OASIS Committee Note

UBL 2.3 JSON Alternative Representation Version 1.0

Committee Note 01

01 December 2021

This stage:
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/UBL-2.3-JSON-v1.0-cn01.html
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/UBL-2.3-JSON-v1.0-cn01.pdf
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/UBL-2.3-JSON-v1.0-cn01.xml (Authoritative)

Previous stage:
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cnd02/UBL-2.3-JSON-v1.0-cnd02.html
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cnd02/UBL-2.3-JSON-v1.0-cnd02.pdf
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cnd02/UBL-2.3-JSON-v1.0-cnd02.xml (Authoritative)

Latest stage:
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/UBL-2.3-JSON-v1.0.html
https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/UBL-2.3-JSON-v1.0.pdf

Technical Committee:
OASIS Universal Business Language TC

Chairs:
Kenneth Bengtsson (kbengtsson@efact.pe), Individual
G. Ken Holman (gkholman@CraneSoftwrights.com), Crane Softwrights Ltd.

Editors:
Kenneth Bengtsson (kenneth@alfa1lab.com), Individual
Erlend Klakegg Bergheim (erlend.klakegg.bergheim@difi.no), Norwegian Digitalisation Agency
G. Ken Holman (gkholman@CraneSoftwrights.com), Crane Softwrights Ltd.

Additional artefacts:
This prose specification is one component of a Work Product that also includes:

- JSON legacy instance examples:
  - https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/json-legacy/
- JSON model instance examples:
  - https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/json-model/
- JSON legacy schemas:

    - https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/json-schema-legacy/

- JSON model schemas:
    - https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/json-schema-model/
- Default validation test environment:
    - https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/val/

The ZIP containing the complete files of this release is:

- https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/UBL-2.3-JSON-v1.0-cn01.zip

Related work:

This note is related to:

Universal Business Language Version 2.3. Edited by G. Ken Holman. 15 June 2021. OASIS Standard. https://docs.oasis-open.org/ubl/UBL-2.3.html.

This note is an implementation of the rules for JSON specified in:

[**BDNDR**] *Business Document Naming and Design Rules (BDNDR) Version 1.1.* Edited by Kenneth Bengtsson, Erlend Klakegg Bergheim and G. Ken Holman. 08 November 2021. Committee Specification 01. https://docs.oasis-open.org/ubl/Business-Document-NDR/v1.1/cs01/Business-Document-NDR-v1.1-cs01.html. Latest stage: https://docs.oasis-open.org/ubl/Business-Document-NDR/v1.1/Business-Document-NDR-v1.1.html.

Abstract:

This committee note supplements the OASIS Universal Business Language version 2.3 release with an alternative expression of the UBL sample XML documents in JSON syntax, and two JSON schema expressions of all 91 XSD schemas in conformance to the OASIS Business Document Naming and Design Rules Version 1.1.

Status:

This document was last revised or approved by the OASIS Universal Business Language TC on the above date. The level of approval is also listed above. Check the “Latest stage” location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ubl#technical.

TC members should send comments on this specification to the TC’s email list. Others should send comments to the TC’s public comment list, after subscribing to it by following the instructions at the “Send A Comment” button on the TC’s web page at https://www.oasis-open.org/committees/ubl/.

See Appendix A, *Release Notes* for more information regarding this release package.

Citation format:

When referencing this note the following citation format should be used:

[**UBL-2.3-JSON**] *UBL 2.3 JSON Alternative Representation Version 1.0.* Edited by Kenneth Bengtsson, Erlend Klakegg Bergheim and G. Ken Holman. 01 December 2021. OASIS Committee Note 01. https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/UBL-2.3-JSON-v1.0-cn01.html. Latest stage: https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/UBL-2.3-JSON-v1.0.html.

Notices

Table of Contents

1 Introduction

1.1 Overview

The OASIS Universal Business Language (UBL) defines a generic XML interchange format for business documents that can be restricted or extended to meet the requirements of particular industries. UBL includes a suite of XML Schemas with which one can validate the structural content of an XML document against the constraints of the vocabulary.

For users of JSON syntax, this note publishes two suites of JSON schemas with which one can validate the structural content of a JSON document against the constraints of the UBL 2.3 vocabulary. Also included are two transliterations of all of the UBL 2.3 example documents in JSON syntax with which one can test a number of the JSON schemas.

The structural patterns exhibited by JSON schemas that conform to the OASIS Business Document Naming and Design Rules Version 1.1 **[BDNDR]** are distinctive as document interchange structures. As such, their intent is only to convey in syntax the information content reflecting the same abstract model of the UN/CEFACT Core Component Technical Specification 2.01 **[CCTS]** with which the document model was designed. The rules govern two possible JSON schema structures, a legacy-based approach supporting future backwards compatibility, and a model-based approach specific to this version of UBL. Accordingly, and in parallel to an application's use of XML syntax, the legacy-based JSON syntax used is generic in nature, supported by schemas of all versions of UBL. The model-based JSON syntax is suitable only for schemas conforming to the UBL 2.3 structures. Neither syntax is streamlined nor optimized for any particular application's objectives

As one would undertake the unmarshalling of XML syntax into internal application data structures suitable for processing, one must also undertake the unmarshalling of JSON interchange syntax into whatever internal application data structures (or other JSON representations) of the content that are suitable for the task at hand. Of note, it has been observed that there are commercial JSON database tools unable to ingest this JSON interchange syntax directly without an application massaging the content first to suit the database schema necessary to enable a particular arbitrary use. Nevertheless, the JSON syntax used does conform to the published standard **[ISO 21778 - ECMA JSON]** and has been successfully demonstrated to be ingested by Python and Node.js applications and so is not a barrier to use for application developers.

The UBL Technical Committee makes no assumptions regarding how the XML syntax of UBL documents is to be managed by applications, and so similarly makes no assumptions regarding how any application's use of the JSON syntax is to be managed. With the only objective being the lossless interchange of document content, the structures in both XML and legacy-based JSON syntax are designed with forward and backward compatibility in mind for applications that ingest the content. Certain constraints in future versions of UBL may be lessened in a backward compatible fashion, such as in regard to cardinality, and so applications can successfully ingest the arrays of objects used in this regard to accommodate changing cardinality. Moreover, transliteration can be accomplished without semantic interpretation of any contents that would require specialized treatment of any given construct.

In contrast, model-based JSON syntax is not designed with forward and backward compatibility due to different conventions for expressing structures of different maximum cardinality. Where a CCTS construct is specified to have an unbounded maximum cardinality, an array is used in the same way arrays are used in legacy-based JSON syntax. Where a CCTS construct is specified to have a bounded maximum cardinality of "1", the structure is serialized without using an array. Accordingly, the serialization of JSON or the transliteration from XML requires knowledge of the maximum cardinality of all constructs in order to use the appropriate structures. When a version of UBL changes a construct's maximum cardinality (which can be from bounded to unbounded only), a past instance no longer will validate with the new schemas. This breaks forward compatibility from the past instance's perspective, and backward compatibility from a new instance's perspective.

The section Section A.3, “Package Structure” outlines the top-level content of the ZIP file associated with this Committee Note. The directories are named such that the directories in this package can be overlaid on top of the directories of the UBL distribution without overwriting any pre-existing UBL file. In another environment, one would copy the entire `json-schema-legacy/` subdirectory or `json-schema-model/` subdirectory contents and then users would access the individual document schemas in the `maindoc/` subdirectory. These schemas make relative references to shared fragments found in the `common/` subdirectory and as such those shared fragments are not referenced directly by users as they are not well-formed schemas in their own right (there is no “`$schema`” property).

The UBL document schemas make provision for constraining UBL extension metadata but not for constraining extension content. The user wishing to validate extension content is expected to replace the `common/UBL-ExtensionContentDataType-2.3.json` fragment with one’s own specification of extension constraints. No other schema fragment is expected to be touched by users.

1.2 Terminology

1.2.1 Terms and Definitions

Document

A set of information components that are exchanged as part of a business transaction; for example, in placing an order.

JSON schema

A JSON document [**ISO 21778 - ECMA JSON**] definition conforming to the JSON schema language [**JSON Schema**].

XSD schema

An XML document [**XML**] definition conforming to the W3C XML Schema language [**XSD1**] [**XSD2**].

1.2.2 Symbols and Abbreviations

JSON

Java Script Object Notation [**ISO 21778 - ECMA JSON**]

XML

Extensible Markup Language [**XML**]

XSD

W3C XML Schema Language [**XSD1**][**XSD2**]

1.3 References

[**BDNDR**] *Business Document Naming and Design Rules (BDNDR) Version 1.1.* Edited by Kenneth Bengtsson, Erlend Klakegg Bergheim and G. Ken Holman. 08 November 2021. Committee Specification 01. https://docs.oasis-open.org/ubl/Business-Document-NDR/v1.1/cs01/Business-Document-NDR-v1.1-cs01.html. Latest stage: https://docs.oasis-open.org/ubl/Business-Document-NDR/v1.1/Business-Document-NDR-v1.1.html.

[**CCTS**] *UN/CEFACT Core Component Technical Specification - Part 8 of the ebXML Framework)* 15 November 2003 Version 2.01 *http://www.unece.org/fileadmin/DAM/cefact/codesfortrade/CCTS/CCTS_V2-01_Final.pdf*

[**ISO 21778 - ECMA JSON**] *ISO/IEC 21778 Information technology — The JSON data interchange format* *http://standards.iso.org/ittf/PubliclyAvailableStandards/*, *ECMA 404 The JSON data interchange format* *https://www.ecma-international.org/publications-and-standards/standards/Ecma-404/*

[**JSON Schema**] *JSON Schema Validation: A Vocabulary for Structural Validation of JSON*, A. Wright, G. Luff, Editors, *http://json-schema.org/latest/json-schema-validation.html*

[**XML**] *Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000*

[**XSD1**] *XML Schema Part 1: Structures. Second Edition. W3C Recommendation 28 October 2004*

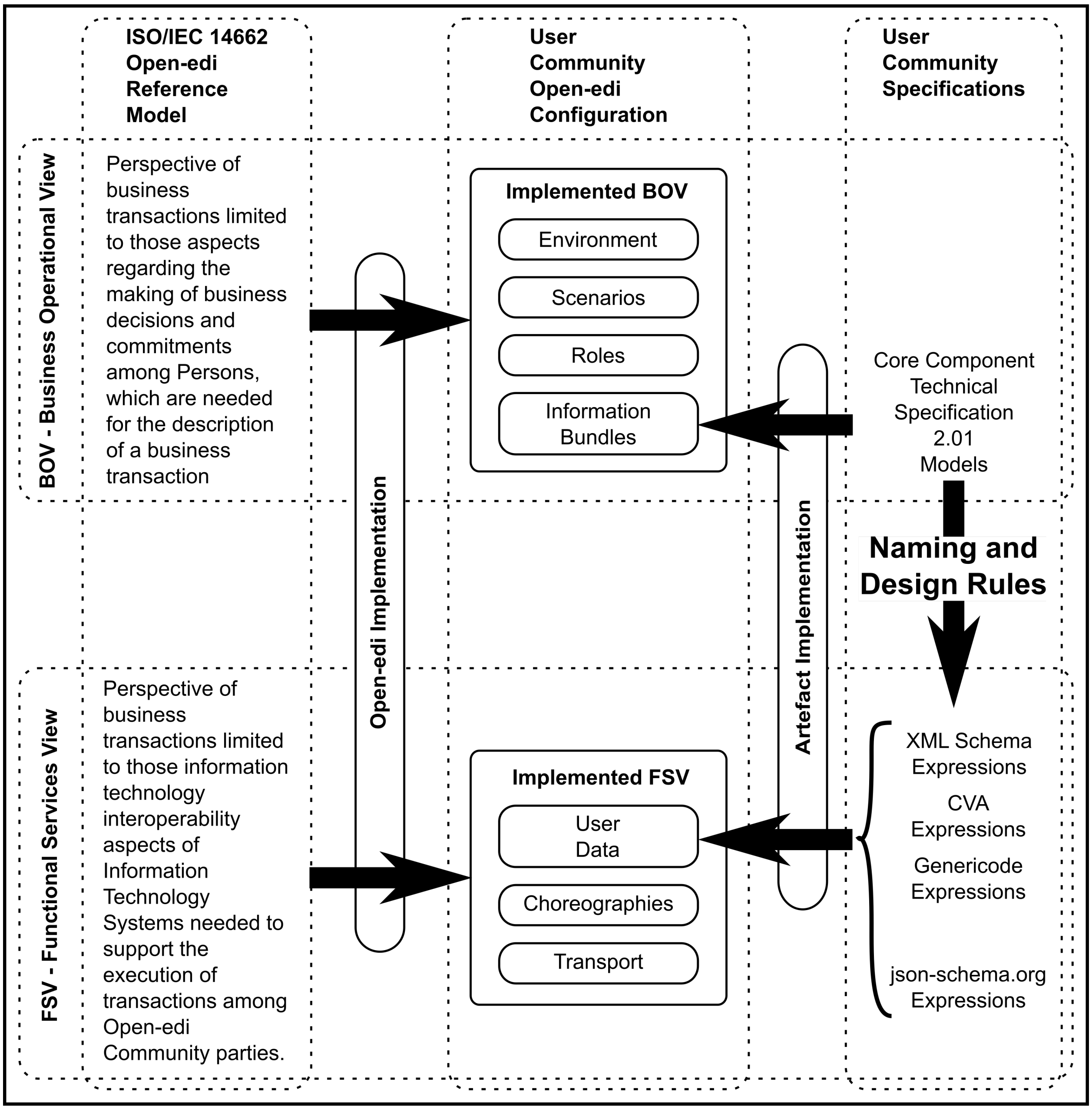
[**XSD2**] *XML Schema Part 2: Datatypes. Second Edition. W3C Recommendation 28 October 2004*

2 CCTS and JSON

2.1 CCTS component review

The UBL Technical Committee uses the UN/CEFACT Core Components Technical Specification (CCTS) 2.01 **[CCTS]** to create the semantic models of business documents. These abstract models are then used to generate concrete syntax models and the validation artefacts with which to constrain instances of that syntax. There is nothing in the model itself that is specific to any syntax. Naming and design rules **[BDNDR]** for building the models (as information bundles) and for generating the syntax validation artefacts for documents (as user data) are used by the committee to create the deliverables for users. This is depicted in Figure 1, "Naming and Design Rules in an Open-edi Application".

Figure 1. Naming and Design Rules in an Open-edi Application


ISO/IEC 14662 Open-edi Reference Model
User Community Open-edi Configuration
User Community Specifications
BOV - Business Operational View
Perspective of business transactions limited to those aspects regarding the making of business decisions and commitments among Persons, which are needed for the description of a business transaction
Implemented BOV
Environment
Scenarios
Roles
Information Bundles
Core Component Technical Specification 2.01 Models
Open-edi Implementation
Artefact Implementation
Naming and Design Rules
FSV - Functional Services View
Perspective of business transactions limited to those information technology interoperability aspects of Information Technology Systems needed to support the execution of transactions among Open-edi Community parties.
Implemented FSV
User Data
Choreographies
Transport
XML Schema Expressions
CVA Expressions
Genericode Expressions
json-schema.org Expressions

When modeling, CCTS governs how to define a class of business documents by specifying the available components as information bundles to be deployed to users as user data. An ABIE (aggregate) component specifies a collection of constituent components of ASBIE components (associations to ABIEs found in a common library) and BBIE components (basic indivisible portions of content that may have metadata). The Document ABIE is simply an ABIE that is not defined in the common library. A Document ABIE specifies the components at the apex of a business document. A Library ABIE specifies the components comprising an ASBIE at the branches of the business document tree form. BBIE components are the leaves of the business document where content is found in the context of the branches.

From an abstract information bundle perspective, a given business document is comprised of its Document ABIE with its indivisible BBIEs and with its (divisible) ASBIEs that each have their own BBIEs and ASBIEs.

Each BBIE is defined by its data type. The available Core Component Types are summarized in Chapter 8 of the CCTS 2.01 specification **[CCTS]**. Each type has a mandatory content component and a number of optional metadata supplementary components. The interpretation of the content is informed by the values of its given supplementary components. The data type content and supplementary component values are, ostensibly, either numbers, strings or Boolean values. Interpreting the strings to be abstract concepts such as date and time is up to the program.

2.2 Marshalling CCTS to JSON

2.2.1 Business Information Entity (BIE) names

There are four groups of names in these document models: the names of the Document ABIEs, the names of BBIEs, the names of both Library ABIEs and the ASBIEs that point to them, and the names of extension scaffolding and metadata items. In fact there are many groups of Document ABIEs as each Document ABIE is in a group of one of only itself.

Names in XML documents are qualified with namespace URI strings. These UBL JSON schemas provide for preserving the four uses of XML namespaces in a source XML document that might have been transliterated to JSON syntax. This allows the JSON document to be “round-tripped” back to XML syntax should it be necessary. Accommodating extension content is up to extension designers.

2.2.2 Document ABIE expression

Each JSON serialization begins as an apex object containing up to five properties in any order.

The following four properties are for documentary or transliteration purposes, and for completeness should be present:

- name “`_D`” with the string value of the Document ABIE namespace URI;
- name “`_A`” with the string value of the ASBIE namespace URI (that is, the Library ABIE namespace URI);
- name “`_B`” with the string value of the BBIE namespace URI; and
- name “`_E`” with the string value of the extension namespace URI (when extensions are being used).

That apex object always contains a property with the name of the Document ABIE. In legacy-based schemas the value is an array containing a single Document ABIE object. In model-based schemas the value is the Document ABIE object, not in an array.

The Document ABIE object contains properties, in any order, of all of the ASBIE and BBIE members in actual use of those available as specified by the Document ABIE. In legacy-based schemas, each property is defined as an array. In model-based schemas, only those properties associated with the model’s definition of the maximum cardinality as unbounded are defined as an array. Properties with bounded maximum cardinality are defined as an object.

The Document ABIE object may contain a single property representing the apex of the scaffolding of foreign extension content. This name is defined in UBL to be “`UBLExtensions`”.

A UBL example of a serialized legacy-based Document ABIE with some content elided by the ellipsis for brevity, leaving three BBIEs followed by an ASBIE that itself has two BBIEs is as follows:

```
{"_D":"urn:oasis:names:specification:ubl:schema:xsd:Invoice-2",
 "_A":"urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2",
 "_B":"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2",
 "Invoice":[{
 "ID":[{"_":"123"}],
 "IssueDate":[{"_":"2011-09-22"}],
 "Note":[{"_":"Information text for the invoice"}],
 "InvoicePeriod":[{
  "StartDate":[{"_":"2011-08-01"}],
  "EndDate":[{"_":"2011-08-31"}]
 }],
...
}]}
```

A UBL example of a serialized model-based Document ABIE with the same content elided by the ellipsis for brevity, leaving three BBIEs followed by an ASBIE that itself has two BBIEs is as follows, given that the identifier and issue date both have a bounded maximum cardinality of “1” and the note and invoice period structures each have an unbounded maximum cardinality:

```
{"_D":"urn:oasis:names:specification:ubl:schema:xsd:Invoice-2",
 "_A":"urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2",
 "_B":"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2",
 "Invoice":{
 "ID":{"_":"123"},
 "IssueDate":{"_":"2011-09-22"},
 "Note":[{"_":"Information text for the invoice"}],
 "InvoicePeriod":[{
  "StartDate":{"_":"2011-08-01"},
  "EndDate":{"_":"2011-08-31"}
 }],
...
}}
```

2.2.3 ASBIE expression

Each sequence of standalone or consecutive ASBIE components of the same name in an information bundle is serialized in user data based on the type of serialization. In legacy-based serialization, the item or items are expressed in a single array property of its parent ASBIE or Document ABIE object. In model-based serialization, only if the model declares an unbounded maximum cardinality for the item will an array be used, otherwise an array is not used.

The one name for that property is the name of the ASBIE. In legacy-based serialization the one value for that property is an array of one or more objects, each object representing, in array order, the sequence order of the consecutive ASBIEs. In model-based serialization, the property is simply the object itself if the maximum cardinality is bounded to “1”, otherwise an array is used as in legacy-based serialization regardless of he number of ASBIEs. If there are no ASBIEs of a given name, the property is absent.

Each object in the ASBIE array contains as its properties all of the ASBIE and BBIE members in actual use of those available as specified by the Library ABIE that defines the ASBIE.

This legacy-based serialization in the context of “`AccountingSupplierParty`” for one “`Party`” ASBIE has one ASBIE child named “`PartyName`”:

```
"Party":[{
  "PartyName":[{
   "Name":[{"_":"Custom Cotter Pins"}]
  }]
 }]
```

This model-based serialization in the context of “`AccountingSupplierParty`” for the same “`Party`” ASBIE has one ASBIE child named “`PartyName`”, but the maximum cardinality of the Party in this context is bounded to be “1”:

```
"Party":{
  "PartyName":[{
   "Name":[{"_":"Custom Cotter Pins"}]
  }]
 }
```

This legacy-based serialization for two “`InvoiceLine`” ASBIEs has as children two BBIEs and an ASBIE, where that ASBIE has on BBIE child:

```
"InvoiceLine":[{
 "ID":[{"_":"1"}],
 "LineExtensionAmount":[{"_":100.00,
  "currencyID":"CAD"}],
 "Item":[{
  "Description":[{"_":"Cotter pin, MIL-SPEC"}]
 }]
},{
 "ID":[{"_":"2"}],
 "LineExtensionAmount":[{"_":200.00,
  "currencyID":"CAD"}],
 "Item":[{
  "Description":[{"_":"Axel"}]
 }]
}]
```

This model-based serialization for the same two “`InvoiceLine`” ASBIEs has as children two BBIEs and an ASBIE, where that ASBIE has on BBIE child:

```
"InvoiceLine":[{
 "ID":{"_":"1"},
 "LineExtensionAmount":{"_":100.00,
  "currencyID":"CAD"},
 "Item":{
  "Description":[{"_":"Cotter pin, MIL-SPEC"}]
 }
},{
 "ID":{"_":"2"},
 "LineExtensionAmount":{"_":200.00,
  "currencyID":"CAD"},
 "Item":{
  "Description":[{"_":"Axel"}]
 }
}]
```

2.2.4 BBIE expression

Each sequence of one or more consecutive BBIE components of the same name in an information bundle are serialized in user data as a single property of its parent ASBIE or Document ABIE object.

The one name for that property is the name of the BBIE. In legacy-based serialization the one value for that property is an array of one or more objects, each object representing, in array order, the sequence order of the consecutive BBIEs. In model-based serialization, the property is simply the object itself if the maximum cardinality is bounded to “1”, otherwise an array is used as in legacy-based serialization regardless of he number of BBIEs. If there are no BIEs of a given name, the property is absent.

Each BBIE has a required property whose name is the underscore symbol and whose value is the value of the business object in the information bundle.

Each BBIE may have as many additional properties with the names and values of optional or required metadata supplementary components of the given instance of the Core Component Type. Such is defined by the BDNDR unqualified data types. Provision is made in the model for qualifying the data types, and such are enumerated in UBL’s qualified data types. UBL provides no qualifications in the JSON model and so it is up to the application to impose qualifications as needed.

The property names are derived from the XML Schema attributes for the approved CCTS 2.01 core component type content and supplementary component names. The list of available property attribute names is summarized as follows, with required properties indicated:

Table 1. CCTS 2.01 approved core component type content and supplementary components for permissible representation terms

Permissible primary and secondary representation terms	Approved content and supplementary components
`Amount`	_ (required content property) `currencyID` (required component property) `currencyCodeListVersionID`

`BinaryObject`
`Graphic`
`Picture`
`Sound`
`Video`
	_ (required content property)
	`mimeCode` (required component property)
	`format`
	`encodingCode`
	`characterSetCode`
	`uri`
	`filename`

`Code`	_ (required content property)
	`listID`
	`listAgencyID`
	`listAgencyName`
	`listName`
	`listVersionID`
	`name`
	`languageID`
	`listURI`
	`listSchemeURI`

`DateTime`	_ (required content property)
`Date`	_ (required content property)
`Time`	_ (required content property)

`Identifier`	_ (required content property)
	`schemeID`
	`schemeName`
	`schemeAgencyID`
	`schemeAgencyName`
	`schemeVersionID`

Permissible primary and secondary representation terms	**Approved content and supplementary components**
	schemeDataURI schemeURI
Indicator	_ (required content property)
Measure	_ (required content property) unitCode (required component property) unitCodeListVersionID
Numeric Value Percent Rate	_ (required content property) format
Quantity	_ (required content property) unitCode unitCodeListID unitCodeListAgencyID unitCodeListAgencyName
Text Name	_ (required content property) languageID languageLocaleID

Some legacy-based serialization examples of BBIEs: an identifier, a date and two notes:

-
  ```
  "ID":[{"_":"123"}]
  ```

-
  ```
  "IssueDate":[{"_":"2011-09-22"}]
  ```

-
  ```
  "Note":[{"_":"Test",
   "LanguageIdentifier":"en"},
  {"_":"Tester",
   "LanguageIdentifier":"fr"}]
  ```

2.2.5 Extension expression

When a business document contains foreign content not defined by the business vocabulary, this content is placed under a single apex containing all of the scaffolding of all foreign extension content. The name of this apex is defined in UBL to be "`UBLExtensions`", declared as an array of one member. The apex name is used as the name of the property of the Document ABIE object.

That one member of the array is an object named "`UBLExtension`", declared to be an array of one or more members, each member being a single extension (of which there may be many). Each member is an object that may have any number of properties of extension metadata, whose names are defined by the vocabulary. Each object has a property with the name "`ExtensionContent`") as an array of one object. That one object defines the foreign extension content.

The definition of foreign content is out of scope of this document. Trading partners must agree on the significance or insignificance of any foreign content.

The example `UBL-Invoice-2.0-Enveloped.json` has three foreign extensions, one of which is a transliteration of the UBL extension for digital signatures. As a transliteration, it no longer functions as a correctly-formed digital signature. At this time there is no specification for the use of digital signatures in JSON documents for UBL.

2.3 Unmarshalling JSON to CCTS

2.3.1 Overview of unmarshalling

A JSON stream loaded into an application's variable will create in that variable the objects and arrays representing the CCTS constructs of UBL.

The examples in this section are copied from the interactive interfaces of the Node.js interpreter and the Python interpreter.

2.3.2 JavaScript syntax for legacy-based serialization

This JSON representation support for CCTS objects fully supports the dot notation defined in JavaScript.

The JSON stream creates an object with the optional properties of the namespaces and the mandatory property of Document ABIE. The name of that property is the name of the Document ABIE. The value of that property is the array of one object defining the contents of the Document ABIE comprised of BBIEs and ASBIEs.

The Document ABIE and each BBIE and ASBIE must be addressed as a list that is guaranteed to have at the least a single member addressed with "`[0]`".

Note

Future versions of UBL may raise the cardinality of any given object and so this approach is future-proof to revisions. This approach is also consistent and is used without thought of cardinality and so should be easier for the programmer (if a little verbose). Although a change in cardinality does not apply to the Document ABIE, the array notation is used for consistency with ASBIE expressions using arrays.

The content of a BBIE is addressed as the object property with the name "_". Supplementary components of a BBIE are addressed by their equivalent XML Schema attribute name. For a complete list, see Table 1, "CCTS 2.01 approved core component type content and supplementary components for permissible representation terms".

Some examples of dot notation syntax for a legacy-based instance:

```
> obj.Invoice[0].Note[0]
{ _: 'Test', languageID: 'en' }
> obj.Invoice[0].Note[1]
{ _: 'Tester', languageID: 'fr' }
> obj.Invoice[0].Note[1]._
'Tester'
> obj.Invoice[0].InvoiceLine[1].LineExtensionAmount[0]._
200
> obj.Invoice[0].InvoiceLine[1].LineExtensionAmount[0].currencyID
'CAD'
>
```

This approach also supports the object notation defined in JavaScript. The same examples using list and object notation syntax for a legacy-based instance:

```
> obj["Invoice"][0]["Note"][0]
{ _: 'Test', languageID: 'en' }
> obj["Invoice"][0]["Note"][1]
{ _: 'Tester', languageID: 'fr' }
> obj["Invoice"][0]["Note"][1]["_"]
```

```
'Tester'
> obj["Invoice"][0]["InvoiceLine"][1]["LineExtensionAmount"][0]["_"]
200
> obj["Invoice"][0]["InvoiceLine"][1]["LineExtensionAmount"][0]["currencyID"]
'CAD'
>
```

2.3.3 Python syntax for legacy-based serialization

This approach fully supports the list and dictionary notation defined in Python.

The JSON stream creates a dictionary with a single array of one property. The name of that property is the name of the Document ABIE. The value of that property is the object defining the contents of the Document ABIE comprised of BBIEs and ASBIEs.

Each BBIE and ASBIE must be addressed as a list that is guaranteed to have at the least a single member addressed with "`[0]`".

Note

Future versions of UBL may raise the cardinality of any given object and so this approach is future-proof to revisions. This approach is also consistent and is used without thought of cardinality and so should be easier for the programmer (if a little verbose). Although a change in cardinality does not apply to the Document ABIE, the array notation is used for consistency with ASBIE expressions using arrays.

The content of a BBIE is addressed as the dictionary property with the name ending in "`Content`". Supplementary components of a BBIE are addressed by their name (compressed removing periods and spaces). For a complete list, see Table 1, "CCTS 2.01 approved core component type content and supplementary components for permissible representation terms".

Some examples using list and dictionary syntax for a legacy-based instance:

```
>>> obj["Invoice"][0]["Note"][0]
{'_': 'Test', 'languageID': 'en'}
>>> obj["Invoice"][0]["Note"][1]
{'_': 'Tester', 'languageID': 'fr'}
>>> obj["Invoice"][0]["Note"][1]["_"]
'Tester'
>>> obj["Invoice"][0]["InvoiceLine"][1]["LineExtensionAmount"][0]["_"]
200.0
>>> obj["Invoice"][0]["InvoiceLine"][1]["LineExtensionAmount"][0]["currencyID"]
'CAD'
>>>
```

Note

Observe how the Python list and dictionary syntax is identical to the JavaScript list and object syntax.

3 Example UBL JSON documents

3.1 Examples overview

The examples in the `json-legacy/` and `json-model/` subdirectories are transliterated from the sample XML instances found in the UBL 2.3 distribution at https://docs.oasis-open.org/ubl/os-UBL-2.3/xml/.

For legibility the examples are indented. Indentation is insignificant white space and optionally can be eliminated from the file.

The remaining sections are examples of legacy-mode JSON instances. See the model-mode directory for examples of the use of model-mode serialization.

3.2 `UBL-Invoice-2.1-Example-Trivial.json`

```
{"_D":"urn:oasis:names:specification:ubl:schema:xsd:Invoice-2",
 "_A":"urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2",
 "_B":"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2",
 "Invoice":[{
 "ID":[{"_":"123"}],
 "IssueDate":[{"_":"2011-09-22"}],
 "InvoicePeriod":[{
  "StartDate":[{"_":"2011-08-01"}],
  "EndDate":[{"_":"2011-08-31"}]
 }],
 "AccountingSupplierParty":[{
  "Party":[{
   "PartyName":[{
    "Name":[{"_":"Custom Cotter Pins"}]
   }]
  }]
 }],
 "AccountingCustomerParty":[{
  "Party":[{
   "PartyName":[{
    "Name":[{"_":"North American Veeblefetzer"}]
   }]
  }]
 }],
 "LegalMonetaryTotal":[{
  "PayableAmount":[{"_":100.00,
   "currencyID":"CAD"}]
 }],
 "InvoiceLine":[{
  "ID":[{"_":"1"}],
  "LineExtensionAmount":[{"_":100.00,
   "currencyID":"CAD"}],
  "Item":[{
   "Description":[{"_":"Cotter pin, MIL-SPEC"}]
  }]
 }]
}]}
```

3.3 Modified invoice example

This example has multiple BBIEs for “`Note`” and multiple ABIEs for “`InvoiceLine`”:

```
{"_D":"urn:oasis:names:specification:ubl:schema:xsd:Invoice-2",
 "_A":"urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2",
 "_B":"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2",
 "Invoice":[{
 "ID":[{"_":"123"}],
 "IssueDate":[{"_":"2011-09-22"}],
 "Note":[{"_":"Test",
  "languageID":"en"},
 {"_":"Tester",
  "languageID":"fr"}],
 "InvoicePeriod":[{
  "StartDate":[{"_":"2011-08-01"}],
  "EndDate":[{"_":"2011-08-31"}]
 }],
 "AccountingSupplierParty":[{
  "Party":[{
   "PartyName":[{
    "Name":[{"_":"Custom Cotter Pins"}]
   }]
  }]
 }],
 "AccountingCustomerParty":[{
  "Party":[{
   "PartyName":[{
    "Name":[{"_":"North American Veeblefetzer"}]
   }]
  }]
 }],
 "LegalMonetaryTotal":[{
  "PayableAmount":[{"_":100.00,
   "currencyID":"CAD"}]
 }],
 "InvoiceLine":[{
  "ID":[{"_":"1"}],
  "LineExtensionAmount":[{"_":100.00,
   "currencyID":"CAD"}],
  "Item":[{
   "Description":[{"_":"Cotter pin, MIL-SPEC"}]
  }]
 },{
  "ID":[{"_":"2"}],
  "LineExtensionAmount":[{"_":200.00,
   "currencyID":"CAD"}],
  "Item":[{
   "Description":[{"_":"Axel"}]
  }]
 }]
}]}
```

3.4 `UBL-Order-2.1-Example.json`

```
{"_D":"urn:oasis:names:specification:ubl:schema:xsd:Order-2",
 "_A":"urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2",
 "_B":"urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2",
 "Order":[{
 "UBLVersionID":[{"_":"2.1"}],
 "CustomizationID":[{"_":"urn:www.cenbii.eu:transaction:biicoretrdm001:ver1.0"}],
 "ProfileID":[{"_":"urn:www.cenbii.eu:profile:BII01:ver1.0",
  "schemeAgencyID":"BII",
  "schemeID":"Profile"}],
 "ID":[{"_":"34"}],
 "IssueDate":[{"_":"2010-01-20"}],
 "IssueTime":[{"_":"12:30:00"}],
 "Note":[{"_":"Information text for the whole order"}],
 "DocumentCurrencyCode":[{"_":"SEK"}],
 "AccountingCostCode":[{"_":"Project123"}],
 "ValidityPeriod":[{
  "EndDate":[{"_":"2010-01-31"}]
 }],
 "QuotationDocumentReference":[{
  "ID":[{"_":"QuoteID123"}]
 }],
 "OrderDocumentReference":[{
  "ID":[{"_":"RjectedOrderID123"}]
 }],
 "OriginatorDocumentReference":[{
  "ID":[{"_":"MAFO"}]
 }],
 "AdditionalDocumentReference":[{
  "ID":[{"_":"Doc1"}],
  "DocumentType":[{"_":"Timesheet"}],
  "Attachment":[{
   "ExternalReference":[{
    "URI":[{"_":"http://www.suppliersite.eu/sheet001.html"}]
   }]
  }]
 },{
  "ID":[{"_":"Doc2"}],
  "DocumentType":[{"_":"Drawing"}],
  "Attachment":[{
   "EmbeddedDocumentBinaryObject":[{"_":"UjBsR09EbGhjZ0dTQUxNQUFBUUNBRU1tQ1p0dU1GUXhEUz
    "mimeCode":"application/pdf"}]
  }]
 }],
 "Contract":[{
  "ID":[{"_":"34322"}],
  "ContractType":[{"_":"FrameworkAgreementID123"}]
 }],
 "BuyerCustomerParty":[{
  "Party":[{
   "EndpointID":[{"_":"7300072311115",
    "schemeAgencyID":"9",
    "schemeID":"GLN"}],
   "PartyIdentification":[{
    "ID":[{"_":"7300070011115",
     "schemeAgencyID":"9",
     "schemeID":"GLN"}]
   },{
    "ID":[{"_":"PartyID123"}]
   }],
   "PartyName":[{
    "Name":[{"_":"Johnssons byggvaror"}]
   }],
   "PostalAddress":[{
    "ID":[{"_":"1234567890123",
     "schemeAgencyID":"9",
     "schemeID":"GLN"}],
    "Postbox":[{"_":"PoBox123"}],
    "StreetName":[{"_":"Rådhusgatan"}],
    "AdditionalStreetName":[{"_":"2nd floor"}],
    "BuildingNumber":[{"_":"5"}],
    "Department":[{"_":"Purchasing department"}],
    "CityName":[{"_":"Stockholm"}],
    "PostalZone":[{"_":"11000"}],
    "CountrySubentity":[{"_":"RegionX"}],
    "Country":[{
     "IdentificationCode":[{"_":"SE"}]
    }]
   }],
   "PartyTaxScheme":[{
    "RegistrationName":[{"_":"Herra Johnssons byggvaror AS"}],
    "CompanyID":[{"_":"SE1234567801"}],
    "RegistrationAddress":[{
     "CityName":[{"_":"Stockholm"}],
     "Country":[{
      "IdentificationCode":[{"_":"SE"}]
     }]
    }],
    "TaxScheme":[{
     "ID":[{"_":"VAT",
      "schemeID":"UN/ECE 5153",
      "schemeAgencyID":"6"}]
    }]
   }],
   "PartyLegalEntity":[{
    "RegistrationName":[{"_":"Johnssons Byggvaror AB"}],
    "CompanyID":[{"_":"5532331183",
     "schemeID":"SE:ORGNR"}],
    "RegistrationAddress":[{
     "CityName":[{"_":"Stockholm"}],
     "CountrySubentity":[{"_":"RegionX"}],
     "Country":[{
      "IdentificationCode":[{"_":"SE"}]
     }]
    }]
   }],
   "Contact":[{
    "Telephone":[{"_":"123456"}],
    "Telefax":[{"_":"123456"}],
    "ElectronicMail":[{"_":"pelle@johnsson.se"}]
   }],
   "Person":[{
    "FirstName":[{"_":"Pelle"}],
    "FamilyName":[{"_":"Svensson"}],
    "MiddleName":[{"_":"X"}],
    "JobTitle":[{"_":"Boss"}]
   }]
  }],
  "DeliveryContact":[{
   "Name":[{"_":"Eva Johnsson"}],
   "Telephone":[{"_":"1234356"}],
   "Telefax":[{"_":"123455"}],
   "ElectronicMail":[{"_":"eva@johnsson.se"}]
  }]
 }],
 "SellerSupplierParty":[{
  "Party":[{
   "EndpointID":[{"_":"7302347231111",
    "schemeAgencyID":"9",
    "schemeID":"GLN"}],
   "PartyIdentification":[{
    "ID":[{"_":"SellerPartyID123"}]
   }],
   "PartyName":[{
    "Name":[{"_":"Moderna Produkter AB"}]
   }],
   "PostalAddress":[{
    "ID":[{"_":"0987654321123",
     "schemeAgencyID":"9",
     "schemeID":"GLN"}],
    "Postbox":[{"_":"321"}],
    "StreetName":[{"_":"Kungsgatan"}],
    "AdditionalStreetName":[{"_":"suite12"}],
    "BuildingNumber":[{"_":"22"}],
    "Department":[{"_":"Sales department"}],
    "CityName":[{"_":"Stockholm"}],
    "PostalZone":[{"_":"11000"}],
    "CountrySubentity":[{"_":"RegionX"}],
    "Country":[{
     "IdentificationCode":[{"_":"SE"}]
    }]
   }],
   "PartyLegalEntity":[{
    "RegistrationName":[{"_":"Moderna Produkter AB"}],
    "CompanyID":[{"_":"5532332283",
     "schemeID":"SE:ORGNR"}],
    "RegistrationAddress":[{
     "CityName":[{"_":"Stockholm"}],
     "CountrySubentity":[{"_":"RegionX"}],
     "Country":[{
      "IdentificationCode":[{"_":"SE"}]
     }]
    }]
   }],
   "Contact":[{
    "Telephone":[{"_":"34557"}],
    "Telefax":[{"_":"3456767"}],
    "ElectronicMail":[{"_":"lars@moderna.se"}]
   }],
   "Person":[{
    "FirstName":[{"_":"Lars"}],
    "FamilyName":[{"_":"Petersen"}],
    "MiddleName":[{"_":"M"}],
    "JobTitle":[{"_":"Sales manager"}]
   }]
  }]
 }],
 "OriginatorCustomerParty":[{
  "Party":[{
   "PartyIdentification":[{
    "ID":[{"_":"0987678321123",
     "schemeAgencyID":"9",
     "schemeID":"GLN"}]
   }],
   "PartyName":[{
    "Name":[{"_":"Moderna Produkter AB"}]
   }],
   "Contact":[{
    "Telephone":[{"_":"346788"}],
    "Telefax":[{"_":"8567443"}],
    "ElectronicMail":[{"_":"sven@moderna.se"}]
   }],
   "Person":[{
    "FirstName":[{"_":"Sven"}],
    "FamilyName":[{"_":"Pereson"}],
    "MiddleName":[{"_":"N"}],
    "JobTitle":[{"_":"Stuffuser"}]
   }]
  }]
 }],
 "Delivery":[{
  "DeliveryLocation":[{
   "Address":[{
    "ID":[{"_":"1234567890123",
     "schemeAgencyID":"9",
     "schemeID":"GLN"}],
    "Postbox":[{"_":"123"}],
    "StreetName":[{"_":"Rådhusgatan"}],
    "AdditionalStreetName":[{"_":"2nd floor"}],
    "BuildingNumber":[{"_":"5"}],
    "Department":[{"_":"Purchasing department"}],
    "CityName":[{"_":"Stockholm"}],
    "PostalZone":[{"_":"11000"}],
    "CountrySubentity":[{"_":"RegionX"}],
    "Country":[{
     "IdentificationCode":[{"_":"SE"}]
    }]
   }]
  }],
  "RequestedDeliveryPeriod":[{
   "StartDate":[{"_":"2010-02-10"}],
   "EndDate":[{"_":"2010-02-25"}]
  }],
  "DeliveryParty":[{
   "PartyIdentification":[{
    "ID":[{"_":"67654328394567",
     "schemeAgencyID":"9",
     "schemeID":"GLN"}]
   }],
   "PartyName":[{
    "Name":[{"_":"Swedish trucking"}]
   }],
   "Contact":[{
    "Name":[{"_":"Per"}],
    "Telephone":[{"_":"987098709"}],
    "Telefax":[{"_":"34673435"}],
    "ElectronicMail":[{"_":"bill@svetruck.se"}]
   }]
  }]
 }],
 "DeliveryTerms":[{
  "ID":[{"_":"FOT",
   "schemeAgencyID":"6",
   "schemeID":"IMCOTERM"}],
  "SpecialTerms":[{"_":"CAD"}],
  "DeliveryLocation":[{
   "ID":[{"_":"STO"}]
  }]
 }],
 "AllowanceCharge":[{
  "ChargeIndicator":[{"_":true}],
  "AllowanceChargeReason":[{"_":"Transport documents"}],
  "Amount":[{"_":100,
   "currencyID":"SEK"}]
 },{
  "ChargeIndicator":[{"_":false}],
  "AllowanceChargeReason":[{"_":"Total order value discount"}],
  "Amount":[{"_":100,
   "currencyID":"SEK"}]
 }],
 "TaxTotal":[{
  "TaxAmount":[{"_":100,
   "currencyID":"SEK"}]
 }],
 "AnticipatedMonetaryTotal":[{
  "LineExtensionAmount":[{"_":6225,
   "currencyID":"SEK"}],
  "AllowanceTotalAmount":[{"_":100,
   "currencyID":"SEK"}],
  "ChargeTotalAmount":[{"_":100,
   "currencyID":"SEK"}],
  "PayableAmount":[{"_":6225,
   "currencyID":"SEK"}]
 }],
 "OrderLine":[{
  "Note":[{"_":"Freetext note on line 1"}],
  "LineItem":[{
   "ID":[{"_":"1"}],
   "Quantity":[{"_":120,
    "unitCode":"LTR"}],
   "LineExtensionAmount":[{"_":6000,
    "currencyID":"SEK"}],
   "TotalTaxAmount":[{"_":10,
    "currencyID":"SEK"}],
   "PartialDeliveryIndicator":[{"_":false}],
   "AccountingCostCode":[{"_":"ProjectID123"}],
   "Delivery":[{
    "RequestedDeliveryPeriod":[{
     "StartDate":[{"_":"2010-02-10"}],
     "EndDate":[{"_":"2010-02-25"}]
    }]
   }],
   "OriginatorParty":[{
    "PartyIdentification":[{
     "ID":[{"_":"EmployeeXXX",
      "schemeAgencyID":"ZZZ",
      "schemeID":"ZZZ"}]
    }],
    "PartyName":[{
     "Name":[{"_":"Josef K."}]
    }]
   }],
   "Price":[{
    "PriceAmount":[{"_":50,
     "currencyID":"SEK"}],
    "BaseQuantity":[{"_":1,
     "unitCode":"LTR"}]
   }],
   "Item":[{
    "Description":[{"_":"Red paint"}],
    "Name":[{"_":"Falu Rödfärg"}],
    "SellersItemIdentification":[{
     "ID":[{"_":"SItemNo001"}]
    }],
    "StandardItemIdentification":[{
     "ID":[{"_":"1234567890123",
      "schemeAgencyID":"6",
      "schemeID":"GTIN"}]
    }],
    "AdditionalItemProperty":[{
     "Name":[{"_":"Paint type"}],
     "Value":[{"_":"Acrylic"}]
    },{
     "Name":[{"_":"Solvant"}],
     "Value":[{"_":"Water"}]
    }]
   }]
  }]
 },{
  "Note":[{"_":"Freetext note on line 2"}],
  "LineItem":[{
   "ID":[{"_":"2"}],
   "Quantity":[{"_":15,
    "unitCode":"C62"}],
   "LineExtensionAmount":[{"_":225,
    "currencyID":"SEK"}],
   "TotalTaxAmount":[{"_":10,
    "currencyID":"SEK"}],
   "PartialDeliveryIndicator":[{"_":false}],
   "AccountingCostCode":[{"_":"ProjectID123"}],
   "Delivery":[{
    "RequestedDeliveryPeriod":[{
     "StartDate":[{"_":"2010-02-10"}],
     "EndDate":[{"_":"2010-02-25"}]
    }]
   }],
   "OriginatorParty":[{
    "PartyIdentification":[{
     "ID":[{"_":"EmployeeXXX",
      "schemeAgencyID":"ZZZ",
      "schemeID":"ZZZ"}]
    }],
    "PartyName":[{
     "Name":[{"_":"Josef K."}]
    }]
   }],
   "Price":[{
    "PriceAmount":[{"_":15,
     "currencyID":"SEK"}],
    "BaseQuantity":[{"_":1,
     "unitCode":"C62"}]
   }],
   "Item":[{
    "Description":[{"_":"Very good pencils for red paint."}],
    "Name":[{"_":"Pensel 20 mm"}],
    "SellersItemIdentification":[{
     "ID":[{"_":"SItemNo011"}]
    }],
    "StandardItemIdentification":[{
     "ID":[{"_":"123452340123",
      "schemeAgencyID":"6",
      "schemeID":"GTIN"}]
    }],
    "AdditionalItemProperty":[{
     "Name":[{"_":"Hair color"}],
     "Value":[{"_":"Black"}]
    },{
     "Name":[{"_":"Width"}],
     "Value":[{"_":"20mm"}]
    }]
   }]
  }]
 }]
}]}
```

4 JSON schema and document demonstration environment

In the `val/` subdirectory of the ZIP file are the following files:

`jsonvalidate.py`

A Python application to invoke the `jsonschema` library documented at https://python-jsonschema.readthedocs.io/en/latest/.

Note

At the time of writing, it is observed that the `jsonschema` library is not functioning correctly under Windows. The Windows invocations are included in this Committee Note distribution in anticipation of a future release of the library working successfully.

`testjson.bat` and `testjson.sh`

Invoking the validate scripts once for each of the four sample files in the `val/` subdirectory: `order-test-bad1.json` (bad syntax), `order-test-bad2.json` (bad property name), `order-test-bad3.json` (missing required property), `order-test-bad4.json` (invalid date value), and `order-test-good.json` (no errors).

Note

At the time of writing, it is observed that the `jsonschema` library is not correctly detecting an invalid date value in the `order-test-bad4.json` test file. This is due to the specification making support of such optional for implementations to offer.

`testjsonsamples.bat` and `testjsonsamples.sh`

Invoking the validate scripts once for each of the sample files in the `json-legacy/` subdirectory.

`validatejson.bat` and `validatejson.sh`

Invoking the Python application once with the arguments of a JSON schema and a JSON document.

To demonstrate the successful validation of all of the sample JSON documents with the corresponding JSON schema, one would invoke the tests in a Unix-based environment as follows:

```
~/t $ cd mytest
~/t/mytest $ ls
json        json-schema    val
~/t/mytest $ cd val
~/t/mytest/val $ ls
jsonvalidate.py        order-test-good.json    testjsonsamples.sh
order-test-bad1.json    output.txt        validatejson.bat
order-test-bad2.json    testjson.bat        validatejson.sh
order-test-bad3.json    testjson.sh
order-test-bad4.json    testjsonsamples.bat
~/t/mytest/val $ sh testjson.sh

############################################################
Validating order-test-bad1.json with ../json-schema-legacy/maindoc/UBL-Order-2.3.json
############################################################
```

```
Unexpected error injesting instance: Expecting , delimiter: line 19 column 30 (char 774

############################################################
Validating order-test-bad2.json with ../json-schema-legacy/maindoc/UBL-Order-2.3.json
############################################################
Validation error: Additional properties are not allowed (u'OrderDocumentReferenceXXXXX'
validator: additionalProperties
path: deque([u'Order', 0])
cause: None
context: []
validator_value: False
schema_path: deque([u'properties', u'Order', u'items', u'additionalProperties'])
parent: None

############################################################
Validating order-test-bad3.json with ../json-schema-legacy/maindoc/UBL-Order-2.3.json
############################################################
Validation error: u'ID' is a required property
validator: required
path: deque([u'Order', 0])
cause: None
context: []
validator_value: [u'ID', u'IssueDate', u'BuyerCustomerParty', u'SellerSupplierParty', u
schema_path: deque([u'properties', u'Order', u'items', u'required'])
parent: None

############################################################
Validating order-test-bad4.json with ../json-schema-legacy/maindoc/UBL-Order-2.3.json
############################################################
No schema validation errors.

############################################################
Validating order-test-good.json with ../json-schema-legacy/maindoc/UBL-Order-2.3.json
############################################################
No schema validation errors.
~/t/mytest/val $
```

Appendix A Release Notes

A.1 Availability

Online and downloadable versions of this release are available from the locations specified at the top of this document.

A.2 Status of this release

Release of this package initiates the process to develop Committee Note 01 of version 1.0.

A.3 Package Structure

This OASIS Committee Note is published as a zip archive in the https://docs.oasis-open.org/ubl/UBL-2.3-JSON/v1.0/cn01/ directory. Unzipping this archive creates a directory tree containing a master DocBook XML file (UBL-2.3-JSON-v1.0-cn01.xml), a generated hypertext version of this file (UBL-2.3-JSON-v1.0-cn01.html), a generated PDF version of this file (UBL-2.3-JSON-v1.0-cn01.pdf), and a number of subdirectories. The files in these subdirectories contain the various components of this release. A description of each subdirectory is given below. Note that while the UBL-2.3-JSON-v1.0-cn01.xml file is the “original” of this specification, it may not be viewable in all currently available web browsers.

`art`

Diagrams used in this note

`db`

DocBook documentation support files

`json-legacy`

JSON UBL 2.3 example documents in legacy-based serialization

`json-model`

JSON UBL 2.3 example documents in model-based serialization

`json-schema-legacy`

JSON UBL 2.3 schemas supporting legacy-based serialization

`json-schema-model`

JSON UBL 2.3 schemas supporting legacy-based serialization

`val`

Validation demonstration environment in Python

A.4 Support

UBL is a volunteer project of the international business community. Inquiries regarding UBL may be posted to the public ubl-dev list, archives for which are located at

https://lists.oasis-open.org/archives/ubl-dev/

Subscriptions to ubl-dev can be made through the OASIS list manager at https://www.oasis-open.org/mlmanage/index.php

OASIS provides an official community gathering place and information resource for UBL at

http://ubl.xml.org/

General information regarding UBL can be found at the Wikipedia article:

http://www.wikipedia.org/wiki/Universal_Business_Language

Appendix B Revision History

All artefacts are created using the latest Business Document Naming and Design Rules2020April-changedwhich

Appendix C (informative) Acknowledgements

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Todd Albers, Federal Reserve Bank of Minneapolis
Rui Barros, Individual
Oriol Bausa Peris, Individual
Kenneth Bengtsson, Individual
Erlend Klakegg Bergheim, Norwegian Digitalisation Agency
Peter Borresen, ClearView Trade
Andrea Caccia, Individual
Ger Clancy, IBM
Kees Duvekot, RFS Holland Holding B.V.
Martin Forsberg, Swedish Association of Local Authorities & Regions
Cecile Guasch, Individual
Philip Helger, Individual
Ken Holman, Crane Softwrights Ltd.
Yves Jordan, Publications Office of the European Union
Kari Korpela, Individual
Ole Madsen, Danish Business Authority
Natalie Muric, Publications Office of the European Union
Levine Naidoo, IBM
Enric Torregrosa, everis, S.L.U.
Matt Vickers, Xero