

Versions

The UBL versions this package models, from 2.1 to 2.5, how to choose one and what changes between them

Anonymous · 13/10/2026

Versions

Every UBL version from 2.1 to 2.5 has its own model, generated from the official XSD of that version.

Version	Documents	Official JSON schemas
2.1	65	yes
2.2	81	yes
2.3	91	yes
2.4	93	no
2.5	101	no

The model of a version lives in its own namespace, `Derafu\Ubl\V21\Oasis` for 2.1, `Derafu\Ubl\V22\Oasis` for 2.2 and so on, with the same layout in each one: `Doc` (the documents), `Cac` (the aggregate components), `Cbc` (the basic components), `Ext`, `Sac` and `Sig` (extensions and signatures), `Enum` and `Xml`. The services (`Serializer`, `Deserializer`, the validators...) are the same for all of them.

Choosing a Version

The XML of a UBL document does not say which version it is: every 2.x version has the same XML namespaces (`urn:oasis:names:specification:ubl:schema:xsd:Invoice-2`), and `UBLVersionID` is optional. So the version is chosen, never detected.

The default is **2.1**, the version of Peppol BIS Billing 3.0. A call can ask for another one with the `version` option:

```
use Derafu\Ubl\Service\Deserializer;  
  
$invoice = (new Deserializer())->deserialize($xml, 'xml', ['version' => '2.3']);
```

`Serializer` does not need it: a document knows its version by its class (`Derafu\Ubl\V23\Oasis\Doc\Invoice`).

An unknown version throws `UnsupportedVersionException`.

The Default Version

The default version belongs to the `VersionRegistry`:

```
use Derafu\Ubl\Service\FormatHandlerFactory;
use Derafu\Ubl\Service\VersionRegistry;
use Derafu\Ubl\Service\Deserializer;

$deserializer = new Deserializer(new FormatHandlerFactory(versions: new
VersionRegistry('2.5')));
```

With Symfony's dependency injection (see **Services**) it is the parameter `derafu_ubl.version`, which an application overrides in its own `services.yaml`:

```
parameters:
    derafu_ubl.version: '2.5'
```

What Changes Between Versions

UBL versions only add. Between 2.1 and 2.5 no document, component or element is removed, and every new element is optional. What changes in what already existed is that some cardinalities are relaxed: a `0..1` that becomes `0..n`, or a required element that becomes optional.

Because of that:

- **What was valid in a version is valid in the later ones.** A 2.1 document read with the model of 2.5 and written again validates against the 2.5 XSD.
- **The model of each version has the shape of its own schema.** A relaxed cardinality changes the PHP type: in 2.1 the street of an address is one value (`Address::getStreetName()`), in 2.5 it is a list. That is why each version has its own classes and the latest one does not replace the others.

Version Registry

`Contract\VersionRegistryInterface (Service\VersionRegistry)` has the versions:

```
use Derafu\Ubl\Service\VersionRegistry;

$versions = new VersionRegistry();

$versions->getDefault()->getId(); // '2.1'
```

```
$versions->get('2.2')->getNamespace(); // 'Derafu\Ubl\V22\Oasis'  
$versions->forClass(Derafu\Ubl\V23\Oasis\Doc\Invoice::class)->getId(); // '2.3'  
  
foreach ($versions->all() as $version) {  
    echo $version->getId(), "\n";  
}
```

A `Contract\VersionInterface` (`Service\Version`) knows everything about one version: the namespace of its model, the document class of a name (`getDocumentClass('Invoice')`), the root element of a document class and where its XSD and its JSON schemas are.

JSON and YAML

JSON and YAML are read and written the same way in every version, but OASIS has published the JSON representation (and its schemas) only for UBL 2.1, 2.2 and 2.3. Validating the JSON or YAML of 2.4 or 2.5 is not possible: it throws a `ValidationException` saying so (see **Validation**).

Last updated on 13/10/2026