Einführung

Der Verband Schweizer Schreiner Software VSSS hat in Zusammenarbeit mit Berufsverbänden und innovativen Anbietern einen gemeinsamen Standard namens ComNorm für den Datenaustausch entwickelt.

Damit können Online-Produktdaten direkt aus Schreinersoftwarelösungen abgefragt und genutzt werden. Darüber hinaus ermöglicht die ComNorm-Schnittstelle die Übertragung und den Austausch von Geschäftsdokumenten.

Die bisherige ComNorm-Schnittstelle ist seit über 20 Jahren im Einsatz. ComNormAPI ist eine neue Version der Schnittstellendefinition. Sie berücksichtigt gesammelte Erfahrungen, neue Erkenntnisse, stattgefundene Entwicklungen und aktuelle Technologien.

Zweck des Dokuments

Dieses Dokument ist für Softwareentwickler der Softwarepartner und Lieferanten bestimmt. Es enthält die Definition und die technischen Details der ComNormAPI-Schnittstelle. Es beschreibt die Architektur, die Prozesse, die Anwendungsprogrammierschnittstelle sowie die verwendeten Datenstrukturen.

Punkte in der Schnittstellen-Dokumentation, welche noch nicht definitiv entschieden wurden und sich bis zur ersten ComNormAPI-Freigabe noch verändern können sind mit solchen gelben Boxen markiert.

Änderungen gegenüber der Vorgängerversion

ComNormAPI enthält Anpassungen in den folgenden Bereichen:

- Der Zugriff auf die Daten des Providerservers erfolgt nicht mehr in einem Benutzerkontext, der einen Login-Prozess erforderte, sondern automatisch über eine Server-zu-Server-Kommunikation und einen hinterlegten API-Schlüssel.
- Die Datenbeschreibungssprache wurde von XML auf JSON umgestellt.
- Methoden zur Suche und zum Durchblättern von Produktkatalogen der Lieferanten wurden aus der API entfernt.
- Die Funktionen im Bereich des Dokumentenaustauschs wurden angepasst.
- Es wurde ein Mechanismus zur Ermittlung von anbieterspezifischen URLs hinzugefügt.
- Es wurden neue Möglichkeiten entwickelt, um die Datenübernahme zu ermöglichen.

Architektur

Die ComNormAPI-Architektur wird in 5 Akteure aufgeteilt:

- **Softwarepartner (ERP):**

  Software, welche mit der ComNorm-Provider-API und mit mehreren ComNorm-Lieferanten-API's kommunizieren kann.
- **ComNorm-Provider REST-API (ComNorm):**

  Von ComNorm zur Verfügung gestellte API zum Abfragen der Provider listen.
- **ComNorm-Lieferanten REST-API (Supplier Backend):**

  Vom Lieferanten zur Verfügung gestellte API für den Datenaustausch zwischen Softwarepartner und Lieferanten. Jeder Lieferant bietet eine separate REST-API an.
- **Webshop des Lieferanten (Supplier Frontend):**

  Im Browser zugängliches Shopsystem des Lieferanten, in dem ein Anwender interagiert.
- **Notifikationsserver fürs Live-Browsing:**

  Um die im Live-Browsing zusammengeklickte Produkte vom Webshop ins ERP zu importieren, werden die Produktnummern auf einem Notifikationsserver gesammelt. Es gibt verschiedene Varianten von Notifikationsservern:
  - *Option 1: Notifikationsserver bei ComNorm:*

    ComNorm bietet einen Notifikationsserver an, welcher die im Lieferantenwebshop ausgewählten Produkte sammelt.
  - *Option 2 - Notifikationsserver direkt beim Softwarehaus-Kunden:*

    Wenn der Softwarehaus-Kunde beispielsweise eine Webapplikation benutzt, kann vom Softwarehaus ein Notifikationsserver direkt an die Applikation angebunden werden. In diesem Fall kann der Notifikationsserver die Applikation direkt notifizieren.
  - *Option 3: Zentraler Notifikationsserver des Softwarehauses:*

    Das Softwarehaus bietet einen zentralen Notifikationsserver an, welcher von allen Kunden der Software benutzt werden kann.

Für die Kommunikation zwischen den verschiedenen Akteuren werden unterschiedliche Authentifikationen verwendet:

- **PartnerKey:** Kommunikation zwischen Softwarepartner und ComNorm-Provider REST-API. Der Partner-Key kann auch verwendet werden um den den ComNorm-Notifikationsserver anzusteuern.
  - Ihren firmenspezifischen Partner-Key erhalten Sie unter **https://api.comnorm.com/comnormkey**
- **ComNormKey:** Kommunikation zwischen Softwarepartner und ComNorm-Lieferanten REST-API

- Der **Webshop** des Lieferanten hat eine von der **Schnittstelle unabhängige Authentifizierung**. Diese kann sich von Lieferant zu Lieferant unterscheiden.
- Für die oben beschriebenen Notifikationsserver-Optionen 2 und 3 ist das Softwarehaus selbst für die Kommunikation und Authentifikation zuständig.

Beispiele zur Kommunikation zwischen den verschiedenen Akteuren werden im Kapitel Use-Cases aufgezeigt.

ComNormAPI Provider-Service (ComNorm-Server-API)

Der ComNorm Provider Service wird zentral von ComNorm GmbH gehostet und aktuell gehalten.

ComNorm-Server-API-Sicherheit

Um sich als Softwarepartner gegenüber von ComNorm zu authentifizieren benötigt man einen PartnerKey, der von ComNorm bereitgestellt wird.
Der PartnerKey wird als Authorization-Attribut im HTTP-Request-Header wiefolgt übermittelt:

```
X-PARTNER-KEY: {PartnerKey}
```

Zum Beispiel:

```
X-PARTNER-KEY: a7c00dd8-b3d1-4bb0-b00c-153ffdb9cff7
```

Wenn der PartnerKey per HTTP-Request nicht übergeben wird, ist die Anfrage nicht authentifiziert und man erhält den HTTP-Status-Code 401 (Unauthorized) zurück. Wird ein ungültiger PartnerKey übergeben, erhält man den HTTP-Status-Code 403 (Forbidden). Im Fehlerfall wird ein Error-Objekt gemäss https://datatracker.ietf.org/doc/html/rfc7807 zurückgeliefert. Dies beinhaltet weitetere Informationen zum Fehler, kann aber auch leer sein (leeres JSON-Objekt).

REST Request Providers

Die Lieferanten (Providers) können mittels folgendem Request abgefragt werden:

	Provider-List Request
Method	**GET**
URL	**https://api.comnorm.com/providers/**
Parameter	area={AreaCode} (optional)
Header	X-PARTNER-KEY: {PartnerKey}
Response	JSON {ServiceList}

Parameter-Beschreibung

area (optional)

- area: Um die Provider-Liste einzuschränken, kann optional der Parameter "area" übermittelt werden. Als Parameterwert kann ein AreaCode übergeben werden. Verfügbare AreaCodes können über den Aufruf Areas-Request aufgerufen werden.

Header-Beschreibung

X-PARTNER-KEY: {PartnerKey}

Um sich als Softwarepartner gegenüber ComNorm zu authentifizieren benötigt der Softwarepartner einen PartnerKey, der von ComNorm bereitgestellt wird.
Wenn der PartnerKey per Request nicht übergeben wird, ist der Softwarepartner nicht authentifiziert und erhält ein leeres JSON.

Antwort

Die ServiceList enthält alle Lieferanten mit Detailinformationen wie beispielsweise Name, Service-URL, Lieferantenlogo.
Die wichtigste Information ist im Array "backendURL" abgelegt. Dieses beinhaltet für jede Lieferanten-API Version eine API-Kommunikationsurl.

Beispiel

```
[
  {
    "serviceId": 1234,
    "serviceName": [
      { "languageCode": "de",
        "text": "Demoshop AG"
      },
      { "languageCode": "en",
        "text": "Demoshop AG"
      },
      { "languageCode": "fr",
        "text": "Demoshop SA"
      }
    ],
    "serviceDesc": [
      { "languageCode": "de",
        "text": "Über 100'000 Artikel für den Holzbau"
      },
      { "languageCode": "en",
        "text": "More than 100'000 products for wood constructions"
      }
    ],
    "backendURL" :
    [
      {
        "version": "1.0",
        "url" : "https://api.demoshop.com/1.0"
      },
```

```
        {
          "version": "1.1",
          "url" : "https://api.demoshop.com/test/2.0"
        }
      ],
      "serviceURL": "https://demoshop.ch",
      "serviceIconURL": "https://demoshop.ch/logo/icon.jpg",
      "serviceLogoURL": "https://demoshop.ch/logo/logo.png"
    },
    {
      ..
    }
  ]
```

REST Request Areas

Mit dem Areas-Request können alle von ComNorm verfügbaren Areas abgeholt werden,

	ServiceArea-List Request
Method	**GET**
URL	**https://api.comnorm.com/areas/**
Header	X-PARTNER-KEY: {PartnerKey}
Response	JSON{ServiceAreaList}

Parameter-Beschreibung

Keine Parameter für Request vorgesehen.

Header-Beschreibung

X-PARTNER-KEY: {PartnerKey}

Um sich als Softwarepartner gegenüber ComNorm zu authentifizieren benötigt der Softwarepartner einen PartnerKey, der von ComNorm bereitgestellt wird.
Wenn der PartnerKey per Request nicht übergeben wird, ist der Softwarepartner nicht authentifiziert und erhält ein leeres JSON.

Antwort

Die ServiceAreaList enthält alle von ComNorm definierten Areas. Eine ServiceArea liefert jeweils einen "areaCode" (Identifikation des Bereiches) und eine "areaName" (Name des Bereiches).

Beispiel

```
[
  {
    "areaCode": "CH",
    "areaName": [
      {
        "languageCode": "de",
        "text": "Schweiz"
      },
      {
        "languageCode": "fr",
        "text": "Suisse"
      },
      {
        "languageCode": "en",
        "text": "Switzerland"
      }
    ]
  },
  {
    "areaCode": "DE",
    "areaName": [
      {
        "languageCode": "de",
        "text": "Deutschland"
      },
      {
        "languageCode": "fr",
        "text": "Allemagne"
      },
      {
        "languageCode": "en",
        "text": "Germany"
      }
    ]
  }
]
```

Lieferanten REST-API

Dieses Kapitel befasst sich mit der vom Lieferanten zu implementierenden API. Die Requests werden von den Software-Partnern zu einem Lieferantensystem (Backend) ausgeführt.

API-Sicherheit

Um sich als Firma gegenüber dem Lieferanten zu authentifizieren, benötigt diese einen ComNormKey, der vom Lieferanten bereitgestellt wird. Wie der ComNormKey vom Lieferanten zur Firma gelangt, ist dem Lieferanten überlassen.
Der ComNormKey wird als Authorization-Attribut im HTTP-Request-Header wiefolgt übermittelt:

```
X-COMNORM-KEY: {ComNormKey}
```

Zum Beispiel:

```
X-COMNORM-KEY: 276ae1f7-2bd3-4ac7-99c4-5f65cd09d0f7
```

Wenn der ComNormKey per HTTP-Request nicht übergeben wird, ist die Anfrage nicht authentifiziert und man erhält den HTTP-Status-Code 401 (Unauthorized) zurück. Wird ein ungültiger ComNormKey übergeben, erhält man den HTTP-Status-Code 403 (Forbidden). Im Fehlerfall wird ein Error-Objekt gemäss https://datatracker.ietf.org/doc/html/rfc7807 zurückgeliefert. Dies beinhaltet weitetere Informationen zum Fehler, kann aber auch leer sein (leeres JSON-Objekt). Wie detailliert das Error-Objekt zurückgeliefert wird ist dem Lieferanten überlassen.

Wichtig

- **ComNormKey (API-Key):** Ein ComNormKey ist einer Firma zugewiesen und nicht einem Benutzer. Das heisst, das Benutzerinformation auf dem Account des Webshops zur Verfügung gestellt werden.
- **Trennung Frontend/Backend:** Die Authentifikation auf dem Lieferanten-Backend Server (API) wird strikt getrennt werden von der Authentifikation des Lieferanten-Frontends bzw. Shopping-Systems.
- **Interaktionen eines Webshop-Benutzers:** Für das Aufrufen der Live-Browsing-URL wird kein ComNormKey (API-Key) verwendet. Mit der URL gelangt man lediglich zum Shopping-System des Lieferanten. Für die Authentifizierung des Benuzers gegenüber dem Webshop, muss sich der Benutzer im Webshop einloggen.

Allgemein

Folgende Konventionen wurden für die ComNormAPI definiert:

Für die URIs wird die REST-Architektur verwendet. Dabei wird ein abschliessender Slash ("/") verwendet, wenn es sich bei der Ressource um eine Collection (z.B. die Liste der Service-Areas:

https://api.comnorm.com/providers/) handelt. Bei einer einzelnen Ressource (z.B. die Provider-Info: {backendURL}/info) wird kein abschliessender Slash verwendet.

HTTP-Request (Anfrage)

- Der Body der Requests wird im JSON Format gemäss den definierten JSON-Schemen definiert

HTTP-Request-Header

Ein HTTP-Request-Header auf die ComNormAPI muss immer ein Feld für die Zugriffs-Autorisierung sowie Informationen zum Softwarehaus enthalten. Diese Informationen werden mit den Attributen **X-COMNORM-KEY** und **X-CLIENT-APPLICATION** im HTTP-Request-Header wie folgt übermittelt.

X-COMNORM-KEY: {ComNormKey}

Ein ComNormKey ist einer Firma zugewiesen und nicht einem Benutzer. Das heisst, das Benutzerinformation auf dem Account des Webshops zur Verfügung gestellt werden.

Beispiel:

```
X-COMNORM-KEY: 276ae1f7-2bd3-4ac7-99c4-5f65cd09d0f7
```

Wenn der ComNormKey per HTTP-Request nicht übergeben wird, ist die Anfrage nicht authentifiziert und man erhält den HTTP-Status-Code 401 (Unauthorized) zurück. Wird ein ungültiger ComNormKey übergeben, erhält man den HTTP-Status-Code 403 (Forbidden). Im Fehlerfall wird ein Error-Objekt gemäss https://datatracker.ietf.org/doc/html/rfc7807 zurückgeliefert. Dies beinhaltet weitetere Informationen zum Fehler, kann aber auch leer sein (leeres JSON-Objekt).

X-CLIENT-APPLICATION: {ClientApplication}

Zusätzlich werden im HTTP-Request-Header Informationen zum Softwarehaus übergeben (Software/Version (Optionen)).

Beispiel:

```
X-CLIENT-APPLICATION: SoftwarehausERP/24.05 (Partlist)
```

HTTP-Response (Antwort)

Als HTTP-Response kann der Benutzer folgende Formate als Antwort erhalten:

- **JSON im Response-Body:** Antwort im JSON-Format gemäss den definierten JSON-Schemen.
- **Binary-Data:** Der File-Request liefert direkt eine binäre Datei zurück.
- **HTTP-Status-Code**

Antwort HTTP-Status-Codes

Von ComNormAPI wird der Gebrauch der folgenden HTTP-Status-Codes empfohlen:

Code	Status	Meaning
200	Ok	Die Anfrage wurde erfolgreich bearbeitet und das Ergebnis der Anfrage wird in der Antwort übertragen.
201	Created	Die Anfrage wurde erfolgreich bearbeitet. Die angeforderte Ressource wurde vor dem Senden der Antwort erstellt. Das „Location"-Header-Feld enthält eventuell die Adresse der erstellten Ressource.
202	Accepted	Die Anfrage wurde akzeptiert, wird aber zu einem späteren Zeitpunkt ausgeführt. Das Gelingen der Anfrage kann nicht garantiert werden.
204	No Content	Der 204 No Content HTTP-Status-Code zeigt an, dass der Server die Anfrage erfolgreich erfüllt hat und dass es keinen weiteren Inhalt in der Nutzlast der Antwort zu senden gibt.
400	Bad Request	Die Anfrage-Nachricht war fehlerhaft aufgebaut.
401	Unauthorized	Die Anfrage kann nicht ohne gültige Authentifizierung durchgeführt werden. Wie die Authentifizierung durchgeführt werden soll, wird im „WWW-Authenticate"-Header-Feld der Antwort übermittelt. Beispiel für Lieferanten-Shops: `WWW-Authenticate: X-COMNORM-KEY`
403	Forbidden	Die Anfrage wurde mangels Berechtigung des Clients nicht durchgeführt, bspw. weil der authentifizierte Benutzer nicht berechtigt ist oder ein als HTTPS konfigurierter URL nur mit HTTP aufgerufen wurde.
404	Not Found	Die angeforderte Ressource wurde nicht gefunden. Dieser Statuscode kann ebenfalls verwendet werden, um eine Anfrage ohne näheren Grund abzuweisen. Links, die auf solche Fehlerseiten verweisen, werden auch als Tote Links bezeichnet. „404" gilt zudem als verbreitetes Meme.
406	Not Acceptable	Die angeforderte Ressource steht nicht in der gewünschten Form zur Verfügung. Gültige „Content-Type"-Werte können in der Antwort übermittelt werden.
500	Server Error	Dies ist ein „Sammel-Statuscode" für unerwartete Serverfehler.
501	Not Implemented	Die Funktionalität, um die Anfrage zu bearbeiten, wird von diesem Server nicht bereitgestellt. Ursache ist zum Beispiel eine unbekannte oder nicht unterstützte HTTP-Methode.

API-Requests Lieferanten

In diesem Kapitel werden die ComNormAPI-Requests beschrieben.
Die Requests werden auf der {backendURL} (API-URL des Lieferanten) vom Lieferanten ausgeführt.

Überblick Endpunkte

Hier ein Überblick der möglichen Lieferanten-API-Requests.

API-Request	Methode	zwingend / optional
/info	GET	
/products/	POST	
/fileContent	GET	
/contentTransfer	POST	
/urls/	POST	
/containers/	GET	

REST Request Provider Info

Weitere Informationen zum Anbieter (Adresse, Unternehmenidentifikationsnummern, ...) können mittels folgendem Request abgefragt werden:

	Provider-Info Request
Method	**GET**
URL	**{backendURL}/info**
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Response	JSON {ProviderInfo}

Parameter-Beschreibung

Keine Parameter für Request vorgesehen.

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Antwort

Die Antwort enthält die Anbieterinformationen gemäss JSON-Schema ProviderInfo.

Beispiel

```
{
  "addressInfo": {
    ...
  },
  "uidch" : "CHE-012.345.678",
  "uideu" : "DE123456789"
}
```

REST-Request Products

Mithilfe des Products-Requests können detaillierte Informationen zu einem Produkt abgefragt werden.

	Products Request
Method	**POST**
URL	**{backendURL}/products/**
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Body	JSON {ProductStateRequestItem}
Response	JSON {ProductInfo}

Parameter-Beschreibung

Keine Parameter für Request vorgesehen.

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Body-Beschreibung

Das ProductStateRequestItem identifiziert das Produkt (mit allfälligen Parametrisierungsinformationen), für welches detaillierte Informationen abgefragt werden sollen.

Beispiel-Body für einen Products-Request:

type: product-state-request-item
JSON: ProductStateRequestItem-Objekt

```
{
  "productNumber": "1234",
  "configurationData": {
    "configurationID": "5678"
  }
}
```

Antwort

Die ProductInfo enthält detaillierte Informationen zum übermittelten Produkt.

```
{
  "productNumber": "1234",
  "productName": [
      { "languageCode": "de",
        "text": "Holzplatte Multiplex Birke 4mm"
      },
      { "languageCode": "en",
        "text": "Wooden panel multiplex birch 4mm"
      },
      { "languageCode": "fr",
        "text": "Panneau de bois Multiplex Bouleau 4mm"
      }
  ],
     ....
}
```

REST-Request FileContent

Mit dem FileContent-Request kann eine binäre Datei gelesen werden.
Diese Funktion wird vorallem verwendet um sensitive Daten ohne öffentliche URL abzuholen.

	File Content Request
Method	**GET**
URL	**{backendURL}/fileContent/{FileId}**
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Response	Binary Data

Parameter-Beschreibung

Mit der {FileId} wird definiert, welche Datei gelesen werden soll. Die FileId kann von eine am Produkt hängende Datei oder ein Dokument (bspw. eine Rechnung) selbst sein.

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Antwort

Die Datei wird als Binäre Datei zurückgeliefert.

Rest-Request ContentTransfer

Der ContentTransfer-Request ist noch nicht endgültig definiert.
Insbesondere müssen die Bezeichnungen der Typen noch fixiert.

Zudem werden noch Use-Cases beschrieben, damit mögliche Abläufe und Datenflüsse noch klarer werden.

Mithilfe des ContentTransfer-Requests können Inhalte verschiedener Art an den Lieferanten übermittelt werden. Dies kann beispielsweise genutzt werden um Bestellungen auszulösen oder um detaillierte Informationen zu mehreren Produkten und deren Validierungsstatus abzufragen.

	Content Transfer Request
Method	**POST**
URL	**{backendURL}/contentTransfer/{type}**
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Body	JSON {document-*}/{productstaterequestlist}
Response	JSON {ContentTransferResponse}

Route-Beschreibung

Mit {type} wird der Typ des JSON-Objekts im Body angegeben.

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Body-Beschreibung

Von der ComNormAPI-Schnittstelle definierte Typen für den Inhalt:

Typ	**Beschreibung**	**JSON-Objekt**
document-inquiry	Anfrage	ContentInquiryInfo
document-offer	Angebot	ContentOfferInfo
document-order	Bestellung	ContentOrderInfo
document-confirmation	Bestätigung	ContentConfirmationInfo
document-invoice	Rechnung	ContentInvoiceInfo
productstate-requestlist	Anfrageliste fuer Produktaktualisierung	ContentProductStateRequestListInfo

Beispiel-Body für eine Bestellung:

type: document-order
JSON: ContentOrderInfo-Objekt

```
{
  "header":
  { ..
  },
  "body":
  { ..
  },
  "footer":
  { ..
  }
}
```

Beispiel-Body für eine Produktaktualisierung:

type: productstate-requestlist
JSON: ContentProductStateRequestListInfo-Objekt

```
{
  "header":
  { ..
  },
  "body":
  {
    "productStateRequestList": [
      { "productNumber": "1234",
        "configurationData": {
          "configurationID": "4321"
        }
      },
      { "productNumber": "5678" }
    ]
  },
  "footer":
  { ..
  }
}
```

Antwort

Als Antwort eines übermittelten Inhalts erhält der Softwarepartner vom Lieferanten ein ContentTransferResponse-Objekt.
Darin kann der Lieferant eine Meldung zur Akzeptanz des Inhalts verfassen. Ob ein Inhalt vom Lieferantensystem akzeptiert wurde oder nicht wird über den HTTP-Status-Code definiert:

HTTP-Status-Code	Beschreibung
200 (OK)	akzeptiert
406 (Not Acceptable)	nicht akzeptiert

Beispiel-Antwort mit einem ContentTransferResponse-Objekt:

```
  { "containerId": "123456789",
    "reponseMessage": "Die Bestellung wurde erfolgreich an den Web-Shop
übermittelt und muss per definierter URL definitiv bestätigt werden.",
  }
```

Der durch die ContainerId definierte Container ist mittels ContainerContent-Request abzurufen. Bei einer Bestellung kann es sein, dass der Container das Attribut "contentState" auf CONTENT_STATE_AWAITING_CONFIRMATION gesetzt hat (siehe Use-Cases). Bei einer Produktaktualisierung ist es wahrscheinlich, dass der Container das Attribut "contentState" initial auf CONTENT_STATE_PENDING gesetzt hat.

Rest-Request URL-Factory

Der URL-Factory Request liefert eine URL zu einem bestimmten Zweck, welcher über den Typ spezifiziert wird.

	Url Factory Request
Method	**POST**
URL	**{backendURL}/urls/{type}**
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Response	JSON {URLInfo}

Route-Beschreibung

Mit dem Paramter {type} wird der Zweck der URL-Factory definiert.
Mögliche Zwecke für den URL-Factory-Request:

Type	Beschreibung	Body
shop-context	Live-Browsing mit Möglichkeit der Produkt-Notifikation an die im Body definierte URL	JSON {NotificationInfo}
shop-product	URL für das im Body definierte Produkt	JSON {ProductItem}

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Antwort

Die Antwort enthält die Informationen gemäss JSON-Schema URLInfo.
Die darin enthaltene URL kann vom Softwarepartner angesprungen werden.

Damit die Eindeutigkeit beim Aufruf aus unterschiedlichen ERP-Kontexten (z.B. aus Stückliste und aus Lagerbewirtschaftung, oder von User A und User B) gewährleistet ist, gilt:

- Es ist Sache des ERPs, für jeden ERP-Kontext eine andere Notifikations-URL zu verwenden.
- Die URL-Factory liefert für jeden Aufruf vom Typ shop-context mit einer anderen Notification-URL eine andere eindeutige Shop-URL zurück. Wird diese aufgerufen, so ist garantiert, dass der Shop an die angegebene Notifkations-URL notifiziert.
- Mehrfache Aufrufe mit der gleichen Notifikations-URL können, müssen aber nicht die gleiche Shop-URL zurückgeben.

Beispiel-Body für Live-Browsing-URL:

type: shop-context
JSON: NotificationInfo-Objekt

```
{
  "notificationURL": "https://demosoftware.ch/notification/abcdefgh",
  "notificationHint": "An Demo-Software übermitteln"
}
```

Beispiel-Antwort für Live-Browsing-URL

```
{
 "url": "https://demoshop.ch/live-browsing?notification_id=123456"
}
```

Beispiel-Body für eine Produktseite:

type: shop-product
JSON: ProductItem-Objekt

```
{
  "productNumber" : "123456-a"
}
```

Beispiel-Antwort für Produktseite

```
{
 "url": "https://demoshop.ch/live-browsing?notification_id=123456"
}
```

Rest-Request ContainerList

Mit dem ContainerList-Request kann eine hierarchisch flache Liste mit Container abgeholt werden. Ein Container kann Dokumente sowie auch Produktlisten beinhalten. Die ContainerInfoList dient zur Auflistung aller Container und beinhaltet daher nur oberflächliche Informationen der Container.

	Container List Request
Method	**GET**
URL	**{backendURL}/containers/**
Parameter	type={type} (Optional)
	name={name} (Optional)
	ownerId={ownerId} (Optional)
	containerId={containerId} (Optional)
	timestamp={timestamp} (Optional)
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Response	JSON {ContainerInfoList}

Von der ComNormAPI-Schnittstelle definierte Container-Content-Typen und Produktlisten:

Typ	**Beschreibung**	**JSON-Objekt**
document-inquiry	Anfrage	ContentInquiryInfo
document-offer	Angebot	ContentOfferInfo
document-order	Bestellung	ContentOrderInfo
document-confirmation	Bestätigung	ContentConfirmationInfo
document-invoice	Rechnung	ContentInvoiceInfo
productstate-responselist	Produktstatus-Antwortliste	ContentProductStateResponseListInfo

Parameter-Beschreibung

Die ContainerInfo-Liste kann optional über folgende Parameter eingeschränkt werden:

- type: Filtert die Liste nach ContainerType. Die Filterung nach "document-invoice" liefert alle Rechnungen zurück.
- name: Filtert die Liste nach dem Namen des Containers
- ownerId: Filtert die Liste nach Container mit gegebener Benutzeridentifikation. Die Benutzeridentifikation wird vom Lieferanten definiert und ist in den meisten Fällen eine E-Mail-Adresse.
- containerId: Filtert die Liste nach ContainerId. Wird beispielsweise verwendet wenn man die URL eines Containers im Browser aufrufen möchte.
- timestamp: Filtert die Liste nach einem Timestamp. Beispiel: Ein Software-Partner möchte nur die Container aus den letzten 90 Tage berücksichtigen.

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Antwort

Die Antwort enthält alle Container gemäss JSON-Schema ContainerInfoList.

Beispiel


```
[
 {
   "containerId": "789453",
   "name": "2.11-Bestellung",
   "type": "productlist-order",
   "timestamp": "2024-08-27T14:56:53",
   "url": "https://demoshop.com/Data/Container/24680",
   "contentState": "CONTENT_STATE_COMPLETED"
 },
 {
   "containerId": "123456",
   "name": "1.04-Bestatigung",
   "type": "document-confirmation",
   "timestamp": "2024-08-27T14:56:53",
   "url": "https://demoshop.com/Data/Container/123456",
   "contentState": "CONTENT_STATE_COMPLETED"
 },
 {
   "containerId": "987654",
   "name": "5.02-Angebot",
   "type": "document-offer",
   "timestamp": "2024-08-27T14:56:53",
   "url": "https://demoshop.com/Data/Container/987654",
   "contentState": "CONTENT_STATE_COMPLETED"
 },
 ..
]
```

Beispiel einer Produktstatusliste, welche noch in Aufbereitung ist

```
{
  "containerId": "12345-56789-6789",
  "name": "Produktstatusliste XYZ",
  "type": "productstate-responselist",
  "contentState": "CONTENT_STATE_PENDING"
  ....
}
```

Wenn die Produktstatusliste dann bereit steht ("contentState": "CONTENT_STATE_COMPLETED") können die Informationen mittels ContainerContent-Request abgerufen werden.
Ist die Produktstatusliste bereit, enthält der Inhalt des Containers eine ContentProductStateResponseListInfo mit Informationen zu den übermittelten Produkten. Die Liste im Body beinhaltet eine Liste von ProductStateResponseItem-Objekten, welche die Produktnummer, den Validierungsstatus ("validateState") des Produktes sowie die detaillierten Produktinformationen ProductInfo beinhaltet.

Folgende Status für angefragte Produkte sind möglich (validateState):

Status	Beschreibung
"PRODUCT_STATE_OK"	Produkt ist vorhanden
"PRODUCT_STATE_DELETED"	Produkt existiert nicht mehr
"PRODUCT_STATE_SUBSTITUTED"	angefragtes Produkt wurde durch ein anderes Produkt ersetzt
"PRODUCT_STATE_UNKNOWN"	Produkt ist unbekannt. ProductInfo ist leer.

Rest-Request ContainerContent

Mit dem ContainerContent-Request kann ein einzelner Container mit allen Informationen gelesen werden.

	ContainerContent Request
Method	**GET**
URL	**{backendURL}/containers/{containerId}**
Header	X-COMNORM-KEY: {ComNormKey}
	X-CLIENT-APPLICATION: {ClientApplication}
Response	JSON {ContainerInfo}

Von der ComNormAPI-Schnittstelle definierte Container-Content-Typen und Produktlisten:

Typ	Beschreibung	JSON-Objekt
document-inquiry	Anfrage	ContentInquiryInfo

Typ	Beschreibung	JSON-Objekt
document-offer	Angebot	ContentOfferInfo
document-order	Bestellung	ContentOrderInfo
document-confirmation	Bestätigung	ContentConfirmationInfo
document-invoice	Rechnung	ContentInvoiceInfo
productstate-responselist	Produktstatus-Antwortliste	ContentProductStateResponseListInfo

Route-Beschreibung

Mit der {containerId} wird definiert, welchen Container gelesen werden soll.

Header-Beschreibung

Standard-Request-Header: HTTP-Request-Header

Antwort

Die Antwort enthält den Inhalt eines Containers gemäss den oben aufgeführten Container-Content-Typen.

Beispiel mit Bestellung (OrderInfo)

```
{
  "header":
  { ..
  },
  "body":
  { ..
  },
  "footer":
  { ..
  }
}
```

Use-Cases

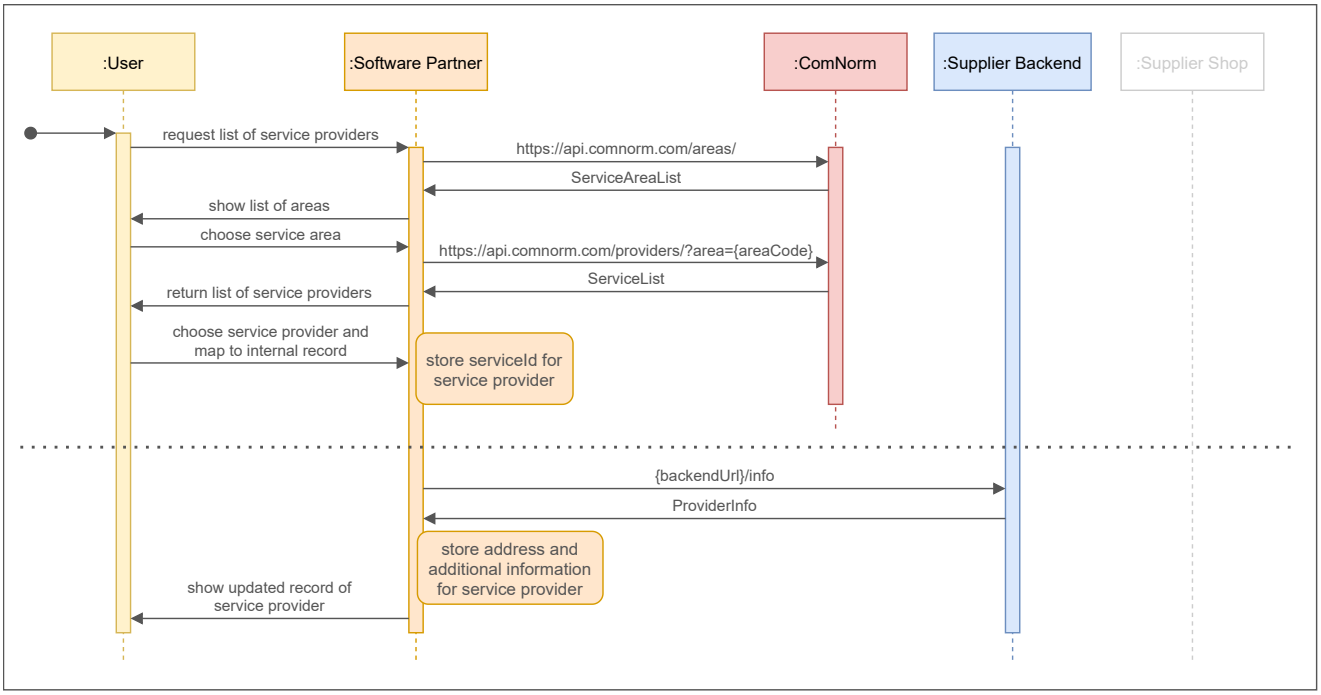
In diesem Kapitel werden verschiedene Anwendungsbeispiele der ComNormAPI beschrieben. Die Beschreibung dient dem Verständnis der Prozesse und vereinfacht so die Implementation der Schnittstelle für die Softwarepartner, sowie auch für die Lieferanten.

Die Use-Cases werden noch ausgearbeitet, damit mögliche Abläufe und Datenflüsse noch klarer werden.

Service-List via ComNorm-Server importieren

Über den ComNormServer können alle Provider (Lieferanten), welche eine ComNormAPI implementiert haben abgeholt werden.

Folgendes Beispiel zeigt auf, wie die Informationen für einen Service vom ComNorm-Server abgeholt werden können und mit Hilfe dieser Informationen erweiterte Informationen zum Lieferanten über dessen API abgeholt werden können.



- **Schritt 1: Areas:**
  Der ERP-Benutzer möchte alle möglichen Service-Areas der Service-Listen abholen, da er schlussendlich beispielsweise nur Lieferanten aus der Schweiz ins ERP importieren möchte. Dazu wird der Areas-Request vom ERP zum ComNorm-Server ausgeführt. Als Antwort wird eine ServiceAreaList zurückgeliefert. Aus dieser Liste wählt der ERP-Benutzer in unserem Beispiel den erhaltenen AreaCode "CH" für die Schweiz aus.

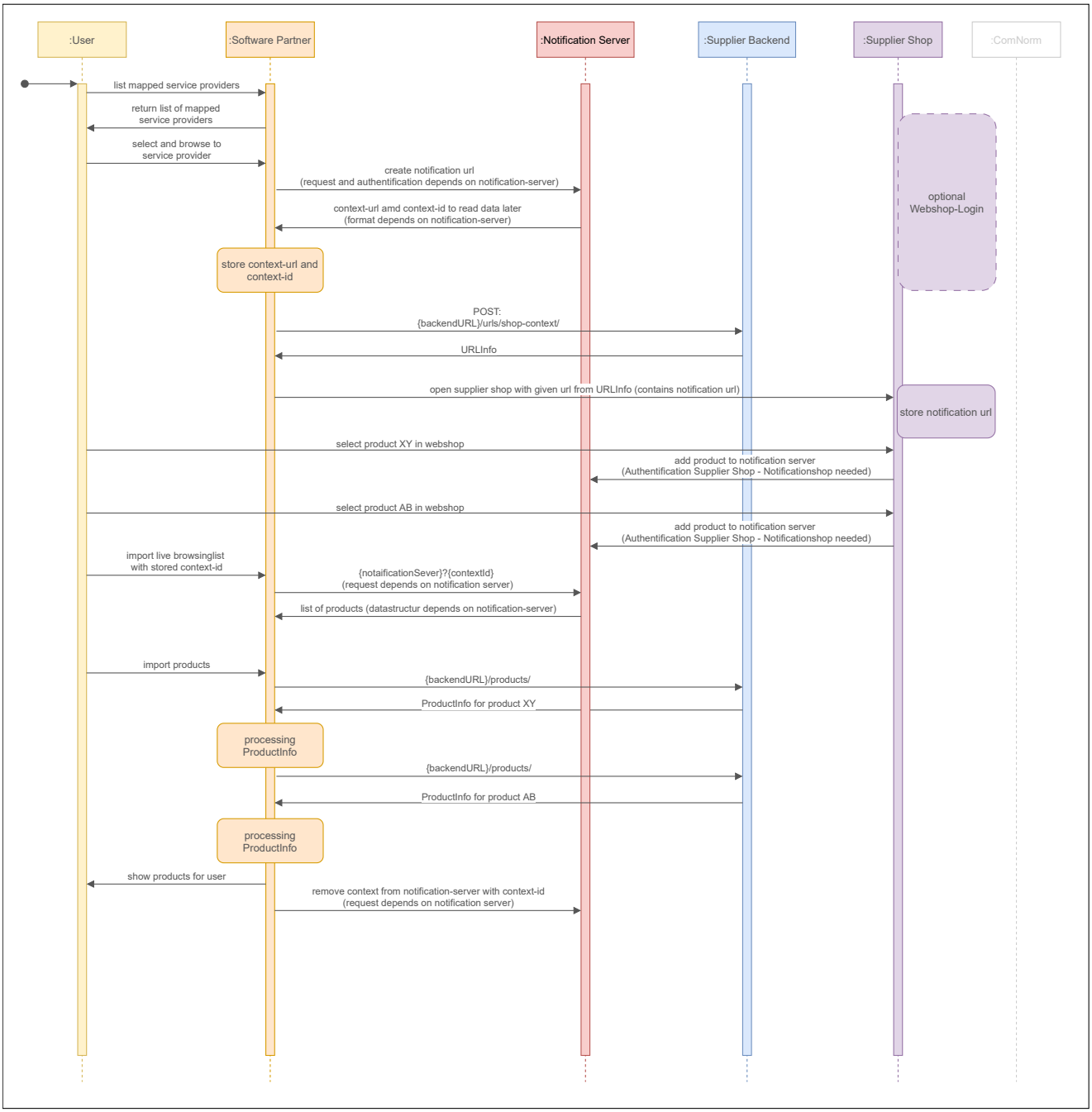
- **Schritt 2: ServiceList für Areas:**
  Im zweiten Schritt holt sich der ERP-Benutzer mithilfe des AreaCodes alle Lieferanten aus der Schweiz. Diese wird durch den Providers-Request vom ERP zum ComNorm-Server abgeholt.

Wichtig: Die Area als Parameter ist optional. Als Antwort erhält das ERP eine nach Area gefilterte ServiceList. Diese kann der ERP-Benutzer mit den ERP-Datensätzen mappen, sodass die Informationen im ERP vorhanden sind.

- **Schritt 3: Weitere Informationen zu einem Lieferanten über dessen ComNormAPI:**
  Mithilfe der abgeholten Provider-Informationen, insbesonderen der {backendURL} aud dem ServiceItem kann das ERP nun mit einer Lieferanten ComNormAPI kommunizieren. Dazu braucht er dann lediglich noch einen ComNormAPI-Key für die Authentifzierung vom ERP zum Lieferanten-Backend.
  Mit dem ProviderInfo-Request kann das ERP dann weitere Informationen wie beispielsweise dem Mehrwertsteuercode des Lieferanten abfragen und im ERP ablegen.

Produkte via Live-Browsing importieren

Das folgende Diagramm bildet den Prozess dar, wie Produkte von einem Lieferanten-Webshop ins ERP übernommen werden können.



- **Schritt 1: Auswahl eines ComNormAPI-Lieferanten:**
  Wenn die ComNormAPI-Lieferanten importiert wurden, kann ein ERP-Benutzer diesen in der Software auswählen.

- **Schritt 2: Erstellen eines Notifikations-Contexts:**
  Nach der Auswahl des Lieferanten wird vom ERP ein neuer Notifikations-Context auf dem Notifikations-Server erstellt. Als Antwort erhält das ERP eine Context-Id und eine Context-URL, welche im System abgelegt werden.

- **Schritt 3: Notifikations-URL mit URL-Factory-Request erstellen:**
  Mithilfe der Context-URL wird per URLFactory-Request eine Einsprungsadresse für das Live-Browsing auf dem Lieferanten-Webshop generiert. Die Einsprungsadresse wird im URLInfo-Objekt zurückgeliefert.
  Der Webshop ist mit dem Aufruf der erhaltenen URL nun in einem speziellen Modus: Produkte können vom Webshop direkt an den Notifikationsserver übermittelt werden. Ob die Buttons "zur

Merkliste hinzufügen" und "Zum Warenkorb hinzufügen" durch einen Button "Produkt notifizieren" ersetzt oder ergänzt werden ist dem Lieferanten selbst überlassen. Der Webshop-Benutzer fügt die Produkte seiner Wahl nun zum Notifikations-Context auf dem Notifikationsserver im Format ProductNotificationInfo hinzu.

- **Schritt 4: Import der ausgewählten Produkte ins ERP:**
  Das ERP kann zu einem beliebigen Zeitpunkt die ausgewählten Produktnummern über den Notifikationsserver ins ERP importieren. Je nach ERP und Notifikationsserver ist es möglich, dass der Notifikationsserver beim Hinzufügen eines Produktes das ERP direkt darüber informiert wird. Das ERP erhält mit der Context-Id die Produktnummern der hinzugefügten Produktnummern.

  Das ERP kann nun über den Products-Request und der Produktnummer (Identifikation des Lieferanten-Produktes) Informationen zu diesem Produkten abholen.

Produkte aus Webshop-Listen (z.B. Warenkorb oder Merklisten) importieren

Das folgende Beispiel zeigt wie eine Liste im Webshop (bspw. Warenkorb oder Merkliste) an den Softwarepartner notifiziert werden kann.

**TODO: drawio **

- **Schritt 1: Auswahl eines ComNormAPI-Lieferanten:**
  Wenn die ComNormAPI-Lieferanten importiert wurden, kann ein ERP-Benutzer diesen in der Software auswählen.

- **Schritt 2: Erstellen eines Notifikations-Contexts:**
  Nach der Auswahl des Lieferanten wird vom ERP ein neuer Notifikations-Context auf dem Notifikations-Server erstellt. Als Antwort erhält das ERP eine Context-Id und eine Context-URL, welche im System abgelegt werden.

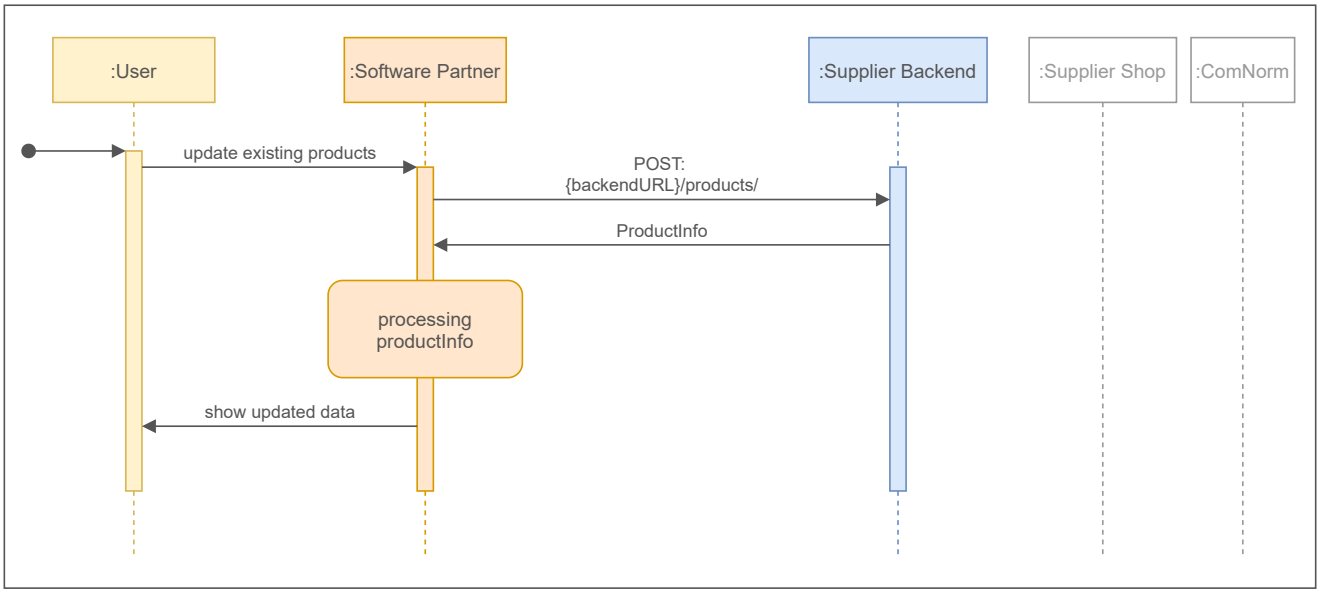
- **Schritt 3: Notifikations-URL mit URL-Factory-Request erstellen:**
  Mithilfe der Context-URL wird per URLFactory-Request eine Einsprungsadresse für das Live-Browsing auf dem Lieferanten-Webshop generiert. Die Einsprungsadresse wird im URLInfo-Objekt zurückgeliefert.
  Der Webshop ist mit dem Aufruf der erhaltenen URL nun in einem speziellen Modus: Produkte können vom Webshop direkt an den Notifikationsserver übermittelt werden. Beim Warenkorb oder einer Merkliste kann ein Button "Alle Produkte notifizieren" platziert werden. Durch diesen Button werden alle Produkte in dieser Liste and zum Notifikations-Context auf dem Notifikationsserver im Format ProductNotificationInfo gesendet.

- **Schritt 4: Import der ausgewählten Produkte ins ERP:**
  Das ERP kann zu einem beliebigen Zeitpunkt die ausgewählten Produktnummern über den Notifikationsserver ins ERP importieren. Je nach ERP und Notifikationsserver ist es möglich, dass der Notifikationsserver direkt darüber informiert wird, dass Produkte hinzugefügt wurden. Das ERP erhält mit der Context-Id die Produktnummern der hinzugefügten Produktnummern.

Das ERP kann nun über den Products-Request und der Produktnummer (Identifikation des Lieferanten-Produktes) Informationen zu diesem Produkten abholen.

Bestehendes ERP-Produkt aktualisieren

Mit Hilfe des Products-Request kann das ERP Informationen von bestehenden Produkten über die ComNormAPI des Lieferanten aktualisieren.

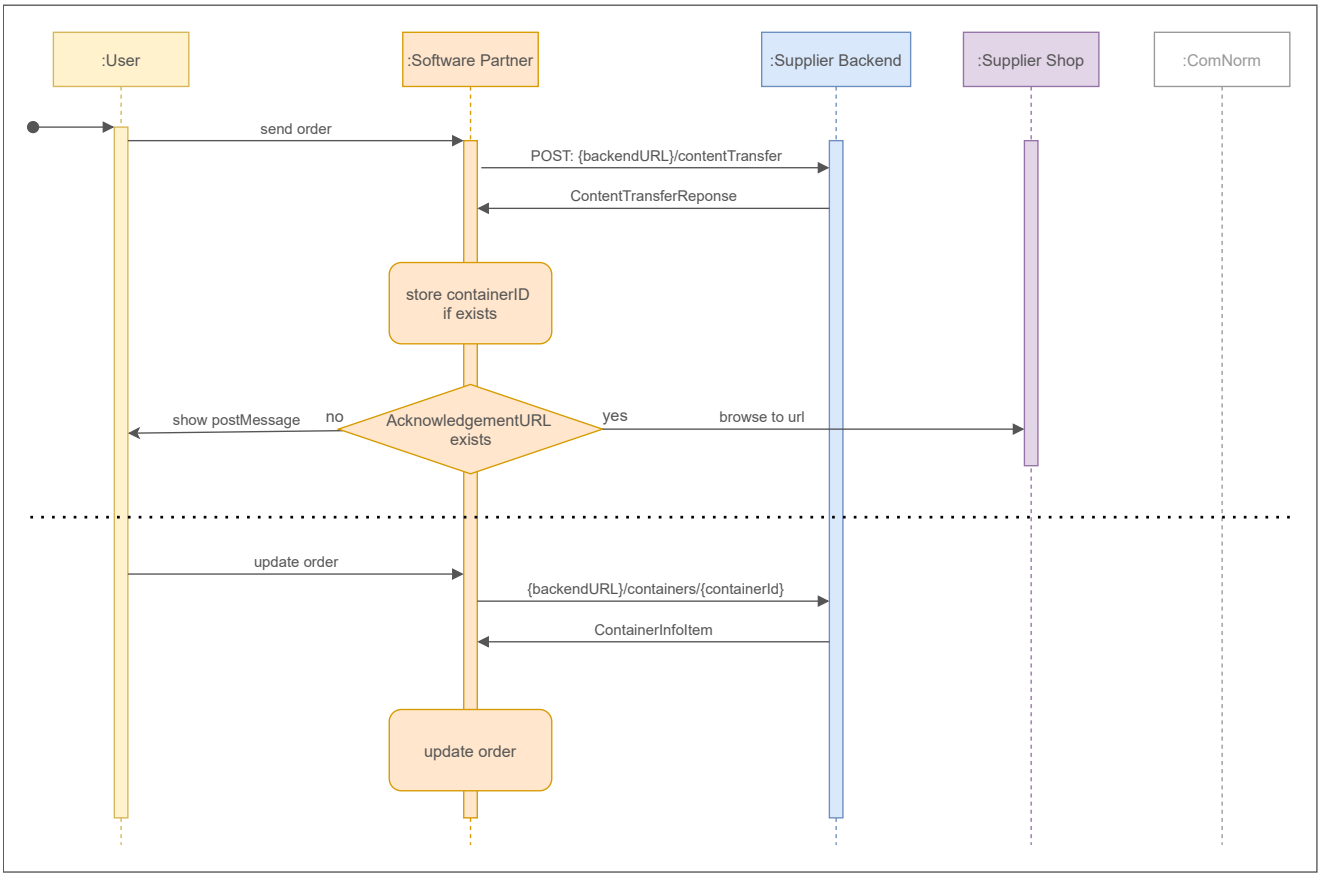


- **Schritt 1: Produkte im ERP auswählen:**
  Der ERP-Benutzer wählt eine Liste von Produkten aus, welche bereits über die ComNormAPI importiert wurden.

- **Schritt 2: Produktinformationen abholen:**
  Das ERP kann über den Products-Request und der Produktnummer (Identifikation des Lieferanten-Produktes) Informationen zu diesem Produkten abholen.
  Als Antwort liefert die ComNormAPI des Lieferanten ein ProductInfo. Das ERP kann nun diese Informationen verarbeiten und den internen Materialstamm aktualisieren.

Bestellung an den Lieferanten übergeben

Für die Übergabe einer Bestellung (Übertrag von Produkten in den Warenkorb des Lieferanten) wird der ContentTransfer-Request vom ERP zur ComNormAPI des Lieferanten verwendet.



- **Schritt 1: Bestellung aus dem ERP abschicken**
  Nach dem der ERP-Benutzer seine Produkte zusammengestellt hat, kann das ERP diese über den ContentTransfer-Request als [ContainerOrderInfo] (Type "document-order") an den Lieferanten-Backendserver übermittelt wird.
  Als Antwort von der ComNorm-API des Lieferanten erhält das ERP ein ContentTransferResponse-Objekt.

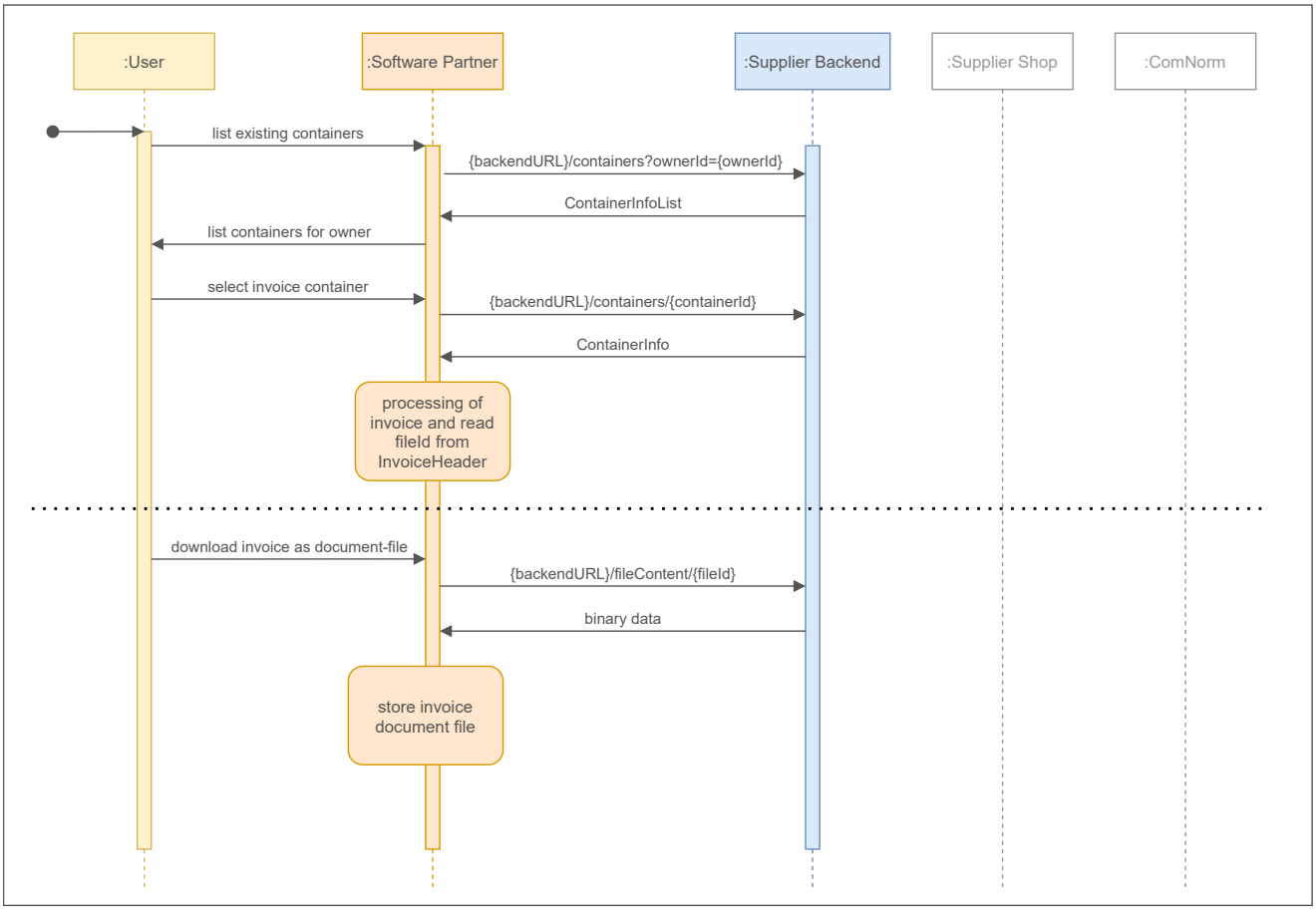
- **Schritt 2: Abschluss der Bestellung**
  Das gelieferte ContentTransferResponse-Objekt entält eine ContainerId der Bestellung. Diese wird im ERP abgelegt. Wenn nun

  - ***Option 1 - Abschluss der Bestellung über Webshop***: Hat der durch die ContainerId im ContentTransferResponse-Objekt definierte Container den Status CONTENT_STATE_AWAITING_CONFIRMATION, muss die Bestellung über die im Container definierte URL abgeschlossen werden
  - ***Option 2 - Bestellung via ContentTransfer-Request bereits abgeschlossen***: Die Bestellung ist durch den Schritt 1 abgeschlossen und die reponseMessage des ContentTransferResponse-Objekt kann am ERP-Benutzer angezeigt werden

- **Schritt 3: Bestellung ins ERP übernehmen:** Mithilfe der ContainerId aus Schritt 2 kann das ERP die Bestellung (Container vom Typ "document-order") mit dem ContainerContent-Request aktualisieren/anlegen.

Rechnung ins ERP Importieren und als Datei herunterladen

Im folgenden Anwendungsfall wird eine Rechnung als Container mit dem ContainerContent-Request und die Rechnungs-Datei mit dem FileContent-Request abgeholt.



- **Schritt 1: ContainerInfo-Liste von bestimmten Webshop-Benutzer abfragen:** Das ERP kann mithilfe der {ownerId} (Webshop-Benutzeridentifikation (Bspw. E-Mail-Adresse)) alle Container eines Benutzers über den ContainerList-Request zum Lieferanten-Backendserver abfragen. Als Antwort erhält das ERP eine ContainerInfoList. Also eine Liste von Containern mit oberflächlichen Container Informationen.

- **Schritt 2: Auswahl einer Rechnung aus der erhaltenen Container-Liste:** Der ERP-Benutzer wählt nun eine Rechnung (ContainerInfoItem-Type "document-invoice") aus.

- **Schritt 3: Weitere Informationen zur ausgewählten Rechnung erhalten:** Um weitere Informationen zur Rechnung zu erhalten kann das ERP mit der ContainerId den ContainerContent-Request zur ComNorm-API des Lieferanten absetzen. Diese liefert dem ERP ein ContentInvoiceInfo-Objektmit allen Information zu dieser Rechnung zurück. Diese Informationen können nun vom ERP verarbeitet werden.

- **Schritt 4: Rechnung als Datei herunterladen:** Ist im Header ("ContentInvoiceHeader") des erhaltenen ContentInvoiceInfo-Objekts das Attribut "documentFiles" gesetzt, können diese als binäre Dateien heruntergeladen und in die ERP-Dateiumgebung gespeichert werden. Dazu ist pro FileListItem entweder das Attribute "fileId" oder "fileURL" gesetzt. Die "fileId" wird als Datei-Identifikation für den FileContent-Request verwendet, um geschützte, nicht für die Öffentlichkeit bestimmte Dateien abzufragen. "fileURL" kann für statische, öffentliche Dateien verwendete werden

(vor allem im Bereich von Produktinformationen sinnvoll und nicht für firmenspezifische Dokumente und Belege).

Datenstrukturen

Die Daten werden im JSON-Format ausgetauscht.
Die JSON-Schemen für die Entwicklung der Schnittstelle und die Validierung der übertragenen Daten befinden sich unter folgendem Link:
https://api.comnorm.com/schemas

Standards

ComNormAPI benutzt die folgendene Standards:

- {LanguageCode} (ISO 639-1, 2-digit, de, en, fr, it, ...) - Liste der Sprachcodes
- {CountryCode} (ISO 3166-1 alpha-2, CH, DE, IT, ...) - Liste der Länder
- {CurrencyCode} (ISO 4217, CHF, EUR, ...) - Liste der Währungscodes
- {UnitCode} (unece.org, MMT=mm, C62=piece, ...) - UNECE-Einheitscodes
- Datum-/Zeitangaben: (1999-02-15, 15:05:27) - ISO 8601
- JSON-Inhalt-Codierung: UTF-8
- Nummerformat: Mit Punkt ohne Tausender-Gruppierungszeichen (1753.30), nicht mehr als 6 Nachkommastellen

ComNormAPI Versionsformat

Das Format für die ComNormAPI-Version ist immer im Format "Versionsnummer"." Unterversion".

Beispiel: "1.3"

Dieses Format gilt sowohl für das Attribut "version" im "backendURL"-Array (unterstützte ComNormAPI-Verisonen und deren API-URL eines Lieferanten) auf dem ServiceItem.

Container

Mit der ComNormAPI-Schnittstelle wurde ein generischer Typ Container erschaffen.
Man unterscheidet in ContainerInfo und ContainerContent.

ContainerInfo

Die ContainerInfo beinhaltet oberflächliche Informationen zum Container. Eine Liste über die ContainerInfoItem kann mit dem ContainerList-Request abgefragt werden. Folgende Informationen stehen auf einem ContainerInfoItem:

```
{
  "containerId": "xyz",
  "type": "Container-Type",
  "timeStamp": "Erstellungszeitpunkt des Container-Objekts",
  "url": "URL zum ProductList-Container, welche über den Browser
aufgerufen werden kann.",
  "ownerId": "Containerzugehörigkeit / Benutzer-ID (in den meisten
```

```
Fällen Email-Adresse des Benutzers)",
  "contentState": "Status des Container-Inhalts
(CONTENT_STATE_AWAITING_CONFIRMATION, CONTENT_STATE_PENDING,
CONTENT_STATE_COMPLETED, CONTENT_STATE_FAILED)"
}
```

ContainerContent

Ein ContainerContent beinhaltet detaillierte Informationen zum Container und übermittelt sowohl Dokumentinhalte sowie auch Produktlisten und kann von ComNorm beliebig erweitert werden.

Ein ContainerContent besteht immer aus einer einem **header** (Header-Informationen zu einem Container), **body** (Eigentlicher Inhalt des Containers) und **footer** (Footer-Informationen zu einem Container):

```
{
  "header":
  {
  },
  "body":
  {
  },
  "footer":
  {
  }
}
```

Beispiel eines Warenkorbs in einem Container:

```
{
  "header":
  { ..
  },
  "body":
  { "productList":
    [ { "productNumber": "5678",
        ...
      },
      { "productNumber": "78962",
        ...
      },
      ...
    ]

  },
  "footer":
  { "comment": "Warenkorb von hans.muster@gmail.com"
  }
}
```

ContainerContent-Typen

Von der ComNormAPI-Schnittstelle definierte Dokumenttypen und Produktlisten:

Typen werden möglicherweise noch umbenannt, da es sich in den JSON-Strukturen um ContentInfos handelt werden die Typen möglicherweise wiefolgt umbenannt:
- content-document-inquiry oder content-inquiry
- content-document-offer oder content-offer
- content-productlist-shoppingcart oder content-shoppingcart
- usw.

Typ	Beschreibung	JSON-Objekt
document-inquiry	Anfrage	ContentInquiryInfo
document-offer	Angebot	ContentOfferInfo
document-order	Bestellung	ContentOrderInfo
document-confirmation	Bestätigung	ContentConfirmationInfo
document-invoice	Rechnung	ContentInvoiceInfo
productstate-responselist	Produktstatus-Antwortliste	ContentProductStateResponseListInfo

Mehrsprachige Felder

Die sprachspezifischen Felder werden in allen vom Lieferanten unterstützten Sprachen als Array auf dem entsprechenden Feld zurückgeliefert.
Die sprachspezifischen Felder sind in den JSON-Schemen definiert und sind von den folgenden Datentypen:

- ShortTextItem für kurze sprachspezifische Felder
- LongTextItem für lange sprachspezifische Felder

Wichtig: Das Attribut "documentFileLanguageCode" auf den Container-Headers bezieht sich auf die Sprache der im Header definierten "fileId".

Beispiel des sprachspezifischen Feldes "serviceDesc" auf dem ServiceItem

```
[
  {
    "serviceId": 1234,
    ...
    "serviceDesc": [
      { "languageCode": "de",
        "text": "Über 100'000 Artikel für den Holzbau"
      },
      { "languageCode": "en",
```

```
          "text": "More than 100'000 products for wood constructions"
        },
        { "languageCode": "fr",
          "text": "100'000 produits"
        }
      ],
      ....
    }
  ]
```

Einheiten

In der ComNormAPI-Schnittstelle werden alle Typen von Einheiten gleich strukturiert.
Jede Einheit enthält einen "UNECE"-Einheitscode für die Identifikation der Einheit. Des weiteren enthält ein Einheitsobjekt einen Faktor ("factor") und Teiler ("divider") um die Einheit auf Ihre Basiseinheit umzurechnen.
Die Datenstruktur einer Einheit sieht somit wiefolgt aus:

```
"Unit"
{
  "code",
  "factor",
  "divider"
}
```

Als Faktor wird immer die Menge an Basiseinheiten verwendet um auf die aktuelle Einheit umzurechnen.

ComNormAPI Einheitstypen

- **BaseUnit:** Basiseinheit: Bildet die kleinstmögliche Einheit ab. Alle weiteren Einheiten können in diese umgerechnet werden.
- **OrderUnit:** Bestelleinheit: Einheit, in der Produkte vom Lieferanten bestellt werden können.
- **PackageUnit:** Verpackungseinheit: Einheit, in der das Produkt verpackt ist
- **PriceUnit:** Legt fest, in für welche Einheit der Preis angegeben wird.

Anwendung der Einheitstypen

Die Einheiten werden immer im Bezug auf die Basiseinheit verwendet. Die defininierte Einheit wird also immer in die Basiseinheit umgerechnet. Damit die entsprechenden Einheiten sternförmig zu der BaseUnit verknüpft werden können, ist es zwingend notwendig, dass der Faktor und Teiler der BaseUnit immer dem Wert 1 entsprechen.
Im folgenden Beispiel ist die Basiseinheit 1 Stück (Einheitscode C62). Die Bestelleinheit wird als Packung (Einheitscode PK) angegeben.
Eine Packung beinhaltet im folgenden Beispiel 100 Mengen der Basiseinheit, also 100 Stück:

```
{
  ...,
  "baseUnit": {
    "code"    : "C62",
    "factor"  : 1,
    "divider" : 1
  },
  "orderInfo": {
    ...,
    "orderUnit": {
      "code"    : "PK",
      "factor"  : 100,
      "divider" : 1
    },
    ...
  },
  ...
}
```

Umrechnungstabelle

Einheits Beispiel	Formel (aktuelle Einheit)	Formel (Basiseinheit)
Bestelleinheit mit Faktor (orderUnit)	Bestelleinheit = Bestelleinheitsfaktor * Basiseinheit	Basiseinheit = Bestelleinheit / Bestelleinheitsfaktor
Preiseinheit mit Faktor und Teiler (priceUnit)	Preiseinheit = (Preiseinheitsfaktor / Teiler) * Basiseinheit	Basiseinheit = Preiseinheit / (Preiseinheitsfaktor / Teiler)

Praxisbeispiel: Schrauben

Im folgenden Beispiel werden Schrauben in folgenden Einheiten bestellt:

- Basiseinheit in Stück (Einheitscode C62): Eine Schraube
- Bestelleinheit als Packung (Einheitscode PK): Eine Packung beinhaltet 100 Schrauben.
- Verpackungseinheit als Packung (Einheitscode PK)g: Eine Verpackungseinheit beinhaltet 100 Schrauben.
- Preiseinheit pro Kilogramm (Einheitscode KGM): Preis wird in Kilogramm angegeben. 1 kg beinhaltet demnach 200 Schrauben.

```
{
  ...,
  "baseUnit": {
    "code"    : "C62",
    "factor"  : 1,
    "divider" : 1
  },
  "orderInfo": {
```

```
    ...,
    "orderUnit": {
      "code"    : "PK",
      "factor"  : 100,
      "divider" : 1
    },
    ...
  },
  "packageUnit":{
    "code"    : "PK",
    "factor"  : 100,
    "divider" : 1
  },
  "priceInfo": {
    ...,
    "priceUnit": {
      "code"    : "KGM",
      "factor"  : 200,
      "divider" : 1
    },
    ...
  },
  ...
}
```

Praxisbeispiel: Holzplatten

Im folgenden Beispiel werden Holzplatten in folgenden Einheiten bestellt:

- Basiseinheit in Quadratmeter (Einheitscode MTK).
- Bestelleinheit als Platte (Einheitscode PG): Eine Platte hat die Masse 2.07 m * 2.8 m = 5.796 m2
- Verpackungseinheit als Paket (Einheitscode PA): Ein Paket beinhaltet 6 Platten -> 6 * 5.769 m2 = 34.776 m2
- Preiseinheit pro Quadratmeter (Einheitscode MTK): Gleich wie Basiseinheit daher Faktor und Teiler = 1

```
{
  ...,
  "baseUnit": {
    "code"    : "MTK",
    "factor"  : 1,
    "divider" : 1
  },
  "orderInfo": {
    ...,
    "orderUnit": {
      "code"    : "PG",
      "factor"  : 5.796,
      "divider" : 1
    },
```

```
      ...
    },
    "packageUnit": {
      "code"    : "PA",
      "factor"  : 34.776,
      "divider" : 1
    },
    "priceInfo": {
      ...,
      "priceUnit": {
        "code"    : "MTK",
        "factor"  : 1,
        "divider" : 1
      },
      ...
    },
    ...
  }
```

Empfohlene UNECE-Einheiten

ComNorm empfiehlt folgende Einheiten der von UNECE definierten Einheiten zu benutzen:

Beschreibung	**Symbol**	**UNECE-Einheitscode**
Dezimeter	dm	DMT
Kilometer	km	KTM
Kubikdezimeter	dm3	DMQ
Kubikmeter	m3	MTQ
Meter	m	MTR
Mikrometer	µm	4H
Millimeter	mm	MMT
Quadratmeter	m2	MTK
Zentimeter	cm	CMT
Ampere	A	AMP
Hertz	Hz	HTZ
Joule	J	JOU
Kilojoule	kJ	KJO
Kilowatt	kW	KWT
Kilowattstunde	kWh	KTH

Beschreibung	Symbol	UNECE-Einheitscode
Ohm	Ω	OHM
Watt	W	WTT
Wattstunde	Wh	WHR
Calenda	cd	CDL
Lumen	lm	LUM
Lux	lx	Lux
Gramm	g	GRM
Hektoliter	hl	HLT
Kilogramm	kg	KGM
Kilogramm pro Kubikmeter	kg/m3	KMQ
Liter	l	LTR
Milliliter	ml	MLT
Tonne	t	TNE
Beutel	-	BG
Block	-	D64
Bogen/Blatt	-	ST
Bündel	-	BE
Container	-	CH
Dose	-	CA
Eimer	-	BJ
Flasche	-	BO
Kanister	-	CA
Kapsel	-	AV
Karton	-	CT
Cartusche	-	CQ
Kübel	-	BJ
Paar	-	PR
Pack	-	PK
Palette	-	LF

Beschreibung	Symbol	UNECE-Einheitscode
Platte	-	PG
Rolle	-	RO
Sack	-	BG
Schachtel	-	BX
Set/Garnitur	-	SET
Spender	-	DI
Stück	-	C62
Trommel	-	DR
Tube	-	TU
Grad Celsius	°C	CEL
Kelvin	K	KEL
Minute	min	MIN
Monat	Mt	MON
Sekunde	s	SEC
Stunde	h	HUR
Tag	d	DAY
Woche	Wo	WEE
Kubikmeter pro Stunde	m3/h	MQH
Newton	N	NEW
Newtonmeter	Nm	NU
Pascal	Pa	PAL

Beschreibung der JSON-Enums

JSON Schema: Enums (Allgemeine Enums)

Link zum Schema: Enums

dispatchCode

Versandart

Enum	Beschreibung
DISPATCH_CODE_NOT_SPECIFIED	Keine Angaben

Enum	Beschreibung
DISPATCH_CODE_POST	Post
DISPATCH_CODE_TRAIN	BAHN
DISPATCH_CODE_SHIP	SCHIFF
DISPATCH_CODE_ROAD_CAMION	Strasse, Camion
DISPATCH_CODE_AIR_FREIGHT	Luftfracht
DISPATCH_CODE_COLLECTIVE_DELIVERY	Sammellieferung
DISPATCH_CODE_PICKUP	Abholung
DISPATCH_CODE_EXPRESS	Express

paymentCode

Zahlungsart

Enum	Beschreibung
PAYMENT_CODE_OTHER	Andere Zahlart
PAYMENT_CODE_INVOICE	RECHNUNG
PAYMENT_CODE_CASH_ON_DELIVERY	Nachnahme
PAYMENT_CODE_DIRECT_DEBIT	Lastschrift
PAYMENT_CODE_CREDIT_CARD	Kreditkarte

shipCode

Einleitungsart der Auslieferung

Enum	Beschreibung
SHIP_CODE_IMMEDIATELY	Sofortige Auslieferung
SHIP_CODE_ON_CALL	Auf Abruf

JSON Schema: DeliveryHeader

Link zum Schema: ContentDeliveryHeader

Enums für Attribut deliverType

Angabe, um welche Lieferungart es sich handelt. Im Falle einer Rücknamhe (DELIVERY_TAKEBACK) Fall weisen die Mengen negative Werte auf.

Enum	Beschreibung

Enum	Beschreibung
DELIVERY_NORMAL	Normale Lieferung
DELIVERY_PARTIAL	Teillieferung
DELIVERY_COLLECTIVE	Sammellieferung
DELIVERY_TAKEBACK	Rücknahme: In diesem Fall weisen die Mengen negative Werte auf.

JSON Schema: InvoiceHeader

Link zum Schema: ContentInvoiceHeader

Enums für Attribut invoiceType

Angabe, um welchen Rechnungstyp es sich handelt. Im Falle einer Gutschrift weisen die Betraege und Mengen negative Werte auf.

Enum	Beschreibung
INVOICE_NORMAL	Normale Rechnung
INVOICE_PARTIAL	Teillrechnung
INVOICE_COLLECTIVE	Sammelrechnung
INVOICE_CREDIT	Rücknahme: In diesem Fall weisen die Beträge und Mengen negative Werte auf.

JSON Schema: FileListItem

Link zum Schema: FileListItem

Enums für Attribut fileArea

Bereich der Datei

Enum	Beschreibung
FILE_CAD_2D	2D-CAD-Zeichnung
FILE_CAD_3D	3D-CAD-Zeichnung
FILE_DOCUMENT	Dokument
FILE_IMAGE	Bilder allgemein
FILE_URL	externe URL
FILE_BIM	BIM Datei
FILE_ARCHIVE	Archiv

Enum	Beschreibung
FILE_PARTLIST	Teile eines Produktes
FILE_UNKNOWN	Breich der Datei nicht spezifiziert

JSON Schema: ProductInfo

Link zum Schema: ProductInfo

JSON Schema: ProductStateResponseItem

Link zum Schema ProductStateResponseItem

Enums für Attribut validateState

Produkt-Validierungsstatus

Enum	Beschreibung
PRODUCT_STATE_OK	Produkt vorhanden
PRODUCT_STATE_DELETED	Produkt nicht mehr vorhanden
PRODUCT_STATE_SUBSTITUTED	Produkt wurde durch anderes Produkt ersetzt
PRODUCT_STATE_UNKNOWN	Produkt unbekannt

JSON Schema: ShipTo

Link zum Schema: ShipTo

Enums für Attribut shipLocationCode

Identifikationscode der Adresse

Enum	Beschreibung
LOCATION_CONSTRUCTIONSITE	Baustelle
LOCATION_PROCESSING_PLACE	Verarbeitungsort
LOCATION_TRANSFER_POINT	Übernahmestelle
LOCATION_INTERMEDIATE_STORE	Zwischenlager

JSON Schema: AllowOrChargeItem

Link zum Schema AllowOrChargeItem

Enums für Attribut allowOrChargeCode

Identifikationscode des Zu- oder Abschlages

Enum	Beschreibung
ALLOW_OR_CHARGE_ADMINISTRATION	Administrationsgebühr
ALLOW_OR_CHARGE_CHARGE	Zuschlag
ALLOW_OR_CHARGE_ALLOW	Abschlag
ALLOW_OR_CHARGE_COD	Nachnahme
ALLOW_OR_CHARGE_CUSTOMS	Zoll
ALLOW_OR_CHARGE_CUSTOMIZATION	Konfektionierung
ALLOW_OR_CHARGE_EXPRESS	Expresslieferung
ALLOW_OR_CHARGE_FREIGHT	Fracht
ALLOW_OR_CHARGE_HANDLING	Bearbeitungsgebühr
ALLOW_OR_CHARGE_INSURANCE	Versicherung
ALLOW_OR_CHARGE_PACKING	Verpackung
ALLOW_OR_CHARGE_DISCOUNT	Rabatt
ALLOW_OR_CHARGE_RECYCLING	Entsorgung
ALLOW_OR_CHARGE_POSTAGE	Porto
ALLOW_OR_CHARGE_TAX	Steuer
ALLOW_OR_CHARGE_VRG	Vorgezogene Recyclinggebühr
""	Gemäss Text

JSON Schema: ProductstateListHeader

Link zum Schema: ContainerInfoItem

Enums für Attribut contentState

Status bzw. Verfügbarkeit eines Containerinhalts

Enum	Beschreibung
CONTENT_STATE_AWAITING_CONFIRMATION	Inhalt muss bestätigt werden
CONTENT_STATE_PENDING	Inhalt ist in Aufbereitung
CONTENT_STATE_COMPLETED	Inhalt steht bereit
CONTENT_STATE_FAILED	Inhalt konnte nicht aufbereitet werden