Pedigree Ratified Standard

Version 1.0 as of January 5th, 2007

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained at the EPCglobal.

This is an EPCglobal Standard as reviewed and ratified by EPCglobal Board Members and other interested parties. Although this is a ratified document, it may be updated, replaced or made obsolete by other documents at any time.

Comments on this document should be sent to the EPCglobal Drug Pedigree mailing list sag_dpm_wg@lists.epcglobalinc.org.

Disclaimer

EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable for any injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this document. Use of said document does not guarantee compliance with applicable state and/or federal laws. User is responsible for the interpretation of and compliance with applicable pedigree laws. The use of said document shall constitute your express consent to the foregoing disclaimer.

Table of Contents

1 Introduction

This document and its associated attachments specify an architecture for the maintenance and exchange of electronic pedigree documents for use by pharmaceutical supply chain participants. The architecture is targeted for use in complying with *document-based* pedigree laws. There are other architectural approaches to electronic pedigree systems to which this standard will not apply. For example, pedigree laws in Italy and Belgium today implement pedigrees using different architectures. Because those laws do not specify the use of an open document model, this standard may not be applicable for compliance to those laws at the time of this v1.0 release date. Nothing in this standard limits its application to only the United States. The standard may be properly applied anywhere that an open document model is specified or implied in the pedigree regulations. Since, at the time of the v1.0 release, the only known document based pedigree laws exist in the United States, the standard was created using the United States requirements as the basis. This pedigree standard should be applicable wherever document-based compliance is required.

The attachments to this document are two XML Schema documents that define a standard electronic pedigree format that can be used in all jurisdictions with open document-based pedigree laws and a standard electronic envelope format that can be used by supply chain partners to package multiple pedigree documents for exchange. The great benefit of these schemas is that they provide standardization between supply chain partners for the exchange and extensibility of electronic pedigrees as they pass down the supply chain.

Enough flexibility was built into the pedigree schema to allow for multiple interpretations of the existing and possible future US state, US federal and international pedigree laws. Even so, a version mechanism is included to allow for changes that may be necessary as the result of future laws.

Interpretation of and compliance with the various pedigree laws is left entirely up to the user. That is, use of the pedigree schema will not guarantee compliance with the laws. On the other hand, EPCglobal believes that all of the necessary ingredients are present to enable the users to comply with those laws.

Electronic pedigree documents created using the pedigree schema and applied as intended will become legal documents. Their contents are tightly regulated by law and the digital signatures applied are legally binding with severe penalties imposable for fraud.

Current legislation in multiple states of the United States dictates the creation and updating of electronic pedigrees at each stop in the pharmaceutical supply chain. Each state law specifies the data content of the electronic pedigree and the digital signature standards but none of them specifies the actual format of the document. The need for a standard electronic document format that can be updated by each supply chain participant is what has driven the creation of this specification. However, there is recognition that not all members of the pharmaceutical supply chain will adopt the XML Schema that is specified in this standard. Toward that end, the schema includes elements that allow the attachment of other document types as MIME documents. This could include scanned images, PDF documents or X.12 documents. Electronic pedigree documents stored in some structured alternate formats will require specialized software to allow them to be auto-authenticated. Some formats, such as PDFs and scanned images, will always require manual authentication.

In compliance with the pedigree law in affect in the US state of Florida, the digital signatures used in the pedigree schema comply with several of the FIPS (Federal Information Processing Standards). The use of FIPS for the digital signatures qualifies them as legally binding signatures just as if an individual had signed a paper legal document with pen in hand. There is an important distinction between digital signatures used to encrypt a document simply to keep it private, and the signatures in an electronic pedigree document, where the signature is used to legally bind an individual or company to the contents of the document. The reader is cautioned to keep this distinction in mind.

One can imagine a day when all of the components of an ePedigree are distributed across the network and single query is issued that collects those components into a complete ePedigree document for validation. However, this vision is inconsistent with the legislation as it stands today. States are looking at an ePedigree as a complete electronic legal document directly containing and signing over pedigree documents created earlier in the chain.

As such, the model for the distribution of ePedigree documents will be the direct transfer of an entire ePedigree document from one trading partner to another. There are a several mechanisms which are likely to be utilized for this transfer. Certainly many existing b2b systems could be augmented to include the ePedigree document. Even mechanisms like FTP or email could be utilized for this exchange.

The Standard does not identify exactly how pedigree documents must be transferred between trading partners. Any mechanism chosen must provide document immutability, non-repudiation and must be secure and authenticated. Although the scope of this standard focuses on the pedigree and pedigree envelope interchange formats, secure transmission relies on the recommendations for securing pedigree transmissions defined by the HLS Information Work Group C [6].

2 Overview

Although the FDA believes domestic counterfeiting of pharmaceutical products is not widespread, regulators have witnessed an increase in the number of pharmaceutical counterfeiting activities. In an attempt to help ensure only authentic pharmaceutical products are distributed through the supply chain, some regulatory agencies have implemented or are considering provisions requiring a “pedigree” for drug products.

A pedigree is a certified record that contains information about each distribution of a prescription drug. It records the sale of an item by a pharmaceutical manufacturer, any acquisitions and sales by wholesalers or repackagers, and final sale to a pharmacy or other entity administering or dispensing the drug. The pedigree contains product information, transaction information, distributor information, recipient information, and signatures.

At this time the specific pedigree requirements vary by state in the United States. Some US pedigree laws require the manufacturer to initiate the pedigree, while others allow the first wholesaler or repackager to create the pedigrees for the items acquired directly from the

manufacturer. Some laws allow paper pedigrees or electronic pedigrees, other laws require electronic pedigrees.

Compliance with US state pedigree laws is mandatory and failure to comply could result in administrative fines, revocation of license and/or criminal prosecution. A pedigree received by or provided by an organization is a document of record for that organization and is subject to regulatory recordkeeping, record retention, and record availability requirements. The state of Florida is viewed as the first state in the US to require pedigrees for all pharmaceutical products regardless of their source and the path they follow through the supply chain. The effective date of the Florida pedigree legislation was July 1, 2006.

Each party engaged in the wholesale distribution of prescription drugs is required to provide pedigrees to the recipients of those drugs. A pedigree contains a signed certification from the originating party that the product is authentic. Many US pedigree laws require that recipients of pedigrees authenticate each previous transaction in the pedigree and add their own certification of receipt and authentication to the pedigree.

A high level, simplified pedigree process would be similar to the following:

- Create pedigree
- Add information to pedigree
- Certify (digitally sign) pedigree
- Send pedigrees for products in shipment to customer
- Receive pedigrees
- Electronically authenticate pedigrees
- Manually authenticate transactions that were not electronic
- Verify products received against authenticated pedigrees
- Certify (digitally sign) pedigree for receipt and authentication

3 Terminology

Within this specification, the terms SHALL, SHALL NOT, SHOULD, SHOULD NOT, MAY, NEED NOT, CAN, and CANNOT are to be interpreted as specified in Annex H of the ISO/IEC Directives, Part 2, 2004, 5th edition [7]. When used in this way, these terms will always be shown in ALL CAPS; when these words appear in ordinary typeface they are intended to have their ordinary English meaning.

All sections of this document, with the exception of Sections 1, 2, 4 and 5 are normative, except where explicitly noted as non-normative.

4 ePedigree Implementation Requirements/Rationale

The content of a valid electronic pedigree is specified by the each law. Each law is slightly different, but it is not difficult to generate a list of data elements that is a super-set of all known laws that allow the use of the document model. This exercise was recently performed by the

Unified Drug Pedigree Coalition [1], a group composed of representatives of companies in the pharmaceutical supply chain, their industry associations and various US state and federal regulatory agencies. The ePedigree schema includes all of these data elements and more.

Beyond just the bare necessity of the basic data elements, a successful electronic pedigree schema must provide a structure suitable for applying digital signatures that comply with these laws and that allows the entire document to be nested by succeeding owners and handlers of the product. At each stop in the supply chain the pedigree documents must include all previous revisions and digital signatures of all previous owners, as required by law. Each signature must be applied so that its payload includes the entire pedigree document up to that point in time. The ePedigree schema does this.

Some document based pedigree regulations allow supply chain participants to supply either electronic or paper pedigree documents—the choice being left to the desires and technical capabilities of the participant. This causes significant complications. It is conceivable, and likely, that some pedigrees will traverse from electronic to paper, and/or from paper to electronic, as they move down the supply chain. Another complication comes from the fact that no electronic pedigree document format is specified in the laws. While this schema is aimed at providing a standard electronic pedigree document format, no law requires its use. Supply chain participants may elect to create their own electronic formats which serve their interests and which comply with the letter of the laws at the same time.

The ePedigree schema takes these likely situations into account by providing the ability to optionally attach scanned images of paper documents and/or EDI documents within optional XML data elements. Because the structure of these attached documents is not known, they cannot be auto-authenticated without additional software. Some types of attached documents—scanned images of paper pedigrees, for example—may only be authenticated manually.

Pedigree laws generally require signatures on the pedigree documents for various purposes [3], [4]. The type of digital signature that is acceptable in the State of Florida, for example, uses a Public Key Infrastructure (PKI) implementation as defined in the US Federal Government FIPS standards [2]. Interpreted together these requirements lead to the necessity to nest the digital signatures on ePedigrees with each succeeding signature fully enclosing the entire electronic pedigree document up to that point, including all previous signatures. The ePedigree schema is carefully constructed to fulfill these requirements through the use of XML digital signature structures [5].

5 Utilization of ePedigree in the Supply Chain

The following is an inventory of actions (use cases) where a Pedigree would be utilized in the pharmaceutical supply chain.

- Creation of pedigrees by a manufacturer before the first wholesale distribution
- Creation of pedigrees by the first wholesaler, including the transaction information for the first wholesale distribution
- Creation of pedigrees by repackagers for repackaged items that include pedigree information from source items
- Adding outbound transaction information to pedigrees as part of a sales, transfer or return transaction

- Adding certification (signature) to pedigrees, signing the transaction information added, and all prior content
- Adding item serial number to pedigrees (if a wholesaler serializes a non-serialized item)
- Adding manual authentication information (for example, invoice, shipping document) to pedigrees to facilitate downstream manual authentication
- Adding receipt information and recipient signature to pedigrees, signing this information and all prior content
- Creating a pedigree for an individual item
- Creating a pedigree for a repackaged item, and including pedigree information for one or more "parent" items
- Creating a pedigree for an item that has a unique serial number
- Creating a pedigree for an item that does not have a serial number
- Creating a "singular" pedigree for each saleable item
- Creating an "aggregate" pedigree for a collection of saleable items that share the same product information (NDC and multiple lots) and prior chain of custody
- Creating an electronic pedigree from a paper pedigree or alternate form, and embedding a copy of the original pedigree in the electronic format
- Including "attachments" to a pedigree, such as scanned and EDI representations of invoices or shipping documents to satisfy regulatory manual authentication requirements
- Accommodating additional data elements in an extensible manner as regulatory requirements evolve
- Displaying all pedigree regulatory information in the pedigree (for example, drug product information, distributor information, recipient information, transaction information, receiving information, digital signatures)
- Representation of pedigrees in a portable format that can be transmitted electronically or via media
- Including container information (for example, relationship of products to cases) in addition to the pedigrees in the pedigree envelope
- Exchange of pedigrees between trading partners using existing business data transfer mechanisms (for example, EDIINT AS2)
- Exchange using a peer-to-peer model
- Electronic verification of each prior signature on the pedigree
- Electronic verification that the original, previously-signed content of the pedigree was unchanged since it was signed
- Attaching copies of manual authentication documents (for example, invoice, shipping document) with an electronic pedigree to facilitate 'self-authenticating' pedigrees

- Creating an “electronic envelope” for transmitting a collection of pedigrees associated with an outbound customer shipment
- Including key routing and identifying information in the “electronic envelope” (for example, shipment identifier, shipment date, originating trading partner, recipient trading partner) to facilitate system-to-system interaction
- Including optional aggregation (for example, association of products to cases) in addition to pedigree information in the transmission envelope

6 Certificates and Digital Signatures

6.1 Certificates

A certificate is a data structure that is used to bind a public key and a subject (e.g., person, server, or device). Confidence in this binding is essential to be able to rely on the keys and cryptographic results (i.e., encrypted data, digitally signed data).

ITU-T X.509 (or ISO|IEC 9594-8) [11] is a standard that defines a certificate format, an extensibility mechanism, and a set of certificate extensions. ITU-T X.509 is broad in the definition of allowable fields and applicability. X.509 certificates SHALL be used for the Electronic Pedigree.

6.2 Certificate Profile

A certificate profile defines elements and extensions that are required or optional in order to specify interoperable implementation and use. The EPCglobal Certificate Profile [8] defines a profile of X.509 certificate issuance and usage by entities in the EPCglobal network and is based on work done in the IETF, specifically IETF RFC 3280 – Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile [9] and IETF RFC 3279 - Algorithms and Identifiers for the Internet X.509 PKI Certificate and CRL Profile [10]. RFC 3280 profiles the format and semantics of certificates and certificate revocation lists (CRLs) for the Internet PKI, and is itself a profile of the ITU X.509 [11] standard. RFC 3279 defines algorithm identifiers and ASN.1 encoding formats for digital signatures and subject public keys used in Internet PKI. These IETF documents have been well implemented, deployed, and tested in many existing environments.

The electronic pedigree digital signature processes SHALL conform to the X.509 certificate profile defined in the EPCglobal Certificate Profile version 1.0.

To rely on digital signatures, it is important to identify the subject of the certificate uniquely. Certain attributes SHALL be included in the certificate to provide this uniqueness. Users are generally identified by attributes such as Name, Organizational Affiliation and email address. To ensure users can be uniquely identified when digitally signing electronic pedigrees, user certificates SHALL include the minimum attributes specified in Section 3.2.1 of the EPCglobal Certificate Profile version 1.0, and SHALL also include the user’s RFC822 email address in conformance with the profile requirements. Servers are typically identified by their Fully Qualified Domain Name (FQDN). To ensure servers can be uniquely identified when digitally signing electronic pedigrees, server certificates SHALL include the server’s FQDN in conformance with the profile requirements.

6.3 Digital Signatures

A method to create and represent digital signatures using XML is specified in *W3C XML-Signature Syntax and Processing* [14]. Refer to http://www.w3.org/TR/xmldsig-core for a complete description. The Electronic Pedigree Interchange Format uses this standard to represent a digital signature.

The application of digital signatures to electronic pedigrees SHALL conform to the following requirements and the Certificate Profile referenced above.

Signature Method: The regulations require the use of FIPS 186-2. The RSA algorithm [2, 14] SHALL be supported. Refer to http://www.w3.org/2000/09/xmldsig#rsa-sha1 for a complete description.

Digest Method: The regulations require the use of FIPS 186-2. SHA1 [2, 14] algorithm SHALL be supported. Refer to http://www.w3.org/2000/09/xmldsig#sha1 for a complete description.

Canonicalization Method and Transforms: The Exclusive C14N XML Canonicalization method, without comments, described in W3C Exclusive XML Canonicalization Version 1.0 [12] SHALL be employed for both canonicalization and transforms. The following canonicalization interoperability methods, based on work performed by the Web Services-Interoperability (WS-I) Organization in the Basic Security Profile Version v1.0 draft [13], SHALL be employed.

- The canonicalization method and transform elements MAY contain inclusive namespaces with a PrefixList attribute. If present, pedigree systems SHALL use the PrefixList in the manner described below.
- The inclusive namespaces PrefixList attribute SHALL support prefixes in any order within the string.
- The inclusive namespaces PrefixList attribute SHALL support arbitrary whitespace before, after and between the prefixes within the string.
- Any inclusive namespaces SHALL contain the prefix of all namespaces that are in-scope and desired to be protected, but not visibly utilized, for the element being signed and its descendants.
- Any inclusive namespaces SHALL contain the string "#default" if a default namespace is in-scope and desired to be protected, but not visibly utilized, for the element being signed and its descendants.

Reference: The same-document URI reference SHALL be employed. See section 4.3.3.3 of http://www.w3.org/TR/xmldsig-core. XPATH SHALL NOT be used for specifying references.

KeyInfo: The KeyInfo element SHALL be present in the Signature element. The use of X.509 SHALL be employed. The KeyInfo element SHALL include one and only one X509Data element and it SHALL be the only element supported. The KeyInfo element MAY include other unsupported elements. The single X509Data element SHALL include one and only one X509IssuerSerial element that identifies the signer's certificate and one X509Certificate element that contains the signer's certificate. The X509Data

element MAY include additional X509Certificate elements containing other certificates in the signer's certificate chain. Refer to sections 4.4 and 4.4.4 of http://www.w3.org/TR/xmldsig-core for more information [14].

The validation of digital signatures applied to electronic pedigrees SHALL conform to the following requirements and the Certificate Profile referenced above.

Core Validation: The core validation method described in section 3.2 of http://www.w3.org/TR/xmldsig-core SHALL be employed [14].

Signer Certificate Validation: The signer's certificate SHALL be validated in accordance with section 6 of RFC 3280 [9].

6.3.1 Examples (non-normative)

The following XML examples illustrate the application of the digital signature requirements described above when digitally signing pedigrees. The examples illustrate signing of the same shippedPedigree layer, without using inclusive namespaces in Transforms and with using inclusive namespaces in Transforms.

The SignedInfo block element contains the CanonicalizationMethod, SignatureMethod, Reference, Transforms and DigestMethod elements described above.

The KeyInfo block element contains the X509Data, X509IssuerSerial and X509Certificate elements described above.

6.3.1.1 Example without Inclusive Namespaces in Transforms

The following example illustrates what the Signature element might look like when inclusive namespaces are not included in Transforms. Refer to the `Signature:Reference:Transforms` element. The inclusion of inclusive namespaces is not required, but may be optionally used.

```
<pedigree xmlns="urn:epcGlobal:Pedigree:xsd:1">
  <shippedPedigree id="id8c53d861-e66a-40df-8a46-a0c151b8ea35">
      ...
  </shippedPedigree>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"></CanonicalizationMethod>
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"></SignatureMethod>
      <Reference URI="#id8c53d861-e66a-40df-8a46-a0c151b8ea35">
        <Transforms>
            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"></Transform>
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod>
        <DigestValue>sMeAUPpiqzd8nGzUtr9I04EE7Ug=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>NKn7NBpM9a8RgoyZ0Sc7c/G0EGO5i1mgASIxpSxqjxS7fM8L0eBNm7nnt0ET8bM3</SignatureValue>
    <KeyInfo>
      <X509Data>
        <X509IssuerSerial>
            <X509IssuerName>O=issuername,C=us</X509IssuerName>
            <X509SerialNumber>1144946731</X509SerialNumber>
        </X509IssuerSerial>
        <X509Certificate>MIIDHTCCAoagAwIBAgIERD6AKzANBgkqhkiG9w0BAQUFADAjMQswC</X509Certificate>
      </X509Data>
    </KeyInfo>
  </Signature>
</pedigree>
```

6.3.1.2 Example with Inclusive Namespaces in Transforms

The following example illustrates what the Signature element might look like when inclusive namespaces are included in Transforms. Refer to the `Signature:Reference:Transforms` element. The inclusion of inclusive namespaces is not required, but may be optionally used. This is considered a valid signature even if it contains redundant information about the namespace prefix 'ped' as it is preserved by simple exclusive canonicalization. The use of inclusive namespaces must be factored when comparing message digests during the pedigree authentication process.

```
<ped:pedigree xmlns:ped="urn:epcGlobal:Pedigree:xsd:1">
  <ped:shippedPedigree id="id8c53d861-e66a-40df-8a46-a0c151b8ea35">
      ...
  </ped:shippedPedigree>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/></CanonicalizationMethod>
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"></SignatureMethod>
      <Reference URI="#id8c53d861-e66a-40df-8a46-a0c151b8ea35">
        <Transforms>
            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
                <InclusiveNamespaces PrefixList="ped" xmlns="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            </Transform>
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod>
        <DigestValue>sMeAUPpiqzd8nGzUtr9I04EE7Ug=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>NKn7NBpM9a8RgoyZ0Sc7c/G0EGO5i1mgASIxpSxqjxS7fM8L0eBNm7nnt0ET8bM3</SignatureValue>
    <KeyInfo>
      <X509Data>
        <X509IssuerSerial>
            <X509IssuerName>O=issuername,C=us</X509IssuerName>
            <X509SerialNumber>1144946731</X509SerialNumber>
        </X509IssuerSerial>
        <X509Certificate>MIIDHTCCAoagAwIBAgIERD6AKzANBgkqhkiG9w0BAQUFADAjMQswC</X509Certificate>
      </X509Data>
        </KeyInfo>
    </Signature>
</ped:pedigree>
```

7 Support for Alternate ePedigree Formats

Current legislation in multiple US states dictates the creation and updating of electronic pedigrees at each stop in the pharmaceutical supply chain. Each law specifies the data content of the electronic pedigree and the digital signature standards but none of them specifies the actual format of the document. The need for a standard electronic document format that can be updated by each supply chain participant is what has driven our efforts. However, there is recognition that not all members of the pharmaceutical supply chain will adopt the XML Schemas that are detailed in this document. Toward that end, the ePedigree schema includes elements that allow the attachment of other document types as MIME documents. This could include scanned images, PDF documents or X.12 documents. Electronic pedigree documents stored in some structured alternate formats will require specialized software to allow them to be auto-authenticated. Some formats, such as PDF's and scanned images, will always require manual authentication.

8 Pedigree Transfer

This Standard does not identify exactly how pedigree documents must be transferred between trading partners. Any mechanism chosen must provide document immutability, non-repudiation and must be secure and authenticated.

- The ePedigree SHALL be captured and delivered as a single immutable document.
- The transfer of ePedigree documents SHALL conform to a push based transfer.

- The transfer SHALL occur via secure and, if transmitted over the public internet, authenticated mechanisms.
- If transmitted over the public internet, the transfer SHALL meet the expectation of Non-Repudiation.

Secure transmission relies on the recommendations for securing pedigree transmissions defined by the HLS Information Work Group C [6].

9 Schema Versioning

9.1 Pre-Standard Version Identification

There is a specified technique for identifying the use of this specific pedigree schema version when creating or updating production pedigree documents prior to its adoption as a standard by EPCglobal or other body. The "version" field of the documentInfo elements SHALL contain the date string that reflects the date of the schema XSD file used. The string SHALL be composed like this:

CCYYMMDD

Where CC is the century, "20" for example

YY is the year, "06" for example

MM is the month, "02" for example

DD is the day of the month, "05" for example.

The date string for the example pre-standard schema version would be "20060205".

The same technique SHALL be used to identify the use of this specific pedigree envelope schema version when creating production pedigree envelope documents prior to its adoption as a standard by EPCglobal or other body. The "version" field of the pedigreeEnvelope element SHALL contain the date string that reflects the date of the schema XSD file used.

9.2 Post-Standard Version Identification

Once the pedigree and pedigree envelope schemas are adopted as a standard, the namespace SHALL be used to identify the major version. The "version" field of the documentInfo elements SHALL be used to identify minor releases between major versions.

9.3 Recommendation for Managing Major Version of Schemas

The following is a recommended approach for managing the major version of the pedigree and pedigree envelope schemas.

9.3.1 Pedigree Schema

Supporting prior versions of the pedigree schema could be accomplished by using a new group type that holds a choice of all possible major versions of the pedigree schema. The new group type would be used where all pedigree elements are currently referenced:

- ShippedPedigreeType

- ReceivedPedigreeType
- UsnignedReceivedPedigreeType
- PreviousPedigreeType.

This approach ensures that all pedigrees created will be of the same major version of the pedigree schema, while supporting nesting of previous pedigrees that could be of different versions. The following is an example of would be added to the pedigree schema document.

The opening schema element would reference the namespaces for all supported versions of the schema.

```
<xs:schema xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ped="urn:epcGlobal:Pedigree:xsd:1.1" xmlns:ped1="urn:epcGlobal:Pedigree:xsd:1"
targetNamespace="urn:epcGlobal:Pedigree:xsd:1.1" elementFormDefault="qualified"
attributeFormDefault="unqualified">
```

The pedigree schema would import all previous pedigree versions.

```
    <xs:import namespace="urn:epcGlobal:Pedigree:xsd:1" schemaLocation="PedigreeSchema_20061030.xsd"/>
```

The new group element `prevPedigreeLayer` would include a choice of all possible major versions of the pedigree schema.

```
    <xs:group name="prevPedigreeLayer ">
        <xs:choice>
            <xs:element ref="ped:pedigree"/>
            <xs:element ref="ped1:pedigree"/>
        </xs:choice>
    </xs:group>
```

The `prevPedigreeLayer` element would be referenced where all pedigree elements are currently referenced so that the different versions can be supported.

```
    <xs:complexType name="ShippedPedigreeType">
        <xs:sequence>
            <xs:element name="documentInfo" type="ped:DocumentInfoType"/>
            <xs:choice>
                <xs:element ref="ped:initialPedigree"/>
                <xs:element ref="ped:repackagedPedigree"/>
                <xs:element ref="ped:unsignedReceivedPedigree"/>
                <xs:group ref="ped:prevPedigreeLayer"/>
            </xs:choice>
            <xs:element name="itemInfo" type="ped:ItemInfoType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="transactionInfo" type="ped:TransactionInfoType"/>
            <xs:element name="signatureInfo" type="ped:SignatureInfoType"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>

    <xs:complexType name="ReceivedPedigreeType">
        <xs:sequence>
            <xs:element name="documentInfo" type="ped:DocumentInfoType"/>
            <xs:choice>
                <xs:group ref="ped:prevPedigreeLayer"/>
                <xs:element ref="ped:initialPedigree"/>
            </xs:choice>
            <xs:element name="receivingInfo" type="ped:ReceivingInfoType"/>
            <xs:element name="signatureInfo" type="ped:SignatureInfoType"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
```

```
    </xs:complexType>

    <xs:complexType name="UnsignedReceivedPedigreeType">
        <xs:sequence>
            <xs:element name="documentInfo" type="ped:DocumentInfoType"/>
            <xs:choice>
                <xs:group ref="ped:prevPedigreeLayer"/>
                <xs:element ref="ped:repackagedPedigree"/>
                <xs:element ref="ped:initialPedigree"/>
            </xs:choice>
            <xs:element name="transactionInfo" type="ped:TransactionInfoType" minOccurs="0"/>
            <xs:element name="receivingInfo" type="ped:ReceivingInfoType"/>
            <xs:element name="attachment" type="ped:ForeignDataType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>

    <xs:complexType name="PreviousPedigreeType">
        <xs:choice>
            <xs:group ref="ped:prevPedigreeLayer"/>
            <xs:element ref="ped:initialPedigree"/>
            <xs:element ref="ped:altPedigree"/>
        </xs:choice>
    </xs:complexType>
```

9.3.2 Pedigree Envelope Schema

Supporting prior versions of pedigree documents in the pedigree envelope schema could be accomplished by providing the full list of namespaces for all supported versions of the pedigree schema in the namespace attribute of the `<xs:any>` element in the `PedigreeEnvelopeType`. The following is an example of would be added to the pedigree schema document.

The opening schema element would reference the namespaces for all supported versions of the schema.

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ped="urn:epcGlobal:Pedigree:xsd:1"
xmlns:pedenv="urn:epcGlobal:PedigreeEnvelope:xsd:1.1" xmlns:ped1="urn:epcGlobal:Pedigree:xsd:1.1"
targetNamespace="urn:epcGlobal:PedigreeEnvelope:xsd:1.1" elementFormDefault="qualified"
attributeFormDefault="unqualified">
```

The pedigree envelope schema would import the current and all previous pedigree versions.

```
    <xs:import namespace="urn:epcGlobal:Pedigree:xsd:1" schemaLocation="PedigreeSchema_20061030.xsd"/>
    <xs:import namespace="urn:epcGlobal:Pedigree:xsd:1.1" schemaLocation="PedigreeSchema_YYYYMMDD.xsd"/>
```

The `PedigreeEnvelopeType` can include all supported versions of the pedigree schema in the `<xs:any>` `namespace` attribute.

```
    <xs:complexType name="PedigreeEnvelopeType">
        <xs:complexContent>
            <xs:extension base="pedenv:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="version" type="xs:string" minOccurs="0"/>
                    <xs:element name="serialNumber" type="xs:string"/>
                    <xs:element name="date" type="xs:date" minOccurs="0"/>
                    <xs:element name="sourceRoutingCode" type="xs:string" minOccurs="0"/>
                    <xs:element name="destinationRoutingCode" type="xs:string" minOccurs="0"/>
                    <xs:element name="container" type="pedenv:ContainerType" minOccurs="0"
                        maxOccurs="unbounded"/>
                    <xs:any namespace="urn:epcGlobal:Pedigree:xsd:1 urn:epcGlobal:Pedigree:xsd:1"
                        processContents="skip" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
```

```
</xs:complexType>
```

9.4 Backward Compatibility with Pre-Standard Version

The Pedigree schema that is adopted as a standard SHALL be backwards compatible with the pre-standard interim version of the Pedigree schema. The Pedigree schema that is adopted as a standard SHALL support nesting of pedigrees created with the pre-standard interim version of the Pedigree schema within one of the pedigree LayerTypes and SHALL maintain the integrity of the digitally signed content of the interim pedigree layers.

Pedigrees and pedigree envelopes created using this Standard (after the interim version) SHALL express pedigree and envelope serial numbers using the UUID in URN format using an xs:string element per RFC 4122. Implementations SHALL support pedigree and pedigree envelope serial numbers with and without the "urn:uuid" namespace prefix to preserve backwards compatibility to pedigrees generated prior to the release of the Standard. The schema version can be used as an indicator to determine which version of the schema was used to apply a serial number.

9.5 Backward and Forward Compatibility of Pedigrees with New Schema Versions

9.5.1 Backward Compatibility of Pedigrees between Minor Versions of Pedigrees within a Major Version of Pedigree Schema

Pedigree implementations SHALL support nesting of pedigrees with an older minor version of the Pedigree schema within a new pedigree LayerType that is of a newer minor version of the Pedigree schema, within the same major version of the Pedigree schema. For example, within version 1 of the Pedigree schema, an older minor version (20060526) could be wrapped by a newer minor version (20061103).

9.5.2 Backward Compatibility of Pedigrees between Two Major Versions of Pedigree Schema

Pedigree implementations SHALL support nesting of pedigrees with an older major version of the Pedigree schema within a new pedigree LayerType that is of a newer major version of the Pedigree schema. For example, an older major version (version 2) could be wrapped by a newer major version (version 1).

9.5.3 Forward Compatibility of Pedigrees between Minor Versions of Pedigrees within a Major Version of Pedigree Schema

Pedigree implementations SHALL support nesting of pedigrees with a newer minor version of the Pedigree schema within a new pedigree LayerType that is of an older minor version of the Pedigree schema, within the same major version of the Pedigree schema. For example, within version 1 of the Pedigree schema, a newer minor version (20061103) could be wrapped by an older minor version (20060526).

When an older version of a pedigree wraps a newer version of a pedigree, the pedigree will fail schema validation against the older schema version because the older version of the schema is not aware of the new optional elements present in the newer version of the schema. This is expected behavior that impacts schema validation only.

9.5.4 Forward Compatibility of Pedigrees between Two Major Versions of Pedigree Schema

Pedigree management software implementations cannot be aware of major changes that will occur to the pedigree schema in the future. For that reason it is impossible to expect implementations to be forward compatible with future major versions. The following requirement stems from this fact of nature.

Pedigree implementations SHALL NOT allow nesting of new layers to pedigrees that already contain layers that use a newer major version of the Pedigree schema when the new pedigree layer would use an older major version of the Pedigree schema. For example, a newer major version (version 2) could not be wrapped by an older major version (version 1).

This leads to an industry deployment problem. If a trading partner who has not yet updated their software receives a pedigree from an upstream partner who has updated their software to support the newer major version, they will not be able to process the pedigree because their software will not understand the newer schema. To solve this problem the ***industry*** must agree on the following deployment rules:

Whenever a new major version of the pedigree schema is adopted it *cannot be used on any pedigree layer until all trading partners in the entire supply chain have had sufficient time to update their software to support that new version*. To ensure that this occurs, the EPCglobal work group that creates the new major version schema must be responsible for estimating the effort for vendors to implement the changes and then for ***all trading partners*** to install the upgrade throughout their networks. The EPCglobal work group shall solicit input from software vendors and supply chain participants as part of the estimating process. At the time the major new version is adopted the EPCglobal work group shall publish the date based on this estimate as the first date that trading partners SHALL be able to receive pedigrees based on the new major schema version. Trading partners SHALL NOT create pedigrees based on the new major schema version prior to that date even if their software is fully upgraded early. Users would have that much time to acquire and install upgraded systems to prepare to send and receive pedigrees based on the new major schema version.

10 ePedigree Data Definition

Conforming implementations SHALL use the pedigree schema for all input and output documents.

All schema elements marked as “Yes” in the “Mandatory” column in the tables below SHALL be included in all pedigree documents. The value supplied in these elements SHALL be non-null.

All schema elements marked as “Conditional” in the “Mandatory” column in the tables below SHALL be included in pedigree documents that are subject to the situations that require them. The value supplied in these elements SHALL be non-null, unless the nillable attribute is explicitly set in the element.

All schema elements marked as “No” in the “Mandatory” column in the tables below SHALL be optional in pedigree documents.

Conforming implementations SHALL offer the Pedigree Envelope schema as an optional way to package one or more pedigree documents for transmission.

All schema elements on incoming pedigrees and pedigree envelopes, regardless of marking, SHALL be accepted and handled properly. "Handled properly" CAN be interpreted as "ignored" for elements not marked as "Mandatory" or "Conditional".

The schemas for the Pedigree and Pedigree Envelope are separate schemas and SHALL be versioned independently of each other. The Pedigree Envelope MAY reference pedigrees that are of a different version than the version of the Pedigree Envelope.

Enumerated lists included in the schema SHALL use the UpperCamelCase capitalization style for new items added to enumerated lists (e.g., ReceivedAndAuthenticated). If an enumeration is an acronym or contains an acronym, the acronym SHALL be specified in all uppercase (e.g., GLN).

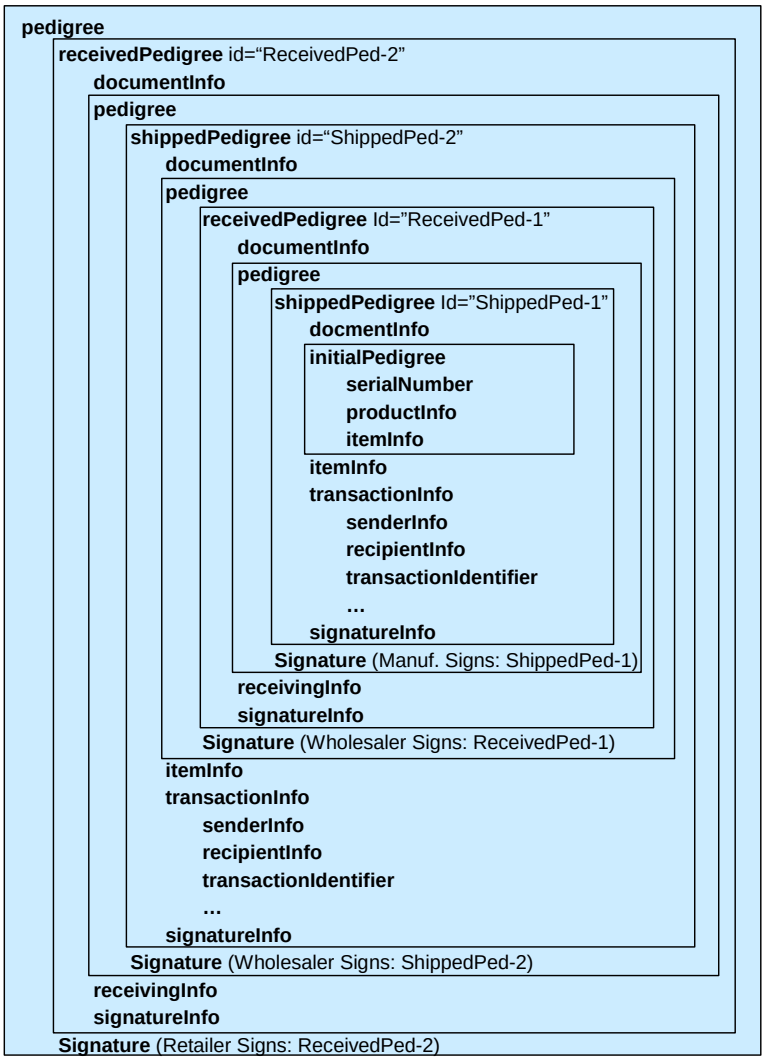
10.1 Character Set Use for Pedigree and Pedigree Envelope XML Documents (non-normative)

It is strongly recommended that pedigree implementations use the UTF-8 character set when generating pedigree and pedigree envelope XML documents so as to avoid interoperability issues from incompatible character sets. Refer to "Encode your XML documents in UTF-8" [18] for an explanation of the general issue.

The intent is to make UTF-8 a normative requirement in the next version of the standard.

10.2 Electronic Pedigree Format

The basic components of a pedigree are shown in the following figure. The components in an actual pedigree depend on the specific business situation in which it is used (e.g., pedigree initiated by manufacturer, pedigree initiated by wholesaler, pedigree for repackaged item, etc.).



The innermost component of the pedigree SHALL always be contained in an `initalPedgiree` or a `repackagedPedigree` element. New pedigrees created by manufacturers or wholesalers for standard (non-repacked) prescription drug products SHALL always be started in an `intialPedigree` element. Pedigrees for repackaged or kitted products (where the kit has its own NDC) SHALL always be started in a `repackagedPedigree` element.

The `pedigree` element is a wrapper element that contains a signed pedigree representing the prior chain of custody for an item. Information associated with each stage in a pedigree transaction in which ownership of an item passes from one supply chain partner to another SHALL be added to the pedigree in its own layer and then digitally signed by that supply chain partner.

The `shippedPedigree` and `receivedPedigree` elements represent a stage (shipping or receiving) in a pedigree exchange transaction in which ownership of a product passes from one supply chain partner to another. These elements SHALL be used to wrap the preexisting pedigree for a product. Signatures SHALL be applied over `shippedPedigree` or `receivedPedigree` elements, signing over new content added to the pedigree and any prior pedigree content from previous transactions.

At any given time, the outermost `pedigree` element SHALL contain a `shippedPedigree` and a `Signature` element, or a `receivedPedigree` and a `Signature` element. As each transaction occurs, the

676 preexisting pedigree for an item SHALL be wrapped inside of a `pedigree` layer element. These
677 successive layers represent the entire chain of ownership and the product description.

678 10.2.1 Forms for Pedigree Components for Specific Business 679 Situations (non-normative)

680 A pedigree and data components within the pedigree may take one of several forms depending on
681 the context of how the pedigree was created or received (e.g., manufacturer initiated pedigree,
682 wholesaler initiated pedigree, pedigree for repackaged item, conversion of alternate pedigree, etc.).
683 The table below provides a non exhaustive list of use cases and the corresponding form a pedigree
684 component may take for each of these use cases.

685 10.2.1.1 Initial Pedigree Components

686 The following diagrams illustrate the different forms the innermost content of the pedigree may
687 take before the content is nested in the first `shippedPedigree` layer. These components do not
688 represent complete shipped and received pedigrees. In order to represent a complete pedigree, the
689 innermost content is embedded in a `shippedPedigree` and digitally signed with a `Signature` element.

Form	Example
Innermost content for a manufacturer pedigree (initiated by manufacturer, before a wholesale distribution)	**initialPedigree** **serialNumber** **productInfo** **drugName** **manufacturer** **…** **itemInfo** **lot** **quantity**
Innermost content for a wholesaler pedigree (initiated by first wholesaler, includes transaction information for first wholesale distribution)	**initialPedigree** **serialNumber** **productInfo** **drugName** **manufacturer** **…** **itemInfo** **lot** **quantity** **transactionInfo** **senderInfo** **recipientInfo** **transactionIdentifier** **…** **receivingInfo** **dateReceived**

Form	Example
Innermost content for a wholesaler pedigree with attachment (initiated by wholesaler, includes ASN data as attachment to facilitate manual authentication by downstream trading partners)	**initialPedigree** **serialNumber** **productInfo** **drugName** **manufacturer** **…** **itemInfo** **lot** **quantity** **transactionInfo** **senderInfo** **recipientInfo** **transactionIdentifier** **…** **receivingInfo** **dateReceived** **attachment** **mimeType** **encoding** **data**
Innermost content for a wholesaler pedigree with scanned source pedigree (initiated by wholesaler, includes previous pedigree which may reflect one or more previous distributions)	**initialPedigree** **serialNumber** **productInfo** **drugName** **manufacturer** **…** **itemInfo** **lot** **quantity** **altPedigree** **mimeType** **encoding** **data** **serialNumber**

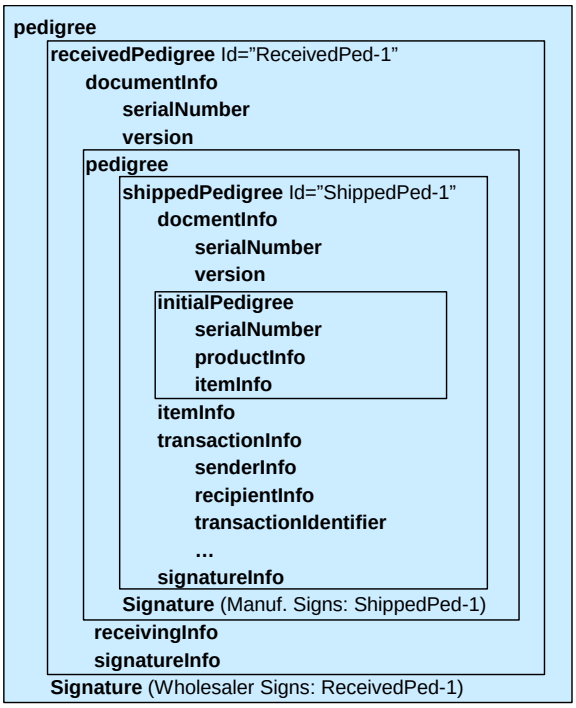
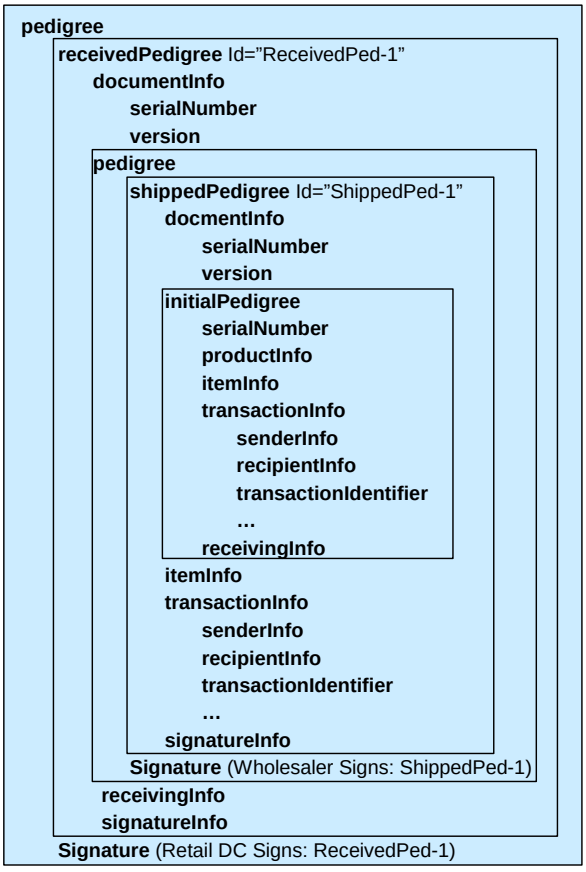
Form	Example
Innermost content for a repacker pedigree (initiated by repacker, repacked item contains two source pedigrees)	**repackagedPedigree** **previousProducts** **serialNumber** **previousProductInfo** **itemInfo** **contactInfo** **previousProducts** **serialNumber** **previousProductInfo** **itemInfo** **contactInfo** **previousPedigrees** **pedigree** **…** **previousPedigrees** **pedigree** **…** **productInfo** **drugName** **manufacturer** **…** **itemInfo** **lot** **quantity**

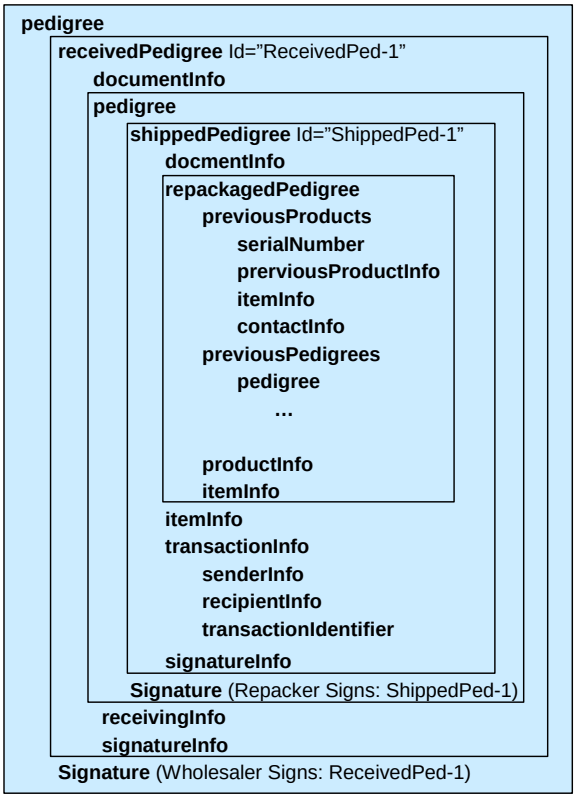
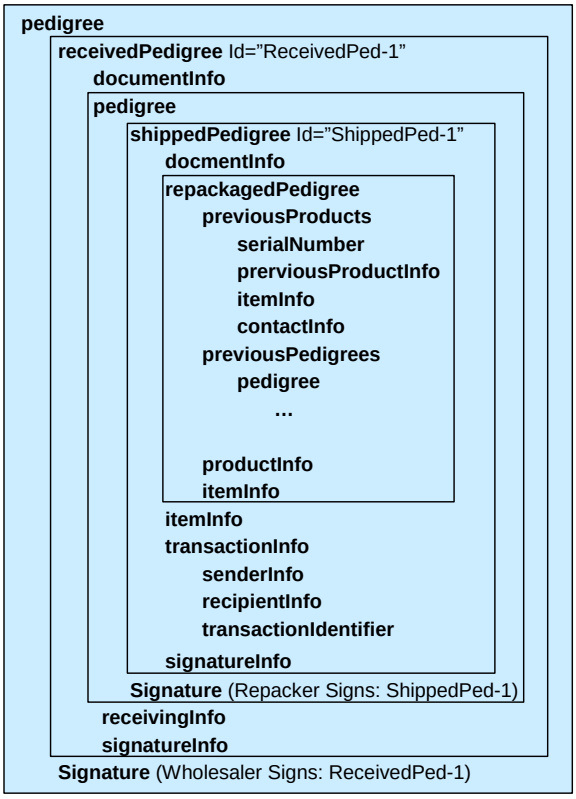
Form	Example
Innermost content for a kit pedigree where the kit has an assigned NDC (initiated by kit manufacturer, kit contains two pedigrees)	**repackagedPedigree** **previousProducts** **serialNumber** **previousProductInfo** **itemInfo** **contactInfo** **previousProducts** **serialNumber** **previousProductInfo** **itemInfo** **contactInfo** **previousPedigrees** **pedigree** **…** **previousPedigrees** **pedigree** **…** **productInfo** **drugName** **manufacturer** **…** **itemInfo** **lot** **quantity**

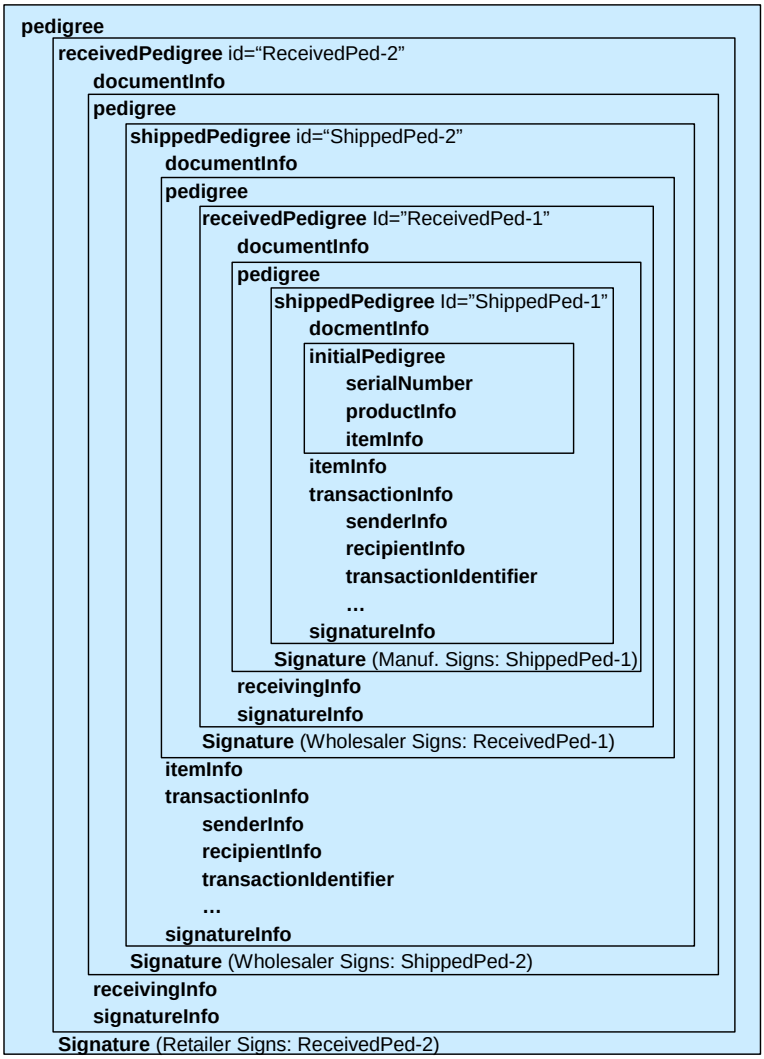
690

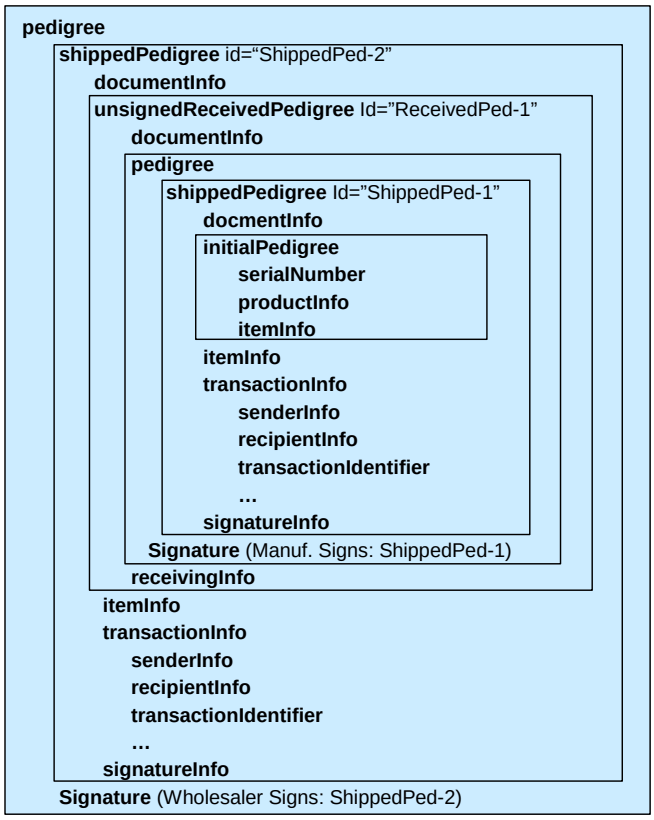
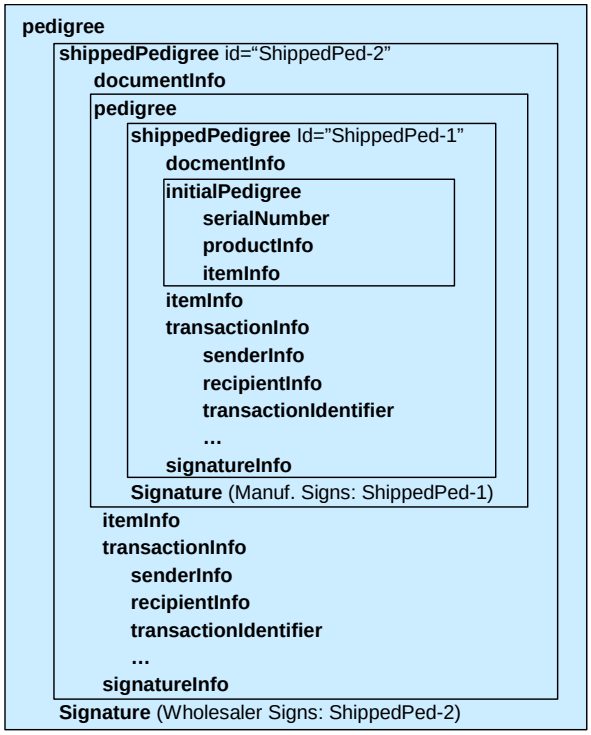
691 **10.2.1.2 Shipped and Received Pedigree Components**

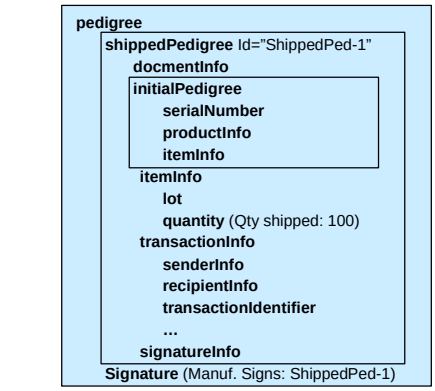
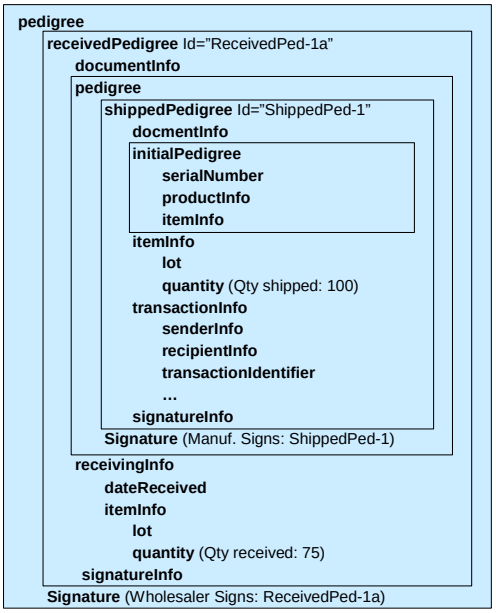
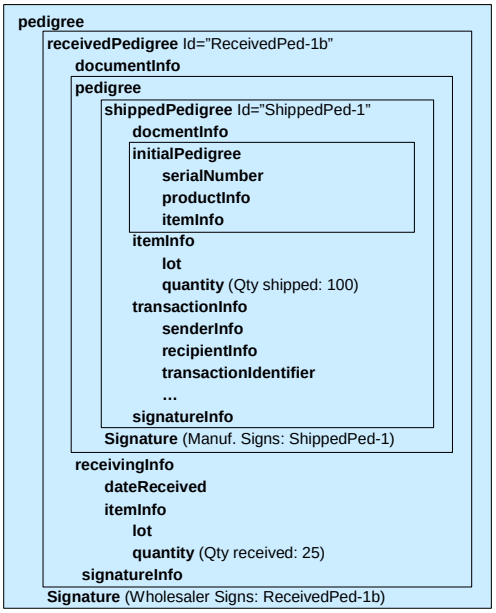
692 The following diagrams illustrate the different forms a complete pedigree may take when
693 pedigrees are exchanged between trading partners.

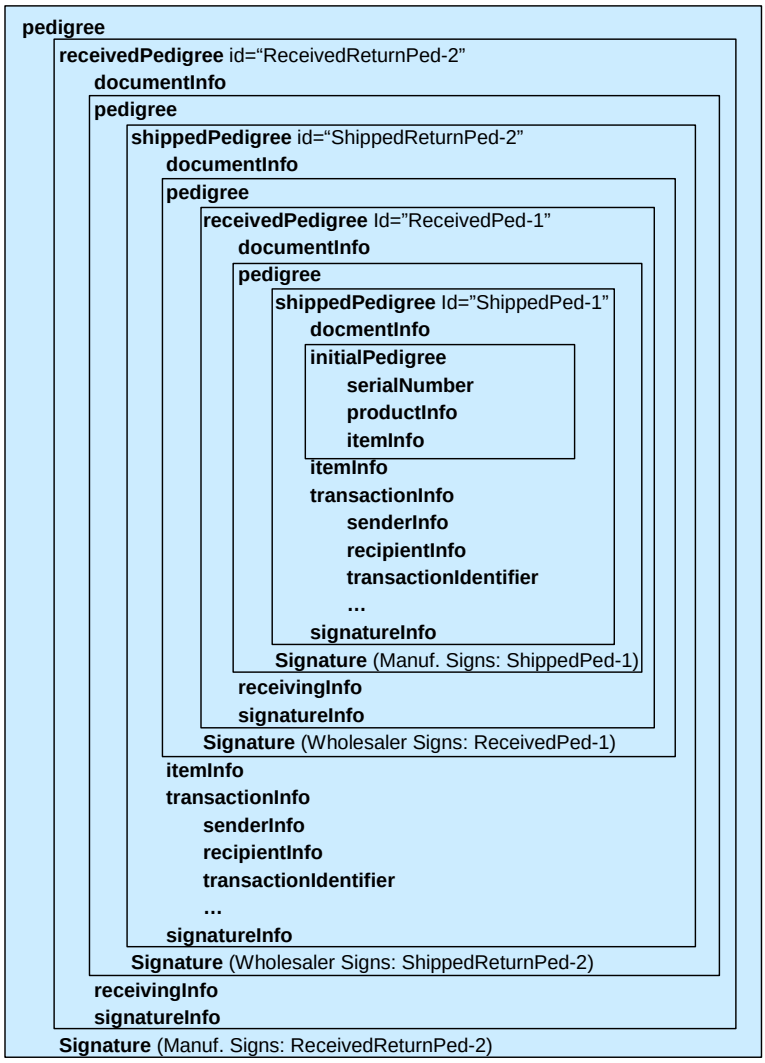
Form	Example
Signed manufacturer pedigree (initiated by manufacturer, after the wholesale distribution, signed by both manufacturer and wholesaler)	
Signed wholesaler pedigree (initiated by wholesaler, after the wholesale distribution, signed by both wholesaler and retailer DC)	

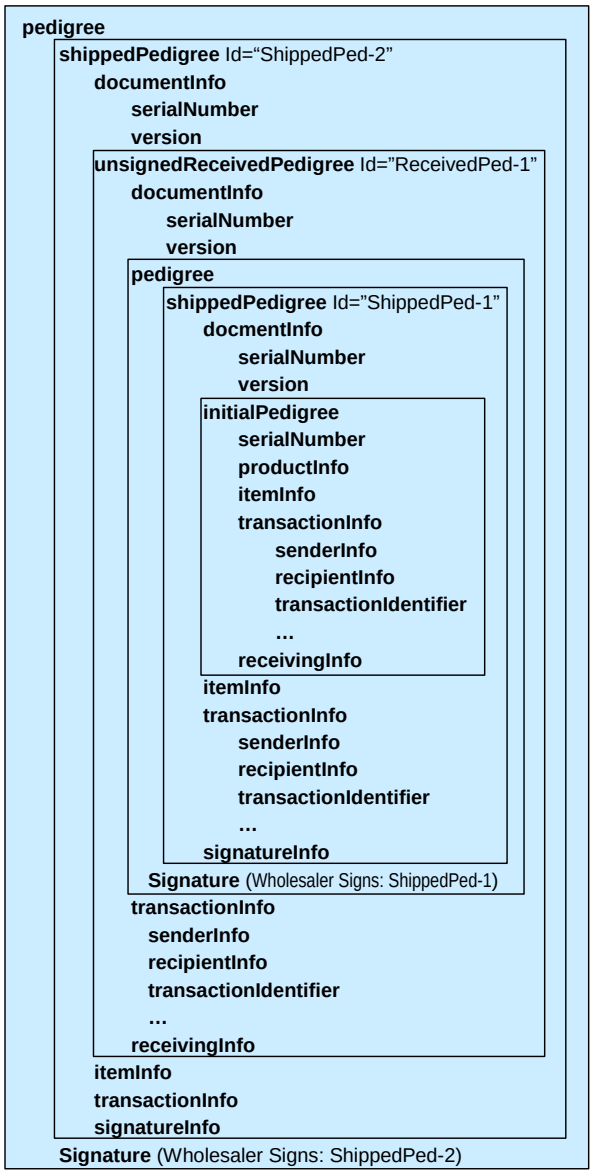
Form	Example
Signed repacker pedigree (initiated by repacker, after wholesale distribution, signed by both repacker and wholesaler recipient)	
Signed kit pedigree (kit has NDC, initiated by kit manufacturer, after wholesale distribution, signed by both kit manufacturer and wholesaler recipient)	

Form	Example
Pedigree with two signed transactions (initiated by manufacturer, received and signed inbound by wholesaler recipient, signed outbound by wholesaler upon shipment to pharmacy, received and signed inbound by pharmacy recipient)	

Form	Example
Pedigree without inbound receipt signature (initiated by manufacturer, received but not signed inbound by wholesaler recipient, signed outbound by wholesaler upon shipment to pharmacy)	
Pedigree without inbound receipt information or signature (initiated by manufacturer, signed outbound by wholesaler upon shipment to pharmacy)	

Form	Example
Pedigree with partial receipt (initiated by manufacturer, updated with partial receipt information and signed inbound by wholesaler recipient for first receipt, and then generation of another received pedigree with remaining receipt information and signature for second receipt)	 Original Pedigree Quantity shipped 100  First receipt for 75  Second receipt for 25

Form	Example
Pedigree with return transaction (initiated by manufacturer, received and signed inbound by wholesaler, return transaction applied by wholesaler for manufacturer return and signed outbound, received and signed inbound by manufacturer)	

Form	Example
Pedigree with return transaction applied by wholesaler on behalf of pharmacy (initiated by wholesaler, signed outbound by wholesaler for shipment to pharmacy, return transaction applied by wholesaler for pharmacy return, signed outbound by wholesaler for subsequent sale)	

10.2.2 XML Elements

XML Schema is used to specify the structure of the Electronic Pedigree Interchange Format. All data types prefixed with "xs" represent standard type definitions imported from the XML Schema specification.

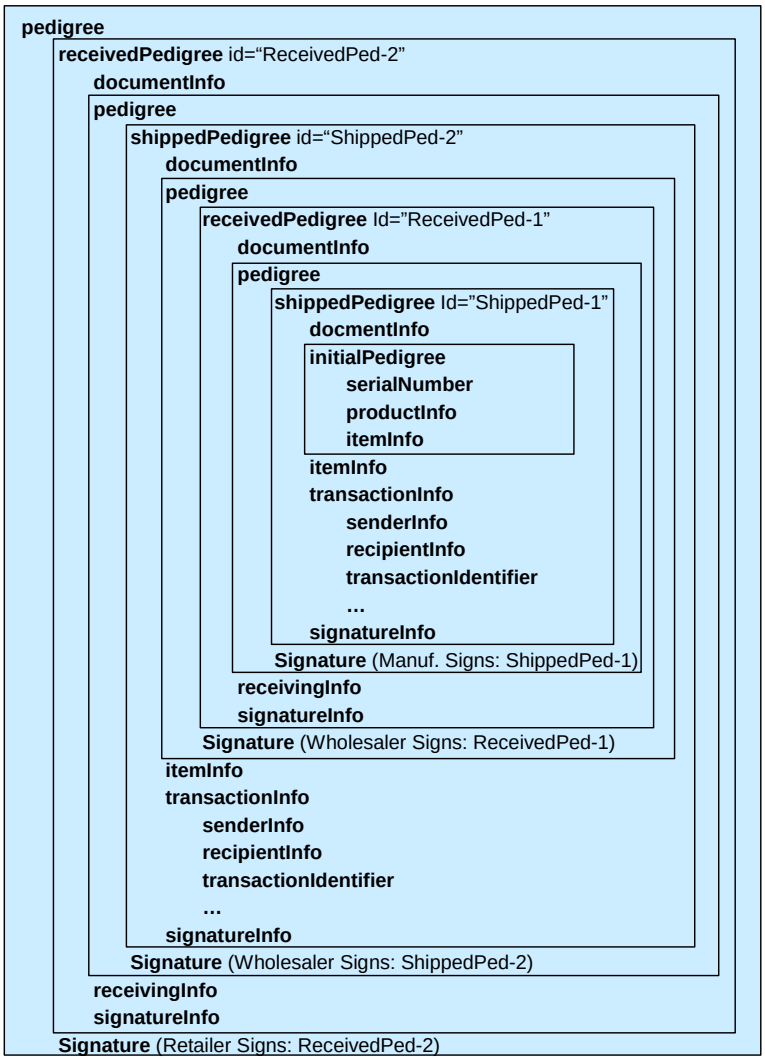
Carriage return and line feed characters SHALL NOT appear in any string elements. Unless otherwise specified there are no length or content restrictions on the elements of the message.

10.2.2.1 pedigree Element

The `pedigree` element is a wrapper element that contains a signed pedigree representing the prior chain of custody for an item. Information associated with each stage in a pedigree transaction in which ownership of an item passes from one supply chain partner to another SHALL be added to the pedigree in its own layer and then digitally signed by that supply chain partner.

The very innermost `shippedPedigree` layer SHALL contain the starting point for the pedigree. The starting point for the pedigree SHALL always be an `initialPedigree` or a `repackagedPedigree` element. The following table describes the different scenarios for how a pedigree CAN be started in its initial form, and the elements used to create that initial form.

Pedigree	Created by	Element Used to Express Initial form of Pedigree
Brand new pedigree for a product	Manufacturer or wholesaler	`initialPedigree`
Pedigree for a repackaged product	Repacker	`repackagedPedigree`
Pedigree for a kit	Kit manufacturer or wholesaler	`repackagedPedigree`
Pedigree that transforms a pedigree received in an alternate format (such as a scanned paper pedigree) into the EPCglobal pedigree format	Wholesaler	`initialPedigree`

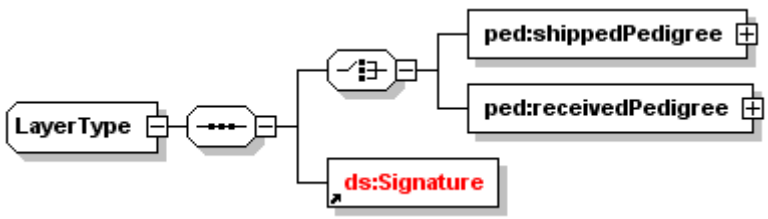


10.2.2.2 LayerType

The `LayerType` elements represent a stage (shipping or receiving) in a pedigree exchange transaction in which ownership of a product passes from one supply chain partner to another. The `LayerType` elements SHALL be used to wrap the preexisting pedigree for a product. Signatures SHALL be applied over `LayerType` elements, signing over new content added to the pedigree and any prior pedigree content from previous transactions.

At any given time, the outermost `pedigree` element SHALL contain a `shippedPedigree` and a `Signature` element, or a `receivedPedigree` and a `Signature` element.

diagram



used by elements **RepackagedPedigreeType/previousPedigrees** **pedigree**

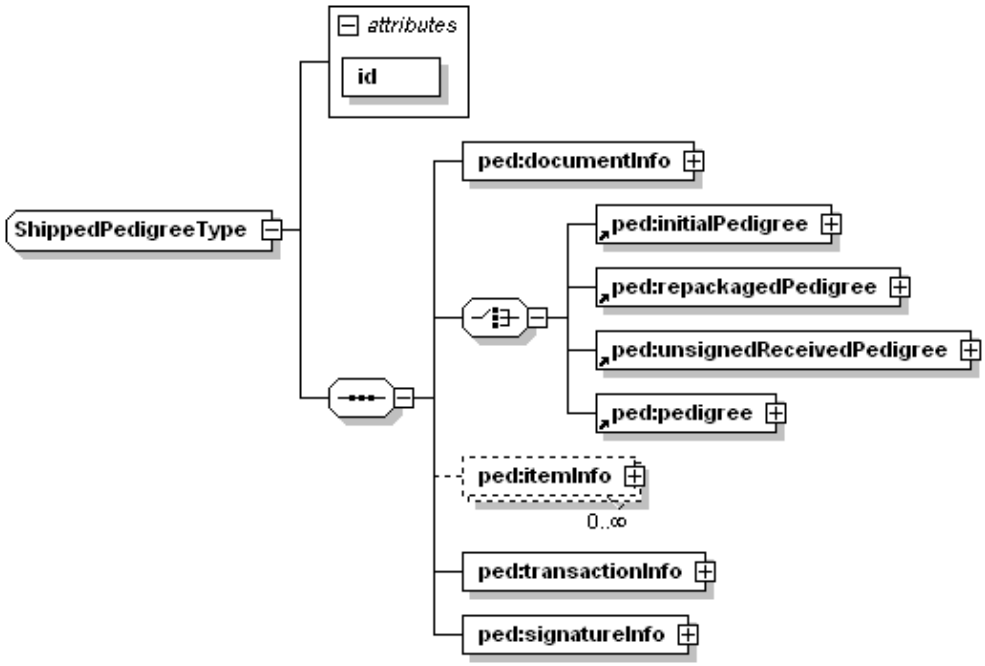
720

Name	Type	Mandatory?	Description
`shippedPedigree`	`ShippedPedigreeType`	Yes (Choice)	Information about a transaction in which ownership of the product passes from one supply chain partner to another. This layer element SHALL be added to the pedigree each time an exchange transaction occurs.
`receivedPedigree`	`ReceivedPedigreeType`	Yes (Choice)	Information about the receipt of products. This layer element SHALL be added to the pedigree each time a product is received as a result of an exchange transaction that requires that the pedigree be updated with the product receipt information and digitally signed.
`Signature`	`ds:signatureType`	Yes	Includes information about the digital signature, algorithms used, properties of the certificate, properties of the certificate issuer, and the message digest. This element SHALL be added to the pedigree to sign over new content added in the preceding layer (shippedPedigree or receivedPedigree) in an exchange transaction.

10.2.2.3 ShippedPedigreeType

The `ShippedPedigreeType` represents the shipping stage in a pedigree exchange transaction in which ownership of a product passes from one supply chain partner to another. The `ShippedPedigreeType` element SHALL wrap the preexisting pedigree (e.g., an initial pedigree, an initial pedigree for repackaged products, an unsigned received pedigree, or a pedigree received with a prior chain of custody) and adds information about the current transaction to the pedigree. The very innermost `shippedPedigree` layer SHALL contain the starting point for the pedigree.

diagram



used by element **LayerType/shippedPedigree**

Name	Type	Mandatory?	Description
`Id`	xs:id (Attribute)	Yes	A unique identifier for the pedigree layer within the pedigree document. This element is used to reference this element when it is signed.
`documentInfo`	`DocumentInfoType`	Yes	Contains an identifier for the pedigree and the version number of the pedigree schema used to create the pedigree.

Name	Type	Mandatory?	Description
initialPedigree	InitialPedigreeType	Yes (Choice)	Contains the initial pedigree information before the first outbound transaction. This element CAN also used to represent the conversion of a pedigree in another form to this pedigree format. A scanned or alternate representation of a pedigree MAY be included in this element to represent the prior chain of custody for the product.
repackagedPedigree	RepackagedPedigreeType	Yes (Choice)	Contains the initial pedigree information for a repackaged product before the first outbound transaction. This includes the product information for the repackaged item and pedigree information about the source items used to create the repackaged items.
pedigree	LayerType	Yes (Choice)	Wrapper element that contains a signed pedigree representing the prior chain of custody for a product.
unsignedReceivedPedigree	UnsignedReceivedPedigreeType	Yes (Choice)	Wrapper element that contains an unsigned received pedigree representing the prior chain of custody for a product.

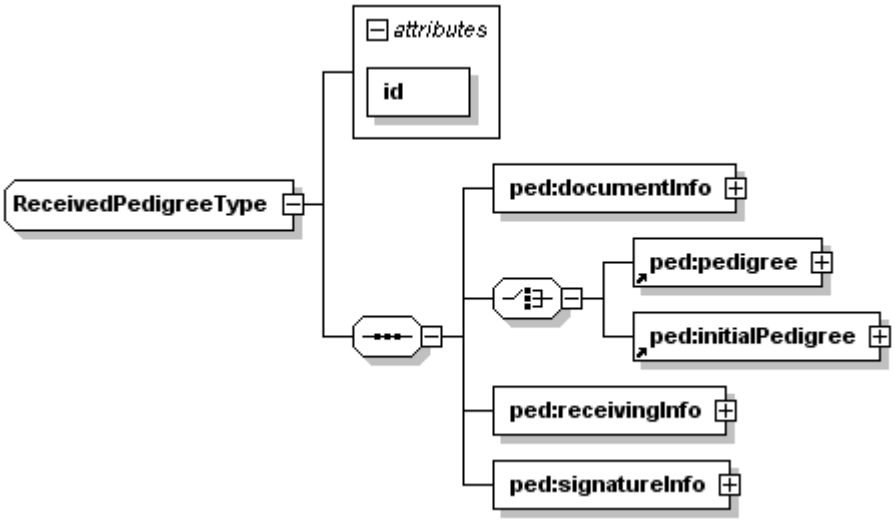
Name	Type	Mandatory?	Description
`itemInfo`	`ItemInfoType`	Conditional	Identifies the physical item(s) associated with a pedigree by lot number, expiration date, quantity of units, and item serial numbers if present. Pedigrees MAY represent quantities of one or more lots for the same product. This element MAY be repeated as many times as necessary to represent each lot number represented by the pedigree for the product. An `itemInfo` MAY be added each time a new transaction is added to the pedigree to record the items that are the subject of the transaction. If the items that are the subject of the transaction are the exact same items in the last transaction's `itemInfo`, then the element MAY be omitted from the current transaction.

Name	Type	Mandatory?	Description
`transactionInfo`	`TransactionInfoType`	Yes	Information about a transaction in which ownership of the product passes from one supply chain partner to another, including information about the sender, recipient, and transaction reference. This element SHALL be added to the pedigree each time an exchange transaction occurs.
`signatureInfo`	`SignatureInfoType`	Yes	Information that identifies the signer of a pedigree and the context of the signature (e.g., Certified).

730 #### 10.2.2.4 ReceivedPedigreeType

731 The `ReceivedPedigreeType` represents a signed receiving stage in a pedigree exchange transaction in
732 which ownership of a product passes from one supply chain partner to another. The
733 `ReceivedPedigreeType` element SHALL wrap the preexisting pedigree and adds information about the
734 receipt to the pedigree. This element SHALL be later signed.

diagram



used by element **LayerType/receivedPedigree**

735

Name	Type	Mandatory?	Description
`Id`	xs:id (Attribute)	Yes	A unique identifier for the pedigree layer within the pedigree document. This element is used to reference the element that will be signed.
`documentInfo`	`DocumentInfoType`	Yes	Contains an identifier for the pedigree and the version number of the pedigree schema used to create the pedigree.
`pedigree`	`Layertype`	Yes (Choice)	Wrapper element that contains a signed pedigree representing the prior chain of custody for a product.
`initialPedigree`	`InitialPedigreeType`	Yes (Choice)	Contains the initial pedigree information before the first outbound transaction. This element CAN also used to represent the conversion of a pedigree in another form to this pedigree format. A scanned or alternate representation of a pedigree MAY be included in this element to represent the prior chain of custody for the product.
`receivingInfo`	`ReceivingInfoType`	Yes	Information about the receipt of items associated with a pedigree.

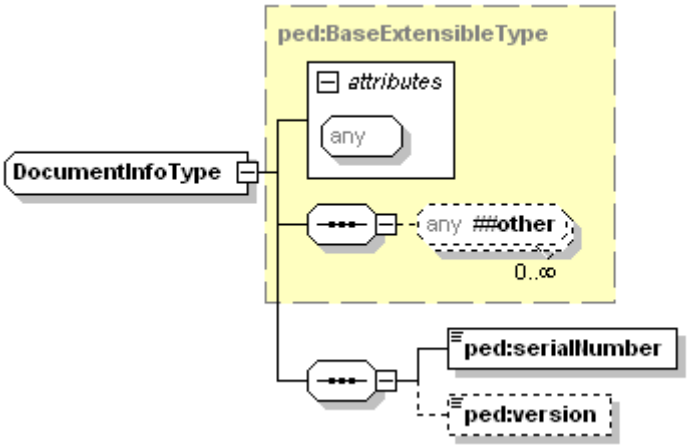
Name	Type	Mandatory?	Description
signatureInfo	SignatureInfoType	Yes	Information that identifies the signer of a pedigree and the context of the signature (e.g., Received and Authenticated).

10.2.2.5 DocumentInfoType

The DocumentInfoType contains the unique identifier for the pedigree and the version number of the pedigree schema used to create the pedigree.

The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

diagram



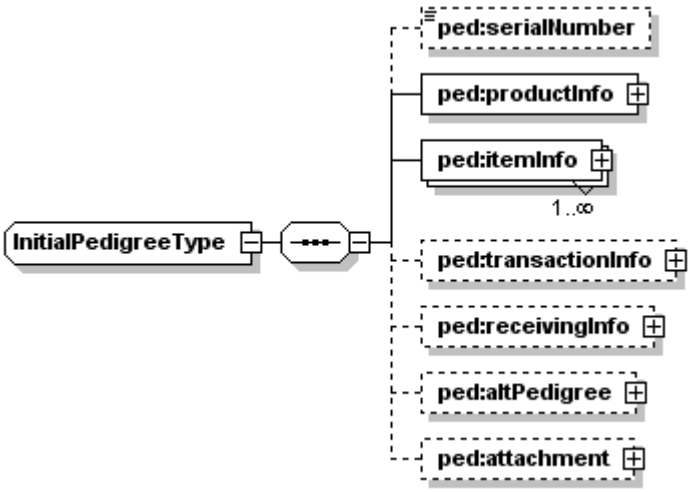
used by elements ShippedPedigreeType/documentInfo
ReceivedPedigreeType/documentInfo

Name	Type	Mandatory?	Description
`serialNumber`	xs:string	Yes	The globally unique identifier for the pedigree document, using the UUID 128-bit identifier per RFC 4122, including the "urn:uuid" namespace prefix. This identifier SHALL be used to reference the pedigree by external systems. A new serial number SHALL be assigned each time the pedigree is updated and signed (in new shippedPedigree or receivedPedigree layer).
`version`	xs:string	Yes	The version number of the pedigree schema used to create the pedigree layer.

10.2.2.6 InitialPedigreeType

The `InitialPedigreeType` contains the initial pedigree information before the first outbound transaction. This element CAN also used to represent the conversion of a pedigree in another form to this pedigree format. A scanned or alternate representation of a pedigree MAY be included in this element to represent the prior chain of custody for the product.

diagram



used by element **initialPedigree**

Name	Type	Mandatory?	Description
`serialNumber`	xs:string	Conditional	The globally unique identifier for the initial pedigree component, using the UUID 128-bit identifier per RFC 4122, including the "urn:uuid" namespace prefix. This element SHALL be inserted into new initial pedigrees when they are first created. This element SHALL be referenced in new `repackagedPedigree` elements when the `repackagedPedigree` references an `initialPedigree` for a previous product. This element is optional in the schema only to support backwards compatibility with the interim (pre-standard) version of the pedigree schema where it was not present. Software implementations SHALL accommodate pedigrees where this element is present and where it is not present.

`productInfo`	`ProductInfoType`	Yes	Information about the product to which the pedigree pertains, such as drug name, strength, dosage form, etc. This element SHALL be inserted in the pedigree when it is first created and SHALL occur only once.
`itemInfo`	`ItemInfoType`	Yes	Identifies the physical item(s) associated with a pedigree by lot number, expiration date, quantity of units, and item serial numbers if present. Pedigrees MAY represent quantities of one or more lots for the same product. This element MAY be repeated as many times as necessary to represent each lot number represented by the pedigree for the product.

`transactionInfo`	`TransactionInfoType`	Conditional	Information about a transaction in which ownership of the item passes from one supply chain partner to another, including information about the sender, recipient, and transaction reference. This element SHALL be present when a wholesaler initiates the pedigree to record the transaction information from the sale from the manufacturer to the wholesaler.
`receivingInfo`	`ReceivingInfoType`	Conditional	Information about the receipt of items associated with a pedigree. This element SHALL be present when a wholesaler initiates the pedigree to record wholesaler's receipt information.

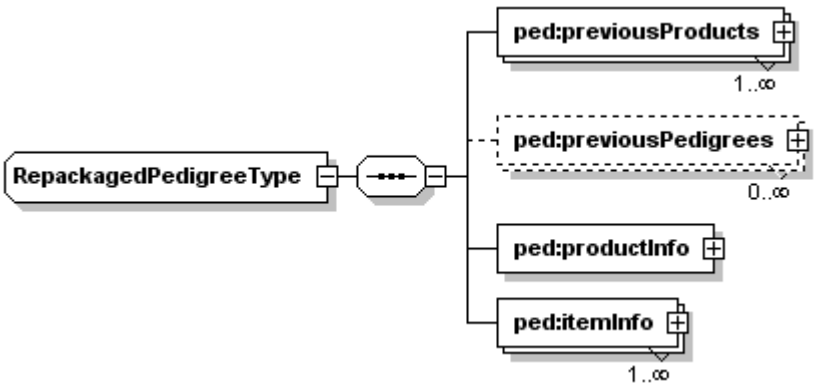
altPedigree	ForeignDataType	No	Scanned or alternate representation of a signed pedigree that contains the prior chain of custody for the item. Examples include scanned image of a paper pedigree, a PDF of a pedigree, or another electronic pedigree format. This element SHALL be present when a wholesaler creates a pedigree based on a pedigree received that was in an alternate format. This element SHALL include the serialNumber element.
wasRepackaged	xs:Boolean (Attribute of altPedigree)	Conditional	Identifies if alternate pedigree represents a repackaged item (full repackaging pedigree information would be included in the altPedigree element). The default value is false. This attribute SHALL be present and TRUE when an altPedigree represents a repackaged item.

attachment	ForeignDataType	No	One or more attachments to a pedigree to facilitate pedigree authentication by downstream trading partners. Examples include an EDI ASN or a scanned invoice or shipping document. This element MAY be used when a wholesaler creates a pedigree.

10.2.2.7 RepackagedPedigreeType

The RepackagedPedigreeType contains the initial pedigree information for a repackaged product before the first outbound transaction. This includes the product information for the repackaged item and pedigree information about the source items used to create the repackaged items.

diagram



used by element **repackagedPedigree**

Name	Type	Mandatory?	Description
previousProducts	PreviousProductType	Yes	Summary information about the source or "parent" pedigrees for the repackaged products. This element SHALL be repeated as many times as necessary to represent each product used to create the repackaged products.

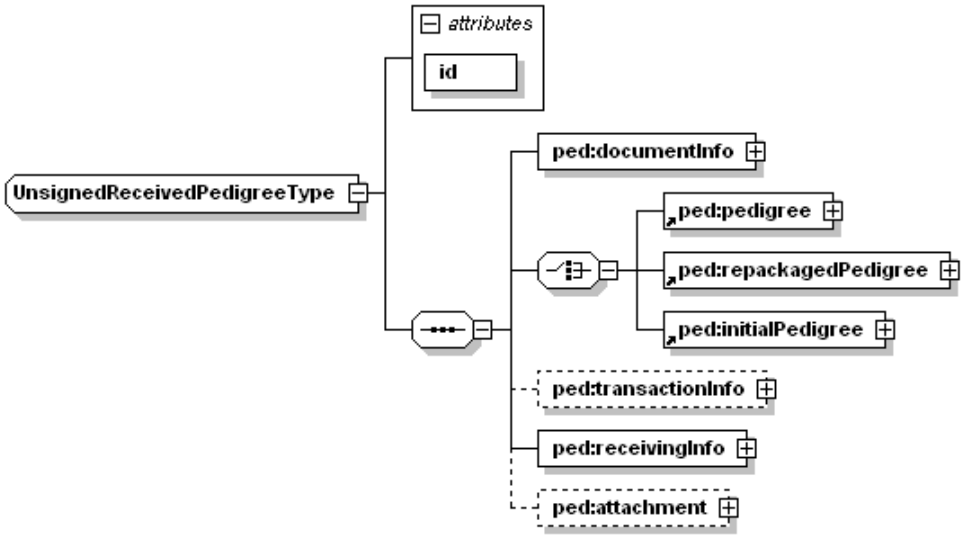
previousPedigrees	PreviousPedigreeType	Conditional	The pedigrees for the source or “parent” pedigrees for the repackaged products. This element SHALL be repeated as many times as necessary to include the pedigree for each product used to create the repackaged products. This element SHALL be present when there is a regulatory requirement to include the source or “parent” pedigrees for the repackaged products.
productInfo	ProductInfoType	Yes	Information about the product to which the new pedigree pertains, such as drug name, strength, dosage form, etc.. This element SHALL be inserted in the pedigree when it is first created and occurs only once.

`itemInfo`	`ItemInfoType`	Yes	Identifies the physical item(s) associated with a pedigree by lot number, expiration date, quantity of units, and item serial numbers if present. Pedigrees MAY represent quantities of one or more lots for the same product. This element MAY be repeated as many times as necessary to represent each lot number represented by the pedigree for the product.

10.2.2.8 UnsignedReceivedPedigreeType

The `UnsignedReceivedPedigreeType` represents an unsigned receiving stage in a pedigree exchange transaction in which ownership of a product passes from one supply chain partner to another. The `UnsignedeceivedPedigreeType` element SHALL wrap the preexisting pedigree and add information about the receipt to the pedigree, but does not get signed.

diagram



used by element **LayerType/unsignedReceivedPedigree**

Name	Type	Mandatory?	Description
`Id`	xs:id (Attribute)	Yes	A unique identifier for the pedigree layer within the pedigree document. This element SHALL be used to reference the element that will be signed.
`documentInfo`	`DocumentInfoType`	Yes	Contains an identifier for the pedigree and the version number of the pedigree schema used to create the pedigree.
`pedigree`	`LayerType`	Yes (Choice)	Wrapper element that contains a signed pedigree representing the prior chain of custody for a product.
`repackagedPedigree`	`RepackagedPedigreeType`	Yes (Choice)	Contains the initial pedigree information for a repackaged product before the first outbound transaction. This includes the product information for the repackaged item and pedigree information about the source items used to create the repackaged items.

Name	Type	Mandatory?	Description
`initialPedigree`	`InitialPedigreeType`	Yes (Choice)	Contains the initial pedigree information before the first outbound transaction. This element CAN be also used to represent the conversion of a pedigree in another form to this pedigree format. A scanned or alternate representation of a pedigree MAY be included in this element to represent the prior chain of custody for the product.
`transactionInfo`	`TransactionInfoType`	Conditional	Information about a transaction in which ownership of the item passes from one supply chain partner to another, including information about the sender, recipient, and transaction reference. This element SHALL be used only when a seller updates the pedigree with a return transaction from the customer back to the seller, and this element SHALL be used only when regulations allow the seller to update the pedigree on behalf of their customer.
`receivingInfo`	`ReceivingInfoType`	Yes	Information about the receipt of items associated with a pedigree.

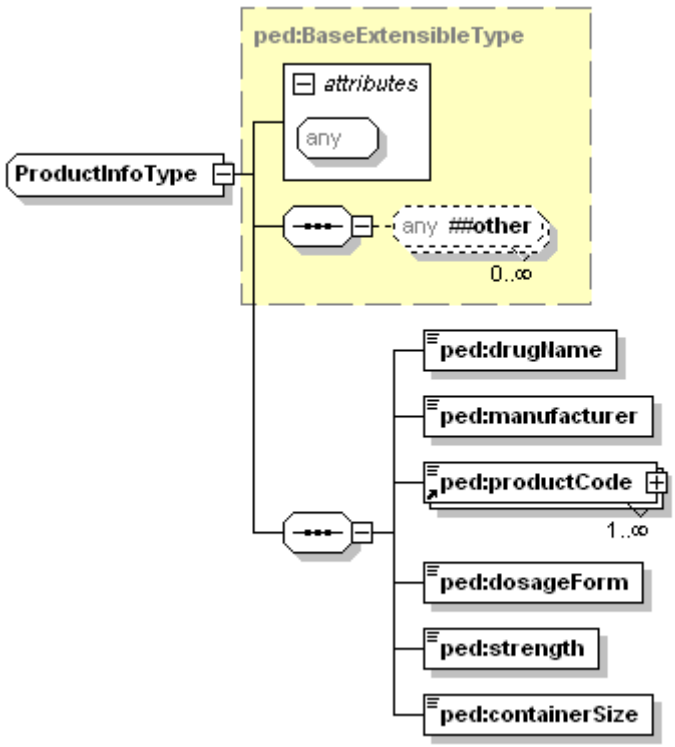
Name	Type	Mandatory?	Description
attachment	ForeignDataType	No	One or more attachments to a pedigree to facilitate pedigree authentication by downstream trading partners. Examples include an EDI ASN or a scanned invoice or shipping document. This element MAY be used when a wholesaler updates the pedigree with a return transaction from the pharmacy back to the wholesaler.

10.2.2.9 ProductInfoType

The `ProductInfoType` represents information about the pharmaceutical product that is the subject of the pedigree.

The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

diagram



used by elements **InitialPedigreeType/productInfo** **RepackagedPedigreeType/productInfo**

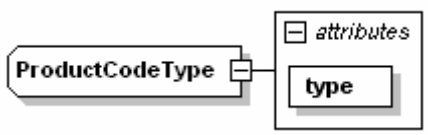
764

Name	Type	Mandatory?	Description
`drugName`	xs:string	Yes	The name of the drug as it appears on the product label.
`manufacturer`	xs:string	Yes	The name of the manufacturer or repackager of the drug as it appears on the product label.
`productCode`	`ProductCodeType`	Yes	The product class identifier for the pharmaceutical product (e.g., NDC value). This element MAY be repeated if multiple product codes for different countries are represented (e.g., NDC and DIN).
`dosageForm`	xs:string	Yes	The dosage form of the product (for example, TABLET, CAPSULE).
`strength`	xs:string	Yes	The strength or potency of the product, including the unit of measure (for example, 60 mg, 25 ml).
`containerSize`	xs:string	Yes	The number of units contained in a package of the product (for example, 60, 100).This is also known as pack size.

765 ### 10.2.2.10 ProductCodeType

766 The `ProductCodeType` represents name or type of the product class identifier supplied in the
767 `productCode`.

diagram



used by elements **PreviousProductInfoType/productCode** **productCode**

768

Name	Type	Mandatory?	Description
type	ProductCodeValueType (Attribute)	Yes	The name or type of the product class identifier supplied in the productCode (e.g., NDC442, NDC532, NDC541, NDC542, GTIN). The ProductCodeValueType contains an enumeration that is a union of NMTOKEN and the values defined by ProductCodeValueTypeType. One of the values defined in the ProductCodeValueTypeType SHOULD be used to ensure interoperability, however if the desired value is not present, a custom name value MAY be used.

10.2.2.11 ItemInfoType

The itemInfoType identifies the physical item(s) associated with a pedigree by lot number, expiration date, quantity of units, and item serial numbers if present. Pedigrees may represent quantities of one or more lots for the same product. This element MAY be repeated as many times as necessary to represent each lot number represented by the pedigree for the product. The itemInfo CAN represent both serialized and non-serialized items.

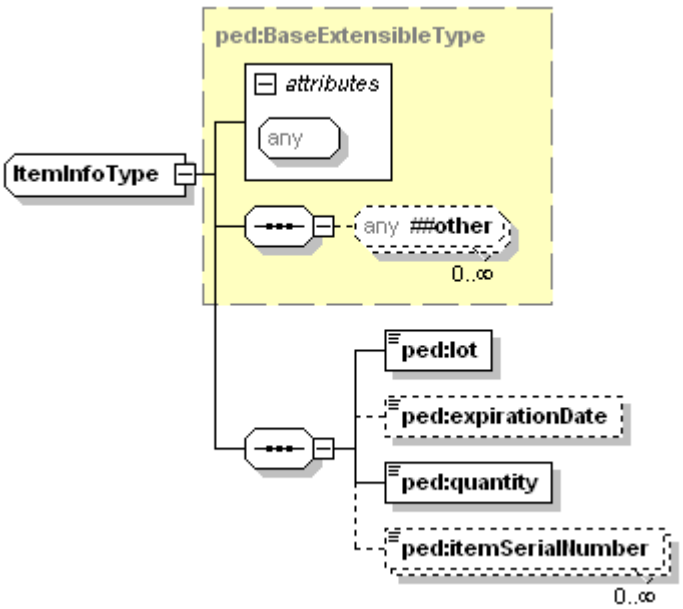
The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

A singular pedigree SHALL contain only one itemInfo and the quantity element SHALL contain "1". There SHALL be only one itemSerialNumber element present if the item is serialized, and this element SHALL contain the serial number associated with the product. If the product does not have a serial number, the itemSerialNumber SHALL be omitted.

An aggregate pedigree SHALL contain as many itemInfo elements as there are lot numbers represented by the pedigree. The quantity element for each lot number SHALL represent the number of items in the lot. If the items are serialized, the number of itemSerialNumber elements should match the quantity. If the products do not have serial numbers, the itemSerialNumber is omitted.

Except for the following special case, an itemInfo SHALL be added to a pedigree each time a new transaction is added to the pedigree to record the items that are the subject of the transaction. If the items that are the subject of the transaction are the exact same items in the last itemInfo, then the element MAY be omitted from the current transaction.

diagram



used by elements [InitialPedigreeType/itemInfo](#) [ShippedPedigreeType/itemInfo](#) [RepackagedPedigreeType/itemInfo](#) [ReceivingInfoType/itemInfo](#) [PreviousProductType/itemInfo](#)

790

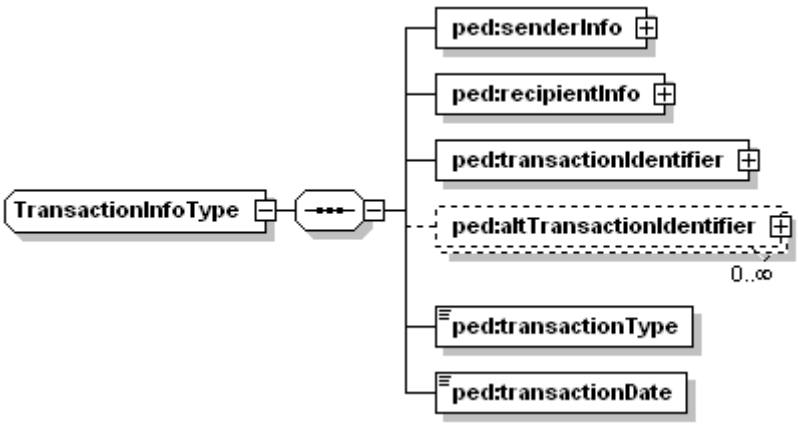
Name	Type	Mandatory?	Description
`lot`	xs:string	Yes	The lot number of the item.
`expirationDate`	xs:date	Conditional	The expiration date of the item. Each lot has its own expiration date. This element SHALL be present when there is a regulatory requirement to include the expiration date on the pedigree.
`quantity`	xs:integer	Yes	The number of items (e.g., eaches) of the NDC and lot that are part of the current exchange transaction. For serialized items, this quantity SHALL match the number of `itemSerialNumber` entries.

Name	Type	Mandatory?	Description
`itemSerialNumber`	xs:string	Conditional	The unique identifier for the physical product(s) represented by the pedigree, typically the EPC in pure-identity URI representation as defined in the EPC Tag Data Standards specification. If the physical product unit does not have a serial number, the `itemSerialNumber` SHALL be omitted. The total number of `itemSerialNumber` elements SHALL match the number of items specified in the `quantity` element. This element SHALL be present when there is a regulatory requirement to include the serial number associated with the saleable product unit on the pedigree. This SHALL NOT apply to products that are not serialized.

10.2.2.12 TransactionInfoType

The `TransactionInfoType` represents a transaction in which the ownership of an item is transferred from one supply chain partner to another, including information about the sender, recipient, and transaction identifier and date reference. The type of business transaction (e.g., Sale, Transfer, or Return) is also identified.

diagram



used by elements **InitialPedigreeType/transactionInfo**
ShippedPedigreeType/transactionInfo

796

Name	Type	Mandatory?	Description
`senderInfo`	`PartnerInfoType`	Yes	Information about the supply chain partner that is sending the shipment.
`recipientInfo`	`PartnerInfoType`	Yes	Information about the supply chain partner that is receiving the shipment.
`transactionIdentifier`	`TransactionIdentifierType`	Yes	The business document identifier, specified by the `TransactionIdentifierType`. Current supported business document types are invoice number, purchase order number, shipping number, return authorization number, and Other. The list of supported document types may be extended in the future.

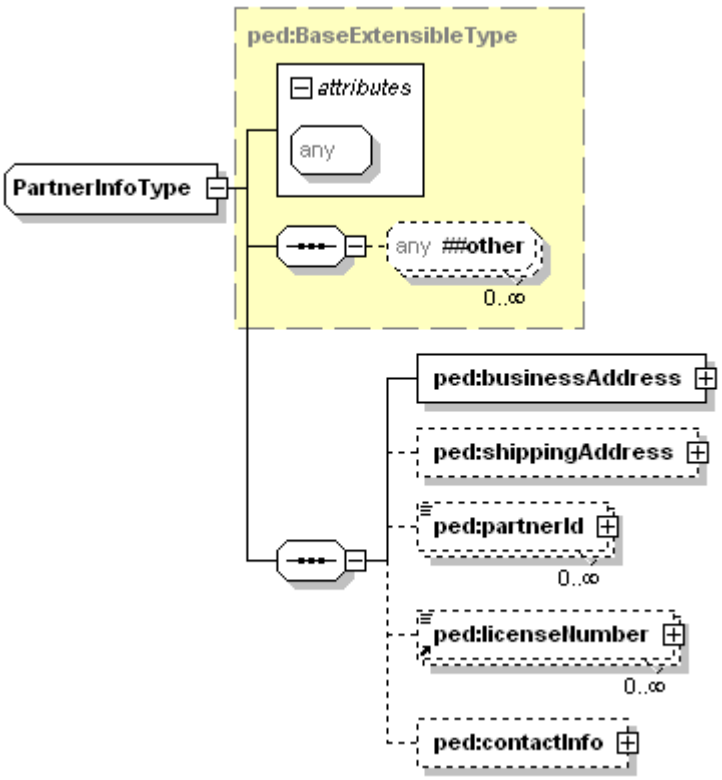
Name	Type	Mandatory?	Description
altTransactionIdentifie	TransactionIdentifierType	No	One more more business document identifiers, specified by the TransactionIdentifierType. This element SHALL be used if it is desired to provide more than one transaction identifier in the pedigree (e.g., both the PO Number and the Invoice Number for the transaction). Current supported business document types are invoice number, purchase order number, shipping number, return authorization number, and Other. The list of supported document types may be extended in the future.
transactionType	TransactionTypeType	Yes	The nature of the pedigree transaction (for example, Sale, Transfer, Return). This element has valid values enumerated by TransactionTypeType which currently include Sale, Return, Transfer, and Other. The list of supported values may be extended in the future.
transactionDate	xs:date	Yes	The date associated with the transactionIdentifier (e.g., PO, Invoice, etc). This element SHALL be paired with the transactionIdentifier element.

10.2.2.13 PartnerInfoType

The `PartnerInfoType` represents the business address and license information for a supply chain partner involved in transferring items that are associated with pedigrees.

The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

diagram



used by elements **TransactionInfoType/recipientInfo** **TransactionInfoType/senderInfo**

Name	Type	Mandatory?	Description
`businessAddress`	`AddressType`	Yes	The business address of the trading partner.

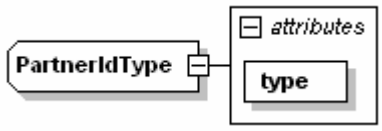
Name	Type	Mandatory?	Description
shippingAddress	AddressType	Conditional	The address that items are being shipped to or from. This element is used only if it is different from the businessAddress. This element SHALL be present when there is a regulatory requirement to record the shipping address when it differs from the business address (this typically applies to both the sender business address and ship-from address and the recipient business address and ship-to address).
partnerId	PartnerIdType	No	Any code used to identify the trading partner. This element MAY be repeated to include as many identifiers as desired (e.g., customer number, supplier code, GLN, etc.).
licenseNumber	xs:string	Conditional	The license number of the trading partner. This element MAY be repeated to include as many license numbers as required (e.g., a Florida license and an NABP VAWD number). This element SHALL be present when there is a regulatory requirement to include the license information for the trading partner on the pedigree.

Name	Type	Mandatory?	Description
State	xs:NMTOKEN (Attribute of licenseNumber)	No	The state or region in which the trading partner is licensed, using the standard two letter abbreviation specified in ISO 3166-2:1998 country sub-division code [16]. This attribute is used to give additional context to the license number.
Agency	xs:string (Attribute of licenseNumber)	No	The agency that granted the license (e.g., Florida DOH, NABP). This attribute is used to give additional context to the license number.
contactInfo	ContactType	Conditional	Contact information for use by downstream trading partners to authenticate (e.g.., verify) the transaction information recorded on the pedigree with the sender party. This element SHALL be present when there is a regulatory requirement to include authenticator contact information on the pedigree.

803 #### 10.2.2.14 ParnterIdType

804 The `PartnerIdType` represents represents the name or type of identifier supplied in the `partnerId`.

diagram



used by elements **PartnerInfoType/partnerId**

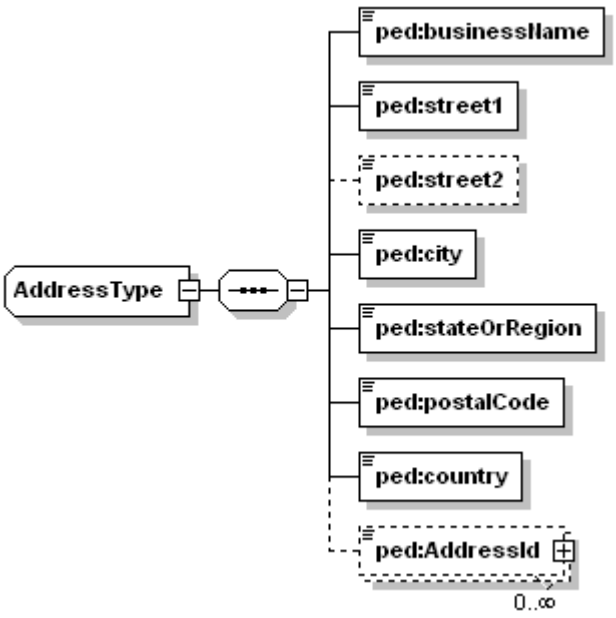
805

Name	Type	Mandatory?	Description
type	PartnerIdValueType (Attribute)	Yes	The name or type of identifier supplied in the partnerId element (e.g., “customer number”, “supplier code”, “GLN”, etc.). This attribute is used to give additional context to the partnerID. The PartnerIdValueType contains an enumeration that is a union of NMTOKEN and the values defined by PartnerIdValueTypeType. One of the values defined in the PartnerIdValueTypeType SHOULD be used to ensure interoperability, however if the desired value is not present, a custom name value MAY be used.

806 **10.2.2.15 AddressType**

807 The AddressType represents a business or shipping address.

diagram



used by elements PartnerInfoType/businessAddress PartnerInfoType/shippingAddress

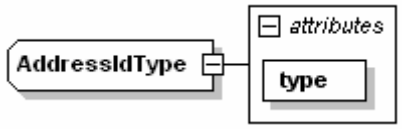
808

Name	Type	Mandatory?	Description
businessName	xs:string	Yes	The business name of the trading partner.
street1	xs:string	Yes	The first line of the street address.
street2	xs:string	No	The second line of the street address.
city	xs:string	Yes	The city.
stateOrRegion	xs:string	Yes	The state, province, or region using the standard two-letter abbreviation specified in ISO 3166-2:1998 country sub-division code [16].
postalCode	xs:string	Yes	The ZIP or other postal code.
country	xs:string	Yes	The country using the standard two-letter abbreviation specified in ISO 3166-1alpha-2:1997 country code [17].
AddressId	AddressIdType	No	Any code used to identify the address of the trading partner. This element MAY be repeated to include as many identifiers as desired (e.g., plant number, GLN, etc.).

809 **10.2.2.16 AddressIdType**

810 The `AddressIdType` represents represents the name or type of identifier supplied in the `addressId`.

diagram



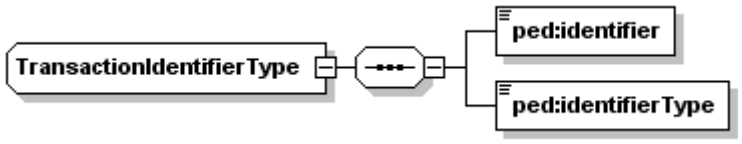
used by elements **AddressType/AddressId**

811

Name	Type	Mandatory?	Description
`type`	`AddressIdValue Type` (Attribute)	Yes	The name or type of identifier supplied in the `addressId` element (e.g., “plant number”, “GLN”, etc.). This attribute is used to give additional context to the `addressID`. The `AddressIdValueType` contains an enumeration that is a union of NMTOKEN and the values defined by `AddressIdValueTypeType`. One of the values defined in the `AddressIdValueTypeType` SHOULD be used to ensure interoperability, however if the desired value is not present, a custom name value MAY be used.

812 **10.2.2.17 TransactionIdentifierType**

diagram



namespace Pedigree

children **ped:identifier** **ped:identifierType**

used by element **TransactionInfoType/transactionIdentifier**

813

814 The `TransactionIdentifierType` represents the business document number and document type.
815 Current supported business document types are invoice number, purchase order number, shipping
816 number, return authorization number, and other. The list of supported document types may be
817 extended in the future.

Name	Type	Mandatory?	Description
`identifier`	xs:string	Yes	The business document number.

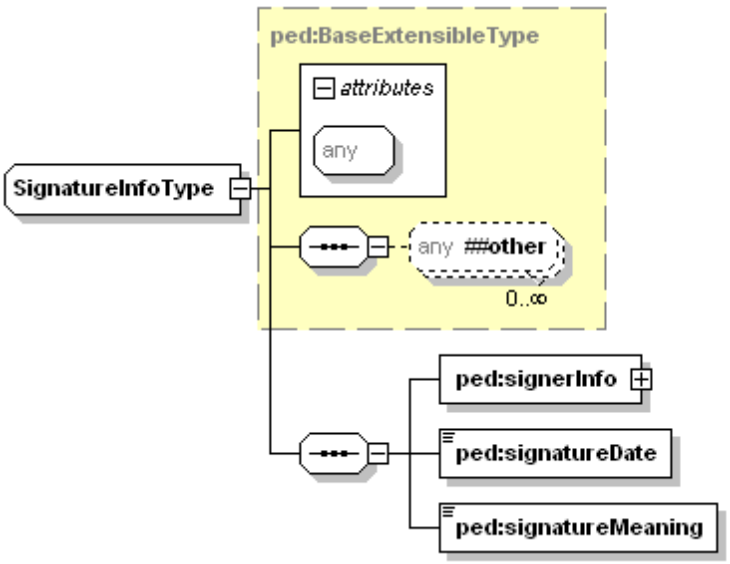
Name	Type	Mandatory?	Description
identifierType	TransactionIdentifierTypeType	Yes	The business document type defined by an enumerated list in the TransactionIdentifierTypeType. Current supported business document types are invoice number, purchase order number, shipping number, return authorization number, and Other. The list of supported document types may be extended in the future.

10.2.2.18 signatureInfo

The signatureInfo element represents information about the signer and context of the signature applied to a pedigree.

The BaseExtensibleType is used for extensibility of pedigree schema elements. Extensibility is only allowed in NameSpace ##other.

diagram



used by elements ShippedPedigreeType/signatureInfo ReceivedPedigreeType/signatureInfo

Name	Type	Mandatory?	Description
signerInfo	ContactType	Yes	Contact information for the signer of the document.

Name	Type	Mandatory?	Description
`signatureDate`	xs:dateTime	Yes	The date the digital signature was executed.
`signatureMeaning`	`signatureMeaningType`	Yes	The context for the application of the digital signature on the pedigree. This element has valid values enumerated by `signatureMeaningType` which currently include "Certified" (used when certifying the content added to a pedigree), "Received" (used by recipient after receiving the item against the pedigree), "Authenticated" (used by recipient after successfully authenticating the pedigree), and "Received and Authenticated" (used by the recipient after successfully authenticating a pedigree and receiving the item against the pedigree). The list of supported values may be extended in the future.

825 #### 10.2.2.19 ContactType

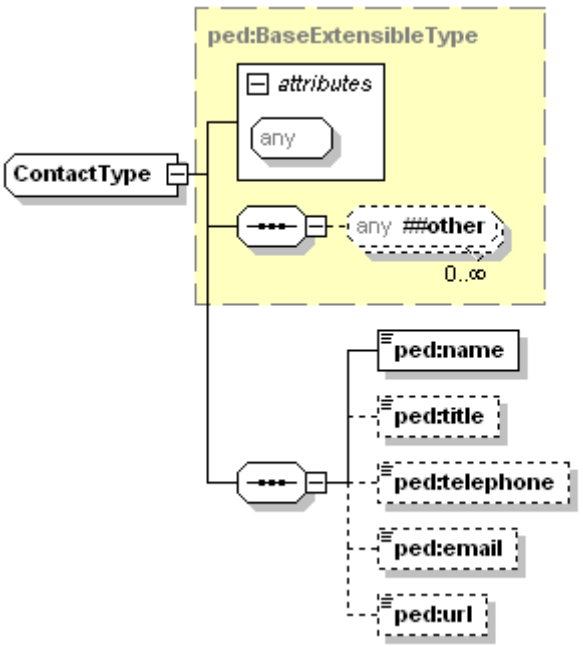
826 The `ContactType` element represents information about a person or company that has had custody of
827 a pedigree. This element MAY be used in two contexts: to represent signer information in a
828 shippedPedigree or receivedPedigree, and to represent contact information at the sender company
829 to manually authenticate the information for the pedigree transaction.

830 For signing information, name and title SHALL be present. All other fields are optional.

831 For authenticator information, name, telephone, and email SHALL be present. All other fields are
832 optional.

833 The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements.
834 Extensibility SHALL be only allowed in NameSpace ##other.

diagram



used by elements **PartnerInfoType/contactInfo** **PreviousProductType/contactInfo** **SignatureInfoType/signerInfo**

835

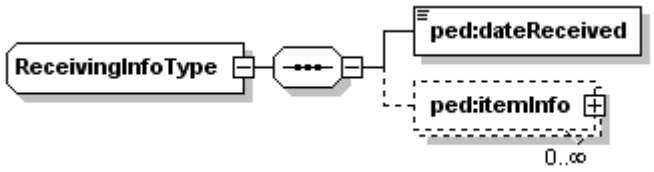
Name	Type	Mandatory?	Description
`name`	xs:string	Yes	Full name of the person or name of the company.
`title`	xs:string	Conditional	Job title of the person. This element MAY be mandatory in some regulatory contexts. See notes above in main description.
`telephone`	xs:string	Conditional	Phone number of the person. This element MAY be mandatory in some regulatory contexts. See notes above in main description.
`email`	xs:string	Conditional	Email of the person. This element MAY be mandatory in some regulatory contexts. See notes above in main description.

Name	Type	Mandatory?	Description
`url`	xs:string	No	Web address to facilitate authentication.

836 **10.2.2.20 ReceivingInfoType**

837 The `ReceivingInfoType` represents information about the receipt of items associated with a pedigree.
838 This element MAY be used to express partial receipts against a pedigree by identifying the specific
839 items received. The element in `ReceivingInfoType` is shown below.

diagram



used by element **ReceivedPedigreeType/receivingInfo**

840

Name	Type	Mandatory?	Description
`dateReceived`	xs:date	Yes for recipient No for distributor	The date the item was received.

Name	Type	Mandatory?	Description
`itemInfo`	`ItemInfoType`	Conditional	Identifies the physical item(s) received against this pedigree by lot number, expiration date, quantity of units, and item serial numbers if present. The items received SHALL be equal to or a subset of the items expressed in the `itemInfo` of the `shippedPedigree`. This element SHALL be repeated as many times as necessary to represent items received against each lot number represented by the pedigree for the product. This element MAY be used to express partial receipts against a pedigree if the regulatory requirements allow partial receipts against a single pedigree (e.g., pedigree represents 20 items, but only 15 items were received).

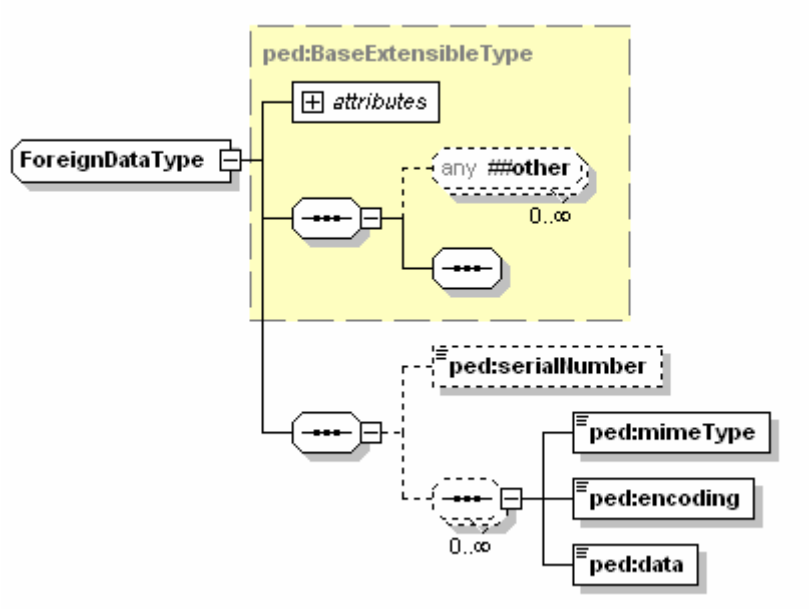
841 #### 10.2.2.21 ForeignDataType

842 The `ForeignDataType` identifies a data object included with a pedigree, specified by its MIME type.

843 The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements.
844 Extensibility SHALL be only allowed in NameSpace ##other.

845

diagram



used by elements [InitialPedigreeType/attachment](#) [InitialPedigreeType/altPedigree](#)

846

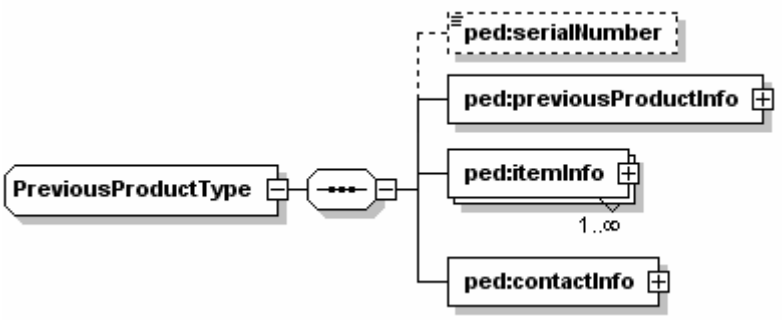
Name	Type	Mandatory?	Description
`serialNumber`	xs:string	Conditional	The globally unique identifier, using the UUID 128-bit identifier per RFC 4122, including the "urn:uuid" namespace prefix. This element SHALL be inserted into new alternate pedigrees when they are first created. This element SHALL be referenced in new `repackagedPedigree` elements when the `repackagedPedigree` references an `altPedigree` for a previous product.
`mimeType`	xs:string	Yes	The MIME type of the content that SHALL be included with the pedigree specified in standard type/subtype representation (e.g., text/plain, application/pdf, image/jpeg).

Name	Type	Mandatory?	Description
`encoding`	`EncodingType`	Yes	The data encoding format of the content that will be included with the pedigree. Only base 64 binary encoding is supported and the value of this element SHALL be base64binary.
`data`	xs:string	Yes	The data representation of the data object that will be included with the pedigree.

10.2.2.22 PreviousProductType

The `PreviousProductType` contains summary information about the source or "parent" products for the repackaged or kitted products. This element SHALL be repeated as many times as necessary to represent each product used to create the repackaged products.

diagram



used by element **RepackagedPedigreeType/previousProducts**

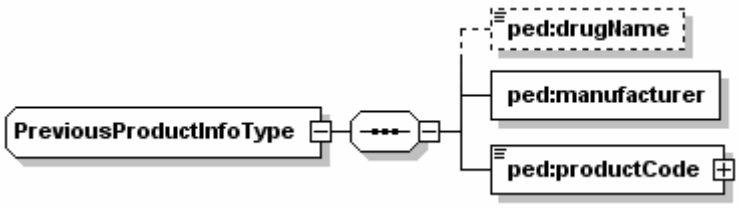
Name	Type	Mandatory?	Description
serialNumber	xs:string	Conditional	Serial number to reference the specific initialPedigree, pedigree, or altPedigree serial number associated with the previous product used in the repackagedPedigree. This element SHALL reference a serial number of an intialPedigree, altPedigree, or pedigree that is defined in the previousPedigrees element. This element is optional in the schema only to support backwards compatibility with the interim (pre-standard) version of the pedigree schema where it was not present. Software implementations SHALL accommodate pedigrees where this element is present and where it is not present.
previousProductInfo	PreviousProdutInfoType	Yes	Summary information about the source or "parent" products for the repackaged products. This element SHALL be a cross-reference to the productInfo:drugName when the previousProduct references an initialPedigree or pedigree. This element MAY reference a generic product when used to refer to items in a kit that may not be prescription drugs or have pedigrees.

Name	Type	Mandatory?	Description
`itemInfo`	`ItemInfoType`	Yes	Identifies the physical item(s) from the source or "parent" pedigrees used to create the repackaged products (because only a subset of the items referenced by the source pedigree may be used in the repackaging). Items are identified by lot number, expiration date, quantity of units, and item serial numbers if present. The items referenced SHALL be equal to or a subset of the items expressed in the `itemInfo` of the parent pedigrees. This element SHALL be repeated as many times as necessary to represent items used in the repackaging against each lot number represented by the pedigree for the parent product.
`contactInfo`	`ContactType`	Yes	Contact information for authenticating the parent pedigrees.

852 **10.2.2.23 PreviousProductInfoType**

853 The `PreviousProductInfoType` contains summary information about the source or "parent" pedigrees
854 for the repackaged products.

diagram



used by element **PreviousProductType/previousProductInfo**

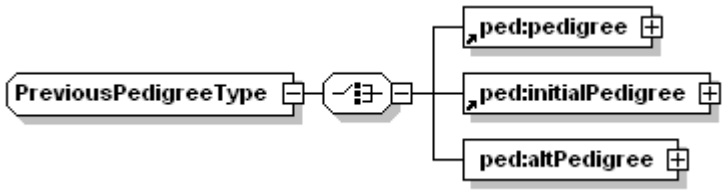
855

Name	Type	Mandatory?	Description
drugName	xs:string	No	The name of the drug as it appears on the product label. This element SHALL be a cross-reference to the productInfo:drugName when the previousProduct references an initialPedigree or pedigree. This element MAY reference a generic product name when used to refer to items in a kit that may not be prescription drugs or have pedigrees.
manufacturer	xs:string	Yes	The name of the manufacturer of the source or "parent" drug as it appears on the product label.
productCode	ProductCodeType	Yes	The product class identifier for the source or "parent" pharmaceutical product (e.g., NDC value). This element MAY be repeated if multiple product codes for different countries are represented (e.g., NDC and DIN).

856 #### 10.2.2.24 PreviousPedigreeType

857 The PreviousPedigreeType contains the source or "parent" pedigrees for the repackaged products.
858 This element SHALL be repeated as many times as necessary to represent each product used to
859 create the repackaged products.

diagram



used by element **RepackagedPedigreeType/previousPedigrees**

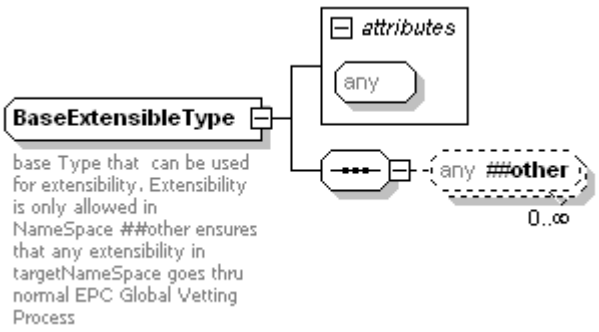
860

Name	Type	Mandatory?	Description
pedigree	LayerType	Yes (Choice)	The source or "parent" pedigrees for the repackaged products. This element SHALL be used if the pedigrees were received in the standard electronic format.
initialPedigree	InitialPedigreeType	Yes (Choice)	The source or "parent" pedigrees for the repackaged products. This element MAY be used if the pedigrees were received in an alternate pedigree format, or without a pedigree and a new pedigree had to be created before repackaging.
altPedigree	ForeignDataType	Yes (Choice)	Scanned or alternate representation of a signed pedigree that contains the prior chain of custody for the item. Examples include scanned image of a paper pedigree, a PDF of a pedigree, or an alternate electronic transmission such as X.12. This element MAY be used if the pedigrees were received in an alternate pedigree format.

10.2.2.25 BaseExtensibleType

The BaseExtensibleType SHALL be used for extensibility of pedigree schema elements. Extensibility SHALL be only allowed in NameSpace ##other ensures that any extensibility in targetNameSpace goes through the normal EPCglobal vetting process.

diagram



used by complexTypes **ContactType** **DocumentInfoType** **ForeignDataType** **ItemInfoType** **PartnerInfoType** **ProductInfoType**

10.2.2.26 EncodingType

The `EncodingType` provides an enumerated list of the allowed encoding types used for the alternate pedigree and pedigree attachment. The value SHALL be "base64binary"

namespace Pedigree

type restriction of **xs:string**

used by element **ForeignDataType/encoding**

facets enumeration base64binary

10.2.2.27 ProductCodeValueTypeType

The `ProductCodeValueTypeType` provides an enumerated list of the allowed product code values.

If one of the NDC enumerations is used, the NDC SHALL be expressed as a contiguous string with no dashes separating the segments. Refer to the FDA National Drug Code Directory [15] for a description of the three segments which make up the NDC. See section 2 "NDC Number" at http://www.fda.gov/cder/ndc/index.htm.

The following examples illustrate how an NDC would be expressed for the supported enumerations.

- NDC532 would be expressed as 1341111303, which is the same as 13411-113-03.
- NDC541 would be expressed as 5456944670, which is the same as 54569-4467-0.
- NDC442 would be expressed as 9781112311, which is the same as 9781-1123-11.
- NDC542 would be expressed as 00071015723, which is the same as 00071-0157-23.

namespace Pedigree

type restriction of **xs:string**

used by simpleType **ProductCodeValueType**

facets	enumeration	NDC442
	enumeration	NDC532
	enumeration	NDC541
	enumeration	NDC542
	Enumeration	GTIN

10.2.2.28 TransactionIdentifierTypeType

The `TransactionIdentifierTypeType` provides an enumerated list of the allowed business document type used to qualify the transaction identifier.

namespace	Pedigree	
type	restriction of **xs:string**	
used by	element	**TransactionIdentifierType/identifierType**
facets	enumeration	InvoiceNumber
	enumeration	PurchaseOrderNumber
	enumeration	ShippingNumber
	enumeration	ReturnAuthorizationNumber
	enumeration	Other

10.2.2.29 TransactionTypeType

The `TransactionIdentifierTypeType` provides an enumerated list of the types that describe the nature of the pedigree transaction. These values are derived from the types of business transactions that require pedigree.

namespace	Pedigree	
type	restriction of **xs:string**	
used by	element	**TransactionInfoType/transactionType**
facets	enumeration	Sale
	enumeration	Return
	enumeration	Transfer
	enumeration	Other

10.2.2.30 signatureMeaningType

The `signatureMeantingType` provides an enumerated list of the types that describe the context for the application of the digital signature on the pedigree. These values are derived from the types of signatures required to be applied to the pedigree. These currently include:

- Certified: Used when certifying the content added to a pedigree.
- Received: Used by recipient after receiving the item against the pedigree.
- Authenticated: Used by recipient after successfully authenticating the pedigree.
- Received and Authenticated: Used by the recipient after successfully authenticating a pedigree and receiving the item against the pedigree.

namespace	Pedigree	
type	restriction of **xs:string**	
used by	element	**SignatureInfoType/signatureMeaning**
facets	enumeration	Certified
	enumeration	Received
	enumeration	Authenticated
	enumeration	ReceivedAndAuthenticated

10.2.2.31 signatureType

The Electronic Pedigree Interchange Format uses the W3C XML Digital Signature standard to represent a digital signature. Refer to section 6 Certificates and Digital Signatures and specifically section 6.3 Digital Signatures for specific schema requirements.

The following table defines the elements that SHALL be signed when certifying pedigrees that support the self-authenticating pedigree model specified in pedigree regulations.

Scenario	Signed Elements
Pedigree sent outbound by a supplier to a customer.	• The `Signature` element signs the outermost `shippedPedigree` element, thereby signing the entire pedigree content, including prior pedigree transactions nested in the interior `Pedigree` element.
Pedigree received by a customer from a supplier.	• The `Signature` element signs the outermost `receivedPedigree` element, thereby signing the receiving information added and the prior pedigree transactions nested in the interior `Pedigree` element.

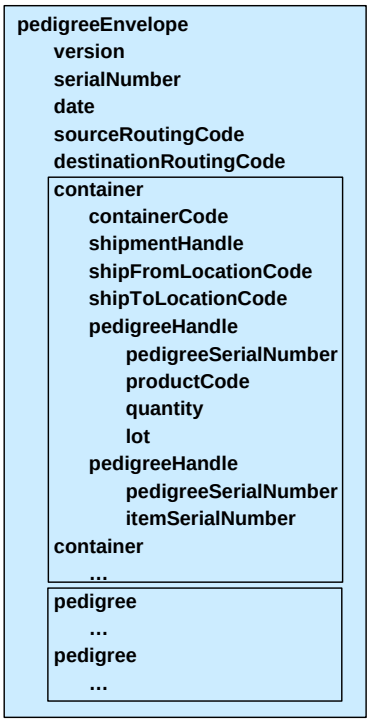
10.3 Pedigree Envelope Format

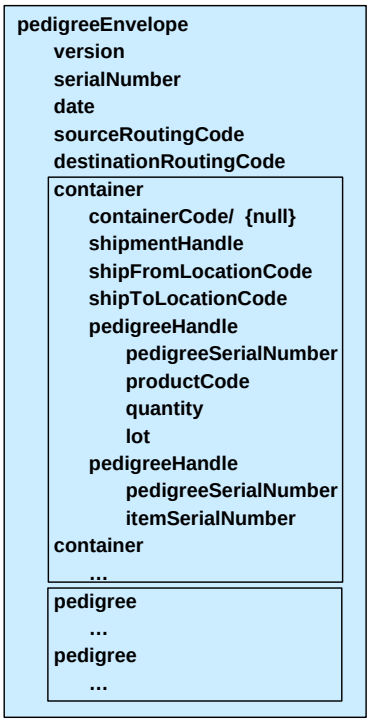
The pedigree envelope SHALL be a schema available to the user as an option. It is an electronic wrapper used to transmit a collection of pedigrees associated with an outbound customer shipment. The pedigree envelope may also contain information about the association of pedigrees to quantities of products in specific cases in the shipment to facilitate product-to-pedigree matching for non-serialized items. The matching of products received to their corresponding pedigrees is a requirement of the pedigree process.

910 The items in the shipment may be non-serialized or serialized. Pedigrees may have a one-to-one
911 mapping to items in the shipment (e.g., one pedigree per item), or may have a one-to-many
912 mapping to items in the shipment (e.g., one pedigree per collection of items with the same NDC).

913 10.3.1 Forms for Specific Business Situations (non-normative)

914 The following is a description of the fields that are used in the pedigree envelope. The table below
915 provides examples of the forms the pedigree envelope can take.

Form	Example
Pedigree envelope with product to pedigree mapping information (with case serialization) and pedigrees	

Form	Example
Pedigree envelope with product to pedigree mapping information (without case serialization) and pedigrees	

10.3.2 XML Elements

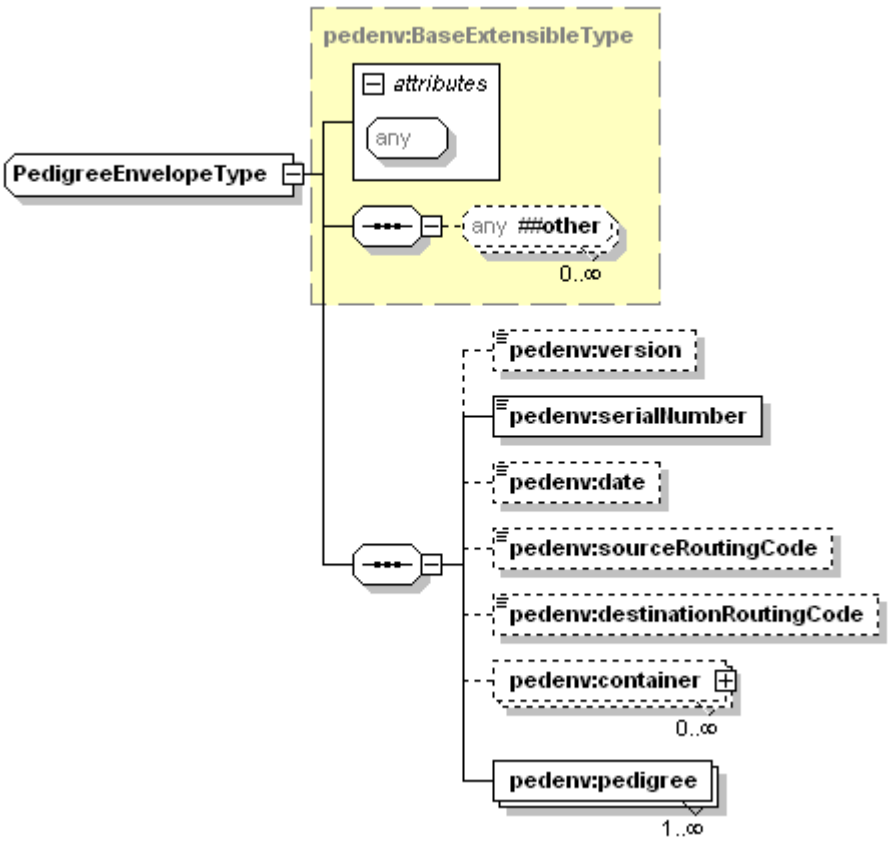
XML Schema is used to specify the structure of the pedigree envelope. All data types prefixed with "xs" represent standard type definitions imported from the XML Schema specification. Unless otherwise specified there are no length or content restrictions on the elements of the message.

10.3.2.1 PedigreeEnvelopeType

The `PedigreeEnvelopeType` represents the collection of pedigrees associated with the physical items in one or more shipments.

The BaseExtensibleType SHALL be used for extensibility of pedigree envelope schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

diagram



used by element **pedigreeEnvelope**

926

Name	Type	Mandatory?	Description
`version`	xs:string	Yes	The version number of the pedigree envelope schema.
`serialNumber`	xs:string	Yes	A unique identifier for the pedigree envelope that contains the pedigrees for a customer shipment. This identifier SHALL be expressed using a UUID 128-bit identifier as defined in RFC 4122, including the "urn:uuid" namespace prefix.
`date`	xs:dateTime	No	The date the pedigree envelope was created and transmitted.

Name	Type	Mandatory?	Description
sourceRoutingCode	xs:string	No	A reference or location code for the source from which the pedigrees were transmitted.
destinationRoutingCode	xs:string	No	A reference or location code for the destination to which the pedigrees are being transmitted.
container	ContainerType	No	Information about the association of pedigrees to items to cases to facilitate product-to-pedigree matching for non-serialized items.
pedigree	ped:PedigreeType	Yes	The pedigrees for each of the items in the shipment, as defined in the pedigree schema.

10.3.2.2 ContainerType

The `ContainerType` provides a common method for standardizing the exchange of the pedigree matching information within the direct context of the pedigree exchange. It provides information about the association of pedigrees to quantities of products in specific cases to facilitate the requirement of matching products received to their corresponding pedigrees.

This information SHALL be optional in the pedigree envelope, and is intended to provide a standard approach to conveying pedigree matching information for products that are not individually serialized. When products are serialized at the unit level, the serial number of the product serves as the matching point to the corresponding pedigree, and therefore no additional information is required. When products are not serialized, the receiving party may require the necessary information to verify quantities of products received against their corresponding pedigrees to satisfy regulatory verification requirements. The `ContainerType` MAY be included in the pedigree envelope to satisfy this requirement, and to provide a standard approach to convey this information.

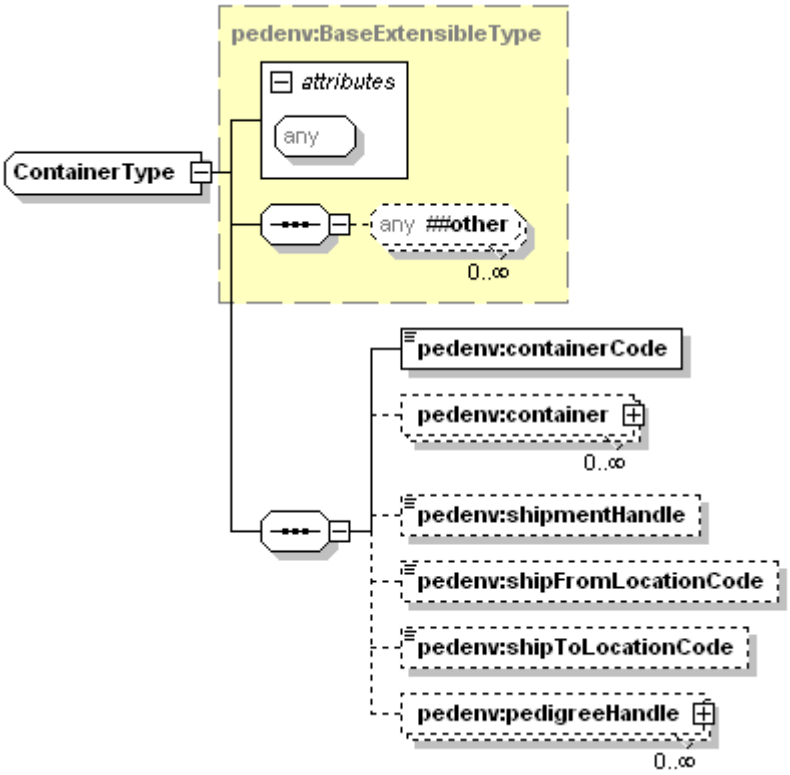
The pedigree format was created with flexibility that allows for a single pedigree to represent multiple products and products of multiple lots, however, the pedigree format does not contain any information that ties specific pedigrees to specific quantities of products packed into specific cases. Rather than conveying this information in the regulatory pedigree document (which must be retained as a document of record for up to three years), this information MAY be conveyed via the pedigree envelope. The decision to include this information in the pedigree envelope was driven by the temporary nature of the data – to convey the product to pedigree relationship to the recipient when the product is transferring ownership. Once a company receives the product, the company can maintain the product to pedigree relationship using the means appropriate to that company.

When the `container` element is omitted from the pedigree envelope, product-to-pedigree matching relies on information contained in the pedigree itself such as NDC, lot number, quantity, PO and unit product serial number (if the unit product is serialized).

When the `container` element is included in the pedigree envelope, additional product-to-pedigree matching approaches are possible when the products are not serialized. This capability is important to achieving compliance until products are serialized, or by other means that may be communicated between specific trading partners.

The BaseExtensibleType SHALL be used for extensibility of pedigree envelope schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

diagram



used by element **PedigreeEnvelopeType/container**

Name	Type	Mandatory?	Description
containerCode	xs:string	Yes (nillable=true)	Serial number of the container (e.g., case, tote, etc.) that contains the pedigreed items. If case serialization information is not available this element MAY represent a “null” value. The “null” value SHALL be represented using XML schema's nil mechanism (e.g., <containerCode xsi:nil="true"></containerCode>).
container	ContainerType	No	Information about the association of pedigrees to items to cases to facilitate product-to-pedigree matching for non-serialized items. These are expressed as parent-child relationships. This element SHALL be used to express subcontainers that have pedigrees associated with items in those subcontainers.
shipmentHandle	xs:string	No	A unique identifier for the shipment that this container and its pedigrees are associated with.
shipFromLocationCode	xs:string	No	A reference or location code for the source or facility from which the items/pedigrees were delivered.
shipToLocationCode	xs:string	No	A reference or location code for the destination or facility to which the items/pedigrees are being delivered.

Name	Type	Mandatory?	Description
pedigreeHandle	PedigreeHandleType	No	A list of one or more pointers to pedigrees that identify which items are present in this container. If a pedigree represents multiple lots of the same product, a separate pedigreeHandle element SHALL be included for each unique lot represented by the pedigree.

10.3.2.3 PedigreeHandleType

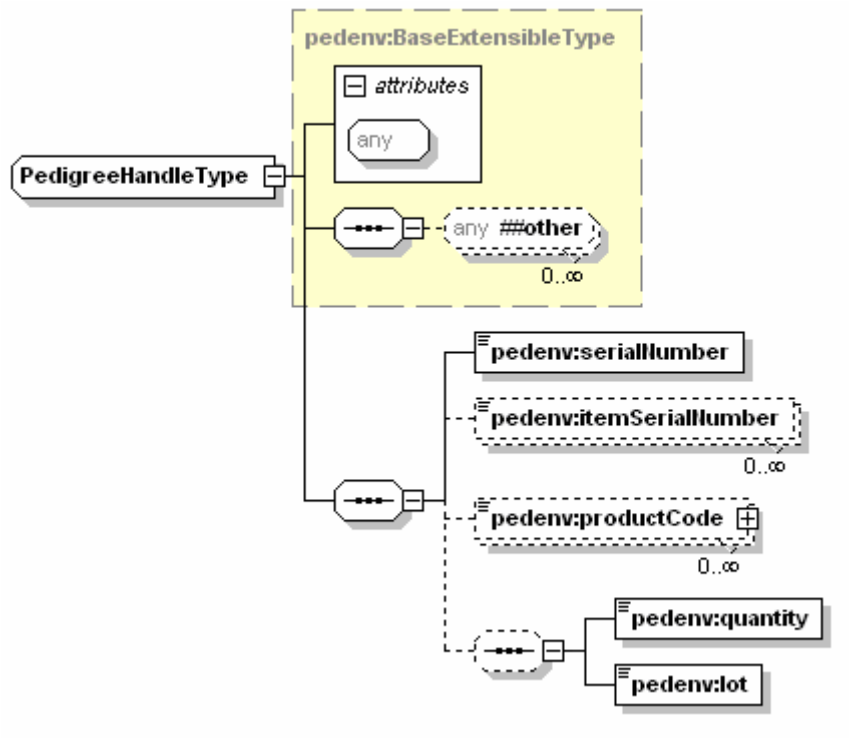
The PedigreeHandleType is a reference to a pedigree that indicates which items are present in a particular container.

The BaseExtensibleType SHALL be used for extensibility of pedigree envelope schema elements. Extensibility SHALL be only allowed in NameSpace ##other.

A pedigreeHandle element SHALL always include the pedigree serialNumber. If the products represented by the pedigree are serialized items, the pedigreeHandle SHALL include the itemSerialNumber for each serialized item represented by the pedigree. If the products represented by the pedigree are not serialized items, the pedigreeHandle MAY include the prodctCode and SHALL include the quantity and lot elements for the lot of product represented by the pedigree.

If a pedigree represents multiple lots of the same product, a separate pedigreeHandle element SHALL be included for each unique lot represented by the pedigree.

diagram



used by element **ContainerType/pedigreeHandle**

972

Name	Type	Required?	Description
`serialNumber`	xs:string	Yes	The serial number of a pedigree document contained in this `pedigreeEnvelope`, using the UUID 128-bit identifier per RFC 4122, including the "urn:uuid" namespace prefix..
`itemSerialNumber`	xs:string	Conditional	The unique identifier(s) for the physical item(s) in the container that are associated with the pedigree. This element is repeated multiple times, one for each item serial number. This element

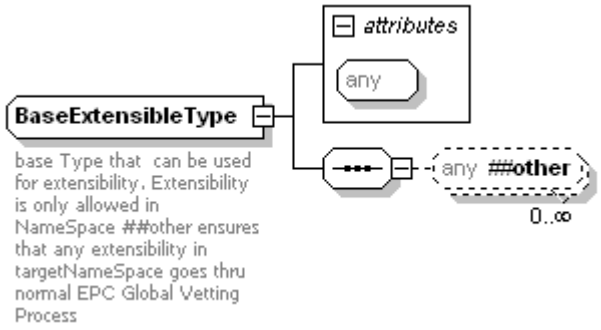
			SHALL be present only when the individual product items are serialized and SHALL be repeated multiple times, one for each item serial number.
`productCode`	`ProductCodeType`	No	The product class identifier for the pharmaceutical product (e.g., NDC value). If present, this element SHALL reflect the `productCodes` enumerated in the `productCode` element of the corresponding pedigree.
`quantity`	xs:integer	Conditional	The number of items in the container. This element is used with the `lot` to describe the non-serialized items associated with the pedigree. This element SHALL be present if the individual product items are not serialized.
`lot`	xs:string	Conditional	The lot number of the items in the container. This is element is used with the `quantity` to describe the non-serialized items associated with the pedigree. This element SHALL be present if

the individual product items are not serialized.

10.3.2.4 BaseExtensibleType

The `BaseExtensibleType` SHALL be used for extensibility of pedigree envelope schema elements. Extensibility SHALL be only allowed in NameSpace ##other ensures that any extensibility in targetNameSpace goes through the normal EPCglobal vetting process.

diagram



used by complexType **ContainerType** **PedigreeEnvelopeType** **PedigreeHandleType**

11 XML Schema Implementation

See attached XML schemas Pedigree.xsd and PedigreeEnvelope.xsd

12 Usage Guidelines (non-normative)

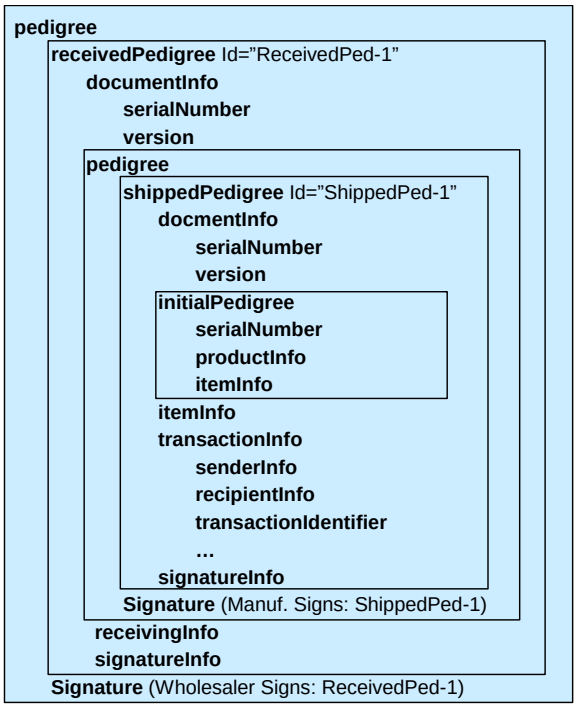
The following usage guidelines provide descriptions of the XML schemas can be used for various use cases.

12.1 Usage Guidelines for Creating and Appending Information to Pedigrees

This section explains how to use the `Pedigree` element and its sub elements to create pedigrees and append transactional and signature information to them. All content in this section is non-normative.

12.1.1 Pedigree Flow Initiated by Manufacturer

The pedigree flow is described for a sale from a manufacturer to a wholesaler, when the manufacturer initiates the pedigree.



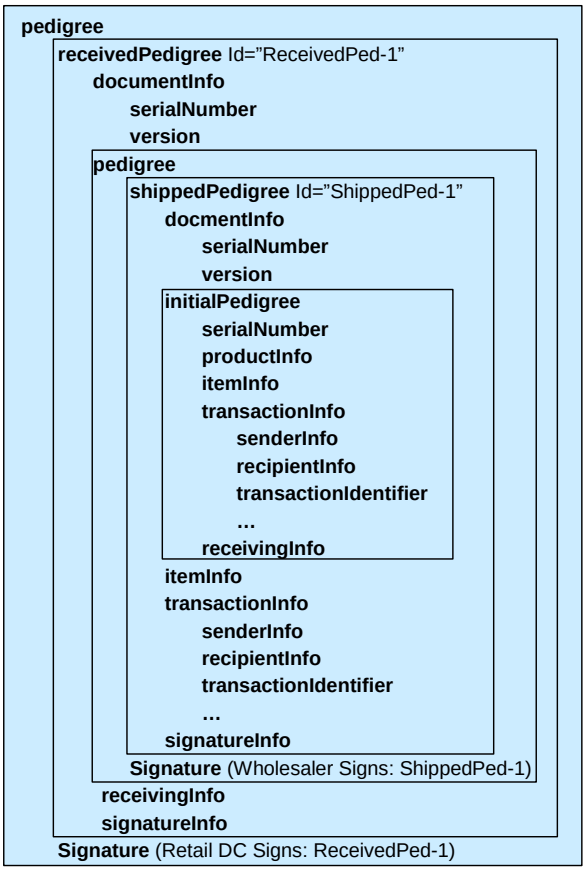
990

Step	Details
Manufacturer creates pedigree	• The manufacturer generates the `initialPedigree`, which contains the `serialNumber` (unique serial number), `productInfo` (generic product information) and `itemInfo` (identifies the specific items represented by the pedigree) elements. • Upon sale of the item, the manufacturer adds transaction information for the sale and signs the pedigree. The manufacturer wraps the `initialPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo, itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Certified"). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

Step	Details
Wholesaler receives product and pedigree	• When the wholesaler organization receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The wholesaler wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Received and Authenticated"). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.2 Pedigree Flow Initiated by Wholesaler

The pedigree flow is described for a sale from a wholesaler to a retail pharmacy DC, when no pedigree is provided by the manufacturer and the wholesaler initiates the pedigree.



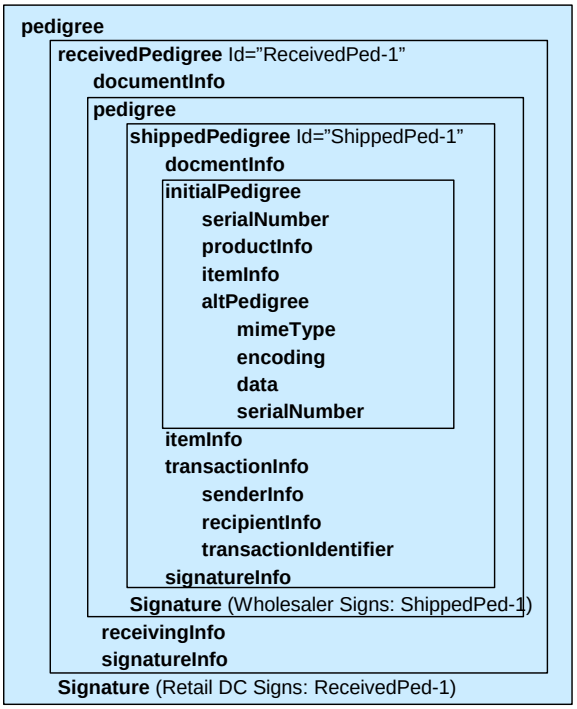
995

Step	Details
Wholesaler creates pedigree for product received from manufacturer	• The wholesaler generates the `initialPedigree`, which contains the `serialNumber` (unique serial number), `productInfo` (generic product information), `itemInfo` (identifies the specific items represented by the pedigree), and `transactionInfo` (describes the sale from the manufacturer to the wholesaler), and `receivingInfo` (describes the receipt information for the shipment from the manufacturer to the wholesaler) elements. The wholesaler may optionally include the `attachment` element with the original EDI ASN document to support downstream trading partner authentication of the sale transaction from the manufacturer to the wholesaler. • Upon sale of the item, the wholesaler adds transaction information for the sale and signs the pedigree. The wholesaler wraps the `initialPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

Step	Details
Pharmacy DC receives product and pedigree	• When the pharmacy DC receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The pharmacy DC wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Received and Authenticated"). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.3 Pedigree Flow Initiated by Wholesaler from Paper Pedigree

The pedigree flow is described for a sale from a wholesaler to a retail pharmacy DC, when the prior pedigree was in paper form and the receiving information was applied to the paper pedigree, and the wholesaler converts the pedigree to electronic form prior to the sale to the retail pharmacy DC.

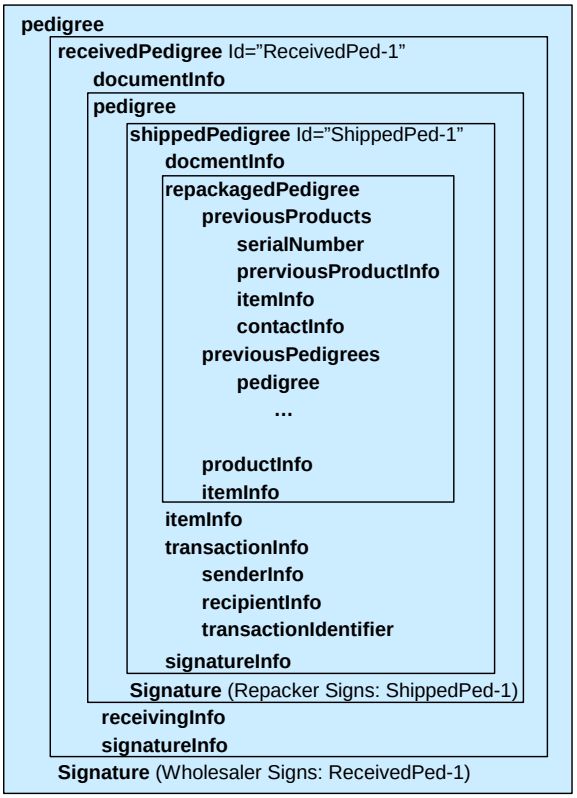


Step	Details
Wholesaler creates pedigree for product received from manufacturer	• The wholesaler receives a paper pedigree and generates an electronic pedigree by creating the `initialPedigree`, which contains the `serialNumber` (unique serial number), `productInfo` (generic product information), `itemInfo` (identifies the specific items represented by the pedigree), and `altPedigree` (contains the prior pedigree information received in paper or other form) elements. • Upon sale of the item, the wholesaler adds transaction information for the sale and signs the pedigree. The wholesaler wraps the `initialPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Certified"). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.
Pharmacy DC receives product and pedigree	• When the pharmacy DC receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The pharmacy DC wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Received and Authenticated"). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

Note: In the above scenario, the wholesaler could also opt to embed the paper pedigree in the electronic pedigree and include the `transactionInfo` and `receivingInfo` in the `intialPedigree`.

12.1.4 Pedigree Flow Initiated by Repacker

The pedigree flow is described for a sale from a repacker to a wholesaler, where the repacker initiates the pedigree for a repackaged item. A repack pedigree may or may not contain the pedigrees for the source products used to create the repack products, depending on the regulatory requirements of a given pedigree law. The usage guideline describes how to construct the pedigree for both scenarios, when the source pedigrees are required and when they are not required. The usage guideline also describes how to include the source pedigree when the source pedigree is an electronic pedigree created or received, or a pedigree received in an alternate form, such as a scanned paper pedigree.



Step	Details
Repackager creates pedigree for repackaged product	• The repacker generates the `repackagedPedigree` and includes the `previousProducts`, optionally includes the `previousPedigrees`, and includes the `productInfo` and `itemInfo` elements. The `previousProducts` element describes the source items used to create the repackaged pedigrees and provides a reference to the `serialNumber` of the source pedigree if the product has one. The `previousPedigrees` contains the actual pedigrees for the source items using either the `pedigree`, `altPedigree` or `initialPedigree` element. This element is included only if the pedigrees for the source items are required to be in the repacked pedigree. If the pedigrees for the source items are not required, the `previousPedigrees` may be omitted. The `productInfo` element describes the generic product information. The `itemInfo` identifies the specific repacked items represented by the pedigree. • If the source pedigrees are electronic pedigrees created or received, then the `pedigree` element is used to represent the source pedigree. If the source pedigrees are in an alternate form, such as a scanned paper pedigree, then the `initialPedigree` or the `altPedigree` element is used to represent the source pedigree. To the `initialPedigree` add the `productInfo` (describing the source product), `itemInfo` (describing the specific source items), optionally add the `transactionInfo` (describing the sales transaction used to receive the source items) and `receivingInfo` (describing the receipt information for the source items), and add the `altPedigree` (the representation of the scanned source pedigree). • Upon sale of the item, the repacker adds transaction information for the sale and signs the pedigree. The repacker wraps the `repackagedPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

Step	Details
Wholesaler receives repackged product and pedigree	• When the wholesaler organization receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The wholesaler wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Received and Authenticated"). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.5 Pedigree Flow for a Kit

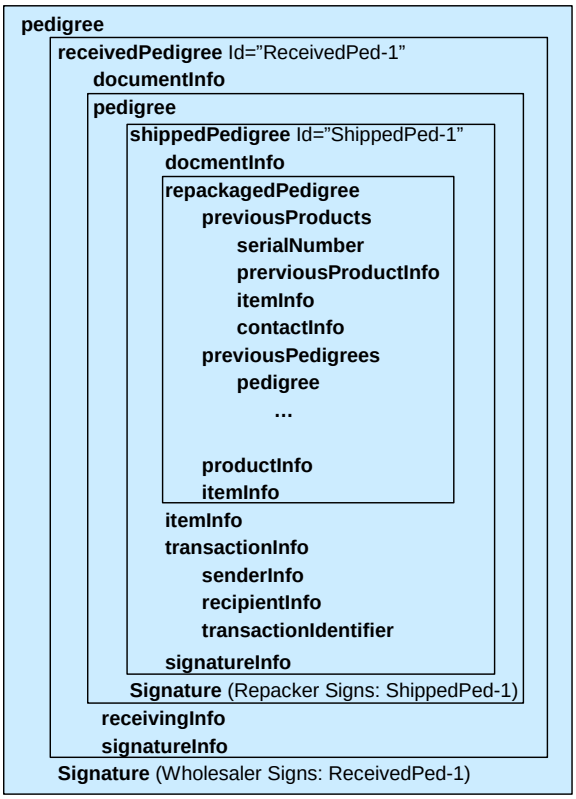
A kit is a packaged product that can contain one more prescription drugs. Kits containing prescription drugs may or may not have an NDC assigned to the kit itself.

This usage guideline describes the process for creating a kit that has an assigned NDC. If the kit does not have an assigned NDC, one of two options can be utilized:

- Kits that do not have their own NDC may be started in a `repackagedPedigree` element, using the manufacturer product code as a NMTOKEN instead of one of the values specified in the `ProductCodeValueTypeType`., or
- Each prescription drug contained within the kit may have its own pedigree.

Creating a pedigree for a kit with an assigned NDC employs the `repackagedPedigree` element to embed the pedigrees for each of the prescription drugs in the kit in the pedigree for the kit. The kit is tracked using the NDC of the kit, and the overall kit pedigree is updated with the transaction information and signed as the kit moves through the supply chain.

The pedigree flow is described for creating a kit with assigned NDC consisting of multiple prescription drugs, where the kit manufacturer initiates the pedigree for the kitted item. If a wholesaler were to create the kit, the same steps for `repackagedPedigree` would be followed.


pedigree
receivedPedigree Id="ReceivedPed-1"
documentInfo
pedigree
shippedPedigree Id="ShippedPed-1"
docmentInfo
repackagedPedigree
previousProducts
serialNumber
prerviousProductInfo
itemInfo
contactInfo
previousPedigrees
pedigree
...
productInfo
itemInfo
itemInfo
transactionInfo
senderInfo
recipientInfo
transactionIdentifier
signatureInfo
Signature (Repacker Signs: ShippedPed-1)
receivingInfo
signatureInfo
Signature (Wholesaler Signs: ReceivedPed-1)

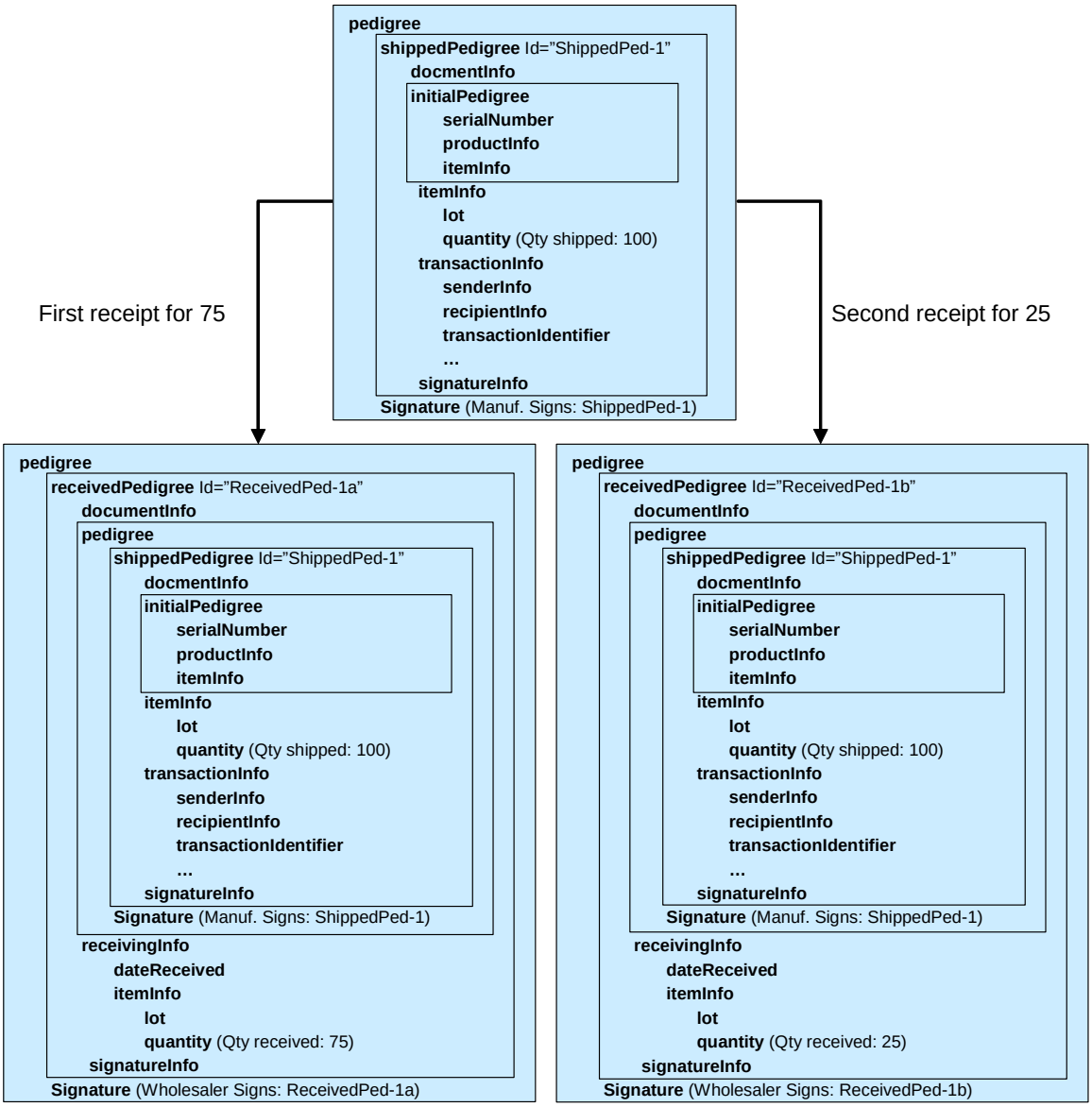
1032
1033

Step	Details
Kit manufacturer creates pedigree	• The kit manufacturer generates the `repackagedPedigree`, which contains the `previousProducts`, optionally includes the `previousPedigrees` elements that describe the source prescription drug items that are embedded in the kit, followed by the `productInfo` and `itemInfo` elements that describe the information about the newly created kit. This would include the NDC and product information for the kit itself. The `previousProducts` element describes the source items used to create the kit pedigrees and provides a reference to the `serialNumber` of the source pedigree if the product has one. The `previousPedigrees` contains the actual pedigrees for the source items using either the `pedigree, altPedigree` or `initialPedigree` element. This element is included only if the pedigrees for the source items are required to be in the kit pedigree. If the pedigrees for the source items are not required, the `previousPedigrees` may be omitted. The `productInfo` element describes the generic product information. The `itemInfo` identifies the specific repacked items represented by the pedigree. • Upon sale of the kit, the kit manufacturer adds transaction information for the sale of the kit and signs the pedigree. The kit manufacturer wraps the `repackagedPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo, itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

Step	Details
Wholesaler receives kit and pedigree	• When the wholesaler organization receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The wholesaler wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Received and Authenticated”). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.6 Partial Receipt of Products against Pedigree

The partial receipt of product against pedigree is described for a sale from a manufacturer to a wholesaler, when the manufacturer initiates the pedigree. The wholesaler receives the products in two partial shipments and updates each partial receipt against the original pedigree, resulting in a new received pedigree for each partial receipt.



1039

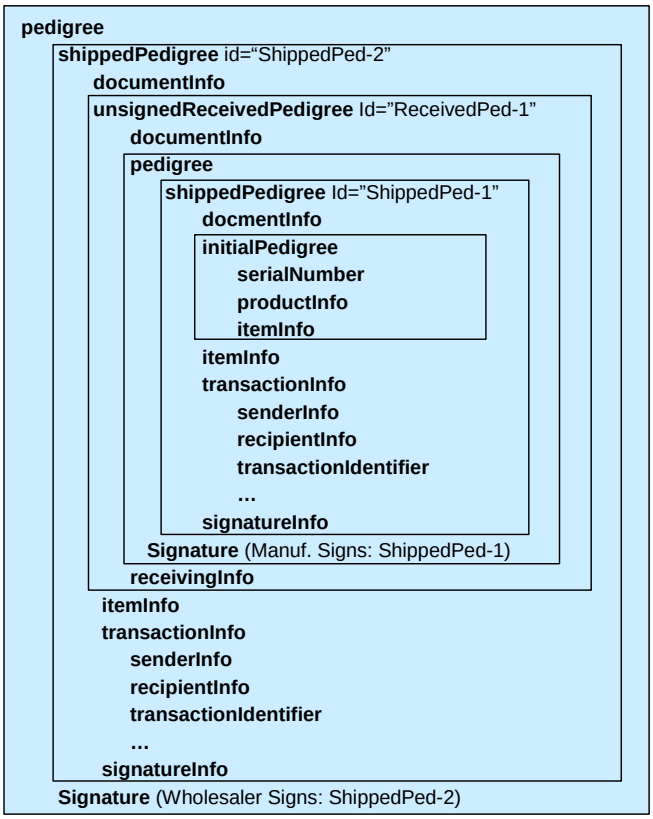
1040

Step	Details
Manufacturer creates pedigree	• The manufacturer generates the pedigree following the steps in section 12.1.1 Pedigree Flow Initiated by Manufacturer.

Step	Details
Wholesaler receives partial quantity against pedigree	• The wholesaler receives and authenticates the pedigree. When the wholesaler receives the first partial shipment, it appends the receiving information to the pedigree for the partial receipt quantity and signs the pedigree. The wholesaler wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date and `itemInfo`). The `itemInfo` sub element contains the lot number and quantity of the items received against the pedigree and product serial numbers if the products are serialized. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Received and Authenticated”). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.
Wholesaler receives remaining quantity against pedigree	• When the wholesaler receives the remainder of product against the pedigree, the wholesaler generates a new received pedigree and appends the receiving information to the pedigree for the remaining partial receipt quantity and signs the pedigree. The wholesaler wraps the `Pedigree` element for the original pedigree received in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date and `itemInfo`). The `itemInfo` sub element contains the lot number and quantity of the remaining items received against the pedigree and product serial numbers if the products are serialized. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Received and Authenticated”). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.7 Pedigree Receipt without Applying Receiving Signature

The flow for the receipt of a pedigree without signing the pedigree on inbound receipt is described. The pedigree is subsequently signed on the next outbound transaction.



1044

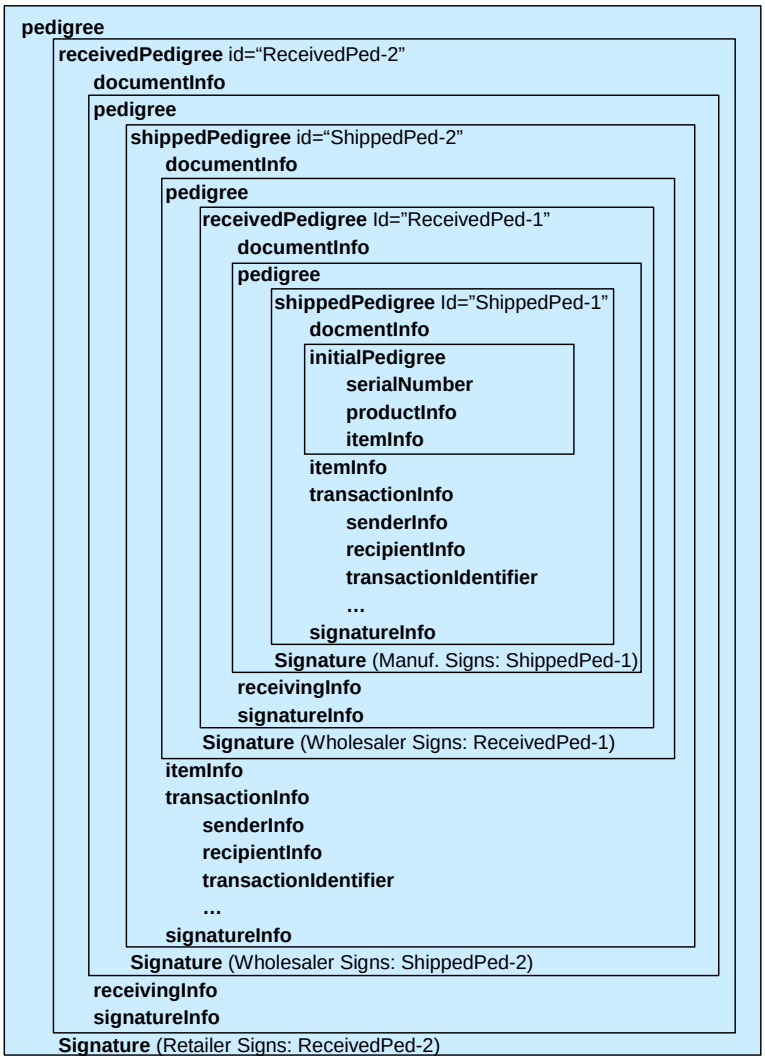
Step	Details
Manufacturer creates pedigree	• The manufacturer generates the pedigree following the steps in section 12.1.1 Pedigree Flow Initiated by Manufacturer.
Wholesaler receives product and pedigree	• When the wholesaler receives and authenticates the pedigree, it appends receiving information to the pedigree for the receipt quantity. The wholesaler wraps the `Pedigree` element in a `unsignedReceivedPedigree` element, and adds the `documentInfo` and `receivingInfo` elements as children of the new (wrapper) `unsignedReceivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date and `itemInfo`). The `itemInfo` sub element contains the lot number and quantity of the items received against the pedigree and product serial numbers if the products are serialized. • The `unsignedReceivedPedigree` element is assigned an identifier that is unique within the pedigree document.

Step	Details
Wholesaler ships product and pedigree to customer	• Upon sale of the item, the wholesaer adds transaction information for the sale to the customer and signs the pedigree. The wholesaler wraps the `unsignedReceivedPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo, itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

1045

1046 12.1.8 Pedigree Flow for Pedigree with Two Transactions

1047 The pedigree flow is described for a sale from a manufacturer to a wholesaler and then the
1048 wholesaler to a pharmacy,



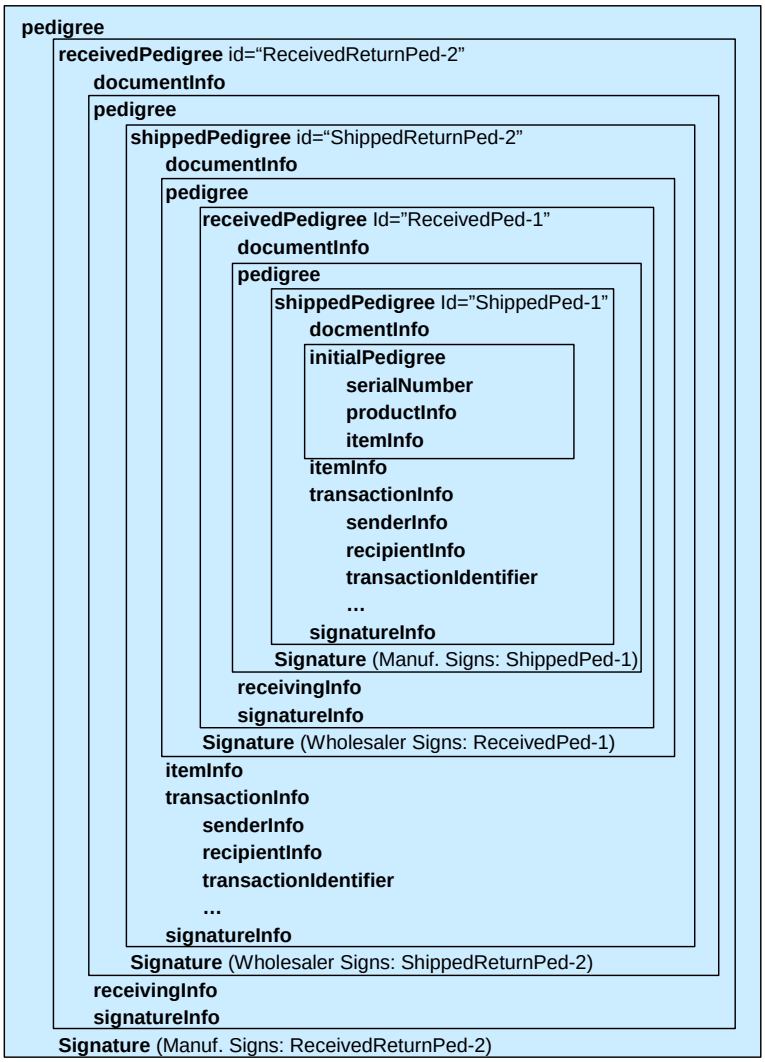
1049

Step	Details
Manufacturer creates pedigree	• The manufacturer generates the pedigree following the steps in section 12.1.1 Pedigree Flow Initiated by Manufacturer.
Wholesaler receives product and pedigree	• The wholesaler receives the pedigree following the steps in section 12.1.1 Pedigree Flow Initiated by Manufacturer.

Step	Details
Wholesaler ships pedigree to pharmacy DC	• Upon sale of the item, the wholesaer adds transaction information for the sale to the customer and signs the pedigree. The wholesaler wraps the `Pedigree` element for the signed `receivedPedigree` in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.
Pharmacy DC receives product and pedigree	• When the pharmacy DC receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The pharmacy DC wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Received and Authenticated”). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.9 Pedigree Flow for Pedigree with Return Transaction

The pedigree flow is described for a sale from a manufacturer to a wholesaler and then with a return from the wholesaler back to the manufacturer. The party making the return applies the return transaction to the pedigree.



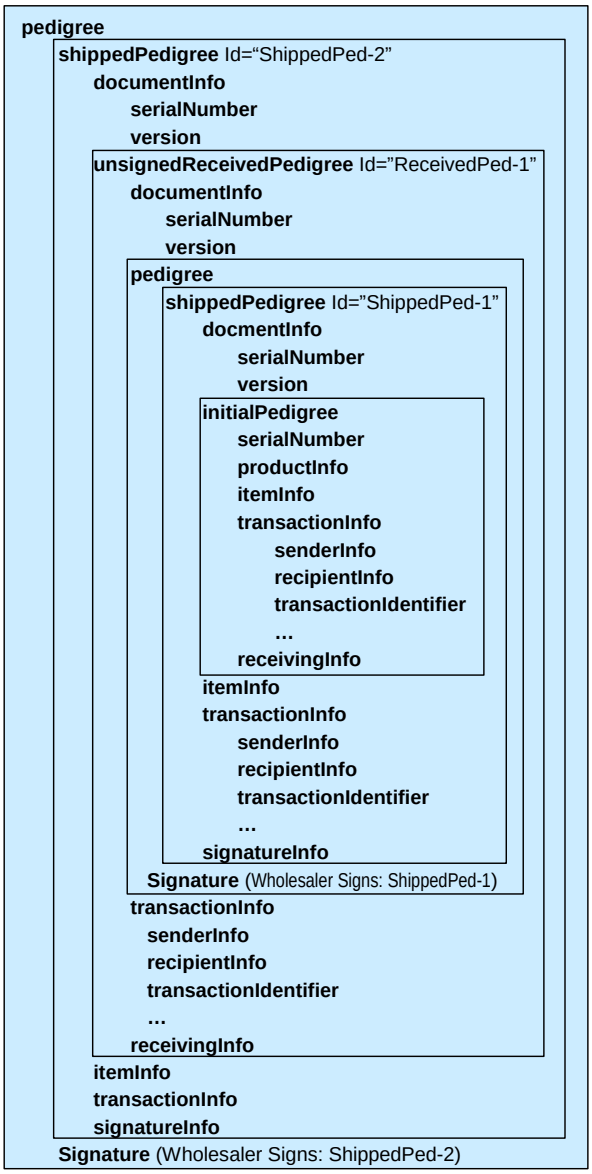
1055

Step	Details
Manufacturer creates pedigree	• The manufacturer generates the pedigree following the steps in section 12.1.1 Pedigree Flow Initiated by Manufacturer.
Wholesaler receives product and pedigree	• The wholesaler receives the pedigree following the steps in section 12.1.1 Pedigree Flow Initiated by Manufacturer.

Step	Details
Wholesaler later returns product and pedigree to manufacturer	• Upon return of the item, the wholesaer adds transaction information for the return to the manufacturer and signs the pedigree. The wholesaler wraps the `Pedigree` element for the signed `receivedPedigree` in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction as a return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.
Manufacturer receives return pedigree	• When the manufacturer receives and authenticates the pedigree, it appends receiving information to the pedigree and signs the pedigree. The manufacturer wraps the `Pedigree` element in a `receivedPedigree` element, and adds the `documentInfo`, `receivingInfo` and `signatureInfo` elements as children of the new (wrapper) `receivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date). The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Received and Authenticated”). • The `receivedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `receivedPedigree` element is then wrapped in a `Pedigree` element, and the inner `receivedPedigree` element is digitally signed using the `Signature` element. The signature references the named `receivedPedigree` element.

12.1.10 Pedigree Flow for Wholesaler Applied Return Transaction to Pedigree

The pedigree flow is described for a sale from a wholesaler to a pharmacy, and then a return from the pharmacy back to the wholesaler with the wholesaler updating the pedigree with the return transaction.



1061

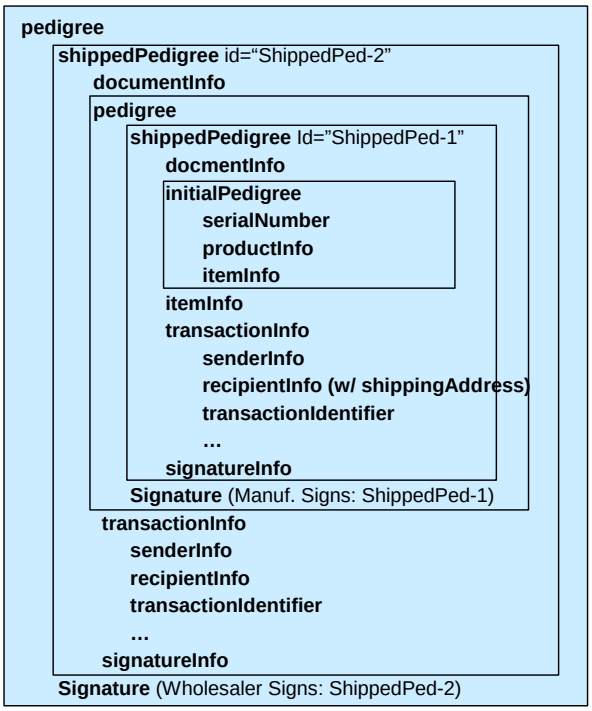
Step	Details
Wholesaler creates pedigree for item received from manufacturer	• The wholesaler generates the pedigree following the steps in section 12.1.2 Pedigree Flow Initiated by Wholesaler.
Pharmacy receives product and pedigree	• The endpoint pharmacy receives the pedigree and does not update or sign the pedigree.

Step	Details
Pharmacy returns product to wholesaler and wholesaler updates pedigree with return transaction	• Upon return of the item, the wholesaler adds the return transaction information to the pedigree. The wholesaler wraps the `Pedigree` element in a `unsignedReceivedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `receivingInfo`, and `attachment` elements as children of the new (wrapper) `unsignedReceivedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `transactionInfo` provides information about the return transaction. The `receivingInfo` provides information about the receipt transaction (e.g., receipt date and `itemInfo`). The `itemInfo` sub element contains the lot number and quantity of the items received against the pedigree and product serial numbers if the products are serialized. The optional `attachment` element can contained authentication material so that downstream trading partners do not have to manually authenticate this transaction. • The `unsignedReceivedPedigree` element is assigned an identifier that is unique within the pedigree document.
Wholesaler ships pedigree to a new customer	• Upon subsequent sale of the item, the wholesaer adds transaction information for the sale to the new customer and signs the pedigree. The wholesaler wraps the `unsignedReceivedPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo`, `itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

12.1.11 Pedigree Flow for a Manufacturer-initiated Drop Ship

The pedigree flow is described for a drop ship transaction brokered by wholesaler, where pharmacy purchases the product from the wholesaler, but the manufacturer ships the product directly to the pharmacy. In this scenario, the manufacturer initiates the start of the drop ship pedigree documenting the sales transaction from the manufacturer to the wholesaler with the shipping information indicating the direct shipment to the pharmacy. The wholesaler adds only the

second part of the drop ship transaction to the pedigree documenting the sales transaction from the wholesaler to the pharmacy.

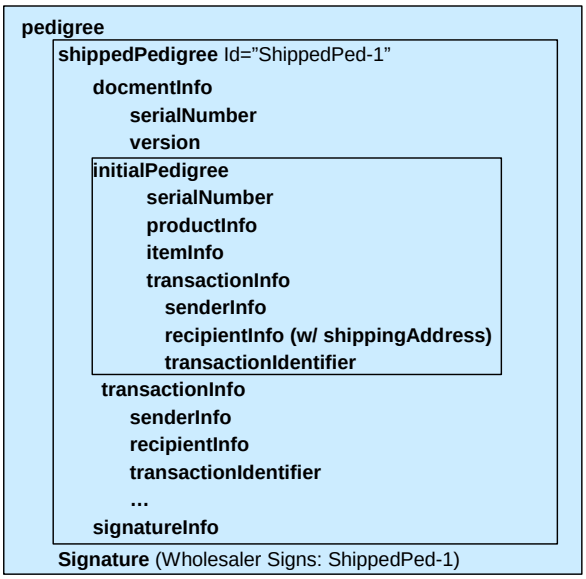


Step	Details
Manufacturer creates pedigree	• The manufacturer generates the `initialPedigree`, which contains the `serialNumber` (unique serial number), `productInfo` (generic product information) and `itemInfo` (identifies the specific items represented by the pedigree) elements. • Upon initiation of the drop shipment, the manufacturer adds transaction information for the sale to the wholesaler and shipping information to the pharmacy and signs the pedigree. The manufacturer wraps the `initialPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo, itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the sales transaction to the wholesaler, including information about the sender, recipient, transaction identifier and date, and transaction type. The `recipientInfo` element contains the `shippingAddress` for the pharmacy. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

Step	Details
Wholesaler receives pedigree (but no product) and updates pedigree with second part of drop ship transaction	• When the wholesaler organization receives the pedigree, it appends the transaction information for the sale from the wholesaler to the pharmacy to the pedigree and signs the pedigree. The wholesaler wraps the `Pedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `transactionInfo` describes the sales transaction to the pharmacy, including information about the sender, recipient, transaction identifier and date, and transaction type. The `senderInfo` element may contain the `shippingAddress` for the manufacturer. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., "Certified"). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

12.1.12 Pedigree Flow for a Wholesaler-initiated Drop Ship

The pedigree flow is described for a drop ship transaction brokered by wholesaler, where pharmacy purchases the product from the wholesaler, but the manufacturer ships the product directly to the pharmacy. In this scenario, the manufacturer does not provide the wholesaler with a pedigree and the wholesaler documents both parts of the drop ship transaction on the pedigree (assuming the wholesaler has access to this information).



Step	Details
Wholesaler creates pedigree for product drop shipped from manufacturer to pharmacy	• The wholesaler initiates the pedigree and adds information for the sale from the manfaucturer to the wholesaler and shipping information to the pharmacy. The wholesaler generates the `initialPedigree`, which contains the `serialNumber` (unique serial number), `productInfo` (generic product information), `itemInfo` (identifies the specific items represented by the pedigree) and `transactionInfo` (describes the sale from the manufacturer to the wholesaler. The `transactionInfo` element includes information about the sender, recipient, transaction identifier and date, and transaction type. The `recipientInfo` element contains the `shippingAddress` for the pharmacy. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. • The wholesaler adds the transaction information for the sale from the wholesaler to the pharmacy and signs the pedigree. The wholesaler wraps the `initialPedigree` element in a `shippedPedigree` element, and adds the `documentInfo`, `transactionInfo, itemInfo` and `signatureInfo` elements as children of the new (wrapper) `shippedPedigree` element. The `documentInfo` identifies the new unique serial number for the pedigree document. The `itemInfo` identifies the products that are the subject of this transaction. The `transactionInfo` describes the transaction, including information about the sender, recipient, transaction identifier and date, and transaction type. The `senderInfo` element may contain the `shippingAddress` for the manufacturer. The `transactionType` element defines whether the business transaction is a sale, transfer, or return. The `signatureInfo` provides information about the signer and the meaning of the signature (e.g., “Certified”). • The `shippedPedigree` element is assigned an identifier that is unique within the pedigree document. • The `shippedPedigree` element is then wrapped in a `Pedigree` element, and the inner `shippedPedigree` element is digitally signed using the `Signature` element. The signature references the named `shippedPedigree` element.

12.2 Usage Guidelines for Voiding and Altering Pedigrees

Some pedigree regulations (see US State of Florida Regulations) allow pedigrees to be altered or voided after they are transferred to downstream trading partners. These regulations contain specific requirements around this type of activity. The current revision of the EPCglobal Pedigree Standard does not contain a mechanism to automate the notification of trading partners when a void or alteration occurs. However, some non-binding best practices are provided as recommendations to assist the industry in handling pedigree alterations and voids until a later revision of this standard may include a way to automate these activities.

1) Pedigree voiding and alterations should be avoided if at all possible since they will create labor intensive activities at one or more trading partner sites.

2) The notification of trading partners that a pedigree has been altered or voided must be done manually (phone call, email, etc.) since there is no standard notification mechanism defined yet.

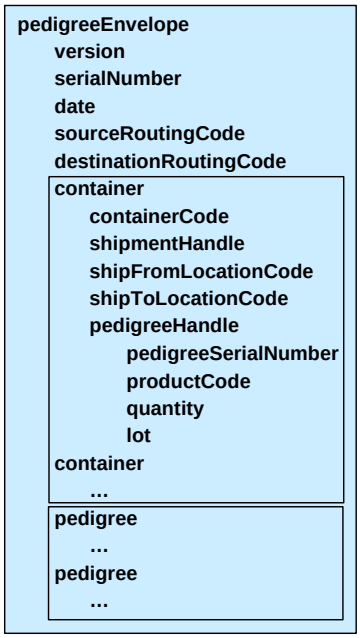
3) It is the responsibility of the trading partners to maintain a history of pedigree alterations and voids as specified by the various pedigree laws. Pedigree management software may assist with this.

4) Pedigree alterations and voids should be initiated only during the short window of time after the document has been transferred from one trading partner to another and prior to the inbound certification of the product received.

5) Recalls should typically never be used as a reason to void or alter a pedigree.

12.3 Usage Guidelines for Creating Pedigree Envelopes

This section explains how to use the `pedigreeEnvelope` element and its sub elements to create pedigree envelopes to transmit pedigrees to trading partners.



Step	Details
Create pedigree envelope for customer order	• The sending party generates the `pedigreeEnvelope`, which contains the `version`, `serialNumber`, `date`, `sourceRoutingCode` and `destinationRoutingCode` elements. The `version` contains the version of schema used. The `serialNumber` is the unique identifier of the pedigree envelope document. The `date` is the date the pedigree envelope was created and transmitted. The `sourceRoutingCode` and `destinationRoutingCode` are the location codes from which the pedigrees were transmitted and received. • Include the `container` element and its sub-elements if the additional information to facilitate the product-to-pedigree matching process is required. o Add a `container` element for the case or tote. To the `container` element add the `containerCode`, optionally add one or more `container` sub elements to represent items in sub-containers that have pedigrees, then add the `shipmentHandle`, `shipFromLocationCode`, and `shipToLocationCode` elements. The `containerCode` is required to identify the case. The `container` sub elements identify any sub-containers located in the container. The `shipmentHandle` identifies the shipment that the case is associated with. The `shipFromLocationCode` identifies the source of the shipment. The `shipToLocationCode` identifies the destination of the shipment. o Add one `pedigreeHandle` element for each pedigree associated with products in the case. Nested within the `pedigreeHandle`, the pedigree `serialNumber` is defined. If the items are serialized, each `itemSerialNumber` associated with the pedigree is listed. If the items are not serialized, the `productCode` is provided, followed by a list of `quantity` and `lot` pairs for each lot of product associated with the pedigree. o Repeat the above for each case in the shipment. • Add each `pedigree` representing each physical prescription drug item in the shipment to the `pedigreeEnvelope`.

12.3.1 Use of Container and PedigreeHandle Elements

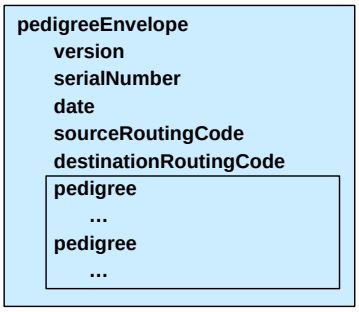
The `container` and `pedigreeHandle` elements provide the product to pedigree mapping information for non-serialized products to describe which pedigrees refer to which products in which cases. There are four key scenarios for this mapping information:

- No mapping information provided at all (e.g., the cases may not be serialized or the trading partners agree to exchange this information through other means or not to exchange it at all).
- Mapping information for a pedigree that refers to multiple products that are located in a single serialized case.
- Mapping information for a pedigree that refers to multiple products that are located in different serialized cases.
- Mapping information for multiple pedigrees for multiple products of the same NDC located in the same serialized case.

Pedigree envelopes should not be used to identify containers that do not have pedigreed items in them. This means that you should only include in the container element information about products that have pedigrees.

12.3.1.1 No Mapping Information

This section explains how to generate a `pedigreeEnvelope` with no mapping information at all (e.g., the cases may not be serialized or the trading partners agree to exchange this information through other means or not to exchange it at all).



Step	Details
Create pedigree envelope for customer order	• The sending party generates the `pedigreeEnvelope`, which contains the `version`, `serialNumber`, `date`, `sourceRoutingCode` and `destinationRoutingCode` elements. The `version` contains the version of schema used. The `serialNumber` is the unique identifier of the pedigree envelope document. The `date` is the date the pedigree envelope was created and transmitted. The `sourceRoutingCode` and `destinationRoutingCode` are the location codes from which the pedigrees were transmitted and received. • Add each `pedigree` representing each physical prescription drug item in the shipment to the `pedigreeEnvelope`.

12.3.1.2 Mapping for a Pedigree Referring to Products in a Single Serialized Case

This section explains how to generate a `pedigreeEnvelope` with mapping information for a pedigree that refers to multiple products that are located in a single serialized case.

Step	Details
Create pedigree envelope for customer order	• The sending party generates the `pedigreeEnvelope`, which contains the `version`, `serialNumber`, `date`, `sourceRoutingCode` and `destinationRoutingCode` elements. The `version` contains the version of schema used. The `serialNumber` is the unique identifier of the pedigree envelope document. The `date` is the date the pedigree envelope was created and transmitted. The `sourceRoutingCode` and `destinationRoutingCode` are the location codes from which the pedigrees were transmitted and received. • Include the `container` element and its sub-elements. o Add a `container` element for the case. To the `container` element add the `containerCode,` then add the `shipmentHandle, shipFromLocationCod and shipToLocationCode`. The `containerCode` identifies the case. The `shipmentHandle` identifies the shipment that the case is associated with. The `shipFromLocationCode` identifies the source of the shipment. The `shipToLocationCode` identifies the destination of the shipment. o Add a `pedigreeHandle` element for the pedigree associated with the products in the case. Nested within the `pedigreeHandle`, the pedigree `serialNumber` is defined. If the items are serialized, each `itemSerialNumber` associated with the pedigree is listed. If the items are not serialized, the `productCode` is provided, followed by a list of `quantity` and `lot` pairs for each lot of product associated with the pedigree. o Repeat the above for each case in the shipment. • Add each `pedigree` representing each physical prescription drug item in the shipment to the `pedigreeEnvelope`.

1130

1131 #### 12.3.1.3 Mapping for a Pedigree Referring to Products in Multiple Serialized
1132 Cases

1133 This section explains how to generate a `pedigreeEnvelope` with mapping information for a pedigree
1134 that refers to multiple products that are located in different serialized cases.

Step	Details
Create pedigree envelope for customer order	• The sending party generates the `pedigreeEnvelope`, which contains the `version`, `serialNumber`, `date`, `sourceRoutingCode` and `destinationRoutingCode` elements. The `version` contains the version of schema used. The `serialNumber` is the unique identifier of the pedigree envelope document. The `date` is the date the pedigree envelope was created and transmitted. The `sourceRoutingCode` and `destinationRoutingCode` are the location codes from which the pedigrees were transmitted and received. • Include the `container` element and its sub-elements for the first case. o To the `container` element add the `containerCode,` then add the `shipmentHandle, shipFromLocationCod and shipToLocationCode`. The `containerCode` identifies the case. The `shipmentHandle` identifies the shipment that the case is associated with. The `shipFromLocationCode` identifies the source of the shipment. The `shipToLocationCode` identifies the destination of the shipment. o Add a `pedigreeHandle` element for the pedigree associated with the products in the case. Nested within the `pedigreeHandle`, the pedigree `serialNumber` is defined. If the items are serialized, each `itemSerialNumber` for each item in this case associated with the pedigree is listed. If the items are not serialized, the `productCode` is provided, followed by a list of `quantity` and `lot` pairs for each lot of product in this case associated with the pedigree. • Include the `container` element and its sub-elements for the second case. o To the `container` element add the `containerCode,` then add the `shipmentHandle, shipFromLocationCod and shipToLocationCode`. The `containerCode` identifies the case. The `shipmentHandle` identifies the shipment that the case is associated with. The `shipFromLocationCode` identifies the source of the shipment. The `shipToLocationCode` identifies the destination of the shipment. o Add a `pedigreeHandle` element for the pedigree associated with the products in the case. Nested within the `pedigreeHandle`, the pedigree `serialNumber` is defined (this will be the same `serialNumber` as the pedigree in the the previous case since the products referred to by the pedigree span multiple cases). If the items are serialized, each `itemSerialNumber` for each item in this case associated with the pedigree is listed. If the items are not serialized, the `productCode` is provided, followed by a list of `quantity` and `lot` pairs for each lot of product in this case associated with the pedigree. • Add each `pedigree` representing each physical prescription drug item in the shipment to the `pedigreeEnvelope`.

1135

1136 **12.3.1.4 Mapping for Multiple Pedigrees Referring to Products of the Same**
1137 **NDC in the Same Serialized Case**

1138 This section explains how to generate a `pedigreeEnvelope` with mapping information for multiple
1139 pedigrees for multiple products of the same NDC located in the same serialized case.

Step	Details
Create pedigree envelope for customer order	• The sending party generates the `pedigreeEnvelope`, which contains the `version`, `serialNumber`, `date`, `sourceRoutingCode` and `destinationRoutingCode` elements. The `version` contains the version of schema used. The `serialNumber` is the unique identifier of the pedigree envelope document. The `date` is the date the pedigree envelope was created and transmitted. The `sourceRoutingCode` and `destinationRoutingCode` are the location codes from which the pedigrees were transmitted and received. • Include the `container` element and its sub-elements. o Add a `container` element for the case. To the `container` element add the `containerCode,` then add the `shipmentHandle, shipFromLocationCod and shipToLocationCode`. The `containerCode` identifies the case. The `shipmentHandle` identifies the shipment that the case is associated with. The `shipFromLocationCode` identifies the source of the shipment. The `shipToLocationCode` identifies the destination of the shipment. o Add a `pedigreeHandle` element for the first pedigree associated with the products in the case. Nested within the `pedigreeHandle`, the pedigree `serialNumber` is defined. If the items are serialized, each `itemSerialNumber` associated with the pedigree is listed. If the items are not serialized, the `productCode` is provided, followed by a list of `quantity` and `lot` pairs for each lot of product associated with the pedigree. o Add a `pedigreeHandle` element for the second pedigree associated with the products in the case. Nested within the `pedigreeHandle`, the pedigree `serialNumber` is defined. If the items are serialized, each `itemSerialNumber` associated with the pedigree is listed. If the items are not serialized, the `productCode` is provided, followed by a list of `quantity` and `lot` pairs for each lot of product associated with the pedigree. • Add each `pedigree` representing each physical prescription drug item in the shipment to the `pedigreeEnvelope`.

13 References

[1] Unified Drug Pedigree Coalition, “Additional Amendments and Discussion to Feb 04, 2005 Attribute List”, August 9, 2005.

[2] U.S. Department of Commerce, National Institute of Standards and Technology, “Digital Signature Standard (DSS)”, FIPS PUB 186-2, January 27, 2000.
For all FIPS standards documents, see http://www.itl.nist.gov/fipspubs/index.htm

[3] Florida Department of Health, “Draft Rules for the 8-17-2005 Workshop”, August 12, 2005.

[4] State of California, “Senate Bill No. 1307”, September 29, 2004. See http://www.leginfo.ca.gov/pub/03-04//bill/sen/sb_1301-1350/sb_1307_bill_20040929_chaptered.html

[5] Bartel, Boyer, Fox, LaMaccia and Simon, “XML-Signature Syntax and Processing”, IETF W3C, February 12, 2002. http://www.w3.org/TR/xmldsig-core/

[6] EPCglobal HLS Information Work Group-C Functional Requirements, May 5 2005.. http://eroom.uc-council.org/eRoom/facility/EPCglobalHealthcareLifeSciencesBAG/WORKING%20GROUPS/HLS%20BAG%20Information/Sub%20Group%20C~3A%20Data%20Ownership%2C%20Security%20and%20Retention/Completed%20Documents/Capabitility%20A%20-%20ePedigree/Requirements%20Tables/EPC%20Requirements%20Table%20IWG-C%20Final%20Draft%20HLS%20May%2005.xls. See Requirements 211, 213 and 214.

[7] ISO, “Rules for the structure and drafting of International Standards (ISO/IEC Directives, Part 2, 2004, 5th edition),” 2004. http://isotc.iso.org/livelink/livelink/fetch/2000/2122/3146825/4229629/4230450/4230456/ISO_IEC_Directives__Part_2__Rules_for_the_structure_and_drafting_of_International_Standards__2004__5th_edition___pdf_format_.pdf?nodeid=4230517&vernum=0

[8] EPCglobal Certificate profile. Ratified Specification 1.0. March 8, 2006. http://www.epcglobalinc.org/standards_technology/SAG%20Security%20Cert%20Profile%20March%208%202006.pdf

[9] IETF RFC 3280. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. R. Housley, W. Polk, W. Ford, D. Solo. April 2002. http://www.ietf.org/rfc/rfc3280.txt?number=3280

[10] IETF RFC 3279. Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. W. Polk, R. Housley, L. Bassham. April 2002. http://www.ietf.org/rfc/rfc3279.txt?number=3279

[11] ITU-T Recommendation X.509 version 3 (1997). "Information Technology - Open Systems Interconnection - The Directory: Authentication Framework" ISO/IEC 9594-8:1997.

[12] W3C Exclusive XML Canonicalization Version 1.0. J Boyer, D. Eastlake 3rd, J. Reagle. July 18, 2002. http://www.w3.org/TR/2002/REC-xml-exc-c14n-20020718/

[13] WS-I Basic Security Profile Version 1.0 (draft). A. Barbir, M. Gugin, M. McIntosh, K.S. Morrison. March 29, 2006. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2006-03-29.html

[14] W3C XML-Signature Syntax and Processing. M. Bartel, J. Boyer, B. Fox, B. LaMacchia, E. Simon. February 12, 2002. http://www.w3.org/TR/xmldsig-core

[15] National Drug Code Directory. US Food and Drug Administration. http://www.fda.gov/cder/ndc/index.htm

[16] ISO 3166-2 :1998 Codes for the representation of names and countries and their subdivisions – Part 2: Country subdivision code. September 23, 2003. http://www.iso.org/iso/en/prods-services/iso3166ma/04background-on-iso-3166/iso3166-2.html

[17] ISO 3166-1 alpha-2 :1997 Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes. September 23, 2003. http://www.iso.org/iso/en/prods-services/iso3166ma/index.html

[18] “Encode your XML documents in UTF-8”, Harold, Elliotte. August 30, 2005. http://www-128.ibm.com/developerworks/xml/library/x-utf8/?ca=dnt-635

14 APPENDIX A – Summary of Pedigree Data Elements (non-normative)

The following table provides a summary of the pedigree data elements. Refer to section 10 ePedigree Data Definition for a complete listing of all data elements and their full descriptions.

Document Information
Pedigree serial number
Item Information
Item serial number(s) of product(s) (if available)
Lot number
Expiration date
Quantity of saleable units in transaction
Product Information
Drug name
Manufacturer
Product code (e.g., the NDC number)
Dosage form
Strength
Container size
Alternate Pedigree Information
Attachment(s) to facilitate manual authentication such as EDI invoice or shipping document (includes mime type and data)
Alternate pedigree to convert pedigree of another format to the standard (includes mime type and data)
Transaction Information
Transaction identifier (for example, invoice or purchase order number)
Transaction document type (e.g., Invoice, Purchase order, Return authorization)
Date of transaction
Transaction type (e.g., sale, transfer, return)
Seller and Recipient Information
Business Address (see below)
Shipping Address (see below; used only if different than Business Address)
License number
License state or region

License agency
Contact Information for seller used for authentication of transaction (see below)
Business and Shipping Address
Business name
Street1
Street 2
City
State or Region
Postal Code
Country
Contact Information[1]
Contact Name
Contact Title
Contact Email
Contact Telephone
Contact URL (for automated authentication)
Receiving Information
Date received
Item Information (e.g., Lot, Quantity, Serial Numbers) for items in partial receipt[2]
Signer Information
Name of signer
Title of signer
Date of signature
Signature meaning (defines certification context such as certified outbound, received and authenticated inbound)
Digital Signature Information[3]
SignedInfo
SignatureValue
KeyInfo
SignatureProperties

[1] Item information for receipt is required only for partial receipts against a pedigree (e.g., pedigree was for 30 items of Lot A and only 20 items were received). Note that Florida does not allow partial receipts against a pedigree at this time. This information can be used in other locals that may allow partial receipts against a pedigree.

[2] Regulations require the authentication contact information for the seller only.

[3] Pedigree uses the W3C XML Digital Signature standard to represent a digital signature. The `SignatureValue` contains the actual value of the digital signature and the `KeyInfo` enables the recipients to obtain the key needed to validate the signature. Regulations require that both the message digest and reference to the public key be in the pedigree for each signature.

15 APPENDIX B – Glossary (non-normative)

Aggregate Pedigree	A pedigree for a collection of saleable items that share the same product information and prior chain of custody. The items in an aggregate pedigree may have different lot numbers.
ASN	Advanced Shipping Notice. This is a type of EDI transaction (transaction code 856). The ASN contains information regarding a specific shipment of product from a seller to a buyer and is transmitted prior to the arrival of the shipment. The fields, data and level of granularity (shipment, pallet, case, etc.) is determined by the trading partners
Authenticate	Affirmatively verify that each previous transaction listed on the pedigree has occurred and that the previous signatures or certifications are complete.
Certification	A sworn statement that the pedigree is a complete and accurate and that prior sales and distributions have been authenticated, if required
Digital Signature	A method for authenticating digital information analogous to ordinary physical signatures on paper, but implemented using techniques from the field of public-key cryptography. A digital signature method generally defines two complementary algorithms, one for signing and the other for verification, and the output of the signing process is also called a digital signature. The term electronic signature, although sometimes used for the same thing, has a distinct meaning in common law: it refers to any of several, not necessarily cryptographic, mechanisms for identifying the originator of an electronic message.
Digitally Signed	Digitally signed data is data that has been stored with the "identity" of an entity, and a digital signature intended to prove that the entity is the source of the data. The data has been digitally signed using the entity's private key (see “PKI”) in an attempt to make it practically impossible to forge or modify the data. In this document this refers to a digitally signed pedigree xml record.
DIN	Drug Identification Number. Canada’s drug code that is analogous to the US NDC (See “NDC”). Health Canada's Therapeutic Products Directorate assigns a single Drug Identification Number (DIN) for products with varying sizes, provided that all other product characteristics including product name, manufacturer's name, dosage form, route of administration, medicinal ingredient(s), and corresponding strength(s) are identical.
Dosage Form	Standard forms of drugs (AEROSOL, CAPSULE, GEL, PILL, TABLET) as defined by the FDA. The FDA currently defines 143 dosage forms.
Downstream Trading Partners	Customers of the current holder of the pedigreed drug product and their customer's customers.
Drug Pedigree Laws	Laws that require the maintenance of a record of a drug product’s chain of custody. These laws apply to specific jurisdictions (individual US States at this time) and vary from jurisdiction to jurisdiction.

Document Model Pedigree Law	A pedigree law that specifies or implies that a single pedigree document (paper or electronic) must be passed from each seller to each buyer from the originator to the final owner of the product and where each intermediate owner of the product must add information to the document.
EDI,	Electronic Data Interchange. The transfer of data between two companies using networks, such as the Internet, using approved standards such as ANSI's X12 standards.
Electronic Pedigree	An electronic record containing all data and information required by one or more pedigree laws including the necessary certifications. This Document defines the requirements for a standard representation of an electronic pedigree.
Electronic Verification	The process of determining the pedigree and each signature or certification is genuine, true and unaltered through the use of electronic technology such as digital signatures, hashing and public and private electronic code pairs. See "signature" and "PKI"
EPC	The Electronic Product Code™ (EPC) is a globally unique serial number that identifies an item in the supply chain. The EPC is a fundamental element of the EPCglobal Network.
EPC-IS	EPC Information Services enables users to exchange EPC-related data with trading partners through the EPCglobal Network. The EPC-IS is a fundamental element of the EPCglobal Network.
ePedigree	See "Electronic Pedigree"
Expiration Date	The date stamped on the drug by the manufacturer at which the manufacturer can still guarantee the full potency and safety of the drug.
FIPS (Federal Information Processing Standards)	Federal standards for information processing. As it relates to pedigree, the Federal standards for cryptographic, digital signature, and hashing technology and processing and as specified by some pedigree laws.
Immutable Document	A document that can not be changed or altered from its original state without detection or notification
Item	The lowest level of packaging that manufacturers offer a prescription product (bottle, vial, box, etc.) This typically does not refer to the unit dose that would be prescribed or administered to a patient or consumer (pill, milliliter, etc). Typically means "Saleable Item".
Legal Documents	A document that has a legal significance and signing it has legal consequences for those signing. A pedigree, whether paper or electronic is a legal document under document based pedigree laws.
License	A company's or individual's commercial licensing by a regulatory body to manufacture, package, sell, store, transport, buy or receive prescription drugs. Licenses are issued by the State and/or Federal Governments. An entity may have multiple licenses depending on the jurisdictions they are conducting commerce and the role they are filling.

Lot	A distinct group of inventory of a given drug. A drug's lot number is assigned by the manufacturer or by a repacker (See “Repackaging”). An individual lot number is assigned to drugs that are produced in a particular time period, on a particular manufacturing or packaging line, or that contain the same genealogy of ingredients.
Manual Authentication	Authentication of a prior transaction or change of ownership of a pedigreed drug product that did not use an acceptable Self-Authenticating technique (See Self-Authenticating Pedigree). This includes directly contacting the previous owners via phone or email, receiving a copy of signed paper pedigree, utilizing a web site provided by the previous owners to authenticate the transaction or some other method approved by the regulating body requiring the pedigree. By their nature, paper pedigrees can only be manually authenticated.
Message Digest	The output of cryptographic hash function which takes a message of any length as input and produces a fixed length string as output.
NABP VAWD	National Association of Boards of Pharmacy’s Verified-Accredited Wholesale Distributors™ (VAWD) is a wholesaler accreditation program.
NDC	Drug products in the United States are identified and approved using a three-segment number, called the National Drug Code (NDC). It is a universal product identifier for human drugs. FDA enters the full NDC number and the information submitted as part of the listing process into a database known as the Drug Registration and Listing System (DRLS). The segments of the NDC identify the manufacturer, the product and the saleable unit package size.
Non-Repudiation	Ensuring that parties to an event, transaction or legal document cannot later disclaim involvement.
Package Size (Pack Size)	The number of individual units of a drug product (pill or tablet count, milliliters or "cc", grams, etc.) included in each item (bottle, vial, box, etc.) See “Item”
Parent Pedigrees	The source pedigrees of manufacturer products used in a repacking operation.
Partial Receipt,	Receipt of only part of a shipment of a specific drug from a supplier for a specific sale or transfer. For example, a supplier provides a pedigree for single shipment of 100 units of a drug. 50 units are received on day one by the purchasing organization 50 units is received on day 4. Both receipts are partial receipts.
Drug Pedigree	A record of each distribution of a prescription drug from the sale by a Manufacturer through acquisition and sale by any Wholesale Distributor until final sale to a Pharmacy or other authorized person administering or dispensing the Prescription Drug.
Pedigree Envelope	An electronic document that encloses one or more pedigrees in a shipment.
Pedigree Layer	A sub-division of a pedigree that encloses a set of information and all previous layers. Each layer extends the pedigree by the new information

	added.
Peer-to-Peer Model	A model used to refer to a means of communicating electronically between two companies. EDI is a form of peer to peer communications. Each party must set up the communication link to the other party.
Potency	See "Strength"
Preexisting Pedigree	A pedigree provided by a supplier of a pedigreed drug shipped to a purchasing organization that describes the previous transactions and changes of ownership.
Prior Chain of Custody	The companies that owned the pedigreed drug prior to the current owner going back to the original sale by the manufacturer and all of the data and certifications required by pedigree laws.
Product Information	Information required in a pedigree related specifically to the identity of the drug. Requirements may vary by pedigree law. Data many include the name of the prescription drug, product code, pack size, quantity, its dosage form and strength, the expiration date(s) and the lot number(s).
Product Label	The label applied to an item package by the manufacturer or repackager. The contents and characteristics of this label must meet strict guidelines and regulations set forth by the FDA
Product-to-Pedigree Matching	The process of ensuring that the product attributes referenced on a pedigree match the attributes of the physical product that the pedigree references. This includes product, NDC if required, lot, quantity, etc.
Properties of the Certificate	Typically includes the public key being signed, A name, which can refer to a person, a computer or an organization, a validity period, the location URL of a revocation center.
Public Key Infrastructure (PKI)	A framework for creating a secure method for exchanging information based on public key cryptography. The foundation of a PKI is the certificate authority (CA), which issues digital certificates that authenticate the identity of organizations and individuals over a public system such as the Internet. The certificates are also used to sign messages which ensure that messages have not been tampered with.
Repackager	A business entity that buys finished products from the manufacture in the manufacturer's packaged form and repackages it into quantities to serve market requirements. For example, a repackager may buy 1000 pill count bottles and repackage them into 10 count blister packs. Repackaged product may or may not be assigned is own NDC number different from the original NDC number.
Saleable Item	The lowest packaging unit a wholesaler would ship a drug to fill and order. See "item"
Self-Authenticating Pedigree	An electronic pedigree that contains the necessary data and data structure so that when it is received it can be authenticated using automation. This Document defines the requirements for a standard of the necessary data and data structures to enable the creation of self-authenticating pedigrees.

Serial Number	A unique character string that identifies a specific instance of a drug packaging unit (case, inner pack, bottle, vial, box, etc.) from all other instances of that drug in the same packaging unit (serialized cases, serialized bottles, serialized vials). Few drugs contain serial numbers at the bottle, vial or item level (see item definition) today.
Signatures	The representation, in any acceptable media, by an individual or corporation that they are swearing or affirming that the information contained on this pedigree is accurate and complete. Each signature has a context and a finite scope.
Signer	The person authorized to bind the company by oath on legal documents.
Strength	The amount of active ingredient in the drug product. Each NDC has one specific strength or a specific combination of strengths. This is also referred to as potency
Transaction Information	Typically refers to mandatory information related to each transaction (sale, movement, return, transfer)of a pedigreed drug's distribution
Unified Drug Pedigree Coalition	A loosely affiliated group of representatives from the pharma industry and US state and federal drug regulatory agencies.
Unsigned	In this document, "unsigned" typically refers to a pedigree record that contains the necessary pedigree information but is not sign and is therefore not yet a legal document. Unsigned pedigree records could be provided by manufacturers who may not be required to provide pedigrees but choose to provide data to facilitate subsequent pedigree processes. Unsigned pedigrees can also be pedigrees that have been compiled within an organization but have not yet been signed.
Verify Products	See product-to-pedigree matching
Wrapped	A technique of enclosing information for the purpose of conveying a hierarchical relationship to other information. When information is "wrapped" by other information, it is considered to be wholly a part of the wrapping information.
Wrapper Element	The data structure that enables the wrapping technique. May also be used to identify the "wrapping" information.

1214

16 APPENDIX C – Electronic Pedigree XSD

The following is the XML schema definition for the pedigree format described in this document.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ped="urn:epcGlobal:Pedigree:xsd:1" targetNamespace="urn:epcGlobal:Pedigree:xsd:1"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
schemaLocation="http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/xmldsig-core-schema.xsd"/>
    <xs:annotation>
        <xs:documentation xml:lang="en">EPCglobal Inc., its members, officers, directors, employees, or
agents shall not be liable for any injury, loss, damages, financial or otherwise, arising from, related
to, or caused by the use of this document.  Use of said document does not guarantee compliance with
applicable state and/or federal laws.  User is responsible for the interpretation of and compliance with
applicable pedigree laws.  The use of said document shall constitute your express consent to the
foregoing disclaimer.</xs:documentation>
    </xs:annotation>
    <xs:complexType name="BaseExtensibleType">
        <xs:annotation>
            <xs:documentation xml:lang="en">base Type that  can be used for extensibility. Extensibility
is only allowed in NameSpace ##other ensures that any extensibility in targetNameSpace goes thru normal
EPC Global Vetting Process </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
            <xs:sequence/>
        </xs:sequence>
        <xs:anyAttribute processContents="lax"/>
    </xs:complexType>
    <xs:element name="pedigree">
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="ped:LayerType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="altPedigree">
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="ped:ForeignDataType">
                    <xs:attribute name="wasRepackaged" type="xs:boolean" default="false"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="initialPedigree" type="ped:InitialPedigreeType"/>
    <xs:element name="repackagedPedigree" type="ped:RepackagedPedigreeType"/>
    <xs:element name="unsignedReceivedPedigree" type="ped:UnsignedReceivedPedigreeType"/>
    <xs:element name="licenseNumber">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="state" type="xs:NMTOKEN" use="optional"/>
                    <xs:attribute name="agency" type="xs:string" use="optional"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="productCode" type="ped:ProductCodeType"/>
    <xs:complexType name="InitialPedigreeType">
        <xs:sequence>
            <xs:element name="serialNumber" type="xs:string" minOccurs="0"/>
            <xs:element name="productInfo" type="ped:ProductInfoType"/>
            <xs:element name="itemInfo" type="ped:ItemInfoType" maxOccurs="unbounded"/>
            <xs:element name="transactionInfo" type="ped:TransactionInfoType" minOccurs="0"/>
            <xs:element name="receivingInfo" type="ped:ReceivingInfoType" minOccurs="0"/>
```

```
            <xs:element ref="ped:altPedigree" minOccurs="0"/>
            <xs:element name="attachment" type="ped:ForeignDataType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ShippedPedigreeType">
        <xs:sequence>
            <xs:element name="documentInfo" type="ped:DocumentInfoType"/>
            <xs:choice>
                <xs:element ref="ped:initialPedigree"/>
                <xs:element ref="ped:repackagedPedigree"/>
                <xs:element ref="ped:unsignedReceivedPedigree"/>
                <xs:element ref="ped:pedigree"/>
            </xs:choice>
            <xs:element name="itemInfo" type="ped:ItemInfoType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="transactionInfo" type="ped:TransactionInfoType"/>
            <xs:element name="signatureInfo" type="ped:SignatureInfoType"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>
    <xs:complexType name="ReceivedPedigreeType">
        <xs:sequence>
            <xs:element name="documentInfo" type="ped:DocumentInfoType"/>
            <xs:choice>
                <xs:element ref="ped:pedigree"/>
                <xs:element ref="ped:initialPedigree"/>
            </xs:choice>
            <xs:element name="receivingInfo" type="ped:ReceivingInfoType"/>
            <xs:element name="signatureInfo" type="ped:SignatureInfoType"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>
    <xs:complexType name="UnsignedReceivedPedigreeType">
        <xs:sequence>
            <xs:element name="documentInfo" type="ped:DocumentInfoType"/>
            <xs:choice>
                <xs:element ref="ped:pedigree"/>
                <xs:element ref="ped:repackagedPedigree"/>
                <xs:element ref="ped:initialPedigree"/>
            </xs:choice>
            <xs:element name="transactionInfo" type="ped:TransactionInfoType" minOccurs="0"/>
            <xs:element name="receivingInfo" type="ped:ReceivingInfoType"/>
            <xs:element name="attachment" type="ped:ForeignDataType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:complexType>
    <xs:complexType name="RepackagedPedigreeType">
        <xs:sequence>
            <xs:element name="previousProducts" type="ped:PreviousProductType" maxOccurs="unbounded"/>
            <xs:element name="previousPedigrees" type="ped:PreviousPedigreeType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="productInfo" type="ped:ProductInfoType"/>
            <xs:element name="itemInfo" type="ped:ItemInfoType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="DocumentInfoType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="serialNumber" type="xs:string"/>
                    <xs:element name="version" type="xs:string"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ForeignDataType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="serialNumber" type="xs:string" minOccurs="0"/>
                    <xs:sequence minOccurs="0" maxOccurs="unbounded">
```

```
                        <xs:element name="mimeType" type="xs:string"/>
                        <xs:element name="encoding" type="ped:EncodingType"/>
                        <xs:element name="data" type="xs:string"/>
                    </xs:sequence>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ItemInfoType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="lot" type="xs:string"/>
                    <xs:element name="expirationDate" type="xs:date" minOccurs="0"/>
                    <xs:element name="quantity" type="xs:integer"/>
                    <xs:element name="itemSerialNumber" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ProductInfoType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="drugName" type="xs:string"/>
                    <xs:element name="manufacturer" type="xs:string"/>
                    <xs:element ref="ped:productCode" maxOccurs="unbounded"/>
                    <xs:element name="dosageForm" type="xs:string"/>
                    <xs:element name="strength" type="xs:string"/>
                    <xs:element name="containerSize" type="xs:string"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="TransactionInfoType">
        <xs:sequence>
            <xs:element name="senderInfo" type="ped:PartnerInfoType"/>
            <xs:element name="recipientInfo" type="ped:PartnerInfoType"/>
            <xs:element name="transactionIdentifier" type="ped:TransactionIdentifierType"/>
            <xs:element name="altTransactionIdentifier" type="ped:TransactionIdentifierType"
minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="transactionType" type="ped:TransactionTypeType"/>
            <xs:element name="transactionDate" type="xs:date"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PartnerInfoType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="businessAddress" type="ped:AddressType"/>
                    <xs:element name="shippingAddress" type="ped:AddressType" minOccurs="0"/>
                    <xs:element name="partnerId" type="ped:PartnerIdType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element ref="ped:licenseNumber" minOccurs="0" maxOccurs="unbounded"/>
                    <xs:element name="contactInfo" type="ped:ContactType" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="AddressType">
        <xs:sequence>
            <xs:element name="businessName" type="xs:string"/>
            <xs:element name="street1" type="xs:string"/>
            <xs:element name="street2" type="xs:string" minOccurs="0"/>
            <xs:element name="city" type="xs:string"/>
            <xs:element name="stateOrRegion" type="xs:string"/>
            <xs:element name="postalCode" type="xs:string"/>
            <xs:element name="country" type="xs:string"/>
            <xs:element name="AddressId" type="ped:AddressIdType" minOccurs="0" maxOccurs="unbounded"/>
```

```
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="LayerType">
        <xs:sequence>
            <xs:choice>
                <xs:element name="shippedPedigree" type="ped:ShippedPedigreeType"/>
                <xs:element name="receivedPedigree" type="ped:ReceivedPedigreeType"/>
            </xs:choice>
            <xs:element ref="ds:Signature"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="TransactionIdentifierType">
        <xs:sequence>
            <xs:element name="identifier" type="xs:string"/>
            <xs:element name="identifierType" type="ped:TransactionIdentifierTypeType"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ContactType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="name" type="xs:string"/>
                    <xs:element name="title" type="xs:string" minOccurs="0"/>
                    <xs:element name="telephone" type="xs:string" minOccurs="0"/>
                    <xs:element name="email" type="xs:string" minOccurs="0"/>
                    <xs:element name="url" type="xs:string" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ReceivingInfoType">
        <xs:sequence>
            <xs:element name="dateReceived" type="xs:date"/>
            <xs:element name="itemInfo" type="ped:ItemInfoType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PreviousProductType">
        <xs:sequence>
            <xs:element name="serialNumber" type="xs:string" minOccurs="0"/>
            <xs:element name="previousProductInfo" type="ped:PreviousProductInfoType"/>
            <xs:element name="itemInfo" type="ped:ItemInfoType" maxOccurs="unbounded"/>
            <xs:element name="contactInfo" type="ped:ContactType"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PreviousPedigreeType">
        <xs:choice>
            <xs:element ref="ped:pedigree"/>
            <xs:element ref="ped:initialPedigree"/>
            <xs:element ref="ped:altPedigree"/>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="SignatureInfoType">
        <xs:complexContent>
            <xs:extension base="ped:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="signerInfo" type="ped:ContactType"/>
                    <xs:element name="signatureDate" type="xs:dateTime"/>
                    <xs:element name="signatureMeaning" type="ped:SignatureMeaningType"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="PreviousProductInfoType">
        <xs:sequence>
            <xs:element name="drugName" type="xs:string" minOccurs="0"/>
            <xs:element name="manufacturer" type="xs:string"/>
            <xs:element name="productCode" type="ped:ProductCodeType"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ProductCodeType">
```

```
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="type" type="ped:ProductCodeValueType" use="required"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
        <xs:simpleType name="ProductCodeValueType">
            <xs:union memberTypes="xs:NMTOKEN ped:ProductCodeValueTypeType"/>
        </xs:simpleType>
        <xs:complexType name="PartnerIdType">
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="type" type="ped:PartnerIdValueType" use="required"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
        <xs:simpleType name="PartnerIdValueType">
            <xs:union memberTypes="xs:NMTOKEN ped:PartnerIdValueTypeType"/>
        </xs:simpleType>
        <xs:complexType name="AddressIdType">
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute name="type" type="ped:AddressIdValueType" use="required"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
        <xs:simpleType name="AddressIdValueType">
            <xs:union memberTypes="xs:NMTOKEN ped:AddressIdValueTypeType"/>
        </xs:simpleType>
        <xs:simpleType name="TransactionIdentifierTypeType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="InvoiceNumber"/>
                <xs:enumeration value="PurchaseOrderNumber"/>
                <xs:enumeration value="ShippingNumber"/>
                <xs:enumeration value="ReturnAuthorizationNumber"/>
                <xs:enumeration value="Other"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="SignatureMeaningType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="Certified"/>
                <xs:enumeration value="Received"/>
                <xs:enumeration value="Authenticated"/>
                <xs:enumeration value="ReceivedAndAuthenticated"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="TransactionTypeType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="Sale"/>
                <xs:enumeration value="Return"/>
                <xs:enumeration value="Transfer"/>
                <xs:enumeration value="Other"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="ProductCodeValueTypeType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="NDC442"/>
                <xs:enumeration value="NDC532"/>
                <xs:enumeration value="NDC541"/>
                <xs:enumeration value="NDC542"/>
                <xs:enumeration value="GTIN"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="EncodingType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="base64binary"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="PartnerIdValueTypeType">
            <xs:restriction base="xs:string">
```

```
            <xs:enumeration value="GLN"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="AddressIdValueTypeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="GLN"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

17 APPENDIX D – Pedigree Envelope XSD

The following is the XML schema definition for the pedigree envelope format described in this document.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ped="urn:epcGlobal:Pedigree:xsd:1"
xmlns:pedenv="urn:epcGlobal:PedigreeEnvelope:xsd:1.1"
targetNamespace="urn:epcGlobal:PedigreeEnvelope:xsd:1.1" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:import namespace="urn:epcGlobal:Pedigree:xsd:1" schemaLocation="PedigreeSchema_20061221.xsd"/>
    <xs:annotation>
        <xs:documentation xml:lang="en">EPCglobal Inc., its members, officers, directors, employees, or
agents shall not be liable for any injury, loss, damages, financial or otherwise, arising from, related
to, or caused by the use of this document.  Use of said document does not guarantee compliance with
applicable state and/or federal laws.  User is responsible for the interpretation of and compliance with
applicable pedigree laws.  The use of said document shall constitute your express consent to the
foregoing disclaimer.</xs:documentation>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation xml:lang="en">This schema needs to import the pedigree
schema.</xs:documentation>
    </xs:annotation>
    <xs:complexType name="BaseExtensibleType">
        <xs:annotation>
            <xs:documentation xml:lang="en">base Type that  can be used for extensibility. Extensibility
is only allowed in NameSpace ##other ensures that any extensibility in targetNameSpace goes thru normal
EPC Global Vetting Process </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:anyAttribute processContents="lax"/>
    </xs:complexType>
    <xs:element name="pedigreeEnvelope">
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="pedenv:PedigreeEnvelopeType"/>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="PedigreeEnvelopeType">
        <xs:complexContent>
            <xs:extension base="pedenv:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="version" type="xs:string" minOccurs="0"/>
                    <xs:element name="serialNumber" type="xs:string"/>
                    <xs:element name="date" type="xs:date" minOccurs="0"/>
                    <xs:element name="sourceRoutingCode" type="xs:string" minOccurs="0"/>
                    <xs:element name="destinationRoutingCode" type="xs:string" minOccurs="0"/>
                    <xs:element name="container" type="pedenv:ContainerType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:any namespace="urn:epcGlobal:Pedigree:xsd:1" processContents="skip"
maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="ContainerType">
        <xs:complexContent>
            <xs:extension base="pedenv:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="containerCode" type="xs:string" nillable="true"/>
                    <xs:element name="container" type="pedenv:ContainerType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="shipmentHandle" type="xs:string" minOccurs="0"/>
                    <xs:element name="shipFromLocationCode" type="xs:string" minOccurs="0"/>
                    <xs:element name="shipToLocationCode" type="xs:string" minOccurs="0"/>
```

```
                    <xs:element name="pedigreeHandle" type="pedenv:PedigreeHandleType" minOccurs="0"
maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="PedigreeHandleType">
        <xs:complexContent>
            <xs:extension base="pedenv:BaseExtensibleType">
                <xs:sequence>
                    <xs:element name="serialNumber" type="xs:string"/>
                    <xs:element name="itemSerialNumber" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="productCode" type="ped:ProductCodeType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:sequence minOccurs="0">
                        <xs:element name="quantity" type="xs:integer"/>
                        <xs:element name="lot" type="xs:string"/>
                    </xs:sequence>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```