# FOURTH ENTERPRISES, USA
# cXML Reference Guide
# Version 1.2.021

## Document Revision Record

| Date | Author | Change |
|---|---|---|
| March 31, 2015 | Saradhi Valluri | Initial build |
| May 11, 2022 | Saradhi Valluri | Added Invoice Transactions, updated urls, security, and removing IE compatibility. |

**Purpose:**
This Draft Standard for Trial Use contains the format and establishes the data contents of the purchasing integrations for use within the context of cXML Integrations. The transaction sets can provide for customary, established business and industry practices relative to the invoicing.

The following pages are the guidelines for implementing the cXML document with Fourth. This document addresses the segments that Fourth Adaco is capable of sending & receiving.

# Contents

# 1. Introduction

This specification contains reference information about Fourth's Adaco capabilities and guidelines to implement cXML PunchOut, Purchase Order, and Invoice with the supplier's network.

This document is based on Ariba's cXML solutions guide, published in November 2009. Only information relevant to Fourth Adaco's implementation is included. In addition, specific notations have been included to clarify Fourth Adaco's use of certain elements.

Only information about Invoice related specifications is included here. For any other information, please get in touch with Fourth's Project Manager.

## 1.1 File Format & Data Mapping

### **c**ommerce e**X**tensible **M**arkup **L**anguage

cXML is a general-purpose language for conveying commerce-related information. Organizations use it to communicate business data between diverse applications.

Trading partners use cXML for communicating purchase orders. When buying organizations generate cXML purchase orders, they include data from their Enterprise Resource Planning (ERP) systems and procurement systems. They route these purchase orders to their suppliers through Ariba Supplier Network (Ariba SN). Suppliers then extract data from the purchase orders and use that data within their order-fulfillment systems.

Trading partners' back-end systems must be able to map data either to or from the cXML documents. The default configurations of these external systems do not always have the data or formatting necessary for the cXML protocol. Even if messages are well-formed cXML, the actual data within these elements might vary. This inconsistency affects buying organizations, and suppliers, and can lead to longer integration times.

This document helps implementers provide consistent data between trading partners. It provides best practices for converting back-end-system data to cXML to help prevent variation in the data transferred between trading partners.

# 2. Important Notes

- Any node/element/attribute marked for Optional can be ignored. We are not currently using the segments that are marked as Optional. Mand.
- If Mand. Is Y→ Mandatory, N → Optional, R → Recommended
- We are currently using only those that are marked with 'Y.'
- These characters must not be part of cXML element data &, > and <
- Any segment or element that is marked as Recommended; we will be using them in the future.

## 2.1 Abbreviations

| *Abbreviation* | *Description* |
|---|---|
| Mand | Mandatory |
| Size | Denotes the size of the element's data. The length can be anything from 1 to the mentioned size. If it says N/A, that means you can have any size for that element (from 1 to unlimited characters) |
| Default Value | → The **default value** Fourth's Adaco populates in the attributes. If Fourth's Adaco mentions something in the document, we will use that value for that element/attribute. |
| String | Alphanumeric/ varchar/ varchar/ text/ String |
| Int | Integer value |
| Decimal | Decimal value |
| Money | Money value but without currency symbol (more or less the same as a decimal) |

# 3. Technical Details

We would support only HTTPS urls for cXML integrations for better security and content encryption. To receive the cXML data over the internet with secure HTTP, the supplier must install an SSL/TLS Web Server Certificate from a trusted certificate authority on the supplier's web server. We support TLS 1.2 or more for HTTPS communication.

Our Postback URL (where the supplier submits the cart information) is below.

| | *Region* | *Postback URL* |
|---|---|---|
| Test | | https://intranet.adaco.com/vendor/cXML/PostPunchOutOrder.aspx |
| Live | For NA/US | HTTPS://<br>punchout.p2p.na1.fourth.com/cXML/PostPunchOutOrder.aspx |
| | For EMEA | https://PunchOut.adaco.com/vendor/cXML/PostPunchOutOrder.aspx |

Our Invoice URL (where the supplier submits the invoice information) usually is anyone below two depending on the region. We don't have a test instance for invoice URL.

| | | |
|---|---|---|
| Live only | For NA/US | https://punchout.p2p.na1.fourth.com/cXML/Invoice |
| | For EMEA | https://punchout.adaco.com/cXML/Invoice |

## 3.1 Document Encryption

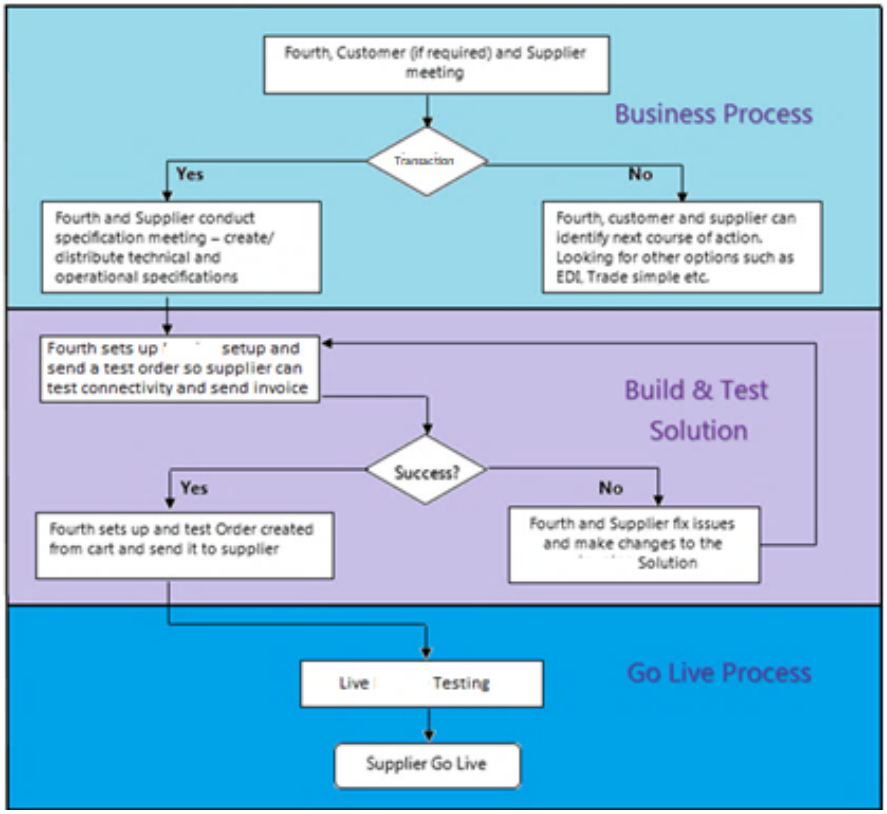
Applications encrypt cXML documents before transmission by sending them through HTTPS connections. HTTPS is a secure form of HTTP (Hypertext Transfer Protocol) supported by most Web servers and Web browsers. It protects network communication by encrypting data so unauthorized parties cannot interpret it.

## 3.2 Encoding and Serialization

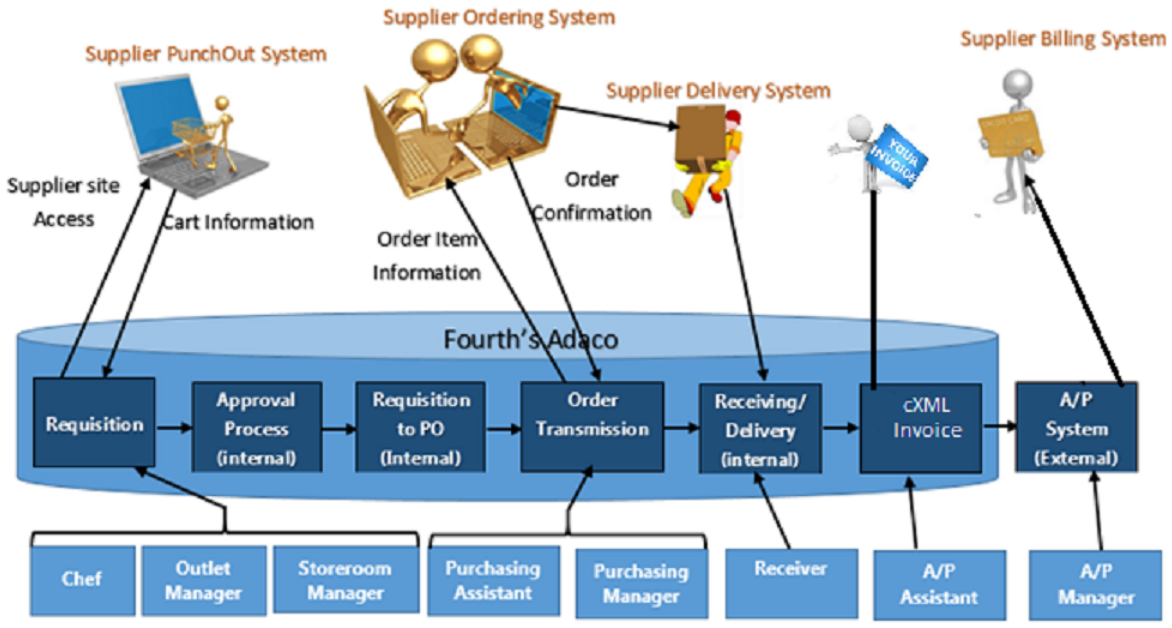
Fourth's Adaco uses URL encoding (UTF8 Encoding) and XML serialization for the requests we send from our application.

Supplier must also support the same type of encoding during response or message documents.

## 4. Go Live Methodology



## 5. cXML Integration Workflow




Currently, Fourth is capable ofsupportingt three cXML documents.

- cXML PunchOut
- cXML Purchase Order
- cXML Invoice

For each of these PunchOut and Purchase Order integrations, we support these session documents.

cXML PunchOut:

- PunchOutSetupRequest (Supplier Site Access)
- PunchOutSetupResponse (Supplier Site Access)
- PunchOutOrderMessage (Cart Information)

cXML Purchase Order:

- OrderRequest (Order Item Information)
- OrderResponse (Order Confirmation Response)

cXML Invoice:

- InvoiceDetailRequest (Invoice Item Information)
- Response (Invoice Confirmation Response)

-

# 6. Data Conventions

## 6.1 Number Format

- Please do not include signs other than numbers, commas (,), and period (.)
- For decimal separator, please use only period (.)
- You can use thousand separators, but only commas (,) for thousand separators.

Correct:
<ItemIn quantity="123.5" lineNumber="1">
<ItemIn quantity="1,123.5" lineNumber="1">

Incorrect:
<ItemIn quantity="123,5" lineNumber="1">
<ItemIn quantity="1.123,5" lineNumber="1">
<Discount>75%</Discount>

## 6.2 Money Format

- Please do not include any signs other than numbers, commas („) and period (.)
- For decimal separator, please use only period (.)
- You can use thousand separators, but only commas (,) for thousand separators.
- Don't include currency symbol with the value
- Specify currency with a three-letter ISO 4217 currency code.

| | |
|---|---|
| Correct | <Money currency="USD">1,234.56</Money> |
| Incorrect | <Money currency="USD">2,56</Money><br><Money currency="USD">1.234,56</Money><br><Money currency="USD">$1,234.56</Money> |

## 6.3 Unit of Measurement

- Please provide the unit of measurement suppliers use in this PunchOut
- Please use ANSI units.

## 6.4 Date & Time Format

- We use ISO 8601 and its subset formats for date and time values.
- Format will be YYYY-MM-DDThh:mm:ss (2015-08-07T08:36:34)

## 7. cXML DTDs

Because cXML is an XML language, it is thoroughly defined by a set of Document Type Definitions (DTDs). These DTDs are text files that describe the precise syntax and order of cXML elements. DTDs enable applications to validate the cXML they read or write.

Each cXML document's header contains the DTD URL that defines the document. cXML applications can retrieve the DTD and use it to validate the document.

For the most robust transaction handling, validate all cXML documents received. If you detect errors, issue the appropriate error code so the sender can retransmit. cXML applications are not required to validate cXML documents received, although it is recommended. However, all cXML documents must be valid and must refer to the cXML DTDs described in the following section.

DTDs for all versions of cXML are available on cXML.org. The various kinds of cXML documents are defined in multiple DTDs to reduce DTD size, which enables faster validation in some parsers.

cXML applications use these DTDs to validate all incoming and outgoing documents.

| Document | DTD |
|---|---|
| Basic cXML documents | http://xml.cXML.org/schemas/cXML/1.2.021/cXML.dtd |
| Confirmation and Ship Notice | http://xml.cXML.org/schemas/cXML/1.2.021/Fulfill.dtd |
| Invoice | http://xml.cXML.org/schemas/cXML/1.2.021/InvoiceDetail.dtd |
| Type Definition | http://xml.cXML.org/schemas/cXML/1.2.021/Catalog.dtd |
| Payment Remittance | http://xml.cXML.org/schemas/cXML/1.2.021/PaymentRemittance.dtd |
| Request for Quotations | http://xml.cXML.org/schemas/cXML/1.2.021/Quote.dtd |

## 8. cXML Basics

There are two communication models for cXML Transactions. Request-Response and one-way. Fourth's Adaco uses both models as sometimes using one model will not be enough. For invoices, the roles of Fourth's Adaco & Supplier system will be reversed.

Two-Way Model (PunchOut Setup request & Response OR Order Request & Response).


Two-Way Model (Invoice Detail request & Response).


One-way model (PunchOut Order Message)

## 8.1. CXML Session Document Types

Fourth's Adaco uses below session document types for this integration. For ease of use and understanding the below document types are divided into three protocols (Request, Response and Message)

From Buyer to Supplier (Outgoing)

- PunchOutSetupRequest (Request)
- OrderRequest (Request)
- InvoiceDetailResponse (Response)

From Supplier to Buyer (Incoming)

- PunchOutSetupResponse (Response)
- PunchOutOrderMessage (Message)
- OrderResponse (Response)
- InvoiceDetailRequest (Request)

## 8.2. Requests

The client sends requests for operations. Only one Request element is allowed for each cXML document. The Request element can contain any type of XML data. Currently, Fourth's Adaco supports the following types of requests.

- PunchOutSetupRequest
- OrderRequest
- InvoiceDetailRequest

Structure of Request cXML will be as follows:

```
<cXML>
    <Header>
        Header information
    </Header>
    <Request>
        Request information
    </Request>
</cXML>
```

## 8.3 Responses

The server sends responses to inform clients of the results of operations. Fourth's Adaco supports these types of responses.

- PunchOutSetupResponse
- OrderResponse
- InvoiceDetailResponse

The structure of Response cXML will be as follows:

```
<cXML>
    <Response>
        Response information
    </Response>
</cXML>
```

We expect the status element to be sent to us in the response protocol. If anything, other than status code 200 ,we will treat that as a failure. Because cXML is layered above HTTP, the transport handles many HTTP errors. If the request is not successful, please send the status code of that erroneous request.

```
<Status xml:lang="en-US" code="200" text="OK"> </Status>
```

### 8.3.1 Response Codes

The below table provides various status codes for the request in Response. You may have more status codes than below. But these are what you normally see from multiple implementations. For Fourth's Adaco anything other than 200 status code is a failure.

| *Status* | *Text* | *Explanation* |
|---|---|---|
| 200 | OK | The server could execute the request or deliver it to the final recipient. The returned Response might contain application warnings or errors: the cXML Request itself generated no errors or warnings. However, this status does not reflect any errors or warnings that might be generated afterward by the application itself. You will receive no further status updates unless an error occurs during later processing. |
| 400 | Bad Request | The request unacceptable to the server, although it parsed correctly. |
| 401 | Unauthorized | The credentials provided in the Request (the Sender element) were not recognized by the server. |
| 402 | Payment Required | This Request must include a complete Payment element. |
| 403 | Forbidden | The user has insufficient privileges to execute this Request. |

| *Status* | *Text* | *Explanation* |
|---|---|---|
| 406 | Not Acceptable | Request was unacceptable to the server, likely due to a parsing failure. |
| 409 | Conflict | The server's current state or its internal data prevented the (update) operation request. An identical Request is unlikely to succeed in the future, but only after another operation has been executed, if at all. |
| 412 | Precondition Failed | A precondition of the Request (for example, a PunchOut session appropriate for a PunchOutSetupRequest edit) was not met. This status normally implies that the client ignored some portion of a previous transmission from a server (e.g., the operationAllowed attribute of a PunchOutOrderMessageHeader). |
| 500 | Internal Server Error | The server was unable to complete the Request. |
| 550 | Unable to reach the cXML server | Unable to reach the next cXML server to complete a transaction requiring upstream connections. An intermediate hub can return this code when a supplier site is unreachable. If upstream connections are complete, intermediate hubs should return errors directly to the client. |
| 551 | Unable to forward a request | Unable to forward request because of supplier misconfiguration. For example, an intermediate hub failed to authenticate itself to a supplier. Clients cannot rectify this error, but this error might be resolved before the client retries. |
| 560 | Temporary server error | For example, a server might be down for maintenance. The client should retry later. |

## 8.4 Message

This element carries all the body-level information in a cXML message. This would be used in messages that are logical responses to request messages:

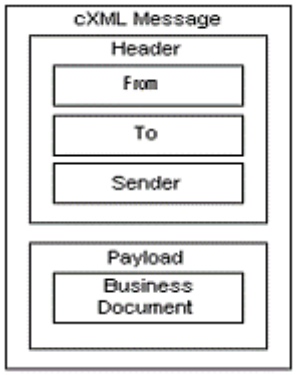
- PunchOutOrderMessage

Structure of Message cXML will be as follows:

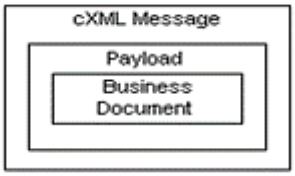
```
<cXML>
    <Header>
        Header information here…
    </Header>
    <Message>
        Message information here…
    </Message>
</cXML>
```

# 9. cXML Document

A simple cXML message includes the following data structures. This is typically used for Request and Message protocols. Payload represents Request or Response or Message:




(Request & Message)




(Response)

Basic cXML segments for Request & Message

1) cXML Message Document
2) cXML Message Envelope
3) cXML Message Header
4) Business Document, which includes item or business information. This section represents the detail section of the document. (This will be explained at the Transaction level.)

Basic cXML segments for Response

1) cXML Message Document
2) cXML Message Envelope
3) Business Document, which includes status and business information. This section represents the detail section of the document. (This will be explained at the Transaction level.)

## 9.1. cXML Declaration

A cXML element is the body of a cXML document. A document might begin as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.020/cXML.dtd">
```

The first characters in cXML documents must be <? Documents must not start with white space or tabs.

The second line in cXML documents must contain the DOCTYPE document type declaration. This is the only external entity that can appear in the cXML documents. This line refers to the cXML DTD.

## 9.2 cXML Envelope

The cXML element is the root of cXML documents and contains all other elements. The cXML element is present in every cXML transaction.

```
- <cXML xml:lang="en-US" timestamp="2015-05-05T04:26:23" payloadID="06516fc06f854823b9f18078f401ad4f" version="">
```

Version → It was depreciated in cXML 1.2.007. We no longer pass this information

Xml:lang → Fourth's Adaco always sends "en-US"

payloadID → A unique number Fourth's Adaco creates and sends in the message

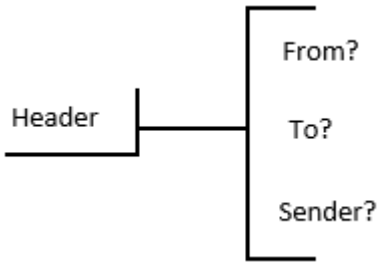
timestamp → The date and time the message was sent in ISO 8601 format.

## 9.3 cXML Header

From cXML.dtd → <!ELEMENT Header (From, To, Sender?)>

cXML credential elements in the header of each cXML document identify the sender and receiver organizations.

Element



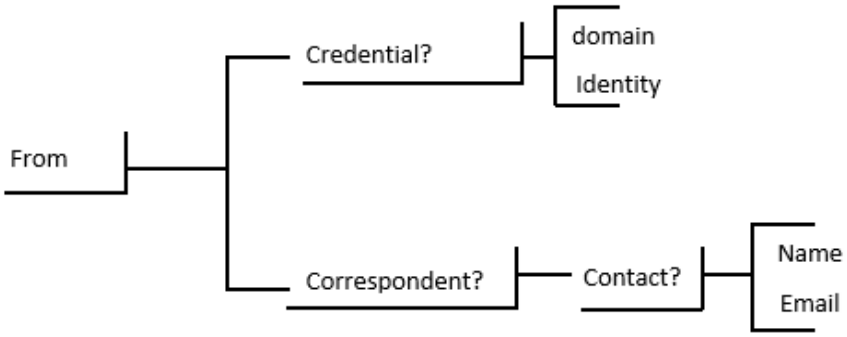

Example:

```
- <Header>
  - <From>
    - <Credential domain="NetworkID">
        <Identity>[redacted]</Identity>
      </Credential>
    - <Correspondent>
      - <Contact>
          <Name xml:lang="en">[redacted]</Name>
          <Email>[redacted]</Email>
        </Contact>
      </Correspondent>
    </From>
  - <To>
    - <Credential domain="DUNS">
        <Identity>[redacted]</Identity>
      </Credential>
    </To>
  - <Sender>
    - <Credential domain="NetworkID">
        <Identity>[redacted]</Identity>
        <SharedSecret>[redacted]</SharedSecret>
      </Credential>
      <UserAgent>ADACO</UserAgent>
    </Sender>
  </Header>
```

### 9.3.1 From

This element identifies the originator organization of the cXML request

In cXML.dtd → <!ELEMENT From (Credential+, Correspondent?)>



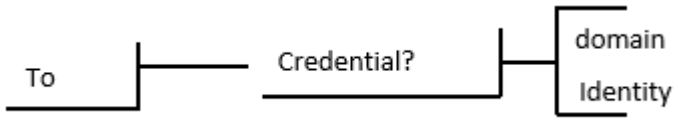

This credential typically denotes the originator/sender/buyer's organization information, either network id or Duns number or mutually defined value for the Outgoing request (from Buyer to Supplier) and supplier's organization information, either network id or Duns number or mutually defined value for the incoming message(from supplier to Buyer).

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| Domain | String | 9 | Y | | NetworkID or DUNS |
| Identity | String | 50 | Y | NetworkID value or DUNS number or mutually defined value | |
| Name | String | 60 | Y | Name of the buyer/user who initiates the session/request | |
| Email | String | 60 | R | Email of the user who initiates the session | |

### 9.3.2 To

In cXML.dtd → <!ELEMENT To (Credential+?)>

This credential typically denotes destination/receiver/supplier organization information, either network id or Duns number or mutually defined value for the Outgoing request (from Buyer to Supplier) and destination/buyer organization information, either network id or Duns number or mutually defined value for the incoming message(from supplier to Buyer).



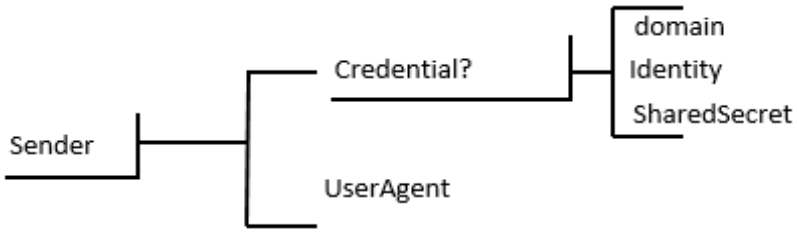
| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| Domain | String | 9 | Y | | NetworkID or DUNS |
| Identity | String | 25 | Y | NetworkID value or DUNS number or mutually defined value | |

### 9.3.3 Sender

In cXML.dtd → <!ELEMENT Sender (Credential+, UserAgent)>

This credential typically denotes relaying entity/network hub information, either network id or Duns number or mutually defined value for the Outgoing request (from Buyer to Supplier) and supplier's organization information, either network id or Duns number or mutually defined value for incoming messages (from supplier to Buyer). This element allows the receiving party to identify and authenticate the party that opened the HTTP/HTTPS connection.

***Note:*** *Fourth's Adaco didn't have Ariba SN information or DUNS number. We can use whatever mutually agreeable values. UserAgent will be "ADACO" for Outgoing messages.*




| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| Domain | String | 9 | Y | | NetworkID or DUNS |
| Identity | String | 50 | Y | NetworkID value or DUNS number or mutually defined value | |
| SharedSecret | String | 50 | R | The shared secret element is used when the Sender has a password that the requester recognizes. | |
| UserAgent | String | N/A | Y | This element identifies who is making the request. This typically denotes the procurement software name. In the case of F,ourth it will be "ADACO" for outgoing me,ssages and for income,ng it would be Supplier name or their product name. | ADACO |

### 9.3.4 Supported ID Domains

Fourth's Adaco supports two types of domains

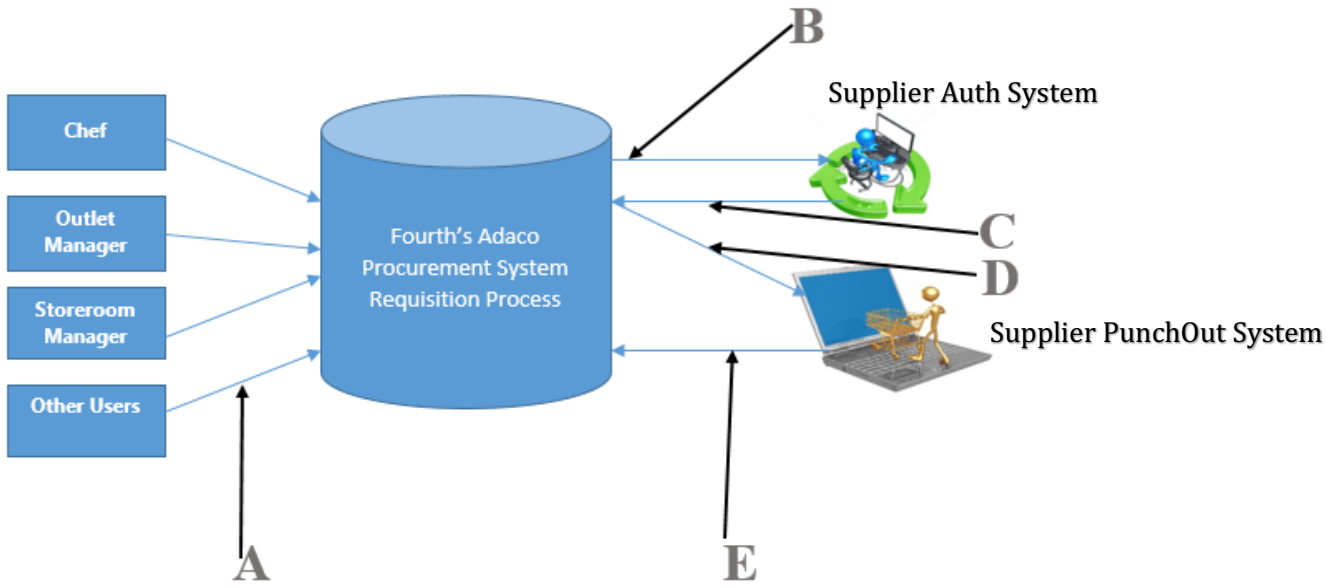
1) NetworkID – A unique alphanumeric value defined by the Supplier.
2) DUNS – A unique number assigned to organizations by Dun & Bradstreet. To request a Dun & Bradstreet D-U-N-S number or to see if your organization already has one, go to www.dnb.com

***Note**: Wherever we see a Name element we always pass xml:lang attribute with the value "en".*

# 10. PunchOut Transaction

## 10.1 Purpose

PunchOut enables users of procurement applications to access supplier contracts for products or services that reside on the supplier's website. It eliminates the need for the suppliers to send whole catalogs to buying organizations. It also enables the user to have an online shopping experience. During this process whatever items, the user added to their cart/basket, those will not be shipped until Purchase Order Transaction happens.




| | |
|---|---|
| A | It represents the user's request for PunchOut through Fourth's Adaco system. |
| B | Fourth's Adaco requests for supplier site access bypassing the credential and storefront URL (PunchOutSetupRequest) |
| C | Supplier Auth system validates the request, credentials, and sends us the shopping URL (PunchOuSetupResponse) |
| D | Using that URL, Fourth's system opens the supplier PunchOut site in Adaco using Web browser control (A, B, C and D happens in one go. User's will not be notified the shopping URL in Fourth's Adaco system). Once user initiates the PunchOut session we open up PunchOut site but behind the scenes B, C and D happens. |
| E | After user makes the necessary purchasing in supplier PunchOut site user will hit checkout. Then Supplier will send the cart information back to Fourth's Adaco Postback URL and closes the session. (PunchOutOrderMessage) |

## 10.2 PunchOutSetupRequest Document

This document initiates the buyer's request for online shopping with the supplier website From Fourth's Adaco procurement system. Fourth's Adaco generates a cXML PunchOutSetupRequest document and sends it to the supplier network system which forwards it to the PunchOut site. This document authenticates the buyer for the supplier.

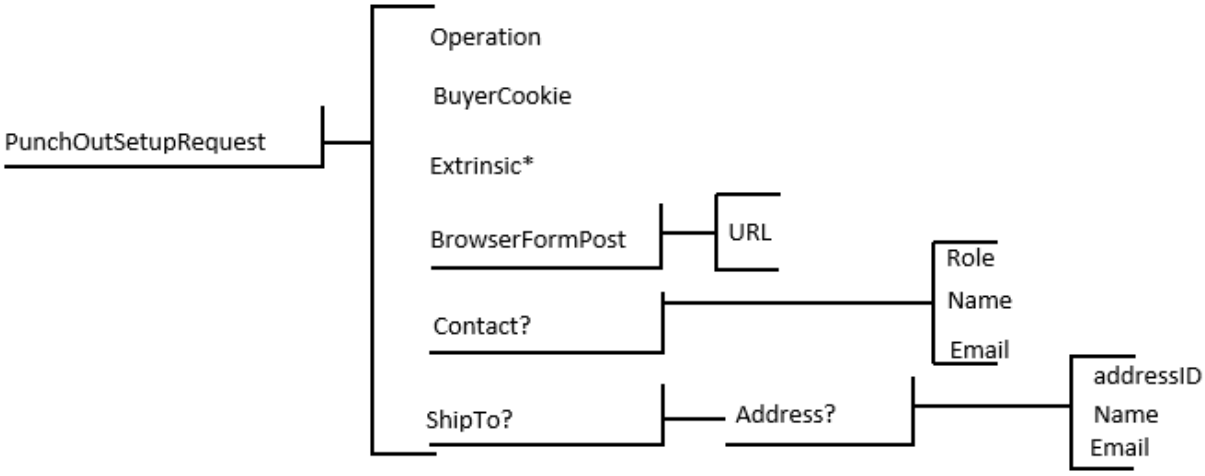
This document contains Declaration, Envelope and Header and PunchOutSetupRequest elements. For Header element please refer the cXML Header section. For Declaration and Envelope elements please refer this section.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.020/cXML.dtd">
- <cXML xml:lang="en-US" timestamp="2015-04-20T04:21:36" payloadID="71c583e0ce8f46a5818d75a38dc66e9c" version="">
  - <Header>
    - <From>
      - <Credential domain="NetworkID">
          <Identity>[redacted]</Identity>
        </Credential>
      - <Correspondent>
        - <Contact>
            <Name xml:lang="en">[redacted]</Name>
            <Email>[redacted]</Email>
          </Contact>
        </Correspondent>
      </From>
    - <To>
      - <Credential domain="DUNS">
          <Identity>[redacted]</Identity>
        </Credential>
      </To>
    - <Sender>
      - <Credential domain="NetworkID">
          <Identity>[redacted]</Identity>
          <SharedSecret>[redacted]</SharedSecret>
        </Credential>
        <UserAgent>ADACO</UserAgent>
      </Sender>
    </Header>
  - <Request deploymentMode="production">
    - <PunchOutSetupRequest operation="create">
        <BuyerCookie>05732aa032584c8589512c27520a2001</BuyerCookie>
        <Extrinsic name="CostCenter">2014</Extrinsic>
        <Extrinsic name="UniqueName">Adaco</Extrinsic>
        <Extrinsic name="User">Adaco[redacted]</Extrinsic>
        <Extrinsic name="UserType">ADC</Extrinsic>
        <Extrinsic name="UserFullName">Adaco2014</Extrinsic>
      - <BrowserFormPost>
          <URL>http://intranet.adaco.com/vendor/cXML/PostPunchOutOrder.aspx</URL>
        </BrowserFormPost>
      - <Contact role="endUser">
          <Name xml:lang="en">[redacted]</Name>
          <Email>[redacted]</Email>
        </Contact>
      - <ShipTo>
        - <Address addressID="1">
            <Name xml:lang="en">[redacted]</Name>
            <Email name="[redacted]">[redacted]</Email>
          </Address>
        </ShipTo>
      </PunchOutSetupRequest>
    </Request>
</cXML>
```

### 10.2.1 PunchOutSetupRequest Element

A PunchOutSetupRequest element is contained within the Request element. The following example shows the element declaration of PunchOutSetupRequest from cXML.dtd:

```
<!ELEMENT PunchOutSetupRequest (
    BuyerCookie,
    Extrinsic*,
    BrowserFormPost?,
    Contact*,
    SupplierSetup?,
    ShipTo?,
    SelectedItem?,
    ItemOut*)>
```




***Note**: For any reason if Supplier don't support EDIT cart functionality, please inform our integration manager.*

| *Element* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| deploymentMode | String | 10 | Y | | production or test |
| PunchOutSetupRequest > operation | String | 6 | Y | Whether the session for create or edit | create or edit |
| BuyerCookie | String | N/A | Y | This element transmits information that is opaque to the supplier Punchout website, but it must be returned to the originator for all subsequent PunchOut operations. This element allows the Fourth's Adaco system to match multiple outstanding PunchOut requests. BuyerCookie is unique per PunchOut session. (Please see 10.2.2 for more information) | |
| BrowserFormPost | String | N/A | Y | This element is the destination for the data in the PunchOutOrderMessage. It contains an URL element where supplier will post the cart information of the user once he/she check's out. | Please refer technical detail section for the URL. |
| Contact > role | String | 7 | Y | Provides the role of originator (buyer/user) | endUser |
| Contact > Name<br>Or<br>ShipTo > Address > Name | String | 60 | Y | Name of the buyer/user who initiates the session/request | |
| Contact > Email<br>Or ShipTo > Address > Email | String | 60 | N | Email of the user who initiates the session | |
| addressID | Int | 4 | Y | Originator Organization's Identity. We define this at site level and unique to each site. Each site can contain multiple hotels/properties. If 2 hotels from 2 different sites want to use this integration, this addressID can be repeated. | |
| xml:lang (Contact/ShipTo) | String | 2 | Y | | en |
| Name | String | 50 | Y | Originator's Organization Name | |

### 10.2.2 BuyerCookie Element

This element is used by procurement applications to associate a given PunchOutOrderMessage with its originating PunchOutSetupRequest. PunchOut sites must return this element whenever it appears. The PunchOutOrderMessage document must contain the same BuyerCookie that was used in the PunchOutSetupRequest document for this PunchOut session.

***Note**: Do not use the BuyerCookie to track PunchOut sessions, because it changes for every session, from create to edit.*

Example:

<BuyerCookie>1TF5JRP11S3W9</BuyerCookie>

### 10.2.3 Extrinsic Element

This optional element contains any additional data that the supplier requests to pass with the request document. The cXML specification does not define the content of Extrinsic elements—it is something that each requestor and remote website must agree on and implement.

Extrinsic elements are intended to provide additional machine-readable information. They extend the cXML protocol to support features not required by all implementations. In the following context, the new data further describes the user initiating the PunchOut request.

```
<Extrinsic name="CostCenter">2014</Extrinsic>
<Extrinsic name="UniqueName">Adaco</Extrinsic>
```

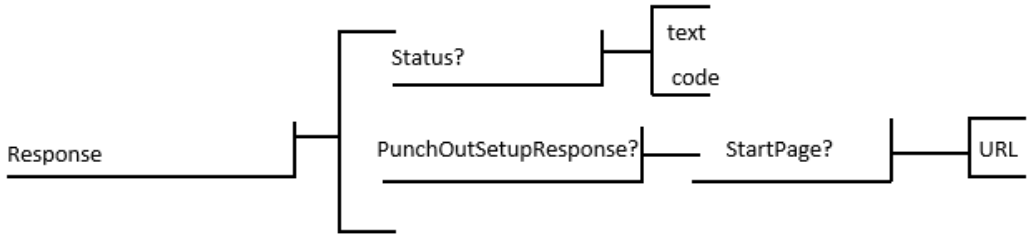
Please refer to the Extrinsic Tag section for more details.

## 10.3 PunchOutResponse Document

After the supplier received the PunchOutSetupRequest, they will response with a PunchOutSetupResponse. This document indicates whether the PunchOutRequest is successful. It also provides the Fourth's Adaco application with a redirect URL to the supplier's website start page.

This document contains Declaration, Envelope and Response elements. For Declaration and Envelope elements please refer this section

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.005/cXML.dtd">
<cXML xml:lang="en-US" timestamp="5/18/2015 2:33:08 AM" payloadID="5/18/2015 2:33:08 AM [redacted] version="1.2.005">
 - <Response>
      <Status text="Authenticated" code="200"/>
    - <PunchOutSetupResponse>
       - <StartPage>
            <URL>[redacted]</URL>
         </StartPage>
      </PunchOutSetupResponse>
   </Response>
</cXML>
```




### 10.3.1 Status Element

Please refer this.

```
<!ELEMENT Status (#PCDATA)>
<!ATTLIST Status
    code      %uint;         #REQUIRED
    text      %string;       #REQUIRED
    xml:lang %xmlLangCode; #IMPLIED
>
```

### 10.3.2 StartPage Element

This element contains a URL element that specifies the URL to pass to the browser to initiate the PunchOut browsing session requested in the PunchOutSetupRequest. This URL must contain enough state information to bind to a session context on the supplier website, such as the requestor identity and the appropriate BuyerCookie element.

At this point, the user who initiated the PunchOutSetupRequest browses the supplier website and selects items to be transferred back to the Fourth's Adaco application through a PunchOutOrderMessage.

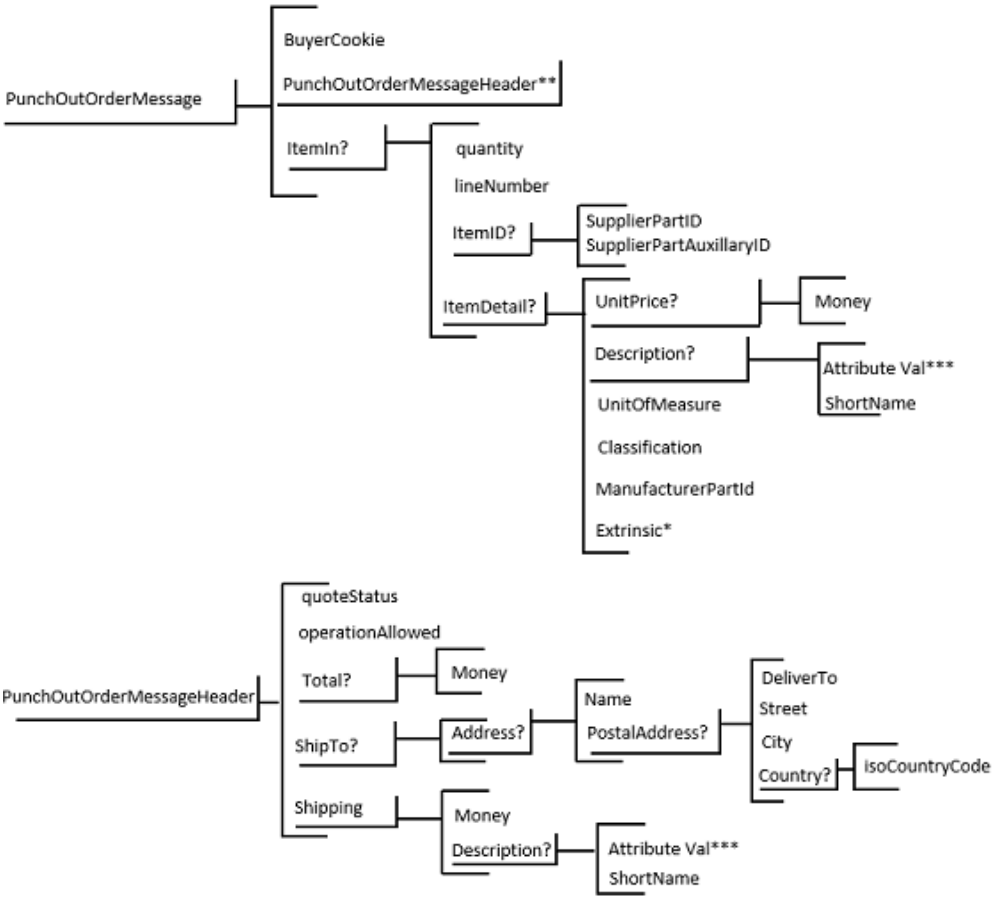
## 10.4 PunchOutOrderMessage Document

The following document is hidden inside a cXML-urlencoded form field like below. Supplier must encode the PunchOutOrderMessage document using URL encoding.

```
<input type="hidden" name="cxml-urlencoded" value=
"<!DOCTYPE cXML SYSTEM &#34;http://xml.cxml.org/schemas/cXML/1.2.014/cXML.dtd&#34;>
<cXML payloadID=&#34;958074737352&amp;www.workchairs.com&#34;
 timestamp=&#34;2004-06-14T12:59:09-07:00&#34;>
   <Header>
      <From>
         <Credential domain=&#34;DUNS&#34;>
            <Identity>12345678</Identity>
         </Credential>
      </From>
```

This element sends the contents of the remote shopping basket/cart to the originator of a PunchOutSetupRequest. It can contain much more data than the other messages because it must be able to fully express the contents of any conceivable shopping basket on the supplier's PunchOut website. This message does not strictly follow the Request/Response model.

The supplier's PunchOut website generates a PunchOutOrderMessage when the user checks out. This message communicates the contents of the remote shopping cart/basket to the Fourth's Adaco system. Supplier needs to submit this content to the "BrowserFormPost" URL in the PunchOutSetupRequest.



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cXML.org/schemas/cXML/1.2.011/cXML.dtd" PUBLIC "cXMLId">
- <cXML xml:lang="EN" version="1.2.009.SYY.01.00" timestamp="Fri Apr 10 20:13:21 UTC 2015" payloadID="2958620_00">
 - <Header>
   - <From>
     - <Credential domain="NetworkID">
         <Identity>[redacted]</Identity>
       </Credential>
     - <Correspondent>
       - <Contact>
           <Name xml:lang="en">[redacted]</Name>
           <Email>[redacted]</Email>
         </Contact>
       </Correspondent>
     </From>
   - <To>
     - <Credential domain="DUNS">
         <Identity>[redacted]</Identity>
       </Credential>
     </To>
   - <Sender>
     - <Credential domain="NetworkID">
         <Identity>[redacted]</Identity>
         <SharedSecret>[redacted]</SharedSecret>
       </Credential>
       <UserAgent>ADACO</UserAgent>
     </Sender>
  </Header>
  <Message deploymentMode="production">
   - <PunchOutOrderMessage>
       <BuyerCookie>cc7f942ba9734ef987b540e0409fb5c3</BuyerCookie>
     - <PunchOutOrderMessageHeader quoteStatus="final" operationAllowed="create">
       - <Total>
           <Money currency="USD">871.95</Money>
         </Total>
       - <ShipTo>
         - <Address addressID="056">
             <Name xml:lang="EN">Boston</Name>
           - <PostalAddress>
               <DeliverTo>071985</DeliverTo>
               <Street>notUsed</Street>
               <City>notUsed</City>
               <Country isoCountryCode="US"/>
             </PostalAddress>
           </Address>
         </ShipTo>
       - <Shipping>
           <Money currency="USD">0.00</Money>
         - <Description xml:lang="EN">
             <ShortName/>
           </Description>
         </Shipping>
       </PunchOutOrderMessageHeader>
     - <ItemIn quantity="158.52" lineNumber="1">
       - <ItemID>
           <SupplierPartID>0107480</SupplierPartID>
           <SupplierPartAuxiliaryID>!!key!!opcoNumber!!value!!056!!key!!customerNumber!!value!!071985!!key!!order!
             key!!UserFullName!!value!!Adaco2014!!key!!ItemPO!!value!!0107480,No</SupplierPartAuxiliaryID>
         </ItemID>
       - <ItemDetail>
         - <UnitPrice>
             <Money currency="USD">4.75</Money>
           </UnitPrice>
         - <Description xml:lang="EN">
             Beef Chuck Flap Meat
             <ShortName>Beef Chuck Flap Meat </ShortName>
           </Description>
           <UnitOfMeasure>LB</UnitOfMeasure>
           <Classification domain="UNSPSC">50000000</Classification>
           <ManufacturerPartID>11285</ManufacturerPartID>
           <Extrinsic name="splitIndicator">No</Extrinsic>
           </ItemDetail>
         </ItemIn>
       + <ItemIn quantity="2" lineNumber="2">
       + <ItemIn quantity="3" lineNumber="3">
       </PunchOutOrderMessage>
     </Message>
  </cXML>
```

### 10.4.1 PunchOutOrderMessage Element

```
<!ELEMENT PunchOutOrderMessage (BuyerCookie, PunchOutOrderMessageHeader,
                                ItemIn*)>

    text     %string;      #REQUIRED
    xml:lang %xmlLangCode; #IMPLIED
>
<!ELEMENT PunchOutOrderMessageHeader (SourcingStatus?, Total, ShipTo?, Shipping?, Tax?, SupplierOrderInfo?)>
<!ATTLIST PunchOutOrderMessageHeader
    operationAllowed  (create | inspect | edit)  #REQUIRED
    quoteStatus (pending|final) "final"
>
<!ELEMENT ItemIn (ItemID, Path?, ItemDetail, SupplierID?, ShipTo?, Shipping?, Tax?, SpendDetail?)>
<!ATTLIST ItemIn
    quantity   %r8;      #REQUIRED
    lineNumber %uint;    #IMPLIED
>
```

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| deploymentMode | String | 10 | Y | | production or test |
| BuyerCookie | String | N/A | Y | The element from the Request. It should match what is sent in the PunchOutSetupRequest. | |
| quoteStatus | String | N/A | Y | We accept what is in the Default Value column | final |
| operationAllowed | String | N/A | Y | We accept what is in the Default Value column | create |
| PunchOutOrderMessage Header > Total | Money | | Y | The total amount of the basket/cart (sum of Quantity * Price without shipping and tax information). | |
| PunchOutOrderMessage Header > ShipTo & Shipping | Element | N/A | N | We ignore what is sent in those elements. | |
| ItemIn > quantity | Decimal | | Y | Quantity of the item that user added to the cart | |
| ItemIn > lineNumber | Int | | Y | Line number coming from the cart/basket | |
| ItemIn > ItemID > SupplierPartID | String | 200 | Y | Supplier's Item/Product # | |

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| ItemIn > ItemDetail > UnitPrice | Money | | Y | Unit Price for each individual UnitOfMeasure. If the Quantity is 5 and UnitOfMeasure is EA, then UnitPrice should be the price buyer should pay for 1 EA not for total 5 EA. | |
| ItemIn > ItemDetail > UnitOfMeasure | String | 10 | Y | Unit with which item can be bought from the supplier. Please inform Fourth's Adaco about possible units you might use in PunchOut site. If you add any additional units in future, please contact our integration manager/contact. | |
| ItemIn > ItemDetail > Description | String | 2000 | Y | Item description | |
| ItemIn > ItemDetail > Classification | Element | | R | We currently ignore this element, but we intend to use this in future | |
| ItemIn > ItemDetail > ManufacturerPartID | Element | | R | We currently ignore this element, but we intend to use this in future | |
| ItemIn > ItemDetail > ManufacturerName | Element | | R | We currently ignore this element, but we intend to use this in future | |
| ItemIn > ItemDetail > LeadTime | Element | | R | We currently ignore this element, but we intend to use this in future | |

### 10.4.2 Extrinsic Elements

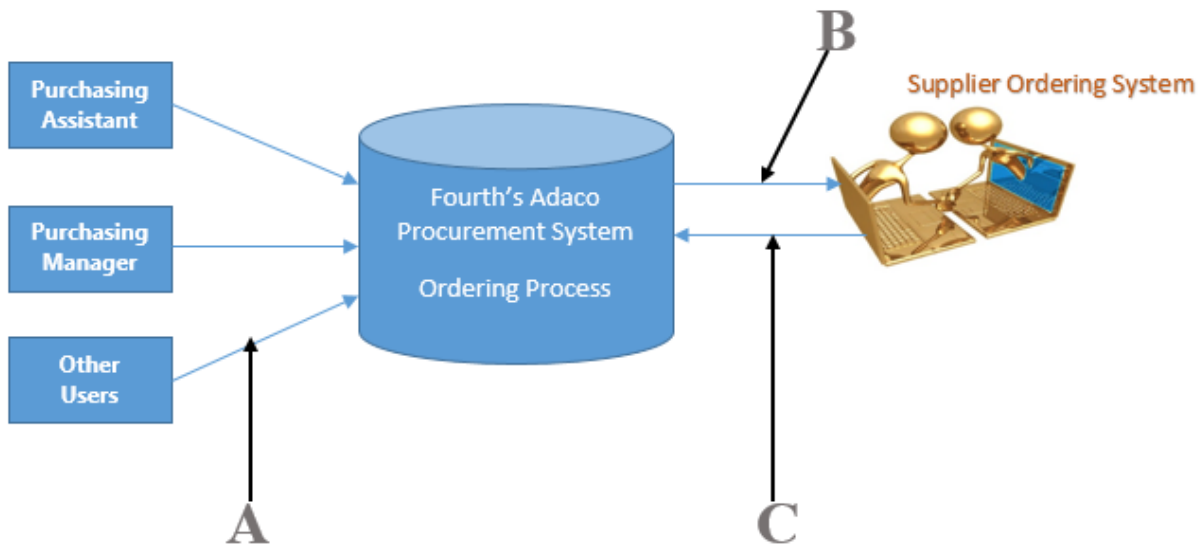
We currently support multiple extrinsic tag values to show certain information to the users such as URL of the image that is customized by user in PunchOut site (some suppliers support customized products such as Uniforms, Business cards and provides the image in an URL). Catchweight item information etc. If you do support Catchweight item information, please let us know the tag name.

Catchweight Extrinsic example:

```
<Extrinsic name="splitIndicator">No</Extrinsic>
<Extrinsic name="itemExtendedPrice">752.97</Extrinsic>
<Extrinsic name="orderDetailLineNumber">1</Extrinsic>
<Extrinsic name="catchWeightFlag">Yes</Extrinsic>
<Extrinsic name="orderedUnitOfMeasure">LB</Extrinsic>
<Extrinsic name="unitPriceInOrderedUnitOfMeasure">188.242</Extrinsic>
<Extrinsic name="averageWeightPerCase">39.63</Extrinsic>
<Extrinsic name="casePack">24</Extrinsic>
<Extrinsic name="caseSize">1.5#</Extrinsic>
<Extrinsic name="itemsPerCase">24</Extrinsic>
<Extrinsic name="specialOrderItem"/>
<Extrinsic name="marketPriceFlag"/>
```

If supplier provides any additional information in extrinsic tags which are required by buyer, please let us know. We will incorporate them into our business (it might require some additional development from Fourth's Adaco).

## 11. Purchase Order Transaction

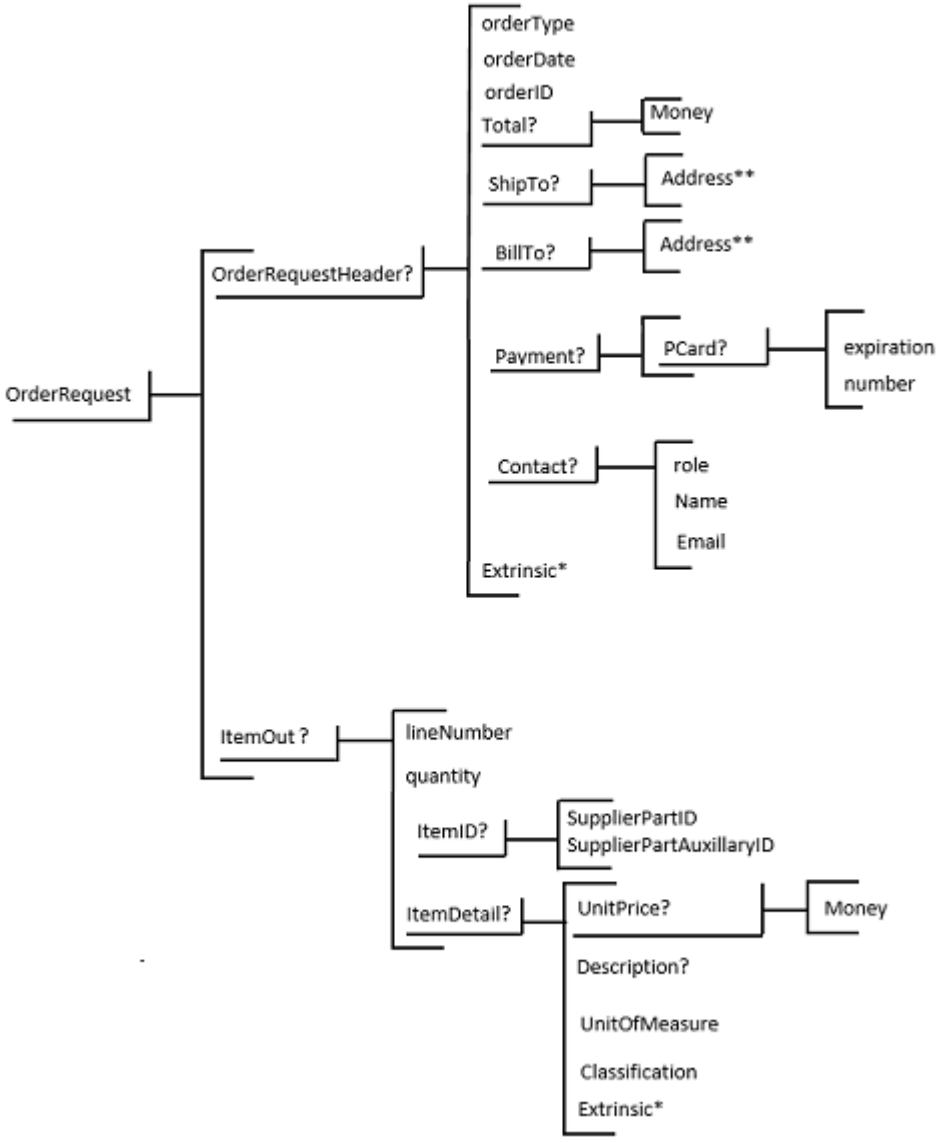


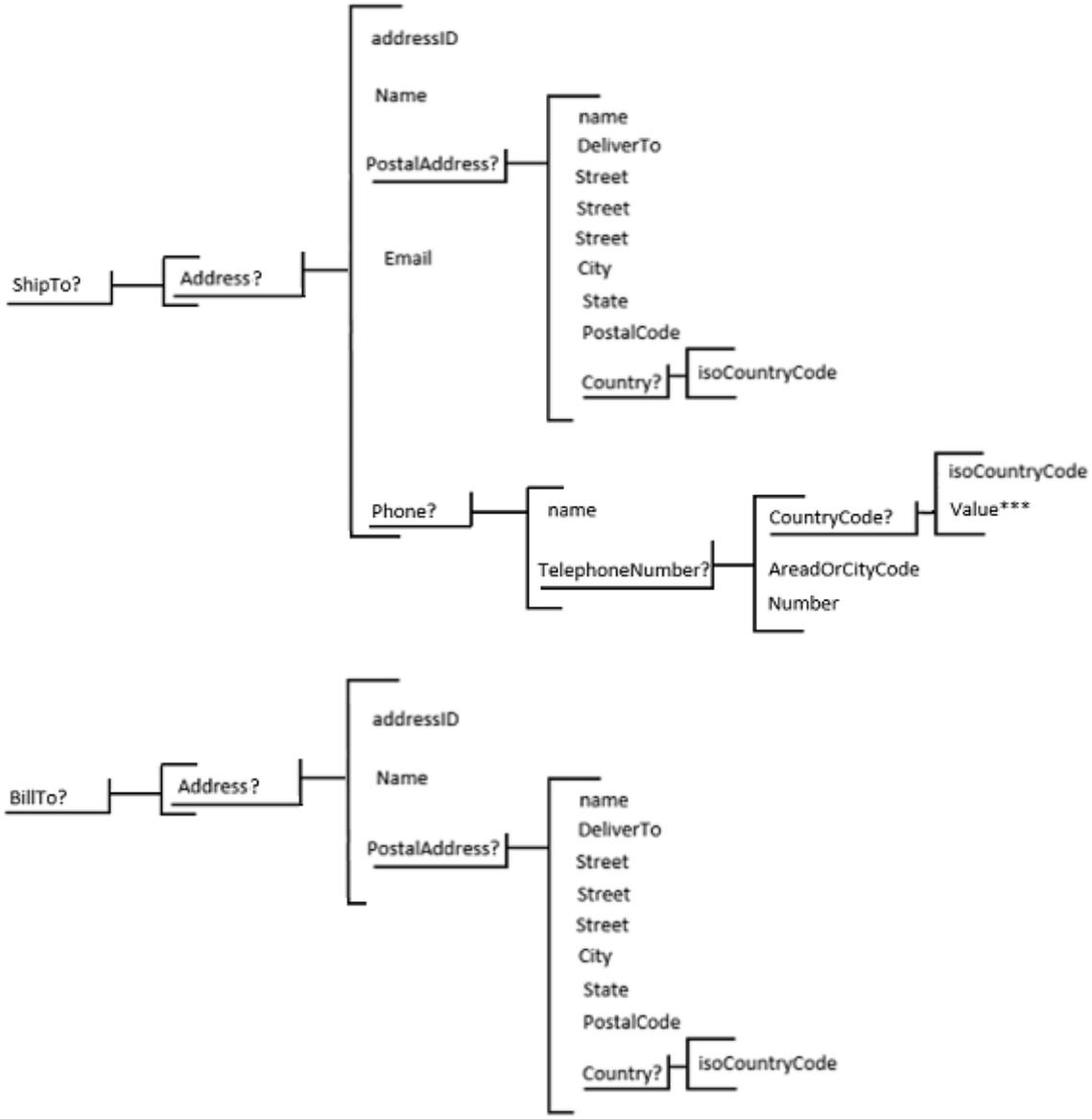

| | |
|---|---|
| A | User transmits the order from Fourth's Adaco to Supplier System |
| B | Fourth's Adaco sends the order item information in cXML format (OrderRequest) |
| C | Supplier sends the response whether they received the order successfully or if there is an issue with the cXML order (OrderResponse) |

## 11.1 OrderRequest Document

This document initiates the buyer's request for ordering the items for delivery.

This document contains Declaration, Envelope and Header OrderRequest elements. For Header element please refer the cXML Header section. For Declaration and Envelope elements please refer this section.




ShipTo?
Address?
addressID
Name
PostalAddress?
Email
Phone?
name
DeliverTo
Street
Street
Street
City
State
PostalCode
Country?
isoCountryCode
name
TelephoneNumber?
CountryCode?
isoCountryCode
Value***
AreadOrCityCode
Number
BillTo?
Address?
addressID
Name
PostalAddress?
name
DeliverTo
Street
Street
Street
City
State
PostalCode
Country?
isoCountryCode

```
<?xml version="1.0" encoding="UTF-8"?>
- <cXML xml:lang="en-US" timestamp="2013-04-08T02:29:19"
payloadID="66ad5edafd6147d3bbbe0c52e8362e42" version="1.2.020"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

- <Header>
  - <From>
    - <Credential domain="NetworkID">
        <Identity>[redacted]</Identity>
      </Credential>
    - <Correspondent>
      - <Contact>
          <Name xml:lang="en">[redacted]</Name>
          <Email>[redacted]</Email>
        </Contact>
      </Correspondent>
    </From>
  - <To>
    - <Credential domain="DUNS">
        <Identity>[redacted]</Identity>
      </Credential>
    </To>
  - <Sender>
    - <Credential domain="NetworkID">
        <Identity>[redacted]</Identity>
        <SharedSecret>[redacted]</SharedSecret>
      </Credential>
      <UserAgent>ADACO</UserAgent>
    </Sender>
  </Header>

- <Request deploymentMode="production">
  - <OrderRequest>
    - <OrderRequestHeader orderType="regular" orderDate="2014-07-10T11:31:21" orderID="79960">
      - <Total>
          <Money currency="USD">644.66</Money>
        </Total>
      - <ShipTo>
        - <Address addressID="1">
            <Name xml:lang="en">[redacted]</Name>
          - <PostalAddress name="[redacted]">
              <DeliverTo>[redacted]</DeliverTo>
              <Street>[redacted]</Street>
              <Street>[redacted]</Street>
              <Street>[redacted]</Street>
              <City>[redacted]</City>
              <State>[redacted]</State>
              <PostalCode>[redacted]</PostalCode>
              <Country isoCountryCode="US"/>
            </PostalAddress>
            <Email name="[redacted]">[redacted]</Email>
          - <Phone name="Work">
            - <TelephoneNumber>
                <CountryCode isoCountryCode="US">1</CountryCode>
                <AreaOrCityCode>[redacted]</AreaOrCityCode>
                <Number>[redacted]</Number>
              </TelephoneNumber>
            </Phone>
          </Address>
        </ShipTo>
```

```
- <BillTo>
  - <Address addressID="1">
      <Name xml:lang="en">[redacted]</Name>
    - <PostalAddress name="[redacted]">
        <Street>[redacted]</Street>
        <Street>[redacted]</Street>
        <Street>[redacted]</Street>
        <City>[redacted]</City>
        <State>[redacted]</State>
        <PostalCode>[redacted]</PostalCode>
        <Country isoCountryCode="US">US</Country>
      </PostalAddress>
    </Address>
  </BillTo>
- <Payment>
    <PCard expiration="2015-08-06" number="[redacted]"/>
  </Payment>
- <Contact role="buyer">
    <Name xml:lang="en">[redacted]</Name>
    <Email name="[redacted]">[redacted]</Email>
  </Contact>
</OrderRequestHeader>

    - <ItemOut isAdHoc="yes" lineNumber="1" quantity="10.00">
      - <ItemID>
          <SupplierPartID>[redacted]</SupplierPartID>
          <SupplierPartAuxiliaryID>[redacted]</SupplierPartAuxiliaryID>
        </ItemID>
      - <ItemDetail>
        - <UnitPrice>
            <Money currency="USD">8.7600</Money>
          </UnitPrice>
          <Description xml:lang="en">TUMBLER,12 OZ,AMBER,STACKABLE </Description>
          <UnitOfMeasure>EA</UnitOfMeasure>
          <Classification domain=""/>
        </ItemDetail>
      </ItemOut>
    + <ItemOut isAdHoc="yes" lineNumber="2" quantity="2.00">
    + <ItemOut isAdHoc="yes" lineNumber="3" quantity="2.00">
    + <ItemOut isAdHoc="yes" lineNumber="4" quantity="3.00">
    </OrderRequest>
  </Request>
</cXML>
```

***Note**: Elements ShipTo > Phone will be populated based on the customer's data in Fourth's Adaco. Payment node will be populated if the supplier agrees for the PCard payment for the order in cXML. If the supplier doesn't agree for the payment via cXML, we don't send that element.*

### 11.1.1 OrderRequest Element

It provides the delivery, billing, and item information of the order along with items that were ordered through PunchOut Requisition. PunchOut process is about requisitioning of products, and it is not an actual order to be delivered. This request makes the actual order request to the supplier.

OrderRequest document represents the purchase orders. These documents consist of one OrderRequestHeader element which represents order header information such as billing address for invoices, delivery address, payment information and the buyer information. The other one being ItemOut element which represents items in the purchase order.

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| deploymentMode | String | 10 | Y | | production or test |
| orderType | String | 7 | Y | This attribute describes the order type: regular, release (a release against a master agreement), or blanket (a blanket purchase order). | regular |
| orderDate | String | N/A | Y | This attribute is the date and time the order was created by the buying organization. Dates are in ISO 8601 format: YYYY-MM-DDThh:mm:ss-hh:mm. | |
| orderID | Int | 1-9 | Y | This attribute represents customer's order number coming from Fourth's Adaco | |
| OrderRequestHeader > Total | Money | | Y | The total value of the order | |
| ShipTo > Address > addressID | Int/String | 25 | Y | By default, we pass the customer's Organization number in Fourth's Adaco. If you want to pass different number or string, please let us know | |
| ShipTo > Address > Name | String | 50 | Y | Customer's Organization Name | |
| ShipTo > Address > PostalAddress > name | String | 50 | Y | It represents the user who initiated the PunchOut request | |
| ShipTo > Address > PostalAddress > DeliverTo | String | 50 | Y | It represents the user who initiated the PunchOut request | |
| ShipTo > Address > PostalAddress > Street (1,2,3) | String | 50 | N | Represents the delivery address street information of the customer where the items of the order to be delivered | |

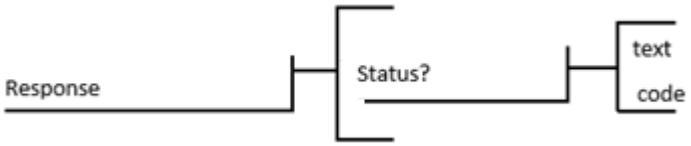
| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| ShipTo > Address > PostalAddress > City | String | 50 | N | Represents the delivery address city information of the customer where the items of the order to be delivered | |
| ShipTo > Address > PostalAddress > State | String | 50 | N | Represents the delivery address state information of the customer where the items of the order to be delivered | |
| ShipTo > Address > PostalAddress > PostalCode | String | 25 | N | Postal code of the delivery address | |
| ShipTo > Address > PostalAddress > Country | Element | 50 | N | Attribute isoCountryCode represents ISO country code (ISO 3166 standard). The content of the country will be the name of the delivery address country. | |
| ShipTo > Address > PostalAddress > Email | Element | 60 | N | Name would be the PunchOut initiator. The content of the email would be initiator's email address. | |
| ShipTo > Address > Phone > name | String | 4 | Y | Phone contact type | work |
| ShipTo > Address > Phone > TelephoneNumber> CountryCode | Element | 1 | N | We follow ISO country code (ISO 3166 standard) for isoCountryCode element and content would be 1 | Value:1 |
| ShipTo > Address > Phone > TelephoneNumber> AreaOrCityCode | String | 25 | N | Area or city code of the buyer's contact # | |
| ShipTo > Address > Phone > TelephoneNumber> Number | String | 25 | N | Rest of the buyer's contact # | |
| BillTo > Address > addressID | Int/String | 25 | Y | By default, we pass the customer's Organization number in Fourth's Adaco. | |
| BillTo > Address > Name | String | 50 | Y | Customer's Organization Name | |
| BillTo > Address > PostalAddress > name | String | 50 | N | It represents the department name of the A/P where the invoice/bill to be sent. | |
| BillTo > Address > PostalAddress > Street (1,2,3) | String | 50 | N | Represents the Bill To address street information of the customer where the invoice/bill to be sent | |
| BillTo > Address > PostalAddress > City | String | 50 | N | Represents the bill to address city information of the customer where the invoice to be sent | |

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| BillTo > Address > PostalAddress > State | String | 50 | N | Represents the Bill To address state information of the customer where the invoice to be sent | |
| BillTo > Address > PostalAddress > PostalCode | String | 25 | N | Postal code of the Bill To address | |
| BillTo > Address > PostalAddress > Country | Element | 50 | N | Attribute isoCountryCode represents ISO country code (ISO 3166 standard). The content of the country will be the name of the Bill To address country. | |
| Payment > PCard > Expiration | Date | | N | Format: YYYY-MM-DD | |
| Payment > PCard > number | Int | 12-16 | N | | |
| Contact > role | String | 5 | Y | | buyer |
| Contact > Name | String | 60 | Y | The name of the buyer who sent the order to the supplier | |
| Contact > Email | Element | | N | Buyer's name and email information | |
| ItemOut > lineNumber | Int | 1-9 | Y | Line number of the item in the customer's order | |
| ItemOut > quantity | Decimal | | Y | Ordered Quantity | |
| ItemOut > ItemID > SupplierPartID | String | 200 | Y | Supplier's Item/Product #. | |
| ItemOut > ItemID > SupplierPartAuxiliaryID | String | 4000 | Y | Supplier cookie set up during PunchOut. We will pass same information that was passed to us during PunchOut Session (The supplier can use the supplier cookie to associate items in a purchase requisition with the corresponding items in a shopping cart at the supplier's PunchOut site) | |
| ItemOut > ItemDetail > UnitPrice > Money | Money | | Y | Unit price (by Unit of measure) of each individual item not the total amount of the item. | |
| ItemOut > ItemDetail > Description | String | 2000 | Y | Ordered Item description | |
| ItemOut > ItemDetail > UnitOfMeasure | String | 5 | Y | Purchasing Unit of measure | |
| ItemOut > ItemDetail > Classification | String | N/A | N | Currently we don't use this information | |

## 11.2 OrderResponse Document

After the supplier received the OrderRequest, supplier must respond with OrderResponse document. This document contains Declaration, Envelope and Response elements. For Declaration and Envelope elements please refer this section

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.005/cXML.dtd">
<cXML xml:lang="en-US" timestamp="5/18/2015 2:33:08 AM" payloadID="5/18/2015 2:33:08 AM [redacted] version="1.2.005">
 - <Response>
      <Status text="OK" code="200"/>
    </Response>
 </cXML>
```

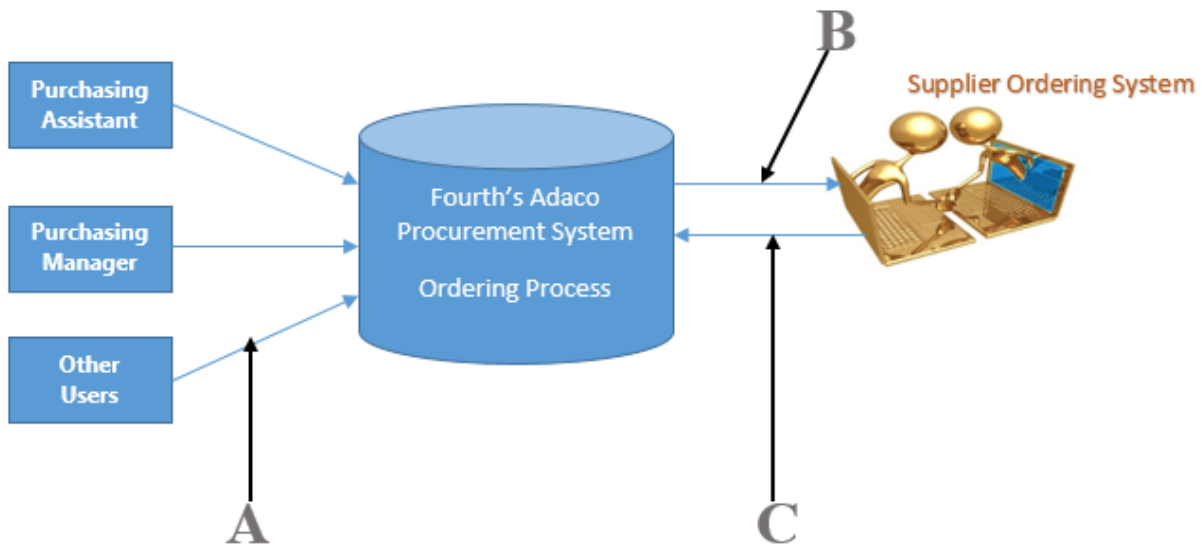



### 11.2.1 Status Element

Please refer this for different status codes

```
<!ELEMENT Status (#PCDATA)>
<!ATTLIST Status
    code      %uint;         #REQUIRED
    text      %string;       #REQUIRED
    xml:lang %xmlLangCode; #IMPLIED
>
```

# 12. Invoice Transaction

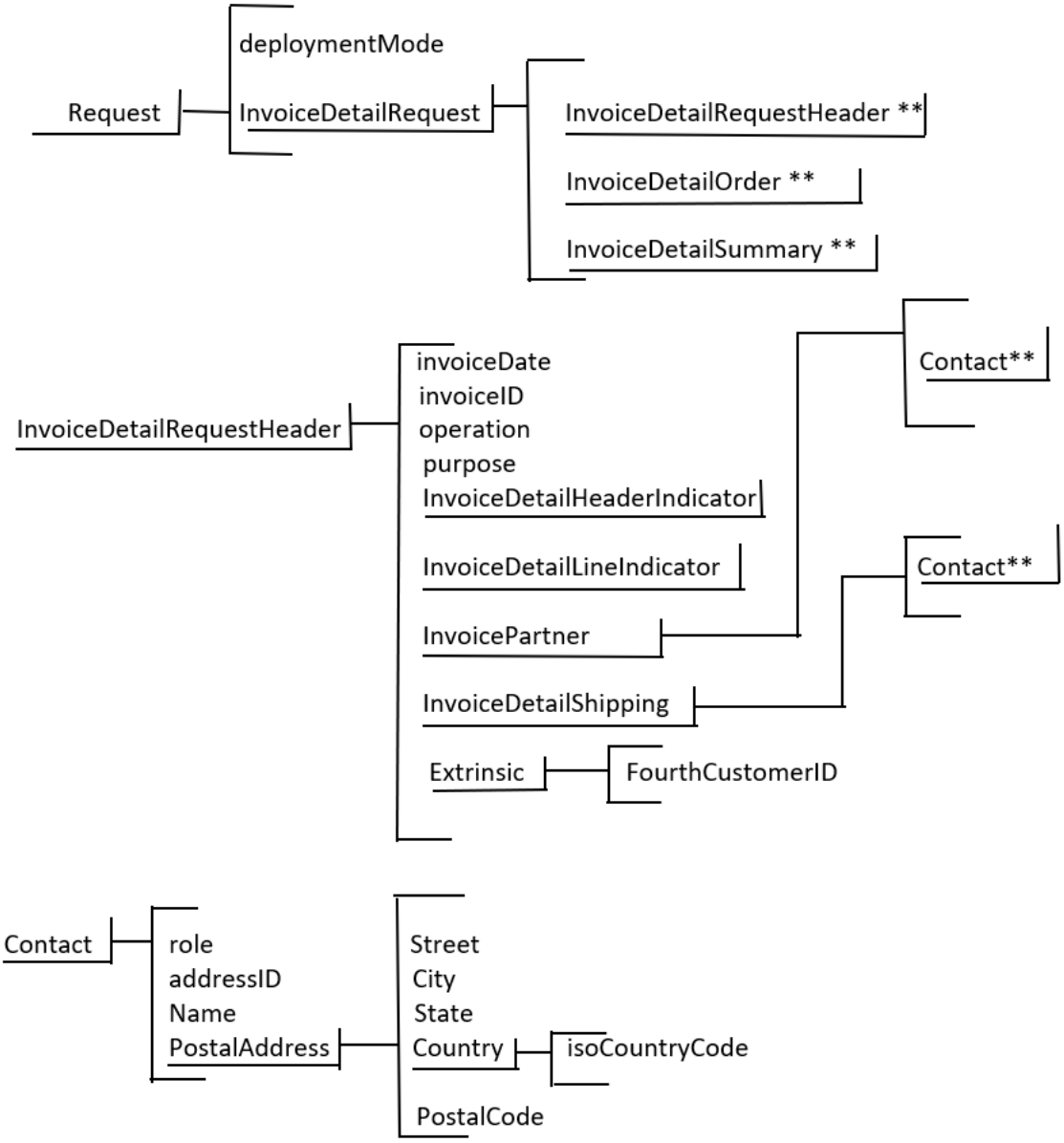


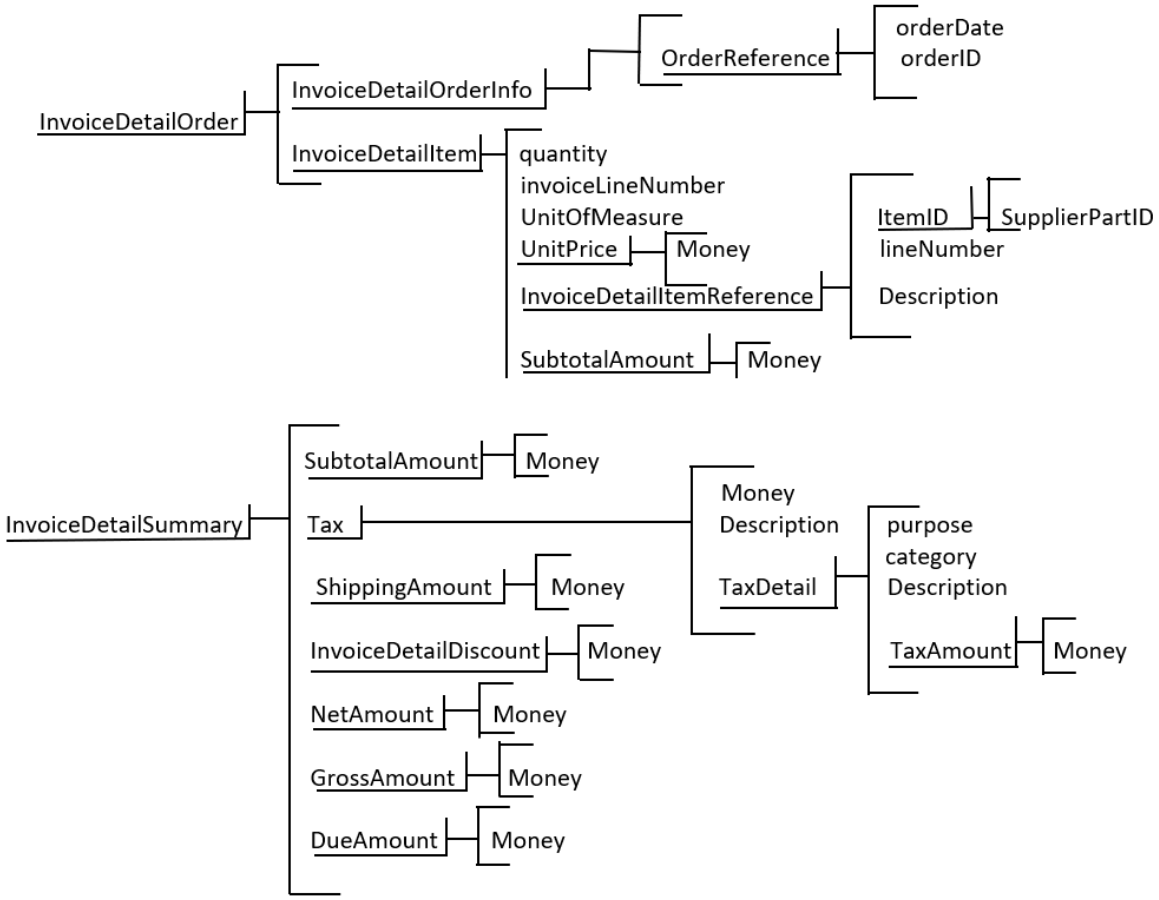

| | |
|---|---|
| A | Supplier submits the invoice information to Fourth's Adaco. (InvoiceDetailRequest) |
| B | Fourth's Adaco sends the invoice confirmation in cXML format (Response) |
| C | |

## 12.1 InvoiceDetailRequest Document

This document initiates the supplier request for invoice information to the Fourth system.

This document contains Declaration, Envelope and Header OrderRequest elements. For Header element please refer the cXML Header section. For Declaration and Envelope elements please refer this section.


deploymentMode
Request
InvoiceDetailRequest
InvoiceDetailRequestHeader **
InvoiceDetailOrder **
InvoiceDetailSummary **
invoiceDate
invoiceID
operation
purpose
InvoiceDetailRequestHeader
InvoiceDetailHeaderIndicator
InvoiceDetailLineIndicator
InvoicePartner
InvoiceDetailShipping
Extrinsic
FourthCustomerID
Contact**
Contact**
Contact
role
addressID
Name
PostalAddress
Street
City
State
Country
isoCountryCode
PostalCode


InvoiceDetailOrder
InvoiceDetailOrderInfo
OrderReference
orderDate
orderID
InvoiceDetailItem
quantity
invoiceLineNumber
UnitOfMeasure
UnitPrice
Money
InvoiceDetailItemReference
ItemID
SupplierPartID
lineNumber
Description
SubtotalAmount
Money
InvoiceDetailSummary
SubtotalAmount
Money
Tax
Money
Description
TaxDetail
purpose
category
Description
TaxAmount
Money
ShippingAmount
Money
InvoiceDetailDiscount
Money
NetAmount
Money
GrossAmount
Money
DueAmount
Money

```
<?xml version="1.0"?>
<!DOCTYPE cXML SYSTEM "https://protect-eu.mimecast.com/s/lOdEC82gNCY7j0gIk54B4?domain=xml.cxml.org">
- <cXML xml:lang="en-US" timestamp="2022-04-13T01:05:39-05:00" payloadID="[redacted]"
version="1.2.011">
  - <Header>
    - <From>
      - <Credential domain="DUNS">
          <Identity>[redacted]</Identity>
        </Credential>
      </From>
    - <To>
      - <Credential domain="NetworkID">
          <Identity>0016A00000VEmofQAD|11|2353</Identity>
        </Credential>
      </To>
    - <Sender>
      - <Credential domain="DUNS">
          <Identity>[redacted]</Identity>
          <SharedSecret>[redacted]</SharedSecret>
        </Credential>
        <UserAgent>[redacted]</UserAgent>
      </Sender>
    </Header>
- <Request deploymentMode="production">
  - <InvoiceDetailRequest>
    - <InvoiceDetailRequestHeader invoiceDate="2022-07-19T12:00:00-05:00" operation="new" purpose="standard" invoiceID="603027890010">
        <InvoiceDetailHeaderIndicator/>
        <InvoiceDetailLineIndicator/>
      - <InvoicePartner>
        - <Contact role="remitTo">
            <Name xml:lang="en-US">[redacted]</Name>
          - <PostalAddress>
              <Street>[redacted]</Street>
              <City>[redacted]</City>
              <State>[redacted]</State>
              <PostalCode>[redacted]</PostalCode>
              <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
          </Contact>
        </InvoicePartner>
      - <InvoicePartner>
        - <Contact role="soldTo">
            <Name xml:lang="en-US">[redacted]</Name>
          - <PostalAddress>
              <Street>[redacted]</Street>
              <City>[redacted]</City>
              <State>[redacted]</State>
              <PostalCode>[redacted]</PostalCode>
              <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
          </Contact>
        </InvoicePartner>
      - <InvoiceDetailShipping>
        - <Contact role="shipTo" addressID=" ">
            <Name xml:lang="en-US">[redacted]</Name>
          - <PostalAddress>
              <Street>[redacted]</Street>
              <City>[redacted]</City>
              <State>[redacted]</State>
              <PostalCode>[redacted]</PostalCode>
              <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
          </Contact>
        - <Contact role="shipFrom">
            <Name xml:lang="en-US">[redacted]</Name>
          </Contact>
        </InvoiceDetailShipping>
        <Extrinsic name="invoice due date">20220812</Extrinsic>
        <Extrinsic name="FourthCustomerID">0016A00000VEmofQAD|11|2353</Extrinsic>
      </InvoiceDetailRequestHeader>
    - <InvoiceDetailOrder>
      - <InvoiceDetailOrderInfo>
        - <OrderReference orderDate="2022-07-19" orderID="284790">
            <DocumentReference payloadID="[redacted]"/>
          </OrderReference>
          <SupplierOrderInfo orderID="603027890001"/>
        </InvoiceDetailOrderInfo>
      - <InvoiceDetailItem quantity="2" invoiceLineNumber="1">
          <UnitOfMeasure>BX</UnitOfMeasure>
        - <UnitPrice>
            <Money currency="USD">6.99</Money>
          </UnitPrice>
        - <InvoiceDetailItemReference lineNumber="1">
          - <ItemID>
              <SupplierPartID>664011</SupplierPartID>
            </ItemID>
            <Description xml:lang="en-US">[redacted]</Description>
          </InvoiceDetailItemReference>
          <Extrinsic name="UNSPSC">[redacted]</Extrinsic>
        </InvoiceDetailItem>
      </InvoiceDetailOrder>
    - <InvoiceDetailSummary>
      - <SubtotalAmount>
          <Money currency="USD">13.98</Money>
        </SubtotalAmount>
      - <Tax>
          <Money currency="USD">0.98</Money>
          <Description xml:lang="en-US">Tax</Description>
        - <TaxDetail purpose="tax" percentageRate="7.00" category="sales">
          - <TaxAmount>
              <Money currency="USD">0.98</Money>
            </TaxAmount>
          </TaxDetail>
        </Tax>
      - <GrossAmount>
          <Money currency="USD">13.98</Money>
        </GrossAmount>
      - <NetAmount>
          <Money currency="USD">14.96</Money>
        </NetAmount>
      </InvoiceDetailSummary>
    </InvoiceDetailRequest>
  </Request>
</cXML>
```

### 12.1.1 InvoiceDetailRequest Element

It provides the delivery, billing, and item information of the order along with items that were ordered through Order Request.

This document represents the item delivery information of submitted purchase orders. These documents consist of one InvoiceDetailRequestHeader element which represents Invoice header information such as billing address for invoices, delivery address, invoice number and invoice date. If supplier cannot provide our credentials at "To" identity, they need to provide the same info at Extrinsic tag with the tag name "FourthCustomerID". This is important as without this we can't identify the invoiced customer. The format of FourthCustomerID would be below and we will provide the information.

<salesforceid of customer>|property ship to id| Supplier number in Adaco Fourth

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| deploymentMode | String | 10 | Y | | production or test |
| InvoiceDetailRequest > InvoiceDetailRequest Header > invoiceDate | String | | Y | This attribute is the date and time the invoice was created by the supplier organization. Dates are in ISO 8601 format: YYYY-MM-DDThh:mm:ss-hh:mm | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > invoiceID | String | N/A | Y | This is the invoice number that is provided by the supplier | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > operation | String | | N | We don't use it. We don't support "EDIT" functionality. | new |
| InvoiceDetailRequest > InvoiceDetailRequest Header > purpose | String | | N | We don't use it. | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailHeaderIndicator | String | | N | We don't use it. | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailLineIndicator | String | | N | We don't use it. | |

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Role | String | | N | We don't use it | billTo, remitTo, shipTo, soldTo |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Name | String | | N | Name of bill to, ship to, remit To, ship From organization | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Postal Address > Street | String | | N | Bill to/ ship to/ remit to/ ship to/ ship from postal address Street | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Postal Address > City | String | | N | Bill to/ ship to/ remit to/ ship to/ ship from postal address city | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Postal Address > State | String | | N | Bill to/ ship to/ remit to/ ship to/ ship from postal address state | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Postal Address > PostalCode | String | | N | Bill to/ ship to/ remit to/ ship to/ ship from postal address postal code | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoicePartner > Contact > Postal Address > Country | Element | 50 | N | Attribute isoCountryCode represents ISO country code (ISO 3166 standard). | |

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Role | String | | N | We don't use it | shipTo, soldTo |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Name | String | | N | ship To, ship From organization | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Postal Address > Street | String | | N | ship to/ ship from postal address Street | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Postal Address > City | String | | N | Ship to/ ship from postal address city | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Postal Address > State | String | | N | Ship to/ ship from state | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Postal Address > PostalCode | Int/String | | N | Ship to/ ship postal address postal code | |
| InvoiceDetailRequest > InvoiceDetailRequest Header > InvoiceDetailShipping > Contact > Postal Address > Country | String | | N | Attribute isoCountryCode represents ISO country code (ISO 3166 standard). | |

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| InvoiceDetailRequest > InvoiceDetailRequest Header > Extrinsic > FourthCustomerID | String | 50 | Y | We will provide the info. If you pass the info @ To credential, then you can ignore this. | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailOrderInfo> OrderReference > orderDate | Date | | N | Format: "YYYY-MM-dd" | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailOrderInfo> OrderReference > orderID | int | | Y | Represents the Fourth's order number that was sent in Order Request. Without this we can't process the invoice. | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> quantity | decimal | | Y | Invoiced quantity | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> invoiceLineNumber | int | | Y | Invoice line number. This should match with physical invoice | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> unitOfMeasure | string | 50 | Y | Item unit of measure. We will use ANSI unit codes. Please share the unit of measure you use with our PM. | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> UnitPrice > Money | Decimal | | Y | Price of the item that corresponds to the unit of measure. | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> InvoiceDetailItemReference > lineNumber | Int | | Y | Line number in the order request | |

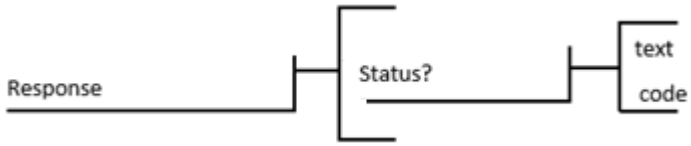
| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> InvoiceDetailItemReference > ItemID > SupplierPartID | String | | Y | Supplier's invoiced Item number. | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailItem> InvoiceDetailItemReference > Description | String | | Y | Supplier's invoiced item description. Please escape XML characters (> < etc) in the description | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary | Element | | N | Invoice total summary | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > SubtotalAmount | Decimal | | Y | Sum of line item totals excluding discounts, tax, and shipping | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > Tax | Element | | Y | Tax element of the invoice | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > Tax > Money | Decimal | | Y | Total tax Amount | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > Tax >Description | String | | Y | | tax |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > Tax > TaxDetail | Element | | Y | If it has multiple tax elements in the invoice loop through | |

| *Element/Attribute* | *Type* | *Size* | *Mand.* | *Comments* | *Default Value* |
|---|---|---|---|---|---|
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > Tax > TaxDetail > Description | String | | Y | Tax Type e.g., sales tax, VAT, county tax | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > Tax > TaxDetail > TaxAmount> Money | Decimal | | Y | Tax amount by tax type | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > ShippingAmount | Decimal | | N | Shipping Amount | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > InvoiceDetailDiscount | Decimal | | N | Discount Amount | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > NetAmount | Decimal | | Y | Includes all item amount + discount amount | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > GrossAmount | Decimal | | Y | Includes all amounts (normally this is the amount that the buyer has to pay to the supplier) | |
| InvoiceDetailRequest > InvoiceDetailOrder> InvoiceDetailSummary > DueAmount | Decimal | | Y/N | Includes all amounts (normally this is the amount that the buyer has to pay to the supplier) | |

## 12.2 Invoice Detail Response Document

After the Fourth received the InvoiceDetailRequest, we will respond with Response document. This document contains Declaration, Envelope and Response elements. For Declaration and Envelope elements please refer this section

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.005/cXML.dtd">
<cXML xml:lang="en-US" timestamp="5/18/2015 2:33:08 AM" payloadID="5/18/2015 2:33:08 AM[redacted]" version="1.2.005">
  - <Response>
      <Status text=" Accepted " code="202"/>
    </Response>
</cXML>
```




### 12.2.1 Status Element

Please refer this for different status codes

```
<!ELEMENT Status (#PCDATA)>
<!ATTLIST Status
    code      %uint;         #REQUIRED
    text      %string;       #REQUIRED
    xml:lang %xmlLangCode; #IMPLIED
>
```

## 13. Extrinsic Tags

Fourth's Adaco have the capability to support extrinsic tags both in Request as well as in Message sections of the documents. These are the three documents we currently support extrinsic tags.

- PunchOutSetupRequest
- PunchOutOrderMessage
- OrderRequest
- InvoiceDetailRequest

This optional element contains any additional data that the supplier requests/sends to pass with the request/message document. The cXML specification does not define the content of extrinsic elements—it is something that each requestor (Fourth's Adaco) and remote website (Supplier) must agree on and implement.

Extrinsic elements are intended to provide additional machine-readable information. They extend the cXML protocol to support features not required by all implementations. In the following context, the new data further describes the user initiating the PunchOut request.

```
<Extrinsic name="CostCenter">2014</Extrinsic>
<Extrinsic name="UniqueName">Adaco</Extrinsic>
```

We currently support 2 levels of extrinsic tags. 1 at Request/Message (Top) level and one at item (bottom) level.

Top Level:

```
- <PunchOutSetupRequest operation="create">
    <BuyerCookie>05732aa032584c8589512c27520a2001</BuyerCookie>
    <Extrinsic name="CostCenter">2014</Extrinsic>
    <Extrinsic name="UniqueName">Adaco</Extrinsic>
    <Extrinsic name="User">Adaco</Extrinsic>
    <Extrinsic name="UserType">ADC</Extrinsic>
    <Extrinsic name="UserFullName">Adaco2014</Extrinsic>
  - <BrowserFormPost>
      <URL>http://intranet.adaco.com/vendor/cXML/PostPunchOutOrder.aspx</URL>
    </BrowserFormPost>
```

Bottom Level:

```
- <ItemDetail>
  - <UnitPrice>
      <Money currency="USD">4.75</Money>
    </UnitPrice>
  - <Description xml:lang="EN">
      Beef Chuck Flap Meat
      <ShortName>Beef Chuck Flap Meat </ShortName>
    </Description>
    <UnitOfMeasure>LB</UnitOfMeasure>
    <Classification domain="UNSPSC">50000000</Classificatio
    <ManufacturerPartID>11285</ManufacturerPartID>
    <Extrinsic name="splitIndicator">No</Extrinsic>
  </ItemDetail>
```

Fourth's Adaco support most of the outgoing extrinsic tags. We also support static value tags along with dynamically populated value tags from Fourth's Adaco application (see the table below). Please let us know, if you need any of the tags to be populated with while sending the Requests.

Below are the tags that we normally capable of sending from Fourth's Adaco application to the buyer (there are couple of tags where we expect to receive for certain suppliers in PunchOutOrder Message such as DocumentURL and CatchWeight information)

**Document**:

P.O.R → PunchOutSetupRequest
P.O.M→ PunchOutOrderMessage
O.R → OrderRequest

**Type**: It represents the data type and in value in braces represents the maximum size we support.

| *Tag Name* | *Type* | *Document* | *Header/Item* | *Description* |
|---|---|---|---|---|
| AccountNumber | Int (9) | O.R | Item | Item's A/P account # in Fourth's Adaco |
| AccountName | String (30) | O.R | Item | Item's A/P account name in Fourth's Adaco |
| AccountXReference | String (30) | O.R | Item | Item's A/P account reference in Fourth's Adaco |
| AverageWeight | decimal | P.O.M, O.R | Item | If an item is a catch weight item, it represents weighted average of the item. For 5 LB it represents 5. |
| CasePack | String (25) | P.O.M, O.R | Item | Represents number of packs with unit in a case for an item. For e.g., if case contains 4/5 LB CasePack will give you 4 EA. |
| CatchWeight | String (25) | P.O.M, O.R | Item | Represents AverageWeight with unit for an item. For e.g., 5 LB |
| CustomerReference | String (50) | P.O.R, O.R | Header | Represents the Buyer's account # with the supplier |
| ExtendedCost | money | P.O.M, O.R | Item | Value which represents Quantity multiplied by price of an item |
| FinalApproverEmail | String (60) | O.R | Header | Email of User who approved the order (if exists) |
| FinalApproverFax | String (50) | O.R | Header | Fax # of User who approved the order (if exists) |
| FinalApproverName | String (60) | O.R | Header | Name of User who approved the order (if exists) |

| *Tag Name* | *Type* | *Document* | *Header/Item* | *Description* |
|---|---|---|---|---|
| FinalApproverPhone | String (50) | O.R | Header | Phone # of User who approved the order (if exists) |
| FobText | String (40) | O.R | Header | Supplier FOB information |
| ItemPerCase | decimal | O.R | Item | Represents number of packs without unit in a case for an item. For e.g., if case contains 4/5 LB CasePack will give you 4. |
| LineNumber | Int | P.O.M, O.R | Item | Sequential number of the items starting from 1,2,3 etc. |
| OrderComments | String (1000) | O.R | Header | Order comments entered by Buyer in Fourth's Adaco |
| OrderCreatorEmail | String (60) | O.R | Header | Order creator (buyer) email (if exists) |
| OrderCreatorFax | String (50) | O.R | Header | Order creator (buyer) fax (if exists) |
| OrderCreatorName | String (60) | O.R | Header | Order creator (buyer) name |
| OrderCreatorPhone | String (50) | O.R | Header | Order creator (buyer) phone (if exists) |
| OutletGlAccount | String (50) | P.O.R, O.R | Header | The department code in Fourth's Adaco who actually ordered these items |
| OutletGlOutlet | String (50) | P.O.R, O.R | Header | The A/P code of the department in Fourth's Adaco who actually ordered these items |
| OutletName | String (50) | P.O.R, O.R | Header | The department name in Fourth's Adaco who actually ordered these items |
| OutletNumber | Int | P.O.R, O.R | Header | The department number in Fourth's Adaco who actually ordered these items |
| PackSizeInformation | String (50) | O.R | Item | In e.g., 4/5 LB. It will provide 5 LB for each of 4. |
| PurchaseInformation | String (100) | O.R | Item | Will provide total pack info of the product. For e.g., 4/5 LB |
| RequisitionComments | String (1000) | P.O.R, O.R | Header | Comments coming from PunchOut session |
| RequisitionCreatorEmail | String (60) | P.O.R, O.R | Header | PunchOut initiator's email (if exists) |
| RequisitionCreatorFax | String (50) | P.O.R, O.R | Header | PunchOut initiator's fax # (if exists) |
| RequisitionCreatorName | String (60) | P.O.R, O.R | Header | PunchOut initiator's name |
| RequisitionCreatorPhone | String (50) | P.O.R, O.R | Header | PunchOut initiator's phone # (if exists) |

| *Tag Name* | *Type* | *Document* | *Header/Item* | *Description* |
|---|---|---|---|---|
| ShipVia | String (50) | O.R | Header | Shipping information |
| SplitItem | Bit/Boolean | P.O.M | Item | If an item can be sold full case as well as individual each (part of case) |
| SupplierCode | String (50) | O.R | Header | Supplier code |
| VendorApCrossReference | String (30) | P.O.R, O.R | Header | Supplier's number in Buyer's A/P |
| VendorContactEmail | String (60) | P.O.R, O.R | Header | Supplier's sales contact email for the buyer |
| VendorContactName | String (60) | P.O.R, O.R | Header | Supplier's sales contact name for the buyer |
| VendorPrice | Price of each item | P.O.M, O.R | Item | Supplier's item price |
| CatchWeightUnit | String(15) | P.O.M, O.R | Item | Unit of catch weight item. For 4 LB of item, it represents LB |
| DocumentURL | String (250) | P.O.M | Item | It will be populated in PunchOutOrderMessage where supplier provides the image/pdf url for the customized product (for e.g., Uniforms, business cards etc.). If you have such functionality, please let us know the tag name. We will map it on our side. |

Currently we consider only "Document URL" , "Catch Weight" and "FourthCustomerID" information from the incoming (to Fourth) cXML documents. These are the only 3 extrinsic tags which will have visual affect in Fourth's Adaco system. If you (supplier) believe there is some other tags that buyer needs to aware of, please let us know we will include them so that buyer can see.

## 14. Miscellaneous

- If PunchOut sessions do not end with a PunchOutOrderMessage document, Fourth's Adaco terminate the session in three hours.
- Fourth's Adaco will not handle the shipping and tax information coming from PunchOutOrderMessage document.
- Please be aware that Fourth's Adaco cannot handle an item being shipped from multiple supplier locations/distribution centers with different pricing in a single PunchOut session.
- Whatever mentioned in the above document are the only elements/attributes Fourth's Adaco can support.
- Currently we don't support cXML invoice, but we plan to support it in future.
- Please notify out Integration manager, if you don't have the capability to support "EDIT" cart during PunchOut session
- Fourth's Adaco cannot support "Change Orders" (type=" update" instead of "new") via cXML.
- Fourth's Adaco can support PCard payment only from cXML orders. If you have the capability and buyer wants to use that functionality, please inform our integration/project manager.
- Will the supplier allow if the buyer sends us the same order multiple times?
- For non-US suppliers, please let inform our integration/project manager about the currency supplier supports.
- Please provide the list of Unit of measures supplier supports in their PunchOut and Ordering system.
- We will support only one invoice per cXML file.
- We don't support cXML invoice "Edit" or "Delete" operations.
- Without our credentials in "To" elements or FourthCustomerID as extrinsic at InvoiceDetailRequestHeader element, we can't process the invoice.