 operators Resource

Extension in v1.1

The `operators` resource can be used by a Client System to request a list of all available operators to which it has access. An operator is an entity that is licensed to operate gaming equipment within a jurisdiction. It is intended that various types of Client Systems — vendors, operators, and regulators — use this resource and that Host Systems that support the shipment resource also support this resource.

If a licensee has several properties to which products are shipped, each property SHOULD be treated as a separate operator and assigned its own operator identifier.

Table 2.26 operators HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/operators	No	Yes	No	No

2.6.1 GET operators Resource

The following table contains information about the `operators` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `operators` resource using the HTTP GET verb, the specified values MUST be used.

Table 2.27 GET operators Resource Information

HTTP Method	GET
Pathname	/cdi/[ver]/operators
Request Content-Type	application/json; charset=utf-8
Request Content	None.
Response Content-Type	application/json; charset=utf-8
Response Content	`operators` Object. See Section 2.6.1.2, operators Object for details.

2.6.1.1 GET operators Parameters

The following table identifies the parameters of the `operators` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `operatorId` parameter is included, only information regarding the specified `operatorId` is included in the response; otherwise, the `operatorId` parameter is ignored and information about all operators is included in the response.

If the included parameters result in no operators being selected, the Host System MUST simply return an empty list of operators to the Client System.

Table 2.28 GET operators Parameters

Parameter	Restrictions	Description
operatorId	type: t_operatorId use: optional	UUID Representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**

2.6.1.2 operators Object

The following table identifies the properties of the operators object. Additional properties MAY be included in the operators object.

Table 2.29 operators Properties

Property	Restrictions	Description
operatorArray	type: operator use: required minItems: 0 maxItems: ∞	Array of operator objects. See Section 2.6.1.3, operator Object for details.

2.6.1.3 operator Object

The following table identifies the properties of the operator object. Additional properties MAY be included in the operator object.

Table 2.30 operator Properties

Property	Restrictions	Description
operatorId	type: t_operatorId use: required	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**
operatorName	type: t_name use: required	The human-readable name of the operator. **Example: A Bettor Operator**
operatorCode	type: t_code use: optional	Host-specific code representing the operator. **Example: ABO**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing the jurisdiction in which the operator is licensed. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**

2.6.1.4 GET operators Example

The following example demonstrates the construction of a GET operators request and a response containing an operators object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.1/operators HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 389
Content-Type: application/json; charset=utf-8

{
    "operatorArray": [
        {
            "operatorId": "987D2301-8976-4321-CC11-098123457634",
            "operatorName": "Gaming Standards",
            "operatorCode": "GSA",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2"
        },
        {
            "operatorId": "3411456A-78FA-3234-B341-0078CA123489",
            "operatorName": "A Better Operator",
            "operatorCode": "ABO",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2"
        }
    ]
}
```

Chapter 3

Submission Resources

3.1 submissions Resource

The `submissions` resource can be used by a Client System to submit certification requests to a Host System, to add new jurisdictions to a previous certification request, and to retrieve status information about the submission. It is intended that vendors use this resource as Client Systems to submit certification requests to testing laboratories acting as Host Systems. After a submission has been accepted by a test laboratory, vendors should use the `certifications` resource to request status information about the testing and approval of the submitted product.

Table 3.1 submissions HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/submissions	Yes	Yes	Yes	No

3.1.1 POST submissions Resource

The following table contains information about the `submissions` resource when the HTTP POST verb is used. It includes the pathname and content type used to access the resource. When accessing the `submissions` resource using the HTTP POST verb, the specified values MUST be used.

This resource is used to submit new certification requests to the Host System. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System.

Table 3.2 POST submissions Resource Information

HTTP Method	`POST`
Pathname	`/cdi/[ver]/submissions`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`submissions` Object. See Section 3.1.1.1, submissions Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

3.1.1.1 submissions Object

The following table identifies the properties of the `submissions` object. Additional properties MAY be included in the `submissions` object.

Table 3.3 submissions Properties

Property	Restrictions	Description

`submissionArray`	`type: certificationSub`
	`use: required`
	`minItems: 1`
	`maxItems: ∞`
		Array of `certificationSub` Objects. See Section 3.1.1.2, certificationSub Object for details.

3.1.1.2 certificationSub Object

The following table identifies the properties of the `certificationSub` object. Additional properties MAY be included in the `certificationSub` object.

The `certificationId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `submission` record. Two `submission` records with the same `certificationId` are considered duplicates.

Table 3.4 certificationSub Properties

Property	Restrictions	Description
`certificationId`	`type: t_certificationId` `use: required`	UUID for the product certification request record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
`certificationNumber`	`type: t_name` `use: required`	The human-readable identification number for the product certification request. **Example: ABV_123456**
`vendorId`	`type: t_vendorId` `use: required`	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`testLabId`	`type: t_testLabId` `use: required`	UUID representing the test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**

`certificationCode1`	`type: t_name`
	`use: optional`
	`default: <empty>`
		Vendor-specific code for tracking the certification request.
		Example: My Code 1

`certificationCode2`	`type: t_name`
	`use: optional`
	`default: <empty>`
		Vendor-specific code for tracking the certification request.
		Example: My Code 2

`noteArray`	`type: note`
	`use: optional`
	`minItems: 1`
	`maxItems: ∞`
		Array of `note` objects about the certification request. See Section 3.1.1.6, note Object for details.

`componentSubArray`	`type: componentSub`
	`use: required`
	`minItems: 1`
	`maxItems: ∞`
		Array of `componentSub` objects. See Section 3.1.1.3, componentSub Object for details.

3.1.1.3 componentSub Object

The following table identifies the properties of the `componentSub` object. Additional properties MAY be included in the `componentSub` object.

The `componentId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `componentSub` record. Two `componentSub` records with the same `componentId` are considered duplicates.

Table 3.5 componentSub Properties

Property	Restrictions	Description
componentId	type: t_componentId use: required	UUID for the product component record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
componentNumber	type: t_name use: required	The human-readable identification number for the product component. **Example: ABV_123456**
version	type: t_version Use: optional default: <empty>	Identifies the version of the component. **Example: 1.2.3**
function	type: t_name use: optional default: <empty>	Identifies the function of the component. **Example: Game Software**
mediaType	type: t_name use: optional default: <empty>	Identifies the type of media on which the component resides. **Example: Download**
mediaSize	type: numeric use: optional multipleOf: 1 default: 0	Identifies the size of media on which the component resides. **Example: 654321**
position	type: t_name use: optional default: <empty>	Identifies the position in which the component is installed; for example, the socket on an EGMs motherboard. **Example: Socket U2**
componentName	type: t_name use: optional default: <empty>	The human-readable name of the component, such as the name of a game. **Example: A Better Game**
componentCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the component. **Example: My Code 1**
componentCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the component. **Example: My Code 2**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the component See Section 3.1.1.6, note Object for details.
paytableArray	type: paytable use: optional minItems: 1 maxItems: ∞	Array of paytable objects associated with the component. See Section 3.1.1.7, paytable Object for details.

Table 3.5 componentSub Properties

Property	Restrictions	Description
jurisdictionSubArray	type: jurisdictionSub use: required minItems: 1 maxItems: ∞	Array of jurisdictionSub objects. See Section 3.1.1.4, jurisdictionSub Object for details.
signatureArray	type: signature use: required minItems: 0 maxItems: ∞	Array of signature objects. See Section 3.1.1.5, signature Object for details.

3.1.1.4 jurisdictionSub Object

The following table identifies the properties of the jurisdictionSub object. Additional properties MAY be included in the jurisdictionSub object.

The submissionId MUST be UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the jurisdictionSub record. Two jurisdictionSub records with the same submissionId are considered duplicates.

Table 3.6 jurisdictionSub Properties

Property	Restrictions	Description
submissionId	type: t_submissionId use: required	UUID for the jurisdiction submission record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
rush	type: boolean use: optional default: false	Indicates whether the submitter considers the submission a rush. **Example: true**
requestedDate	type: string format: date use: optional default: <empty>	Requested date for when the certification will be issued. **Example: 2015-12-31**
jurisdictionCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the jurisdictional testing and approval. **Example: My Code 1**
jurisdictionCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the jurisdictional testing and approval. **Example: My Code 2**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects for the jurisdiction. See Section 3.1.1.6, note Object for details.

Table 3.6 jurisdictionSub Properties

Property	Restrictions	Description
documentArray	type: document use: optional minItems: 1 maxItems: ∞	Array of document objects associated with the submission for the jurisdiction. See Section 3.1.1.8, document Object for details.

3.1.1.5 signature Object

The following table identifies the properties of the signature object. Additional properties MAY be included in the signature object.

The signatureId MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8: 2014 standard to guarantee uniqueness. It serves as the unique identifier for the signature record. Two signature records with the same signatureId are considered duplicates.

Like other data reported to an end-point, only signatures relevant to a specific end-point should be provided to the end-point. If a specific set of signatures has been generated for a specific jurisdiction, the set of signatures should only be provided to that jurisdiction; the set of signatures should be restricted to that jurisdiction. In such cases, the signatureRestricted property SHOULD identify the jurisdiction (or other entity) to which the signatures are restricted. If the signatureRestricted property is not <empty>, the signature MUST only be provided to the entities identified in the signatureRestricted property and the signatureSource property.

Table 3.7 signature Properties

Property	Restrictions	Description
signatureId	type: t_signatureId use: required	UUID for the signature record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
algorithmId	type: t_algorithmId use: required	UUID representing the authentication algorithm. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
usedSeed	type: boolean use: optional default: false	Indicates whether a seed was used with the algorithm. Certain algorithms, such as checksums (CRC), can use a seed to define the starting value.
usedSalt	type: boolean use: optional default: false	Indicates whether a salt was used with the algorithm. Certain algorithms, such as SHA1 and MD5, can prepend arbitrary bytes to the component's byte buffer before the hash is generated (and after the offsets are applied).
usedOffsets	type: boolean use: optional default: false	Indicates whether starting and/or ending offsets were used with the algorithm.
verifyResult	type: t_verifyResult use: required	The software signature calculated by the Client System. **Example: 0F1E2D3C4B5A69788796A5B4C3D2E1F0**

Table 3.7 signature Properties

Property	Restrictions	Description
signatureSource	type: t_UUID use: required	Used to identify the source of the signature — that is, the vendor, test laboratory, or jurisdiction that originated the note. **Example: 456A3411-78FA-3234-B3410078CA123489**
signatureRestricted	type: t_UUID use: optional default: <empty>	Used to identify the entity to which the signature is restricted — that is, the vendor, test laboratory, or jurisdiction for which the signature was specifically generated. **Example: 456A3411-78FA-3234-B3410078CA123489**

3.1.1.6 note Object

The following table identifies the properties of the note object. Additional properties MAY be included in the note object.

The noteType property should identify the type, category, or subject of the note. The noteText property should contain the text for the note.

The noteId MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the note record. Two note records with the same noteId are considered duplicates.

Table 3.8 note Properties

Property	Restrictions	Description
noteId	type: t_noteId use: required	UUID for the note record. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
noteType	type: t_name use: required	Identifies the type or category of note – for example, Modification, Special Request, etc. **Example: Special Request**
noteText	type: t_notes use: required	Contains the full text of the note. **Example: Please handle this request with high priority.**
noteSource	type: t_UUID use: required	Used to identify the source of the note — that is, the vendor, test laboratory, or jurisdiction that originated the note. **Example: 456A3411-78FA-3234-B3410078CA123489**

3.1.1.7 paytable Object

The following table identifies the properties of the paytable object. Additional properties MAY be included in the paytable object.

The paytableId MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the paytable record. Two paytable records with the same paytableId are considered duplicates.

Table 3.9 paytable Properties

Property	Restrictions	Description
paytableId	type: t_paytableId use: required	UUID for the paytable record. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
paytableName	type: t_name use: required	The human-readable identifier for the paytable. **Example: ABV_123456_9600**
paytableMax	type: t_percent use: required	Maximum payback percentage for the paytable; expressed in hundredths of a percent; for example, 97.35% is expressed as 9735. **Example: 9735**
paytableMin	type: t_percent use: required	Minimum payback percentage for the paytable; expressed in hundredths of a percent; for example, 96.65% is expressed as 9665. **Example: 9665**
paytableVI	type: t_paytableVI use: optional default: <empty>	Volatility index for the paytable. **Example: 16.552**
paytableCI	type: t_paytableCI use: optional default: <empty>	Confidence interval for the paytable. **Example: 90**

3.1.1.8 document Object

The following table identifies the properties of the document object. Additional properties MAY be included in the document object.

The documentId MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the document record. Two document records with the same documentId are considered duplicates.

Table 3.10 document Properties

Property	Restrictions	Description
documentId	type: t_documentId use: required	UUID for the document record. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
documentType	type: t_name use: required	Identifies the type or category of the document – for example, Product Documentation, Par Sheet, Certification Report, Approval Letter, etc. **Example: Product Documentation**

Table 3.10 document Properties

Property	Restrictions	Description
documentName	type: t_name use: required	The human-readable identifier for the document. **Example: ABV_Package_123456**
documentSource	type: t_UUID use: required	Used to identify the source of the document — that is, the vendor, test laboratory, or jurisdiction that originated the document. **Example: 456A3411-78FA-3234-B3410078CA123489**
documentFormat	type: t_code use: required	Document format; typically, the file extension for the document; for example, pdf, doc, zip, etc. **Example: pdf**
documentLanguage	type: t_language use: optional default: en	Language of the document. **Example: en**
documentDate	type: string format: t_date use: required	Document Date; the date on which the document was last updated.
documentPath	type: string format: uri use: required	Network URI whence the document can be retrieved. **Example: https://www.abv.com/subs/ABV_Package_123456.pdf**

3.1.1.9 POST submissions Example

The following example demonstrates the construction of a POST submissions request and a response indicating that the submission was accepted. In practice, additional HTTP headers may be included in the messages.

Request:

```
POST /cdi/1.0/submissions HTTP/1.1
Content-Length: 2271
Content-Type: application/json; charset=utf-8

{
    "submissionArray": [
        {
            "certificationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
            "certificationNumber": "ABV_Cert_123",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "testLabId": "1230987D-8976-4321-CC11-098123457634",
            "certificationCode1": "My Certification Code 1",
            "certificationCode1": "My Certification Code 2",
            "noteArray": [
                {
                    "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "This submission is a rush.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "componentSubArray": [
```

```
{
    "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
    "componentNumber": "ABV_Comp_1234",
    "version": "1.2a",
    "function": "Game Software",
    "mediaType": "Download",
    "mediaSize": 654321,
    "position": "Socket U2",
    "componentName": "Triple 7s",
    "componentCode1": "My Component Code 1",
    "componentCode2": "My Component Code 2",
    "noteArray": [
        {
            "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
            "noteType": "Note",
            "noteText": "This game requires skill to achieve max payback.",
            "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
        }
    ],
    "paytableArray": [
        {
            "paytableId": "3896CA8F-A692-4748-847A-A1DA00D2B95C",
            "paytableName": "ABV_123456_9600",
            "paytableMax": "9600",
            "paytableMin": "9450",
            "paytableVI": "16.552",
            "paytableCI": "90"
        }
    ],
    "jurisdictionSubArray": [
        {
            "submissionId": "7FDE945E-24FE-414C-BF6E-E0F3A1DA00E3",
            "jursidictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
            "rush": "true",
            "requestedDate": "2015-12-31",
            "noteArray": [
                {
                    "noteId": "968F38CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "Field trial requested.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "documentArray": [
                {
                    "documentId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "documentType": "Submission Package",
                    "documentName": "ABV_Package_123456",
                    "documentSource": "456A3411-78FA-3234-B341-0078CA123489"
                    "documentFormat": "zip",
                    "documentLanguage": "en",
                    "documentDate": "2015-10-19",
                    "documentPath": "https://www.abv.com/packages/
                    ABV_Package_123456"
                }
            ]
        }
    ],
    "signatureArray": [
        {
            "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
            "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
```

```
                        "verifyResult": "12345678901234567890123456789012345678901234567890"
                    },
                    {
                        "signatureId": "E4758045-2B24-4DF9-A583-884046695F39",
                        "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
                        "seedUsed": true,
                        "verifyResult": "1234"
                    }
                ]
            }
        ]
    }
]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

3.1.2 GET submissions Resource

The following table contains information about the `submissions` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `submissions` resource using the HTTP GET verb, the specified values MUST be used.

This resource can be used to retrieve status information about previously submitted certification requests from the Host System.

Table 3.11 GET submissions Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/submissions`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`submissionStatuses` Object. See Section 3.1.2.2, submissionStatuses Object for details.

3.1.2.1 GET submissions Parameters

The following table identifies the parameters of the `submissions` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `certificationId` parameter is included, only status information regarding certification requests with the specified `certificationId` are included in the response; otherwise, the `certificationId` parameter is ignored.
- If the `submissionStatus` parameter is included, only status information regarding certification requests with the specified `submissionStatus` are included in the response; otherwise, the `submissionStatus` parameter is ignored.

- If the `submissionStart` parameter is included, only information regarding certification requests with `submissionDateTime` values greater than or equal to the specified `submissionStart` value are included in the response; otherwise, the `submissionStart` parameter is ignored.
- If the `jurisdictionId` parameter is included, only status information regarding certification requests with the specified `jurisdictionId` are included in the response; otherwise, the `jurisdictionId` parameter is ignored.
- If no parameters are included, information regarding all certification requests to which the Client System has access are included in the response.

If the included parameters result in no certification requests being selected, the Host System MUST simply return an empty list of certification requests to the Client System.

Table 3.12 GET submissions Parameters

Parameter	Restrictions	Description
`certificationId`	`type: t_certificationId` `use: optional`	UUID representing a particular certification request record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
`jurisdictionId`	`type: t_jurisdictionId` `use: optional`	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
`submissionStatus`	`type: t_subStatuses` `use: optional`	The status of a submission for a jurisdiction. **Example: Approved**

`submissionStart`	`type: string`
	`format: dateTime`
	`use: optional`
		The earliest date/time for which changes to the submission status are requested.
		Example: 2010-06-03T00:00:00

3.1.2.2 submissionStatuses Object

The following table identifies the properties of the `submissionStatuses` object. Additional properties MAY be included in the `submissionStatuses` object.

Table 3.13 submissionStatuses Properties

Property	Restrictions	Description

`submissionStatusArray`	`type: submissionStatus`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `submissionStatus` Objects. See Section 3.1.2.3, submissionStatus Object for details.

3.1.2.3 submissionStatus Object

The following table identifies the properties of the `submissionStatus` object. Additional properties MAY be included in the `submissionStatus` object.

Table 3.14 submissionStatus Properties

Property	Restrictions	Description
certificationId	type: t_certificationId use: required	UUID for the product certification request record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
certificationNumber	type: t_name use: required	The human-readable identification number for the product certification request. **Example: ABV_123456**
vendorId	type: t_vendorId use: required	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
testLabId	type: t_testLabId use: required	UUID representing the test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**
certificationCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the certification request. **Example: My Code 1**
certificationCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the certification request. **Example: My Code 2**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the certification request. See Section 3.1.1.6, note Object for details.
componentStatusArray	type: componentStatus use: required minItems: 1 maxItems: ∞	Array of componentStatus objects. See Section 3.1.2.4, componentStatus Object for details.

3.1.2.4 componentStatus Object

The following table identifies the properties of the componentStatus object. Additional properties MAY be included in the componentStatus object.

Table 3.15 componentStatus Properties

Property	Restrictions	Description
componentId	type: t_componentId use: required	UUID for the product component record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**

Table 3.15 componentStatus Properties

Property	Restrictions	Description
componentNumber	type: t_name use: required	The human-readable identification number for the product component. **Example: ABV_123456**
version	type: t_version Use: optional default: <empty>	Identifies the version of the component. **Example: 1.2.3**
function	type: t_name use: optional default: <empty>	Identifies the function of the component. **Example: Game Software**
mediaType	type: t_name use: optional default: <empty>	Identifies the type of media on which the component resides. **Example: Download**
mediaSize	type: numeric use: optional multipleOf: 1 default: 0	Identifies the size of media on which the component resides. **Example: 654321**
position	type: t_name use: optional default: <empty>	Identifies the position in which the component is installed; for example, the socket on an EGMs motherboard. **Example: Socket U2**
componentName	type: t_name use: optional default: <empty>	The human-readable name of the component, such as the name of a game. **Example: A Better Game**
componentCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the component. **Example: My Code 1**
componentCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the component. **Example: My Code 2**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the certification request. See Section 3.1.1.6, note Object for details.
paytableArray	type: paytable use: optional minItems: 1 maxItems: ∞	Array of paytable objects associated with the component. See Section 3.1.1.7, paytable Object for details.
jurisdictionStatusArray	type: jurisdictionStatus use: required minItems: 1 maxItems: ∞	Array of jurisdictionStatus objects. See Section 3.1.2.5, jurisdictionStatus Object for details.

Table 3.15 componentStatus Properties

Property	Restrictions	Description
signatureArray	type: signature use: required minItems: 0 maxItems: ∞	Array of signature objects. See Section 3.1.1.5, signature Object for details.

3.1.2.5 jurisdictionStatus Object

The following table identifies the properties of the jurisdictionStatus object. Additional properties MAY be included in the jurisdictionStatus object.

Table 3.16 jurisdictionStatus Properties

Property	Restrictions	Description
submissionId	type: t_submissionId use: required	UUID for the jurisdiction submission record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
rush	type: boolean use: optional default: false	Indicates whether the submitter considers the submission a rush. **Example: true**
requestedDate	type: string format: date use: optional default: <empty>	Requested date for when the certification will be issued. **Example: 2015-12-31**
jurisdictionCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the jurisdictional testing and approval. **Example: My Code 1**
jurisdictionCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the jurisdictional testing and approval. **Example: My Code 2**
submissionDateTime	type: string format: dateTime use: required	Date/time that the submissionStatus property was last updated. **Example: 2015-12-10T14:42:00**
submissionStatus	type: t_subStatuses use: required	Status of the certification request for the jurisdiction. **Example: Approved**

Table 3.16 jurisdictionStatus Properties

Property	Restrictions	Description
vendorSubmitted	type: string format: date use: required	Date that the certification request for the jurisdiction was received by the test laboratory. **Example: 2015-12-10**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the certification request. See Section 3.1.1.6, note Object for details.
documentArray	type: document use: optional minItems: 1 maxItems: ∞	Array of document objects associated with the submission for the jurisdiction. See Section 3.1.1.8, document Object for details.

3.1.2.6 GET submissions Example

The following example demonstrates the construction of a GET submissions request and a response containing a submissionStatus object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/submissions?certificationId=96CA8F38-A692-4748-847A-A1DA00D2B95C HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 2390
Content-Type: application/json; charset=utf-8


{
    "submissionStatusArray": [
        {
            "certificationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
            "certificationNumber": "ABV_Cert_123",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "testLabId": "1230987D-8976-4321-CC11-098123457634",
            "certificationCode1": "My Certification Code 1",
            "certificationCode1": "My Certification Code 2",
            "noteArray": [
                {
                    "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "This submission is a rush.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "componentStatusArray": [
                {
                    "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
                    "componentNumber": "ABV_Comp_1234",
                    "version": "1.2a",
                    "function": "Game Software",
                    "mediaType": "Download",
```

```
"mediaSize": 654321,
"position": "Socket U2",
"componentName": "Triple 7s",
"componentCode1": "My Component Code 1",
"componentCode2": "My Component Code 2",
"noteArray": [
    {
        "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
        "noteType": "Note",
        "noteText": "This game requires skill to achieve max payback.",
        "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
    }
],
"paytableArray": [
    {
        "paytableId": "3896CA8F-A692-4748-847A-A1DA00D2B95C",
        "paytableName": "ABV_123456_9600",
        "paytableMax": "9600",
        "paytableMin": "9450",
        "paytableVI": "16.552",
        "paytableCI": "90"
    }
],
"jurisdictionStatusArray": [
    {
        "submissionId": "7FDE945E-24FE-414C-BF6E-E0F3A1DA00E3",
        "jursidictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
        "rush": "true",
        "requestedDate": "2015-12-31",
        "submissionDateTime": "2015-12-12T14:42:00",
        "submissionStatus": "Approved",
        "vendorSubmitted": "2015-12-11",
        "noteArray": [
            {
                "noteId": "968F38CA-A692-4748-847A-A1DA00D2B95C",
                "noteType": "Special Request",
                "noteText": "Field trial requested.",
                "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
            }
        ],
        "documentArray": [
            {
                "documentId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                "documentType": "Submission Package",
                "documentName": "ABV_Package_123456",
                "documentSource": "456A3411-78FA-3234-B341-0078CA123489"
                "documentFormat": "zip",
                "documentLanguage": "en",
                "documentDate": "2015-10-19",
                "documentPath": "https://www.abv.com/packages/
                ABV_Package_123456"
            }
        ]
    }
],
"signatureArray": [
    {
        "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
        "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
        "verifyResult": "12345678901234567890123456789012345678901234567890"
    },
    {
```

```
                        "signatureId": "E4758045-2B24-4DF9-A583-884046695F39",
                        "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
                        "seedUsed": true,
                        "verifyResult": "1234"
                    }
                ]
            }
        ]
    }
]
}
```

3.1.3 PUT submissions Resource

The following table contains information about the `submissions` resource when the HTTP PUT verb is used. It includes the pathname and content type used to access the resource. When accessing the `submissions` resource using the HTTP PUT verb, the specified values MUST be used.

This resource is used to add new certification requests for specific jurisdictions onto a previous submission. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System.

Table 3.17 PUT submissions Resource Information

HTTP Method	`PUT`
Pathname	`/cdi/[ver]/submissions`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`newJurisdictions` Object. See Section 3.1.3.1, newJurisdictions Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

3.1.3.1 newJurisdictions Object

The following table identifies the properties of the `newJurisdictions` object. Additional properties MAY be included in the `newJurisdictions` object.

Table 3.18 newJurisdictions Properties

Property	Restrictions	Description

`newJurisdictionArray`	`type: newJurisdiction`
	`use: required`
	`mixItems: 1`
	`maxItems: ∞`
		Array of `newJurisdiction` Objects. See Section 3.1.3.2, newJurisdiction Object for details.

3.1.3.2 newJurisdiction Object

The following table identifies the properties of the `newJurisdiction` object. Additional properties MAY be included in the `newJurisdiction` object.

- The `certificationId` and `componentId` MUST match the `certificationId` and `componentId` from a previously submitted certification request.

Table 3.19 newJurisdiction Properties

Property	Restrictions	Description
`certificationId`	`type: t_certificationId` `use: required`	UUID from a previous certification request record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
`componentId`	`type: t_componentId` `use: required`	UUID for the product component. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**

`jurisdictionSubArray`	`type: jurisdictionSub`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `jurisdictionSub` objects. See Section 3.1.1.4, jurisdictionSub Object for details.

`signatureArray`	`type: signature`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `signature` objects. See Section 3.1.1.5, signature Object for details.

3.1.3.3 PUT submissions Example

The following example demonstrates the construction of a PUT submissions request and a response indicating that the request was accepted. In practice, additional HTTP headers may be included in the messages.

Request:

```
PUT /cdi/1.0/submissions HTTP/1.1
Content-Length: 1177
Content-Type: application/json; charset=utf-8


{
   "newJurisdictionArray": [
      {
         "certificationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
         "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
         "jurisdictionSubArray": [
            {
               "submissionId": "7FDE945E-24FE-414C-BF6E-E0F3A1DA00E3",
               "jursidictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
               "rush": "true",
               "requestedDate": "2015-12-31",
               "noteArray": [
                  {
                     "noteId": "968F38CA-A692-4748-847A-A1DA00D2B95C",
                     "noteType": "Special Request",
                     "noteText": "Field trial requested.",
                     "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                  }
               ],
               "documentArray": [
                  {
```

```
                        "documentId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                        "documentType": "Submission Package",
                        "documentName": "ABV_Package_123456",
                        "documentSource": "456A3411-78FA-3234-B341-0078CA123489"
                        "documentFormat": "zip",
                        "documentLanguage": "en",
                        "documentDate": "2015-10-19",
                        "documentPath": "https://www.abv.com/packages/
                        ABV_Package_123456"
                    }
                ]
            }
        ],
        "signatureArray": [
            {
                "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
                "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
                "verifyResult": "12345678901234567890123456789012345678901234567890"
            },
            {
                "signatureId": "E4758045-2B24-4DF9-A583-884046695F39",
                "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
                "seedUsed": true,
                "verifyResult": "1234"
            }
        ]
    }
  ]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

Chapter 4

Certification Resources

4.1 certifications Resource

The `certifications` resource can be used by a Client System to request a list of product certifications from a Host system. It is intended that regulators and vendors use this resource as Client Systems to retrieve the results of certification requests from testing laboratories acting as Host Systems. Regulators may also use this resource to retrieve the same information from vendors acting as Host Systems.

Table 4.1 certifications HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/certifications	No	Yes	No	No

4.1.1 GET certifications Resource

The following table contains information about the `certifications` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `certifications` resource using the HTTP GET verb, the specified values MUST be used.

Table 4.2 GET certifications Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/certifications`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	certifications Object. See Section 4.1.1.2, certifications Object for details.

4.1.1.1 GET certifications Parameters

The following table identifies the parameters of the certifications resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `certificationId` parameter is included, only information regarding certification requests with the specified `certificationId` are included in the response; otherwise, the `certificationId` parameter is ignored.
- If the `certificationNumber` parameter is included, only information regarding certification requests with the specified `certificationNumber` are included in the response; otherwise, the `certificationNumber` parameter is ignored.
- If the `componentId` parameter is included, only information regarding certification requests containing the specified `componentId` are included in the response; otherwise, the `componentId` parameter is ignored.
- If the `componentNumber` parameter is included, only information regarding certification requests containing the specified `componentNumber` are included in the response; otherwise, the `componentNumber` parameter is ignored.

- If the `componentName` parameter is included, only information regarding certification requests containing the specified `componentName` are included in the response; otherwise, the `componentName` parameter is ignored.
- If the `paytableId` parameter is included, only information regarding certification requests containing the specified `paytableId` are included in the response; otherwise, the `paytableId` parameter is ignored.
- If the `paytableName` parameter is included, only information regarding certification requests containing the specified `paytableName` are included in the response; otherwise, the `paytableName` parameter is ignored.
- If the `jurisdictionId` parameter is included, only information regarding certification requests with the specified `jurisdictionId` are included in the response; otherwise, the `jurisdictionId` parameter is ignored.
- If the `testLabStatusId` parameter is included, only information regarding certification requests with the specified `testLabStatusId` are included in the response; otherwise, the `testLabStatusId` parameter is ignored.
- If the `testLabStart` parameter is included, only information regarding certification requests with `testLabDateTime` values greater than or equal to the specified `testLabStart` value are included in the response; otherwise, the `testLabStart` parameter is ignored.
- If the `testLabEnd` parameter is included, only information regarding certification requests with `testLabDateTime` values less than or equal to the specified `testLabEnd` value are included in the response; otherwise, the `testLabEnd` parameter is ignored.
- If the `regulatorStatusId` parameter is included, only information regarding certification requests with the specified `regulatorStatusId` are included in the response; otherwise, the `regulatorStatusId` parameter is ignored.
- If the `regulatorStart` parameter is included, only information regarding certification requests with `regulatorDateTime` values greater than or equal to the specified `regulatorStart` value are included in the response; otherwise, the `regulatorStart` parameter is ignored.
- If the `regulatorEnd` parameter is included, only information regarding certification requests with `regulatorDateTime` values less than or equal to the specified `regulatorEnd` value are included in the response; otherwise, the `regulatorEnd` parameter is ignored.
- If no parameters are included, information regarding all certification requests to which the Client System has access are included in the response.

If the included parameters result in no certifications being selected, the Host System MUST simply return an empty list of certifications to the Client System.

Table 4.3 GET certifications Parameters

Parameter	Restrictions	Description
`certificationId`	`type: t_certificationId` `use: optional`	UUID representing a particular certification record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**

Table 4.3 GET certifications Parameters

Parameter	Restrictions	Description
certificationNumber	type: t_name use: optional	The human-readable identification number for the product certification request. **Example: ABV_123456**
componentId	type: t_componentId use: optional	UUID for the product component record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
componentNumber	type: t_name use: optional	The human-readable identification number for the product component. **Example: ABV_123456**
componentName	type: t_name use: optional	The human-readable name of the component, such as the name of a game. **Example: A Better Game**
paytableId	type: t_paytableId use: optional	UUID for the paytable record. **Example: 754800E4-2B24_4DF9-A583-469688405F39**
paytableName	type: t_name use: optional	The human-readable identifier for the paytable. **Example: ABV_123456-9600**
jurisdictionId	type: t_jurisdictionId use: optional	UUID representing a particular jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
testLabStatusId	type: t_statusId use: optional	UUID representing a particular status for the test laboratory. **Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593**
testLabStart	type: string format: dateTime use: optional	The earliest date/time for which changes to test laboratory status information is requested. **Example: 2010-06-03T00:00:00**
testLabEnd	type: string format: dateTime use: optional	The latest date/time for which changes to test laboratory status information is requested. **Example: 2010-06-03T23:59:59**
regulatorStatusId	type: t_statusId use: optional	UUID representing a particular status for the regulator. **Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593**

Table 4.3 GET certifications Parameters

Parameter	Restrictions	Description
regulatorStart	type: string format: dateTime use: optional	The earliest date/time for which changes to regulator status information is requested. **Example: 2010-06-03T00:00:00**
regulatorEnd	type: string format: dateTime use: optional	The latest date/time for which changes to regulator status information is requested. **Example: 2010-06-03T23:59:59**

4.1.1.2 certifications Object

The following table identifies the properties of the `certifications` object. Additional properties MAY be included in the `certifications` object.

Table 4.4 certifications Properties

Property	Restrictions	Description
certificationArray	type: certification use: required minItems: 0 maxItems: ∞	Array of `certification` Objects. See Section 4.1.1.3, certification Object for details.

4.1.1.3 certification Object

The following table identifies the properties of the `certification` object. Additional properties MAY be included in the `certification` object.

The `certificationId` serves as the unique identifier for the certification record. Two certification records with the same `certificationId` are considered duplicates.

Table 4.5 certification Properties

Property	Restrictions	Description
certificationId	type: t_certificationId use: required	UUID for the product certification request record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
certificationNumber	type: t_name use: required	The human-readable identification number for the product certification request. **Example: ABV_123456**
vendorId	type: t_vendorId use: required	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
testLabId	type: t_testLabId use: required	UUID representing the test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**

Table 4.5 certification Properties

Property	Restrictions	Description
certificationCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the certification request. **Example: My Code 1**
certificationCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the certification request. **Example: My Code 2**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the certification request. See Section 3.1.1.6, note Object for details.
componentArray	type: component use: required minItems: 1 maxItems: ∞	Array of component objects. See Section 4.1.1.4, component Object for details.

4.1.1.4 component Object

The following table identifies the properties of the component object. Additional properties MAY be included in the component object.

The componentId serves as the unique identifier for the component record. Two component records with the same componentId are considered duplicates.

Table 4.6 component Properties

Property	Restrictions	Description
componentId	type: t_componentId use: required	UUID for the product component record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
componentNumber	type: t_name use: required	The human-readable identification number for the product component. **Example: ABV_123456**
version	type: t_version Use: optional default: <empty>	Identifies the version of the component. **Example: 1.2.3**
function	type: t_name use: optional default: <empty>	Identifies the function of the component. **Example: Game Software**
mediaType	type: t_name use: optional default: <empty>	Identifies the type of media on which the component resides. **Example: Download**

Table 4.6 component Properties

Property	Restrictions	Description
mediaSize	type: numeric use: optional multipleOf: 1 default: 0	Identifies the size of media on which the component resides. **Example: 654321**
position	type: t_name use: optional default: <empty>	Identifies the position in which the component is installed; for example, the socket on an EGMs motherboard. **Example: Socket U2**
componentName	type: t_name use: optional default: <empty>	The human-readable name of the component, such as the name of a game. **Example: A Better Game**
componentCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the component. **Example: My Code 1**
componentCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the component. **Example: My Code 2**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the component. See Section 3.1.1.6, note Object for details.
paytableArray	type: paytable use: optional minItems: 1 maxItems: ∞	Array of paytable objects associated with the component. See Section 3.1.1.7, paytable Object for details.
jurisdictionArray	type: jurisdiction use: required minItems: 1 maxItems: ∞	Array of jurisdiction objects. See Section 4.1.1.5, jurisdiction Object for details.
signatureArray	type: signature use: required minItems: 0 maxItems: ∞	Array of signature objects. See Section 3.1.1.5, signature Object for details.

4.1.1.5 jurisdiction Object

The following table identifies the properties of the jurisdiction object. Additional properties MAY be included in the jurisdiction object.

The submissionId serves as the unique identifier for the jurisdiction record. Two jurisdiction records with the same submissionId are considered duplicates.

Table 4.7 jurisdiction Properties

Property	Restrictions	Description
submissionId	type: t_submissionId use: required	UUID for the jurisdiction submission record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
rush	type: boolean use: optional default: false	Indicates whether the submitter considers the submission a rush. **Example: true**
requestedDate	type: string format: date use: optional default: <empty>	Requested date for when the certification will be issued. **Example: 2015-12-31**
jurisdictionCode1	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the jurisdictional testing and approval. **Example: My Code 1**
jurisdictionCode2	type: t_name use: optional default: <empty>	Vendor-specific code for tracking the jurisdictional testing and approval. **Example: My Code 2**
vendorSubmitted	type: string format: date use: required	Date that the certification request for the jurisdiction was received by the test laboratory. **Example: 2015-12-10**
testLabStatusId	type: t_statusId use: required	UUID representing the test laboratory's status for the certification request for the jurisdiction. **Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593**
testLabDateTime	type: string status: dateTime use: required	Date/time that the information about the certification request for the jurisdiction was last updated; the testLabDateTime property SHOULD be updated any time that any property or sub-property of the jurisdiction object is updated. **Example: 2015-12-10T11:01:00**
testLabCertification	type: t_testLabCert use: optional default: <empty>	Contains the test laboratory's identifier for the certified component. **Example: MO-22-GSA-15-15**

Table 4.7 jurisdiction Properties

Property	Restrictions	Description
testLabCertified	type: string format: date use: optional default: <empty>	Date the certification request was approved by the test laboratory. **Example: 2015-12-10**
regulatorStatusId	type: t_statusId use: optional default: <empty>	UUID representing the regulator's status for the certification request for the jurisdiction. **Example: 2015-12-10**
regulatorDateTime	type: string status: dateTime use: optional defult: <empty>	Date/time that the information about the regulator approval was last updated; the regulatorDateTime property SHOULD be updated any time that any property or sub-property of the jurisdictionApproval object is updated. **Example: 2016-01-02T14:28:00**
regulatorApproved	type: string format: date use: optional default: <empty>	Date the component was approved by the regulator. **Example: 2016-01-02**
regulatorRevoked	type: string format: date use: optional default: <empty>	Date the component approval was revoked by the regulator. **Example: 2016-02-03**
regulatorReplaceBy	type: string format: date use: optional default: <empty>	Date the component must be removed from the field. **Example: 2016-10-04**
regulatorReplaceId	type: t_componentId use: optional default: <empty>	UUID of the component that replaces the revoked component. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects for the jurisdiction. See Section 3.1.1.6, note Object for details.
documentArray	type: document use: optional minItems: 1 maxItems: ∞	Array of document objects associated with the submission for the jurisdiction. See Section 3.1.1.8, document Object for details.
revokedArray	type: revoked use: optional minItems: 1 maxItems: ∞	Array of revoked component objects for the jurisdiction. See Section 4.1.1.6, revoked Object for details.

4.1.1.6 revoked Object

The following table identifies the properties of the `revoked` object. Additional properties MAY be included in the `revoked` object.

Table 4.8 revoked Properties

Property	Restrictions	Description
componentId	type: t_componentId use: required	UUID for the component that was revoked. **Example: 96CA8F38-F692-4748-847AA1DA00D2B95C**

4.1.1.7 GET certifications Example

The following example demonstrates the construction of a GET certifications request and a response containing a `certifications` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/certifications?certificationId=96CA8F38-A692-4748-847A-A1DA00D2B95C HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 2944
Content-Type: application/json; charset=utf-8


{
    "certificationArray": [
        {
            "certificationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
            "certificationNumber": "ABV_Cert_123",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "testLabId": "1230987D-8976-4321-CC11-098123457634",
            "certificationCode1": "My Certification Code 1",
            "certificationCode1": "My Certification Code 2",
            "noteArray": [
                {
                    "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "This submission is a rush.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "componentArray": [
                {
                    "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
                    "componentNumber": "ABV_Comp_1234",
                    "version": "1.2a",
                    "function": "Game Software",
                    "mediaType": "Download",
                    "mediaSize": 654321,
                    "position": "Socket U2",
                    "componentName": "Triple 7s",
                    "componentCode1": "My Component Code 1",
                    "componentCode2": "My Component Code 2",
                    "noteArray": [
```

```
        {
            "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
            "noteType": "Note",
            "noteText": "This game requires skill to achieve max payback.",
            "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
        }
    ],
    "paytableArray": [
        {
            "paytableId": "3896CA8F-A692-4748-847A-A1DA00D2B95C",
            "paytableName": "ABV_123456_9600",
            "paytableMax": "9600",
            "paytableMin": "9450",
            "paytableVI": "16.552",
            "paytableCI": "90"
        }
    ],
    "jurisdictionArray": [
        {
            "submissionId": "7FDE945E-24FE-414C-BF6E-E0F3A1DA00E3",
            "jurisdictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
            "rush": "true",
            "requestedDate": "2015-12-31",
            "vendorSubmitted": "2015-12-11",
            "testLabStatusId": "450FBA8B-8C2F-43D3-8FEF-66543D04B593",
            "testLabDateTime": "2015-12-14T08:53:00",
            "testLabCertification": "MO-22-GSA-15-15",
            "testLabCertified": "2015-12-14",
            "regulatorStatusId": "450FBA8B-8C2F-43D3-8FEF-66543D04B593",
            "regulatorDateTime": "2015-12-15T09:08:00",
            "regulatorApproved": "2015-12-15",
            "regulatorRevoked": "",
            "regulatorReplaceBy": "",
            "regulatorReplaceId": "",
            "noteArray": [
                {
                    "noteId": "968F38CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "Field trial requested.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "documentArray": [
                {
                    "documentId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "documentType": "Submission Package",
                    "documentName": "ABV_Package_123456",
                    "documentSource": "456A3411-78FA-3234-B341-0078CA123489"
                    "documentFormat": "zip",
                    "documentLanguage": "en",
                    "documentDate": "2015-10-19",
                    "documentPath": "https://www.abv.com/packages/
                    ABV_Package_123456"
                },
                {
                    "documentId": "38968FCA-A692-4748-847A-A1DA00D2B95C",
                    "documentType": "Test Lab Report",
                    "documentName": "ABL_ABV_Comp_1234_Report1",
                    "documentSource": "1230987D-8976-4321-CC11-098123457634"
                    "documentFormat": "zip",
                    "documentLanguage": "en",
                    "documentDate": "2015-10-30",
```

```
                            "documentPath": "https://www.abl.com/
                            ABL_ABV_Comp_1234_Report1"
                        }
                    ],
                    "revokedArray": [
                        {
                            "componentId": "96CAA1DA-A692-4748-847A-8F3800D2B95C"
                        }
                    ]
                }
            ],
            "signatureArray": [
                {
                    "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
                    "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
                    "verifyResult": "12345678901234567890123456789012345678901234567890"
                },
                {
                    "signatureId": "E4758045-2B24-4DF9-A583-884046695F39",
                    "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
                    "seedUsed": true,
                    "verifyResult": "1234"
                }
            ]
        }
    ]
  }
]
}
```

Chapter 5

Approval Resources

5.1 approvals Resource

The approvals resource can be used by a Client System to request a list of product approvals from a Host system. It is intended that vendors and test laboratories use this resource as Client Systems to retrieve the product approval information from regulators acting as Host Systems.

The approvals resource can also be used by a Client System to send product approval information to a host system. In this case, it is intended that regulators use this resource as Client Systems to send product approval information to vendors and test laboratories acting as Host systems.

Table 5.1 approvals HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/cdi/[ver]/approvals	No	Yes	Yes	No

5.1.1 GET approvals Resource

The following table contains information about the `approvals` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `approvals` resource using the HTTP GET verb, the specified values MUST be used.

Table 5.2 GET approvals Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/approvals`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`approvals` Object. See Section 5.1.1.2, approvals Object for details.

5.1.1.1 GET approvals Parameters

The following table identifies the parameters of the `approvals` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `certificationId` parameter is included, only information regarding approvals with the specified `certificationId` are included in the response; otherwise, the `certificationId` parameter is ignored.
- If the `vendorId` parameter is included, only information regarding approvals with the specified `vendorId` are included in the response; otherwise, the `vendorId` parameter is ignored.
- If the `testLabId` parameter is included, only information regarding approvals with the specified `testLabId` are included in the response; otherwise, the `testLabId` parameter is ignored.

- If the `regulatorStatusId` parameter is included, only information regarding approvals with the specified `regulatorStatusId` are included in the response; otherwise, the `regulatorStatusId` parameter is ignored.
- If the `regulatorStart` parameter is included, only information regarding approvals with `regulatorDateTime` values greater than or equal to the specified `regulatorStart` value are included in the response; otherwise, the `regulatorStart` parameter is ignored.
- If no parameters are included, information regarding all certification requests to which the Client System has access are included in the response.

If the included parameters result in no certifications being selected, the Host System MUST simply return an empty list of approvals to the Client System.

Table 5.3 GET approvals Parameters

Parameter	Restrictions	Description
`certificationId`	`type: t_certificationId` `use: optional`	UUID representing a particular certification record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
`vendorId`	`type: t_vendorId` `use: optional`	UUID representing a particular vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`testLabId`	`type: t_testLabId` `use: optional`	UUID representing a particular test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**
`regulatorStatusId`	`type: t_statusId` `use: optional`	UUID representing a particular status for the regulator. **Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593**

`regulatorStart`	`type: string`
	`format: date`
	`use: optional`
		The earliest date/time for which regulator approval updates are requested.
		Example: 2010-06-03T00:00:00

5.1.1.2 approvals Object

The following table identifies the properties of the `approvals` object. Additional properties MAY be included in the `approvals` object.

Table 5.4 approvals Properties

Property	Restrictions	Description

`approvalArray`	`type: certificationApproval`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `certificationApproval` Objects. See Section 5.1.1.3, certificationApproval Object for details.

5.1.1.3 certificationApproval Object

The following table identifies the properties of the `certificationApproval` object. Additional properties MAY be included in the `certificationApproval` object.

The `certificationId` serves as the unique identifier for the `certificationApproval` record. Two `certificationApproval` records with the same `certificationId` are considered duplicates.

Table 5.5 certificationApproval Properties

Property	Restrictions	Description
`certificationId`	`type: t_certificationId` `use: required`	UUID for the product certification request record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
`certificationNumber`	`type: t_name` `use: required`	The human-readable identification number for the product certification request. **Example: ABV_123456**
`vendorId`	`type: t_vendorId` `use: required`	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`testLabId`	`type: t_testLabId` `use: required`	UUID representing the test laboratory. **Example: 1230987D-8976-4321-CC11-098123457634**

`noteArray`	`type: note`
	`use: optional`
	`minItems: 1`
	`maxItems: ∞`
		Array of `note` objects about the certification request. See Section 3.1.1.6, note Object for details.

`componentApprovalArray`	`type: componentApproval`
	`use: required`
	`minItems: 1`
	`maxItems: ∞`
		Array of `componentApproval` objects. See Section 5.1.1.4, componentApproval Object for details

5.1.1.4 componentApproval Object

The following table identifies the properties of the `componentApproval` object. Additional properties MAY be included in the `componentApproval` object.

The `componentId` serves as the unique identifier for the `componentApproval` record. Two `componentApproval` records with the same `componentId` are considered duplicates.

Table 5.6 componentApproval Properties

Property	Restrictions	Description
`componentId`	`type: t_componentId` `use: required`	UUID for the product component record. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**

Table 5.6 componentApproval Properties

Property	Restrictions	Description
componentNumber	type: t_name use: required	The human-readable identification number for the product component. **Example: ABV_123456**
version	Type: t_version Use: optional default: <empty>	Identifies the version of the component. **Example: 1.2.3**
function	type: t_name use: optional default: <empty>	Identifies the function of the component. **Example: Game Software**
mediaType	type: t_name use: optional default: <empty>	Identifies the type of media on which the component resides. **Example: Download**
mediaSize	type: numeric use: optional multipleOf: 1 default: 0	Identifies the size of media on which the component resides. **Example: 654321**
position	type: t_name use: optional default: <empty>	Identifies the position in which the component is installed; for example, the socket on an EGMs motherboard. **Example: Socket U2**
componentName	type: t_name use: optional default: <empty>	The human-readable name of the component, such as the name of a game. **Example: A Better Game**
noteArray	type: note use: optional minItems: 1 maxItems: ∞	Array of note objects about the component. See Section 3.1.1.6, note Object for details.
paytableArray	type: paytable use: optional minItems: 1 maxItems: ∞	Array of paytable objects associated with the component. See Section 3.1.1.7, paytable Object for details.
jurisdictionApprovalArray	type: jurisdictionApproval use: required minItems: 1 maxItems: ∞	Array of jurisdictionApproval objects. See Section 5.1.1.5, jurisdictionApproval Object for details.
signatureArray	type: signature use: required minItems: 0 maxItems: ∞	Array of signature objects. See Section 3.1.1.5, signature Object for details.

5.1.1.5 jurisdictionApproval Object

The following table identifies the properties of the `jurisdictionApproval` object. Additional properties MAY be included in the `jurisdictionApproval` object.

The `submissionId` serves as the unique identifier for the `jurisdictionApproval` record. Two `jurisdictionApproval` records with the same `submissionId` are considered duplicates.

Table 5.7 jurisdictionApproval Properties

Property	Restrictions	Description
`submissionId`	`type: t_submissionId` `use: required`	UUID for the jurisdiction submission record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
`jurisdictionId`	`type: t_jurisdictionId` `use: required`	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**

`regulatorStatusId`	`type: t_statusId`
	`use: optional`
	`default: <empty>`
		UUID representing the regulator's status for the certification request.
		Example: 450FBA8B-8C2F-43D3-8FEF-66543D04B593

`regulatorDateTime`	`type: string`
	`status: dateTime`
	`use: optional`
	`defult: <empty>`
		Date/time that the information about the regulator approval was last updated; the `regulatorDateTime` property SHOULD be updated any time that any property or sub-property of the `jurisdictionApproval` object is updated.
		Example: 2016-01-02T14:28:00

`regulatorApproved`	`type: string`
	`format: date`
	`use: optional`
	`default: <empty>`
		Date the component was approved by the regulator.
		Example: 2016-01-02

`regulatorRevoked`	`type: string`
	`format: date`
	`use: optional`
	`default: <empty>`
		Date the component approval was revoked by the regulator.
		Example: 2016-02-03

`regulatorReplaceBy`	`type: string`
	`format: date`
	`use: optional`
	`default: <empty>`
		Date the component must be removed from the field.
		Example: 2016-10-04

`regulatorReplaceId`	`type: t_componentId`
	`use: optional`
	`default: <empty>`
		UUID of the component that replaces the revoked component.
		Example: 96CA8F38-A692-4748-847AA1DA00D2B95C

`noteArray`	`type: note`
	`use: optional`
	`minItems: 1`
	`maxItems: ∞`
		Array of `note` objects for the jurisdiction. See Section 3.1.1.6, note Object for details.

Table 5.7 jurisdictionApproval Properties

Property	Restrictions	Description
documentArray	type: document use: optional minItems: 1 maxItems: ∞	Array of document objects associated with the submission for the jurisdiction. See Section 3.1.1.8, document Object for details.
revokedArray	type: revoked use: optional minItems: 1 maxItems: ∞	Array of revoked component objects for the jurisdiction. See Section 4.1.1.6, revoked Object for details.

5.1.1.6 GET approvals Example

The following example demonstrates the construction of a GET approvals request and a response containing an approvals object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/approvals?certificationId=96CA8F38-A692-4748-847A-A1DA00D2B95C HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 2628
Content-Type: application/json; charset=utf-8


{
    "approvalArray": [
        {
            "certificationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
            "certificationNumber": "ABV_Cert_123",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "testLabId": "1230987D-8976-4321-CC11-098123457634",
            "noteArray": [
                {
                    "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "This submission is a rush.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "componentApprovalArray": [
                {
                    "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
                    "componentNumber": "ABV_Comp_1234",
                    "version": "1.2a",
                    "function": "Game Software",
                    "mediaType": "Download",
                    "mediaSize": 654321,
                    "position": "Socket U2",
                    "componentName": "Triple 7s",
                    "componentCode1": "My Component Code 1",
                    "componentCode2": "My Component Code 2",
                    "noteArray": [
                        {
```

```
            "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
            "noteType": "Note",
            "noteText": "This game requires skill to achieve max payback.",
            "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
        }
    ],
    "paytableArray": [
        {
            "paytableId": "3896CA8F-A692-4748-847A-A1DA00D2B95C",
            "paytableName": "ABV_123456_9600",
            "paytableMax": "9600",
            "paytableMin": "9450",
            "paytableVI": "16.552",
            "paytableCI": "90"
        }
    ],
    "jurisdictionApprovalArray": [
        {
            "submissionId": "7FDE945E-24FE-414C-BF6E-E0F3A1DA00E3",
            "jursidictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
            "regulatorStatusId": "450FBA8B-8C2F-43D3-8FEF-66543D04B593",
            "regulatorDateTime": "2015-12-15T09:08:00",
            "regulatorApproved": "2015-12-15",
            "regulatorRevoked": "",
            "regulatorReplaceBy": "",
            "regulatorReplaceId": "",
            "noteArray": [
                {
                    "noteId": "968F38CA-A692-4748-847A-A1DA00D2B95C",
                    "noteType": "Special Request",
                    "noteText": "Field trial requested.",
                    "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                }
            ],
            "documentArray": [
                {
                    "documentId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                    "documentType": "Submission Package",
                    "documentName": "ABV_Package_123456",
                    "documentSource": "456A3411-78FA-3234-B341-0078CA123489"
                    "documentFormat": "zip",
                    "documentLanguage": "en",
                    "documentDate"" "2015-10-19",
                    "documentPath": "https://www.abv.com/packages/
                    ABV_Package_123456"
                },
                {
                    "documentId": "38968FCA-A692-4748-847A-A1DA00D2B95C",
                    "documentType": "Test Lab Report",
                    "documentName": "ABL_ABV_Comp_1234_Report1",
                    "documentSource": "1230987D-8976-4321-CC11-098123457634"
                    "documentFormat": "zip",
                    "documentLanguage": "en",
                    "documentDate": "2015-10-30",
                    "documentPath": "https://www.abl.com/
                    ABL_ABV_Comp_1234_Report1"
                }
            ],
            "revokedArray": [
                {
                    "componentId": "96CAA1DA-A692-4748-847A-8F3800D2B95C"
                }
```

```
                        ]
                    }
                ],
                "signatureArray": [
                    {
                        "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
                        "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
                        "verifyResult": "12345678901234567890123456789012345678901234567890"
                    },
                    {
                        "signatureId": "E4758045-2B24-4DF9-A583-884046695F39",
                        "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
                        "seedUsed": true,
                        "verifyResult": "1234"
                    }
                ]
            }
        ]
    }
]
}
```

5.1.2 PUT approvals Resource

The following table contains information about the `approvals` resource when the HTTP PUT verb is used. It includes the pathname and content type used to access the resource. When accessing the `approvals` resource using the HTTP PUT verb, the specified values MUST be used.

This resource is used by regulators to send product approval information to vendors and test laboratories. Alternatively, the GET approvals resource can be used by vendors and test laboratories to collect product approval information from regulators.

Table 5.8 PUT approvals Resource Information

HTTP Method	`PUT`
Pathname	`/cdi/[ver]/approvals`
Request Content-Type	`application/json; charset=utf-8`
Request Content	approvals Object. See Section 5.1.1.2, approvals Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

5.1.2.1 PUT approvals Example

The following example demonstrates the construction of a PUT approvals request and HTTP response. In practice, additional HTTP headers may be included in the messages.

Request:

```
PUT /cdi/1.0/approvals HTTP/1.1
Content-Length: 2631
Content-Type: application/json; charset=utf-8


{
```

```
"approvalArray": [
    {
        "certificationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
        "certificationNumber": "ABV_Cert_123",
        "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
        "testLabId": "1230987D-8976-4321-CC11-098123457634",
        "noteArray": [
            {
                "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                "noteType": "Special Request",
                "noteText": "This submission is a rush.",
                "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
            }
        ],
        "componentApprovalArray": [
            {
                "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
                "componentNumber": "ABV_Comp_1234",
                "version": "1.2a",
                "function": "Game Software",
                "mediaType": "Download",
                "mediaSize": 654321,
                "position": "Socket U2",
                "componentName": "Triple 7s",
                "componentCode1": "My Component Code 1",
                "componentCode2": "My Component Code 2",
                "noteArray": [
                    {
                        "noteId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                        "noteType": "Note",
                        "noteText": "This game requires skill to achieve max payback.",
                        "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                    }
                ],
                "paytableArray": [
                    {
                        "paytableId": "3896CA8F-A692-4748-847A-A1DA00D2B95C",
                        "paytableName": "ABV_123456_9600",
                        "paytableMax": "9600",
                        "paytableMin": "9450",
                        "paytableVI": "16.552",
                        "paytableCI": "90"
                    }
                ],
                "jurisdictionApprovalArray": [
                    {
                        "submissionId": "7FDE945E-24FE-414C-BF6E-E0F3A1DA00E3",
                        "jurisdictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
                        "regulatorStatusId": "450FBA8B-8C2F-43D3-8FEF-66543D04B593",
                        "regulatorDateTime": "2015-12-15T09:08:00",
                        "regulatorApproved": "2015-12-15",
                        "regulatorRevoked": "",
                        "regulatorReplaceBy": "",
                        "regulatorReplaceId": "",
                        "noteArray": [
                            {
                                "noteId": "968F38CA-A692-4748-847A-A1DA00D2B95C",
                                "noteType": "Special Request",
                                "noteText": "Field trial requested.",
                                "noteSource": "456A3411-78FA-3234-B341-0078CA123489"
                            }
                        ],
```

```
                        "documentArray": [
                            {
                                "documentId": "8F3896CA-A692-4748-847A-A1DA00D2B95C",
                                "documentType": "Submission Package",
                                "documentName": "ABV_Package_123456",
                                "documentSource": "456A3411-78FA-3234-B341-0078CA123489"
                                "documentFormat": "zip",
                                "documentLanguage": "en",
                                "documentDate": "2015-10-19",
                                "documentPath": "https://www.abv.com/packages/
                                ABV_Package_123456"
                            },
                            {
                                "documentId": "38968FCA-A692-4748-847A-A1DA00D2B95C",
                                "documentType": "Test Lab Report",
                                "documentName": "ABL_ABV_Comp_1234_Report1",
                                "documentSource": "1230987D-8976-4321-CC11-098123457634"
                                "documentFormat": "zip",
                                "documentLanguage": "en",
                                "documentDate": "2015-10-30",
                                "documentPath": "https://www.abl.com/
                                ABL_ABV_Comp_1234_Report1"
                            }
                        ],
                        "revokedArray": [
                            {
                                "componentId": "96CAA1DA-A692-4748-847A-8F3800D2B95C"
                            }
                        ]
                    }
                ],
                "signatureArray": [
                    {
                        "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
                        "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
                        "verifyResult": "12345678901234567890123456789012345678901234567890"
                    },
                    {
                        "signatureId": "E4758045-2B24-4DF9-A583-884046695F39",
                        "algorithmId": "0BEEB53E-45AE-4F53-9ED5-456E1CE8FFB8",
                        "seedUsed": true,
                        "verifyResult": "1234"
                    }
                ]
            }
        ]
    }
]
}
```


Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

Chapter 6

Calculation Resources

6.1 calculations Resource

The `calculations` resource can be used by a Client System to submit requests for software signature calculations to a Host System and to retrieve status information and the resulting signature from the Host. It is intended that regulators use this resource as Client Systems to submit calculation requests to testing laboratories and vendors acting as Host Systems. After a calculation request has been accepted by a test laboratory or vendor, regulators should use the `calculations` resource to request status information about the request and to retrieve the requested signatures.

Table 6.1 calculations HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
`/cdi/[ver]/calculations`	Yes	Yes	No	No

6.1.1 POST calculations Resource

The following table contains information about the `calculations` resource when the HTTP POST verb is used. It includes the pathname and content type used to access the resource. When accessing the `calculations` resource using the HTTP POST verb, the specified values MUST be used.

This resource is used to submit new calculation requests to the Host System. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System.

Table 6.2 POST calculations Resource Information

HTTP Method	`POST`
Pathname	`/cdi/[ver]/calculations`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`calculationRequest` Object. See Section 6.1.1.1, calculationRequest Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

6.1.1.1 calculationRequest Object

The following table identifies the properties of the `calculationRequest` object. Additional properties MAY be included in the `calculationRequest` object.

The `calculationId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `calculationRequest` record. Two `calculationRequest` records with the same `calculationId` are considered duplicates.

Table 6.3 calculationRequest Properties

Property	Restrictions	Description
calculationId	type: t_calculationId use: required	UUID for the calculation request record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7CD8A27D7C0BF2**
componentCalcArray	type: componentCalc use: required minItems: 1 maxItems: ∞	Array of componentCalc Objects. See Section 6.1.1.2, componentCalc Object for details.

6.1.1.2 componentCalc Object

The following table identifies the properties of the componentCalc object. Additional properties MAY be included in the componentCalc object.

Table 6.4 componentCalc Properties

Property	Restrictions	Description
componentId	type: t_componentId use: optional default: <empty>	UUID for the product component record. If <empty>, signatures MUST be calculated for all components approved for the jurisdiction that support the specified algorithm. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
algorithmId	type: t_algorithmId use: required	UUID representing the authentication algorithm. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
seed	type: t_seed use: optional default: <empty>	The seed for the algorithm. Certain algorithms, such as checksums (CRC), require a seed to define the starting value. **Example: 12345678**
salt	type: t_salt use: optional default: <empty>	The salt for the algorithm. Arbitrary bytes that are prefixed to the component's byte buffer before the hash is generated (and after the offsets are applied). **Example: FEDCBA9876543210**
startOffset	type: numeric use: optional multipleOf: 1 default: 0 minIncl: 0	The starting offset for the calculation. **Example: 4321**

Table 6.4 componentCalc Properties

Property	Restrictions	Description
endOffset	type: numeric use: optional multipleOf: 1 default: -1 minIncl: -1	The ending offset for the calculation. **Example: -1**

6.1.1.3 POST calculations Example

The following example demonstrates the construction of a POST calculations request and HTTP response. In practice, additional HTTP headers may be included in the messages.

Request:

```
POST /cdi/1.0/calculations HTTP/1.1
Content-Length: 237
Content-Type: application/json; charset=utf-8

{
    "calculationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
    "jurisdictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
    "componentCalcArray": [
        {
            "componentId": "",
            "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
            "salt": "MYNEWSALTVALUE"
        }
    ]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

6.1.2 GET calculations Resource

The following table contains information about the `calculations` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `calculations` resource using the HTTP GET verb, the specified values MUST be used.

This resource is used to retrieve the status of calculation requests previously submitted to the Host System.

Table 6.5 GET calculations Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/calculations`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`

Table 6.5 GET calculations Resource Information

Response Content	`calculationStatus` Object. See Section 6.1.2.2, calculationStatus Object for details.

6.1.2.1 GET calculations Parameters

The following table identifies the parameters of the `calculations` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

Table 6.6 GET calculations Parameters

Parameter	Restrictions	Description
`calculationId`	`type: t_calculationId` `use: required`	UUID for the calculation request record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**

6.1.2.2 calculationStatus Object

The following table identifies the properties of the `calculationStatus` object. Additional properties MAY be included in the `calculationStatus` object.

The `componentStatusArray` MAY be empty until the calculation request has been accepted — that is, the `requestStatus` property is set to `Accepted`.

Table 6.7 calculationStatus Properties

Property	Restrictions	Description
`calculationId`	`type: t_calculationId` `use: required`	UUID for the calculation request record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
`jurisdictionId`	`type: t_jurisdictionId` `use: required`	UUID representing a specific jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7CD8A27D7C0BF2**
`requestStatus`	`type: t_subStatuses` `use: required`	The status of the calculation request. **Example: Accepted**

`componentStatusArray`	`type: componentStatus`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `componentStatus` Objects. See Section 6.1.2.3, componentStatus Object for details.

6.1.2.3 componentStatus Object

The following table identifies the properties of the `componentStatus` object. Additional properties MAY be included in the `componentStatus` object.

When a calculation request has been made for all components — that is, `componentId` set to <`empty`> — the fully expanded list of qualifying components MUST be returned in the status response even if the calculations for some of the components have not been completed.

Table 6.8 componentStatus Properties

Property	Restrictions	Description
componentId	type: t_componentId use: required	UUID for the product component record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
algorithmId	type: t_algorithmId use: required	UUID representing the authentication algorithm. **Example: 754580E4-2B24-4DF9-A583-469688405F39**
calculationStatus	type: t_calcStatuses use: required	The status of the calculation for the component. **Example: Completed**
usedSeed	type: boolean use: optional default: false	Indicates whether a seed was used with the algorithm. Certain algorithms, such as checksums (CRC), can use a seed to define the starting value.
usedSalt	type: boolean use: optional default: false	Indicates whether a salt was used with the algorithm. Certain algorithms, such as SHA1 and MD5, can prepend arbitrary bytes to the component's byte buffer before the hash is generated (and after the offsets are applied).
usedOffsets	type: boolean use: optional default: false	Indicates whether starting and/or ending offsets were used with the algorithm.
signatureId	type: t_signatureId use: required	UUID for the signature record. **Example: 96CA8F38-A692-4748-847AA1DA00D2B95C**
verifyResult	type: t_verifyResult use: required	The software signature calculated by the Client System. **Example: 0F1E2D3C4B5A69788796A5B4C3D2E1F0**
signatureSource	type: t_UUID use: required	Used to identify the source of the signature — that is, the vendor, test laboratory, or jurisdiction that originated the note. **Example: 456A3411-78FA-3234-B3410078CA123489**
signatureRestricted	type: t_UUID use: optional default: <empty>	Used to identify the entity to which the signature is restricted — that is, the vendor, test laboratory, or jurisdiction for which the signature was specifically generated. **Example: 456A3411-78FA-3234-B3410078CA123489**

6.1.2.4 GET calculations Example

The following example demonstrates the construction of a GET calculations request and a response containing a `calculationStatus` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/calculations?calculationId=96CA8F38-A692-4748-847A-A1DA00D2B95C HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 562
Content-Type: application/json; charset=utf-8

{
    "calculationId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
    "jurisdictionId": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3",
    "requestStatus": "Accepted",
    "componentStatusArray": [
        {
            "componentId": "BA8B450F-8C2F-43D3-8FEF-04B59366543D",
            "algorithmId": "754580E4-2B24-4DF9-A583-469688405F39",
            "calculationStatus": "Completed",
            "usedSalt": true,
            "signatureId": "80E47545-2B24-4DF9-A583-688404695F39",
            "verifyResult": "12345678901234567890123456789012345678901234567890",
            "signatureSource": "1230987D-8976-4321-CC11-098123457634",
            "signatureRestricted": "E9457FDE-24FE-414C-BF6E-A1DA00E0F3E3"
        }
    ]
}
```

NEW CHAPTER

Chapter 7

Shipment Resources

Extension in v1.1

NEW CHAPTER

7.1 shipments Resource

The `shipments` resource is used by a Client System to submit shipping requests for gaming products to a Host System and, subsequently, to retrieve status information (approvals/denials) about those requests back from the Host. It is intended that vendors and operators use this resource as Client Systems to submit shipping requests to regulators acting as Host Systems. After a shipping request has been submitted to a regulator, vendors and operators can use the GET verb to request status information about the request. Vendors and operators can use the PUT verb to resubmit an amended request.

Table 7.1 shipments HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
`/cdi/[ver]/shipments`	Yes	Yes	Yes	No

7.1.1 POST shipments Resource

The following table contains information about the `shipments` resource when the HTTP POST verb is used. It includes the pathname and content type used to access the resource. When accessing the `shipments` resource using the HTTP POST verb, the specified values MUST be used.

This resource is used to submit new shipping requests to the Host System. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System.

Host Systems SHOULD NOT accept duplicate shipping requests when the POST verb is used — that is, two shipping requests with the same `shipmentId` should not be accepted. The PUT verb can be used to resubmit an amended shipping request with the same `shipmentId`.

In addition, Host Systems SHOULD NOT accept duplicate products into an operator's inventory when the POST verb or the PUT verb is used — that is, a `productId` that is already in the operator's inventory. Likewise, Host Systems SHOULD NOT remove duplicate products from an operator's inventory — that is, a `productId` that has already been removed from the operator's inventory or is not in the operator's inventory.

Table 7.2 POST shipments Resource Information

HTTP Method	`POST`
Pathname	`/cdi/[ver]/shipments`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`shipment` Object. See Section 7.1.1.1, shipment Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

7.1.1.1 shipment Object

The following table identifies the properties of the `shipment` object. Additional properties MAY be included in the `shipment` object.

NEW CHAPTER

The `shipmentId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `shipment` record. Two `shipment` records with the same `shipmentId` are considered duplicates.

Table 7.3 shipment Properties

Property	Restrictions	Description
`shipmentId`	`type: t_shipmentId` `use: required`	UUID representing the shipment. **Example: 1145346A-78FA-3234-B341-0078CA123489**
`shipmentName`	`type: t_name` `use: required`	The human-readable identification number for the shipping request. **Example: ABV_987654**
`jurisdictionId`	`type: t_jurisdictionId` `use: required`	UUID representing the jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
`shipperId`	`type: t_UUID` `use: required`	UUID representing the shipper; Vendor Identifier or Operator Identifier. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`shipperAddress`	`type: address` `use: required`	An `address` object containing the address of the shipper; see Section 7.1.1.4, address Object for details.
`receiverId`	`type: t_UUID` `use: optional`	UUID representing the receiver; Vendor Identifier or Operator Identifier; MAY be omitted when `disposal` is set to true; otherwise, MUST be included. **Example: 3411456A-78FA-3234-B341-0078CA123489**
`receiverAddress`	`type: address` `use: optional`	An `address` object containing the address of the receiver; MAY be omitted when `disposal` is set to true; otherwise, MUST be included; see Section 7.1.1.4, address Object for details.

`shippingDate`	`type: string`
	`format: date`
	`use: required`
		Estimated Shipping Date; date that the products are planned to be shipped.
		Example: 2018-01-23

`arrivalDate`	`type: string`
	`format: date`
	`use: optional`
		Estimated Arrival Date; date that the products are scheduled to arrive.
		Example: 2018-11-13

`disposal`	`type: boolean`
	`use: optional`
	`default: false`
		Disposal indicator.
		Example: true

`productArray`	`type: product`
	`use: required`
	`minItems: 1`
	`maxItems: ∞`
		Array of `product` objects for the shipment. See Section 7.1.1.2, product Object for details.

NEW CHAPTER

7.1.1.2 product Object

The following table identifies the properties of the `product` object. Additional properties MAY be included in the `product` object.

The `productId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `product` record. Two `product` records within the same `shipment` record with the same `productId` are considered duplicates.

Table 7.4 product Properties

Property	Restrictions	Description
`productId`	`type: t_productId` `use: required`	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**
`vendorId`	`type: t_vendorId` `use: required`	UUID representing the vendor of the product. **Example: 456A3411-78FA-3234-B341-0078CA123489**

`productName`	`type: t_name`
	`use: optional`
	`default: <empty>`
		The human-readable name of the product, such as the name of a game.
		Example: A Better Game

`productModel`	`type: t_name`
	`use: optional`
	`default: <empty>`
		Model of the product.
		Example: Super Duper

`productType`	`type: t_productTypes` `use: required`	Product Type. **Example: EGM**
`serialNumber`	`type: t_name` `use: required`	Serial Number of the product. **Example: ABV18012345**

`productComponentArray`	`type: productComponent`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `productComponent` objects for the shipment. See Section 7.1.1.3, productComponent Object for details.

7.1.1.3 productComponent Object

The following table identifies the properties of the `productComponent` object. Additional properties MAY be included in the `productComponent` object.

The `componentId` serves as the unique identifier for the `productComponent` record. Two `productComponent` records for the same `product` record with the same `componentId` are considered duplicates.

Table 7.5 productComponent Properties

Property	Restrictions	Description
componentId	type: t_componentId use: required	UUID for the component; identifies the type of product being shipped. **Example: 96CA8F38-A692-4748-847A-A1DA00D2B95C**
componentNumber	type: t_name use: required	The human-readable identification number for the component. **Example: ABV_123456**
version	type: t_version use: optional default: <empty>	Identifies the version of the component. **Example: 1.2.3**
function	type: t_name use: optional default: <empty>	Identifies the function of the component. **Example: Game Software**
componentName	type: t_name use: optional default: <empty>	The human-readable name of the component, such as the name of a game. **Example: A Better Game**

7.1.1.4 address Object

The following table identifies the properties of the address object. Additional properties MAY be included in the address object.

Table 7.6 address Properties

Property	Restrictions	Description
addressName	type: t_description use: required	Name of the address. **Example: ABC Warehouse**
addressLine1	type: t_description use: required	First line of the address. **Example: 1 Industrial Way**
addressLine2	type: t_description use: optional default: <empty>	Second line of address. **Example: Building C**
addressCity	type: t_description use: required	City name of the address. **Example: Anywhere**
stateProvCode	type: t_stateProvCode use: optional default: <empty>	State/Province code of the address. **Example: CT**
countryCode	type: t_countryCode use: required	Country code of the address. **Example: US**

Table 7.6 address Properties

Property	Restrictions	Description
postCode	type: t_postCode use: required	Postal code of the address. **Example: 06355**

7.1.1.5 POST shipments Example

The following example demonstrates the construction of a POST shipments request and a response indicating that the shipping request was accepted. In practice, additional HTTP headers may be included in the messages.

Request:

```
POST /cdi/1.1/shipments HTTP/1.1
Content-Length: 1157
Content-Type: application/json; charset=utf-8

{
    "shipmentId": "1145346A-78FA-3234-B341-0078CA123489",
    "shipmentName": "ABV_987654",
    "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
    "shipperId": "456A3411-78FA-3234-B341-0078CA123489",
    "shipperAddress": {
        "addressName": "ABC Warehouse",
        "addressLine1": "1 Industrial Way",
        "addressLine2": "Building C",
        "addressCity": "Anywhere",
        "stateProvCode": "CT",
        "countryCode": "US",
        "postCode": "06355"
    },
    "receiverId": "3411456A-78FA-3234-B341-0078CA123489",
    "receiverAddress": {
        "addressName": "Big Casino",
        "addressLine1": "777 Lucky Street",
        "addressCity": "Somewhere",
        "stateProvCode": "MA",
        "countryCode": "US",
        "postCode": "02030"
    },
    "shippingDate": "2018-01-30",
    "arrivalDate": "2018-01-30",
    "disposal": "false",
    "productArray": [
        {
            "productId": "CA121145-78FA-3234-B341-0078346A3489",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "version": "1.2.3",
            "function": "Game Software",
            "productName": "A Better Product",
            "productModel": "Super Duper",
            "productType": "software",
            "serialNumber": "ABV18012345",
            "productComponentArray": [
                {
                    "componentId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
```

```
                "componentNumber": "ABV_123456",
                "version": "1.2.3",
                "function": "Game Software",
                "componentName": "A Better Game"
            }
        ]
    }
  ]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

7.1.2 GET shipments Resource

The following table contains information about the `shipments` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `shipments` resource using the HTTP GET verb, the specified values MUST be used.

This resource can be used to retrieve status information from the Host System about previously submitted shipping requests.

Table 7.7 GET shipments Resource Information

HTTP Method	`POST`
Pathname	`/cdi/[ver]/shipments`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`shipmentStatuses` Object. See Section 7.1.2.2, shipmentStatuses Object for details.

7.1.2.1 GET shipments Parameters

The following table identifies the parameters of the `shipments` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `shipmentId` parameter is included, only status information regarding shipping requests with the specified `shipmentId` are included in the response; otherwise, the `shipmentId` parameter is ignored.
- If the `shipmentStatus` parameter is included, only status information regarding shipping requests with the specified `shipmentStatus` are included in the response; otherwise, the `shipmentStatus` parameter is ignored.
- If the `shipmentStart` parameter is included, only information regarding shipping requests with `shipmentDateTime` values greater than or equal to the specified `shipmentStart` value are included in the response; otherwise, the `shipmentStart` parameter is ignored.

- If the `shipmentEnd` parameter is included, only information regarding shipping requests with `shipmentDateTime` values less than or equal to the specified `shipmentEnd` value are included in the response; otherwise, the `shipmentEnd` parameter is ignored.
- If the `vendorId` parameter is included, only status information regarding shipping requests with the specified `vendorId` as the `shipperId` or `receiverId` are included in the response; otherwise, the `vendorId` parameter is ignored.
- If the `operatorId` parameter is included, only status information regarding shipping requests with the specified `operatorId` as the `shipperId` or `receiverId` are included in the response; otherwise, the `operatorId` parameter is ignored.
- If no parameters are included, information regarding all shipping requests to which the Client System has access are included in the response.

If the included parameters result in no shipping requests being selected, the Host System MUST simply return an empty list of shipping requests to the Client System.

Table 7.8 GET shipments Parameters

Parameters	Restrictions	Description
`shipmentId`	`type: t_shipmentId` `use: optional`	UUID representing the shipment. **Example: 1145346A-78FA-3234-B341-0078CA123489**
`shipmentStatus`	`type: t_shipmentStatuses` `use: optional`	Shipment Status. **Example: pending**

`shipmentStart`	`type: string`
	`format: dateTime`
	`use: optional`
		The earliest date/time for which changes to the shipment status are requested.
		Example: 2018-01-23T00:00:00-05:00

`shipmentEnd`	`type: string`
	`format: dateTime`
	`use: optional`
		The latest date/time for which changes to the shipment status are requested.
		Example: 2018-01-23T23:59:59-05:00

`vendorId`	`type: t_vendorId` `use: required`	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`operatorId`	`type: t_operatorId` `use: optional`	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**

7.1.2.2 shipmentStatuses Object

The following table identifies the properties of the `shipmentStatuses` object. Additional properties MAY be included in the `shipmentStatuses` object.

Table 7.9 shipmentStatuses Properties

Property	Restrictions	Description
shipmentStatusArray	type: shipmentStatus use: required minItems: 0 maxItems: ∞	Array of shipmentStatus Objects. See Section 7.1.2.3, shipmentStatus Object for details.

7.1.2.3 shipmentStatus Object

The following table identifies the properties of the shipmentStatus object. Additional properties MAY be included in the shipmentStatus object.

Table 7.10 shipmentStatus Properties

Property	Restrictions	Description
shipmentId	type: t_shipmentId use: required	UUID representing the shipment. **Example: 1145346A-78FA-3234-B341-0078CA123489**
shipmentName	type: t_name use: required	The human-readable identification number for the shipping request. **Example: ABV_987654**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing the jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
shipperId	type: t_UUID use: required	UUID representing the shipper; Vendor Identifier or Operator Identifier. **Example: 456A3411-78FA-3234-B341-0078CA123489**
shipperAddress	type: address use: required	An address object containing the address of the shipper; see Section 7.1.1.4, address Object for details.
receiverId	type: t_UUID use: optional	UUID representing the receiver; Vendor Identifier or Operator Identifier; MUST be omitted when the shipmentType is disposal; otherwise, MUST be included. **Example: 3411456A-78FA-3234-B341-0078CA123489**
receiverAddress	type: address use: optional	An address object containing the address of the receiver; MAY be omitted when disposal is set to true; otherwise, MUST be included; see Section 7.1.1.4, address Object for details.

NEW CHAPTER

Table 7.10 shipmentStatus Properties

Property	Restrictions	Description
shippingDate	type: string format: date use: required	Estimated Shipping Date; date that the products are planned to be shipped. **Example: 2018-01-23**
arrivalDate	type: string format: date use: optional	Estimated Arrival Date; date that the products are scheduled to arrive. **Example: 2018-11-13**
disposal	type: boolean use: optional default: false	Disposal indicator. **Example: true**
submittedDateTime	type: string format: dateTime use: required	Date/time that a new or amended shipping request was received by the regulator. **Example: 2018-01-23T13:46:59-05:00**
shipmentDateTime	type: string format: dateTime use: required	Date/time that the Shipping Request Status was last updated. **Example: 2018-01-23T13:46:59-05:00**
shipmentStatus	type: t_shipmentStatuses use: required	Shipping Request Status. **Example: pending**
shipmentAmended	type: boolean use: optional default: false	Amendment indicator. **Example: false**
productStatusArray	type: productStatus use: required minItems: 0 maxItems: ∞	Array of productStatus objects for the shipment. See Section 7.1.2.4, productStatus Object for details.

7.1.2.4 productStatus Object

The following table identifies the properties of the productStatus object. Additional properties MAY be included in the productStatus object.

Table 7.11 productStatus Properties

Property	Restrictions	Description
productId	type: t_productId use: required	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**
vendorId	type: t_vendorId use: required	UUID representing the vendor of the product. **Example: 456A3411-78FA-3234-B341-0078CA123489**

NEW CHAPTER

Table 7.11 productStatus Properties

Property	Restrictions	Description
productName	type: t_name use: optional default: <empty>	The human-readable name of the product, such as the name of a game. **Example: A Better Game**
productModel	type: t_name use: optional default: <empty>	Model of the product. **Example: Super Duper**
productType	type: t_productTypes use: required	Product Type. **Example: EGM**
serialNumber	type: t_name use: required	Serial Number of the product. **Example: ABV18012345**
productDateTime	type: string format: dateTime use: required	Date/time that the Product Shipping Status or Product Location was last updated. **Example: 2018-01-23T13:46:59-05:00**
productStatus	type: t_productStatuses use: required	Product Shipping Status. **Example: pending**
productComponentArray	type: productComponent use: required minItems: 0 maxItems: ∞	Array of productComponent objects for the shipment. See Section 7.1.1.3, productComponent Object for details.

7.1.2.5 GET shipments Example

The following example demonstrates the construction of a GET shipments request and a response containing a shipmentStatuses object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.1/shipments?shipmentId=1145346A-78FA-3234-B341-0078CA123489 HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 1397
Content-Type: application/json; charset=utf-8

{
    "shipmentStatusArray": [
        {
            "shipmentId": "1145346A-78FA-3234-B341-0078CA123489",
            "shipmentName": "ABV_987654",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
            "shipperId": "456A3411-78FA-3234-B341-0078CA123489",
```

NEW CHAPTER


```
            "shipperAddress": {
                "addressName": "ABC Warehouse",
                "addressLine1": "1 Industrial Way",
                "addressLine2": "Building C",
                "addressCity": "Anywhere",
                "stateProvCode": "CT",
                "countryCode": "US",
                "postCode": "06355"
            },
            "receiverId": "3411456A-78FA-3234-B341-0078CA123489",
            "receiverAddress": {
                "addressName": "Big Casino",
                "addressLine1": "777 Lucky Street",
                "addressCity": "Somewhere",
                "stateProvCode": "MA",
                "countryCode": "US",
                "postCode": "02030"
            },
            "shippingDate": "2018-01-30",
            "arrivalDate": "2018-01-30",
            "disposal": "false",
            "submittedDateTime": "2018-01-23T13:46:59-05:00",
            "shipmentDateTime": "2018-01-23T13:46:59-05:00",
            "shipmentStatus": "pending",
            "shipmentAmended": "true",
            "productStatusArray": [
                {
                    "productId": "CA121145-78FA-3234-B341-0078346A3489",
                    "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
                    "version": "1.2.3",
                    "function": "Game Software",
                    "productName": "A Better Product",
                    "productModel": "Super Duper",
                    "productType": "software",
                    "serialNumber": "ABV18012345",
                    "productDateTime": "2018-01-23T13:46:59-05:00",
                    "productStatus": "pending",
                    "productComponentArray": [
                        {
                            "componentId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
                            "componentNumber": "ABV_123456",
                            "version": "1.2.3",
                            "function": "Game Software",
                            "componentName": "A Better Game"
                        }
                    ]
                }
            ]
        }
    ]
}
```


7.1.3 PUT shipments Resource

The following table contains information about the `shipments` resource when the HTTP PUT verb is used. It includes the pathname and content type used to access the resource. When accessing the `shipments` resource using the HTTP PUT verb, the specified values MUST be used.

NEW CHAPTER

This resource is used to resubmit amended shipping requests to the regulator. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System. Upon receipt of an amended shipping request, the host system SHOULD reset the `shipmentStatus` to `pending` and set `shipmentAmended` to `true`.

Table 7.12 PUT shipments Resource Information

HTTP Method	`PUT`
Pathname	`/cdi/[ver]/shipments`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`shipment` Object. See Section 7.1.1.1, shipment Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

7.1.3.1 PUT shipments Example

The following example demonstrates the construction of a PUT shipments request and a response indicating that the request was accepted. In practice, additional HTTP headers may be included in the messages.

```
Request:

PUT /cdi/1.1/shipments HTTP/1.1
Content-Length: 1157
Content-Type: application/json; charset=utf-8

{
    "shipmentId": "1145346A-78FA-3234-B341-0078CA123489",
    "shipmentName": "ABV_987654",
    "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
    "shipperId": "456A3411-78FA-3234-B341-0078CA123489",
    "shipperAddress": {
        "addressName": "ABC Warehouse",
        "addressLine1": "1 Industrial Way",
        "addressLine2": "Building C",
        "addressCity": "Anywhere",
        "stateProvCode": "CT",
        "countryCode": "US",
        "postCode": "06355"
    },
    "receiverId": "3411456A-78FA-3234-B341-0078CA123489",
    "receiverAddress": {
        "addressName": "Big Casino",
        "addressLine1": "777 Lucky Street",
        "addressCity": "Somewhere",
        "stateProvCode": "MA",
        "countryCode": "US",
        "postCode": "02030"
    },
    "shippingDate": "2018-01-30",
    "arrivalDate": "2018-01-30",
    "disposal": "false",
    "productArray": [
        {
```

```
            "productId": "CA121145-78FA-3234-B341-0078346A3489",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "version": "1.2.3",
            "function": "Game Software",
            "productName": "A Better Product",
            "productModel": "Super Duper",
            "productType": "software",
            "serialNumber": "ABV18012345",
            "productComponentArray": [
                {
                    "componentId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
                    "componentNumber": "ABV_123456",
                    "version": "1.2.3",
                    "function": "Game Software",
                    "componentName": "A Better Game"
                }
            ]
        }
    ]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

NEW CHAPTER

7.2 receipts Resource

The `receipts` resource is used by a Client System to record the results of product shipments — that is, whether products were accepted by the receiver or rejected. It is intended that vendors and operators use this resource as Client Systems to report the results of product shipments to regulators acting as Host Systems. Client Systems can also use this resource to retrieve receiving information from Host Systems.

Table 7.13 receipts HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
`/cdi/[ver]/receipts`	No	Yes	Yes	No

7.2.1 PUT receipts Resource

The following table contains information about the `receipts` resource when the HTTP PUT verb is used. It includes the pathname and content type used to access the resource. When accessing the `receipts` resource using the HTTP PUT verb, the specified values MUST be used.

The Host System SHOULD NOT accept updates to product receiving information through this resource unless the `shipmentStatus` property of the shipping request is set to `approved`.

Table 7.14 PUT receipts Resource Information

HTTP Method	`PUT`
Pathname	`/cdi/[ver]/receipts`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`receipt` Object. See Section 7.2.1.1, receipt Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

7.2.1.1 receipt Object

The following table identifies the properties of the `receipt` object. Additional properties MAY be included in the `receipt` object.

Table 7.15 receipt Properties

Property	Restrictions	Description
`shipmentId`	`type: t_shipmentId` `use: required`	UUID representing the shipment. **Example: 1145346A-78FA-3234-B341-0078CA123489**

Table 7.15 receipt Properties

Property	Restrictions	Description

`productReceiptArray`	`type: t_productReceipt`
	`use: required`
	`minItems: 1`
	`maxItems: ∞`
		Array of `productReceipt` objects for the shipment. See Section 7.2.1.2, productReceipt Object for details.

7.2.1.2 productReceipt Object

The following table identifies the properties of the `productReceipt` object. Additional properties MAY be included in the `productReceipt` object.

Table 7.16 productReceipt Properties

Property	Restrictions	Description
`productId`	`type: t_productId` `use: required`	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**
`productStatus`	`type: t_productStatuses` `use: required`	Product shipping status. **Example: accepted**

7.2.1.3 PUT receipts Example

The following example demonstrates the construction of a PUT receipts request and a response indicating that the request was accepted. In practice, additional HTTP headers may be included in the messages.

```
Request:

PUT /cdi/1.1/receipts HTTP/1.1
Content-Length: 160
Content-Type: application/json; charset=utf-8

{
    "shipmentId": "1145346A-78FA-3234-B341-0078CA123489",
    "productReceiptArray": [
        {
            "productId": "CA121145-78FA-3234-B341-0078346A3489",
            "productStatus": "accepted"
        }
    ]
}


Response:

HTTP/1.1 200 OK
Content-Length: 0
```

NEW CHAPTER

7.2.2 GET receipts Resource

The following table contains information about the `receipts` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `receipts` resource using the HTTP GET verb, the specified values MUST be used.

This resource can be used to retrieve information from Host Systems about updates to product receiving information.

Table 7.17 GET shipments Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/receipts`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`receiptStatuses` Object. See Section 7.2.2.2, receiptStatuses Object for details.

7.2.2.1 GET receipts Parameters

The following table identifies the parameters of the `receipts` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `shipmentId` parameter is included, only status information regarding products with the specified `shipmentId` are included in the response; otherwise, the `shipmentId` parameter is ignored.
- If the `productStatus` parameter is included, only status information regarding products with the specified `productStatus` are included in the response; otherwise, the `productStatus` parameter is ignored.
- If the `productStart` parameter is included, only information regarding products with `productDateTime` values greater than or equal to the specified `productStart` value are included in the response; otherwise, the `productStart` parameter is ignored.
- If the `productEnd` parameter is included, only information regarding products with `productDateTime` values less than or equal to the specified `productEnd` value are included in the response; otherwise, the `productEnd` parameter is ignored.
- If the `vendorId` parameter is included, only status information regarding products with the specified `vendorId` as the `shipperId` or `receiverId` are included in the response; otherwise, the `vendorId` parameter is ignored.
- If the `operatorId` parameter is included, only status information regarding products with the specified `operatorId` as the `shipperId` or `receiverId` are included in the response; otherwise, the `operatorId` parameter is ignored.
- If no parameters are included, status information regarding all products to which the Client System has access are included in the response.

If the included parameters result in no products being selected, the Host System MUST simply return an empty list of products to the Client System.

Table 7.18 GET receipts Parameters

Parameters	Restrictions	Description
shipmentId	type: t_shipmentId use: optional	UUID representing the shipment. **Example: 1145346A-78FA-3234-B341-0078CA123489**
productStatus	type: t_productStatuses use: optional	Product Shipping Status. **Example: accepted**
productStart	type: string format: dateTime use: optional	The earliest date/time for which changes to the product status are requested. **Example: 2018-01-23T00:00:00-05:00**
productEnd	type: string format: dateTime use: optional	The latest date/time for which changes to the product status are requested. **Example: 2018-01-23T23:59:59-05:00**
vendorId	type: t_vendorId use: required	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
operatorId	type: t_operatorId use: optional	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**

7.2.2.2 receiptStatuses Object

The following table identifies the properties of the `receiptStatuses` object. Additional properties MAY be included in the `receiptStatuses` object.

Table 7.19 receiptStatuses Properties

Property	Restrictions	Description
receiptStatusArray	type: t_shipmentStatus use: required minItems: 0 maxItems: ∞	Array of `shipmentStatus` Objects. See Section 7.1.2.3, shipmentStatus Objectfor details.

7.2.2.3 GET receipts Example

The following example demonstrates the construction of a GET receipts request and a response containing a `receiptStatuses` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.0/receipts?shipmentId=1145346A-78FA-3234-B341-0078CA123489 HTTP/1.1
Accept: application/json
```

```
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 1416
Content-Type: application/json; charset=utf-8

{
    "receiptStatusArray": [
        {
            "shipmentId": "1145346A-78FA-3234-B341-0078CA123489",
            "shipmentName": "ABV_987654",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
            "shipperId": "456A3411-78FA-3234-B341-0078CA123489",
            "shipperAddress": {
                "addressName": "ABC Warehouse",
                "addressLine1": "1 Industrial Way",
                "addressLine2": "Building C",
                "addressCity": "Anywhere",
                "stateProvCode": "CT",
                "countryCode": "US",
                "postCode": "06355"
            },
            "receiverId": "3411456A-78FA-3234-B341-0078CA123489",
            "receiverAddress": {
                "addressName": "Big Casino",
                "addressLine1": "777 Lucky Street",
                "addressCity": "Somewhere",
                "stateProvCode": "MA",
                "countryCode": "US",
                "postCode": "02030"
            },
            "shippingDate": "2018-01-30",
            "arrivalDate": "2018-01-30",
            "disposal": "false",
            "submittedDateTime": "2018-01-23T13:46:59-05:00",
            "shipmentDateTime": "2018-01-23T13:46:59-05:00",
            "shipmentStatus": "approved",
            "shipmentAmended": "true",
            "productStatusArray": [
                {
                    "productId": "CA121145-78FA-3234-B341-0078346A3489",
                    "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
                    "version": "1.2.3",
                    "function": "Game Software",
                    "productName": "A Better Product",
                    "productModel": "Super Duper",
                    "productType": "software",
                    "serialNumber": "ABV18012345",
                    "productDateTime": "2018-01-23T13:46:59-05:00",
                    "productStatus": "accepted",
                    "productComponentArray": [
                        {
                            "componentId": "96CA8F38-A692-4748-847A-A1DA00D2B95C",
                            "componentNumber": "ABV_123456",
                            "version": "1.2.3",
                            "function": "Game Software",
                            "componentName": "A Better Game"
                        }
```

NEW CHAPTER

```
                        ]
                    }
                ]
            }
        ]
}
```

NEW CHAPTER

7.3 inventory Resource

The `inventory` resource is used by a Client System to request product inventory information for operators — that is, a list of products currently at an operator's property. It is intended that vendors and operators use this resource as Client Systems to retrieve product inventory information from regulators acting as Host Systems.

When reporting product inventory information, only products that were accepted by the operator and not subsequently reshipped or disposed SHOULD be included. Likewise, the Host System SHOULD only allow updates to product inventory information for products that were accepted by the operator and not subsequently reshipped or disposed.

Table 7.20 inventory HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
`/cdi/[ver]/inventory`	No	Yes	No	No

7.3.1 GET inventory Resource

The following table contains information about the `inventory` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `inventory` resource using the HTTP GET verb, the specified values MUST be used.

Table 7.21 GET inventory Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/inventory`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`inventories` Object. See Section 7.3.1.2, inventories Object for details.

7.3.1.1 GET inventory Parameters

The following table identifies the parameters of the `inventory` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `operatorId` parameter is included, only inventory information for the specified `operatorId` is included in the response; otherwise, the `operatorId` parameter is ignored.
- If the `vendorId` parameter is included, only inventory information for products from the specified `vendorId` is included in the response; otherwise, the `vendorId` parameter is ignored.
- If the `productId` parameter is included, only inventory information for products with the specified `productId` is included in the response; otherwise, the `productId` parameter is ignored.

- If the `inventoryStart` parameter is included, only inventory information for products with `inventoryDateTime` values greater than or equal to the specified `inventoryStart` value are included in the response; otherwise, the `inventoryStart` parameter is ignored.
- If the `inventoryEnd` parameter is included, only inventory information for products with `inventoryDateTime` values less than or equal to the specified `inventoryEnd` value are included in the response; otherwise, the `inventoryEnd` parameter is ignored.
- If no parameters are included, information regarding all inventory information to which the Client System has access is included in the response.

If the included parameters result in no products being selected, the Host System MUST simply return an empty list of products to the Client System.

Table 7.22 GET inventory Parameters

Parameters	Restrictions	Description
`operatorId`	`type: t_operatorId` `use: optional`	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**
`vendorId`	`type: t_vendorId` `use: optional`	UUID representing the vendor. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`productId`	`type: t_productId` `use: optional`	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**

`inventoryStart`	`type: string`
	`format: dateTime`
	`use: optional`
		The earliest date/time for which changes to the inventory information are requested.
		Example: 2018-01-23T00:00:00-05:00

`inventoryEnd`	`type: string`
	`format: dateTime`
	`use: optional`
		The latest date/time for which changes to the inventory information are requested.
		Example: 2018-01-23T23:59:59-05:00

7.3.1.2 inventories Object

The following table identifies the properties of the `inventories` object. Additional properties MAY be included in the `inventories` object.

Table 7.23 inventories Properties

Property	Restrictions	Description

`inventoryArray`	`type: inventory`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `inventory` Objects. See Section 7.3.1.3, inventory Object for details.

7.3.1.3 inventory Object

The following table identifies the properties of the `inventory` object. Additional properties MAY be included in the `inventory` object.

Table 7.24 inventory Properties

Property	Restrictions	Description
`operatorId`	`type: t_operatorId` `use: required`	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**
`operatorName`	`type: t_name` `use: required`	The human-readable name of the operator. **Example: A Bettor Operator**
`operatorCode`	`type: t_code` `use: optional`	Host-specific code representing the operator. **Example: ABO**
`jurisdictionId`	`type: t_jurisdictionId` `use: required`	UUID representing the jurisdiction in which the operator is licensed. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**

`productInventoryArray`	`type: productInventory`
	`use: required`
	`minItems: 0`
	`maxItems: ∞`
		Array of `productInventory` objects for the shipment. See Section 7.3.1.4, productInventory Object for details.

7.3.1.4 productInventory Object

The following table identifies the properties of the `productInventory` object. Additional properties MAY be included in the `productInventory` object.

Table 7.25 productInventory Properties

Property	Restrictions	Description
`productId`	`type: t_productId` `use: required`	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**
`vendorId`	`type: t_vendorId` `use: required`	UUID representing the vendor of the component. **Example: 456A3411-78FA-3234-B341-0078CA123489**
`productName`	`type: t_name` `use: optional`	The human-readable name of the product, such as the name of a game. **Example: A Better Game**
`productModel`	`type: t_name` `use: required`	Model of the product. **Example: Super Duper**

NEW CHAPTER

Table 7.25 productInventory Properties

Property	Restrictions	Description
productType	type: t_productTypes use: required	Product Type. **Example: EGM**
serialNumber	type: t_name use: required	Serial Number of the product. **Example: ABV18012345**
inventoryDateTime	type: string format: dateTime use: required	Date/time that the Asset Number, Location, or Regulatory Identification Number was last updated. **Example: 2018-01-23T13:46:59-05:00**
assetName	type: t_name use: optional default: <empty>	Asset Number; operator-assigned. **Example: ABO_1234**
locationName	type: t_name use: optional default: <empty>	Location; operator-assigned. **Example: AA-12-17**
regulatorName	type: t_name use: optional default: <empty>	Regulatory Identification Number; regulator-assigned. **Example: ABR_654321**

7.3.1.5 GET inventory Example

The following example demonstrates the construction of a GET inventory request and a response containing an `inventories` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.1/inventory?operatorId=3411456A-78FA-3234-B341-0078CA123489 HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 620
Content-Type: application/json; charset=utf-8

{
    "inventoryArray": [
        {
            "operatorId": "3411456A-78FA-3234-B341-0078CA123489",
            "operatorName": "A Better Operator",
            "operatorCode": "ABO",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
            "productInventoryArray": [
                {
```

NEW CHAPTER


```
                "productId": "CA121145-78FA-3234-B341-0078346A3489",
                "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
                "version": "1.2.3",
                "function": "Game Software",
                "productName": "A Better Product",
                "productModel": "Super Duper",
                "productType": "software",
                "serialNumber": "ABV18012345",
                "inventoryDateTime": "2018-01-24T09:47:56-05:00",
                "assetName": "ABO_1234",
                "locationName": "AA-12-17",
                "regulatorName": "ABR_654321"
            }
        ]
    }
  ]
}
```

NEW CHAPTER

NEW CHAPTER

Chapter 8

Work Order Resources

Extension in v1.1

NEW CHAPTER

8.1 workOrders Resource

The `workOrders` resource is used by a Client System to submit work order requests for gaming products to a Host System and, subsequently, to retrieve status information (approvals/denials) about those requests back from the Host. It is intended that operators use this resource as Client Systems to submit work order requests to regulators acting as Host Systems. After a work order request has been submitted to a regulator, operators can use the GET verb to request status information about the request. Operators can use the PUT verb to resubmit an amended request.

Table 8.1 workOrders HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
`/cdi/[ver]/workOrders`	Yes	Yes	Yes	No

8.1.1 POST workOrders Resource

The following table contains information about the `workOrders` resource when the HTTP POST verb is used. It includes the pathname and content type used to access the resource. When accessing the `workOrders` resource using the HTTP POST verb, the specified values MUST be used.

This resource is used to submit new work order requests to the Host System. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System.

Host Systems SHOULD NOT accept duplicate work order requests when the POST verb is used — that is, two work order requests with the same `workOrderId` should not be accepted. The PUT verb can be used to resubmit an amended work order request with the same `workOrderId`.

Host Systems SHOULD NOT accept work orders for products that are not in an operator's inventory when the POST verb or the PUT verb is used — that is, products with the specified `productId` should already be in the operator's inventory.

Table 8.2 POST workOrders Resource Information

HTTP Method	`POST`
Pathname	`/cdi/[ver]/workOrders`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`workOrder` Object. See Section 8.1.1.1, workOrder Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

8.1.1.1 workOrder Object

The following table identifies the properties of the `workOrder` object. Additional properties MAY be included in the `workOrder` object.

NEW CHAPTER

The `workOrderId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `workOrder` record. Two `workOrder` records with the same `workOrderId` are considered duplicates.

Table 8.3 workOrder Properties

Property	Restrictions	Description
`workOrderId`	`type: t_workOrderId` `use: required`	UUID representing the work order. **Example: 1145346A-78FA-3234-B341-0078CA123489**
`workOrderName`	`type: t_name` `use: required`	The human-readable identification number for the work order request. **Example: ABO_654321**
`jurisdictionId`	`type: t_jurisdictionId` `use: required`	UUID representing the jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
`operatorId`	`type: t_operatorId` `use: required`	Operator Identifier. **Example: 456A3411-78FA-3234-B341-0078CA123489**

`estimatedDate`	`type: string`
	`format: date`
	`use: required`
		Estimated Work Date; date that the work is planned to be performed.
		Example: 2018-06-18

`workItemArray`	`type: workItem`
	`use: required`
	`minItems: 1`
	`maxItems: ∞`
		Array of `workItem` objects for the work order. See Section 8.1.1.2, workItem Object for details.

8.1.1.2 workItem Object

The following table identifies the properties of the `workItem` object. Additional properties MAY be included in the `workItem` object.

The `workItemId` MUST be a UUID generated in a manner compliant with the ISO/IEC 9834-8:2014 standard to guarantee uniqueness. It serves as the unique identifier for the `workItem` record. Two `workItem` records with the same `workItemId` are considered duplicates.

Table 8.4 workItem Properties

Property	Restrictions	Description
`workItemId`	`type: t_workItemId` `use: required`	UUID representing the work item. **Example: 346A1145-78FA-3234-B341-00783489CA12**
`productId`	`type: t_productId` `use: required`	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**

NEW CHAPTER

Table 8.4 workItem Properties

Property	Restrictions	Description
vendorId	type:t_vendorId use: required	UUID representing the vendor of the product. **Example: 456A3411-78FA-3234-B341-0078CA123489**
productName	type: t_name use: optional default: <empty>	The human-readable name of the product, such as the name of a game. **Example: A Better Game**
productModel	type: t_name use: optional default: <empty>	Model of the product. **Example: Super Duper**
productType	type: t_productTypes use: required	Product Type. **Example: EGM**
serialNumber	type: t_name use: required	Serial Number of the product. **Example: ABV18012345**
assetName	type:t_name use: optional default: <empty>	Asset Number; operator-assigned. **Example: ABO_1234**
locationName	type: t_name use: optional default: <empty>	Location; operator-assigned. **Example: AA-12-17**
regulatorName	type: t_name use: optional default: <empty>	Regulatory Identification Number; regulator-assigned. **Example: ABR_654321**
workType	type: t_workTypes use: required	Work Type; type of work to be performed. **Example: RL**
workName	type: t_name use: optional default: <empty>	Description of the work to be performed. **Example: Relocate to AA-01-01**
newAssetName	type: t_name use: optional default: <empty>	New Asset Number; <empty> implies no change. **Example: ABO_1234**
newLocationName	type: t_name use: optional default: <empty>	New Location; <empty> implies no change. **Example: AA-12-17**
newRegulatorName	type: t_name use: optional default: <empty>	New Regulatory Identification Number; <empty> implies no change. **Example: ABR_654321**

NEW CHAPTER

8.1.1.3 POST workOrders Example

The following example demonstrates the construction of a POST work order request and a response indicating that the work order request was accepted. In practice, additional HTTP headers may be included in the messages.

Request:

```
POST /cdi/1.1/workOrders HTTP/1.1
Content-Length: 685
Content-Type: application/json; charset=utf-8

{
    "workOrderId": "1145346A-78FA-3234-B341-0078CA123489",
    "workOrderName": "ABO_654321",
    "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
    "operatorId": "3411456A-78FA-3234-B341-0078CA123489",
    "estimatedDate": "2018-06-18",
    "workItemArray": [
        {
            "workItemId": "346A1145-78FA-3234-B341-00783489CA12",
            "productId": "CA121145-78FA-3234-B341-0078346A3489",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "productName": "A Better Product",
            "productModel": "Super Duper",
            "productType": "software",
            "serialNumber": "ABV18012345",
            "assetName": "ABO_1234",
            "locationName": "AA-12-17",
            "regulatorName": "ABR_654321",
            "workType": "RL",
            "workName": "Relocate to AA-01-01",
            "newLocationName": "AA-01-01"
        }
    ]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

8.1.2 GET workOrders Resource

The following table contains information about the `workOrders` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `workOrders` resource using the HTTP GET verb, the specified values MUST be used.

This resource can be used to retrieve status information from the Host System about previously submitted work order requests.

Table 8.5 GET workOrders Resource Information

HTTP Method	`GET`

NEW CHAPTER

Table 8.5 GET workOrders Resource Information

Pathname	`/cdi/[ver]/workOrders`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`workOrderStatuses` Object. See Section 8.1.2.2, workOrderStatuses Object for details.

8.1.2.1 GET workOrders Parameters

The following table identifies the parameters of the `workOrders` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `workOrderId` parameter is included, only status information regarding work orders with the specified `workOrderId` are included in the response; otherwise, the `workOrderId` parameter is ignored.
- If the `workOrderStatus` parameter is included, only status information regarding work orders with the specified `workOrderStatus` are included in the response; otherwise, the `workOrderStatus` parameter is ignored.
- If the `workOrderStart` parameter is included, only information regarding work orders with `workOrderDateTime` values greater than or equal to the specified `workOrderStart` value are included in the response; otherwise, the `workOrderStart` parameter is ignored.
- If the `workOrderEnd` parameter is included, only information regarding work orders with `workOrderDateTime` values less than or equal to the specified `workOrderEnd` value are included in the response; otherwise, the `workOrderEnd` parameter is ignored.
- If the `operatorId` parameter is included, only status information regarding work orders with the specified `operatorId` are included in the response; otherwise, the `operatorId` parameter is ignored.
- If no parameters are included, information regarding all work orders to which the Client System has access are included in the response.

If the included parameters result in no work orders being selected, the Host System MUST simply return an empty list of work orders to the Client System.

Table 8.6 GET workOrders Parameters

Parameters	Restrictions	Description
`workOrderId`	`type: t_workOrderId` `use: optional`	UUID representing the work order. **Example: 1145346A-78FA-3234-B341-0078CA123489**
`workOrderStatus`	`type: t_workOrderStatuses` `use: optional`	Work Order Status. **Example: pending**

`workOrderStart`	`type: string`
	`format: dateTime`
	`use: optional`
		The earliest date/time for which changes to the work order status are requested.
		Example: 2018-01-23T00:00:00-05:00

Table 8.6 GET workOrders Parameters

Parameters	Restrictions	Description
workOrderEnd	type: string format: dateTime use: optional	The latest date/time for which changes to the work order status are requested. **Example: 2018-01-23T23:59:59-05:00**
operatorId	type: t_operatorId use: optional	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**

8.1.2.2 workOrderStatuses Object

The following table identifies the properties of the `workOrderStatuses` object. Additional properties MAY be included in the `workOrderStatuses` object.

Table 8.7 workOrderStatuses Properties

Property	Restrictions	Description
workOrderStatusArray	type: workOrderStatus use: required minItems: 1 maxItems: ∞	Array of `workOrderStatus` Objects. See Section 8.1.2.3, workOrderStatus Object for details.

8.1.2.3 workOrderStatus Object

The following table identifies the properties of the `workOrderStatus` object. Additional properties MAY be included in the `workOrderStatus` object.

Table 8.8 workOrderStatus Properties

Property	Restrictions	Description
workOrderId	type: t_workOrderId use: required	UUID representing the work order. **Example: 1145346A-78FA-3234-B341-00783489CA12**
workOrderName	type: t_name use: required	The human-readable identification number for the work order request. Example: **ABO_654321**
jurisdictionId	type: t_jurisdictionId use: required	UUID representing the jurisdiction. **Example: CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2**
operatorId	type: t_operatorId use: required	Operator Identifier. **Example: 456A3411-78FA-3234-B341-0078CA123489**

NEW CHAPTER

Table 8.8 workOrderStatus Properties

Property	Restrictions	Description
estimatedDate	type: string format: date use: required	Estimated Work Date; date that the work is planned to be performed. **Example: 2018-06-18**
submittedDateTime	type: string format: dateTime use: required	Date/time that a new or amended work order request was received by the regulator. **Example: 2018-06-18T14:03:17-05:00**
workOrderDateTime	type: string format: dateTime use: required	Date/time that the Work Order Status was last updated. **Example: 2018-06-18T14:03:17-05:00**
workOrderStatus	type: t_workOrderStatuses use: required	Work Order Status. **Example: pending**
workOrderAmended	type: boolean use: optional default: false	Amendment indicator. **Example: true**
workItemStatusArray	type: workItemStatus use: required minItems: 1 maxItems: ∞	Array of workItemStatus objects for the work order. See Section 8.1.2.4, workItemStatus Object for details.

8.1.2.4 workItemStatus Object

The following table identifies the properties of the workItemStatus object. Additional properties MAY be included in the workItemStatus object.

Table 8.9 workItemStatus Properties

Property	Restrictions	Description
workItemId	type: t_workItemId use: required	UUID representing the work item. **Example: 346A1145-78FA-3234-B341-00783489CA12**
productId	type: t_productId use: required	UUID representing the product. **Example: CA121145-78FA-3234-B341-0078346A3489**
vendorId	type: t_vendorId use: required	UUID representing the vendor of the product. **Example: 456A3411-78FA-3234-B341-0078CA123489**
productName	type: t_name use: optional default: <empty>	The human-readable name of the product, such as the name of a game. **Example: A Better Game**

NEW CHAPTER

Table 8.9 workItemStatus Properties

Property	Restrictions	Description
productModel	type: t_name use: optional default: <empty>	Model of the product. **Example: Super Duper**
productType	type: t_productTypes use: required	Product Type. **Example: EGM**
serialNumber	type: t_name use: required	Serial Number of the product. **Example: ABV18012345**
assetName	type: t_name use: optional default: <empty>	Asset Number; operator-assigned. **Example: ABO_1234**
locationName	type: t_name use: optional default: <empty>	Location; operator-assigned. **Example: AA-12-17**
regulatorName	type: t_name use: optional default: <empty>	Regulatory Identification Number; regulator-assigned. **Example: ABR_654321**
workType	type: t_workTypes use: required	Work Type; type of work to be performed. **Example: RL**
workName	type: t_name use: optional default: <empty>	Description of the work to be performed. **Example: Relocate to AA-01-01**
newAssetName	type: t_name use: optional default: <empty>	New Asset Number; <empty> implies no change. **Example: ABO_1234**
newLocationName	type: t_name use: optional default: <empty>	New Location; <empty> implies no change. **Example: AA-12-17**
newRegulatorName	type: t_name use: optional default: <empty>	New Regulatory Identification Number; <empty> implies no change. **Example: ABR_654321**
workItemDateTime	type: string format: dateTime use: required	Date/time that the Work Item Status was last updated. **Example: 2018-01-23T13:46:59-05:00**
workItemStatus	type: t_workItemStatuses use: required	Work Item Status. **Example: pending**

NEW CHAPTER

8.1.2.5 GET workOrders Example

The following example demonstrates the construction of a GET work orders request and a response containing a `workOrderStatuses` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.1/workOrders?workOrderId=1145346A-78FA-3234-B341-0078CA123489 HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 928
Content-Type: application/json; charset=utf-8

{
    "workOrderStatusArray": [
        {
            "workOrderId": "1145346A-78FA-3234-B341-0078CA123489",
            "workOrderName": "ABO_654321",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
            "operatorId": "3411456A-78FA-3234-B341-0078CA123489",
            "estimatedDate": "2018-06-18",
            "submittedDateTime": "2018-06-18T14:03:17-05:00",
            "workOrderDateTime": "2018-06-18T14:03:17-05:00",
            "workOrderStatus": "pending",
            "workItemStatusArray": [
                {
                    "workItemId": "346A1145-78FA-3234-B341-00783489CA12",
                    "productId": "CA121145-78FA-3234-B341-0078346A3489",
                    "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
                    "productName": "A Better Product",
                    "productModel": "Super Duper",
                    "productType": "software",
                    "serialNumber": "ABV18012345",
                    "assetName": "ABO_1234",
                    "locationName": "AA-12-17",
                    "regulatorName": "ABR_654321",
                    "workType": "RL",
                    "workName": "Relocate to AA-01-01",
                    "newLocationName": "AA-01-01"
                    "workItemDateTime": "2018-06-18T14:03:17-05:00",
                    "workItemStatus": "pending",
                }
            ]
        }
    ]
}
```

NEW CHAPTER

8.1.3 PUT workOrders Resource

The following table contains information about the `workOrders` resource when the HTTP PUT verb is used. It includes the pathname and content type used to access the resource. When accessing the `workOrders` resource using the HTTP PUT verb, the specified values MUST be used.

This resource is used to resubmit amended work orders to the regulator. The requests MAY not be accepted immediately. A manual review may be required before the requests are entered into the Host System.

Upon receipt of an amended shipping request, the host system SHOULD reset the `workOrderStatus` to pending and set `workOrderAmended` to `true`.

Table 8.10 PUT workOrders Resource Information

HTTP Method	`PUT`
Pathname	`/cdi/[ver]/workOrders`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`workOrder` Object. See Section 8.1.1.1, workOrder Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

8.1.3.1 PUT workOrders Example

The following example demonstrates the construction of a PUT `workOrders` request and a response indicating that the request was accepted. In practice, additional HTTP headers may be included in the messages.

Request:

```
PUT /cdi/1.1/workOrders HTTP/1.1
Content-Length: 685
Content-Type: application/json; charset=utf-8

{
    "workOrderId": "1145346A-78FA-3234-B341-0078CA123489",
    "workOrderName": "ABO_654321",
    "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
    "operatorId": "3411456A-78FA-3234-B341-0078CA123489",
    "estimatedDate": "2018-06-18",
    "workItemArray": [
        {
            "workItemId": "346A1145-78FA-3234-B341-00783489CA12",
            "productId": "CA121145-78FA-3234-B341-0078346A3489",
            "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
            "productName": "A Better Product",
            "productModel": "Super Duper",
            "productType": "software",
            "serialNumber": "ABV18012345",
            "assetName": "ABO_1234",
            "locationName": "AA-12-17",
            "regulatorName": "ABR_654321",
            "workType": "RL",
            "workName": "Relocate to AA-01-01",
            "newLocationName": "AA-01-01"
        }
    ]
```

```
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

NEW CHAPTER

8.2 completions Resource

The `completions` resource is used by a Client System to record the results of work orders — that is, whether specific work items were completed by an operator as planned. It is intended that operators use this resource as Client Systems to report the results of work orders to regulators acting as Host Systems. Client Systems can also use this resource to retrieve completion information from Host Systems.

Table 8.11 completions HTTP Verbs

Resource	HTTP Verbs			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
`/cdi/[ver]/completions`	No	Yes	Yes	No

8.2.1 PUT completions Resource

The following table contains information about the `completions` resource when the HTTP PUT verb is used. It includes the pathname and content type used to access the resource. When accessing the `completions` resource using the HTTP PUT verb, the specified values MUST be used.

The Host System SHOULD NOT accept updates to work order completion information through this resource unless the `workOrderStatus` property of the work order is set to `approved`.

Table 8.12 PUT completions Resource Information

HTTP Method	`PUT`
Pathname	`/cdi/[ver]/completions`
Request Content-Type	`application/json; charset=utf-8`
Request Content	`completion` Object. See Section 8.2.1.1, completion Object for details.
Response Content-Type	`application/json; charset=utf-8`
Response Content	None.

8.2.1.1 completion Object

The following table identifies the properties of the `completion` object. Additional properties MAY be included in the `completion` object.

Table 8.13 completion Properties

Property	Restrictions	Description
`workOrderId`	`type: t_workOrderId` `use: required`	UUID representing the work order. **Example: 1145346A-78FA-3234-B341-0078CA123489**

NEW CHAPTER

Table 8.13 completion Properties

Property	Restrictions	Description
workItemCompletionArray	type: workItemCompletion use: required minItems: 1 maxItems: ∞	Array of workItemCompletion objects for the shipment. See Section 8.2.1.2, workItemCompletion Object for details.

8.2.1.2 workItemCompletion Object

The following table identifies the properties of the workItemCompletion object. Additional properties MAY be included in the workItemCompletion object.

Table 8.14 workItemcompletion Properties

Property	Restrictions	Description
workItemId	type: t_workItemId use: required	UUID representing the work item. **Example: 346A1145-78FA-3234-B341-00783489CA12**
workItemStatus	type: t_workItemStatuses use: required	Work Item Status. **Example: completed**

8.2.1.3 PUT completions Example

The following example demonstrates the construction of a PUT completions request and a response indicating that the request was accepted. In practice, additional HTTP headers may be included in the messages.

Request:

```
PUT /cdi/1.1/completions HTTP/1.1
Content-Length: 169
Content-Type: application/json; charset=utf-8

{
    "workOrderId": "1145346A-78FA-3234-B341-0078CA123489",
    "workItemCompletionArray": [
        {
            "workItemId": "346A1145-78FA-3234-B341-00783489CA12",
            "workItemStatus": "completed"
        }
    ]
}
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
```

NEW CHAPTER

8.2.2 GET completions Resource

The following table contains information about the `completions` resource when the HTTP GET verb is used. It includes the pathname and content type used to access the resource. When accessing the `completions` resource using the HTTP GET verb, the specified values MUST be used.

This resource can be used to retrieve information from Host Systems about updates to work order completion information.

Table 8.15 GET completions Resource Information

HTTP Method	`GET`
Pathname	`/cdi/[ver]/completions`
Request Content-Type	`application/json; charset=utf-8`
Request Content	None.
Response Content-Type	`application/json; charset=utf-8`
Response Content	`completionStatuses` Object. See Section 8.2.2.2, completionStatuses Object for details.

8.2.2.1 GET completions Parameters

The following table identifies the parameters of the `completions` resource when the HTTP GET verb is used. The parameters are appended to the resource URI in the query component of the HTTP request.

- If the `workOrderId` parameter is included, only status information regarding work orders with the specified `workOrderId` are included in the response; otherwise, the `workOrderId` parameter is ignored.
- If the `workItemStatus` parameter is included, only status information regarding work orders with the specified `workItemStatus` are included in the response; otherwise, the `workItemStatus` parameter is ignored.
- If the `workItemStart` parameter is included, only status information regarding work orders with `workItemDateTime` values greater than or equal to the specified `workItemStart` value are included in the response; otherwise, the `workItemStart` parameter is ignored.
- If the `workItemEnd` parameter is included, only status information regarding work orders with `workItemDateTime` values less than or equal to the specified `workItemEnd` value are included in the response; otherwise, the `workItemEnd` parameter is ignored.
- If the `operatorId` parameter is included, only status information regarding work orders with the specified `operatorId` are included in the response; otherwise, the `operatorId` parameter is ignored.
- If no parameters are included, status information regarding all work orders to which the Client System has access are included in the response.

If the included parameters result in no products being selected, the Host System MUST simply return an empty list of products to the Client System.

Table 8.16 GET completions Parameters

Parameters	Restrictions	Description
workOrderId	type: t_workOrderId use: optional	UUID representing the work order. **Example: 1145346A-78FA-3234-B341-0078CA123489**
workItemStatus	type: t_workItemStatuses use: optional	Work Order Status. **Example: completed**
workItemStart	type: string format: dateTime use: optional	The earliest date/time for which changes to the work item status are requested. **Example: 2018-01-23T00:00:00-05:00**
workItemEnd	type: string format: dateTime use: optional	The latest date/time for which changes to the work item status are requested. **Example: 2018-01-23T23:59:59-05:00**
operatorId	type: t_operatorId use: optional	UUID representing the operator. **Example: 3411456A-78FA-3234-B341-0078CA123489**

8.2.2.2 completionStatuses Object

The following table identifies the properties of the `completionStatuses` object. Additional properties MAY be included in the `completionStatuses` object.

Table 8.17 completionStatuses Properties

Property	Restrictions	Description
completionStatusArray	type: workOrderStatus use: required minItems: 1 maxItems: ∞	Array of `workOrderStatus` objects for the shipment. See Section 8.2.1.2, workItemCompletion Object for details.

8.2.2.3 GET completions Example

The following example demonstrates the construction of a GET completions request and a response containing a `completionStatuses` object. In practice, additional HTTP headers may be included in the messages.

Request:

```
GET /cdi/1.1/completions?workOrderId=1145346A-78FA-3234-B341-0078CA123489 HTTP/1.1
Accept: application/json
Accept-Charset: utf-8
```

Response:

NEW CHAPTER

```
HTTP/1.1 200 OK
Content-Length: 928
Content-Type: application/json; charset=utf-8

{
    "completionStatusArray": [
        {
            "workOrderId": "1145346A-78FA-3234-B341-0078CA123489",
            "workOrderName": "ABO_654321",
            "jurisdictionId": "CFFFC5A7-38BE-4351-9A7C-D8A27D7C0BF2",
            "operatorId": "3411456A-78FA-3234-B341-0078CA123489",
            "estimatedDate": "2018-06-18",
            "submittedDateTime": "2018-06-18T14:03:17-05:00",
            "workOrderDateTime": "2018-06-18T14:03:17-05:00",
            "workOrderStatus": "pending",
            "workItemStatusArray": [
                {
                    "workItemId": "346A1145-78FA-3234-B341-00783489CA12",
                    "productId": "CA121145-78FA-3234-B341-0078346A3489",
                    "vendorId": "456A3411-78FA-3234-B341-0078CA123489",
                    "productName": "A Better Product",
                    "productModel": "Super Duper",
                    "productType": "software",
                    "serialNumber": "ABV18012345",
                    "assetName": "ABO_1234",
                    "locationName": "AA-12-17",
                    "regulatorName": "ABR_654321",
                    "workType": "RL",
                    "workName": "Relocate to AA-01-01",
                    "newLocationName": "AA-01-01"
                    "workItemDateTime": "2018-06-18T14:03:17-05:00",
                    "workItemStatus": "pending",
                }
            ]
        }
    ]
}
```

NEW CHAPTER

Chapter 9

Data Types

9.1 Defined Data Types

The following table describes the data types defined within this specification.

Table 9.1 Defined Data Types

Data Type	Restrictions	Description
t_algorithmId	type: t_UUID	Algorithm identifier.
t_algorithmTypes	type: t_code	Algorithm types.
t_calculationId	type: t_UUID	Calculation request identifier.
t_calcStatuses	type: t_code enumerations: Pending Completed Failed	Calculation Statuses.
t_certificationId	type: t_UUID	Certification request identifier.
t_code	type: string maxLength: 32	Human-readable code or identifier.
t_componentId	type: t_UUID	Component identifier.
t_documentId	type: t_UUID	Document identifier.
t_jurisdictionId	type: t_UUID	Jurisdiction identifier.
t_language	type: t_code	Language code. ISO 639-1 2-character alphabetic codes.
t_name	type: string maxLength: 32	Human-readable name or description.
t_noteId	type: t_UUID	Note identifier.
t_notes	type: string maxLength: 1024	Human-readable notes.
t_paytableCI	type: string maxLength: 32	Confidence index.
t_paytableId	type: t_UUID	Paytable identifier.
t_paytableVI	type: string maxLength: 32	Volatility index.
t_percent	type: numeric minimum: 0 multipleOf: 1	Percentage; expressed in hundredths of a percent; for example, 98.75% is expressed as 9875.
t_salt	type: string maxLength: 1024	Salt value.
t_seed	type: string maxLength: 256	Seed value.
t_signatureId	type: t_UUID	Signature identifier.

Table 9.1 Defined Data Types

Data Type	Restrictions	Description
t_statusId	type: t_UUID	Status identifier.
t_submissionId	type: t_UUID	Jurisdiction submission identifier.
t_subStatuses	type: t_code enumerations: Pending Accepted Rejected	Submission Statuses.
t_testLabCert	type: t_name	Test laboratory certification number.
t_testLabId	type: t_UUID	Test laboratory identifier.
t_UUID	type: string maxLength: 36	A UUID generated in a manner compliant with ISO/IEC 9834-8:2014 to guarantee uniqueness.
t_vendorId	type: t_UUID	Vendor identifier.
t_verifyResult	type: string maxLength: 256	Software signature.
t_version	type: t_name	Version number.
Extension in v1.1		
t_countryCode	type: string maxLength: 2	Country code; MUST conform to ISO 3166-1 alpha-2 standard (2 characters); for example, US.
t_description	type: string maxLength: 128	Description; 128-characters.
t_operatorId	type: t_UUID	Operator Identifier.
t_postCode	type: string maxLength: 8	Postal Code.
t_productId	type: t_UUID	Product Identifier.
t_productStatuses	type: t_code enumerations: pending accepted rejected	Product Statuses; products must remain in the pending state until the shipping request is approved; once approved, the product status can be changed to another value; accepted means that the shipment or disposal took place; rejected means that the shipment or disposal did not take place.
t_productTypes	type: t_code enumerations: controller egm peripheral software tableGame	Product Types.
t_shipmentId	type: t_UUID	Shipment Identifier.

Table 9.1 Defined Data Types

Data Type	Restrictions	Description
t_shipmentStatuses	type: t_code enumerations: pending approved denied	Shipment Status.
t_stateProvCode	type: string maxLength: 3	State/Province Code; MUST conform to ISO 3166-2 standard (3 character); for example, CT.
t_workItemId	type: t_UUID	Work Item Identifier.
t_workItemStatuses	type: t_code enumerations: pending completed notDone	Work Item Status; work items must remain in the pending state until the work order request is approved; once approved, the work item status can be changed to another value; completed means that the work item was performed; notDone means that the work item was not performed.
t_workOrderId	type: t_UUID	Work Order Identifier.
t_workOrderStatuses	type: t_code enumerations: pending approved denied	Work Order Status.
t_workTypes	type: t_code enumerations: IN - installation RM - removal RP - repair MT - maintenance MU - mandatory upgrade NU - non-mandatory upgrade RC - reconfigure RL - relocate	Work Type.

9.2 JSON Data Types

The following table identifies the JSON data types referenced in this specification. End-entities MUST comply with the requirements contained in the JSON specification for these data types. See the JSON Specification for more details about these data types. Accompanying each data type, there is a list of restrictions used within this specification as well as the requirements associated with the restrictions. End-entities MUST comply with the requirements specified for these restrictions.

Table 9.2 JSON Data Types

JSON Data Type	Restrictions	Requirements
`numeric`	`type: required`	Indicates that the property MUST be included in the object.
	`type: optional`	Indicates that the property MAY be omitted from the object.
	`default: value`	If the property is omitted, the recipient MUST use the value specified as the value for the property.
	`minimum: value`	The value of the property MUST NOT be less than the specified value.
	`maximum: value`	The value of the property MUST NOT be greater than the specified value.
	`multipleOf: value`	The value of the property MUST be a multiple of the specified value.
`string`*	`type: required`	Indicates that the property MUST be included in the object.
	`type: optional`	Indicates that the property MAY be omitted from the object.
	`default: value`	If the property is omitted, the recipient MUST use the value specified as the value for the property.
	`format: uri`	The value of the property MUST comply with the format for Universal Resource Identifiers as specified in RFC 3986.
	`format: date`	The value of the property MUST comply with the full-date format specified in RFC 3339.
	`format: date-time`	The value of the property MUST comply with the date-time format specified in RFC 3339.
	`maxLength: value`	The number of characters contained in the property MUST NOT be greater than the specified value.
	`enumerations: values`	The value of the property SHOULD be one of the specified values; to avoid interoperability issues, other values SHOULD NOT be used.

Table 9.2 JSON Data Types

JSON Data Type	Restrictions	Requirements
`array`	`type: required`	Indicates that the array MUST be included in the object.
	`type: optional`	Indicates that the array MAY be omitted from the object.
	`minItems: value`	The number of entries in the array MUST NOT be less than the specified value.
	`maxItems: value`	The number of entries in the array MUST NOT be greater than the specified value.
`boolean`	`type: required`	Indicates that the property MUST be included in the object.
	`type: optional`	Indicates that the property MAY be omitted from the object.
	`default: value`	If the property is omitted, the recipient MUST use the *`value`* specified as the value for the property.

* Note: Reserved JSON characters, which appear in strings, must be escaped. See the JSON specification for more detail.

END OF DOCUMENT

Released: 2020/03/23