Welcome to
Visa XML Invoice Implementation Guide

The *Visa XML Invoice Implementation Guide* is a new publication for Visa members and processors, industry vendors, software developers and system providers who are coding invoice forms for commercial clients that may be processed through VisaNet. It includes the following information:

- Visa Extensible Markup Language (XML) Invoice Document Structure and Details
- Element Dictionary
- Sector-Specific Mapping
- Code Lists
- Example invoices, Document Type Defininition (DTD) and XML Stylesheet

The Visa *Confidential* label indicates the information in this document is intended for use by Visa employees, member banks, and external business partners that have agreed to the terms of the XML Invoice Specification License Agreement with Visa. This information is not for public release under any other condition.

If you have comments concerning how this manual could better meet your needs, please e-mail buspubs@visa.com. Please provide the manual title and effective date in your correspondence. Your opinion is important to us.

Effective: 31 January 2000

Visa XML Invoice

Implementation Guide

Effective: 31 January 2000

Notice: Permission to copy and implement the material contained herein is granted only pursuant to separate license from Visa International Service Association and is subject to the conditions that (i) any copy or re-publication must bear this legend in full, (ii) any derivative work must bear a notice that it is not the Visa XML Invoice Specification published by Visa, and (iii) Visa shall have no responsibility or liability whatsoever to any other party arising from the use or publication of the material contained herein.

Visa makes no representation or warranty regarding whether any particular physical implementation of any part of this Specification does or does not violate, infringe, or otherwise use the patents, copyrights, trademarks, trade secrets, know-how, and/or other intellectual property of third parties, and thus any person who implements any part of this Specification should consult an intellectual property attorney before any such implementation. Any party seeking to implement this Specification is solely responsible for determining whether their activities require a license to any technology including, but not limited to, patents on public key encryption technology. Visa International Service Association shall not be liable for any party's infringement of any intellectual property right.

Contents

About This Manual

Chapter 1 • XML Invoice Document

Chapter 2 • XML Invoice Document Structure

Chapter 3 • Element Details

Appendix A • The Visa XML Invoice DTD

Appendix B • Frequently Asked Questions

Appendix C • Code List Reference

Appendix D • Tax Treatments

Appendix E • The Visa XML Invoice Stylesheet

About This Manual

Businesses need advanced technology solutions to enable them to use enhanced electronic purchase data to monitor and manage expenses. This enhanced data consists of detailed information over and above that contained in a typical electronic payment transaction. These businesses are increasingly requesting transmission of enhanced data from various merchant segments. To meet this need, Visa has adopted the Extensible Markup Language (XML) enhanced data message format to provide a standard means of capturing and delivering enhanced data.

This publication provides the information necessary to implement the Visa XML invoice document. The specification explains in detail the file structure, the business usage of the elements, and all the elements and attributes of the Visa XML invoice document.

Background

In an effort to address the incompatibility between systems and industries that process various business enhanced data elements, technology vendors, consortiums and standards bodies have adopted the Extensible Markup Language (XML). XML addresses a broad need in conducting e-business in a consistent and globally-interoperable manner. Many view XML as a more flexible and capable solution to Electronic Data Interchange (EDI).

In 1998, XML 1.0 was adopted by the industry standards body W3C. The goal of W3C is to develop common protocols that promote e-business evolution and ensure its interoperability. W3C is comprised of over 360 technology innovators including Sun Microsystems, IBM, Microsoft, CommerceOne, and many other internet and e-business solutions providers.

In cooperation with some of these same technology leaders, Visa developed the XML Invoice standard in 1999. This document contains important invoice, itinerary and receipt data elements, initially targeted toward business-to-business procurement, airline and travel, hotel and lodging, and car rental market segments.

With significant industry and software provider endorsement, XML enables development of end-to-end business solutions more quickly than other formats. Designed to accommodate various data needs among industry suppliers, XML is highly flexible and adaptable. Because it is “extensible,” XML can grow and change as business dictates.

While XML is a standard, adoption and implementation of the standard will take place over a migration period. Developers of new systems can adopt the XML Invoice standard immediately. Software developers, integrators, processors and solutions providers should plan to integrate XML into their systems as soon as possible.

This specification provides the necessary information to begin implementing XML invoice documents.

Audience

The intended audience for this publication is composed of Visa members and processors, industry vendors, software developers, and system providers who are coding invoice forms for commercial clients that may be processed through VisaNet.

Assumptions

This specification provides no explanation of XML itself. It is assumed that the reader has sufficient knowledge of XML to know how an XML file is structured, the function of a Document Type Definition (DTD), the use of elements and attributes, etc.

Scope of This Publication

This publication contains the following chapters:

- **About This Manual**—This chapter provides the background for the development of the XML Invoice standard and an overview of this publication.
- **Chapter 1, The XML Invoice Document**—This chapter introduces the XML invoice document and describes the use of attributes, elements, and how code lists are implemented.

- **Chapter 2, XML Invoice Document Structure**—This chapter describes the structure of the XML Invoice Document as a hierarchical tree, showing the iteration of each element, its constraints and conditions. The chapter also explains in detail each logical section of the Invoice Document, and the business usage of each element.
- **Chapter 3, Element Details**—This chapter provides an alphabetical listing of all elements used in the Invoice Document, giving technical details such as data formats and code lists.
- **Chapter 4, Sector-Specific Mapping**—This chapter documents mapping data items specific to the Lodging, Car Rental, and Passenger Itinerary market sectors.
- **Appendix A, The Visa XML Invoice DTD**—This appendix provides the full text of the Document Type Definition that supports the examples shown in Chapter 4.
- **Appendix B**, **Frequently Asked Questions**—This appendix provides answers to FAQs about this message format and this guide.
- **Appendix C**, **Code List Reference**—This appendix provides a summary of the standard code lists used in the Visa XML Invoice Document.
- **Appendix D, Tax Treatments**—This appendix provides details and examples for each of the allowable Tax Treatment types.
- **Appendix E, The Visa XML Invoice Stylesheet**—This appendix provides the full text of the stylesheet that supports the invoice examples shown in Chapter 4.

Using This Specification

The file structure listing in Chapter 2 (Table 2–1) may be a useful starting point to identify the overall structure of the XML document hierarchy. Each of the three top level groupings identified in the file structure, the InvoiceHeader, InvoiceDetails and InvoiceSummary, are then described in greater detail. Additional detail is also provided in this chapter for the other major element groups.

The Element Dictionary in Chapter 3 is sequenced alphabetically for ease of reference. Chapter 3 provides the details for each element and any associated attributes. Note that an element may occur at different points in the document's hierarchy. Thus the table in Chapter 2 shows whether the element is mandatory or optional, and how many repeats it may have, for each specific position at which it might occur in the structure.

Excerpts from the example XML invoices given in Chapter 4, and throughout this guide, are also used to illustrate specific points, in varying levels of detail, wherever relevant in order to aid understanding.

A FAQ in Appendix B explains why certain design and structuring decisions were taken in constructing the DTD, this Implementation Guide, and the stylesheet.

Technical Pack

The Technical Pack associated with this guide includes the latest versions of the Visa XML Invoice DTD, the Visa XML Invoice Stylesheet, and example invoice document files in a zipped file. This file is downloadable from www.visa.com. The examples, readable in Microsoft Internet Explorer version 5, correspond to the screen shots captured in Chapter 4 of this guide. The Technical Pack may contain additional examples not illustrated in this guide, and may also contain a later iteration of the DTD and stylesheet that will be linked to the example files included with them.

Document Conventions

The following table shows the document conventions used in this guide.

Document Conventions Table

Document Convention	Purpose In This Guide
ALL UPPERCASE LETTERS	Drive letters, subdirectory names, file names; system statuses, modes, and states; certain abbreviations and acronyms.
`Letter Gothic` typeface	Entries typed at the keyboard, messages displayed by the system, and the typeface used to re-create screen captures and sample code in text.
EXAMPLE	Identifies an example of what the accompanying text describes or explains.
IMPORTANT	Highlights important information in the text.
italics	Document titles; emphasis.
“text in quote marks”	Section names referenced in a chapter.
Note:	Provides more information about the preceding topic.

Related Documents

For further information on the Visa XML Invoice Document and enhanced data processing within VisaNet, refer to:

- *XML Invoice Technical Pack*, January 2000.
- *Visa Global Enhanced Data Service Member Interface Specification*, January 2000.

For further information on the generic XML Invoice DTD, refer to:

- *General XML Invoice Implementation Guide*, January 2000.
- *XML Invoice Technical Pack*, January 2000.

XML Invoice Document 1

Invoice details flow from specific merchants to the corporate clients who are buying their goods and services. The Visa XML invoice document satisfies very broad market needs and encompass both online and physical world activities, addressing established trading partners and ad hoc purchases, across every possible country, sector, and tax and accounting regime.

The Visa XML invoice document is based on CommerceOne's CBL v2.0. Several enhancements and changes have been made—some as a result of experience in implementing business-to-business electronically-traded invoices, and some as a result of specific Visa requirements. The XML invoice Document Type Definition (DTD) has been developed to provide all the functionality required for Visa and each sector's specific requirements, as well as to be suitable for use as a standard, non-Visa invoice.

Support has been specifically provided for European Union Value-Added Tax (VAT) requirements. Further verification is required to ensure this is sufficient for the needs of the more than 100 countries where VAT is applicable.

The Visa XML invoice includes specific elements for simple data elements that are common in an invoice, for example, the specific InvoiceDate element. The Visa XML invoice also provides generic elements, such as Date, that have a coded function qualifier to explain their function. This structure enables the invoice document to be extended without amending the DTD and still maintain its simplicity.

The latest version of the Visa XML invoice DTD and stylesheet are available in the Visa XML Invoice Technical Pack at www.visa.com.

Use of Code Lists

The function qualifiers are, wherever possible, based on international standard code lists. The recommended codes are documented in the Element Dictionary. Only the codes documented in the Element Dictionary in Chapter 3, Element Details, and in Chapter 4, Sector-Specific Mapping, should be used.

Where codes are used, they are always included in an element's stdValue attribute, rather than as an element's data value. Wherever there is a stdValue attribute, there is also a corresponding stdName attribute that specifies the code's details, such as the body that administers the code and the code list. The administration body and the code are delimited by a colon—for example, UNTDID:2475. These values are defaulted in the DTD. If an element's stdValue attribute also has a default value, this value will also be defaulted in the DTD.

When an attribute has a default value, it is not necessary to include that attribute within the element in the XML Document. Therefore

```
<InvoiceTreatment stdValue=”P” stdName=”VISA:INVT”/>
```

can simply be included in the document as

```
<InvoiceTreatment/>.
```

To allow the extensibility of the invoice DTD, most code lists are not included in the attribute definition in the DTD. Some elements' stdValue codes, however, are included in the DTD as a fixed list, and the stdName value is fixed, too—for example, the InvoiceTreatment element. Refer to the Element Dictionary for further details of those elements with fixed lists.

The usage of elements, attributes and code lists that are specific to a sector are documented in Chapter 4, Sector-Specific Mapping.

EXAMPLE: NO EXPLICITLY INCLUDED DEFAULT CODED VALUES

The following example shows a Lodging invoice with a single line item. It does not explicitly include any stdValue or stdName values, which means that the default values will apply.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml:stylesheet type="text/xsl" href="invoice.1.0.xsl"?>
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<Invoice sectorUsageVersion="1">
     <InvoiceHeader>
          <InvoiceType/>
          <InvoiceStatus/>
          <TaxTreatment/>
          <DiscountTreatment/>
          <InvoiceTreatment/>
          <InvoiceNumber>B003983</InvoiceNumber>
          <InvoiceDate>1999-02-11</InvoiceDate>
```

```
<Currency stdValue="DEM"/>
<Party stdValue="SU">
    <PartyID>5011234567890</PartyID>
    <Name>
        <Name1>Crowne International</Name1>
        <Name2>Frankfurt</Name2>
    </Name>
    <Street>
        <Street1>2022 Market Street</Street1>
    </Street>
    <PostalInfo>
        <City>Frankfurt</City>
        <CountrySubEntity/>
        <PostalCode>69500</PostalCode>
        <Country>Germany</Country>
    </PostalInfo>
    <Contact>
        <TelNum>+49 812 1234 222</TelNum>
        <Function>Accounts Dept</Function>
    </Contact>
    <Ref>DE1234567890</Ref>
    <Ref stdValue="XA">98398351</Ref>
</Party>
<Party stdValue="BY">
    <PartyID>LC100</PartyID>
    <Name>
        <Name1>Tom Proctor</Name1>
        <Name2>eCommerce Department</Name2>
        <Name3>Large Company Inc.</Name3>
    </Name>
    <Street>
        <Street1>Metro 1</Street1>
        <Street2>Metro Boulevard</Street2>
        <Street3>Ludgate Circus</Street3>
    </Street>
    <PostalInfo>
        <City>Foster City</City>
        <CountrySubEntity>CA</CountrySubEntity>
        <PostalCode>95118</PostalCode>
        <Country>USA</Country>
    </PostalInfo>
    <Contact>
        <Name1>Tom Proctor</Name1>
    </Contact>
    <Ref>CA12345678901234</Ref>
</Party>
<Party stdValue="PI">
    <PartyID>CROWNE002</PartyID>
    <Name>
        <Name1>CROWNE HOTEL GROUP</Name1>
    </Name>
    <PostalInfo>
        <City>HAMBURG</City>
        <PostalCode>00000</PostalCode>
        <Country>DE</Country>
    </PostalInfo>
```

```
            <Ref stdValue="ADQ">200000</Ref>
        </Party>
        <PONum>PO00001</PONum>
        <Ref stdValue="ADQ">LG</Ref>
        <Ref stdValue="AWE">98345</Ref>
        <Ref stdValue="RMNO" stdName="VISA:REF">908</Ref>
        <Ref stdValue="RSNO" stdName="VISA:REF">212</Ref>
        <Ref stdValue="RMRT" stdName="VISA:REF">100.00</Ref>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-20T14:11:54</Date>
        <Date stdValue="END" stdName="VISA:">1999-02-11T08:30:12</Date>
    </InvoiceHeader>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>1</LineItemNum>
            <PartNumDetail>
                <PartNum>100</PartNum>
                <PartDesc>Room Charge</PartDesc>
            </PartNumDetail>
            <PartNumDetail stdValue="CC">
                <PartNum>H100</PartNum>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>100.00</UnitPrice>
        <LineItemSubtotal>100.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>15.00</TaxPercent>
            <TaxAmount>15.00</TaxAmount>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>15.00</TaxPercent>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>100.00</NetValue>
            <TaxValue>15.00</TaxValue>
            <GrossValue>115.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>115.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>1999-02-11</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
</Invoice>
```

EXAMPLE: ALL CODED VALUES EXPLICITY INCLUDED

This example shows the same invoice as above, with all the default stdValue and stdName attributes explicitly (unnecessarily) included in the data.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml:stylesheet type="text/xsl" href="invoice.1.0.xsl"?>
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<Invoice sectorUsageVersion="1" >
   <InvoiceHeader>
      <InvoiceType stdValue="380" stdName="UNTDID:1001" />
      <InvoiceStatus stdValue="9" stdName= "UNTDID:1225"/>
      <TaxTreatment stdValue="NIL" stdName="VISA:TAXT" />
      <DiscountTreatment stdValue="TN" stdName="VISA:DSCT"/>
      <InvoiceTreatment stdValue="P" stdName="VISA:INVT" />
      <InvoiceNumber>B003983</InvoiceNumber>
      <InvoiceDate>1999-02-11</InvoiceDate>
      <Currency stdValue="DEM" stdName="ISO:4217"/>
      <Party stdValue="SU" stdName="UNTDID:3035">
         <PartyID>5011234567890</PartyID>
         <Name>
            <Name1>Crowne International</Name1>
            <Name2>Frankfurt</Name2>
         </Name>
         <Street>
            <Street1>2022 Market Street</Street1>
         </Street>
         <PostalInfo>
            <City>Frankfurt</City>
            <CountrySubEntity/>
            <PostalCode>69500</PostalCode>
            <Country>Germany</Country>
         </PostalInfo>
         <Contact>
            <TelNum>+49 812 1234 222</TelNum>
            <Function>Accounts Dept</Function>
         </Contact>
         <Ref stdValue="VA" stdName="UNTDID:1153">DE1234567890 </Ref>
         <Ref stdValue="XA" stdName="UNTDID:1153">98398351</Ref>
      </Party>
      <Party stdValue="BY" stdName="UNTDID:3035">
         <PartyID>LC100</PartyID>
         <Name>
            <Name1>Tom Proctor</Name1>
            <Name2>eCommerce Department</Name2>
            <Name3>Large Company Inc.</Name3>
         </Name>
         <Street>
            <Street1>Metro 1</Street1>
            <Street2>Metro Boulevard</Street2>
            <Street3>Ludgate Circus</Street3>
         </Street>
         <PostalInfo>
            <City>Foster City</City>
            <CountrySubEntity>CA</CountrySubEntity>
            <PostalCode>95118</PostalCode>
            <Country>USA</Country>
         </PostalInfo>
         <Contact>
            <Name1>Tom Proctor</Name1>
         </Contact>
         <Ref stdValue="VA" stdName="UNTDID:1153">CA12345678901234</Ref>
      </Party>
      <Party stdValue="PI" stdName=" UNTDID:3035">
         <PartyID>CROWNE002</PartyID>
         <Name>
            <Name1>CROWNE HOTEL GROUP</Name1>
         </Name>
         <PostalInfo>
            <City>HAMBURG</City>
            <PostalCode>00000</PostalCode>
            <Country>DE</Country>
         </PostalInfo>
         <Ref stdValue="ADQ" stdName="UNTDID:1153">200000</Ref>
      </Party>
      <PONum>PO00001</PONum>
      <Ref stdValue="ADQ" stdName="UNTDID:1153">LG</Ref>
      <Ref stdValue="AWE" stdName="UNTDID:1153">98345</Ref>
      <Ref stdValue="RMNO" stdName="VISA:REF">908</Ref>
      <Ref stdValue="RSNO" stdName="VISA:REF">212</Ref>
      <Ref stdValue="RMRT" stdName="VISA:REF">100.00</Ref>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-20T14:11:54</Date>
      <Date stdValue="END" stdName="VISA:DATE">1999-02-11T08:30:12</Date>
   </InvoiceHeader>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>1</LineItemNum>
         <PartNumDetail stdValue="VP" stdName="UNTDID:7143">
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
         </PartNumDetail>
         <PartNumDetail stdValue="CC" stdName="UNTDID:7143">
            <PartNum>H100</PartNum>
         </PartNumDetail>
         <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure stdValue="6411" stdName="UNECE:20UNECE:20"/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>100.00</UnitPrice>
      <LineItemSubtotal>100.00</LineItemSubtotal>
      <Tax>
          <TaxFunction stdValue="7" stdName="UNTDID:5283"/>
         <TaxType stdValue="VAT" stdName="UNTDID:5153"/>
         <TaxCategory stdValue="S" stdName="5305"/>
         <TaxPercent>15.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
   </InvoiceDetails>
   <InvoiceSummary>
      <TaxSummary>
         <Tax>
            <TaxFunction stdValue="7" stdName="UNTDID:5283"/>
            <TaxType stdValue="VAT" stdName="UNTDID:5153"/>
            <TaxCategory stdValue="S" stdName="5305"/>
            <TaxPercent>15.00</TaxPercent>
            <TaxableAmount>100</TaxableAmount>
            <TaxAmount>15.00</TaxAmount>
         </Tax>
      </TaxSummary>
      <InvoiceTotals>
         <NetValue>100.00</NetValue>
         <TaxValue>15.00</TaxValue>
         <GrossValue>115.00</GrossValue>
      </InvoiceTotals>
      <ActualPayment>
         <PaymentAmount>
            <LocalCurrencyAmt>115.00</LocalCurrencyAmt>
         </PaymentAmount>
         <PaymentMean stdValue="ZZZ" stdName="UNTDID:4461"/>
         <PaymentDate>1999-02-11</PaymentDate>
         <CardInfo>
            <CardNum>4917876543212345</CardNum>
            <CardExpirationDate>1199</CardExpirationDate>
            <CardType stdValue="VS" stdName="VISA:VISA:1.CARD"/>
         </CardInfo>
      </ActualPayment>
   </InvoiceSummary>
</Invoice>
```

Invoice Document as a Credit Note

The invoice document may be used as a credit note simply by setting the element InvoiceType’s stdValue attribute to the value “381”, which denotes a credit note.

When the document is a credit note the InvoiceNumber element contains the credit note number. There should be a Ref element within the InvoiceHeader element, with the stdValue attribute set to “IV”, containing the original invoice number that this document is crediting.

All invoice and payment amounts are then implicit credit amounts – therefore amounts should not be shown as negative values on a credit note unless they are charges.

EXAMPLE: INVOICE DOCUMENT AS A CREDIT NOTE

The example below shows credit note B003989, which credits original invoice B003983.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml:stylesheet type="text/xsl" href="invoice.1.0.xsl"?>
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<Invoice sectorUsageVersion="1">
    <InvoiceHeader>
        <InvoiceType stdValue="381"/>
        <InvoiceStatus/>
        <TaxTreatment/>
        <DiscountTreatment/>
        <InvoiceTreatment/>
        <InvoiceNumber>B003989</InvoiceNumber>
        <InvoiceDate>1999-02-11</InvoiceDate>
        <Currency stdValue="DEM"/>
        <Party stdValue="SU">
            <PartyID>5011234567890</PartyID>
            <Name>
                <Name1>Crowne International</Name1>
                <Name2>Frankfurt</Name2>
            </Name>
            <Street>
                <Street1>2022 Market Street</Street1>
            </Street>
            <PostalInfo>
                <City>Frankfurt</City>
                <CountrySubEntity/>
                <PostalCode>69500</PostalCode>
                <Country>Germany</Country>
            </PostalInfo>
            <Contact>
                <TelNum>+49 812 1234 222</TelNum>
                <Function>Accounts Dept</Function>
            </Contact>
            <Ref>DE1234567890</Ref>
            <Ref stdValue="XA">98398351</Ref>
        </Party>
        <Party stdValue="BY">
            <PartyID>LC100</PartyID>
            <Name>
                <Name1>Tom Proctor</Name1>
                <Name2>eCommerce Department</Name2>
                <Name3>Large Company Inc.</Name3>
            </Name>
            <Street>
                <Street1>Metro 1</Street1>
                <Street2>Metro Boulevard</Street2>
                <Street3>Ludgate Circus</Street3>
            </Street>
            <PostalInfo>
                <City>Foster City</City>
                <CountrySubEntity>CA</CountrySubEntity>
                <PostalCode>95118</PostalCode>
                <Country>US</Country>
            </PostalInfo>
            <Contact>
                <Name1>Tom Proctor</Name1>
            </Contact>
            <Ref>CA12345678901234</Ref>
        </Party>
        <Party stdValue="PI">
            <PartyID>CROWNE002</PartyID>
            <Name>
                <Name1>Crowne Hotel Group</Name1>
            </Name>
            <PostalInfo>
                <City>Hamburg</City>
                <PostalCode>12345</PostalCode>
                <Country>567</Country>
            </PostalInfo>
            <Ref stdValue="ADQ">1234</Ref>
        </Party>
        <PONum>PO00001</PONum>
        <Ref stdValue="IV">B003983</Ref>
        <Ref stdValue="ADQ">LG</Ref>
        <Ref stdValue="AWE">98345</Ref>
        <Ref stdValue="RMNO" stdName="VISA:REF">908</Ref>
        <Ref stdValue="RSNO" stdName="VISA:REF">212</Ref>
        <Ref stdValue="RMRT" stdName="VISA:REF">100.00</Ref>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-20T14:11:54</Date>
        <Date stdValue="END" stdName="VISA:">1999-02-11T08:30:12</Date>
    </InvoiceHeader>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>1</LineItemNum>
            <PartNumDetail>
                <PartNum>100</PartNum>
                <PartDesc>Room Charge</PartDesc>
            </PartNumDetail>
            <PartNumDetail stdValue="CC">
                <PartNum>H100</PartNum>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>100.00</UnitPrice>
        <LineItemSubtotal>100.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>15.00</TaxPercent>
            <TaxAmount>15.00</TaxAmount>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
```

```
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>15.00</TaxPercent>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>100.00</NetValue>
            <TaxValue>15.00</TaxValue>
            <GrossValue>115.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>115.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>1999-02-11</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
</Invoice>
```

Additional Functionality

The Visa XML invoice implementation is based on a generic XML invoice DTD which has been developed within Visa. That specification is intended for general, non-Visa-specific global implementations, and provides functionality over and above that documented in this publication. This additional functionality includes the use of payment terms and settlement discounts, and additional code lists.

If this additional functionality is required, please refer to the *General XML Invoice Implementation Guide.*

IMPORTANT: PARTIES DEFINING DATA ELEMENTS NOT INCLUDED IN THIS SPECIFICATION MUST FIRST COLLECTIVELY AGREE TO SUPPORT THE NEW DATA ELEMENTS.

XML Invoice Document Structure 2

This chapter describes the logical structure of the XML invoice file and the business usage, and provides explanations of the mapping required to and from application data. It also details the usage for the data that is required in the Visa implementation. Note that the individual elements are documented in full detail in Chapter 3, Element Details. Refer to that chapter to find data formats and recommended code lists.

Table 2–1 shows the hierarchical structure of the invoice document.

The Occurs column indicates how many times an element may occur. For example 0-n means that it can occur zero or more times, with no maximum upper limit (therefore it is optional), and a value of 1 means that it must occur once, and once only and therefore is mandatory.

The Condition column indicates any conditions that apply to the element. See Table 2–2 for condition explanations.

Table 2–1: XML Invoice File Structure (1 of 3)

File Structure	Occurs	Condition
Invoice	1	
Invoice Header	1	
InvoiceType	1	
InvoiceStatus	1	
TaxTreatment	1	
DiscountTreatment	0–1	C1
InvoiceTreatment	1	
InvoiceNumber	1	
InvoiceDate	1	
TaxPointDate	0–1	
Currency	1	
Party	3–n	C2
PartyID	0–1	
Name	0–1	
Name1	1	
Name2	0–1	
Name3	0–1	
Street	0–1	
Street1	1	
Street2	0–1	
Street3	0–1	
Street4	0–1	
PostalInfo	0–1	
City	0–1	
CountrySubEntity	0–1	
PostalCode	0–1	
Country	0–1	
Contact	0–n	
Contact	0–1	
TelNum	0–1	
Email	0–1	
Function	0–1	
Ref	0–n	
PONum	0–1	
DeliveryNoteNum	0–1	
Ref	0–n	
Date	0–n	
GenText	0–n	
InvoiceDetails	1–n	
BaseItemDetail	1	
LineItemNum	1	
SubLineItemNum	0–1	C4
PartNumDetail	1–n	
PartNum	0–1	C5

Table 2–1: XML Invoice File Structure (2 of 3)

File Structure	Occurs	Condition
PartDesc	0–1	C5
Quantity	1	C6
Qty	1	
UnitOfMeasure	1	
UnitPrice	1	C6
POLineNum	0–1	
LineItemSubTotal	1	C6
Tax	0–n	C6, C7
TaxFunction	1	
TaxType	1	
TaxCategory	1	
TaxPercent	1	
TaxableAmount	0–1	
TaxAmount	0–1	
Location	0–1	
LineDiscountInfo	0–1	
DiscountValue	0–1	C9
DiscountPercent	0–1	C9
UnitPricePreDiscount	0–1	
Date	0–n	
SpecialCond	0–1	
Ref	0–n	
GenText	0–n	
InvoiceSummary	1	
TaxSummary	0–n	C8
DiscountSummary	0–1	
LineItemTotals	1	
QtyDiscount	0–1	
ValueDiscount	0–1	
SubTotalAfterQtyValueDiscount	1	
Tax	1	
TaxFunction	1	
TaxType	1	
TaxCategory	1	
TaxPercent	1	
TaxableAmount	1	
TaxAmount	1	
Location	0–1	
InvoiceTotals	1	
NetValue	1	
TaxValue	0–1	
GrossValue	1	
DiscountSummary	0–1	
LineItemTotals	1	
QtyDiscount	0–1	

Table 2–1: XML Invoice File Structure (3 of 3)

File Structure	Occurs	Condition
ValueDiscount	0–1	
SubTotalAfterQtyValueDiscount	1	
ActualPayment	0–n	
PaymentAmount	1	
LocalCurrencyAmt	1	
ForeignCurrencyPayment	0–1	
ForeignCurrencyAmount	1	
Currency	1	
PaymentMean	1	
PaymentDate	1	
CardInfo	0–1	
CardNum	1	
CardAuthCode	0–1	
CardRefNum	0–1	
CardExpirationDate	1	
CardType	0–1	
CardholderName	0–1	
Ref	0–n	
Ref	0–n	

Table 2–2: Conditions Applicable to Elements

C1	DiscountTreatment must be present if line-level discounts are used. If line-level discounts are not used, then this element need not be present.
C2	There must be at least three instances of Party in an invoice – one each for the supplier (Merchant) and buyer (Corporate) and one for additional Merchant details.
C4	SubLineItemNum must only have a value if it is used as a subline within the current line. The presence of a value in this element indicates that the current instance of InvoiceDetails is a subline.
C5	In PartNumDetail either PartNum or PartDesc, or both PartNum and PartDesc may be present (that is, either one, or both).
C6	If the InvoiceDetail instance is a subline, for example, if SubLineItemNum has a value, then —with the exception of PartNumDetail—the remaining elements are optional. The InvoiceDetail elements are documented assuming a normal line, i.e. SubLineItemNum is absent.
C7	There must be one Tax element for each tax category associated with the line. If tax is not applied to this invoice, then this element should not occur.
C8	There must be one TaxSummary element for each tax category within the invoice. If tax is not applied to this invoice then this element should not occur.
C9	If LineDiscountInfo is present, then either—but not both—DiscountValue or DiscountPercent must be present.

Logical Structure Examples

The following discussion includes extracts from XML files to demonstrate the structure, and the usage of the elements. Note that some of these extracts have their details “hidden” so that the main point being explained is clear, and not lost in the detail. In the code samples, hidden detail is indicated by the use of colons. For example,

```
<InvoiceHeader>

:     :

</InvoiceHeader>
```

indicates that the detail between the InvoiceHeader opening and closing tags is hidden.

The Invoice Element

As depicted in Table 2–1, the root element of the XML invoice document is the Invoice element. It has a single attribute associated with it to hold the sector usage version number of the message.

Below the root element, there are three top-level sections: InvoiceHeader, InvoiceDetails, and InvoiceSummary as shown in Table 2–3.

Table 2–3: Top-Level Subelements Below the Invoice Element

Subelement	Description
Invoice Header	This element holds all invoice-level information, for example invoice type, invoice number, buyer and supplier name and address details. This element repeats once within the document.
Invoice Details	This element holds all invoice line details. InvoiceDetails repeats within the invoice document, and there is one instance for each invoice line item.
Invoice Summary	This element holds all invoice summary information, for example invoice and tax summary totals, and details of the actual payments that have been made against the invoice. This element repeats once within the document.

EXAMPLE

The following example shows the main structure of the XML invoice document. Note the reference to the DTD in the DOCTYPE declaration.

```
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<Invoice sectorUsageVersion=”1”>
   <InvoiceHeader>
     :    :
   </InvoiceHeader>
   <InvoiceDetails>
     :    :
   </InvoiceDetails>
   <InvoiceSummary>
     :    :
   </InvoiceSummary>
</Invoice>
```

InvoiceHeader

This container element contains subelements that hold all the data associated with the invoice as a whole that is not classed as summary information.

Table 2–4: InvoiceHeader Subelements (1 of 3)

Subelement	Description
InvoiceType	The stdValue attribute indicates the type of invoice, such as whether the invoice document is an invoice or a credit note. Note that the only difference between an invoice and a credit note is the stdValue attribute value. The rest of the document content is the same as documented. This element is mandatory. The default attribute indicates invoice.
InvoiceStatus	The stdValue attribute indicates the status of the invoice, such as whether it is an original invoice, a copy invoice, or a test invoice. This element is mandatory. The default attribute indicates original invoice.
TaxTreatment	The stdValue attribute indicates the tax treatment, such as whether gross or net pricing is used, and whether tax is calculated at line or invoice level, or no tax is applied. This element is mandatory. The default attribute indicates net pricing, line level tax.

Table 2–4: InvoiceHeader Subelements (2 of 3)

Subelement	Description
DiscountTreatment	The stdValue attribute indicates whether line-level discount amounts are based on the UnitPrice or LineItemSubtotal, and whether the amounts are net or gross of discount. If no line-level discounts are present then this element need not be present; otherwise it is mandatory. The default attribute indicates that the discount value applies to the unit price, and that the unit price is gross (that is, no discount has been applied).
InvoiceTreatment	The stdValue attribute indicates whether the invoice is also printed, and if so which version (paper or electronic) is used for tax reclaim. This element is mandatory. The default attribute indicates that the invoice is printed and given to purchaser for tax reclaim purposes.
InvoiceNumber	The element holds the invoice number. This element is mandatory.
InvoiceDate	The element holds the invoice date. This element is mandatory.
TaxPointDate	The element holds the tax point date. This is an optional element—if the tax point date is the same as the invoice date, then this element need not be present.
Currency	The stdValue attribute holds a coded currency identifier. The currency value applies to the whole invoice. This element is mandatory. The default value indicates U.S. Dollars.

Table 2–4: InvoiceHeader Subelements (3 of 3)

Subelement	Description
Party	This container element holds name and address details of parties associated with the invoice. It also allows contact details for a party to be stored, and references associated with the party, for example tax registration numbers. It is mandatory that there are at least three instances of Party within the document. Party is documented separately in the “Other Elements” section that follows.
PONum	This element holds the original purchase order number that the invoice refers to. An original purchase order may only be referenced at invoice level. This element is optional.
DeliveryNoteNum	This element holds the delivery note number that the invoice refers to. A delivery note may only be referenced at invoice level. This element is optional.
Ref	This element can hold any other invoice-level references, for example the buyer’s cost centre or a price list number that do not have specific elements. The stdValue attribute indicates the type of reference, e.g. cost centre, and the element holds the actual reference number. This element is optional and can repeat as often as required.
Date	This optional element can hold any other invoice-level dates that do not have specific elements. This element can repeat as required.
GenText	This element can hold any other general text, for example delivery instructions, company share capital value (which is mandatory in France). The stdValue attribute indicates the type of text, and the element holds the actual text. This element is optional and can repeat as often as required.

Additional InvoiceHeader Mapping Requirements

The Ref elements shown in Table 2–5 may also be required in an InvoiceHeader.

Table 2–5: InvoiceHeader Mapping

Ref Element	Description
Sector type **Mandatory**	There must be a Ref element where stdValue has the value “ADQ” – the element data must then be the Sector Type. The Sector Type value is itself a coded representation, and these codes are documented in Appendix C – Code Lists.
Customer cost code **Optional**	If required there may be a Ref element where stdValue has the value “AWE” – the element data must then be the customer’s cost code.
Original document number **Conditional**	If the invoice document is a credit note, then there must be a Ref element where stdValue has the value “IV” – the element data must then be the original invoice number.
Market share capital value **Conditional**	Invoices originating from companies located in France must indicate their market share capital value. This should be sent in the GenText element, where stdValue has the value “AHR”.

EXAMPLE

The following InvoiceHeader example shows the header section of an invoice, without the Party and Payment detail.

This code sample demonstrates an original invoice, with a currency of Deutsche Mark. There are three Party elements—one each for Buyer, Supplier and Merchant (used for matching the invoice with the financial transaction).

```
<Invoice sectorUsageVersion="1">
    <InvoiceHeader>
        <InvoiceType/>
        <InvoiceStatus/>
        <TaxTreatment/>
        <DiscountTreatment/>
        <InvoiceTreatment/>
        <InvoiceNumber>B003983</InvoiceNumber>
        <InvoiceDate>1999-02-11</InvoiceDate>
        <Currency stdValue="DEM"/>
        <Party stdValue="SU">
            :   :   :
        </Party>
        <Party stdValue="BY">
            :   :   :
        </Party>
        <Party stdValue="PI">
            :   :   :
        </Party>
        <PONum>PO00001</PONum>
        <Ref stdValue="ADQ">LG</Ref>
        <!--Sector type-->
        <Ref stdValue="AWE">98345</Ref>
        <!--Cost code-->
        <Ref stdValue="RMNO" stdName="VISA:REF">908</Ref>
        <!--Room Number-->
        <Ref stdValue="RSNO" stdName="VISA:REF">212</Ref>
        <!--Reservation Number-->
        <Ref stdValue="RMRT" stdName="VISA:REF">100.00</Ref>
        <!--Room Rate-->
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-20T14:11:54</Date>
        <!--Arrival/CheckIn Date/Time-->
        <Date stdValue="END" stdName="VISA:DATE">1999-02-11T08:30:12</Date>
        <!--Departure/CheckOut Date/Time-->
    </InvoiceHeader>
        :   :   :
</Invoice>
```

InvoiceDetails

This container element contains subelements that hold all the invoice-line-level data.

There is one instance of InvoiceDetails for each invoice line.

If BaseItemDetail/SubLineItemNum has a value then this indicates that the current InvoiceDetails instance refers to a subline item. The SubLineItemNum element is not normally used, but where it is, it is documented in Chapter 4, Sector-Specific Mapping. Note that several elements are mandatory except when the current line is a subline. This is documented explicitly below.

Please refer to Line Level Discount Treatments later in this chapter for further information about the affect that line level discounts have on the InvoiceDetails elements' values.

Table 2–6: InvoiceDetails Subelements (1 of 3)

Subelement	Description
BaseItemDetail	This container element holds basic line item information—the line number, the sub-line number, the product identification, and quantity. This element is mandatory.
BaseItemDetail/ LineItemNum	This element contains the current line number. It should normally start at 1 and increment sequentially. If the current instance of InvoiceDetails is a subline (that is, SubLineItemNum has a value) then the value of LineItemNum will remain static for its child sublines. This element is mandatory.
BaseItemDetail/ SubLineItemNum	This element contains the subline number. It should normally start at 1 and increment sequentially. This element is conditional—it should only be present if the current line is a sub-line.
BaseItemDetail/ PartNumDetail	This container element holds details—part number and description—of the line item. The stdValue attribute indicates whether the details are the Vendor's details, or the Buyer's details, or a commodity code. The default value indicates Vendor's details. It is a conditional element - it must occur at least once, maximum three times.
BaseItemDetail/ PartNumDetail PartNum	This element holds the part number for the current line item. This number could be a commodity code, the Vendor's product ID or the Buyer's product ID. This element is conditional—either one or both of PartNum and PartDesc must be present.
BaseItemDetail/ PartNumDetail/ PartDesc	This element holds the description for the current line item. This description could be a commodity code description, the Vendor's description or the Buyer's description. This element is conditional—either one or both of PartNum and PartDesc must be present.

Table 2–6: InvoiceDetails Subelements (2 of 3)

Subelement	Description
BaseItemDetail/ Quantity	This container element has subelements to hold the quantity and associated unit of measure This element is mandatory.
BaseItemDetail/ Quantity/ Qty	This element holds the actual invoiced quantity. This element is mandatory.
BaseItemDetail/ Quantity/ UnitOfMeasure	The stdAttribute element holds the unit of measure. The default value indicates each. This element is mandatory.
UnitPrice	This element holds the line item unit price. This element is mandatory at line level, optional at subline level.
POLineNum	This element holds the line number on the original purchase order that this invoiced line item refers to. This element is optional.
LineItemSubtotal	This element holds the sub total value for this line item. This element is mandatory at line level, optional at subline level.
Tax	This container element holds tax information relating to the current line. This element is conditional—tax information must be provided if required by the tax authority. Refer to the Invoice Tax Treatments section for further information.
LineDiscountInfo	This container element holds line-level discount information.
LineDiscountInfo/ DiscountValue	This element holds the value of the line discount. This element is conditional—if line discounts apply either this or LineDiscountInfo/DiscountPercent must be present.

Table 2–6: InvoiceDetails Subelements (3 of 3)

Subelement	Description
LineDiscountInfo/ DiscountPercent	This element holds the percentage rate of the line discount (the rate at which discount is applied). This element is conditional—if line discounts apply either this or LineDiscountInfo/DiscountValue must be present.
LineDiscountInfo/ UnitPricePreDiscount	This element holds the UnitPrice before discount was applied. It is only relevant if unit-price line-level discount has been applied. This element is optional.
Date	This element can hold any other invoice-level dates that do not have specific elements. This element can repeat as required. This element is optional.
SpecialCond	This element can be used to hold any special conditions to which the line item is subject. The stdValue attribute indicates the special conditions which apply. This element is optional.
Ref	This element can hold any other line-level references. The stdValue attribute indicates the type of reference and the element holds the actual reference number. This element is optional, and can repeat as often as required.
GenText	This element can hold any other general text, which does not have an explicit element. The stdValue attribute indicates the type of text, and the element holds the actual text. This element is optional, and can repeat as often as required.

InvoiceSummary

This container element contains subelements that hold all the invoice summary details.

There is one instance of InvoiceSummary in the invoice.

Table 2–7: InvoiceSummary Subelements (1 of 3)

Subelement	Description
TaxSummary	This container element holds Tax and Discount summary information for the invoice, for the tax category as specified in the Tax element. If tax does not apply to this invoice then there should not be an occurrence of TaxSummary. If invoice discounts have been applied on a non-tax invoice then these can be detailed in the InvoiceTotals/DiscountSummary section. If tax does apply to the invoice, there must be one occurrence of TaxSummary for each tax category for which there are line level tax categories.
TaxSummary/ DiscountSummary	This container element holds summary amounts, including discount summary information, for the current tax category, such as the category denoted in the corresponding Tax element. Refer to the Invoice Tax Treatments section for further information on the values in the DiscountSummary elements. This element is optional.
TaxSummary/ DiscountSummary/ LineItemTotals	This element holds the sum of the invoice lines sub-total amounts. This element is mandatory.
TaxSummary/ DiscountSummary/ QtyDiscount	This element holds the proportion of invoice quantity discount that is to be applied. This element is optional.
TaxSummary/ DiscountSummary/ ValueDiscount	This element holds the proportion of invoice value discount that is to be applied. This element is optional.

Table 2–7: InvoiceSummary Subelements (2 of 3)

Subelement	Description
TaxSummary/ DiscountSummary/ SubTotalAfterQty/ ValueDiscount	This element holds the invoice sub-total after quantity and value discounts. This is calculated as LineItemTotals - QtyDiscount - ValueDiscount This element is mandatory.
TaxSummary/ Tax	This container element holds tax information. Please refer to the Invoice Tax Treatments section for further information with regard to the values in the Tax elements. This element is mandatory.
TaxSummary/ Tax/ TaxFunction	The stdValue attribute of this element indicates the function of the Tax element, for example, Tax, Customs duty. The default value indicates Tax. This element is mandatory.
TaxSummary/ Tax/ TaxType	The stdValue attribute of this element indicates the type of tax, for example, VAT, GST. The default value indicates VAT. This element is mandatory.
TaxSummary/ Tax/ TaxCategory	The stdValue attribute of this element indicates the tax category. This element is mandatory.
TaxSummary/ Tax/ TaxPercent	This element holds the tax rate as a percentage. This element is mandatory.
TaxSummary/ Tax/ TaxableAmount	This element holds the taxable amount. This element is mandatory.
TaxSummary/ Tax/ TaxAmount	This element holds the tax amount. This element is mandatory.

Table 2–7: InvoiceSummary Subelements (3 of 3)

Subelement	Description
TaxSummary/ Tax/ Location	This element holds the tax location. This element is conditional—it need only be present if the local tax authority requires it to be.
InvoiceTotals	This container element holds the invoice totals. This element is mandatory.
InvoiceTotals/ DiscountSummary	This container element holds invoice-summary-level discount information. The values are the discount summary information for all tax codes, or discount information if the invoice does not have a tax element. Please refer to the TaxSummary/DiscountSummary descriptions above for descriptions of the elements, and to the Invoice Tax Treatments section for details with regard to the values in the DiscountSummary elements.
InvoiceTotals/ NetValue	This element holds the net value of the invoice. Please refer to the Invoice Tax Treatments section for details with regard to the value in this element. This element is mandatory.
InvoiceTotals/ TaxValue	This element holds the tax value for the invoice. Please refer to the Invoice Tax Treatments section for details with regard to the value in this element. This is a conditional element – it must be present if the invoice has tax amounts associated with it, otherwise it can be absent.
InvoiceTotals/ GrossValue	This element is the gross value of the invoice. Please refer to the Invoice Tax Treatments section for details with regard to the value in this element. This element is mandatory.
ActualPayment	This container element holds details of actual payments that have been made on the invoice. Please refer to the Actual Payment Element section for further details. This element is mandatory.

Other Elements

There are several other elements that are most commonly used in invoice documents. These are associated with the Party and ActualPayment elements and are introduced below.

Party

The Party element allows Name and Address details to be stored, an ID for the party (such as Account number), and contact details and references associated with the party (for example, tax registration number).

The Party element's stdValue attribute qualifies the Party element as to its purpose. For example stdValue "BY" indicates that the Party details relate to the Buyer of the goods being invoiced.

Normally within an invoice there will be at least two instances of the Party element, one for the buyer – or corporate client- qualified with "BY", and one for the supplier – or merchant - qualified with "SU".

Normally the invoicer is the same as the supplier, and the invoicee is the same as the buyer. However, if either is different then there must be further instances of Party qualified with PE (payee) for the invoicer, and IV for the invoicee.

The invoicer and invoicee Party instances must then contain the full name and address of the party concerned. If so required by the tax authority, the invoicer's tax registration number must also be present in the Party's Ref element.

In the Visa implementation there is also a Party instance that is qualified with "PI". This holds the details of the Merchant as the Merchant is known to Visa, including the Merchant Site Identity, and the doing-business-as name. This information is used to help match the invoice details with the financial transaction provided via the Acquirer-VisaNet-Issuer route.

Table 2–8: Party Subelements (1 of 3)

Subelement	Description
PartyID	This element holds a value that identifies the Party. For example, it may hold the party's EAN number. This element is optional.

Table 2–8: Party Subelements (2 of 3)

Subelement	Description
Name	This container element contains subelements Name1, Name2 and Name3 to hold the party's name details. This element is optional, but should be present if the Party instance refers to the Buyer or Seller.
Name/ Name1 – Name3	These elements hold the Name details of the party, as would appear on a posted envelope, such as person name, department and company name. The first element (Name1) is mandatory, and can be followed by the optional Name2 and Name3 elements.
Street	This container element contains subelements Street1, Street2, Street3 and Street4 to hold the party's name details. This element is optional, but should be present if the Party instance refers to the Buyer or Seller.
Street/ Street 1 – Street4	These elements hold the Street details of the party—such as any address details that do not fall into the Name or PostalInfo elements. For example, office number, floor number, building name, street address. The first element (Street1) is mandatory, and can be followed by the optional Street2, Street3 and Street4 elements.
PostalInfo	This container element contains sub-elements City, CountrySubEntity, PostalCode and Country to hold the party's Postal address details. This element is optional, but should be present if the Party instance refers to the Buyer or Seller.
PostalInfo/ City	This element contains the city. This element is optional.
PostalInfo/ CountrySubEntity	This element contains the CountrySubEntity value—such as State in the U.S., County in the U.K., province in Canada or France. This element is optional.

Table 2–8: Party Subelements (3 of 3)

Subelement	Description
PostalInfo/ PostalCode	This element contains the PostalCode value—for example, zip code in the U.S., postcode in the U.K. This element is optional.
PostalInfo/ Country	This element contains the country name. This element is optional.
Contact	This container element contains subelements ContactName, TelNum, Email and Function to hold any contact information about the party. This element may repeat as required. This element is optional.
Contact/ Name1	This element contains the contact's name. This element is optional.
Contact/ TelNum	This element contains the contact's telephone number This element is optional.
Contact/ Email	This element contains the contact's email address This element is optional.
Contact/ Function	This element contains the contact's function. For example, this could be a job title, or a department name. This element is optional.
Contact/ Ref	This element can hold any other references that apply to the party, for example the party's TAX registration number, company registration number. The stdValue attribute indicates the type of reference, such as vat registration number, and the element holds the actual reference number. This element can repeat as required. This element is optional.

Additional Party Mapping Requirements

The Ref elements shown in Table 2–9 may also be required within the Party element.

Table 2–9: Party Mapping (1 of 2)

Ref Element	Description
Supplier's tax reg no. **Conditional**	If the local tax regulations require this data, then there must be a Ref within the Supplier Part. The stdValue must then have the value "VA", and the element data must then be the tax registration number.
Buyer's tax reg no. **Conditional**	If the local tax regulations require this data, then there must be a Ref within the Buyer Party. The stdValue must then have the value "VA", and the element data must then be the tax registration number.
Buyer's name **Mandatory**	The buyer's name must populate the Buyer's Party Contact/Name1 element. Typically this will be the cardholder name as it appears on the Visa card.
Customer code **Optional**	Customer code, such as the customer account number as known by the Supplier, may populate the Buyer Party's PartyID element.
Supplier's co. registration no. **Optional**	If there is a requirement to include the Supplier's company registration number, then it can be held in a Ref within the Supplier Party. The stdValue must then have the value "XA", and the element data must then be the company registration number.
Merchant doing-business-as name **Mandatory**	This data must be mapped to the Party "PI" Name1 element. The maximum field length is 25 characters. This data must be spelled exactly the same as that provided on the financial TC05/TC06 Merchant Name field, including any spaces.
Merchant city **Mandatory**	This data must be mapped to the Party "PI" PostalInfo/City element. The maximum field length is 13 characters. This data must be spelled exactly the same as that provided on the financial TC 05/TC 06 Merchant City field, including any spaces.

Table 2–9: Party Mapping (2 of 2)

Ref Element	Description
Merchant country code **Mandatory**	This must be the 2 character alpha (plus trailing space) Merchant Country Code, which must be mapped to Party "PI" PostalInfo/Country element. This data must be spelled exactly the same as that provided on the financial TC 05/ TC 06 Merchant Country field.
Merchant category code **Mandatory**	This must be the 4-digit (numeric) Merchant Category Code, which must be mapped to the Ref element within the Party "PI" element. The Ref element's stdValue value must be "ADQ".
Merchant zip code **Mandatory**	This must be the 5-digit (numeric) Merchant Zip Code value, which must be mapped to the Party "PI" PostalInfo/ PostalCode element.
Merchant site ID **Mandatory**	This data must be mapped to the Party "PI" PartyID element. The maximum field length is 15 characters. This identifier is provided by Visa to uniquely identify an XML Invoice Enhanced Data enabled Merchant.
Acquirer reference number **Conditional**	Wherever possible, the unique acquirer reference number should be provided by the acquirer bank, thus providing a link between the invoice and its corresponding financial data. Note that this information is only provided by the acquirer bank, not the merchant. This information is mapped to the Ref element within the Party "PI" element. The Ref element's stdValue value must be "ACD".

The following example shows three Party elements within the InvoiceHeader element.

The first Party element demonstrates a Supplier name and address (stdValue has the value "SU"), with contact details and Ref elements with a VAT registration code (which is the default), and company registration number.

The second Party element demonstrates Buyer name and address (stdValue has the value "BY"). Again, there are details for one Contact, and the Ref element holds the Buyer's tax registration number.

The third Party element demonstrates the Merchant-details instance (stdValue has the value "PI").

EXAMPLE

```
<InvoiceHeader>
    :   :
<Party stdValue="SU">
    <PartyID>5011234567890</PartyID>
        <Name>
            <Name1>Crowne International</Name1>
            <Name2>Frankfurt</Name2>
        </Name>
        <Street>
            <Street1>2022 Market Street</Street1>
        </Street>
        <PostalInfo>
            <City>Frankfurt</City>
            <CountrySubEntity/>
            <PostalCode>69500</PostalCode>
            <Country>Germany</Country>
        </PostalInfo>
        <Contact>
            <TelNum>+49 812 1234 222</TelNum>
            <Function>Accounts Dept</Function>
        </Contact>
        <Ref>DE1234567890</Ref>
        <Ref stdValue="XA">98398351</Ref>
    </Party>
    <Party stdValue="BY">
        <PartyID>LC100</PartyID>
        <Name>
            <Name1>Walter Franklin</Name1>
            <Name2>eCommerce Department</Name2>
            <Name3>Large Company Inc.</Name3>
        </Name>
        <Street>
            <Street1>Metro 1</Street1>
            <Street2>Metro Boulevard</Street2>
            <Street3>Ludgate Circus</Street3>
        </Street>
        <PostalInfo>
            <City>Foster City</City>
            <CountrySubEntity>CA</CountrySubEntity>
            <PostalCode>95118</PostalCode>
            <Country>USA</Country>
        </PostalInfo>
        <Contact>
            <Name1>Walter Franklin</Name1>
        </Contact>
        <Ref>CA12345678901234</Ref>
    </Party>
    <Party stdValue="PI">
        <PartyID>CROWNE002</PartyID>
        <Name>
            <Name1>CROWNE HOTEL GROUP</Name1>
        </Name>
        <PostalInfo>
            <City>HAMBURG</City>
            <PostalCode>00000</PostalCode>
            <Country>DE</Country>
        </PostalInfo>
        <Ref stdValue="ADQ">200000</Ref>
    </Party>
    :   :
</InvoiceHeader
```

ActualPayment

The ActualPayment container element is a subelement of the InvoiceSummary element, and holds details of actual payments that have been made in settlement of the invoice. The ActualPayment element holds the payment amount, method and date. It also holds card details for any payments that have been made with a Visa card.

There must be at least one occurrence of ActualPayment, as this holds the Visa card details.

Table 2–10: ActualPayment Subelements (1 of 3)

Subelement	Description
PaymentAmount	This container element contains sub-elements PaymentAmount, PaymentMean, PaymentDate, CardInfo, Ref. This element may repeat as required. There must be at least one instance of this element, with the Visa card number in the CardInfo/CardNum element.
PaymentAmount/ LocalCurrencyAmt	This element holds the amount, in the invoice currency, for this payment. This element is mandatory. Even if the payment is in a foreign currency, this element must still hold the corresponding amount in the invoice currency.
PaymentAmount/ ForeignCurrency Payment	This container element holds the currency and amount if the payment was not made in the invoice currency. This element is optional, but must be used if the payment was not made in the invoice currency.
PaymentAmount/ ForeignCurrency Payment/ ForeignCurrencyAmt	This element holds the amount paid in foreign currency. This element is mandatory within the ForeignCurrencyPayment element.

Table 2–10: ActualPayment Subelements (2 of 3)

Subelement	Description
PaymentAmount/ ForeignCurrency Payment/ Currency	This element holds the currency in which the foreign currency payment was made. This element is mandatory within the ForeignCurrencyPayment element.
PaymentMean	The stdValue attribute indicates the payment method for this payment. This is a mandatory element. The default attribute indicates credit/debit card. Note that, in the Visa implementation, there should only be two values in stdValue – the default attribute of “ZZZ” and “OTHER”. If it is “OTHER” then the PaymentMean element should not contain text.
PaymentDate	This element holds the date the payment was made. This element is mandatory.
CardInfo	This container element holds card details if the card was a Visa card. This element is optional, but must be present if the payment was made with a Visa card. There must always be at least one instance of CardInfo in a Visa-implementation invoice, as this is where the card number is held. ***Note:*** *In the Visa implementation, there should only be instances of CardInfo that refer to Visa cards. There should never be a CardInfo element that refers to another type of card.*
Ref	This container element may be used to hold additional information regarding the actual payment made. Please refer to the Other Codes section in Appendix C to see codes that have been defined to be used at this level. This element may repeat as required. This element is optional.
CardInfo/CardNum	This element holds the Visa card number. This element is mandatory within CardInfo.

Table 2–10: ActualPayment Subelements (3 of 3)

Subelement	Description
CardInfo/ CardAuthCode	This element holds the authorization code. This element is optional, but should be supplied whenever possible.
CardInfo/ CardRefNum	This element may hold any additional a customer-specific reference number. This element is optional.
CardInfo/ CardExpirationDate	This element is holds the expiry date of the card in MMYY format. This element is optional within CardInfo.
CardInfo/CardType	The stdValue attribute indicates the card type. This is a mandatory element. Note that the default attribute, Visa, is the only permitted attribute.
CardInfo/ CardholderName	This element holds the cardholder name, as appears on the card itself. This element is optional within CardInfo.
CardInfo/Ref	This container element may be used to hold additional information regarding the card. Please refer to the Other Codes section in Appendix C to see codes that have been defined to be used at this level. This element may repeat as required. This element is optional.

The following example shows a section of an InvoiceSummary that has information regarding four payments.

The first payment has been made for a value of 100 in the local (that is, invoice) currency. The payment method was credit card (the default of PaymentMean) and the card type was Visa (the default of CardType).

The second payment has been made for a value of 40 in the local currency, and the payment method was cash (PaymentMean stdValue = “10”).

The third payment has been made for a value of 20 in the local currency, and the payment method was “Account” (PaymentMean stdValue = “OTHER”, with account in the element’s data).

The fourth payment has been made for a value of 4.25 in the local currency, but the payment was actually made in a foreign currency – the Euro as denoted in the Currency element. The amount in Euros was 2.00. The payment method was by cheque (PaymentMean stdValue = “20”).

EXAMPLE

```
<InvoiceSummary>
    :   :   :
    <ActualPayment>
        <PaymentAmount>
            <LocalCurrencyAmt>100.00</LocalCurrencyAmt>
        </PaymentAmount>
        <PaymentMean/>
        <PaymentDate>1999-02-11</PaymentDate>
        <CardInfo>
            <CardNum>4917876543212345</CardNum>
            <CardExpirationDate>1199</CardExpirationDate>
            <CardType/>
        </CardInfo>
    </ActualPayment>
    <ActualPayment>
        <PaymentAmount>
            <LocalCurrencyAmt>40.00</LocalCurrencyAmt>
        </PaymentAmount>
        <PaymentMean stdValue="10"/>
        <PaymentDate>1999-02-11</PaymentDate>
    </ActualPayment>
    <ActualPayment>
        <PaymentAmount>
            <LocalCurrencyAmt>20.00</LocalCurrencyAmt>
        </PaymentAmount>
        <PaymentMean stdValue="OTHER">Account</PaymentMean>
        <PaymentDate>1999-02-11</PaymentDate>
    </ActualPayment>
    <ActualPayment>
        <PaymentAmount>
            <LocalCurrencyAmt>4.25</LocalCurrencyAmt>
            <ForeignCurrencyPayment>
                <ForeignCurrencyAmt>2.00</ForeignCurrencyAmt>
                <Currency stdValue="EUR"/>
            </ForeignCurrencyPayment>
        </PaymentAmount>
            <PaymentMean stdValue="20"/>
        <PaymentDate>1999-02-11</PaymentDate>
    </ActualPayment>
</InvoiceSummary>
```

Invoice-Level Discounts

Invoices can have discounts at line and invoice level. Line level discounts are documented in the Line-level Discount Treatments section later in this chapter.

Invoice level discounts can be either quantity or value discounts. These discounts can either be sent as line items with a negative value, or they can be deducted from the total invoice value. If they are deducted from the total invoice value, then the required values are documented in the next section, Invoice Tax Treatments. Invoice level discounts that are not sent as negative-amount line items are documented in the Tax Treatment section because they affect the calculation of tax to be paid. If there is no tax associated with the invoice, then refer to the details with regard to TaxTreatment value NON.

The Visa implementation does not provide support for settlement discounts, however please refer to the section on Additional Functionality in Chapter 1.

Invoice Tax Treatments

To support “Open Trading” it is necessary to identify within each invoice message what tax model has been used in calculating the taxes on the invoice. It is not possible to enforce that every supplier will always calculate tax in a particular way and then populate elements with a mandatory calculation method. Their existing business applications and processes will not support such an approach, nor should it.

However, for subsequent automated processing of electronically provided invoices at the buyer’s systems it is essential that certain information is provided about the supplier’s business processes if ambiguity and incorrect handling is to be avoided. The TaxTreatment element is one of these mandatory elements and although it will be the same for every invoice generated by any one supplier, it may change from supplier to supplier depending on their business software, business process, type of goods or services offered, country and tax or fiscal regime.

Basic Tax Treatments

There are four basic ways in which the tax on an invoice is calculated, and there is a fifth tax treatment in that an invoice may not have any tax associated with it. The InvoiceHeader/TaxTreatment element indicates which one of these five possible tax treatments applies to the invoice.

The tax treatment options are:

1. Line item amounts are net amounts, and tax is calculated at invoice level (NIL)

2. Line item amounts are gross amounts, and tax is calculated at invoice level (GIL)
3. Line item amounts are net amounts, and tax is calculated at line level (NLL), which is the default tax treatment.
4. Line item amounts are gross amounts, and tax is calculated at line level (GLL)
5. No tax is associated with the invoice (NON).

Where invoices have tax associated with them, any invoice-level discounts (such as quantity discount and value discount) have to be taken into consideration when calculating tax-related amounts.

Multiple Tax Codes

It is possible to have multiple tax codes associated with an invoice line. There are two ways in which this can occur:

- The whole amount is subject to multiple tax codes. For example a line with a LineItemSubtotal value of 100 and with TaxPercent rates of 10% and 5% will have tax calculated at both these rates on the value 100. This usage will be referred to as multicategory tax codes on total amount tax.
- The amount is split across multiple tax codes. For example a line with a LineItemSubtotal value of 150 may have a rate of 10% associated with the first 100, and a rate of 5% associated with the remaining 50. This usage will be referred to a split-total multicategory tax. In the case of split-total multi-category tax, the TaxableAmount element should be used in the InvoiceDetails/Tax element to indicate how the LineItemSubtotal is split across the tax categories.

Note that if there are multicategory tax codes on total amount tax in an invoice line the sum of the InvoiceSummary/TaxSummary/Tax/TaxableAmount elements will not be the total net amount for the invoice, because the taxable amount appears in the Tax element for each tax code. Therefore, the invoice total net amount is calculated as, or from, the sum of the InvoiceDetails/LineItemSubtotals elements.

A rounding principle has been used in the examples whereby a digit of 5 or more is rounded up, and a digit of 4 or less is rounded down. Again, there are a number of tax and fiscal authority allowed variations in how a Supplier performs rounding. These are not explicitly defined within the message but applications receiving such invoice messages may need to recognize and appropriately handle possible rounding variations.

Tax Treatment Examples and Details

Full explanations, details and examples for each Tax Treatment type is documented in Appendix D. This details the values that need to be in the monetary elements that are affected by the invoice tax treatment and invoice-level discount values, according to various scenarios.

The first example in each Tax Treatment type is the simple implementation, such that there are no multiple tax codes per line, and no invoice discounts. Descriptions and examples are provided for each of the Tax treatment types and discount scenarios that are most likely to be encountered during implementation. For example, invoice-level discounts are only documented where tax is calculated at invoice level (NIL and GIL), or there is no tax (NON), and split-total multi-category tax is only documented where tax is calculated at line level (NLL and GLL).

The scenarios that are documented in Appendix D are listed below, in the order in which they appear in that appendix.

- NON with no invoice-level discounts
- NON with invoice-level discounts
- NIL with no discounts, no multicategory tax codes
- NIL with no discounts, and multicategory tax codes
- NIL with an invoice level discount
- GIL with no discounts, no multicategory tax codes
- GIL with no discounts, but multicategory tax codes on total amount tax
- GIL with invoice-level discounts, and multicategory tax codes on total amount tax
- NLL with no discounts, no multicategory tax codes
- NLL with multicategory tax codes on total amount tax and split-total multi-category tax
- GLL with no multicategory tax codes
- GLL with multicategory tax codes on total amount tax and split-total multi-category tax

Line-level Discount Treatments

The InvoiceHeader/DiscountTreatment element indicates how line-level discounts are treated. If there are no line-level discounts in the invoice, then this element may be omitted from the invoice document.

The discount treatment types are:

- **UN**—Discount value/percent relates to the unit price, and the UnitPrice value has had discount applied.
- **UG**—Discount value/percent relates to the unit price, but the UnitPrice value has not had discount applied.
- **TN**—Discount value/percent relates to the line item sub-total, and the LineItemSubTotal value has had discount applied.

The manner in which line-level discounts are treated affects the values in the InvoiceDetails elements UnitPrice and LineDiscountInfo/DiscountValue elements, and how the LineItemSubTotal value is calculated.

The amount discounted can be represented as either an amount or a percentage, and these values can relate to either the unit price or the line item sub-total, depending on the discount treatment. The unit price can either be net of discount (that is, discount has been applied) or gross (that is, discount has yet to be applied). The line item sub-total value is always net of discount. The discount percent value will be the same irrespective of the DiscountTreatment type, whereas the discount value amount will change according to whether the discount treatment is at unit price or line item sub-total level – but will achieve the same result. Refer to the XML excerpts below for examples of this.

Elements related to line-level discounts are held in the LineDiscountInfo element. These elements are DiscountValue, DiscountPercent and UnitPricePreDiscount. The DiscountValue element holds the discount value, the DiscountPercent holds the discount percentage. One or the other, but not both, of these elements must be present if line-level discounts apply. The UnitPricePreDiscount is optional and is only relevant when the DiscountTreatment is at unit price level (UN), when it differs from the UnitPrice value.

Note that line level discount value amounts are applied to the UnitPrice or LineItemSubTotal (according to DiscountTreatment) regardless of whether the UnitPrice or LineItemSubTotal value is net or gross of tax. Therefore, the TaxTreatment code does not affect the manner in which line level discounts are handled.

The following tables and examples demonstrate the affects of each valid DiscountTreatment type.

DiscountTreatment is UN (UnitPrice, Net)

Table 2–11 shows the usage of subelements when the discount treatment type is UN.

Table 2–11: UN (UnitPrice, Net) DiscountTreatment Subelements

Subelement	Description
UnitPrice	UnitPrice is net of discount, where discount has been applied to the unit price. UnitPrice is therefore: UnitPricePreDiscount – DiscountValue or UnitPricePreDiscount - calculated DiscountPercent amount
DiscountValue or DiscountPercent	The discount value or the percentage that has been applied to UnitPrice
UnitPricePreDiscount	The UnitPrice before discount was applied
LineItemSubTotal	LineItemSubTotal is calculated as UnitPrice x Qty

EXAMPLE

The following example demonstrates a DiscountTreatment of UN with a DiscountValue of 10:

```
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>90.00</UnitPrice>
    <LineItemSubtotal>270.00</LineItemSubtotal>
    <Tax>
    <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>15.00</TaxPercent>
    </Tax>
    <LineDiscountInfo>
        <DiscountValue>10</DiscountValue>
        <UnitPricePreDiscount>100.00</UnitPricePreDiscount>
    </LineDiscountInfo>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
```

EXAMPLE

This example demonstrates a DiscountTreatment of UN with a DiscountPercent of 10:

```
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>90.00</UnitPrice>
    <LineItemSubtotal>270.00</LineItemSubtotal>
    <Tax>
    <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>15.00</TaxPercent>
    </Tax>
    <LineDiscountInfo>
        <DiscountPercent>10</DiscountPercent>
        <UnitPricePreDiscount>100.00</UnitPricePreDiscount>
    </LineDiscountInfo>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
```

DiscountTreatment is UG (UnitPrice, Gross)

Table 2–12 shows the usage of subelements when the discount treatment type is UG.

Table 2–12: UG (UnitPrice, Gross) DiscountTreatment Subelements

Subelement	Description
UnitPrice	Discount is not applied to the UnitPrice, for example, this value is the unit price before discount
DiscountValue or DiscountPercent	The discount value or discount percentage that is to be applied to UnitPrice
UnitPricePreDiscount	This value is the same value as UnitPrice and is therefore not needed

LineItemSubTotal	LineItemSubTotal = (UnitPrice - DiscountValue) x Qty
	or
	LineItemSubTotal = (UnitPrice – calculated DiscountPercent amount) x Qty

EXAMPLE

This example demonstrates a DiscountTreatment of UG with a DiscountValue of 10. Note the absence of the UnitPricePreDiscount element.

```
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>270.00</LineItemSubtotal>
    <Tax>
    <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>15.00</TaxPercent>
    </Tax>
    <LineDiscountInfo>
        <DiscountValue>10</DiscountValue>
    </LineDiscountInfo>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
```

DiscountTreatment is TN (LineItemSubTotal, Net)

Table 2–13 shows the usage of subelements when the discount treatment type is TN.

Table 2–13: TN (LineItemSubTotal, Net) DiscountTreatment Subelements

Subelement	Description
UnitPrice	Discount does not apply to UnitPrice at all, as it is applied to the LineItemSubTotal value
DiscountValue or DiscountPercent	The discount value or discount percentage that is to be applied to LineItemSubTotal
UnitPricePreDiscount	This is the same value as UnitPrice and is therefore not needed

LineItemSubTotal	LineItemSubTotal = (UnitPrice x Qty) – DiscountValue
	or
	LineItemSubTotal = (UnitPrice x Qty) – calculated DiscountPercent amount

EXAMPLE

This example demonstrates a DiscountTreatment of TN with a DiscountValue of 30:

```
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>270.00</LineItemSubtotal>
    <Tax>
    <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>15.00</TaxPercent>
    </Tax>
    <LineDiscountInfo>
        <DiscountValue>30</DiscountValue>
    </LineDiscountInfo>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails
```

EXAMPLE

The following example demonstrates a DiscountTreatment of TN with a DiscountPercent of 30:

```
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>270.00</LineItemSubtotal>
    <Tax>
    <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>15.00</TaxPercent>
    </Tax>
    <LineDiscountInfo>
        <DiscountPercent>10</DiscountPercent>
    </LineDiscountInfo>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
```

Element Details 3

The Element Dictionary in this chapter details all the elements used within the Visa implementation of the XML invoice document. Elements appear in alphabetical order.

In the Element Dictionary all simple elements have a data format associated with them. The data within the element must take this format. Data formats are defined in the following section of this chapter.

Where an element has an associated code list, this is included in the Element's details. The source of the code list, or the code list administrator, is held in stdName attribute and the default values for these are included in the DTD. Within the Visa implementation these code lists will not change, so the stdName default value and the documented code lists should always be used within the XML document.

Where possible, recommended subsets of internationally-agreed codes have been used, for example codes administered by the International Organisation for Standardisation (ISO) and the United Nations (UN). Where no internationally recognized code currently exists, Visa has developed a code list, and these code lists will be submitted to the appropriate standards bodies for adoption.

The Element Dictionary includes extracts from XML files to demonstrate the structure, and the usage of the elements. Note that some of these extracts have their details "hidden" so that the main point being explained is clear, and not lost in the detail. Hidden detail is indicated by the use of colons. For example:

```
<InvoiceHeader>
    :   :
</InvoiceHeader>
```

indicates that the detail between the InvoiceHeader opening and closing tags is hidden.

Data Formats

There are several types of data that are associated with elements, and the data within the elements must take the specified format. These data types and their associated formats are listed here.

Integer

A whole number.

This is documented in the format Integer (3), where 3 denotes the number of digits. The number of digits allowed will depend on the element’s usage.

Therefore, an element with the format Integer (3) may have a value in the range of 0 to 999.

DateTime

A date and/or time.

This takes the format CCYY-MM-DDTHH:MM:SS and follows ISO 8601.

For a date CCYY-MM-DD must be replaced with a valid date,
for example 1999-09-23, for 23rd September 1999. The date must be in this full format; that is, 99-09-23 would not be valid.

If a Time is to be represented then the character T must be included in the data.

It is not necessary to include the seconds; in other words, the time can be truncated to HH:MM

A time can either be appended to a date, or can be a data value in its own right – if no time is included the T should be omitted.

EXAMPLE: VALID DATETIME USAGE

1999-09-23
1999-09-23T10:13
1999-09-23T10:13:45
T10:13
T10:13:45

Quantity

A decimal number.

This is documented in the format Quantity (Decimal 15.4), where 15 denotes the maximum number of digits before the decimal point, and 4 denotes the number of decimal places allowed. Quantities are always in this format, such as Decimal 15.4

A decimal number may be in integer format – for example, a value of 15 is allowed; it is not necessary to include the decimal point.

When a decimal point is included in the data, at least one digit must be included before and after the decimal point.

EXAMPLE: ALLOWED VALUES

0.5
2.0
25
1973452.345

EXAMPLE: VALUES THAT ARE NOT ALLOWED

.5
2.
2.56985

Monetary Amount

A decimal number.

This is documented in the format MonetaryAmount (Decimal 18.3) where 18 denotes the maximum number of digits before the decimal point, and 3 denotes the number of decimal places allowed. Monetary amounts are always in this format, such as Decimal 18.3

A decimal number may be in integer format – for example, a value of 15 is allowed; it is not necessary to include the decimal point.

When a decimal point is included in the data, at least one digit must be included before and after the decimal point.

EXAMPLE: ALLOWED VALUES

0.5
2.0
25
1973452.345

EXAMPLE: VALUES THAT ARE NOT ALLOWED

.5
2.
2.5698

Percentage

A decimal number.

This is documented in the format Percentage (Decimal 3.4) where 3 denotes the maximum number of digits before the decimal point, and 4 denotes the number of decimal places allowed. Percentages are always in this format, such as Decimal 3.4

A decimal number may appear in integer format – for example, a value of 15 is allowed; it is not necessary to include the decimal point.

When a decimal point is included in the data, at least one digit must be included before and after the decimal point.

EXAMPLE: ALLOWED VALUES

0.5
2.0
25
10.25

EXAMPLE: VALUES THAT ARE NOT ALLOWED

.5
2.
2.56985

String

Alphanumeric data.

The allowed length of the string is documented separately for each individual element.

Signs

Numeric data element values are regarded as positive.

If a value must be indicated as negative, place a minus sign (-) symbol immediately before the value in transmission. Do not count the minus sign as a character of value when computing the maximum length of a data element but allow for the character in transmission and reception.

EXAMPLE: ALLOWED VALUES

-0.5
-25

EXAMPLE: VALUES THAT ARE NOT ALLOWED

-    0.5

Element Dictionary

ActualPayment

This container element holds payments that have been made in settlement of the invoice.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<ActualPayment>
       <PaymentAmount>
          <LocalCurrencyAmt>115</LocalCurrencyAmt>
          <ForeignCurrencyPayment>
             <ForeignCurrencyAmt>52</ForeignCurrencyAmt>
             <Currency stdValue=”GBP”/>
          </ForeignCurrencyPayment>
       </PaymentAmount>
       <PaymentMean stdValue=”ZZZ”/>
       <PaymentDate>1999-10-05</PaymentDate>
       <CardInfo>
          :  :  :
       </CardInfo>
</ActualPayment>
```

BaseItemDetail

Container element that holds elements that are common between business document line items, such as line numbers, product information and quantity.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<BaseItemDetail>
       <LineItemNum>1</LineItemNum>
       <PartNumDetail stdValue="VP">
          <PartNum>198D983HGX</PartNum>
       </PartNumDetail>
       <Quantity>
          <Qty>30</Qty>
          <UnitOfMeasure/>
       </Quantity>
</BaseItemDetail>
```

CardAuthCode

This element holds the 6-digit Visa payment authorization code

Element Type: Simple element

Content Format: String, 1 to 6 characters

Attributes: None

EXAMPLE

`<CardAuthCode>123456</CardAuthCode>`

CardExpirationDate

This element holds the card expiry date.

Element Type: Simple element

Content Format: String, 4 characters, in the format MMYY.

Attributes: None

EXAMPLE

`<CardExpirationDate>1299</CardExpirationDate>`

CardholderName

This element holds the cardholder name (the name that appears on the card).

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

`<CardholderName>Mr John Jones</CardholderName >`

CardInfo

This container element holds credit/charge card details.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<CardInfo>
       <CardNum>4402882365913000</CardNum>
       <CardExpirationDate>0100</CardExpirationDate>
       <CardType stdValue="VS"/>
</CardInfo>
```

CardNum

This element holds a credit / charge card number

Element Type: Simple element

Content Format: String, 1 to 35 characters

Attributes: None

EXAMPLE

```
<CardNum>4402882365913000</CardNum>
```

CardRefNum

This element can hold a customer-specific reference number.

Element Type: Simple element

Content Format: String, 1 to 35 characters

Attributes: None

EXAMPLE

```
<CardRefNum>143</CardRefNum>
```

CardType

This element is used to hold the type of credit or charge card.

Note that this element's stdValue attribute has a coded value to represent the card type. In the Visa implementation the only permissable card type is Visa, which is the default.

Element Type: Simple element

Content Format: String, 1 to 70 characters.

Table 3–1: CardType Attributes

stdValue	Denotes the card type. Default value is VS
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is VISA:CARD

EXAMPLE

```
<CardType stdValue="VS"/>
```

City

This element holds a city.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<PostalInfo>
       <City>San Francisco</City>
       <CountrySubEntity>CA</CountrySubEntity>
       <PostalCode>00000</PostalCode>
       <Country>USA</Country>
</PostalInfo>
```

Contact

This container element holds the elements that contain Contact details. This includes the name, telephone number, email address and the function of the contact.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<Contact>
       <Name1>Mike Jones</Name1>
       <TelNum>01272 345987</TelNum>
       <Function>General account enquiries</Function>
</Contact>
```

Country

This element holds the Country element of an address.

Note that the value here should be the full country as would appear on an envelope, not a coded value.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<PostalInfo>
       <City>San Francisco</City>
       <CountrySubEntity>CA</CountrySubEntity>
       <PostalCode>00000</PostalCode>
       <Country>USA</Country>
</PostalInfo>
```

CountrySubEntity

This element holds the CountrySubEntity element of an address – in the US this would be the state, in the UK the county.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<PostalInfo>
       <City>San Francisco</City>
       <CountrySubEntity>CA</CountrySubEntity>
       <PostalCode>00000</PostalCode>
       <Country>USA</Country>
```

Currency

Defines a currency; for example, at Invoice level defines the invoice currency.

Only the ISO Alpha Currency Code (3-character abbreviation) must be used. This code is shown in the fourth column of Table 3–3.

Although the numerical code equivalent is provided in the table it must not be used and is given for information only.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–2: Currency Attributes

stdValue	Indicates the currency Default value is USD
stdName	Indicates the code list from which the stdValue element has been obtained. Fixed value is ISO:4217

EXAMPLE

```
<Currency stdValue="GBP"/>
```

Table 3–3: stdValue Code List—Country and Currency Codes (1 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Afghanistan	AF	Afghani	AFA	004
Albania	AL	Lek	ALL	008
Algeria	DZ	Algerian Dinar	DZD	012
American Samoa	AS	U.S. Dollar	USD	840
Andorra	AD	Spanish Peseta	ESP	724
Angola	AO	New Kwanza	AOK	024
Anguilla	AI	E. Caribbean Dollar	XCD	951
Antarctica	AQ	Norwegian Krone	NOK	578
Antigua and Barbuda	AG	E. Caribbean Dollar	XCD	951
Argentina	AR	Argentine Peso	ARS	032
Armenia	AM	Armenian Dram	AMD	051
Aruba	AW	Aruban Guilder	AWG	533
Australia	AU	Australian Dollar	AUD	036
Austria	AT	Austrian Schilling	ATS	040

Table 3–3: stdValue Code List—Country and Currency Codes (2 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Azerbaijan	AZ	Azerbaijan Manat	AZM	031
Bahamas	BS	Bahamian Dollar	BSD	044
Bahrain	BH	Bahraini Dinar	BHD	048
Bangladesh	BD	Taka	BDT	050
Barbados	BB	Barbados Dollar	BBD	052
Belarus	BY	Belarussian Ruble	BYB	112
Belgium	BE	Belgian Franc	BEF	056
Belize	BZ	Belize Dollar	BZD	084
Benin	BJ	CFA Franc BCEAO	XOF	952
Bermuda	BM	Bermudian Dollar	BMD	060
Bhutan	BT	Indian Rupee	INR	356
Bolivia	BO	Boliviano	BOB	068
Bosnia and Herzegovina	BA	Bosnian Convertible Mark	BAM	977
Botswana	BW	Pula	BWP	072
Bouvet Is.	BV	Norwegian Krone	NOK	578
Brazil	BR	Brazilian Real	BRL	986

Table 3–3: stdValue Code List—Country and Currency Codes (3 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
British Indian Ocean Territory	IO	U.S. Dollar	USD	840
British Virgin Is.	VG	U.S. Dollar	USD	840
Brunei Darussalam	BN	Brunei Dollar	BND	096
Bulgaria	BG	Lev	BGL	100
Burkina Faso	BF	CFA Franc BCEAO	XOF	952
Burundi	BI	Burundi Franc	BIF	108
Cambodia	KH	Riel	KHR	116
Cameroon, United Republic of	CM	CFA Franc BEAC	XAF	950
Canada	CA	Canadian Dollar	CAD	124
Cape Verde Is.	CV	Cape Verde Escudo	CVE	132
Cayman Is.	KY	Cayman Is. Dollar	KYD	136
Central African Republic	CF	CFA Franc BEAC	XAF	950
Chad	TD	CFA Franc BEAC	XAF	950
Chile	CL	Chilean Peso	CLP	152
China	CN	Yuan Renminbi	CNY	156

Table 3–3: stdValue Code List—Country and Currency Codes (4 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Christmas Is.	CX	Australian Dollar	AUD	036
Cocos (Keeling) Is.	CC	Australian Dollar	AUD	036
Colombia	CO	Colombian Peso	COP	170
Comoros	KM	Comoro Franc	KMF	174
Congo	CG	CFA Franc BEAC	XAF	950
Cook Is.	CK	New Zealand Dollar	NZD	554
Costa Rica	CR	Costa Rican Colon	CRC	188
Côte D’Ivoire (Ivory Coast)	CI	CFA Franc BCEAO	XOF	952
Croatia	HR	Croatian Kuna	HRK	191
Cuba	CU	Cuban Peso	CUP	192
Cyprus	CY	Cyprus Pound	CYP	196
Czech Republic	CZ	Czech Koruna	CZK	203
Democratic Republic of the Congo (Zaire)	CD	Congolais Franc (New Zaire)	ZRN	180
Denmark	DK	Danish Krone	DKK	208
Djibouti	DJ	Djibouti Franc	DJF	262
Dominica	DM	E. Caribbean Dollar	XCD	951

Table 3–3: stdValue Code List—Country and Currency Codes (5 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Dominican Rep.	DO	Dominican Peso	DOP	214
East Timor	TP	Timor Escudo	TPE	626
Ecuador	EC	Sucre	ECS	218
Egypt	EG	Egyptian Pound	EGP	818
El Salvador	SV	El Salvador Colon	SVC	222
Equatorial Guinea	GQ	CFA Franc BEAC	XAF	950
Eritrea	ER	Eritean Nakfa	ERN	232
Estonia	EE	Kroon	EEK	233
Ethiopia	ET	Ethiopian Birr	ETB	230
European Union	N/A	euro	EUR	978
Faeroe Is.	FO	Danish Krone	DKK	208
Falkland Is. (Malvinas)	FK	Falkland Is. Pound	FKP	238
Fiji	FJ	Fiji Dollar	FJD	242
Finland	FI	Markka	FIM	246
France	FR	French Franc	FRF	250
France, Metropolitan	FX	French Franc	FRF	250
French Guiana	GF	French Franc	FRF	250

Table 3–3: stdValue Code List—Country and Currency Codes (6 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
French Polynesia	PF	CFP Franc	XPF	953
French Southern Territory	TF	French Franc	FRF	250
Gabon	GA	CFA Franc BEAC	XAF	950
Gambia	GM	Dalasi	GMD	270
Georgia	GE	Georgian Lari	GEL	981
Germany	DE	Deutsche Mark	DEM	280
Ghana	GH	Cedi	GHC	288
Gibraltar	GI	Gibraltar Pound	GIP	292
Greece	GR	Drachma	GRD	300
Greenland	GL	Danish Krone	DKK	208
Grenada	GD	E. Caribbean Dollar	XCD	951
Guadeloupe	GP	French Franc	FRF	250
Guam	GU	U.S. Dollar	USD	840
Guatemala	GT	Quetzal	GTQ	320
Guinea	GN	Guinea Franc	GNF	324
Guinea—Bissau	GW	Guinea-Bissau Peso	GWP	624
Guyana	GY	Guyana Dollar	GYD	328

Table 3–3: stdValue Code List—Country and Currency Codes (7 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Haiti	HT	Gourde	HTG	332
Heard and McDonald Is.	HM	Australian Dollar	AUD	036
Honduras	HN	Lempira	HNL	340
Hong Kong, China	HK	Hong Kong Dollar	HKD	344
Hungary	HU	Forint	HUF	348
Iceland	IS	Iceland Krona	ISK	352
India	IN	Indian Rupee	INR	356
Indonesia	ID	Rupiah	IDR	360
Iran, Islamic Republic of	IR	Iranian Rial	IRR	364
Iran Airlines	N/A	Iranian Airline Rate	IRA	365
Iraq	IQ	Iraqi Dinar	IQD	368
Ireland, Republic of	IE	Irish Pound	IEP	372
Israel	IL	New Israeli, Shekel	ILS	376
Italy	IT	Italian Lira	ITL	380
Jamaica	JM	Jamaican Dollar	JMD	388
Japan	JP	Yen	JPY	392
Jordan	JO	Jordanian Dinar	JOD	400

Table 3–3: stdValue Code List—Country and Currency Codes (8 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Kazakhstan	KZ	Tenge	KZT	398
Kenya	KE	Kenyan Shilling	KES	404
Kiribati	KI	Australian Dollar	AUD	036
Korea, Democratic People’s Republic of (North Korea)	KP	North Korean Won	KPW	408
Korea, Republic of	KR	Won	KRW	410
Kuwait	KW	Kuwaiti Dinar	KWD	414
Kyrgyzstan	KG	Som	KGS	417
Lao People’s Democratic Republic	LA	Kip	LAK	418
Latvia	LV	Latvian Lats	LVL	428
Lebanon	LB	Lebanese Pound	LBP	422
Lesotho	LS	Rand	ZAR	710
Liberia	LR	Liberian Dollar	LRD	430
Libyan Arab Jamahiriya	LY	Libyan Dinar	LYD	434
Liechtenstein	LI	Swiss Franc	CHF	756
Lithuania	LT	Lithuanian Litas	LTL	440

Table 3–3: stdValue Code List—Country and Currency Codes (9 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Luxembourg	LU	Luxembourg Franc	LUF	442
Macau, Special Administrative Region of China	MO	Pataca	MOP	446
Macedonia, the Former Yugoslav Republic of	MK	Denar	MKD	807
Madagascar	MG	Malagasy Franc	MGF	450
Malawi	MW	Malawi Kwacha	MWK	454
Malaysia	MY	Malaysian Ringgit	MYR	458
Maldives	MV	Rufiyaa	MVR	462
Mali	ML	CFA Franc BCEAO	XOF	952
Malta	MT	Maltese Lira	MTL	470
Marshall Islands	MH	U.S. Dollar	USD	840
Martinique	MQ	French Franc	FRF	250
Mauritania	MR	Ouguiya	MRO	478
Mauritius	MU	Mauritius Rupee	MUR	480
Mayotte	YT	French Franc	FRF	250
Mexico	MX	Mexican Peso	MXN	484

Table 3–3: stdValue Code List—Country and Currency Codes (10 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Micronesia	FM	U.S. Dollar	USD	840
Moldova, Republic of	MD	Moldovan Leu	MDL	498
Monaco	MC	French Franc	FRF	250
Mongolia	MN	Tugrik	MNT	496
Montenegro	NT	Yugoslavian New Dinar	YUM	891
Montserrat	MS	E. Caribbean Dollar	XCD	951
Morocco	MA	Moroccan Dirham	MAD	504
Mozambique	MZ	Metical	MZM	508
Myanmar	MM	Kyat	MMK	104
Namibia	NA	Namibia Dollar	NAD	516
Nauru	NR	Australian Dollar	AUD	036
Nepal	NP	Nepalese Rupee	NPR	524
Netherlands	NL	Netherlands Guilder	NLG	528
Netherlands Antilles	AN	Nether. Antillian Guilder	ANG	532
New Caledonia	NC	CFP Franc	XPF	953
New Zealand	NZ	New Zealand Dollar	NZD	554

Table 3–3: stdValue Code List—Country and Currency Codes (11 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Nicaragua	NI	Cordoba Oro	NIO	558
Niger	NE	CFA Franc BCEAO	XOF	952
Nigeria	NG	Naira	NGN	566
Niue	NU	New Zealand Dollar	NZD	554
Norfolk Is.	NF	Australian Dollar	AUD	036
Northern Mariana Islands	MP	U.S. Dollar	USD	840
Norway	NO	Norwegian Krone	NOK	578
Oman	OM	Rial Omani	OMR	512
Pakistan	PK	Pakistan Rupee	PKR	586
Palau	PW	U.S. Dollar	USD	840
Panama	PA	Balboa	PAB	590
Papua New Guinea	PG	Kina	PGK	598
Paraguay	PY	Guarani	PYG	600
Peru	PE	Nuevo Sol	PEN	604
Philippines	PH	Philippine Peso	PHP	608
Pitcairn	PN	New Zealand Dollar	NZD	554

Table 3–3: stdValue Code List—Country and Currency Codes (12 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Poland	PL	Polish New Zloty	PLN	985
Portugal	PT	Portuguese Escudo	PTE	620
Puerto Rico	PR	U.S. Dollar	USD	840
Qatar	QA	Qatari Rial	QAR	634
Reunion	RE	French Franc	FRF	250
Romania	RO	Leu	ROL	642
Russian Federation	RU	Russian Ruble (International)	RUB	643
		Russian Ruble (Domestic)	RUR	810
Rwanda	RW	Rwanda Franc	RWF	646
Samoa	WS	Tala	WST	882
San Marino	SM	Italian Lira	ITL	380
Sao Tome and Principe	ST	Dobra	STD	678
Saudi Arabia	SA	Saudi Riyal	SAR	682
Senegal	SN	CFA Franc BCEAO	XOF	952
Seychelles	SC	Seychelles Rupee	SCR	690
Sierra Leone	SL	Leone	SLL	694
Singapore	SG	Singapore Dollar	SGD	702

Table 3–3: stdValue Code List—Country and Currency Codes (13 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Slovakia	SK	Slovak Koruna	SKK	703
Slovenia	SI	Tolar	SIT	705
Solomon Is.	SB	Solomon Is. Dollar	SBD	090
Somalia	SO	Somali Shilling	SOS	706
South Africa	ZA	Rand	ZAR	710
So. Georgia and So. Sandwich Is.	GS	Pound Sterling	GBP	826
Spain	ES	Spanish Peseta	ESP	724
Sri Lanka	LK	Sri Lanka Rupee	LKR	144
St. Helena	SH	St. Helena Pound	SHP	654
St. Kitts-Nevis	KN	E. Caribbean Dollar	XCD	951
St. Lucia	LC	E. Caribbean Dollar	XCD	951
St. Pierre and Miquelon	PM	French Franc	FRF	250
St. Vincent and The Grenadines	VC	E. Caribbean Dollar	XCD	951
Sudan	SD	Sudanese Pound	SDP	736
Sudan Airlines	N/A	Sudan Airline Rate	SDA	737

Table 3–3: stdValue Code List—Country and Currency Codes (14 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Suriname	SR	Surinam Guilder	SRG	740
Svalbard and Jan Mayen Is.	SJ	Norwegian Krone	NOK	578
Swaziland	SZ	Lilangeni	SZL	748
Sweden	SE	Swedish Krona	SEK	752
Switzerland	CH	Swiss Franc	CHF	756
Syrian Arab Rep.	SY	Syrian Pound	SYP	760
Taiwan	TW	New Taiwan Dollar	TWD	901
Tajikistan	TJ	Tajik Ruble	TJR	762
Tanzania, United Republic of	TZ	Tanzanian Shilling	TZS	834
Thailand	TH	Thailand Baht	THB	764
Togo	TG	CFA Franc BCEAO	XOF	952
Tokelau	TK	New Zealand Dollar	NZD	554
Tonga	TO	Pa’anga	TOP	776
Trinidad and Tobago	TT	Trinidad and Tobago Dollar	TTD	780
Tunisia	TN	Tunisian Dinar	TND	788
Turkey	TR	Turkish Lira	TRL	792
Turkmenistan	TM	Manat	TMM	795

Table 3–3: stdValue Code List—Country and Currency Codes (15 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Turks and Caicos Is.	TC	U.S. Dollar	USD	840
Tuvalu	TV	Australian Dollar	AUD	036
Uganda	UG	Uganda Shilling	UGX	800
Ukraine	UA	Ukrainian Hryvnia	UAH	980
United Arab Emirates	AE	U.A.E. Dirham	AED	784
United Kingdom	GB	Pound Sterling	GBP	826
United States	US	U.S. Dollar	USD	840
U.S. Minor Outlying Islands	UM	U.S. Dollar	USD	840
U.S. Virgin Is.	VI	U.S. Dollar	USD	840
Uruguay	UY	Peso Uruguayo	UYU	858
Uzbekistan	UZ	Uzbekistan Sum	UZS	860
Vanuatu	VU	Vatu	VUV	548
Vatican City State	VA	Italian Lira	ITL	380
Venezuela	VE	Bolivar	VEB	862
Vietnam	VN	Dong	VND	704

Table 3–3: stdValue Code List—Country and Currency Codes (16 of 16)

ISO Country Name	ISO Alpha Country (2-char.) Code	ISO Currency Name	ISO Alpha Currency Code	Default ISO Numeric Currency Code
Wallis and Futuna Is.	WF	CFP Franc	XPF	953
Western Sahara	EH	Moroccan Dirham	MAD	504
Yemen	YE	Yemeni Rial	YER	886
Yugoslavia	YU	Yugoslavian New Dinar	YUM	891
Zambia	ZM	Zambian Kwacha	ZMK	894
Zimbabwe	ZW	Zimbabwe Dollar	ZWD	716

Date

This element is used to hold a date for which there is no explicit element available.

Note that this element's stdValue attribute has a coded value to qualify the date as to its purpose.

Element Type: Simple element

Content Format: DateTime (CCYY-MM-DDTHH:MM:SS)

Table 3–4: Date Attributes

stdValue	Denotes the function, or purpose, of the date No default value
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:2005

EXAMPLE

```
<Date stdValue="194">1999-09-27</Date>
```

or

```
<Date stdValue="206">1999-09-25T09:32</Date>
```

stdValue code list:

Date is used in sector specific implementations, please refer to the relevant Sector-Specific Mapping section for code list values used.

Please refer to the full code list UNTDID:2005 if there is an undocumented requirement to use this element.

DeliveryNoteNum

This element holds the DeliveryNoteNum, to which this invoice refers.

Element Type: Simple element

Content Format: String, 1 to 35

Attributes: None

EXAMPLE

```
<DeliveryNoteNum>398230CD</DeliveryNoteNum>
```

DiscountPercent

This element holds the discount, shown as a percentage.

Element Type: Simple element

Content Format: Percentage (Decimal 3.4)

Attributes: None

EXAMPLE

```
<DiscountPercent>7.25</DiscountPercent>
```

DiscountSummary

This container element has subelements that hold summary discount information.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<DiscountSummary>
<LineItemTotals>1575.00<LineItemTotals/>
<QtyDiscount>50.0<QtyDiscount/>
<ValueDiscount>25.0<ValueDiscount/>
<SubTotalAfterQtyValueDiscount>1500.0<SubTotalAfterQtyValueDiscount/>
</DiscountSummary>
```

DiscountTreatment

Defines the discount treatment, that is whether line-level discounts are based on the unit price (UnitPrice) or on the subtotal line amount (LineItemSubtotal).

If line-level discounts are not used in the invoice, then this element need not be present.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–5: DiscountTreatment Attributes

StdValue	Indicates the line-level discount treatment. Default value is TN
StdName	Indicates the code list from which the stdValue element has been obtained. Fixed value is VISA:DSCT

EXAMPLE

```
<DiscountTreatment stdValue="TN"/>
```

Table 3–6: DiscountTreatment stdValue Code List

UN	Line item unit price, net of discount
UG	Line item unit price, gross of discount
TN	Line item sub-total, net of discount

NOTE: *Code list values are VISA:DSCT*

DiscountValue

This element holds the value of the discount, that is the discount amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<DiscountValue>23.45</DiscountValue>
```

Email

This element holds an email address.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<Contact>
      <Email>smithj@visa.com</Email>
</Contact>
```

ForeignCurrencyAmt

This element holds an amount in a foreign currency, and the actual currency is indicated in the corresponding Currency element.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLEEXAMPLES:

```
<ForeignCurrencyAmt>52</ForeignCurrencyAmt>
```

ForeignCurrencyPayment

This container element holds details of a payment made in a foreign currency It has subelements to hold the foreign currency amount and the currency.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<ForeignCurrencyPayment>
      <ForeignCurrencyAmt>52</ForeignCurrencyAmt>
      <Currency stdValue=”GBP”/>
</ForeignCurrencyPayment>
```

Function

This element holds a function description, for example a job title or department name.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<Contact>
      <Function>Accounts Enquiries</Function>
</Contact>
```

GenText

This element holds general text.

Element Type: Simple element

Content Format: String, 1 to 80 characters.

Table 3–7: GenText Attributes

stdValue	Indicates the purpose of the data in the element. Default value is AHR
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:4451

EXAMPLE

```
<GenText stdValue="AHR">S.A. AU CAPITAL DE 99.000.000 F</GenText>
```

Table 3–8: GenText stdValue Code List

AAI	General information
ACB	Additional information
AHR	Shareholding information

NOTE: *Code list values are a subset of UNTDID:4451*

GrossValue

This element holds the gross value of the invoice, i.e. the total net amount + the total tax amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

`<GrossValue>705.29</GrossValue>`

Invoice

The document root element that contains the InvoiceHeader, InvoiceDetails and InvoiceSummary elements.

Element Type: Container element

Content Format: n/a

Table 3–9: Invoice Attributes

sectorUsageVersion	Indicates the merchant sector usage version number. This enables merchant sectors to change the usage of Ref elements within the DTD and note (by changing this version number in the XML document) that the usage has been changed.

EXAMPLE

`<Invoice sectorUsageVersion="1">`

InvoiceDate

Holds the invoice date.

Element Type: Simple element

Content Format: DateTime (CCYY-MM-DDTHH:MM:SS)

Attributes: None

EXAMPLE

`<InvoiceDate>1999-09-20</InvoiceDate>`

InvoiceDetails

Top-level container element that holds all invoice-line-level elements. There is one instance of InvoiceDetails for each line on the invoice.

Element Type: Container element

Content Format: n/a

Attributes: None

InvoiceHeader

Top-level container element that holds all invoice-level header elements.

Element Type: Container element

Content Format: n/a

Attributes: None

InvoiceNumber

Element that holds the invoice number.

Element Type: Simple element

Content Format: String, 1..35 characters

Attributes: None

EXAMPLE

```
<InvoiceNumber>35798A</InvoiceNumber>
```

InvoiceStatus

Defines the status of the invoice, for example original, copy or test.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–10: InvoiceStatus Attributes

stdValue	Indicates the status of the invoice, see code list below. Default value is 9
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:1225

EXAMPLE

```
<InvoiceStatus stdValue="10"/>
```

Table 3–11: InvoiceStatus stdValue Code List

9	Original
10	Copy
53	Test

NOTE: *Code list values are a subset of UNTDID:1225*

InvoiceSummary

This top-level container element holds all invoice-summary elements. There is one instance of InvoiceSummary for each invoice. InvoiceSummary contains tax-level summaries, invoice totals, and details of payments that have been made on the invoice.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<InvoiceSummary>
<TaxSummary>
       :   :   :
<TaxSummary>
<InvoiceTotals>
       :   :   :
</InvoiceTotals>
<ActualPayment>
       :   :   :
       </ActualPayment>
</InvoiceSummary>
```

InvoiceTotals

This container element holds totals that apply to the whole invoice. The DiscountSummary element need only be present if discounts have been applied to the invoice.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<InvoiceTotals>
       <DiscountSummary>
          <LineItemTotals>600</LineItemTotals>
          <QtyDiscount>50.00</QtyDiscount>
          <SubTotalAfterQtyValueDiscount>550.00
          </SubTotalAfterQtyValueDiscount>
       </DiscountSummary>
       <NetValue>550.00</NetValue>
       <TaxValue>77.91</TaxValue>
       <GrossValue>627.91</GrossValue>
</InvoiceTotals>
```

InvoiceTreatment

Defines the manner in which the invoice is treated, such as if a paper copy is also produced which version (for example, electronic or printed) is used for tax reclaim purposes

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–12: InvoiceTreatment Attributes

stdValue	Indicates the manner in which the invoice is treated, see code list below. Default value is P
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is VISA:INVT

EXAMPLE

```
<InvoiceTreatment stdValue="E">
```

Table 3–13: InvoiceTreatment stdValue Code List

P	Invoice printed and given to purchaser, and then used for tax reclaim
S	Printed, but printed invoice treated as supplemental invoice since electronic copy used for tax reclaim
E	Printed invoice suppressed since electronic master version used for tax reclaim

NOTE: *Code list values are VISA:INVT*

InvoiceType

Defines the type of invoice, such as Invoice or Credit Note.

The element itself is empty, the attribute stdValue denotes the type of invoice.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–14: InvoiceType Attributes

stdValue	Indicates the type of invoice, see code list below Default value is 380
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:1001

EXAMPLE

```
<InvoiceType stdValue="381"/>
```

Table 3–15: InvoiceType stdValue Code List

380	Invoice
381	Credit note

NOTE: *Code list values are a subset of UNTDID:1001*

LineDiscountInfo

This container element holds line-level discount information. This element need only be present if discounts have been applied to the line item.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<LineDiscountInfo>
       <DiscountValue>10</DiscountValue>
       <UnitPricePreDiscount>100.00</UnitPricePreDiscount>
</LineDiscountInfo>
```

LineItemNum

This element holds the invoice line number. It should normally start at 1 and increase by one for each line, that is for each instance of the InvoiceDetails element.

Element Type: Simple element

Content Format: Integer (10)

Attributes: None

EXAMPLE

`<LineItemNum>23</LineItemNum>`

LineItemSubTotal

This element holds the line item sub-total (that is, the UnitPrice x Quantity).

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

`<LineItemSubTotal>1525.0</LineItemSubTotal>`

LineItemTotals

This element holds the sum of the LineItemSubtotal amounts.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

`<LineItemTotals>1575.00<LineItemTotals/>`

LocalCurrencyAmt

This element holds an amount in the local, that is invoice, currency.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<LocalCurrencyAmt>115</LocalCurrencyAmt>
```

Location

This element holds the tax location.

Element Type: Simple element

Content Format: String, 1 to 35 characters

Attributes: None

EXAMPLE

```
<Location>California</Location>
```

Name

This container element holds the elements that hold the Name details.

The contained elements are Name1, Name2 and Name3. Name1 is mandatory, Name2 and Name3 are optional.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<Name>
      <Name1>Accounts Dept</Name1>
      <Name2>Acme Inc</Name2>
</Name>
```

Name1, Name2 and Name3

These elements hold Name details.

Element Type: Simple element

Content Format: String, 1 to 60 characters.

Attributes: None

EXAMPLE

```
<Name>
      <Name1>Accounts Dept</Name1>
      <Name2>Acme Inc</Name2>
</Name>
```

NetValue

This element holds the net value of the invoice.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<NetValue>600.25</NetValue>
```

PartDesc

This element holds the item description.

Element Type: Simple element

Content Format: String, 1 to 80 characters

Attributes: None

EXAMPLE

```
<PartDesc>Hewlett Packard LaserJet 1100</PartDesc>
```

PartNum

This element holds the item part number.

Element Type: Simple element

Content Format: String, 1 to 35 characters

Attributes: None

EXAMPLE

```
<PartNum>C4224A</PartNum>
```

PartNumDetail

This container element holds details about the line item product, such as part number and/or description.

The stdValue attribute denotes the type of part number/description, such as commodity code, vendor's part number, etc.

Element Type: Container element

Content Format: n/a

Table 3–16: PartNumDetail Attributes

stdValue	Indicates the type of part number. Default value is VP.
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:7143

EXAMPLE

```
<PartNumDetail stdValue="BP">
      <PartNum>198D983HGX</PartNum>
      <PartDesc>10cm circlips</PartDesc>
</PartNumDetail>
```

Table 3–17: PartNumDetail stdValue Code List

BP	Buyer's part number
VP	Vendor's part number
CC	Industry commodity code

NOTE: *Code list values are a subset of UNTDID:7143*

Party

Container element that holds the elements that hold party details.

The stdValue attribute denotes the type of party that this element contains.

Element Type: Container element

Content Format: n/a

Table 3–18: Party Attributes

stdValue	Indicates the type of party that is contained in this element. There is no default value.
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:3035

EXAMPLE

```
<Party stdValue="SU">
      <PartyID>GBSMS50</PartyID>
</Party>
```

Table 3–19: Party stdValue Code List

BY	Buyer, or the company buying the goods or services (Corporate)
SU	Supplier, or the provider of goods or services (Merchant)
IV	Invoicee, or the party to whom the invoice is addressed
PE	Payee, or the party who will receive payment
PI	Merchant details, as known by Visa.
DP	Delivery party, or the party that will receive the goods
SF	Ship from party, or the company from which the goods or services will be shipped.

NOTE: *Code list values are based on* `UNTDID:3035`

PartyID

This element holds an identifier for the Party, for example the EAN code, or the Account Code.

Element Type: Simple element

Content Format: String, 1 to 80 characters.

Attributes: None

EXAMPLE

```
<PartyID>A011954</PartyID>
```

PaymentAmount

This container element holds details of payment amounts. It has subelements to hold the local currency amount (which must always be present), and foreign currency payment details (which should only be present if a payment is made in a currency other than the invoice currency).

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<PaymentAmount>
       <LocalCurrencyAmt>115</LocalCurrencyAmt>
       <ForeignCurrencyPayment>
          <ForeignCurrencyAmt>52</ForeignCurrencyAmt>
          <Currency stdValue=”GBP”/>
       </ForeignCurrencyPayment>
</PaymentAmount>
```

PaymentDate

This element holds the date a payment was made.

Element Type: Simple element

Content Format: DateTime (CCYY-MM-DDTHH:MM:SS)

Attributes: None

EXAMPLE

```
<PaymentDate>1999-10-05</PaymentDate>
```

PaymentMean

This element denotes the payment mean (or payment method).

Note that this element's stdValue attribute has a coded value to represent the payment mean (see below). It is possible for this attribute to take the value "OTHER", whereupon the content of PaymentMean will then hold the payment method as free-format text.

Element Type: Simple element

Content Format: String, 1 to 50 characters.

Table 3–20: PaymentMean Attributes

stdValue	Denotes the payment method Default value is ZZZ
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:4461

EXAMPLE

```
<PaymentMean stdValue="10"/>
```

or

```
<PaymentMean stdValue="OTHER">Some other payment method</PaymentMean>
```

Table 3–21: PaymentMean stdValue Code List

10	In cash
20	Cheque
30	Credit transfer
ZZZ	Credit / debit card
OTHER	Indicates element content will hold textual payment mean

NOTE: *Code list values are a subset of UNTDID:4461, with the addition of the code OTHER*

Note also that code ZZZ (mutually defined) is used to represent credit/debit card, as there is no existing code in the list for this purpose.

POLineNum

This element holds the line number on the original purchase order to which this invoice line item refers.

Element Type: Simple element

Content Format: Integer (10)

Attributes: None

EXAMPLE

```
<POLineNum>26</POLineNum>
```

PONum

This element holds the original purchase order to which this invoice refers.

Element Type: Simple element

Content Format: String, 1 to 35

Attributes: None

EXAMPLE

```
<PONum>B46893</PONum>
```

PostalCode

This element holds the PostalCode element of an address. For example, in the U.S. it would be the ZIP code, in the U.K. it would be the postcode.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<PostalInfo>
       <City>San Francisco</City>
       <CountrySubEntity>CA</CountrySubEntity>
       <PostalCode>00000</PostalCode>
       <Country>USA</Country>
</PostalInfo>
```

PostalInfo

This container element holds the elements that hold a Party's address PostalInfo details. This includes the city, state or county (or similar), ZIP code or postcode (or similar), and country.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<PostalInfo>
      <City>San Francisco</City>
      <CountrySubEntity>CA</CountrySubEntity>
      <PostalCode>00000</PostalCode>
      <Country>USA</Country>
</PostalInfo>
```

Qty

This element holds the actual quantity.

Element Type: Simple element

Content Format: Quantity (Decimal 15.4)

Attributes: None

EXAMPLE

```
   <Qty>100</Qty>
```

QtyDiscount

This element holds the invoice quantity discount amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<QtyDiscount>50.0<QtyDiscount/>
```

Quantity

This container element holds quantity details, such as quantity and unit of measure.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<Quantity>
      <Qty>3</Qty>
      <UnitOfMeasure stdValue="DAY"/>
</Quantity>
```

Ref

This element holds a general reference.

It is also used to hold sector-specific data, and this usage is documented in the Sector-Specific Mapping section of this document.

The code list below documents codes that are common to all sectors.

Element Type: Simple element

Content Format: String, 1 to 80 characters.

Table 3–22: Ref Attributes

stdValue	Indicates the purpose of the data in the element. Default value is VA
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:1153

EXAMPLE

```
<Ref stdValue="VA">GB109 2345 7123</Ref>
```

Table 3–23: Ref stdValue Code List (1 of 2)

VA	Tax registration number
XA	Company/place registration number

Table 3–23: Ref stdValue Code List (2 of 2)

AWE	Cost centre
IV	Invoice number (used on a credit note to refer to the original invoice number)
ACD	Additional reference number. Within the Party "PI" element's Ref element, used to hold the Acquirer reference number (internal Visa-usage only – Merchant's do not need to create a Ref element with this qualifer/data)
ACW	Reference number to previous message. Within the Party "PI" element's Ref element, holds Last Message ID (internal Visa-usage only – Merchant's do not need to create a Ref element with this qualifer/ data)
ADQ	Unique market reference. Within Ref at header level this is used to indicate Sector Type. Within the Party "PI" element's Ref element, this indicates the Merchant Category Code.
ANV	Refund Notice Number (used on an airline ticket credit note to refer to the BSP Refund Notice number).
BN	Booking Reference (typically used to identify who made the booking on behalf of the traveller)

NOTE: *Code list values are a subset of UNTDID:1153*

SpecialCond

This element is used to hold any special conditions to which the line item is subject.

Note that this element's stdValue attribute has a coded value to represent the special condition (see below). It is possible for this attribute to take the value "OTHER", whereupon the content of SpecialCond will then hold the special condition as free-format text.

Element Type: Simple element

Content Format: String, 1 to 50 characters.

Table 3–24: SpecialCond Attributes

stdValue	Denotes the special condition There is no default value
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:4183

EXAMPLE

```
<SpecialCond stdValue="97"/>
```

or

```
<SpecialCond stdValue="OTHER">Some other special condition</SpecialCond>
```

Table 3–25: SpecialCond stdValue Code List (1 of 2)

6	Subject to bonus
7	Subject to commission
11	Price includes excise
12	Price includes tax
18	Item subject to national export restrictions
97	Promotional price
94	Service

Table 3–25: SpecialCond stdValue Code List (2 of 2)

103	Loan
104	Rental
105	Processing
106	Exchange
140	Return of goods
OTHER	Indicates element content will hold textual special condition

NOTE: *Code list values are a subset of UNTDID:4183, with the addition of the code OTHER*

Street

This container element holds the elements that hold a Party's address Street details.

The contained elements are Street1, Street2, Street3 and Street4. Street1 is mandatory, Street2, Street3 and Street4 are optional.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<Street>
      <Street1>Acme House</Street1>
      <Street2>1 Main Street</Street2>
</Street>
```

Street1, Street2, Street3 and Street4

These elements hold a Party's Street details.

Element Type: Simple element

Content Format: String, 1 to 55 characters.

Attributes: None

EXAMPLE

```
<Street>
      <Street1>Acme House</Street1>
      <Street2>1 Main Street</Street2>
</Street>
```

SubLineItemNum

This element holds the invoice subline number. It should normally start at 1 and increase by one for each subline.

Note that this element must not be present unless a subline is being used.

Element Type: Simple element

Content Format: Integer (10)

Attributes: None

EXAMPLE

```
<SubLineItemNum>1</SubLineItemNum>
```

SubTotalAfterQtyValueDiscount

This element holds the sub-total amount after quantity and value discounts have been applied, which is LineItemTotals – QtyDiscount – ValueDiscount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<SubTotalAfterQtyValueDiscount>1500.0<SubTotalAfterQtyValueDiscount/>
```

Tax

This container element holds tax information.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<Tax>
      <TaxFunction stdValue="7"/>
      <TaxType stdValue="VAT"/>
      <TaxCategory stdValue="S"/>
      <TaxPercent>17.5</TaxPercent>
</Tax>
```

TaxableAmount

This element holds the taxable amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<TaxableAmount>200.0</TaxableAmount>
```

TaxAmount

This element holds the tax amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<TaxAmount>35.0</TaxAmount>
```

TaxCategory

This element denotes the tax category.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–26: TaxCategory Attributes

stdValue	Indicates the tax category Default value is S
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:5305

EXAMPLE

```
<TaxCategory stdValue="Z"/>
```

Table 3–27: TaxCategory stdValue Code List

A	Mixed
E	Exempt
G	Free export item
S	Standard
Z	Zero

NOTE: *Code list values are a subset of UNTDID:5305*

TaxFunction

This element denotes the function of the parent Tax element, for example Tax, Customs duty.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–28: TaxFunction Attributes

stdValue	Indicates the function of the Tax element Default value is 7
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:5283

EXAMPLE

```
<TaxFunction stdValue="7"/>
```

Table 3–29: TaxFunction stdValue Code List

5	Customs duty
7	Tax

NOTE: *Code list values are a subset of UNTDID:5283*

TaxPercent

This element holds the tax rate, shown as a percentage.

Element Type: Simple element

Content Format: Percentage (Decimal 3.4)

Attributes: None

EXAMPLE

```
<TaxPercent>17.5</TaxPercent>
```

TaxPointDate

Holds the invoice tax point date.

Element Type: Simple element

Content Format: DateTime (CCYY-MM-DDTHH:MM:SS)

Attributes: None

EXAMPLE

```
<TaxPointDate>1999-09-20</TaxPointDate>
```

TaxSummary

This container element has subelements to hold summary discount and tax information. There must be one TaxSummary element for each tax category code in the invoice. DiscountSummary must hold information about any discounts that have been applied for the tax category in the corresponding Tax element. If no discounts have been applied for the current tax category, then DiscountSummary need not be present.

Element Type: Container element

Content Format: n/a

Attributes: None

EXAMPLE

```
<TaxSummary>
      <DiscountSummary>
         :  :  :
      </DiscountSummary>
      <Tax>
         :  :  :
      </Tax>
</TaxSummary>
```

TaxTreatment

Defines the tax treatment, such as whether gross or net pricing is used and whether tax is calculated at line or invoice level.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–30: TaxTreatment Attributes

stdValue	Indicates how tax is treated in the invoice. Default value is NLL
stdName	Indicates the code list from which the stdValue element has been obtained. Fixed value is VISA:TAXT

EXAMPLE

```
<TaxTreatment stdValue="GLL" stdName="VISA:TAXT"/>
```

Table 3–31: TaxTreatment stdValue Code List

NIL	Line item amounts are net amounts, and tax is calculated at invoice level
GIL	Line item amounts are gross amounts, and tax is calculated at invoice level
NLL	Line item amounts are net amounts, and tax is calculated at line level
GLL	Line item amounts are gross amounts, and tax is calculated at line level
NON	Tax does not apply to this invoice

NOTE: *Code list values are VISA:TAXT*

TaxType

This element denotes the type of tax, for example, VAT, GST.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–32: TaxType Attributes

stdValue	Indicates the type of tax Default value is VAT
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNTDID:5153

EXAMPLE

```
<TaxType stdValue="GST"/>
```

Table 3–33: TaxType stdValue Code List

VAT	Value Added Tax
GST	Goods and Services Tax
STT	State/Provincial Tax

NOTE: *Code list values are a subset of UNTDID:5153*

TaxValue

This element holds the tax value of the invoice, that is, the total tax amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<TaxValue>105.04</TaxValue>
```

TelNum

This element holds a telephone number.

Element Type: Simple element

Content Format: String, 1 to 35 characters.

Attributes: None

EXAMPLE

```
<Contact>
   <TelNum>01420 541667</TelNum>
</Contact>
```

UnitOfMeasure

This element denotes the unit of measure of the corresponding Qty element value.

Element Type: Simple element (Empty)

Content Format: n/a

Table 3–34: UnitOfMeasure Attributes

stdValue	Denotes the unit of measure Default value is EA
stdName	Indicates the code list from which the stdValue element has been obtained. Default value is UNECE:20

EXAMPLE

```
<UnitOfMeasure stdValue="DAY"/>
```

or – assuming default of EA:

```
<UnitOfMeasure/>
```

Table 3–35: UnitOfMeasure stdValue Code List

CMT	Centimeter
DAY	Day
EA	Each
GRM	Gram
HUR	Hour
KGM	Kilogram
KTM	Kilometer
LTR	Liter
MIN	Minute
MTR	Meter
SEC	Second

NOTE: *Code list values are a subset of UNECE:20.*

UnitOfMeasure codes are used in sector specific implementations, please refer to the relevant Sector-Specific Mapping section for code list values used.

UnitPrice

This element holds the line item unit price.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<UnitPrice>15.25</UnitPrice>
```

UnitPricePreDiscount

This element holds the line item unit price, before any line-level discounts have been applied.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<UnitPricePreDiscount>15.25</UnitPricePreDiscount>
```

ValueDiscount

This element holds the invoice value discount amount.

Element Type: Simple element

Content Format: MonetaryAmount (Decimal 18.3)

Attributes: None

EXAMPLE

```
<ValueDiscount>25.0<ValueDiscount/>
```

Sector-Specific Mapping 4

This chapter documents the mapping of data items that are specific to particular market sectors:

- Lodging
- Car Rental
- Passenger Itinerary

Each section includes applicable reference codes, date codes, and an example XML invoice file. Working versions of these examples are included in the Visa XML Invoice Technical Pack. The Technical Pack contains the latest version of the DTD and Stylesheet which may not correspond in every detail with the examples included in this chapter, which are provided for illustration purposes only.

It is necessary to be familiar with the information in the preceding chapters to understand the XML invoice file structure.

Lodging

This section documents the mapping of data items that are specific to the Lodging sector, and it should be used as an addendum to the preceding sections of this document.

Sector-specific data items are mapped to the generic Ref and Date elements, with suitable qualifiers to indicate their usage. Commodity codes are also provided.

For example, the room rate is mapped to the InvoiceHeader Ref element, with a qualifier of RMRT, as shown here:

```
<Ref stdValue =”RMRT” stdName=”VISA:REF”>128.89</Ref>
```

Note that the default stdName value does not apply here because Visa codes are used, therefore the default stdName value must be overridden with the appropriate value as documented in the tables below.

The guest name must be mapped to the Buyer Party/Contact/Name1 field.

Ref Codes

Table 4–1 contains the codes that are used when mapping to a Ref element, and whether a header-level, line-level or subline-level Ref element should be used.

This table is based on code list VISA:REF

Table 4–1: Lodging Ref Code Mapping

Code Value	Description	Format	Level
RMNO	Room number	String, 1 to 80 characters	Header
RMRT	Room rate	MonetaryAmount (Decimal 18.3)	Header
RSNO	Reservation number	String, 1 to 80 characters	Header

Date Codes

Table 4–2 contains the codes that are used when mapping to a Date element, and whether a header-level, line-level or subline-level Date element should be used.

All values written to the Date element should be in the DateTime format, such as CCYY-MM-DDTHH:MM:SS, although some elements may have specific DateTime format requirements and these are documented below.

This table is based on code list VISA:DATE

Table 4–2: Lodging Date Code Mapping

Code Value	Description	Format	Level
STRT	Check in date–time	DateTime	Header
END	Check out date–time	DateTime	Header

Lodging Commodity Codes

Table 4–3 contains the industry commodity codes in the PartNum element for the Lodging sector when the PartNumDetail stdValue attribute equals “CC”.

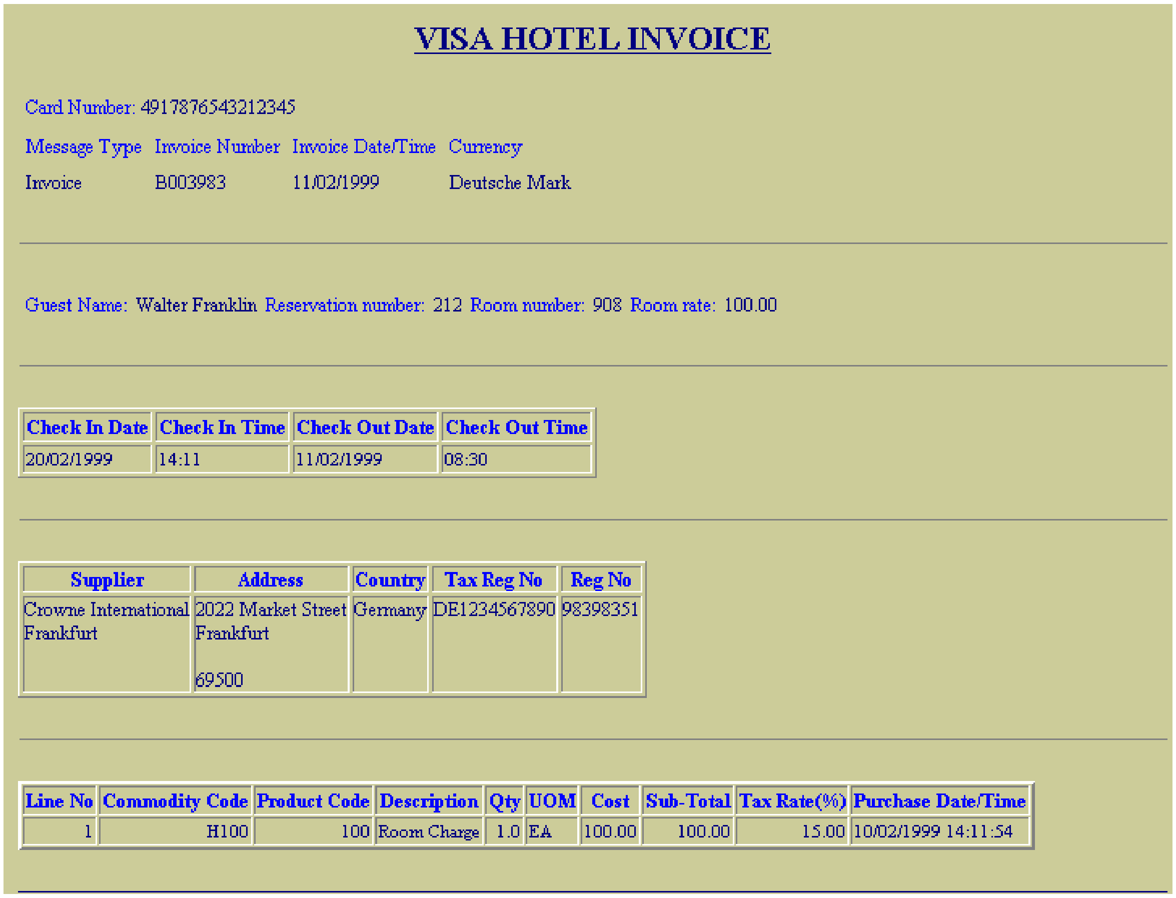
Table 4–3: Lodging Commodity Codes

Code	Description
H100	Room charges
H210	Restaurant food and beverage
H220	Banquet food and beverage
H230	Room service
H240	Minibar
H310	Parking
H320	Laundry
H400	Business center and office services
H500	Telephone and fax

Example Lodging XML Invoice File

Figure 4–1 and Figure 4–2 display the output of the following XML code example. The figure is divided into two images strictly due to the limitations of the printed page. The code does create a single invoice document.

Figure 4–1: Output for Lodging XML Invoice Example File, part 1 of 2



VISA HOTEL INVOICE

Card Number: 4917876543212345

Message Type	Invoice Number	Invoice Date/Time	Currency
Invoice	B003983	11/02/1999	Deutsche Mark

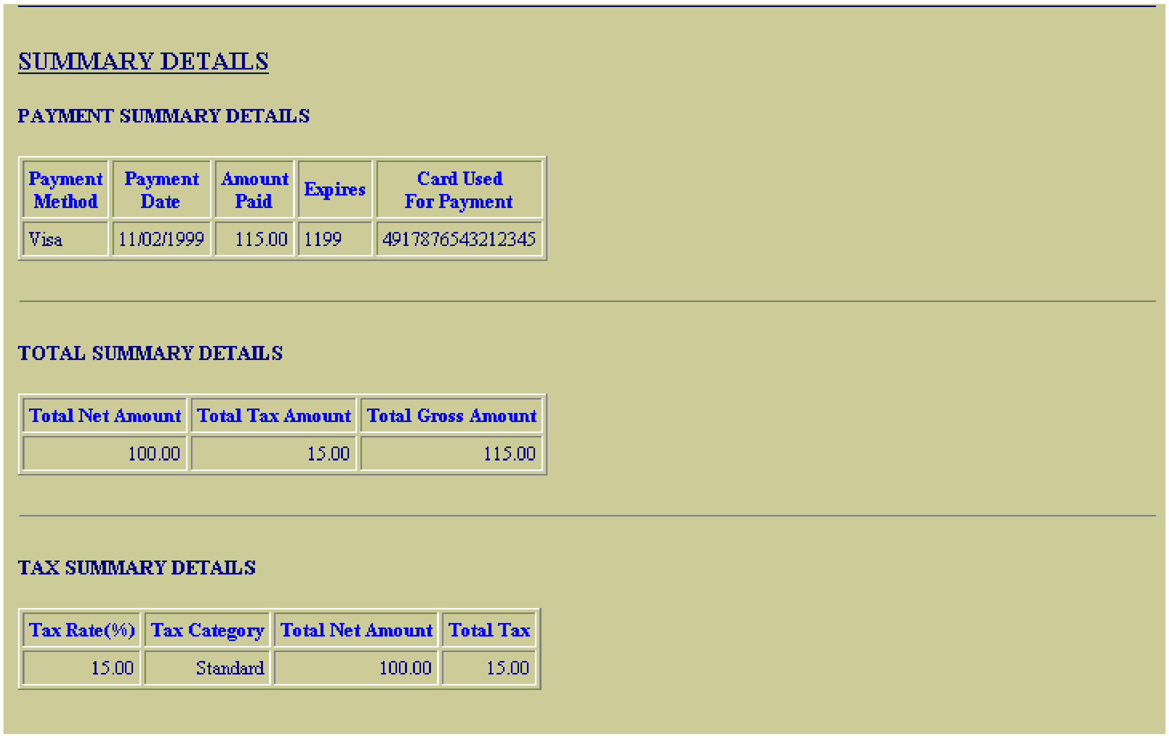
Guest Name: Walter Franklin Reservation number: 212 Room number: 908 Room rate: 100.00

Check In Date	Check In Time	Check Out Date	Check Out Time
20/02/1999	14:11	11/02/1999	08:30

Supplier	Address	Country	Tax Reg No	Reg No
Crowne International Frankfurt	2022 Market Street Frankfurt 69500	Germany	DE1234567890	98398351

Line No	Commodity Code	Product Code	Description	Qty	UOM	Cost	Sub-Total	Tax Rate(%)	Purchase Date/Time
1	H100	100	Room Charge	1.0	EA	100.00	100.00	15.00	10/02/1999 14:11:54

Figure 4–2: Output for Lodging XML Invoice Example File, part 2 of 2



SUMMARY DETAILS

PAYMENT SUMMARY DETAILS

Payment Method	Payment Date	Amount Paid	Expires	Card Used For Payment
Visa	11/02/1999	115.00	1199	4917876543212345

TOTAL SUMMARY DETAILS

Total Net Amount	Total Tax Amount	Total Gross Amount
100.00	15.00	115.00

TAX SUMMARY DETAILS

Tax Rate(%)	Tax Category	Total Net Amount	Total Tax
15.00	Standard	100.00	15.00

EXAMPLE

The following is example code for the Lodging Invoice shown in Figure 4–1 and Figure 4–2 above.

```
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<?xml:stylesheet type="text/xsl" href="invoice.1.0.xsl"?>
<Invoice sectorUsageVersion="1">
   <InvoiceHeader>
      <InvoiceType/>
      <InvoiceStatus/>
      <TaxTreatment/>
      <DiscountTreatment/>
      <InvoiceTreatment/>
      <InvoiceNumber>B003983</InvoiceNumber>
      <InvoiceDate>1999-02-11</InvoiceDate>
      <Currency stdValue="DEM"/>
      <Party stdValue="SU">
         <PartyID>5011234567890</PartyID>
         <Name>
            <Name1>Crowne International</Name1>
            <Name2>Frankfurt</Name2>
         </Name>
         <Street>
            <Street1>2022 Market Street</Street1>
         </Street>
         <PostalInfo>
            <City>Frankfurt</City>
            <CountrySubEntity/>
            <PostalCode>69500</PostalCode>
            <Country>Germany</Country>
         </PostalInfo>
         <Contact>
            <TelNum>+49 812 1234 222</TelNum>
            <Function>Accounts Dept</Function>
         </Contact>
         <Ref>DE1234567890</Ref>
         <Ref stdValue="XA">98398351</Ref>
      </Party>
      <Party stdValue="BY">
         <PartyID>LC100</PartyID>
         <Name>
            <Name1>Walter Franklin</Name1>
            <Name2>eCommerce Department</Name2>
            <Name3>Large Company Inc.</Name3>
         </Name>
         <Street>
            <Street1>Metro 1</Street1>
            <Street2>Metro Boulevard</Street2>
            <Street3>Ludgate Circus</Street3>
         </Street>
         <PostalInfo>
            <City>Foster City</City>
            <CountrySubEntity>CA</CountrySubEntity>
            <PostalCode>95118</PostalCode>
            <Country>USA</Country>
         </PostalInfo>
         <Contact>
            <Name1>Walter Franklin</Name1>
         </Contact>
         <Ref>CA12345678901234</Ref>
      </Party>
      <Party stdValue="PI">
         <PartyID>CROWNE002</PartyID>
         <Name>
            <Name1>CROWNE HOTEL GROUP</Name1>
         </Name>
         <PostalInfo>
            <City>HAMBURG</City>
            <PostalCode>00000</PostalCode>
            <Country>DE</Country>
         </PostalInfo>
         <Ref stdValue="ADQ">200000</Ref>
      </Party>
      <Payment>
         <PaymentDueDate>
            <RelativeDate>
               <RefDate/>
               <TimeRelation/>
               <TypeOfPeriod/>
               <NumberOfPeriods>30</NumberOfPeriods>
            </RelativeDate>
         </PaymentDueDate>
         <CardInfo>
            <CardNum>4917876543212345</CardNum>
            <CardExpirationDate>1199</CardExpirationDate>
            <CardType/>
         </CardInfo>
      </Payment>
      <PONum>PO00001</PONum>
      <Ref stdValue="ADQ">LG</Ref>
      <Ref stdValue="AWE">98345</Ref>
      <Ref stdValue="RMNO" stdName="VISA:REF">908</Ref>
      <Ref stdValue="RSNO" stdName="VISA:REF">212</Ref>
      <Ref stdValue="RMRT" stdName="VISA:REF">100.00</Ref>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-20T14:11:54</Date>
      <Date stdValue="END" stdName="VISA:DATE">1999-02-11T08:30:12</Date>
   </InvoiceHeader>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>1</LineItemNum>
         <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
         </PartNumDetail>
         <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
         </PartNumDetail>
         <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>100.00</UnitPrice>
      <LineItemSubtotal>100.00</LineItemSubtotal>
      <Tax>
         <TaxFunction/>
         <TaxType/>
         <TaxCategory stdValue="S"/>
         <TaxPercent>15.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
   </InvoiceDetails>
   <InvoiceSummary>
      <TaxSummary>
         <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>15.00</TaxPercent>
            <TaxableAmount>100</TaxableAmount>
            <TaxAmount>15.00</TaxAmount>
         </Tax>
      </TaxSummary>
      <InvoiceTotals>
         <NetValue>100.00</NetValue>
         <TaxValue>15.00</TaxValue>
         <GrossValue>115.00</GrossValue>
      </InvoiceTotals>
      <ActualPayment>
         <PaymentAmount>
```

```
            <LocalCurrencyAmt>115.00</LocalCurrencyAmt>
         </PaymentAmount>
         <PaymentMean/>
         <PaymentDate>1999-02-11</PaymentDate>
         <CardInfo>
            <CardNum>4917876543212345</CardNum>
            <CardExpirationDate>1199</CardExpirationDate>
            <CardType/>
         </CardInfo>
      </ActualPayment>
   </InvoiceSummary>
</Invoice>
```

Car Rental

This section documents the mapping of data items that are specific to the Car Rental sector, and it should be used as an addendum to the preceding sections of this document.

Sector-specific data items are mapped to the generic Ref and Date elements, with suitable qualifiers to indicate their usage.

For example, the Vehicle Registration No. is mapped to the InvoiceHeader Ref element, with a qualifier of VREG, for example:

```
<Ref stdValue =”VREG” stdName=”VISA:REF”>V698VCF</Ref>
```

Note that the default stdName value does not apply here because Visa codes are used, therefore the default stdName value must be overridden with the appropriate value as documented in the tables below.

The car renter name must be mapped to the Buyer Party/Contact/Name1 field.

Ref Codes

Table 4–4 contains the codes that are used when mapping to a Ref element, and whether a header-level, line-level or subline-level Ref element should be used.

This table is based on code list VISA:REF

Table 4–4: Car Rental Ref Code Mapping (1 of 2)

Code Value	Description	Format	Level
DCI	The number of miles or km at check-in	Integer (10)	Header
DCO	The number of miles or km at check-out	Integer (10)	Header
DDV	The number of miles or km driven	Integer (10)	Header
LOC1	Location from which vehicle was rented	String, 1 to 80 characters	Header
LOC2	Location to which vehicle was returned	String, 1 to 80 characters	Header

Table 4–4: Car Rental Ref Code Mapping (2 of 2)

Code Value	Description	Format	Level
OVD	Other vehicle data	String, 1 to 80 characters	Header
RSNO	Reservation no.	String, 1 to 80 characters	Header
UOD	Unit of distance, coded	Permissible values are M or KM	Header
VGCH	Vehicle group charged	String, 1 to 80 characters	Header
VREG	Vehicle registration number	String, 1 to 80 characters	Header
VT	Vehicle type car or van	String, 1 to 80 characters	Header

Date Codes

Table 4–5 contains the codes that are used when mapping to a Date element, and whether a header-level, line-level or subline-level Date element should be used.

All values written to the Date element should be in the DateTime format, such as CCYY-MM-DDTHH:MM:SS, although some elements may have specific DateTime format requirements and these are documented below.

This table is based on code list VISA:DATE

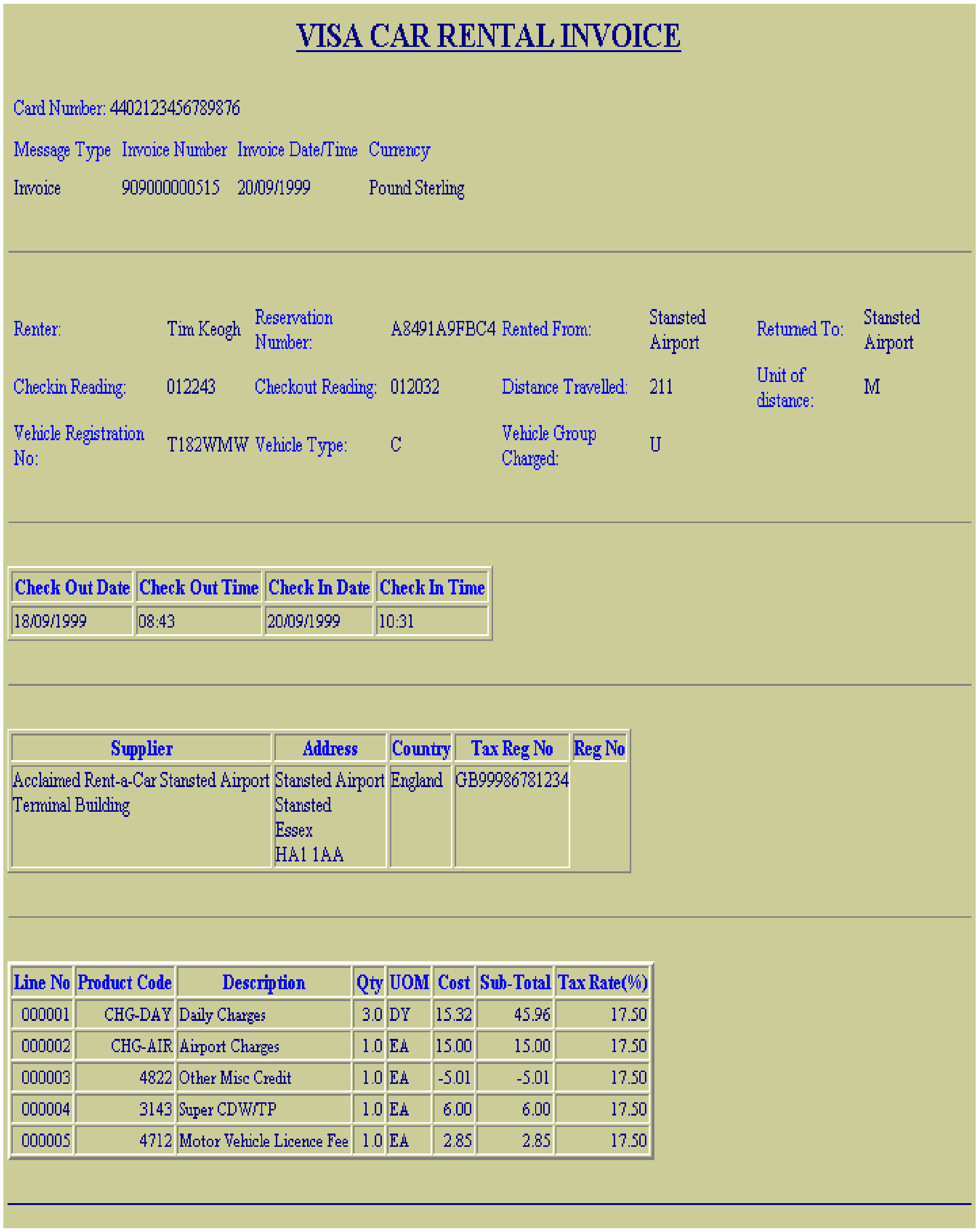
Table 4–5: Car Rental Date Code Mapping

Code Value	Description	Format	Level
STRT	Check out date–time	DateTime	Header
END	Check in date–time	DateTime	Header

Example Car Rental XML Invoice File

Figure 4–3 and Figure 4–4 display the output of the following XML code example. The figure is divided into two images strictly due to the limitations of the printed page. The code does create a single invoice document.

Figure 4–3: Output for Car Rental XML Invoice Example File, part 1 of 2



VISA CAR RENTAL INVOICE

Card Number: 4402123456789876

Message Type	Invoice Number	Invoice Date/Time	Currency
Invoice	909000000515	20/09/1999	Pound Sterling

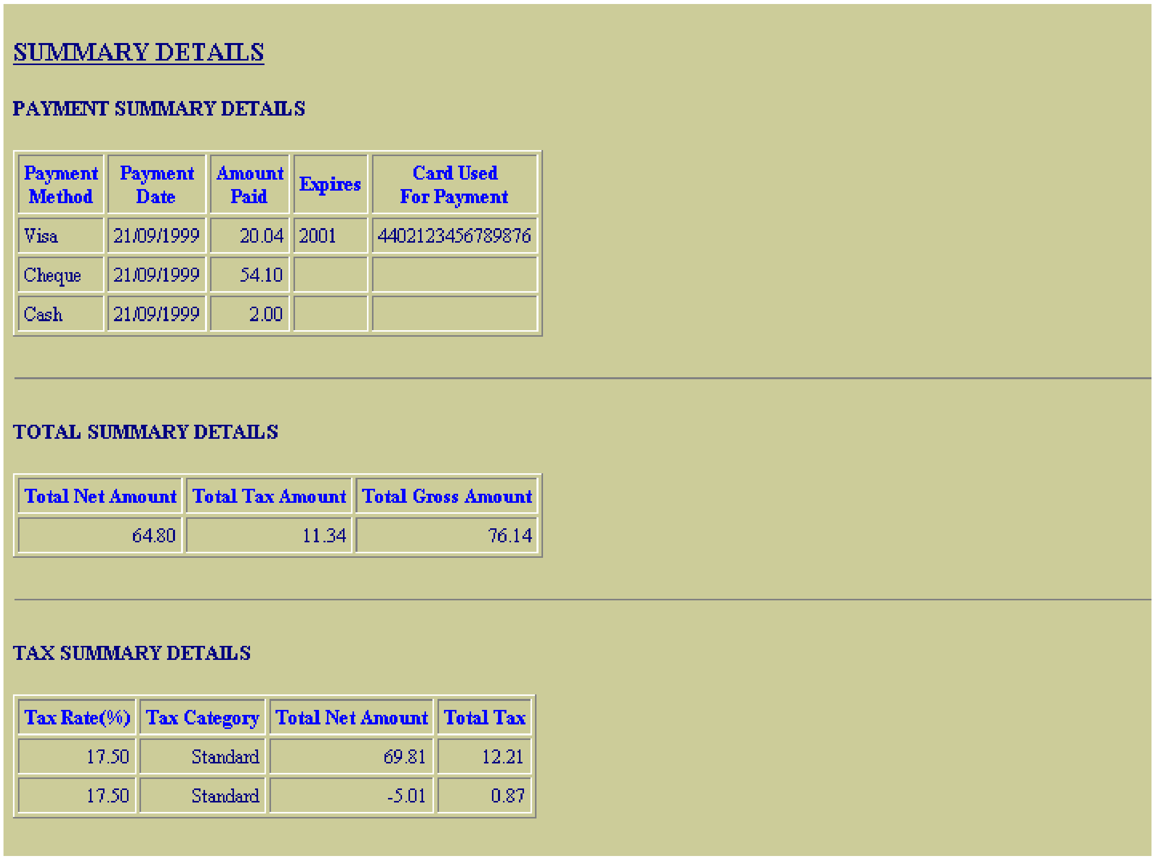
Renter:	Tim Keogh	Reservation Number:	A8491A9FBC4	Rented From:	Stansted Airport	Returned To:	Stansted Airport
Checkin Reading:	012243	Checkout Reading:	012032	Distance Travelled:	211	Unit of distance:	M
Vehicle Registration No:	T182WMW	Vehicle Type:	C	Vehicle Group Charged:	U		

Check Out Date	Check Out Time	Check In Date	Check In Time
18/09/1999	08:43	20/09/1999	10:31

Supplier	Address	Country	Tax Reg No	Reg No
Acclaimed Rent-a-Car Stansted Airport Terminal Building	Stansted Airport Stansted Essex HA1 1AA	England	GB99986781234	

Line No	Product Code	Description	Qty	UOM	Cost	Sub-Total	Tax Rate(%)
000001	CHG-DAY	Daily Charges	3.0	DY	15.32	45.96	17.50
000002	CHG-AIR	Airport Charges	1.0	EA	15.00	15.00	17.50
000003	4822	Other Misc Credit	1.0	EA	-5.01	-5.01	17.50
000004	3143	Super CDW/TP	1.0	EA	6.00	6.00	17.50
000005	4712	Motor Vehicle Licence Fee	1.0	EA	2.85	2.85	17.50

Figure 4–4: Output for Car Rental XML Invoice Example File, part 2 of 2



SUMMARY DETAILS

PAYMENT SUMMARY DETAILS

Payment Method	Payment Date	Amount Paid	Expires	Card Used For Payment
Visa	21/09/1999	20.04	2001	4402123456789876
Cheque	21/09/1999	54.10		
Cash	21/09/1999	2.00		

TOTAL SUMMARY DETAILS

Total Net Amount	Total Tax Amount	Total Gross Amount
64.80	11.34	76.14

TAX SUMMARY DETAILS

Tax Rate(%)	Tax Category	Total Net Amount	Total Tax
17.50	Standard	69.81	12.21
17.50	Standard	-5.01	0.87

EXAMPLE

The following is example code for the Car Rental Invoice shown in Figure 4–3 and Figure 4–4 above.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml:stylesheet type="text/xsl" href="invoice.1.0.xsl"?>
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<Invoice sectorUsageVersion="1">
   <InvoiceHeader>
      <InvoiceType/>
      <InvoiceStatus/>
      <TaxTreatment/>
      <DiscountTreatment/>
      <InvoiceTreatment/>
      <InvoiceNumber>909000000515</InvoiceNumber>
      <InvoiceDate>1999-09-20</InvoiceDate>
      <Currency stdValue="GBP"/>
      <Party stdValue="SU">
         <PartyID>GBSMS50</PartyID>
         <Name>
            <Name1>Acclaimed Rent-a-Car Stansted Airport</Name1>
            <Name2>Terminal Building</Name2>
         </Name>
         <Street>
```

```
            <Street1>Stansted Airport</Street1>
         </Street>
         <PostalInfo>
            <City>Stansted</City>
            <CountrySubEntity>Essex</CountrySubEntity>
            <PostalCode>HA1 1AA</PostalCode>
            <Country>England</Country>
         </PostalInfo>
         <Contact>
            <TelNum>+44 777 1234</TelNum>
         </Contact>
         <Ref>
            GB99986781234
         </Ref>            <!--VAT Number    -->
      </Party>
      <Party stdValue="BY">
         <!--Buyer-->
         <PartyID/>
         <Name>
            <Name1>Tim Keogh</Name1>
            <Name2>Unknown Company Inc</Name2>
         </Name>
         <Street>
            <Street1>206 Dalmally Street</Street1>
         </Street>
         <PostalInfo>
            <City>East Croydon</City>
            <CountrySubEntity>London</CountrySubEntity>
            <PostalCode>CR0 9TT</PostalCode>
            <Country>England</Country>
         </PostalInfo>
         <Contact>
            <TelNum>0181 683 7475</TelNum>
            <Function>Home Address</Function>
         </Contact>
         <Ref>GB109 2345 7123</Ref>
      </Party>
      <Payment>
         <PaymentDueDate>
            <RelativeDate>
               <RefDate/>
               <!-- Date of Invoice, Payment due date-->
               <TimeRelation/>
               <!-- After reference date-->
               <TypeOfPeriod/>
               <!-- calendar days-->
               <NumberOfPeriods>30</NumberOfPeriods>
            </RelativeDate>
         </PaymentDueDate>
         <PaymentTerms>
            <PaymentTermType/>
            <AbsoluteDate>1999-09-30</AbsoluteDate>
            <DiscountPercent/>
         </PaymentTerms>
       <CardInfo>
             <CardNum>4402123456789876</CardNum>
         </CardInfo>
      </Payment>
      <PONum>241185265</PONum>
      <Ref stdValue="ADQ">CR</Ref>
      <Ref stdValue="LOC1" stdName="VISA:REF">
         Stansted Airport
         <!--Location Rented From-->
      </Ref>
      <Ref stdValue="LOC2" stdName="VISA:REF">
         Stansted Airport
      </Ref>         <!--Location Returned to-->
      <Ref stdValue="VGCH" stdName="VISA:REF">
         U
      </Ref>         <!--Vehicle Group Charged-->
      <Ref stdValue="UOD" stdName="VISA:REF">
         M
      </Ref>         <!--Unit of Distance-->
      <Ref stdValue="DCI" stdName="VISA:REF">
         012243
      </Ref>         <!--Mileage/KM at Check-In-->
      <Ref stdValue="DCO" stdName="VISA:REF">
         012032
      </Ref>        <!--Mileage/KM at Check-Out-->
       <Ref stdValue="DDV" stdName="VISA:REF">
         211
      </Ref>        <!--Mileage/KM driven-->
       <Ref stdValue="VREG" stdName="VISA:REF">
         T182WMW
      </Ref>        <!--Vehicle Registration Number-->
       <Ref stdValue="VT" stdName="VISA:REF">
         C
      </Ref>        <!--Vehicle Type Car or Van-->
       <Ref stdValue="RSNO" stdName="VISA:REF">
          A8491A9FBC4
      </Ref>        <!--Reservation Number-->
       <Date stdValue="STRT" stdName="VISA:DATE">
         1999-09-18T08:43:00
      </Date>        <!--Check Out Date and Time-->
       <Date stdValue="END" stdName="VISA:DATE">
         1999-09-20T10:31:00
      </Date>         <!--Check In Date and Time-->
    </InvoiceHeader>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>1</LineItemNum>
         <PartNumDetail>
            <PartNum>CHG-DAY</PartNum>
            <PartDesc>Daily Charges</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure stdValue="DAY"/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>15.32</UnitPrice>
      <LineItemSubtotal>45.96</LineItemSubtotal>
```

```
      <Tax>
         <TaxFunction/>
         <!--e.g. Tax, Customs duty. Default = 7, Tax-->
         <TaxType/>
         <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
         <TaxCategory stdValue="S"/>
         <!--e.g. Standards, zero-rated. Default = S, Standard-->
         <TaxPercent>17.5</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>2</LineItemNum>
         <PartNumDetail>
            <PartNum>CHG-AIR</PartNum>
            <PartDesc>Airport Charges</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>15.00</UnitPrice>
      <LineItemSubtotal>15.00</LineItemSubtotal>
      <Tax>
         <TaxFunction/>
         <!--e.g. Tax, Customs duty. Default = 7, Tax-->
         <TaxType/>
         <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
         <TaxCategory stdValue="S"/>
         <!--e.g. Standards, zero-rated. Default = S, Standard-->
         <TaxPercent>17.5</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-11</Date>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>3</LineItemNum>
         <PartNumDetail>
            <PartNum>4822</PartNum>
            <PartDesc>Other Misc Credit</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>-5.01</UnitPrice>
      <LineItemSubtotal>-5.01</LineItemSubtotal>
      <Tax>
         <TaxFunction/>
         <!--e.g. Tax, Customs duty. Default = 7, Tax-->
         <TaxType/>
         <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
         <TaxCategory stdValue="S"/>
         <!--e.g. Standards, zero-rated. Default = S, Standard-->
         <TaxPercent>17.5</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10</Date>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>4</LineItemNum>
         <PartNumDetail>
            <PartNum>3143</PartNum>
            <PartDesc>Super CDW/TP</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>6.00</UnitPrice>
      <LineItemSubtotal>6.00</LineItemSubtotal>
      <Tax>
         <TaxFunction/>
         <!--e.g. Tax, Customs duty. Default = 7, Tax-->
         <TaxType/>
         <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
         <TaxCategory stdValue="S"/>
         <!--e.g. Standards, zero-rated. Default = S, Standard-->
         <TaxPercent>17.5</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>5</LineItemNum>
         <PartNumDetail stdValue="VP" stdName="UNTDID:7143">
            <PartNum>4712</PartNum>
            <PartDesc>Motor Vehicle Licence Fee</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure stdValue="EA"/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>2.85</UnitPrice>
      <LineItemSubtotal>2.85</LineItemSubtotal>
      <Tax>
         <TaxFunction/>
         <!--e.g. Tax, Customs duty. Default = 7, Tax-->
         <TaxType/>
         <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
         <TaxCategory stdValue="S"/>
         <!--e.g. Standards, zero-rated. Default = S, Standard-->
         <TaxPercent>17.5</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
   </InvoiceDetails>
   <InvoiceSummary>
```

```
      <!--One for each tax code in the invoice-->
      <TaxSummary>
         <Tax>
            <TaxFunction/>
            <!--e.g. Tax, Customs duty. Default = 7, Tax-->
            <TaxType/>
            <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
            <TaxCategory stdValue="S"/>
            <!--e.g. Standards, zero-rated. Default = S, Standard-->
           <!--Taxable amount equates to SubTotalAfterQtyValueDiscount - SettlementDiscoun-
tAmt if DiscountInfo is used, otherwise it's Sum of LineItemSubtotals for this tax code-->
            <TaxPercent>17.5</TaxPercent>
            <TaxableAmount>69.81</TaxableAmount>
            <TaxAmount>12.21</TaxAmount>
         </Tax>
      </TaxSummary>
      <TaxSummary>
         <Tax>
            <TaxFunction/>
            <!--e.g. Tax, Customs duty. Default = 7, Tax-->
            <TaxType/>
            <!--e.g. VAT, GST. EU default is VAT (US is GST)-->
            <TaxCategory stdValue="S"/>
            <!--e.g. Standards, zero-rated. Default = S, Standard-->
           <!--Taxable amount equates to SubTotalAfterQtyValueDiscount - SettlementDiscoun-
tAmt if DiscountInfo is used, otherwise it's Sum of LineItemSubtotals for this tax code-->
            <TaxPercent>17.5</TaxPercent>
            <TaxableAmount>-5.01</TaxableAmount>
            <TaxAmount>0.87</TaxAmount>
         </Tax>
      </TaxSummary>
      <InvoiceTotals>
         <!--Sum of TaxSummary/Tax/TaxableAmount elements for all tax codes-->
         <NetValue>64.80</NetValue>
         <!--Sum of TaxSummary/Tax/TaxAmount elements for all tax codes-->
         <TaxValue>11.34</TaxValue>
         <!--Sum of TaxSummary/Tax/TaxableAmount and TaxSummary/Tax/TaxAmount for all tax
codes-->
         <GrossValue>76.14</GrossValue>
      </InvoiceTotals>
      <ActualPayment>
         <PaymentAmount>
            <LocalCurrencyAmt>20.04</LocalCurrencyAmt>
         </PaymentAmount>
         <PaymentMean/>
         <PaymentDate>1999-09-21</PaymentDate>
         <CardInfo>
            <CardNum>4402123456789876</CardNum>
            <CardExpirationDate>0101</CardExpirationDate>
            <CardType/>
         </CardInfo>
      </ActualPayment>
      <ActualPayment>
         <PaymentAmount>
            <LocalCurrencyAmt>2.00</LocalCurrencyAmt>
         </PaymentAmount>
         <PaymentMean stdValue="10"/>
         <PaymentDate>1999-09-21</PaymentDate>
      </ActualPayment>
      <ActualPayment>
         <PaymentAmount>
            <LocalCurrencyAmt>54.10</LocalCurrencyAmt>
         </PaymentAmount>
         <PaymentMean stdValue="20"/>
         <PaymentDate>1999-09-21</PaymentDate>
      </ActualPayment>
   </InvoiceSummary>
</Invoice>
```

Passenger Itinerary

This section documents the mapping of data items that are specific to the Passenger Itinerary sector, and it should be used as an addendum to the preceding chapters of this document.

Most data items that are specific to the Passenger Itinerary sector are mapped to the generic Ref and Date elements, with suitable qualifiers to indicate their usage. These data item cross-references are documented in the tables below.

For example, a journey's trip leg flight number is mapped to the InvoiceDetail/Ref element, with a qualifier of FLNO, for example:

```
<Ref stdValue =”FLNO” stdName=”VISA:REF”>367</Ref>
```

Passenger Itinerary sector trip leg information is written to the InvoiceDetail subline—therefore, the SubLineItemNum element must be populated with the Sector Number (Coupon Number) of each trip leg. Note that these numbers might not increment in sequence. For example, it is possible to have a sequence of values 1, 2 and 4.

The passenger name must be mapped to the Buyer Party/Contact/Name1 field.

The SectorFareAmount must be mapped to the LineItemSubTotal element.

Note that the default stdName value does not apply here because Visa codes are used, therefore the default stdName value must be overridden with the appropriate value as documented in the following tables.

Ref Codes

Table 4–6 contains the codes that are used when mapping to a Ref element, and whether a header-level, line-level or subline-level Ref element should be used.

This table is based on code list VISA:REF

Table 4–6: Passenger Itinerary Ref Code Mapping (1 of 2)

Code Value	Description	Format	Level
CARR	Carrier code	String, 1 to 2 characters	Subline
FBC	Fare basis code	String, 1 to 80 characters	Subline
FLNO	Flight number	String, 1 to 5 characters	Subline

Table 4–6: Passenger Itinerary Ref Code Mapping (2 of 2)

Code Value	Description	Format	Level
LOC1	Origination city, coded	String, 1 to 3 characters	Subline
LOC2	Destination city, coded	String, 1 to 3 characters	Subline
REFI	Refund indicator	String, 1 character	Subline
RSNO	Reservation Number	String, 1 to 80 characters	Header
SRVC	Service class	String, 1 character	Subline
STOP	Stopover indicator	String, 1 character	Subline

Date Codes

Table 4–7 contains the codes that are used when mapping to a Date element, and whether a header-level, line-level or subline-level Date element should be used.

All values written to the Date element should be in the DateTime format, such as CCYY-MM-DDTHH:MM:SS, although some elements may have specific DateTime format requirements and these are documented below.

This table is based on code list VISA:DATE

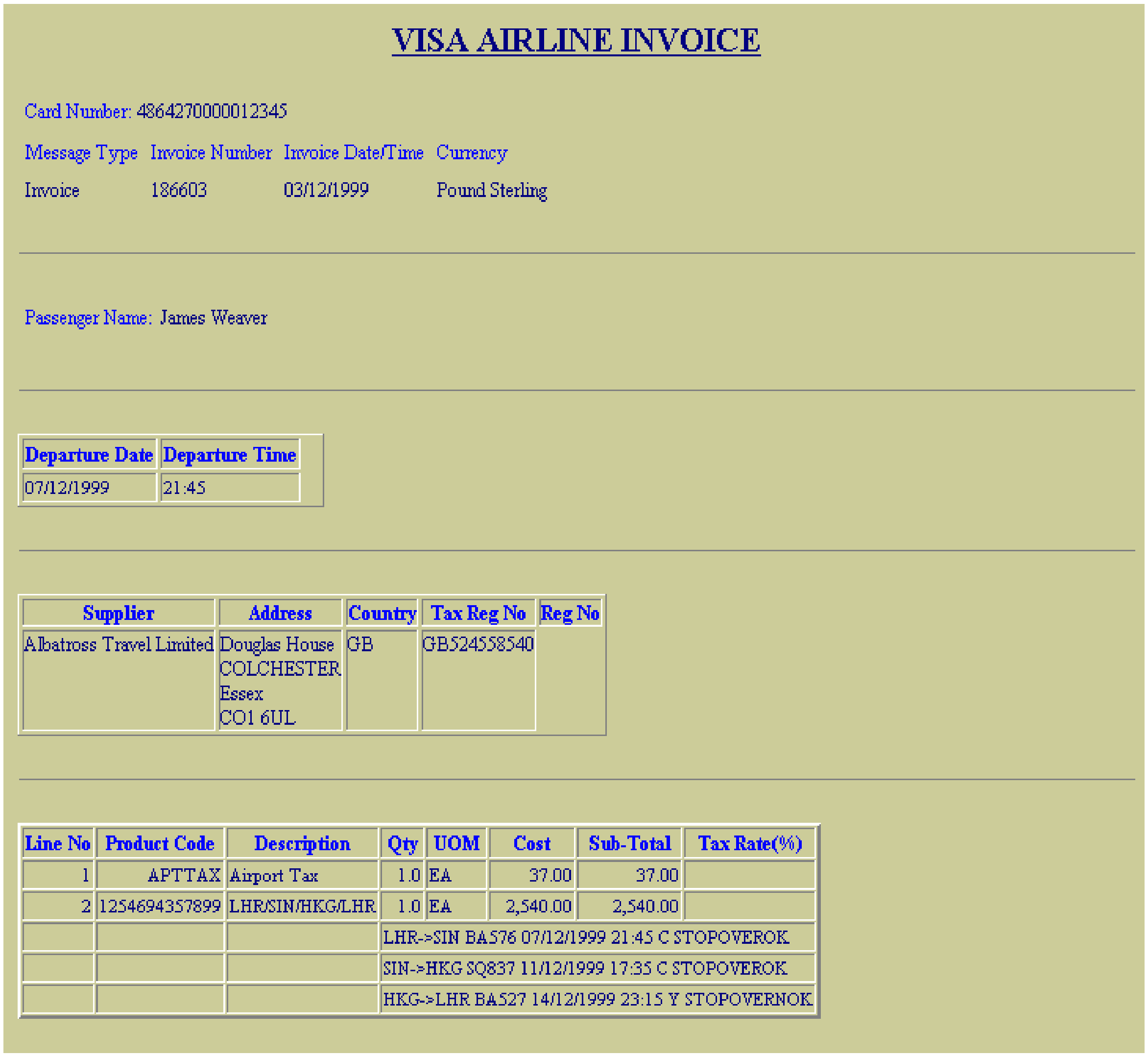
Table 4–7: Passenger Itinerary Date Code Mapping

Code Value	Description	Format	Level
STRT	Departure date–time	DateTime	Header
END	Arrival date–time	DateTime	Header

Example Passenger Itinerary XML Invoice File

Figure 4–5 and Figure 4–6 display the output of the following XML code example. The figure is divided into two images strictly due to the limitations of the printed page. The code does created a single invoice document.

Figure 4–5: Output for Passenger Itinerary XML Invoice Example File, part 1 of 2



VISA AIRLINE INVOICE

Card Number: 4864270000012345

Message Type	Invoice Number	Invoice Date/Time	Currency
Invoice	186603	03/12/1999	Pound Sterling

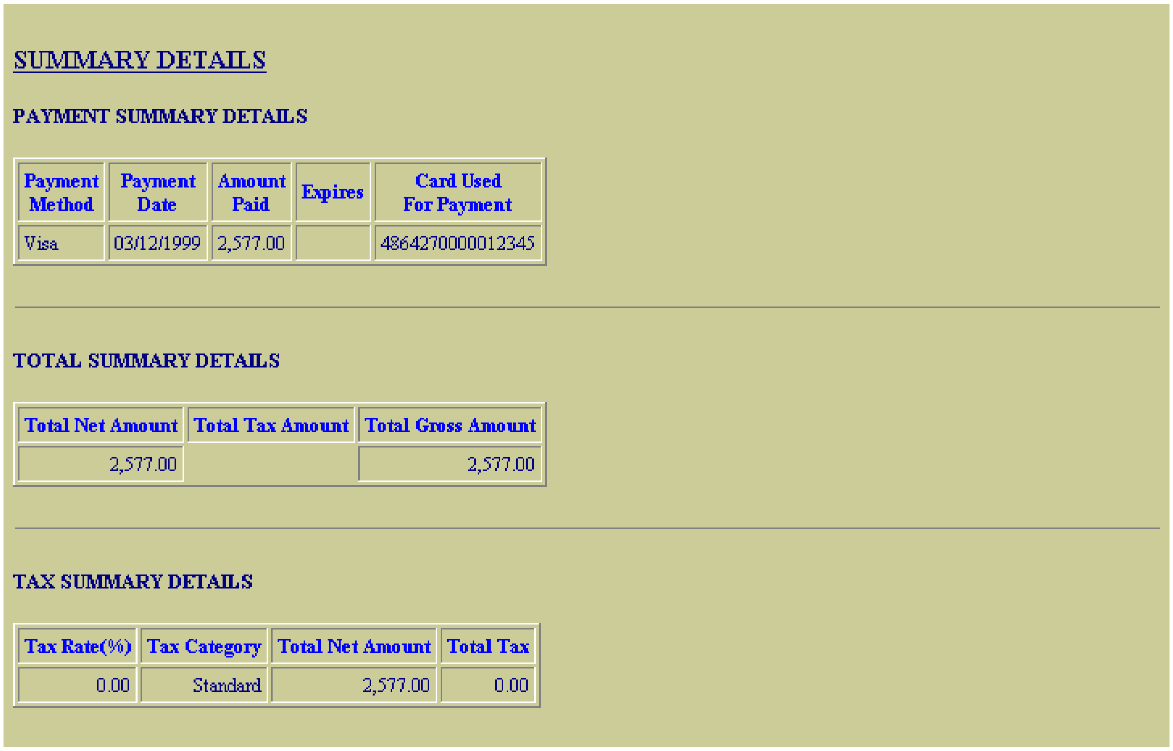
Passenger Name: James Weaver

Departure Date	Departure Time
07/12/1999	21:45

Supplier	Address	Country	Tax Reg No	Reg No
Albatross Travel Limited	Douglas House COLCHESTER Essex CO1 6UL	GB	GB524558540	

Line No	Product Code	Description	Qty	UOM	Cost	Sub-Total	Tax Rate(%)
1	APTTAX	Airport Tax	1.0	EA	37.00	37.00	
2	1254694357899	LHR/SIN/HKG/LHR	1.0	EA	2,540.00	2,540.00	
			LHR->SIN BA576 07/12/1999 21:45 C STOPOVEROK				
			SIN->HKG SQ837 11/12/1999 17:35 C STOPOVEROK				
			HKG->LHR BA527 14/12/1999 23:15 Y STOPOVERNOK				

Figure 4–6: Output for Passenger Itinerary XML Invoice Example File, part 2 of 2



SUMMARY DETAILS

PAYMENT SUMMARY DETAILS

Payment Method	Payment Date	Amount Paid	Expires	Card Used For Payment
Visa	03/12/1999	2,577.00		4864270000012345

TOTAL SUMMARY DETAILS

Total Net Amount	Total Tax Amount	Total Gross Amount
2,577.00		2,577.00

TAX SUMMARY DETAILS

Tax Rate(%)	Tax Category	Total Net Amount	Total Tax
0.00	Standard	2,577.00	0.00

EXAMPLE

The following is example code for the Passenger Itinerary Invoice shown in Figure 4–5 and Figure 4–6 above.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml:stylesheet type="text/xsl" href="invoice.1.0.xsl"?>
<!DOCTYPE Invoice SYSTEM "invoice.1.0.dtd">
<Invoice sectorUsageVersion="0">
   <InvoiceHeader>
      <InvoiceType/>
      <InvoiceStatus/>
      <TaxTreatment stdValue="NIL"/>
      <InvoiceTreatment stdValue="P"/>
      <InvoiceNumber>186603</InvoiceNumber>
      <InvoiceDate>1999-12-03</InvoiceDate>
      <Currency stdValue="GBP"/>
      <Party stdValue="BY">
         <PartyID>S73333</PartyID>
         <Name>
            <Name1>Grosvenor Plastics Ltd</Name1>
         </Name>
         <Street>
            <Street1>1 Grosvenor Square</Street1>
         </Street>
         <PostalInfo>
            <PostalCode>W1 7SB</PostalCode>
            <Country>GB</Country>
         </PostalInfo>
         <Contact>
            <Name1>James Weaver</Name1>
         </Contact>
      </Party>
      <Party stdValue="SU">
         <Name>
            <Name1>Albatross Travel Limited</Name1>
         </Name>
         <Street>
            <Street1>Douglas House</Street1>
            <Street2>Appletree Road</Street2>
            <Street3>Stanford</Street3>
         </Street>
         <PostalInfo>
            <City>COLCHESTER</City>
            <CountrySubEntity>Essex</CountrySubEntity>
            <PostalCode>CO1 6UL</PostalCode>
            <Country>GB</Country>
         </PostalInfo>
         <Ref stdValue="VA">GB524558540</Ref>
      </Party>
      <Party stdValue="PI">
         <PartyID>312345</PartyID>
         <Name>
            <Name1>ALBATROSS TRAVEL LIMITED</Name1>
         </Name>
         <Street>
            <Street1/>
            <Street2>COLCHESTER</Street2>
         </Street>
         <PostalInfo>
            <CountrySubEntity>GB</CountrySubEntity>
         </PostalInfo>
         <Ref stdValue="ADQ">1234</Ref>
      </Party>
      <Ref stdValue="ADQ">AI</Ref>
      <Ref stdValue="AWE">HO475</Ref>
      <Ref stdValue="RSNO">PZDHHX</Ref>
      <Ref stdValue="BN">VX4</Ref>
      <Date stdValue="STRT">1999-12-07T21:45:00</Date>
      <GenText stdValue="AFP">BA</GenText>
   </InvoiceHeader>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>1</LineItemNum>
         <PartNumDetail stdValue="VP">
            <PartNum>APTTAX</PartNum>
            <PartDesc>Airport Tax</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>1.0</Qty>
            <UnitOfMeasure stdValue="EA"/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>37.0</UnitPrice>
      <LineItemSubtotal>37.0</LineItemSubtotal>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>2</LineItemNum>
         <PartNumDetail stdValue="VP">
            <PartNum>1254694357899</PartNum>
            <PartDesc>LHR/SIN/HKG/LHR</PartDesc>
         </PartNumDetail>
         <Quantity>
            <Qty>1.0</Qty>
            <UnitOfMeasure stdValue="EA"/>
         </Quantity>
      </BaseItemDetail>
      <UnitPrice>2540.0</UnitPrice>
      <LineItemSubtotal>2540.0</LineItemSubtotal>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>2</LineItemNum>
         <SubLineItemNum>1</SubLineItemNum>
         <PartNumDetail stdValue="VP">
            <PartNum>BA0015</PartNum>
         </PartNumDetail>
         <Quantity>
            <Qty>1.0</Qty>
            <UnitOfMeasure stdValue="EA"/>
         </Quantity>
      </BaseItemDetail>
      <Date stdValue="STRT">1999-12-07T21:45:00</Date>
      <Date stdValue="END">T00:18:30</Date>
      <Ref stdValue="LOC1">LHR</Ref>
      <Ref stdValue="LOC2">SIN</Ref>
      <Ref stdValue="CARR">BA</Ref>
      <Ref stdValue="FLNO">576</Ref>
      <Ref stdValue="SRVC">C</Ref>
      <Ref stdValue="STOP">O</Ref>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>2</LineItemNum>
         <SubLineItemNum>2</SubLineItemNum>
         <PartNumDetail stdValue="VP">
            <PartNum>CX0716</PartNum>
         </PartNumDetail>
         <Quantity>
            <Qty>1.0</Qty>
            <UnitOfMeasure stdValue="EA"/>
         </Quantity>
      </BaseItemDetail>
      <Date stdValue="STRT">1999-12-11T17:35:00</Date>
      <Date stdValue="END">T00:21:15</Date>
      <Ref stdValue="LOC1">SIN</Ref>
      <Ref stdValue="LOC2">HKG</Ref>
      <Ref stdValue="CARR">SQ</Ref>
      <Ref stdValue="FLNO">837</Ref>
      <Ref stdValue="SRVC">C</Ref>
      <Ref stdValue="STOP">O</Ref>
   </InvoiceDetails>
   <InvoiceDetails>
      <BaseItemDetail>
         <LineItemNum>2</LineItemNum>
         <SubLineItemNum>3</SubLineItemNum>
         <PartNumDetail stdValue="VP">
            <PartNum>BA0026</PartNum>
         </PartNumDetail>
         <Quantity>
            <Qty>1.0</Qty>
            <UnitOfMeasure stdValue="EA"/>
         </Quantity>
      </BaseItemDetail>
      <Date stdValue="STRT">1999-12-14T23:15:00</Date>
      <Date stdValue="END">T00:04:55</Date>
      <Ref stdValue="LOC1">HKG</Ref>
      <Ref stdValue="LOC2">LHR</Ref>
      <Ref stdValue="CARR">BA</Ref>
      <Ref stdValue="FLNO">527</Ref>
      <Ref stdValue="SRVC">Y</Ref>
      <Ref stdValue="STOP">X</Ref>
   </InvoiceDetails>
   <InvoiceSummary>
      <TaxSummary>
         <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>0</TaxPercent>
            <TaxableAmount>2577.0</TaxableAmount>
            <TaxAmount>0</TaxAmount>
         </Tax>
      </TaxSummary>
      <InvoiceTotals>
         <NetValue>2577.0</NetValue>
         <GrossValue>2577.0</GrossValue>
      </InvoiceTotals>
      <ActualPayment>
         <PaymentAmount>
            <LocalCurrencyAmt>2577.0</LocalCurrencyAmt>
         </PaymentAmount>
         <PaymentMean stdValue="ZZZ"/>
         <PaymentDate>1999-12-03</PaymentDate>
         <CardInfo>
            <CardNum>4864270000012345</CardNum>
            <CardExpirationDate/>
            <CardType stdValue="VS"/>
         </CardInfo>
      </ActualPayment>
   </InvoiceSummary>
</Invoice>
```

The Visa XML Invoice DTD A

This appendix contains the full text of the Document Type Definition (DTD) version 1.0. This is the version that supports the examples given in Chapter 4, and is related to the XML Stylesheet shown in Appendix E. These appendixes are provided to assist in interpreting the examples given in this guide.

The DTD and Stylesheet provided in the XML Invoice Technical Pack are the latest iteration and may be a different version than those shown here. The examples provided in the Technical Pack are dependent on the DTD and Stylesheet that accompany them.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Invoice (InvoiceHeader, InvoiceDetails+, InvoiceSummary)>
<!ATTLIST Invoice sectorUsageVersion CDATA #REQUIRED>
<!ELEMENT InvoiceHeader (InvoiceType, InvoiceStatus, TaxTreatment, DiscountTreatment?,
                           InvoiceTreatment, InvoiceNumber, InvoiceDate, TaxPointDate?,
Currency,
                            Party, Party, Party*, Payment?, PONum?,
                            DeliveryNoteNum?, Ref*, Date*, GenText*)>
<!ELEMENT InvoiceType EMPTY>
<!ATTLIST InvoiceType stdValue        (380|381) "380"
stdName         (UNTDID:1001) "UNTDID:1001">
<!-- 380 = Invoice 381 = Credit Note  -->
<!ELEMENT InvoiceStatus EMPTY>
<!ATTLIST InvoiceStatus stdValue        (9|10|53) "9"
stdName         (UNTDID:1225) "UNTDID:1225">
<!-- 9 = Original, 10 = Copy, 53 = Test -->
<!ELEMENT TaxTreatment EMPTY>
<!ATTLIST TaxTreatment stdValue        (NIL|GIL|NLL|GLL|NON) "NLL"
stdName         (VISA:TAXT) "VISA:TAXT">
<!-- NIL = Line item net amounts, invoice level tax GIL = Line item gross amounts, invoice
level tax NLL = Line item net amounts, line level tax GLL = Line item gross amounts, line
level tax NON = Tax does not apply to this invoice -->
<!ELEMENT DiscountTreatment EMPTY>
<!ATTLIST DiscountTreatment stdValue        (UN|UG|TN) "UG"
stdName         (VISA:DSCT) "VISA:DSCT">
```

```
<!-- UN = Line item unit price, net of discount UG = Line item unit price, gross of discount
TN = Line item sub-total, net of discount TG = Line item sub-total, gross of discount. -->
<!ELEMENT InvoiceTreatment EMPTY>
<!ATTLIST InvoiceTreatment stdValue        (P|EP|E) "P"
stdName         (VISA:INVT) "VISA:INVT">
<!-- P = Invoice printed and given to purchaser, and then used for tax reclaim S = Printed,
but printed invoice treated as supplemental invoice since electronic copy used for tax re-
claim E = Printed invoice suppressed since electronic master version used for tax reclaim -->
<!ELEMENT InvoiceNumber (#PCDATA)>
<!-- String, 1..35 characters -->
<!ELEMENT InvoiceDate (#PCDATA)>
<!-- String, 1..19 Character DateTime (CCYY-MM-DDTHH:MM:SS) -->
<!ELEMENT TaxPointDate (#PCDATA)>
<!-- String, 1..19 Character DateTime (CCYY-MM-DDTHH:MM:SS) -->
<!ELEMENT Currency EMPTY>
<!ATTLIST Currency stdValue         CDATA "USD"
stdName         (ISO:4217)  "ISO:4217">
<!-- ISO 4217 Code -->
<!-- ******** Party definition   *****-->
<!ELEMENT Party (PartyID?, Name?, Street?, PostalInfo?, Contact*, Ref*)>
<!ATTLIST Party stdValue         CDATA #REQUIRED
stdName         CDATA "UNTDID:3035">
<!-- BY = Buyer (Corporate)  SU = Supplier (Merchant) IV = Invoicee (Invoiced Party) PE =
Payee (receives payment) PI = Merchant Details as known by VISA DP = Delivery party (receives
the goods) SF = Ship from party -->
<!ELEMENT PartyID (#PCDATA)>
<!-- String 1..80 Character -->
<!ELEMENT Name (Name1, Name2?, Name3?)>
<!ELEMENT Name1 (#PCDATA)>
<!-- String 1..60 Character -->
<!ELEMENT Name2 (#PCDATA)>
<!-- String 1..60 Character -->
<!ELEMENT Name3 (#PCDATA)>
<!-- String 1..60 Character -->
<!ELEMENT Street (Street1, Street2?, Street3?, Street4?)>
<!ELEMENT Street1 (#PCDATA)>
<!-- String 1..55 Character -->
<!ELEMENT Street2 (#PCDATA)>
<!-- String 1..55 Character -->
<!ELEMENT Street3 (#PCDATA)>
<!-- String 1..55 Character -->
<!ELEMENT Street4 (#PCDATA)>
<!-- String 1..55 Character -->
<!ELEMENT PostalInfo (City?, CountrySubEntity?, PostalCode?, Country?)>
<!ELEMENT City (#PCDATA)>
<!-- String 1..35 Character -->
<!ELEMENT CountrySubEntity (#PCDATA)>
<!-- String 1..35 Character e.g State or County -->
<!ELEMENT PostalCode (#PCDATA)>
<!-- String 1..35 Character -->
<!ELEMENT Country (#PCDATA)>
<!-- String 1..35 Character Full country name -->
<!ELEMENT Contact (Name1?, TelNum?, EMail?, Function?)>
<!ELEMENT TelNum (#PCDATA)>
<!-- String 1..35 Character Telephone number -->
<!ELEMENT EMail (#PCDATA)>
```

```
<!-- String 1..35 Character Email Address -->
<!ELEMENT Function (#PCDATA)>
<!-- String 1..35 Character Function Description eg Job Title -->
<!-- ******** End of Party definition   *****-->
<!-- ******** Payment definition   *****-->
<!ELEMENT Payment (PaymentDueDate?, PaymentTerms*, PaymentMean?, CardInfo?)>
<!ELEMENT PaymentDueDate (AbsoluteDate|RelativeDate)>
<!ELEMENT AbsoluteDate (#PCDATA)>
<!-- String, 1..19 Character DateTime (CCYY-MM-DDTHH:MM:SS) -->
<!ELEMENT RelativeDate (RefDate, TimeRelation, TypeOfPeriod, NumberOfPeriods)>
<!ELEMENT RefDate EMPTY>
<!ATTLIST RefDate stdValue        CDATA "5"
stdName         CDATA "UNTDID:2475">
<!-- 5 = Date of invoice 9 = Date invoice received 21 = Goods received by buyer 26 = Date
of arrival of transport 81 = Date of shipment (as evidenced by transport documentation) 82
= Date payment due -->
<!ELEMENT TimeRelation EMPTY>
<!-- String, 1..19 Character DateTime (CCYY-MM-DDTHH:MM:SS) -->
<!ATTLIST TimeRelation stdValue        CDATA  "3"
stdName         CDATA "UNTDID:2009">
<!-- 1 =Reference date2 = Before reference date 3 = After reference date -->
<!ELEMENT TypeOfPeriod EMPTY>
<!ATTLIST TypeOfPeriod stdValue        CDATA  "CD"
stdName         CDATA "UNTDID:2151">
<!-- CD = Calendar day DW = Work day M = Month W = Week Y = Year -->
<!ELEMENT NumberOfPeriods (#PCDATA)>
<!--  Integer 1..3 Characters -->
<!ELEMENT PaymentTerms (PaymentTermType, (AbsoluteDate|RelativeDate), DiscountPercent)>
<!ELEMENT PaymentTermType (#PCDATA)>
<!-- String, 1..50 Character -->
<!ATTLIST PaymentTermType stdValue        CDATA  "22"
stdName         CDATA "UNTDID:4279">
<!-- 1 = Basic 3 = Fixed date 8 = Basic discount 10 = Instant 22 = DiscountOTHER = Indi-
cates element content will hold textual payment term type -->
<!-- AbsoluteDate, RelativeDate and DiscountPercent have already been declared -->
<!ELEMENT PaymentMean (#PCDATA)>
<!-- String, 1..50 Character -->
<!--Can have Other here-->
<!ATTLIST PaymentMean stdValue        CDATA "ZZZ"
stdName         CDATA "UNTDID:4461">
<!-- 10 = In cash 20 = Cheque 30 = Credit transfer ZZZ = Credit / debit card OTHER = Indicates
element content will hold textual payment mean -->
<!-- ******** End of Payment definition   ********-->
<!-- ******** CardInfo Definition ******** -->
<!ELEMENT CardInfo (CardNum, CardAuthCode?, CardRefNum?, CardExpirationDate?, CardType?,
CardholderName?, Ref*)>
<!ELEMENT CardNum (#PCDATA)>
<!-- String, 1..35 Character -->
<!ELEMENT CardAuthCode (#PCDATA)>
<!-- String, 1..35 Character -->
<!ELEMENT CardExpirationDate (#PCDATA)>
<!-- String, 4 Character Format MMYY -->
<!ELEMENT CardType (#PCDATA)>
<!-- String, 1..70 Character -->
<!ATTLIST CardType stdValue        CDATA "VS"
stdName         CDATA "VISA:CARD">
```

```
<!-- VS = Visa AMEX =American Express MC =Mastercard DINERS= Diners JCB = JCB DSCVR = Discover
OTHER = Indicates element content will hold textual card type -->
<!ELEMENT CardRefNum (#PCDATA)>
<!-- String, 1..35 Character -->
<!ELEMENT CardholderName (#PCDATA)> <!-- String, 1..35 Character -->
<!--  *** End of CardInfo Definition ****-->

<!ELEMENT PONum (#PCDATA)>
<!-- String, 1..35 Character Purchase Order Number -->
<!ELEMENT DeliveryNoteNum (#PCDATA)>
<!-- String, 1..35 Character Delivary Note Number -->
<!ELEMENT Ref (#PCDATA)>
<!-- String, 1..80 Character -->
<!ATTLIST Ref stdValue        CDATA "VA"
stdName         CDATA "UNTDID:1153">
<!-- VA =Tax registration number XA = Company/place registration number AWE = Cost centre
IV = Invoice number (used on a credit note to refer to the original invoice number) ACD =
Additional reference number – used to hold the Acquirer reference number ACW = Reference
number to previous message – holds Last Message ID ADQ = Unique market reference  Within Ref
at header level this is used to indicate Sector Type.  Within the Party "PI" element's Ref
element, this indicates the Merchant Category Code. -->
<!ELEMENT Date (#PCDATA)>
<!-- String, 1..19 Character DateTime (CCYY-MM-DDTHH:MM:SS) -->
<!ATTLIST Date stdValue        CDATA #REQUIRED
stdName         CDATA "UNTDID:2005">
<!-- See UNTDID:2005 for codes -->
<!ELEMENT GenText (#PCDATA)>
<!-- String, 1..80 Character -->
<!ATTLIST GenText stdValue        CDATA #REQUIRED
stdName         CDATA "UNTDID:4451">
<!-- AHR = Shareholder information -->
<!-- ******** InvoiceDetails definition   *****-->
<!ELEMENT InvoiceDetails (BaseItemDetail, UnitPrice?, POLineNum?,
                         LineItemSubtotal?, Tax*, LineDiscountInfo?, Date*, SpecialCond?,
Ref*, GenText*)>
<!--  ******* BaseItemDetail definition  ******* -->
<!ELEMENT BaseItemDetail (LineItemNum, SubLineItemNum?, PartNumDetail+, Quantity)>
<!ELEMENT LineItemNum (#PCDATA)><!--  Integer 1..10 Characters -->
<!ELEMENT SubLineItemNum (#PCDATA)> <!--  Integer 1..10 Characters -->
<!-- The following content model allows either PartNum or PartDesc or both -->
<!ELEMENT PartNumDetail ((PartNum,PartDesc?)|PartDesc)>
<!ATTLIST PartNumDetail stdValue        CDATA  "VP"
stdName         CDATA "UNTDID:7143">
<!-- BP = Buyer's Part No VP = Vendor's Part No CC = Industry commodity code -->
<!ELEMENT PartNum (#PCDATA)>
<!-- String, 1..35 Character -->
<!ELEMENT PartDesc (#PCDATA)>
<!-- String, 1..80 Character -->
<!ELEMENT Quantity (Qty, UnitOfMeasure?)>
<!ELEMENT Qty (#PCDATA)>
<!-- Decimal, 1..20 Characters format 15.4 -->
<!ELEMENT UnitOfMeasure EMPTY>
<!ATTLIST UnitOfMeasure stdValue        CDATA "EA"
stdName         CDATA "UNTDID:6411">
<!-- CMT = Centimetre DAY = Day EA = Each GRM = Gram HUR = Hour KGM = Kilogram KTM = Kilometre
LTR = Litre MIN = Minute MTR = Metre SEC = Second -->
```

```
<!-- ******* End of BaseItemDetail definition ******* -->
<!ELEMENT UnitPrice (#PCDATA)>
<!-- Decimal, 1..22 Characters format 18.3 -->
<!ELEMENT POLineNum (#PCDATA)>
<!-- Integer, 1..10 Characters -->
<!ELEMENT LineItemSubtotal (#PCDATA)>
<!-- Decimal, 1..22 Characters format 18.3 -->
<!-- ***** Tax definition  ***** -->
<!ELEMENT Tax (TaxFunction, TaxType, TaxCategory, TaxPercent,
                     TaxableAmount?, TaxAmount?, Location?)>
<!ELEMENT TaxFunction EMPTY>
<!ATTLIST TaxFunction stdValue        CDATA  "7"
stdName         (UNTDID:5283) "UNTDID:5283">
<!-- 5 =Customs duty 7 = Tax -->
<!ELEMENT TaxType EMPTY>
<!ATTLIST TaxType stdValue        CDATA  "VAT"
stdName         (UNTDID:5153) "UNTDID:5153">
<!-- VAT =Value Added Tax GST = Goods and Services Tax -->
<!ELEMENT TaxCategory EMPTY>
<!ATTLIST TaxCategory stdValue        CDATA #REQUIRED
stdName         CDATA "UNTDID:5305">
<!-- A =Mixed E = Exempt G = Free export item S = Standard Z = Zero -->
<!ELEMENT TaxPercent (#PCDATA)>
<!-- Decimal, 1..8 Characters format 3.4 -->
<!ELEMENT TaxableAmount (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT TaxAmount (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT Location (#PCDATA)>
<!-- String, 1..35 Character -->
<!-- ***** End of Tax definition  ***** -->
<!ELEMENT LineDiscountInfo ((DiscountValue|DiscountPercent),UnitPricePreDiscount?)>
<!ELEMENT DiscountPercent (#PCDATA)>
<!-- Decimal, 1..8 Characters format 3.4 -->
<!ELEMENT DiscountValue (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT UnitPricePreDiscount (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT SpecialCond (#PCDATA)>
<!-- String, 1..50 Character -->
<!ATTLIST SpecialCond stdValue        CDATA  "OTHER"
stdName         CDATA "UNTDID:4183">                                  <!-- 6 = Subject to
bonus 7 = Subject to commission 11 = Price includes excise 12 = Price includes tax 18 = Item
subject to national export restrictions 97 = Promotinal price 94 = Service 103 = Loan 104 =
Rental 105 = Processing 106 = Exchange 140 = Return of goods OTHER =Indicates element content
will hold textual special condition -->
<!--   ***************** InvoiceSummary Definition  ***********-->
<!ELEMENT InvoiceSummary (TaxSummary*, InvoiceTotals, ActualPayment*)>
<!ELEMENT TaxSummary (DiscountSummary?, Tax)>
<!--  **** DiscountSummary Definition *** -->
<!ELEMENT DiscountSummary (LineItemTotals, QtyDiscount?, ValueDiscount?,
                               SubTotalAfterQtyValueDiscount, SettlementDiscountAmt?,
                               SubTotalAfterSettDiscount?)>
<!ELEMENT LineItemTotals (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT QtyDiscount (#PCDATA)>
```

```
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT ValueDiscount (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT SubTotalAfterQtyValueDiscount (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT SettlementDiscountAmt (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT SubTotalAfterSettDiscount (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!-- *** InvoiceTotals definition **** -->
<!ELEMENT InvoiceTotals (DiscountSummary?, NetValue, TaxValue?, GrossValue )>
<!ELEMENT NetValue (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT TaxValue (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT GrossValue (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!-- *** ActualPayment Definition **** -->
<!ELEMENT ActualPayment (PaymentAmount, PaymentMean,
                                        PaymentDate, CardInfo?, Ref*)>
<!ELEMENT PaymentAmount (LocalCurrencyAmt, ForeignCurrencyPayment?)>
<!ELEMENT LocalCurrencyAmt (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT ForeignCurrencyPayment (ForeignCurrencyAmt, Currency)>
<!ELEMENT ForeignCurrencyAmt (#PCDATA)>
<!-- Decimal, 22 Characters format 18.3 -->
<!ELEMENT PaymentDate (#PCDATA)>
<!-- String, 1..19 Character DateTime (CCYY-MM-DDTHH:MM:SS) -->
```

Frequently Asked Questions B

Question 1: Why a DTD with documentation rather than an XML schema?

As the formal definitions of the full suite of XML component parts are yet to be finalized, any use of them at this time will result in changes. It is therefore more appropriate to document the characteristics of element sizes and formats with conversion to the use of an XML schema with links and pointers as appropriate, when these recommendations are stable. It is anticipated that this migration to using a schema will occur during the summer of 2000.

Question 2: Why elements versus attributes?

Attributes are used wherever a coded value is involved. Elements are used everywhere else.

Question 3: Why are default values used?

One objective is to enable a very simple XML invoice to be achieved with minimal data having to be provided.

However, as use of this invoice definition is required across global market sectors, tax regimes and fiscal regulatory bodies, there are situations where more sophisticated information is required to be provided. Thus, provision has been made to support these capabilities within the elements defined. In many cases the selected default values can be used for certain code lists and controlling bodies. If these are inappropriate, then they can be overridden where necessary.

Question 4: Why code values rather than text strings?

For the majority of ISO or UNTDID agreed code lists there are two, three or four numeric or character codes that are coded forms of longer textual strings or phrases. Selection of a shortened and relevant textual string accompanying a particular code value may not always be selected in the same way by different implementers as there is currently no guidance on such matters. As XML is case sensitive, the capitalizing of letters within such a phrase may also be handled differently by different people.

For example, in CBL2.0 the NameAddress Element uses “CarrierInfo” as the textual code for a particular type of Party and which is related back to “CA” in the relevant UNTDID code list. The corresponding UNTDID entry for “CA” is merely “Carrier”. Thus it would have been equally as valid to use “Carrier” rather than “CarrierInfo”. However, in the same CBL list of possible Parties, “Buyer” is used rather than “BuyerInfo”. If there is a need in the future for an extension to this list to provide a Party related to say the UNTDID code value “MT”, for “Party designated to execute sanitary procedures”, or “BT”, for “Party to be billed for other than freight (bill to)”, then the scope for mis-coding such textual strings as shortened coded forms could be considerable. Multiple language variants of textual strings would also be employed unless American spellings of English words were somehow enforced.

Consequently, there will be less scope for confusion if the internationally agreed code value is used throughout. The addition of an accompanying XSL stylesheet will then enable an invoice document to be displayed with these codes being translated into suitable phrases when presentation in human readable form is required.

Question 5: Why are the code lists limited and not extensible?

Where code list values are provided they are the instances that will normally be expected to be used. However, in some cases extra meanings may be required. These should always be obtained from the latest version of the full UNTDID list of codes for the relevant code list. The stdName identifies the owner and number of the code list involved.

Only in exceptional circumstances should a new user-defined, non-internationally agreed code list be constructed to work around the lack of a suitable code value being already available. If this is essential then all possible recipients of such XML documents must be able to identify how to interpret such a code. In the short term the use of XLinks and Xpointers are not fully stable. However, it is expected that during the calendar year 2000 these will provide robust mechanisms for easier location of and interpretation of such additional code values.

Question 6: How are Unicode (double byte) character sets supported?

The XML encoding mechanism is identified explicitly in the first line of each invoice document. Support for the ISO-Latin-1 character set uses UTF-8 encoding. Other encoding schemes such as UTF-16 can support the full range of Oriental and more complex character representations. The originator of the XML invoice document defines the encoding scheme employed in the document. The recipient may need to be able to handle completely different character sets for consecutive messages received.

Question 7: How are accented character sets (such as French, German, and Scandinavian) supported?

Originators of invoice documents must recognize that in most cases, the recipient's system(s) will only support one specific character set. Where an ASCII character represents different characters depending on the originating countries character and keyboard map then care must be exercised. There are standard “one accented character” to “two nonaccented character” translations that are commonly used when such national characters are used internationally. These should be performed prior to sending the message to avoid sending accented characters wherever possible.

Question 8: Why are DiscountTreatment, TaxTreatment and InvoiceTreatment used?

Buyers will sometimes obtain goods or services from suppliers that they have not previously bought from, and may never purchase from again. This may be from a physical supplier or via the Internet. An invoice will still be raised and provided.

For this “Open Trading” there will not be any previously established contract or definition of how to interpret the data fields within the electronically provided invoice message. Thus, it is necessary for the supplier to define, as part of the invoice message, how any discounts and tax calculations have been made. It is also important to identify for tax accounting and reporting purposes whether there was a paper invoice provided as the “master” document, or whether the electronic invoice is the master tax document.

For any one supplier this information will be the same, or “static”, for each of the invoices they generate. They will normally be determined by the type of business application and how it performs the calculations. As long as the calculations are performed consistently there are many different methods that the legal and fiscal regulatory bodies will accept and valid.

These processes occur today but, because the invoice is paper based intelligently interpreted by a human being, there is little difficulty. However, when loading this information electronically into the recipient business system, there needs to be clearer definitions of how such calculations have been made so that validations can be made without the introduction of acceptable errors.

Code List Reference C

This appendix contains a variety of codes that are necessary in the day-to-day use of XML invoices. Additional code listings will be incorporated in later editions of this publication.

Code Categories

Table C–1 describes the categorical meanings of various codes that may appear in XML invoice messages, and indicates the elements with which such codes would usually be associated.

Table C–1: Code Categories and Element Usage (1 of 2)

Code	Description	Element Usage
ISO:4217	Code that specifies a currency	Currency
UNTDID:1001	Code that specifies the document name	InvoiceType
UNTDID:1153	Code that gives the specific meaning to a reference	Ref
UNTDID:1225	Code that indicates the function of the document	InvoiceStatus
UNTDID:2005	Code that gives a specific meaning to a date	Date
UNTDID:3035	Code that gives the specific meaning to a party	Party

Table C–1: Code Categories and Element Usage (2 of 2)

Code	Description	Element Usage
UNTDID:4183	Code that indicates a specific condition	SpecialCond
UNTDID:4451	Code that indicates the function of the general text	GenText
UNTDID:4461	Code that indicates the payment method	PaymentMean
UNTDID:5153	Code that identifies the tax type	TaxType
UNTDID:5283	Code that identifies the function of the tax information	TaxFunction
UNTDID:5305	Code that identifies the tax category	TaxCategory
UNECE:20	Code that indicates the unit of measure in the quantity is expressed	UnitOfMeasure
UNTDID:7143	Code that identifies the type of part number	PartNumDetail
VISA:CARD	Code that denotes the card type	CardType
VISA:DSCT	Code that indicates the line-level discount treatment type	DiscountTreatment
VISA:INVT	Code that defines the manner in which the invoice is treated	InvoiceTreatment
VISA:TAXT	Code that defines the document’s tax treatment	TaxTreatment

Other Code Lists

Table C–2 lists sector type codes that define the market sector in which a transaction has taken place. This code is used in the InvoiceHeader/Ref element, where this Ref element is qualified with the stdValue attribute value “ADQ”.

Table C–2: Sector Type Codes

Code	Description
AI	Passenger Itinerary (Airline, Rail, Ferry and Cruise Liners)
CR	Car rental agencies
HC[1]	Healthcare (business-to-healthcare provider products)
IG[1]	Government (intra- and inter-governmental agencies)
LG	Lodging
TS[1]	Temporary services (temporary manpower services)
MM[1]	Miscellaneous merchants (all other merchants not otherwise classified as a separate merchant sector)

[1] Invoice elements specific to these market sectors have not yet been developed for the XML invoice.

Table C–3 shows codes that have been defined for additional detail.

Table C–3: PaymentInfo and CardInfo Ref Codes[1]

Code	Description
PLOC	Payment level info – Identification of where the purchase took place
TTYP	Payment level info – Transaction type
CHPV	Cardinfo level info – Chip verification results for chip card
STID	Cardinfo level info – Indication of SET transaction

Note: *When using the above Ref codes the stdName attribute should have the value VISA:REF*

Individual markets may have adopted codes that are not detailed in this document. Developers may want to treat other undefined charges or details within the LineItemDetail element.

Tax Treatments D

This appendix provides details and examples for each Tax Treatment type. The various tables show the values that need to be in the monetary elements that are affected by the invoice tax treatment and invoice-level discount values, according to various scenarios.

The first example in each Tax Treatment type is the simple implementation, that is there are no multiple tax codes per line, and no invoice discounts. Tax treatment types and discount scenarios that are unlikely to be implemented are not documented. For example, invoice-level discounts are only documented where tax is calculated at invoice level (NIL and GIL), or there is no tax (NON), and split-total multicategory tax is only documented where tax is calculated at line level (NLL and GLL).

No Tax (NON)

There are two different ways to treat no tax invoices:

- NON with no invoice-level discounts
- NON with invoice-level discounts

NON, With No Invoice-Level Discounts

The following table and sample XML file demonstrate a simple example of an invoice with a TaxTreatment of NON, and with no discounts.

Element	Value
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax – this element must not be present	
InvoiceSummary/TaxSummary/Tax – this element must not be present	
InvoiceSummary/InvoiceTotals	
NetValue	Sum of the InvoiceDetails/LineItemSubtotal elements
TaxValue	Absent
GrossValue	Sum of the InvoiceDetails/LineItemSubtotal elements (i.e. same value as NetValue)

The following example is of an invoice with Tax Treatment NON, with no discounts.

There are three line items, providing three instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 300.00.

Line 2 has a LineItemSubtotal value of 200.00.

Line 3 has a LineItemSubtotal value of 50.00

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Rate
1	Room Charge	3	EA	100.00	300.00	
2	Bar Meal	1	EA	200.00	200.00	
3	Bar Meal	1	EA	50.00	50.00	

EXAMPLE

```
<InvoiceHeader>
:   :   :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>300.00</LineItemSubtotal>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
            <PartDesc>Bar Meal</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>200.00</UnitPrice>
    <LineItemSubtotal>200.00</LineItemSubtotal>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
  <InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>3</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
            <PartDesc>Bar Meal</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>50.00</UnitPrice>
    <LineItemSubtotal>50.00</LineItemSubtotal>
</InvoiceDetails>
<InvoiceSummary>
    <InvoiceTotals>
```

```
            <NetValue>550.00</NetValue>
            <GrossValue>550.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>550.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>1999-02-11</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
```

NON, With Invoice-Level Discounts

The following table and example XML file demonstrate a simple example of an invoice with a TaxTreatment of NON, and with invoice-level discounts.

Element	**Value**
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax – this element must not be present	
InvoiceSummary/TaxSummary/DiscountSummary – this element must not be present	
InvoiceSummary/TaxSummary/Tax – this element must not be present	
InvoiceSummary/InvoiceTotals/DiscountSummary	
LineItemTotals	Sum of InvoiceDetails/LineItemSubtotal elements
QtyDiscount	Invoice quantity discount amount.
ValueDiscount	Invoice value discount amount.
SubTotalAfterQtyValueDisc	LineItemTotals – QtyDiscount – ValueDiscount
InvoiceSummary/InvoiceTotals	
NetValue	InvoiceSummary/InvoiceTotals/DiscountSummary/ SubTotalAfterQtyValueDisc
TaxValue	Absent
GrossValue	InvoiceSummary/InvoiceTotals/DiscountSummary/ SubTotalAfterQtyValueDisc (i.e. same value as NetValue

The following example is of an invoice with Tax Treatment NON, with a value discount of 25.00, and a quantity discount of 10.00

There are three line items, providing three instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 300.00.

Line 2 has a LineItemSubtotal value of 200.00.

Line 3 has a LineItemSubtotal value of 50.00.

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Rate
1	Room Charge	3	EA	100.00	300.00	
2	Bar Meal	1	EA	200.00	200.00	
3	Bar Meal	1	EA	50.00	50.00	

EXAMPLE

```
</InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>300.00</LineItemSubtotal>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
            <PartDesc>Bar Meal</PartDesc>
        </PartNumDetail>
        <Quantity>
```

```
            <Qty>1</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>200.00</UnitPrice>
    <LineItemSubtotal>200.00</LineItemSubtotal>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
  <InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>3</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
            <PartDesc>Bar Meal</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>50.00</UnitPrice>
    <LineItemSubtotal>50.00</LineItemSubtotal>
</InvoiceDetails>
<InvoiceSummary>
    <InvoiceTotals>
        <DiscountSummary>
            <LineItemTotals>550.00</LineItemTotals>
            <QtyDiscount>10.00</QtyDiscount>
            <ValueDiscount>25.00</ValueDiscount>
            <SubTotalAfterQtyValueDiscount>515.00</SubTotalAfterQtyValueDiscount>
        </DiscountSummary>
        <NetValue>515.00</NetValue>
        <GrossValue>515.00</GrossValue>
    </InvoiceTotals>
    <ActualPayment>
        <PaymentAmount>
            <LocalCurrencyAmt>515.00</LocalCurrencyAmt>
        </PaymentAmount>
        <PaymentMean/>
        <PaymentDate>1999-02-11</PaymentDate>
        <CardInfo>
            <CardNum>4917876543212345</CardNum>
            <CardExpirationDate>1199</CardExpirationDate>
            <CardType/>
        </CardInfo>
    </ActualPayment>
</InvoiceSummary>
```

Net Price, Tax Calculated at Invoice Level (NIL)

There are three ways to treat net price invoices where tax is calculated at invoice level:

- NIL with no discounts, no multicategory tax codes
- NIL with no discounts, and multicategory tax codes

- NIL with an invoice level discount

NIL With No Discounts, No Multicategory Tax Codes

The following tables and example XML file demonstrate a simple example of an invoice with a TaxTreatment of NIL with no discounts, and no multicategory tax codes.

Element	**Value**
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	Absent
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Sum of the InvoiceDetails/LineItemSubtotal amounts for the current tax code, i.e. the total net line amount for the tax code.
TaxAmount	TaxableAmount x TaxPercent%
InvoiceSummary/InvoiceTotals	
NetValue	Sum of all InvoiceDetails/LineItemSubtotal elements
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of NetValue and TaxValue

The following example is of an invoice with Tax Treatment NIL, with no discounts and no multi-tax-category line items.

There are two line items, providing two instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 300.00, with TaxCategory A.

Line 2 has a LineItemSubtotal value of 200.00, with TaxCategory S.

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Cat.	Tax Rate
1	Room Charge	3	EA	100.00	300.00	A	5
2	Bar Meal	1	EA	200.00	200.00	S	10

This table demonstrates how the InvoiceSummary values are calculated:

InvoiceSummary/TaxSummary/Tax

	TaxCode A	***TaxCode S***
TaxCategory	A	S
TaxPercent	5	10
TaxableAmount	300.00	200
TaxAmount	300 x 5% = 15	200 x 10% = 20

InvoiceSummary/InvoiceTotals

NetValue	300 + 200 = 500
TaxValue	15 + 20 =35
GrossValue	500 + 35 = 535

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
     <BaseItemDetail>
          <LineItemNum>1</LineItemNum>
          <PartNumDetail>
               <PartNum>100</PartNum>
               <PartDesc>Room Charge</PartDesc>
          </PartNumDetail>
          <PartNumDetail stdValue="CC">
               <PartNum>H100</PartNum>
          </PartNumDetail>
          <Quantity>
               <Qty>3</Qty>
               <UnitOfMeasure/>
          </Quantity>
     </BaseItemDetail>
     <UnitPrice>100.00</UnitPrice>
     <LineItemSubtotal>300.00</LineItemSubtotal>
     <Tax>
          <TaxFunction/>
          <TaxType/>
```

```
            <TaxCategory stdValue="A"/>
            <TaxPercent>5.00</TaxPercent>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>2</LineItemNum>
            <PartNumDetail>
                <PartNum>78</PartNum>
                <PartDesc>Bar Meal</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>200.00</UnitPrice>
        <LineItemSubtotal>200.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>10.00</TaxPercent>
                <TaxableAmount>200</TaxableAmount>
                <TaxAmount>20.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="A"/>
                <TaxPercent>5.00</TaxPercent>
                <TaxableAmount>300</TaxableAmount>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>500.00</NetValue>
            <TaxValue>35.00</TaxValue>
            <GrossValue>535.00</GrossValue>
        </InvoiceTotals>
    :   :   :
    </InvoiceSummary>
```

NIL With No Discounts, and Multicategory Tax Codes

The following tables and example XML file demonstrate an example of an invoice with a TaxTreatment of NIL, and with no discounts, but with multicategory tax codes.

Element	Value
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	Absent
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Sum of the InvoiceDetails/LineItemSubtotal amounts for the current tax code, i.e. the total net line amount for the tax code.
TaxAmount	TaxableAmount x TaxPercent%
InvoiceSummary/InvoiceTotals	
NetValue	Sum of all InvoiceDetails/LineItemSubtotal elements
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of NetValue and TaxValue

The following example is of an invoice with Tax Treatment NIL, with no discounts but with both multicategory tax codes on total amount tax, and split-total multicategory tax.

There are two line items, providing two instances of the InvoiceDetails element .

Line 1 has two multicategory tax codes on total amount tax – the LineItemSubtotal value is 300, and the tax categories are S and A.

Line 2 has a single tax category – the LineItemSubtotal value is 200, and the tax category is S.

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Cat.	Tax Rate
1	Room Charge	3	EA	100.00	300.00	S A	10 5
2	Bar Meal	1	EA	200	200	S	10

This table demonstrates how the InvoiceSummary values are calculated:

InvoiceSummary/TaxSummary/Tax		
	TaxCode S	***TaxCode A***
TaxCategory	S	A
TaxPercent	10	5
TaxableAmount	300 +200 = 500	300
TaxAmount	30 + 20 = 50	15

InvoiceSummary/InvoiceTotals	
NetValue	300 + 200 = 500
TaxValue	50 + 15 = 65
GrossValue	500 + 65 = 565

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>300.00</LineItemSubtotal>
    <Tax>
```

```
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="S"/>
          <TaxPercent>10.00</TaxPercent>
      </Tax>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="A"/>
          <TaxPercent>5.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
  </InvoiceDetails>
  <InvoiceDetails>
      <BaseItemDetail>
          <LineItemNum>2</LineItemNum>
          <PartNumDetail>
              <PartNum>78</PartNum>
              <PartDesc>Bar Meal</PartDesc>
          </PartNumDetail>
          <Quantity>
              <Qty>1</Qty>
              <UnitOfMeasure/>
          </Quantity>
      </BaseItemDetail>
      <UnitPrice>200.00</UnitPrice>
      <LineItemSubtotal>200.00</LineItemSubtotal>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="S"/>
          <TaxPercent>10.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
  </InvoiceDetails>
<InvoiceSummary>
      <TaxSummary>
          <Tax>
              <TaxFunction/>
              <TaxType/>
              <TaxCategory stdValue="S"/>
              <TaxPercent>10.00</TaxPercent>
              <TaxableAmount>500</TaxableAmount>
              <TaxAmount>50.00</TaxAmount>
          </Tax>
      </TaxSummary>
      <TaxSummary>
          <Tax>
              <TaxFunction/>
              <TaxType/>
              <TaxCategory stdValue="A"/>
              <TaxPercent>5.00</TaxPercent>
              <TaxableAmount>300</TaxableAmount>
              <TaxAmount>15.00</TaxAmount>
          </Tax>
      </TaxSummary>
```

```
        <InvoiceTotals>
            <NetValue>500.00</NetValue>
            <TaxValue>65.00</TaxValue>
            <GrossValue>565.00</GrossValue>
        </InvoiceTotals>
    :   :   :
    </InvoiceSummary>
```

NIL With an Invoice-Level Discount

The following tables and example XML file demonstrate an example of an invoice with invoice-level quantity discount. This also includes a multicategory tax code on total amount tax.

Element	**Value**
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	Absent
InvoiceSummary/TaxSummary/DiscountSummary	
LineItemTotals	Sum of the InvoiceDetails/LineItemSubtotal amounts for the current tax code, or the total net line amount for the tax code.
QtyDiscount	Proportion of invoice-level quantity discount that applies to this tax code. This can be calculated thus: LineItemTotals / (Sum of InvoiceDetails/LineItemSubtotal elements that discount applies to) x invoice quantity discount amount
ValueDiscount	Proportion of invoice-level value discount that applies to this tax code. This can be calculated thus: LineItemTotals / (Sum of InvoiceDetails/LineItemSubtotal elements that discount applies to) x invoice value discount amount
SubTotalAfterQtyValueDisc	LineItemTotals - QtyDiscount – ValueDiscount
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category

TaxPercent	Tax percentage rate
TaxableAmount	DiscountSummary/SubTotalAfterQtyValueDisc
TaxAmount	TaxableAmount x TaxPercent%
InvoiceSummary/InvoiceTotals/DiscountSummary	
LineItemTotals	Sum of InvoiceDetails/LineItemSubtotal elements
QtyDiscount	Invoice quantity discount amount.
ValueDiscount	Invoice value discount amount.
SubTotalAfterQtyValueDisc	LineItemTotals – QtyDiscount – ValueDiscount
InvoiceSummary/InvoiceTotals	
NetValue	**InvoiceSummary/InvoiceTotals/DiscountSummary/ SubTotalAfterQtyValueDisc**
TaxValue	**Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements**
GrossValue	**Sum of NetValue and TaxValue**

The following example demonstrates an invoice with Tax Treatment NIL, with an invoice-level quantity discount of 50.00, that applies to all invoice lines. Note that the first line item is multicategory tax codes in total amount tax.

There are two line items, providing two instances of the InvoiceDetails element.

Line 1 has two multicategory tax codes on total amount tax – the LineItemSubtotal value is 300, and the tax categories are S and A.

Line 2 has a single tax category – the LineItemSubtotal value is 200, and TaxCategory is S.

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Cat.	Tax Rate
1	Room Charge	3	EA	100.00	300.00	S A	10 5
2	Bar Meal	1	EA	200	200	S	10

This table demonstrates how the InvoiceSummary values are calculated:

InvoiceSummary/TaxSummary/DiscountSummary

	TaxCode S	**TaxCode A**
LineItemTotals	300.00 + 200.00 = 500.00	300
QtyDiscount	500.00 / 500.00 x 50.00 = 50.00	300.00 / 500.00 x 50.00 = 30.00
SubTotalAfterQtyValueDisc	500.00 – 50.00 = 450.00	300.00 – 30.00 = 270.00

InvoiceSummary/TaxSummary/Tax

	TaxCode S	**TaxCode A**
TaxCategory	S	A
TaxPercent	10	5
TaxableAmount	450	270
TaxAmount	450 x 10 / 100 = 45	270 x 5 / 100 = 13.50

InvoiceSummary/InvoiceTotals/DiscountSummary

LineItemTotals	300 + 200 = 500
QtyDiscount	50
SubTotalAfterQtyValueDisc	500 – 50 = 450

InvoiceSummary/InvoiceTotals

NetValue	450
TaxValue	45 + 13.50 = 58.50
GrossValue	450 + 58.50 = 508.50

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
```

```
      <UnitPrice>100.00</UnitPrice>
      <LineItemSubtotal>300.00</LineItemSubtotal>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="S"/>
          <TaxPercent>10.00</TaxPercent>
      </Tax>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="A"/>
          <TaxPercent>5.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
  </InvoiceDetails>
  <InvoiceDetails>
      <BaseItemDetail>
          <LineItemNum>2</LineItemNum>
          <PartNumDetail>
              <PartNum>78</PartNum>
              <PartDesc>Bar Meal</PartDesc>
          </PartNumDetail>
          <Quantity>
              <Qty>1</Qty>
              <UnitOfMeasure/>
          </Quantity>
      </BaseItemDetail>
      <UnitPrice>200.00</UnitPrice>
      <LineItemSubtotal>200.00</LineItemSubtotal>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="S"/>
          <TaxPercent>10.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
  </InvoiceDetails>
<InvoiceSummary>
      <TaxSummary>
          <DiscountSummary>
              <LineItemTotals>500</LineItemTotals>
              <QtyDiscount>50.00</QtyDiscount>
              <SubTotalAfterQtyValueDiscount>450.00</SubTotalAfterQtyValueDiscount>
          </DiscountSummary>
          <Tax>
              <TaxFunction/>
              <TaxType/>
              <TaxCategory stdValue="S"/>
              <TaxPercent>10.00</TaxPercent>
              <TaxableAmount>450.00</TaxableAmount>
              <TaxAmount>45.00</TaxAmount>
          </Tax>
      </TaxSummary>
      <TaxSummary>
          <DiscountSummary>
```

```
                <LineItemTotals>300</LineItemTotals>
                <QtyDiscount>30.00</QtyDiscount>
                <SubTotalAfterQtyValueDiscount>270.00</SubTotalAfterQtyValueDiscount>
            </DiscountSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="A"/>
                <TaxPercent>5.00</TaxPercent>
                <TaxableAmount>270</TaxableAmount>
                <TaxAmount>13.50</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <DiscountSummary>
                <LineItemTotals>500</LineItemTotals>
                <QtyDiscount>50.00</QtyDiscount>
                <SubTotalAfterQtyValueDiscount>450.00</SubTotalAfterQtyValueDiscount>
            </DiscountSummary>
            <NetValue>450.00</NetValue>
            <TaxValue>58.50</TaxValue>
            <GrossValue>508.50</GrossValue>
        </InvoiceTotals>
    </InvoiceSummary>
```

Gross Price, Tax Calculated at Invoice-Level (GIL)

There are three ways to handle invoices that show gross price with tax calculated at invoice level:

- GIL with no discounts, no multicategory tax codes
- GIL with no discounts, but multicategory tax codes on total amount tax
- GIL with invoice level discounts and multicategory tax codes on total amount tax

GIL With No Discounts, No Multicategory Tax Codes

The following tables and example XML file demonstrate a simple example of an invoice with a TaxTreatment of GIL, and with no discounts, and no multicategory tax codes.

Element	**Value**
InvoiceDetails	
UnitPrice	Gross of tax (i.e. includes tax)
LineItemSubtotal	Gross of tax (i.e. includes tax)

InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	Absent
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Sum of the taxable amounts for each line for the current tax code, i.e. the total gross line amount for the tax code, less the tax amount. The taxable amount of each line item can be calculated as: InvoiceDetails/LineItemSubtotal element / (1 + (InvoiceDetails/Tax/TaxPercent / 100))
TaxAmount	The total gross line amount for the tax code - TaxableAmount
InvoiceSummary/InvoiceTotals	
NetValue	Sum of all InvoiceDetails/LineItemSubtotal elements less the sum of the InvoiceSummary/TaxSummary/Tax/TaxAmount elements
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of NetValue and TaxValue

The following example is of an invoice with Tax Treatment GIL, with no discounts and no multi-tax-category line items.

There are two line items, providing two instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 315.00, with TaxCategory A.

Line 2 has a LineItemSubtotal value of 220.00, with TaxCategory S.

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Cat.	Tax Rate
1	Room Charge	3	EA	105.00	315.00	A	5
2	Bar Meal	1	EA	220	220	S	10

This table demonstrates how the InvoiceSummary values are calculated:

InvoiceSummary/TaxSummary/Tax

	TaxCode A	***TaxCode S***
TaxCategory	A	S
TaxPercent	5	10
TaxableAmount	315 / 1.05 = 300	220 / 1.10 = 200
TaxAmount	315 – 300 = 15	220 – 200 = 20

InvoiceSummary/InvoiceTotals

NetValue	(315 + 220) – (15 + 20) = 500
TaxValue	15 + 20 = 35
GrossValue	500 + 35 = 535

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>105.00</UnitPrice>
    <LineItemSubtotal>315.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="A"/>
        <TaxPercent>5.00</TaxPercent>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
```

```
            <PartNumDetail>
                <PartNum>78</PartNum>
                <PartDesc>Bar Meal</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>220.00</UnitPrice>
        <LineItemSubtotal>220.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>10.00</TaxPercent>
                <TaxableAmount>200</TaxableAmount>
                <TaxAmount>20.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="A"/>
                <TaxPercent>5.00</TaxPercent>
                <TaxableAmount>300</TaxableAmount>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>500.00</NetValue>
            <TaxValue>35.00</TaxValue>
            <GrossValue>535.00</GrossValue>
        </InvoiceTotals>
    :   :   :
    </InvoiceSummary>
```

GIL With No Discounts, but Multicategory Tax Codes

The following tables and example XML file demonstrate an example of an invoice with a TaxTreatment of GIL, with no discounts but with multicategory tax codes on the total amount tax.

Element	Value
InvoiceDetails	
UnitPrice	Gross of tax (i.e., includes tax)
LineItemSubtotal	Gross of tax (i.e., includes tax)
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	Absent
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Sum of the taxable amounts for each line for the current tax code, that is the total gross line amount for the tax code, less the tax amount. The taxable amount of each line item can be calculated as: InvoiceDetails/LineItemSubtotal element / (1 + (the sum of the InvoiceDetails/Tax/TaxPercent values associated with the line / 100))
TaxAmount	TaxableAmount x TaxPercent / 100
InvoiceSummary/InvoiceTotals	
NetValue	Sum of all InvoiceDetails/LineItemSubtotal elements less the sum of the InvoiceSummary/TaxSummary/Tax/TaxAmount elements
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of NetValue and TaxValue

The following example is of an invoice with Tax Treatment GIL, with no discounts, but with multicategory tax codes on total amount tax.

Three line items provide three instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 352.00, and the tax category is H.

Line 2 has a LineItemSubtotal value of 105.00, and the tax category is A.

Line 3 has a LineItemSubtotal value of 230. The tax categories are S and A.

Line No.	**Product**	**Qty.**	**UOM**	**Cost**	**Subtotal**	**Tax Cat.**	**Tax Rate**
1	Room Charge	3	EA	117.50	352.50	H	17.5
2	Telephone	2	EA	52.50	105.00	A	5
3	Bar Meal	1	EA	230.00	230.00	S A	10 5

This table demonstrates how the InvoiceSummary values have been calculated:

InvoiceSummary/TaxSummary/Tax

	TaxCode S	**TaxCode A**	**TaxCode H**
TaxCategory	S	A	H
TaxPercent	10	5	17.5
TaxableAmount	230 / 1.15 = 200	((105 / 1.05) + (230 / 1.15)) = 300 Note: See below for further explanation	352.50 / 1.175 = 300
TaxAmount	200 x 10 / 100 = 20	300 x 5 / 100 = 15	300 x 17.5 / 100 = 52.50

InvoiceSummary/InvoiceTotals

NetValue	(352.50 + 105 + 230) – (20 + 15 + 52.50) = 600
TaxValue	20 + 15 + 52.50 = 87.50
GrossValue	600 + 87.50 = 687.50

TaxableAmount for TaxCode A is expanded below:
The second line item is straightforward – there is only one tax code associated with the line so the first part of the equation is:
LineItemSubtotal / (1 + (TaxPercent / 100)),
which gives 105 / (1 + (5/100),
simplified to 105 / 1.05, which gives a result of 100

The third line item is more complex because there are two tax codes, and thus two tax rates, associated with it.
The equation is therefore:
LineItemSubTotal / (1 + ((first TaxPercent + second TaxPercent) / 100))
which gives 230 / (1 + ((10 + 5) / 100))
simplified to 230 / 1.15, which gives a result of 200

The final result is the addition of 100 and 200, to give 300

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>117.50</UnitPrice>
    <LineItemSubtotal>352.50</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="H"/>
        <TaxPercent>17.5</TaxPercent>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>98</PartNum>
            <PartDesc>Telephone</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>2</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>52.50</UnitPrice>
    <LineItemSubtotal>105.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="A"/>
        <TaxPercent>5.00</TaxPercent>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>3</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
```

```
              <PartDesc>Bar Meal</PartDesc>
          </PartNumDetail>
          <Quantity>
              <Qty>1</Qty>
              <UnitOfMeasure/>
          </Quantity>
      </BaseItemDetail>
      <UnitPrice>230.00</UnitPrice>
      <LineItemSubtotal>230.00</LineItemSubtotal>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="S"/>
          <TaxPercent>10.00</TaxPercent>
      </Tax>
      <Tax>
          <TaxFunction/>
          <TaxType/>
          <TaxCategory stdValue="A"/>
          <TaxPercent>5.00</TaxPercent>
      </Tax>
      <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
  </InvoiceDetails>
<InvoiceSummary>
      <TaxSummary>
          <Tax>
              <TaxFunction/>
              <TaxType/>
              <TaxCategory stdValue="S"/>
              <TaxPercent>10.00</TaxPercent>
              <TaxableAmount>200</TaxableAmount>
              <TaxAmount>20.00</TaxAmount>
          </Tax>
          <Tax>
              <TaxFunction/>
              <TaxType/>
              <TaxCategory stdValue="A"/>
              <TaxPercent>5.00</TaxPercent>
              <TaxableAmount>300</TaxableAmount>
              <TaxAmount>15.00</TaxAmount>
          </Tax>
          <Tax>
              <TaxFunction/>
              <TaxType/>
              <TaxCategory stdValue="H"/>
              <TaxPercent>17.5</TaxPercent>
              <TaxableAmount>300</TaxableAmount>
              <TaxAmount>52.50</TaxAmount>
          </Tax>
      </TaxSummary>
      <InvoiceTotals>
          <NetValue>600.00</NetValue>
          <TaxValue>87.50</TaxValue>
          <GrossValue>687.50</GrossValue>
      </InvoiceTotals>
      <ActualPayment>
```

```
            <PaymentAmount>
                <LocalCurrencyAmt>687.50</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>1999-02-11</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
</Invoice>
```

GIL With Invoice-Level Discounts, and Multicategory Tax Codes

Note that an invoice-level discount on an invoice with a Tax Treatment of GIL applies to the gross value of the invoice, not to the net value.

The following tables and example XML file demonstrate an example of an invoice with a TaxTreatment of GIL, with discounts and with multicategory tax codes on the total amount tax.

Element	**Value**
InvoiceDetails	
UnitPrice	Gross of tax, i.e. includes tax
LineItemSubtotal	Gross of tax, i.e. includes tax
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	Absent
InvoiceSummary/TaxSummary/DiscountSummary	

LineItemTotals	Sum of the line item totals (gross) to which this tax code applies.This can be calculated thus: For each line that this tax rate applies to, the proportion of the LineItemSubtotal for the tax rate is: (LineItemSubtotal / (1 + (the sum of the InvoiceDetails/Tax/TaxPercent values associated with the line / 100))) + (LineItemSubtotal / (1 + (the sum of the InvoiceDetails/Tax/TaxPercent values associated with the line / 100)) x TaxPercent) The LineItemTotals for the tax code is then the sum of these amounts.
QtyDiscount	Proportion of invoice-level quantity discount that applies to this tax code. This can be calculated thus: LineItemTotals / (Sum of InvoiceDetails/LineItemSubtotal elements that discount applies to) x invoice quantity discount amount
ValueDiscount	Proportion of invoice-level value discount that applies to this tax code. This can be calculated thus: LineItemTotals / (Sum of InvoiceDetails/LineItemSubtotal elements that discount applies to) x invoice value discount amount
SubTotalAfterQtyValueDisc	LineItemTotals - QtyDiscount – ValueDiscount
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Net value of SubTotalAfterQtyValueDisc, which can be calculated thus: SubTotalAfterQtyValueDisc / (1 + (TaxPercent/100))
TaxAmount	SubTotalAfterQtyValueDisc – TaxableAmount
InvoiceSummary/InvoiceTotals/DiscountSummary	
LineItemTotals	Sum of InvoiceDetails/LineItemSubtotal elements
QtyDiscount	Invoice quantity discount amount
ValueDiscount	Invoice value discount amount
SubTotalAfterQtyValueDisc	LineItemTotals – QtyDiscount – ValueDiscount
InvoiceSummary/InvoiceTotals	
NetValue	GrossValue – TaxValue
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	InvoiceSummary/InvoiceTotals/DiscountSummary/SubTotalAfterQtyValueDisc

The following example demonstrates an invoice with TaxTreatment GIL, with an invoice-level quantity discount of 30.00, that applies to all invoice lines and tax categories.

There are three line items, providing three instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 352.00, and the tax category is H.

Line 2 has a LineItemSubtotal value of 105.00, and the tax category is A.

Line 3 has a LineItemSubtotal value of 230.00, and the tax categories are S and A.

Line No.	Product	Qty.	UOM	Cost	Subtotal	Tax Cat.	Tax Rate
1	Room Charge	3	EA	117.50	352.50	H	17.5
2	Telephone	2	EA	52.50	105.00	A	5
3	Bar Meal	1	EA	230.00	230.00	S A	10 5

This table demonstrates how the InvoiceSummary values have been calculated:

InvoiceSummary/TaxSummary/DiscountSummary

	TaxCode H	*TaxCode A*	*TaxCode S*
LineItemTotals	352.50	105 + 210 = 315 nb See below for further explanation	220
QtyDiscount	352.50 / 687.50 x 30 = 15.38	315.00 / 687.50 x 30 = 13.75	220 / 687.50 x 30 = 9.60
SubTotalAfterQtyValueDisc	352.20 – 15.38 = 337.12	315 – 13.75 = 301.25	220 – 9.60 = 210.40

InvoiceSummary/TaxSummary/Tax

	TaxCode H	*TaxCode A*	*TaxCode S*
TaxCategory	H	A	S
TaxPercent	17.5	5	10
TaxableAmount	337.12 / 1.175 = 286.91	301.25 / 1.05 = 286.90	210.40 / 1.10 = 191.27

TaxAmount	337.12 – 286.91 = 50.21	301.25 – 286.90 = 14.35	210.40 – 191.27 = 19.13
InvoiceSummary/InvoiceTotals/DiscountSummary			
LineItemTotals	352.50 + 105 + 230 = 687.50		
QtyDiscount	30		
SubTotalAfterQtyValueDisc	687.50 – 30 = 657.50		
InvoiceSummary/InvoiceTotals			
NetValue	657.50 – 83.69 = 573.81		
TaxValue	50.21 + 14.35 + 19.13 = 83.69		
GrossValue	657.50		

LineItemTotals for TaxCode A is expanded below:

((105/1.05) + (105/1.05 x 5%)) + ((230/1.15) + (230/1.15 x 5%))

EXAMPLE

```
<Invoice>
    <InvoiceHeader>
    :   :   :
    </InvoiceHeader>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>1</LineItemNum>
            <PartNumDetail>
                <PartNum>100</PartNum>
                <PartDesc>Room Charge</PartDesc>
            </PartNumDetail>
            <PartNumDetail stdValue="CC">
                <PartNum>H100</PartNum>
            </PartNumDetail>
            <Quantity>
                <Qty>3</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>117.50</UnitPrice>
        <LineItemSubtotal>352.50</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="H"/>
            <TaxPercent>17.5</TaxPercent>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>2</LineItemNum>
            <PartNumDetail>
                <PartNum>98</PartNum>
                <PartDesc>Telephone</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>2</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>52.50</UnitPrice>
        <LineItemSubtotal>105.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="A"/>
            <TaxPercent>5.00</TaxPercent>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>3</LineItemNum>
            <PartNumDetail>
                <PartNum>78</PartNum>
                <PartDesc>Bar Meal</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>230.00</UnitPrice>
        <LineItemSubtotal>230.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
        </Tax>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="A"/>
            <TaxPercent>5.00</TaxPercent>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
            <DiscountSummary>
                <LineItemTotals>220.00</LineItemTotals>
                <QtyDiscount>9.60</QtyDiscount>
                <SubTotalAfterQtyValueDiscount>210.40</SubTotalAfterQtyValueDiscount>
            </DiscountSummary>
            <Tax>
                <TaxFunction/>
```

```
                  <TaxType/>
                  <TaxCategory stdValue="S"/>
                  <TaxPercent>10.00</TaxPercent>
                  <TaxableAmount>191.27</TaxableAmount>
                  <TaxAmount>19.13</TaxAmount>
             </Tax>
         </TaxSummary>
         <TaxSummary>
              <DiscountSummary>
                  <LineItemTotals>315.00</LineItemTotals>
                  <QtyDiscount>13.75</QtyDiscount>
                  <SubTotalAfterQtyValueDiscount>301.25</SubTotalAfterQtyValueDiscount>
             </DiscountSummary>
           <Tax>
                  <TaxFunction/>
                  <TaxType/>
                  <TaxCategory stdValue="A"/>
                  <TaxPercent>5.00</TaxPercent>
                  <TaxableAmount>286.90</TaxableAmount>
                  <TaxAmount>14.35</TaxAmount>
             </Tax>
         </TaxSummary>
         <TaxSummary>
              <DiscountSummary>
                  <LineItemTotals>352.50</LineItemTotals>
                  <QtyDiscount>15.38</QtyDiscount>
                  <SubTotalAfterQtyValueDiscount>337.12</SubTotalAfterQtyValueDiscount>
             </DiscountSummary>
           <Tax>
                  <TaxFunction/>
                  <TaxType/>
                  <TaxCategory stdValue="H"/>
                  <TaxPercent>17.5</TaxPercent>
                  <TaxableAmount>286.91</TaxableAmount>
                  <TaxAmount>50.21</TaxAmount>
             </Tax>
         </TaxSummary>
         <InvoiceTotals>
             <DiscountSummary>
                  <LineItemTotals>687.50</LineItemTotals>
                  <QtyDiscount>30.00</QtyDiscount>
                  <SubTotalAfterQtyValueDiscount>657.50</SubTotalAfterQtyValueDiscount>
             </DiscountSummary>
             <NetValue>573.81</NetValue>
             <TaxValue>83.69</TaxValue>
             <GrossValue>657.50</GrossValue>
         </InvoiceTotals>
    :    :    :
    </InvoiceSummary>
</Invoice>
```

Net Price, Tax Calculated at Line Level (NLL)

There are two ways to treat invoices that show net price with the tax calculated at line level:

- NLL with no discounts, no multicategory tax codes
- NLL with multicategory tax codes on total amount tax and split-total multicategory tax

Note that invoice-level discounts are not documented where tax is calculated at line level.

NLL With No Discounts, No Multicategory Tax Codes

The following tables and example XML file demonstrate a simple example of an invoice with a TaxTreatment of NLL, and with no discounts, and no multicategory tax codes.

Element	**Value**
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	InvoiceDetails/LineItemSubtotal x TaxPercent%.
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Total taxable amount for this TaxCategory. Sum of the LineItemSubtotal values for this tax category
TaxAmount	The sum of the InvoiceDetails/Tax/TaxAmount elements for this tax category.
InvoiceSummary/InvoiceTotals	
NetValue	Sum of all InvoiceDetails/LineItemSubtotal elements.

TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of NetValue and TaxValue

The following example is of an invoice with Tax Treatment NLL, with no discounts and no multi-tax-category line items.

There are three line items, providing three instances of the InvoiceDetails element.

Line 1 has a LineItemSubtotal value of 300.00, with TaxCategory A.

Line 2 has a LineItemSubtotal value of 200.00, with TaxCategory S.

Line 3 has a LineItemSubtotal value of 50.00 with TaxCategory S.

Line No.	**Product**	**Qty.**	**UOM**	**Cost**	**Subtotal**	**Tax Cat.**	**Tax Rate**	**Tax Amt**
1	Room Charge	3	EA	100.00	300.00	A	5	15.00
2	Telephone	1	EA	200.00	200.00	S	10	20.00
3	Bar Meal	1	EA	50.00	50.00	S	10	5.00

- This table demonstrates how the InvoiceSummary values have been calculated:

InvoiceDetails/Tax

	Line 1	***Line 2***	***Line 3***
TaxCategory	A	S	S
TaxPercent	5	10	10
TaxableAmount	Absent	Absent	Absent
TaxAmount	300 x 5% = 15	200 x 10% = 20	50 x 10% = 5

InvoiceSummary/TaxSummary/Tax

	TaxCode A	***TaxCode S***
TaxCategory	A	S
TaxPercent	5	10
TaxableAmount	300	200 + 50 = 250

TaxAmount	300 x 5% = 15	250 x 10% = 25

InvoiceSummary/InvoiceTotals

NetValue	300 + 200 + 50 = 550
TaxValue	15 + 25 = 40
GrossValue	550 + 40 = 590

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>300.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="A"/>
        <TaxPercent>5.00</TaxPercent>
        <TaxAmount>15.00</TaxAmount>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
            <PartDesc>Bar Meal</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>200.00</UnitPrice>
    <LineItemSubtotal>200.00</LineItemSubtotal>
```

```
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
            <TaxAmount>20.00</TaxAmount>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
      <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>3</LineItemNum>
            <PartNumDetail>
                <PartNum>78</PartNum>
                <PartDesc>Bar Meal</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>50.00</UnitPrice>
        <LineItemSubtotal>50.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
            <TaxAmount>5.00</TaxAmount>
      </Tax>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>10.00</TaxPercent>
                <TaxableAmount>250.00</TaxableAmount>
                <TaxAmount>25.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="A"/>
                <TaxPercent>5.00</TaxPercent>
                <TaxableAmount>300</TaxableAmount>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>550.00</NetValue>
            <TaxValue>40.00</TaxValue>
            <GrossValue>590.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>590.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>1999-02-11</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
```

NLL With Multicategory Tax Codes

The following tables and example XML file demonstrate an example of an invoice with a TaxTreatment of NLL, with no discounts but with multicategory tax codes.

Element	**Value**
InvoiceDetails	
UnitPrice	Net of tax
LineItemSubtotal	Net of tax
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	If this line-item is subject to split-total multicategory tax then this element should hold the proportion of the LineItemSubtotal amount liable to this tax category's tax Otherwise, it should be absent.
TaxAmount	If TaxableAmount is absent, this is the InvoiceDetails/ LineItemSubtotal x TaxPercent%. If TaxableAmount is used, this is the TaxableAmount x TaxPercent%.
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate

TaxableAmount	Total taxable amount for this TaxCategory. Calculated as: for each InvoiceDetails record subject to this TaxCategory, the hashed total of the TaxableAmount for the associated InvoiceDetails/Tax record if this has a value, otherwise the addition of the LineItemSubtotal value.
TaxAmount	The sum of the InvoiceDetails/Tax/TaxAmount elements.

InvoiceSummary/InvoiceTotals

NetValue	Sum of all InvoiceDetails/LineItemSubtotal elements.
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of NetValue and TaxValue

The following example is of an invoice with Tax Treatment NLL, where line 1 is a multicategory tax codes on total amount tax, and line 2 has split-total multicategory tax.

There are three line items, providing three instances of the InvoiceDetails element:

- Line 1 has a LineItemSubtotal value of 300.00, with tax codes S and A, and is subject to multicategory tax codes on total amount tax.
- Line 2 has a LineItemSubtotal value of 400.00, with tax codes S and H. This is subject to split-total multicategory tax, with 300.00 at tax code S, and 100.00 at tax code H.
- Line 3 has a LineItemSubtotal value of 200.00 and is subject to tax code S.

Line No.	Product	Qty	UOM	Cost	Sub-Total	Tax Cat.	Tax Rate	Tax Amt
1	Room Charge	3	EA	100.00	300.00	S A	10 5	30.00 15.00
2	Telephone	2	EA	200.00	400.00	300 @ S 100 @ H	10 20	30.00 20.00
3	Bar Meal	1	EA	200.00	200.00	S	10	20.00

This table demonstrates how the InvoiceDetail and InvoiceSummary values have been calculated:

InvoiceDetails/Tax

	Line 1, Tax Category S	*Line 1, Tax Category A*	*Line 2, Tax Category S*	*Line 2, Tax Category H*	*Line 3, Tax Category S*
Tax Category	S	A	S	H	S
Tax Percent	10	5	10	20	10
Taxable Amount	Absent	Absent	300	100	Absent
Tax Amount	300 x 10% =30	300 x 5% = 15	300 x 10% = 30	100 x 20% = 20	200 x 10% = 20

InvoiceSummary/TaxSummary/Tax

	TaxCode S	*TaxCode A*	*TaxCode H*
TaxCategory	S	A	H
TaxPercent	10	5	20
TaxableAmount	300 + 300 + 200 = 800 nb See below for further explanation	300	100
TaxAmount	30 + 30 + 20 = 80	15	20

InvoiceSummary/InvoiceTotals

NetValue	300 + 400 + 200 = 900
TaxValue	80 + 15 + 20 = 115
GrossValue	900 + 115 = 1015

TaxableAmount for TaxCategory “S” is sourced from:

- From Line 1, sourced from LineItemSubtotal (InvoiceDetails/Tax/TaxableAmount is absent), i.e. 300
- From Line 2, sourced from InvoiceDetails/Tax/TaxableAmount (as it has a value), i.e. 300
- From Line 3, sourced from LineItemSubtotal (InvoiceDetails/Tax/TaxableAmount is absent), i.e. 200

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>100.00</UnitPrice>
    <LineItemSubtotal>300.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>10.00</TaxPercent>
        <TaxAmount>30.00</TaxAmount>
    </Tax>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="A"/>
        <TaxPercent>5.00</TaxPercent>
        <TaxAmount>15.00</TaxAmount>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>98</PartNum>
            <PartDesc>Telephone</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>2</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>200.00</UnitPrice>
    <LineItemSubtotal>400.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>10.00</TaxPercent>
```

```
             <TaxableAmount>300.00</TaxableAmount>
             <TaxAmount>30.00</TaxAmount>
         </Tax>
          <Tax>
             <TaxFunction/>
             <TaxType/>
             <TaxCategory stdValue="H"/>
             <TaxPercent>20.00</TaxPercent>
             <TaxableAmount>100.00</TaxableAmount>
           <TaxAmount>20.00</TaxAmount>
         </Tax>
         <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceDetails>
         <BaseItemDetail>
             <LineItemNum>3</LineItemNum>
             <PartNumDetail>
                  <PartNum>78</PartNum>
                  <PartDesc>Bar Meal</PartDesc>
             </PartNumDetail>
             <Quantity>
                  <Qty>1</Qty>
                  <UnitOfMeasure/>
             </Quantity>
         </BaseItemDetail>
         <UnitPrice>200.00</UnitPrice>
         <LineItemSubtotal>200.00</LineItemSubtotal>
         <Tax>
             <TaxFunction/>
             <TaxType/>
             <TaxCategory stdValue="S"/>
             <TaxPercent>10.00</TaxPercent>
             <TaxAmount>20.00</TaxAmount>
         </Tax>
         <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
  <InvoiceSummary>
         <TaxSummary>
             <Tax>
                  <TaxFunction/>
                  <TaxType/>
                  <TaxCategory stdValue="S"/>
                  <TaxPercent>10.00</TaxPercent>
                  <TaxableAmount>800.00</TaxableAmount>
                  <TaxAmount>80.00</TaxAmount>
             </Tax>
         </TaxSummary>
         <TaxSummary>
             <Tax>
                  <TaxFunction/>
                  <TaxType/>
                  <TaxCategory stdValue="A"/>
                  <TaxPercent>5.00</TaxPercent>
                  <TaxableAmount>300.00</TaxableAmount>
                  <TaxAmount>15.00</TaxAmount>
             </Tax>
```

```
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="H"/>
                <TaxPercent>20.00</TaxPercent>
                <TaxableAmount>100.00</TaxableAmount>
                <TaxAmount>20.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>900.00</NetValue>
            <TaxValue>115.00</TaxValue>
            <GrossValue>1015.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>1015.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>19990211</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
```

Gross Amounts, Tax Calculated at Line Level (GLL)

There are two ways to handle invoices that show gross amounts with tax calculated at line level:

- GLL with no multicategory tax codes
- GIL with multicategory tax codes on total amount tax and split-total multicategory tax

GLL With No Multicategory Tax Codes

The following tables and example XML file demonstrate a simple example of an invoice with a TaxTreatment of GLL, and with no discounts, and no multicategory tax codes.

Element	**Value**
InvoiceDetails	
UnitPrice	Gross of tax (i.e. includes tax)
LineItemSubtotal	Gross of tax (i.e. includes tax)
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Absent
TaxAmount	The tax amount for the current line item. This can be calculated as: LineItemSubtotal - (LineItemSubtotal / (1+ (TaxPercent / 100)))
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	Total taxable amount for this TaxCategory. Calculated as: Sum of InvoiceDetails/LineItemSubtotal elements for this tax code – TaxAmount
TaxAmount	Sum of all InvoiceDetails/Tax/TaxAmount elements for this tax code
InvoiceSummary/InvoiceTotals	
NetValue	GrossValue - TaxValue
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of all the InvoiceDetails/LineItemSubtotal elements

The following example is of an invoice with Tax Treatment GLL, with no discounts and no multitax-category line items.

There are three line items, providing three instances of the InvoiceDetails element:

- Line 1 has a LineItemSubtotal value of 315.00, with TaxCategory A.
- Line 2 has a LineItemSubtotal value of 220.00, with TaxCategory S.
- Line 3 has a LineItemSubtotal value of 55.00 with TaxCategory S.

Line No.	**Product**	**Qty.**	**UOM**	**Cost**	**Subtotal**	**Tax Cat.**	**Tax Rate**	**Tax Amt**
1	Room Charge	3	EA	105.00	315.00	A	5	15.00
2	Telephone	1	EA	220.00	220.00	S	10	20.00
3	Bar Meal	1	EA	55.00	55.00	S	10	5.00

This table demonstrates how the InvoiceSummary values are calculated:

InvoiceDetails/Tax

	Line 1	***Line 2***	***Line 3***
TaxCategory	A	S	S
TaxPercent	5	10	10
TaxableAmount	Absent	Absent	Absent
TaxAmount	315 – (315 / 1.05) = 15	220 – (220 / 1.10) = 20	55 – (55 / 1.10) = 5

InvoiceSummary/TaxSummary/Tax

	TaxCode A	***TaxCode S***
TaxCategory	A	S
TaxPercent	5	10
TaxableAmount	315 – 15 = 300	(220 + 55) – (20 + 5) = 250
TaxAmount	15	20 + 5 = 25

InvoiceSummary/InvoiceTotals

NetValue	590 – 40 = 550
TaxValue	15 + 20 + 5 = 40
GrossValue	315 + 220 +55 = 590

EXAMPLE

```
<InvoiceHeader>
    :    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>105.00</UnitPrice>
    <LineItemSubtotal>315.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="A"/>
        <TaxPercent>5.00</TaxPercent>
        <TaxAmount>15.00</TaxAmount>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>78</PartNum>
            <PartDesc>Bar Meal</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>1</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>220.00</UnitPrice>
    <LineItemSubtotal>220.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>10.00</TaxPercent>
        <TaxAmount>20.00</TaxAmount>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
  <InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>3</LineItemNum>
```

```
            <PartNumDetail>
                <PartNum>78</PartNum>
                <PartDesc>Bar Meal</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>55.00</UnitPrice>
        <LineItemSubtotal>55.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
            <TaxAmount>5.00</TaxAmount>
      </Tax>
    </InvoiceDetails>
    <InvoiceSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>10.00</TaxPercent>
                <TaxableAmount>250.00</TaxableAmount>
                <TaxAmount>25.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="A"/>
                <TaxPercent>5.00</TaxPercent>
                <TaxableAmount>300</TaxableAmount>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>550.00</NetValue>
            <TaxValue>40.00</TaxValue>
            <GrossValue>590.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>590.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>1999-02-11</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
```

GLL With Multicategory Tax Codes

The following tables and example XML file demonstrate an example of an invoice with a TaxTreatment of GLL, with no discounts, but with multicategory tax codes.

Element	**Value**
InvoiceDetails	
UnitPrice	Gross of tax (i.e. includes tax)
LineItemSubtotal	Gross of tax (i.e. includes tax)
InvoiceDetails/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate
TaxableAmount	If this line-item is subject to split-total multicategory tax then this element should hold the taxable amount of the proportion of the LineItemSubtotal amount that is liable to this tax category's tax. This can be calculated as: The gross amount portion for this tax code / (1+ (TaxPercent / 100)) Otherwise, if this line-item is not subject to split-total multicategory tax, then this element should be absent
TaxAmount	If this line-item is subject to split-total multicategory tax, then this can be calculated as: The gross amount portion for this tax code – TaxableAmount Otherwise, if this line-item is not subject to split-total multicategory tax, this can be calculated as: LineItemSubtotal - (LineItemSubtotal / (1+(the sum of the TaxPercent values associated with the current line /100))) x TaxPercent%
InvoiceSummary/TaxSummary/Tax	
TaxCategory	Tax category
TaxPercent	Tax percentage rate

TaxableAmount	Total taxable amount for this TaxCategory. Calculated as: For each InvoiceDetails record subject to this TaxCategory hash total the taxable amounts using the following method: If the InvoiceDetails/Tax/TaxableAmount element has data then add the contents to the hash total Otherwise (i.e. if InvoiceDetails/Tax/TaxableAmount is absent) then get the amount to add to the hash total from the InvoiceDetails thus: LineItemSubtotal / (1 + (the sum of the TaxPercent values associated with the current line /100)) nb This is demonstrated in the example, with some further explanation after the tables showing the example calculations.
TaxAmount	Sum of all InvoiceDetails/Tax/TaxAmount elements for this tax code

InvoiceSummary/InvoiceTotals

NetValue	GrossValue - TaxValue
TaxValue	Sum of all InvoiceSummary/TaxSummary/Tax/TaxAmount elements
GrossValue	Sum of all the InvoiceDetails/LineItemSubtotal elements

EXAMPLE

The following example is of an invoice with Tax Treatment GLL, where line 1 is a multicategory tax codes on total amount tax, and line 2 has split-total multicategory tax.

There are three line items, providing three instances of the InvoiceDetails element:

- Line 1 has a LineItemSubtotal value of 345.00, with tax codes S and A, and is subject to multicategory tax codes on total amount tax.
- Line 2 has a LineItemSubtotal value of 450.00, with tax codes S and H. This is subject to split-total multicategory tax, with 330.00 at tax code S, and 120.00 at tax code H.
- Line 3 has a LineItemSubtotal value of 220.00 and is subject to tax code S.

Line No.	Product	Qty	UOM	Cost	Sub-Total	Tax Cat.	Tax Rate	Tax Amt
1	Room Charge	3	EA	115.00	345.00	S A	10 5	30.00 15.00

2	Telephone	2	EA	225.00	450.00	300 @ S 100 @ H	10 20	30.00 20.00
3	Bar Meal	1	EA	220.00	220.00	S	10	20.00

This table demonstrates how the InvoiceDetail and InvoiceSummary values have been calculated:

InvoiceDetails/Tax

	Line 1, Tax Category S	***Line 1, Tax Category A***	***Line 2, Tax Category S***	***Line 2, Tax Category H***	***Line 3, Tax Category S***
Tax Category	S	A	S	H	S
Tax Percent	10	5	10	20	10
Taxable Amount	Absent	Absent	330/1.10 = 300	120/1.20 = 100	Absent
Tax Amount	(345 / 1.15) x 10% = 30 nb See i) below for further explanation	(345 / 1.15) x 5% = 15 nb See ii) below for further explanation	330 – 300 = 30	120 – 100 = 20	(220 / 1.10) x 10% = 20

InvoiceSummary/TaxSummary/Tax

	TaxCode S	***TaxCode A***	***TaxCode H***
TaxCategory	S	A	H
TaxPercent	10	5	20
TaxableAmount	(345 / 1.15) + 300 + (220 / 1.10) = 800	345 / 1.15 = 300	100
TaxAmount	30 + 30 + 20 = 80	15	20

InvoiceSummary/InvoiceTotals

NetValue	1015 - 115 = 900
TaxValue	80 + 15 + 20 = 115
GrossValue	345 + 450 + 220 = 1015

i) The expanded equation is:
(345 – (345 / (1 + ((10 + 5) / 100))) x 10%
ii) The expanded equation is:
(345 – (345 / (1 + ((10 + 5) / 100))) x 5%

EXAMPLE

```
<InvoiceHeader>
:    :    :
</InvoiceHeader>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>1</LineItemNum>
        <PartNumDetail>
            <PartNum>100</PartNum>
            <PartDesc>Room Charge</PartDesc>
        </PartNumDetail>
        <PartNumDetail stdValue="CC">
            <PartNum>H100</PartNum>
        </PartNumDetail>
        <Quantity>
            <Qty>3</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>115.00</UnitPrice>
    <LineItemSubtotal>345.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>10.00</TaxPercent>
        <TaxAmount>30.00</TaxAmount>
    </Tax>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="A"/>
        <TaxPercent>5.00</TaxPercent>
        <TaxAmount>15.00</TaxAmount>
    </Tax>
    <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
</InvoiceDetails>
<InvoiceDetails>
    <BaseItemDetail>
        <LineItemNum>2</LineItemNum>
        <PartNumDetail>
            <PartNum>98</PartNum>
            <PartDesc>Telephone</PartDesc>
        </PartNumDetail>
        <Quantity>
            <Qty>2</Qty>
            <UnitOfMeasure/>
        </Quantity>
    </BaseItemDetail>
    <UnitPrice>225.00</UnitPrice>
    <LineItemSubtotal>450.00</LineItemSubtotal>
    <Tax>
        <TaxFunction/>
        <TaxType/>
        <TaxCategory stdValue="S"/>
        <TaxPercent>10.00</TaxPercent>
```

```
            <TaxableAmount>300.00</TaxableAmount>
            <TaxAmount>30.00</TaxAmount>
        </Tax>
         <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="H"/>
            <TaxPercent>20.00</TaxPercent>
            <TaxableAmount>100.00</TaxableAmount>
          <TaxAmount>20.00</TaxAmount>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-021-0T14:11:54</Date>
    </InvoiceDetails>
    <InvoiceDetails>
        <BaseItemDetail>
            <LineItemNum>3</LineItemNum>
            <PartNumDetail>
                <PartNum>78</PartNum>
                <PartDesc>Bar Meal</PartDesc>
            </PartNumDetail>
            <Quantity>
                <Qty>1</Qty>
                <UnitOfMeasure/>
            </Quantity>
        </BaseItemDetail>
        <UnitPrice>220.00</UnitPrice>
        <LineItemSubtotal>220.00</LineItemSubtotal>
        <Tax>
            <TaxFunction/>
            <TaxType/>
            <TaxCategory stdValue="S"/>
            <TaxPercent>10.00</TaxPercent>
            <TaxAmount>20.00</TaxAmount>
        </Tax>
        <Date stdValue="STRT" stdName="VISA:DATE">1999-02-10T14:11:54</Date>
    </InvoiceDetails>
  <InvoiceSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="S"/>
                <TaxPercent>10.00</TaxPercent>
                <TaxableAmount>800.00</TaxableAmount>
                <TaxAmount>80.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="A"/>
                <TaxPercent>5.00</TaxPercent>
                <TaxableAmount>300.00</TaxableAmount>
                <TaxAmount>15.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <TaxSummary>
            <Tax>
                <TaxFunction/>
                <TaxType/>
                <TaxCategory stdValue="H"/>
                <TaxPercent>20.00</TaxPercent>
                <TaxableAmount>100.00</TaxableAmount>
                <TaxAmount>20.00</TaxAmount>
            </Tax>
        </TaxSummary>
        <InvoiceTotals>
            <NetValue>900.00</NetValue>
            <TaxValue>115.00</TaxValue>
            <GrossValue>1015.00</GrossValue>
        </InvoiceTotals>
        <ActualPayment>
            <PaymentAmount>
                <LocalCurrencyAmt>1015.00</LocalCurrencyAmt>
            </PaymentAmount>
            <PaymentMean/>
            <PaymentDate>19990211</PaymentDate>
            <CardInfo>
                <CardNum>4917876543212345</CardNum>
                <CardExpirationDate>1199</CardExpirationDate>
                <CardType/>
            </CardInfo>
        </ActualPayment>
    </InvoiceSummary>
```

The Visa XML Invoice Stylesheet E

This appendix contains the full text of the XML invoice stylesheet version 1.0. This is the version that supports the examples given in Chapter 4, and is related to the XML invoice DTD shown in Appendix A. These appendixes are provided to assist in interpreting the examples given in this guide.

The DTD and stylesheet provided in the XML Invoice Technical Pack are the latest iteration and may be a different version than those shown here. The examples provided in the Technical Pack are dependent on the DTD and stylesheet that accompany them.

```
</xsl:script>

  <xsl:script language="VBScript">
    Function MoneyFormat(AmtStr)
     If IsNull(AmtStr) or IsEmpty(AmtStr)Then
          MoneyFormat = AmtStr
     Else
            MoneyFormat = FormatNumber(CDbl(AmtStr),2)
     End If
    End Function
  </xsl:script>

  <xsl:script language="VBScript">
    Function PercentFormat(AmtStr)
     If IsNull(AmtStr) or IsEmpty(AmtStr)Then
          PercentFormat  = AmtStr
     Else
            PercentFormat = FormatNumber(CDbl(AmtStr),2)
     End If
    End Function
  </xsl:script>
<!-- The remainder is the formatting for the document -->

<xsl:template match="/">
```

```
<html>

<xsl:for-each select="Invoice">
<head>
<title>Visa Invoice</title>

<xsl:choose>
    <xsl:when test="InvoiceHeader/InvoiceType[@stdValue=’381’]">
        <xsl:choose>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’AI’]"><h1 align="cen-
ter"><font color="navy"><u>VISA AIRLINE CREDIT NOTE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’LG’]"><h1 align="cen-
ter"><font color="navy"><u>VISA HOTEL CREDIT NOTE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’CR’]"><h1 align="cen-
ter"><font color="navy"><u>VISA CAR RENTAL CREDIT NOTE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’HC’]"><h1 align="cen-
ter"><font color="navy"><u>VISA HEALTH CARE CREDIT NOTE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’IG’]"><h1 align="cen-
ter"><font color="navy"><u>VISA GOVERNMENT CREDIT NOTE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’TS’]"><h1 align="cen-
ter"><font color="navy"><u>VISA TEMPORARY SUPPLIER CREDIT NOTE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’MM’]"><h1 align="cen-
ter"><font color="navy"><u>VISA MISCELLANEOUS SUPPLIER CREDIT NOTE</u></font></h1></
xsl:when>
            <xsl:otherwise><h1 align="center"><font color="navy"><u>VISA INVOICE</u></
font></h1></xsl:otherwise>
        </xsl:choose>
    </xsl:when>
    <xsl:otherwise>
        <xsl:choose>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’AI’]"><h1 align="cen-
ter"><font color="navy"><u>VISA AIRLINE INVOICE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’LG’]"><h1 align="cen-
ter"><font color="navy"><u>VISA HOTEL INVOICE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’CR’]"><h1 align="cen-
ter"><font color="navy"><u>VISA CAR RENTAL INVOICE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’HC’]"><h1 align="cen-
ter"><font color="navy"><u>VISA HEALTH CARE INVOICE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’IG’]"><h1 align="cen-
ter"><font color="navy"><u>VISA GOVERNMENT INVOICE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’TS’]"><h1 align="cen-
ter"><font color="navy"><u>VISA TEMPORARY SUPPLIER INVOICE</u></font></h1></xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’MM’]"><h1 align="cen-
ter"><font color="navy"><u>VISA MISCELLANEOUS SUPPLIER INVOICE</u></font></h1></xsl:when>
            <xsl:otherwise><h1 align="center"><font color="navy"><u>VISA INVOICE</u></
font></h1></xsl:otherwise>
        </xsl:choose>
    </xsl:otherwise>
</xsl:choose>
</head>

<body bgcolor="burlywood" text ="navy">

<!-- Invoice number, Message Type and Invoice Currency -->
<table border="0" cellpadding="3" cellspacing="2">
        <tr>
```

```
            <td><font color="blue">Card Number: </font>
            <xsl:value-of select="InvoiceSummary/ActualPayment/CardInfo/CardNum[0][../
CardType/@stdValue='VS']" />
            </td>
            <td></td>
        </tr>
</table>
<table border="0" cellpadding="3" cellspacing="2">
    <tr>
        <td><font color="blue">Message Type</font></td>
        <td><font color="blue">Invoice Number</font></td>
        <td><font color="blue">Invoice Date/Time</font></td>
        <td><font color="blue">Currency</font></td>
    </tr>

    <xsl:for-each select="InvoiceHeader">

<!-- This following line substitutes an attribute and a code lookup
InvoiceType Field -->
    <tr>
        <xsl:apply-templates select="InvoiceType" />

<!-- Invoice number -->

        <td><xsl:value-of select="InvoiceNumber" /></td>
<!-- Invoice Date
This entry checks to see if the date has a value of 3 and enters that in
the field if it does
-->
    <xsl:apply-templates select="InvoiceDate" />

<!-- Lookup for Currency Code as above for InvoiceType -->
    <xsl:apply-templates select="Currency" />

    </tr>
</xsl:for-each>
</table>

<p />
<hr size="1"></hr>

<!-- Guest details block -->
<p>
    <table border="0" cellpadding="0" cellspacing="5" align="centre">
        <xsl:choose>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='AI']">
                <tr>
                    <td><font color="blue">Passenger Name:</font></td>
                    <td><xsl:value-of select="InvoiceHeader/Party[@stdValue='BY']/Con-
tact/Name1" /></td>
<!-- Airline Specific Data -->
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='FLNO']"><td><font col-
or="blue">Flight Number:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='FLNO']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='CARR']"><td><font col-
or="blue">Carrier Code:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='CARR']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='LOC1']"><td><font col-
or="blue">Origination City:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value='LOC1']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='LOC2']"><td><font col-
or="blue">Destination City:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value='LOC2']" /></td></xsl:if>
                </tr>
                <tr>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='SRVC']"><td><font col-
or="blue">Service Class:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='SRVC']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='CPNO']"><td><font col-
or="blue">Coupon Number:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='CPNO']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='STOP']"><td><font col-
or="blue">Stopover Indicator:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value='STOP']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='FBC']"><td><font col-
or="blue">Fare Basis Code:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='FBC']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='REFI']"><td><font col-
or="blue">Refund Indicator:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value='REFI']" /></td></xsl:if>
                </tr>
            </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='LG']">
                <tr>
                    <td><font color="blue">Guest Name:</font></td>
                    <td><xsl:value-of select="InvoiceHeader/Party[@stdValue='BY']/Con-
tact/Name1" /></td>
<!-- Hotel Specific Data -->
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='RSNO']"><td><font col-
or="blue">Reservation number:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value='RSNO']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='RMNO']"><td><font col-
or="blue">Room number:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='RMNO']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='RMRT']"><td><font col-
or="blue">Room rate:</font></td><td align="right"><xsl:for-each select="InvoiceHeader/
Ref[@stdValue='RMRT'][0]" ><xsl:eval>MoneyFormat(nodeTypedValue)</xsl:eval></xsl:for-
each></td></xsl:if>
                </tr>
            </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='CR']">
                <tr>
                    <td><font color="blue">Renter:</font></td>
                    <td><xsl:value-of select="InvoiceHeader/Party[@stdValue='BY']/Name/
Name1" /></td>
<!-- Car Rental Specific Data -->
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='RSNO']"><td><font col-
or="blue">Reservation Number:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value='RSNO']" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue='LOC1']"><td><font col-
or="blue">Rented From:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue='LOC1']" /></td></xsl:if>
```

```
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’LOC2’]"><td><font col-
or="blue">Returned To:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue=’LOC2’]" /></td></xsl:if>
                </tr>
                <tr>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’DCI’]"><td><font col-
or="blue">Checkin Reading:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue=’DCI’]" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’DCO’]"><td><font col-
or="blue">Checkout Reading:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value=’DCO’]" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’DDV’]"><td><font col-
or="blue">Distance Travelled:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value=’DDV’]" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’UOD’]"><td><font col-
or="blue">Unit of distance:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@std-
Value=’UOD’]" /></td></xsl:if>
                </tr>
                <tr>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’VREG’]"><td><font col-
or="blue">Vehicle Registration No:</font></td><td><xsl:value-of select="InvoiceHeader/
Ref[@stdValue=’VREG’]" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’VT’]"><td><font col-
or="blue">Vehicle Type:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue=’VT’]" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’VGCH’]"><td><font col-
or="blue">Vehicle Group Charged:</font></td><td><xsl:value-of select="InvoiceHeader/
Ref[@stdValue=’VGCH’]" /></td></xsl:if>
                    <xsl:if test="InvoiceHeader/Ref[@stdValue=’OVD’]"><td><font col-
or="blue">Other Data:</font></td><td><xsl:value-of select="InvoiceHeader/Ref[@stdVal-
ue=’OVD’]" /></td></xsl:if>
                </tr>
            </xsl:when>
<!--        Other Sectors ???
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’HC’]"> </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’IG’]"> </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’TS’]"> </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’MM’]"> </xsl:when>
-->
            <xsl:otherwise>
                <tr>
                    <td><font color="blue">Invoicee:</font></td>
                    <td><xsl:value-of select="InvoiceHeader/Party[@stdValue=’BY’]/Name/
Name1" /></td>
                </tr>
            </xsl:otherwise>
        </xsl:choose>
    </table>
</p>
<hr size="1"></hr>
<!-- Date Information -->
<xsl:if test="InvoiceHeader/Ref[@stdValue=’ADQ’][.=’AI’ || . =’CR’ || . =’LG’]">
<p>
<table border="1">
    <tr>
        <xsl:choose>
```

```
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='AI']">
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Departure Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Departure Time</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">Arrival Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">Arrival Time</font></th></xsl:if>
            </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='LG']">
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Check In Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Check In Time</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">Check Out Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">Check Out Time</font></th></xsl:if>
            </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='CR']">
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Check Out Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Check Out Time</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">Check In Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">Check In Time</font></th></xsl:if>
            </xsl:when>
<!--        Other Sectors do not have these dates ???
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='HC']"> </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='IG']"> </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='TS']"> </xsl:when>
            <xsl:when test="InvoiceHeader/Ref[@stdValue='ADQ'][.='MM']"> </xsl:when>
-->         <xsl:otherwise>
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Start Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='STRT']"><th><font col-
or="blue">Start Time</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">End Date</font></th></xsl:if>
                <xsl:if test="InvoiceHeader/Date[@stdValue='END']"><th><font col-
or="blue">End Time</font></th></xsl:if>
</xsl:otherwise>
        </xsl:choose>
    </tr>

    <xsl:for-each select="InvoiceHeader">
    <!--
    The following syntax <xsl:for-each select="Date[@stdValue='STRT']">
    allows the selection of date based on the value of the stdValue attribute.
    This is extremely useful and important
    -->
    <tr>
        <td><font size="-1"><xsl:for-each select="Date[@stdValue='STRT']"><xsl:eval>Date-
```

```
Format(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
        <td><font size="-1"><xsl:for-each select="Date[@stdVal-
ue=’STRT’]"><xsl:eval>ShortTimeFormat(nodeTypedValue)</xsl:eval></xsl:for-each></font></
td>
        <td><font size="-1"><xsl:for-each select="Date[@stdValue=’END’]"><xsl:eval>Date-
Format(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
        <td><font size="-1"><xsl:for-each select="Date[@stdValue=’END’]"><xsl:eval>Short-
TimeFormat(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
    </tr>

</xsl:for-each>
</table>
</p>
<hr size="1"></hr>
</xsl:if>

<!-- Supplier Details -->
<p>
<table border="1" cellpadding="0">
    <tr>
        <th><font color="blue">Supplier</font></th>
        <th><font color="blue">Address</font></th>
        <th><font color="blue">Country</font></th>
        <th><font color="blue">Tax Reg No</font></th>
        <th><font color="blue">Reg No</font></th>
    </tr>
    <tr><font size="-1">
    <xsl:for-each select="InvoiceHeader/Party[@stdValue= ’SU’]">
        <td valign="top"><xsl:value-of select="Name/Name1" /><BR /><xsl:value-of se-
lect="Name/Name2" /></td>
        <td valign="top">
            <xsl:if test="Street/Street1"><xsl:value-of select="Street/Street1" /><BR /
></xsl:if>
            <xsl:if test="PostalInfo/City"><xsl:value-of select="PostalInfo/City" /><BR /
></xsl:if>
            <xsl:if test="PostalInfo/CountrySubEntity"><xsl:value-of select="PostalInfo/
CountrySubEntity" /><BR /></xsl:if>
            <xsl:if test="PostalInfo/PostalCode"><xsl:value-of select="PostalInfo/Postal-
Code" /></xsl:if>
        </td>
        <td valign="top"><xsl:value-of select="PostalInfo/Country" /></td>
        <td valign="top"><xsl:value-of select="Ref[@stdValue=’VA’]" /></td>
        <td valign="top"><xsl:value-of select="Ref[@stdValue=’XA’]" /></td>
    </xsl:for-each>
    </font></tr>
</table>
</p>
<hr size="1"></hr>
<p>
<!-- Line item details -->
<table border="2" cellpadding="1" cellspacing="1" cellheight="2" frame="border" >
<xsl:for-each select="InvoiceDetails">
<xsl:if test="context()[0]">
<tr>
<xsl:if test="//InvoiceDetails/BaseItemDetail/LineItemNum">
    <th><font color="blue">Line No</font></th>
```

```
</xsl:if>
<xsl:if test="//InvoiceDetails/BaseItemDetail/PartNumDetail[@stdValue=’CC’]/PartNum">
    <th><font color="blue">Commodity Code</font></th>
</xsl:if>

<xsl:if test="//InvoiceDetails/BaseItemDetail/PartNumDetail[@stdValue=’VP’]/PartNum">
    <th><font color="blue">Product Code</font></th>
</xsl:if>
<th><font color="blue">Description</font></th>
<th><font color="blue">Qty</font></th>
<th><font color="blue">UOM</font></th>
<th><font color="blue">Cost</font></th>
<th><font color="blue">Sub-Total</font></th>
<th><font color="blue">Tax Rate(%)</font></th>
<xsl:if test="//InvoiceHeader/Ref[@stdValue=’ADQ’][.=’LG’]">
    <th><font color="blue">Purchase Date/Time</font></th>
</xsl:if>
</tr>
</xsl:if>
<tr>
<xsl:choose>
    <xsl:when test="BaseItemDetail[./SubLineItemNum]">
        <xsl:if test="//InvoiceDetails/BaseItemDetail/LineItemNum">
            <td ></td>
        </xsl:if>
        <xsl:if test="//InvoiceDetails/BaseItemDetail/PartNumDetail[@stdValue=’CC’]/PartNum">
            <td ></td>
        </xsl:if>
        <xsl:if test="//InvoiceDetails/BaseItemDetail/PartNumDetail[@stdValue=’VP’]/PartNum">
            <td ></td>
        </xsl:if>
        <td ></td>
        <td align="left" colspan="5">
            <font size="-1">
                <xsl:value-of select="Ref[@stdValue=’LOC1’]" />-><xsl:value-of select="Ref[@stdValue=’LOC2’]" /> <xsl:value-of select="Ref[@stdValue=’CARR’]" /><xsl:value-of select="Ref[@stdValue=’FLNO’]" /> <xsl:for-each select="Date[@stdValue=’STRT’]"><xsl:eval>DateFormat(nodeTypedValue)</xsl:eval> <xsl:eval>ShortTimeFormat(nodeTypedValue)</xsl:eval> </xsl:for-each> <xsl:value-of select="Ref[@stdValue=’SRVC’]" />
<xsl:apply-templates select="Ref[@stdValue=’STOP’]" /> <xsl:value-of select="Ref[@stdValue=’REFI’]" />
            </font>
        </td>
    </xsl:when>
    <xsl:otherwise>
        <xsl:if test="//InvoiceDetails/BaseItemDetail/LineItemNum">
            <td align="right"><font size="-1"><xsl:value-of select="BaseItemDetail/LineItemNum" /></font></td>
        </xsl:if>
        <xsl:if test="//InvoiceDetails/BaseItemDetail/PartNumDetail[@stdValue=’CC’]/PartNum">
            <td align="right"><font size="-1"><xsl:value-of select="BaseItemDetail/PartNumDetail[@stdValue=’CC’]/PartNum" /></font></td>
        </xsl:if>
        <xsl:if test="//InvoiceDetails/BaseItemDetail/PartNumDetail[@stdValue=’VP’]/Part-
Num">
            <td align="right"><font size="-1"><xsl:value-of select="BaseItemDetail/Part-
NumDetail[@stdValue=’VP’]/PartNum" /></font></td>
        </xsl:if>
        <td ><font size="-1"><xsl:value-of select="BaseItemDetail/PartNumDetail[@stdVal-
ue=’VP’]/PartDesc" /></font></td>
        <td align="right"><font size="-1"><xsl:for-each select="BaseItemDetail/Quantity/
Qty" ><xsl:eval>QtyFormat(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
        <td ><font size="-1"><xsl:value-of select="BaseItemDetail/Quantity/UnitOfMeasure/
@stdValue" /></font></td>
        <td align="right"><font size="-1"><xsl:for-each select="UnitPrice" ><xsl:eval>Mon-
eyFormat(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
        <td align="right"><font size="-1"><xsl:for-each select="LineItemSubtotal"
><xsl:eval>MoneyFormat(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
        <td align="right"><font size="-1"><xsl:for-each select="Tax[0]/TaxPercent"
><xsl:eval>PercentFormat(nodeTypedValue)</xsl:eval></xsl:for-each></font></td>
        <xsl:if test="//InvoiceHeader/Ref[@stdValue=’ADQ’][.=’LG’]">
            <td nowrap=""><font size="-1"><xsl:for-each select="Date[@stdVal-
ue=’STRT’]"><xsl:eval>DateFormat(nodeTypedValue)</xsl:eval> <xsl:eval>TimeFormat(node-
TypedValue)</xsl:eval></xsl:for-each></font></td>
        </xsl:if>
    </xsl:otherwise>
</xsl:choose>
</tr>
</xsl:for-each>
</table>
</p>

<!-- Summary Details -->
<hr size="1" color="navy"></hr>
<h2><u>SUMMARY DETAILS</u></h2>

<!-- Payment details -->
<xsl:if test=" InvoiceSummary/ActualPayment">
<p>
<H3>PAYMENT SUMMARY DETAILS</H3>
<table border="1" cellpadding="3" cellspacing="2" frame="border">
<xsl:for-each select="InvoiceSummary/ActualPayment" order-by="-CardInfo/CardExpiration-
Date">
    <xsl:if test="context()[0]">
        <tr>
            <th><font color="blue">Payment<br/>Method</font></th>
            <th><font color="blue">Payment <br/>Date</font></th>
            <th><font color="blue">Amount<br/>Paid</font></th>
            <xsl:if test="../ActualPayment/PaymentAmount/ForeignCurrencyPayment/Foreign-
CurrencyAmt"><th><font color="blue">Foreign Current Amt</font></th><th><font col-
or="blue">Foreign Currency Code</font></th></xsl:if>
            <th><font color="blue">Expires</font></th>
            <th><font color="blue">Card Used<br/>For Payment</font></th>
        </tr>
    </xsl:if>

    <tr>
        <xsl:choose>
            <xsl:when test="CardInfo/CardType/@stdValue"><xsl:apply-templates select="
```

```
CardInfo/CardType" /></xsl:when>
            <xsl:otherwise><xsl:apply-templates select="PaymentMean" /></xsl:otherwise>
        </xsl:choose>
        <td><xsl:for-each select="PaymentDate"><xsl:eval>DateFormat(nodeTypedValue)</
xsl:eval></xsl:for-each></td>
        <td align="right"><xsl:for-each select="PaymentAmount/LocalCurrencyAmt"
><xsl:eval>MoneyFormat(nodeTypedValue)</xsl:eval></xsl:for-each></td>
        <xsl:choose>
            <xsl:when test="PaymentAmount/ForeignCurrencyPayment/ForeignCurrencyAmt">
                <td align="right"><xsl:for-each select="PaymentAmount/ForeignCurrency-
Payment/ForeignCurrencyAmt" ><xsl:eval>MoneyFormat(nodeTypedValue)</xsl:eval></xsl:for-
each></td>
                <xsl:apply-templates select="PaymentAmount/ForeignCurrencyPayment/Cur-
rency" />
            </xsl:when>
            <xsl:otherwise>
                    <xsl:if test="../ActualPayment/PaymentAmount/ForeignCurrencyPayment/
ForeignCurrencyAmt"><td /><td /></xsl:if>
            </xsl:otherwise>
        </xsl:choose>
        <td><xsl:value-of select="CardInfo/CardExpirationDate" /></td>
        <td><xsl:value-of select="CardInfo/CardNum" /></td>
    </tr>
</xsl:for-each>
</table>
</p>
<hr size="1" />
</xsl:if>

<xsl:if test="InvoiceSummary/InvoiceTotals">
<p>
    <!-- Total Details -->
    <H3>TOTAL SUMMARY DETAILS</H3>
    <table border="1" cellpadding="3" cellspacing="2">
    <xsl:for-each select="InvoiceSummary/InvoiceTotals">
        <xsl:if test="context()[0]">
            <tr>
                <th><font color="blue">Total Net Amount</font></th>
                <th><font color="blue">Total Tax Amount </font></th>
                <th><font color="blue">Total Gross Amount </font></th>
            </tr>
        </xsl:if>
        <tr>
            <td align="right"><xsl:for-each select="NetValue" ><xsl:eval>MoneyFormat(no-
deTypedValue)</xsl:eval></xsl:for-each></td>
            <td align="right"><xsl:for-each select="TaxValue" ><xsl:eval>MoneyFormat(no-
deTypedValue)</xsl:eval></xsl:for-each></td>
            <td align="right"><xsl:for-each select="GrossValue" ><xsl:eval>MoneyFor-
mat(nodeTypedValue)</xsl:eval></xsl:for-each></td>
        </tr>
    </xsl:for-each>
    </table>
    </p>
    <hr size="1" />
</xsl:if>
```

```
<!-- Tax Details -->
<xsl:if test="InvoiceSummary/TaxSummary/Tax">
<p>
<H3>TAX SUMMARY DETAILS</H3>
<table border="1" cellpadding="3" cellspacing="2">
    <xsl:for-each select="InvoiceSummary/TaxSummary/Tax" order-by="TaxPercent">
        <xsl:if test="context()[0]">
            <tr>
                <th><font color="blue">Tax Rate(%)</font></th>
                <xsl:if test="../../TaxSummary/Tax/TaxCategory[@stdValue]">
                    <th><font color="blue">Tax Category</font></th>
                </xsl:if>
                <th><font color="blue">Total Net Amount</font></th>
                <th><font color="blue">Total Tax</font></th>
            </tr>
        </xsl:if>
        <tr>
            <td align="right"><xsl:for-each select="TaxPercent" ><xsl:eval>PercentFor-
mat(nodeTypedValue)</xsl:eval></xsl:for-each></td>
            <xsl:if test="../../TaxSummary/Tax/TaxCategory[@stdValue]">
                <xsl:apply-templates select="TaxCategory" />
            </xsl:if>
            <td align="right"><xsl:for-each select="TaxableAmount" ><xsl:eval>MoneyFor-
mat(nodeTypedValue)</xsl:eval></xsl:for-each></td>
            <td align="right"><xsl:for-each select="TaxAmount" ><xsl:eval>MoneyFormat(no-
deTypedValue)</xsl:eval></xsl:for-each></td>
        </tr>
    </xsl:for-each>
</table>
</p>
</xsl:if>

</body>
</xsl:for-each>
</html>
<xsl:apply-templates />
</xsl:template>

<xsl:template match="InvoiceType">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’381’]"><td>Credit Note</td></xsl:when>
        <xsl:when test="@stdValue[.=’380’]"><td>Invoice</td></xsl:when>
        <xsl:otherwise><td>Value not in list</td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="InvoiceStatus">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’9’]"><td>Original</td></xsl:when>
        <xsl:when test="@stdValue[.=’10’]"><td>Copy</td></xsl:when>
        <xsl:when test="@stdValue[.=’53’]"><td>Test</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Invoice
Status <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
```

```
</xsl:template>

<xsl:template match="InvoiceTreatment">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’P’]"><td>Invoice printed and given to purchaser, and
used for tax reclaim </td></xsl:when>
        <xsl:when test="@stdValue[.=’EP’]"><td>Printed, but printed invoice treated as
supplemental invoice since electronic copy used for tax reclaim</td></xsl:when>
        <xsl:when test="@stdValue[.=’E’]"><td>Printed invoice suppressed since electronic
master version used for tax reclaim</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Invoice
Treatment <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="Currency">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’AED’]"><td>Dirham</td></xsl:when>
        <xsl:when test="@stdValue[.=’AFA’]"><td>Afghani</td></xsl:when>
        <xsl:when test="@stdValue[.=’ALL’]"><td>Lek</td></xsl:when>
        <xsl:when test="@stdValue[.=’AMD’]"><td>Dram</td></xsl:when>
        <xsl:when test="@stdValue[.=’ANG’]"><td>Guilder</td></xsl:when>
        <xsl:when test="@stdValue[.=’AOK’]"><td>New Kwanza</td></xsl:when>
        <xsl:when test="@stdValue[.=’ARP’]"><td>Peso</td></xsl:when>
        <xsl:when test="@stdValue[.=’ATS’]"><td>Schilling</td></xsl:when>
        <xsl:when test="@stdValue[.=’AUD’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’AWF’]"><td>Florin</td></xsl:when>
        <xsl:when test="@stdValue[.=’AZM’]"><td>Manat</td></xsl:when>
        <xsl:when test="@stdValue[.=’BAK’]"><td>Convertible Mk</td></xsl:when>
        <xsl:when test="@stdValue[.=’BBD’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’BDT’]"><td>Taka</td></xsl:when>
        <xsl:when test="@stdValue[.=’BEF’]"><td>Franc</td></xsl:when>
        <xsl:when test="@stdValue[.=’BGL’]"><td>Lev</td></xsl:when>
        <xsl:when test="@stdValue[.=’BHD’]"><td>Dinar</td></xsl:when>
        <xsl:when test="@stdValue[.=’BIF’]"><td>Burundi Franc</td></xsl:when>
        <xsl:when test="@stdValue[.=’BMD’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’BND’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’BOB’]"><td>Boliviano</td></xsl:when>
        <xsl:when test="@stdValue[.=’BRR’]"><td>Brazilian Real</td></xsl:when>
        <xsl:when test="@stdValue[.=’BSD’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’BTR’]"><td>Rupee</td></xsl:when>
        <xsl:when test="@stdValue[.=’BWP’]"><td>Pula</td></xsl:when>
        <xsl:when test="@stdValue[.=’BYR’]"><td>Ruble</td></xsl:when>
        <xsl:when test="@stdValue[.=’BZD’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’CAD’]"><td>Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’CHF’]"><td>Franc</td></xsl:when>
        <xsl:when test="@stdValue[.=’CLP’]"><td>Peso</td></xsl:when>
        <xsl:when test="@stdValue[.=’CNY’]"><td>Yuan Renminbi</td></xsl:when>
        <xsl:when test="@stdValue[.=’COP’]"><td>Peso</td></xsl:when>
        <xsl:when test="@stdValue[.=’CRC’]"><td>Colon</td></xsl:when>
        <xsl:when test="@stdValue[.=’CUP’]"><td>Peso</td></xsl:when>
        <xsl:when test="@stdValue[.=’CVE’]"><td>Escudo</td></xsl:when>
        <xsl:when test="@stdValue[.=’CYP’]"><td>Pound</td></xsl:when>
        <xsl:when test="@stdValue[.=’DEM’]"><td>Deutsche Mark</td></xsl:when>
        <xsl:when test="@stdValue[.=’DJF’]"><td>Franc</td></xsl:when>
```

```
<xsl:when test="@stdValue[.=’DKK’]"><td>Krone</td></xsl:when>
<xsl:when test="@stdValue[.=’DOP’]"><td>Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’DZD’]"><td>Algerian Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’ECS’]"><td>Sucre</td></xsl:when>
<xsl:when test="@stdValue[.=’EEK’]"><td>Kroon</td></xsl:when>
<xsl:when test="@stdValue[.=’EGP’]"><td>Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’ERN’]"><td>Nakfa</td></xsl:when>
<xsl:when test="@stdValue[.=’ESP’]"><td>Peseta</td></xsl:when>
<xsl:when test="@stdValue[.=’ETB’]"><td>Birr</td></xsl:when>
<xsl:when test="@stdValue[.=’EUR’]"><td>Euro</td></xsl:when>
<xsl:when test="@stdValue[.=’FIM’]"><td>Markka</td></xsl:when>
<xsl:when test="@stdValue[.=’FJD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’FKP’]"><td>Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’FRF’]"><td>Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’GBP’]"><td>Pound Sterling</td></xsl:when>
<xsl:when test="@stdValue[.=’GEL’]"><td>Lari</td></xsl:when>
<xsl:when test="@stdValue[.=’GHC’]"><td>Cedi</td></xsl:when>
<xsl:when test="@stdValue[.=’GIP’]"><td>Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’GMD’]"><td>Dalasi</td></xsl:when>
<xsl:when test="@stdValue[.=’GNF’]"><td>Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’GRD’]"><td>Drachma</td></xsl:when>
<xsl:when test="@stdValue[.=’GTQ’]"><td>Quetzal</td></xsl:when>
<xsl:when test="@stdValue[.=’GYD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’HKD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’HNL’]"><td>Lempira</td></xsl:when>
<xsl:when test="@stdValue[.=’HRK’]"><td>Kuna</td></xsl:when>
<xsl:when test="@stdValue[.=’HTG’]"><td>Gourde</td></xsl:when>
<xsl:when test="@stdValue[.=’HUF’]"><td>Forint</td></xsl:when>
<xsl:when test="@stdValue[.=’IDR’]"><td>Rupiah</td></xsl:when>
<xsl:when test="@stdValue[.=’IEP’]"><td>Punt</td></xsl:when>
<xsl:when test="@stdValue[.=’ILS’]"><td>Shekel</td></xsl:when>
<xsl:when test="@stdValue[.=’INR’]"><td>Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’IQD’]"><td>Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’IRR’]"><td>Rial</td></xsl:when>
<xsl:when test="@stdValue[.=’ISK’]"><td>Krona</td></xsl:when>
<xsl:when test="@stdValue[.=’ITL’]"><td>Lira</td></xsl:when>
<xsl:when test="@stdValue[.=’JMD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’JOD’]"><td>Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’JPY’]"><td>Yen</td></xsl:when>
<xsl:when test="@stdValue[.=’KES’]"><td>Shilling</td></xsl:when>
<xsl:when test="@stdValue[.=’KGS’]"><td>Som</td></xsl:when>
<xsl:when test="@stdValue[.=’KHR’]"><td>Riel</td></xsl:when>
<xsl:when test="@stdValue[.=’KMF’]"><td>Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’KPW’]"><td>Won</td></xsl:when>
<xsl:when test="@stdValue[.=’KRW’]"><td>Won</td></xsl:when>
<xsl:when test="@stdValue[.=’KWD’]"><td>Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’KYD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’KZT’]"><td>Tenge</td></xsl:when>
<xsl:when test="@stdValue[.=’LAK’]"><td>Kip</td></xsl:when>
<xsl:when test="@stdValue[.=’LBP’]"><td>Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’LKR’]"><td>Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’LRD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’LSL’]"><td>Loti</td></xsl:when>
<xsl:when test="@stdValue[.=’LTL’]"><td>Lita</td></xsl:when>
<xsl:when test="@stdValue[.=’LUF’]"><td>Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’LVL’]"><td>Lat</td></xsl:when>
```

```
<xsl:when test="@stdValue[.=’LYD’]"><td>Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’MAD’]"><td>Dirham</td></xsl:when>
<xsl:when test="@stdValue[.=’MDL’]"><td>Leu</td></xsl:when>
<xsl:when test="@stdValue[.=’MGF’]"><td>Malagasy Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’MKD’]"><td>Denar</td></xsl:when>
<xsl:when test="@stdValue[.=’MMK’]"><td>Kyat</td></xsl:when>
<xsl:when test="@stdValue[.=’MNT’]"><td>Tugrik</td></xsl:when>
<xsl:when test="@stdValue[.=’MOP’]"><td>Pataca</td></xsl:when>
<xsl:when test="@stdValue[.=’MRO’]"><td>Ouguiya</td></xsl:when>
<xsl:when test="@stdValue[.=’MTL’]"><td>Lira</td></xsl:when>
<xsl:when test="@stdValue[.=’MUR’]"><td>Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’MVR’]"><td>Rufiyaa</td></xsl:when>
<xsl:when test="@stdValue[.=’MWK’]"><td>Kwacha</td></xsl:when>
<xsl:when test="@stdValue[.=’MXN’]"><td>Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’MYR’]"><td>Ringgit</td></xsl:when>
<xsl:when test="@stdValue[.=’MZM’]"><td>Metical</td></xsl:when>
<xsl:when test="@stdValue[.=’NAD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’NGN’]"><td>Naira</td></xsl:when>
<xsl:when test="@stdValue[.=’NIO’]"><td>Cordoba Oro</td></xsl:when>
<xsl:when test="@stdValue[.=’NLG’]"><td>Guilder</td></xsl:when>
<xsl:when test="@stdValue[.=’NOK’]"><td>Krone</td></xsl:when>
<xsl:when test="@stdValue[.=’NPR’]"><td>Nepalese Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’NZD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’OMR’]"><td>Sul Rial</td></xsl:when>
<xsl:when test="@stdValue[.=’PAB’]"><td>Balboa</td></xsl:when>
<xsl:when test="@stdValue[.=’PEN’]"><td>Nuevo Sol</td></xsl:when>
<xsl:when test="@stdValue[.=’PGK’]"><td>Kina</td></xsl:when>
<xsl:when test="@stdValue[.=’PHP’]"><td>Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’PKR’]"><td>Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’PLZ’]"><td>Zloty</td></xsl:when>
<xsl:when test="@stdValue[.=’PTE’]"><td>Escudo</td></xsl:when>
<xsl:when test="@stdValue[.=’PYG’]"><td>Guarani</td></xsl:when>
<xsl:when test="@stdValue[.=’QAR’]"><td>Rial</td></xsl:when>
<xsl:when test="@stdValue[.=’ROL’]"><td>Leu</td></xsl:when>
<xsl:when test="@stdValue[.=’RUR’]"><td>Ruble</td></xsl:when>
<xsl:when test="@stdValue[.=’RWF’]"><td>Rwanda Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’SAR’]"><td>Riyal</td></xsl:when>
<xsl:when test="@stdValue[.=’SBD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’SBL’]"><td>Luigino</td></xsl:when>
<xsl:when test="@stdValue[.=’SCR’]"><td>Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’SDD’]"><td>Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’SEK’]"><td>Krona</td></xsl:when>
<xsl:when test="@stdValue[.=’SGD’]"><td>Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’SHP’]"><td>Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’SIT’]"><td>Tolar</td></xsl:when>
<xsl:when test="@stdValue[.=’SKK’]"><td>Koruna</td></xsl:when>
<xsl:when test="@stdValue[.=’SLL’]"><td>Leone</td></xsl:when>
<xsl:when test="@stdValue[.=’SOS’]"><td>Shilling</td></xsl:when>
<xsl:when test="@stdValue[.=’SRG’]"><td>Guilder</td></xsl:when>
<xsl:when test="@stdValue[.=’STD’]"><td>Dobra</td></xsl:when>
<xsl:when test="@stdValue[.=’SVC’]"><td>Colon</td></xsl:when>
<xsl:when test="@stdValue[.=’SYP’]"><td>Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’SZL’]"><td>Lilangeni</td></xsl:when>
<xsl:when test="@stdValue[.=’THB’]"><td>Baht</td></xsl:when>
<xsl:when test="@stdValue[.=’TJR’]"><td>Ruble</td></xsl:when>
<xsl:when test="@stdValue[.=’TMM’]"><td>Manat</td></xsl:when>
```

```
         <xsl:when test="@stdValue[.=’TND’]"><td>Dinar</td></xsl:when>
         <xsl:when test="@stdValue[.=’TOP’]"><td>Pa’anga</td></xsl:when>
         <xsl:when test="@stdValue[.=’TRL’]"><td>Lira</td></xsl:when>
         <xsl:when test="@stdValue[.=’TTD’]"><td>Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’TWD’]"><td>Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’TZS’]"><td>Shilling</td></xsl:when>
         <xsl:when test="@stdValue[.=’UAH’]"><td>Hryvnia</td></xsl:when>
         <xsl:when test="@stdValue[.=’UGX’]"><td>Shilling</td></xsl:when>
         <xsl:when test="@stdValue[.=’USD’]"><td>U.S. Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’UYU’]"><td>Peso</td></xsl:when>
         <xsl:when test="@stdValue[.=’UZS’]"><td>Som</td></xsl:when>
         <xsl:when test="@stdValue[.=’VEB’]"><td>Bolivar</td></xsl:when>
         <xsl:when test="@stdValue[.=’VND’]"><td>Dong</td></xsl:when>
         <xsl:when test="@stdValue[.=’VUV’]"><td>Vatu</td></xsl:when>
         <xsl:when test="@stdValue[.=’WST’]"><td>Tala</td></xsl:when>
         <xsl:when test="@stdValue[.=’XAF’]"><td>Franc BEAC</td></xsl:when>
         <xsl:when test="@stdValue[.=’XAG’]"><td>Ounces</td></xsl:when>
         <xsl:when test="@stdValue[.=’XAU’]"><td>Ounces</td></xsl:when>
         <xsl:when test="@stdValue[.=’XCD’]"><td>E. Caribbean Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’XOF’]"><td>CFA Franc BCEAO</td></xsl:when>
         <xsl:when test="@stdValue[.=’XPF’]"><td>CFP Franc</td></xsl:when>
         <xsl:when test="@stdValue[.=’XPT’]"><td>Ounces</td></xsl:when>
         <xsl:when test="@stdValue[.=’YER’]"><td>Rial</td></xsl:when>
         <xsl:when test="@stdValue[.=’ZAR’]"><td>Rand</td></xsl:when>
         <xsl:when test="@stdValue[.=’ZMK’]"><td>Kwacha</td></xsl:when>
         <xsl:when test="@stdValue[.=’ZRN’]"><td>New Zaire</td></xsl:when>
         <xsl:when test="@stdValue[.=’ZWD’]"><td>Zimbabwe Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’004’]"><td>Afghani</td></xsl:when>
         <xsl:when test="@stdValue[.=’008’]"><td>Lek</td></xsl:when>
         <xsl:when test="@stdValue[.=’012’]"><td>Algerian Dinar</td></xsl:when>
         <xsl:when test="@stdValue[.=’020’]"><td>Spanish Peseta EDP 724</td></xsl:when>
         <xsl:when test="@stdValue[.=’024’]"><td>New Kwanza</td></xsl:when>
         <xsl:when test="@stdValue[.=’031’]"><td>Azerbaijan Manat</td></xsl:when>
         <xsl:when test="@stdValue[.=’032’]"><td>Argentine Peso</td></xsl:when>
         <xsl:when test="@stdValue[.=’040’]"><td>Austrian Schilling</td></xsl:when>
         <xsl:when test="@stdValue[.=’044’]"><td>Bahamian Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’048’]"><td>Bahraini Dinar</td></xsl:when>
         <xsl:when test="@stdValue[.=’050’]"><td>Taka</td></xsl:when>
         <xsl:when test="@stdValue[.=’051’]"><td>Armenian Dram</td></xsl:when>
         <xsl:when test="@stdValue[.=’052’]"><td>Barbados Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’056’]"><td>Belgian Franc</td></xsl:when>
         <xsl:when test="@stdValue[.=’060’]"><td>Bermudian Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’064’]"><td>Indian Rupee, INR 356</td></xsl:when>
         <xsl:when test="@stdValue[.=’068’]"><td>Boliviano</td></xsl:when>
         <xsl:when test="@stdValue[.=’070’]"><td>Bosnian Convertible Mark, BAM 977</td></
xsl:when>
         <xsl:when test="@stdValue[.=’072’]"><td>Pula</td></xsl:when>
         <xsl:when test="@stdValue[.=’076’]"><td>Brazillian Real, BRL 986</td></xsl:when>
         <xsl:when test="@stdValue[.=’084’]"><td>Belize Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’096’]"><td>Brunei Dollar</td></xsl:when>
         <xsl:when test="@stdValue[.=’100’]"><td>Lev</td></xsl:when>
         <xsl:when test="@stdValue[.=’104’]"><td>Kyat</td></xsl:when>
         <xsl:when test="@stdValue[.=’108’]"><td>Burundi Franc</td></xsl:when>
         <xsl:when test="@stdValue[.=’112’]"><td>Belarussian Ruble</td></xsl:when>
         <xsl:when test="@stdValue[.=’116’]"><td>Riel</td></xsl:when>
         <xsl:when test="@stdValue[.=’124’]"><td>Canadian Dollar</td></xsl:when>
```

```
<xsl:when test="@stdValue[.=’132’]"><td>Escudo</td></xsl:when>
<xsl:when test="@stdValue[.=’136’]"><td>Cayman Is. Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’144’]"><td>Sri Lanka Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’152’]"><td>Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’156’]"><td>Yuan Renminbi</td></xsl:when>
<xsl:when test="@stdValue[.=’170’]"><td>Colombian Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’174’]"><td>Comoro Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’180’]"><td>New Zaire</td></xsl:when>
<xsl:when test="@stdValue[.=’188’]"><td>Costa Rican Colon</td></xsl:when>
<xsl:when test="@stdValue[.=’191’]"><td>Croatian Kuna</td></xsl:when>
<xsl:when test="@stdValue[.=’192’]"><td>Cuban Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’196’]"><td>Cyprus Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’200’]"><td>KORUNA</td></xsl:when>
<xsl:when test="@stdValue[.=’203’]"><td>Czech Koruna</td></xsl:when>
<xsl:when test="@stdValue[.=’208’]"><td>Danish Krone</td></xsl:when>
<xsl:when test="@stdValue[.=’214’]"><td>Dominican Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’218’]"><td>Sucre</td></xsl:when>
<xsl:when test="@stdValue[.=’222’]"><td>El Salvador Colon</td></xsl:when>
<xsl:when test="@stdValue[.=’226’]"><td>CFA Franc BEAC, XAF 950</td></xsl:when>
<xsl:when test="@stdValue[.=’230’]"><td>Ethiopian Birr</td></xsl:when>
<xsl:when test="@stdValue[.=’232’]"><td>Eritrean Nakfa</td></xsl:when>
<xsl:when test="@stdValue[.=’233’]"><td>Kroon</td></xsl:when>
<xsl:when test="@stdValue[.=’238’]"><td>Falkland Is. Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’242’]"><td>Fiji Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’246’]"><td>Markka</td></xsl:when>
<xsl:when test="@stdValue[.=’250’]"><td>French Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’262’]"><td>Djibouti Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’268’]"><td>Georgian Lari, GEL 981</td></xsl:when>
<xsl:when test="@stdValue[.=’270’]"><td>Dalasi</td></xsl:when>
<xsl:when test="@stdValue[.=’280’]"><td>Deutsche Mark</td></xsl:when>
<xsl:when test="@stdValue[.=’288’]"><td>Cedi</td></xsl:when>
<xsl:when test="@stdValue[.=’292’]"><td>Gibraltar Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’300’]"><td>Drachma</td></xsl:when>
<xsl:when test="@stdValue[.=’320’]"><td>Quetzal</td></xsl:when>
<xsl:when test="@stdValue[.=’324’]"><td>Guinea Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’328’]"><td>Guyana Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’332’]"><td>Gourde</td></xsl:when>
<xsl:when test="@stdValue[.=’340’]"><td>Lempira</td></xsl:when>
<xsl:when test="@stdValue[.=’344’]"><td>Hong Kong Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’348’]"><td>Forint</td></xsl:when>
<xsl:when test="@stdValue[.=’352’]"><td>ICELAND KRONA</td></xsl:when>
<xsl:when test="@stdValue[.=’356’]"><td>Indian Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’36 ’]"><td>Australian Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’360’]"><td>Rupiah</td></xsl:when>
<xsl:when test="@stdValue[.=’364’]"><td>Iranian Rial</td></xsl:when>
<xsl:when test="@stdValue[.=’365’]"><td>Iranian Airline Rate</td></xsl:when>
<xsl:when test="@stdValue[.=’368’]"><td>Iraqi Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’372’]"><td>Irish Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’376’]"><td>Shekel</td></xsl:when>
<xsl:when test="@stdValue[.=’380’]"><td>Italian Lira</td></xsl:when>
<xsl:when test="@stdValue[.=’388’]"><td>Jamaican Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’392’]"><td>Yen</td></xsl:when>
<xsl:when test="@stdValue[.=’398’]"><td>Tenge</td></xsl:when>
<xsl:when test="@stdValue[.=’400’]"><td>Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’404’]"><td>Kenyan Shilling</td></xsl:when>
<xsl:when test="@stdValue[.=’408’]"><td>Won</td></xsl:when>
```

```
<xsl:when test="@stdValue[.=’410’]"><td>Won</td></xsl:when>
<xsl:when test="@stdValue[.=’414’]"><td>Kuwaiti Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’417’]"><td>Som</td></xsl:when>
<xsl:when test="@stdValue[.=’418’]"><td>KIP</td></xsl:when>
<xsl:when test="@stdValue[.=’422’]"><td>Lebanese Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’426’]"><td>Rand, ZAR 710</td></xsl:when>
<xsl:when test="@stdValue[.=’428’]"><td>Latvian Lats</td></xsl:when>
<xsl:when test="@stdValue[.=’430’]"><td>Liberian Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’434’]"><td>Libyan Dinar</td></xsl:when>
<xsl:when test="@stdValue[.=’440’]"><td>Lithuanian Litas</td></xsl:when>
<xsl:when test="@stdValue[.=’442’]"><td>Luxembourg Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’446’]"><td>Pataca</td></xsl:when>
<xsl:when test="@stdValue[.=’450’]"><td>Malagasy Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’454’]"><td>Kwacha</td></xsl:when>
<xsl:when test="@stdValue[.=’458’]"><td>Malaysian Ringgit</td></xsl:when>
<xsl:when test="@stdValue[.=’462’]"><td>Rufiyaa</td></xsl:when>
<xsl:when test="@stdValue[.=’470’]"><td>Maltese Lira</td></xsl:when>
<xsl:when test="@stdValue[.=’478’]"><td>Ouguiya</td></xsl:when>
<xsl:when test="@stdValue[.=’480’]"><td>Mauritius Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’484’]"><td>Mexican Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’496’]"><td>Tugrik</td></xsl:when>
<xsl:when test="@stdValue[.=’498’]"><td>Moldovan Leu</td></xsl:when>
<xsl:when test="@stdValue[.=’504’]"><td>Moroccan Dirham</td></xsl:when>
<xsl:when test="@stdValue[.=’508’]"><td>Metical</td></xsl:when>
<xsl:when test="@stdValue[.=’512’]"><td>Rial Omani</td></xsl:when>
<xsl:when test="@stdValue[.=’516’]"><td>Namibia Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’524’]"><td>Nepalese Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’528’]"><td>Netherlands Guilder</td></xsl:when>
<xsl:when test="@stdValue[.=’532’]"><td>Nether. Antillian Guilder</td></xsl:when>
<xsl:when test="@stdValue[.=’533’]"><td>Aruban Guilder</td></xsl:when>
<xsl:when test="@stdValue[.=’548’]"><td>Vatu</td></xsl:when>
<xsl:when test="@stdValue[.=’554’]"><td>New Zealand Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’558’]"><td>Cordoba Oro</td></xsl:when>
<xsl:when test="@stdValue[.=’566’]"><td>Naira</td></xsl:when>
<xsl:when test="@stdValue[.=’578’]"><td>Norwegian Krone</td></xsl:when>
<xsl:when test="@stdValue[.=’586’]"><td>Pakistan Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’590’]"><td>Balboa</td></xsl:when>
<xsl:when test="@stdValue[.=’598’]"><td>Kina</td></xsl:when>
<xsl:when test="@stdValue[.=’600’]"><td>Guarani</td></xsl:when>
<xsl:when test="@stdValue[.=’604’]"><td>Nuevo Sol</td></xsl:when>
<xsl:when test="@stdValue[.=’608’]"><td>Philippine Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’616’]"><td>Polish New Zloty, PLN 985</td></xsl:when>
<xsl:when test="@stdValue[.=’620’]"><td>Portuguese Escudo</td></xsl:when>
<xsl:when test="@stdValue[.=’624’]"><td>Guinea-Bissau Peso</td></xsl:when>
<xsl:when test="@stdValue[.=’626’]"><td>Timor Escudo</td></xsl:when>
<xsl:when test="@stdValue[.=’634’]"><td>Qatari Rial</td></xsl:when>
<xsl:when test="@stdValue[.=’642’]"><td>Leu</td></xsl:when>
<xsl:when test="@stdValue[.=’643’]"><td>Russian Ruble</td></xsl:when>
<xsl:when test="@stdValue[.=’646’]"><td>Rwanda Franc</td></xsl:when>
<xsl:when test="@stdValue[.=’654’]"><td>St. Helena Pound</td></xsl:when>
<xsl:when test="@stdValue[.=’678’]"><td>Dobra</td></xsl:when>
<xsl:when test="@stdValue[.=’682’]"><td>Saudi Riyal</td></xsl:when>
<xsl:when test="@stdValue[.=’690’]"><td>Seychelles Rupee</td></xsl:when>
<xsl:when test="@stdValue[.=’694’]"><td>Leone</td></xsl:when>
<xsl:when test="@stdValue[.=’702’]"><td>Singapore Dollar</td></xsl:when>
<xsl:when test="@stdValue[.=’703’]"><td>Slovak Koruna</td></xsl:when>
```

```
        <xsl:when test="@stdValue[.=’704’]"><td>Dong</td></xsl:when>
        <xsl:when test="@stdValue[.=’705’]"><td>Tolar</td></xsl:when>
        <xsl:when test="@stdValue[.=’706’]"><td>Somali Shilling</td></xsl:when>
        <xsl:when test="@stdValue[.=’710’]"><td>Africa Rand</td></xsl:when>
        <xsl:when test="@stdValue[.=’716’]"><td>Zimbabwe Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’724’]"><td>Spanish Peseta</td></xsl:when>
        <xsl:when test="@stdValue[.=’736’]"><td>Sudanese Pound</td></xsl:when>
        <xsl:when test="@stdValue[.=’737’]"><td>Sudan Airline Rate</td></xsl:when>
        <xsl:when test="@stdValue[.=’740’]"><td>Surinam Guilder</td></xsl:when>
        <xsl:when test="@stdValue[.=’748’]"><td>Lilangeni</td></xsl:when>
        <xsl:when test="@stdValue[.=’752’]"><td>Swedish Krona</td></xsl:when>
        <xsl:when test="@stdValue[.=’756’]"><td>Swiss Franc</td></xsl:when>
        <xsl:when test="@stdValue[.=’760’]"><td>Syrian Pound</td></xsl:when>
        <xsl:when test="@stdValue[.=’762’]"><td>Tajik Ruble</td></xsl:when>
        <xsl:when test="@stdValue[.=’764’]"><td>Thailand Baht</td></xsl:when>
        <xsl:when test="@stdValue[.=’776’]"><td>Pa’anga</td></xsl:when>
        <xsl:when test="@stdValue[.=’780’]"><td>Trinidad and Tobago Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’784’]"><td>Dirham</td></xsl:when>
        <xsl:when test="@stdValue[.=’788’]"><td>Tunisian Dinar</td></xsl:when>
        <xsl:when test="@stdValue[.=’792’]"><td>Turkish Lira</td></xsl:when>
        <xsl:when test="@stdValue[.=’795’]"><td>Manat</td></xsl:when>
        <xsl:when test="@stdValue[.=’800’]"><td>Uganda Shilling</td></xsl:when>
        <xsl:when test="@stdValue[.=’804’]"><td>Ukranian Hryvnia, UAH 980</td></xsl:when>
        <xsl:when test="@stdValue[.=’807’]"><td>Denar</td></xsl:when>
        <xsl:when test="@stdValue[.=’810’]"><td>Russian Ruble</td></xsl:when>
        <xsl:when test="@stdValue[.=’818’]"><td>Egyptian Pound</td></xsl:when>
        <xsl:when test="@stdValue[.=’826’]"><td>Pound Sterling</td></xsl:when>
        <xsl:when test="@stdValue[.=’834’]"><td>Tanzanian Shilling</td></xsl:when>
        <xsl:when test="@stdValue[.=’840’]"><td>U.S. Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’858’]"><td>Uruguayo</td></xsl:when>
        <xsl:when test="@stdValue[.=’860’]"><td>Uzbekistan Sum</td></xsl:when>
        <xsl:when test="@stdValue[.=’862’]"><td>Bolivar</td></xsl:when>
        <xsl:when test="@stdValue[.=’882’]"><td>Tala</td></xsl:when>
        <xsl:when test="@stdValue[.=’886’]"><td>Yemeni Rial</td></xsl:when>
        <xsl:when test="@stdValue[.=’891’]"><td>Yugoslavian New Dinar</td></xsl:when>
        <xsl:when test="@stdValue[.=’894’]"><td>Zambian Kwacha</td></xsl:when>
        <xsl:when test="@stdValue[.=’090’]"><td>Solomon Is. Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’901’]"><td>New Taiwan Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’950’]"><td>CFA Franc BEAC</td></xsl:when>
        <xsl:when test="@stdValue[.=’951’]"><td>E. Caribbean Dollar</td></xsl:when>
        <xsl:when test="@stdValue[.=’952’]"><td>CFA Franc BCEAO</td></xsl:when>
        <xsl:when test="@stdValue[.=’953’]"><td>CFP Franc</td></xsl:when>
        <xsl:when test="@stdValue[.=’977’]"><td>Herzegovina Bosnian Convertible Mar</td></
xsl:when>
        <xsl:when test="@stdValue[.=’978’]"><td>euro</td></xsl:when>
        <xsl:when test="@stdValue[.=’980’]"><td>UKRAINIAN HRYVNIA</td></xsl:when>
        <xsl:when test="@stdValue[.=’981’]"><td>Georgian Lari</td></xsl:when>
        <xsl:when test="@stdValue[.=’985’]"><td>Polish New Zloty</td></xsl:when>
        <xsl:when test="@stdValue[.=’986’]"><td>Brazilian Real</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Currency
<xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="Country">
```

```
    <xsl:choose>
        <xsl:when test=".[.=’AD’]"><td valign="top">Andorra</td></xsl:when>
        <xsl:when test=".[.=’AE’]"><td valign="top">United Arab Emirates</td></xsl:when>
        <xsl:when test=".[.=’AF’]"><td valign="top">Afghanistan</td></xsl:when>
        <xsl:when test=".[.=’AG’]"><td valign="top">Antigua</td></xsl:when>
        <xsl:when test=".[.=’AL’]"><td valign="top">Albania</td></xsl:when>
        <xsl:when test=".[.=’AM’]"><td valign="top">Armenia</td></xsl:when>
        <xsl:when test=".[.=’AN’]"><td valign="top">Netherlands Antilles</td></xsl:when>
        <xsl:when test=".[.=’AO’]"><td valign="top">Angola</td></xsl:when>
        <xsl:when test=".[.=’AQ’]"><td valign="top">Antarctica</td></xsl:when>
        <xsl:when test=".[.=’AR’]"><td valign="top">Argentina</td></xsl:when>
        <xsl:when test=".[.=’AS’]"><td valign="top">American Samoa</td></xsl:when>
        <xsl:when test=".[.=’AT’]"><td valign="top">Austria</td></xsl:when>
        <xsl:when test=".[.=’AU’]"><td valign="top">Australia</td></xsl:when>
        <xsl:when test=".[.=’AZ’]"><td valign="top">Azerbaijan</td></xsl:when>
        <xsl:when test=".[.=’BA’]"><td valign="top">Bosnia-Herzegovina</td></xsl:when>
        <xsl:when test=".[.=’BB’]"><td valign="top">Barbados</td></xsl:when>
        <xsl:when test=".[.=’BD’]"><td valign="top">Bangladesh</td></xsl:when>
        <xsl:when test=".[.=’BE’]"><td valign="top">Belgium</td></xsl:when>
        <xsl:when test=".[.=’BG’]"><td valign="top">Bulgaria</td></xsl:when>
        <xsl:when test=".[.=’BH’]"><td valign="top">Bahrain</td></xsl:when>
        <xsl:when test=".[.=’BI’]"><td valign="top">Burundi</td></xsl:when>
        <xsl:when test=".[.=’BJ’]"><td valign="top">Benin</td></xsl:when>
        <xsl:when test=".[.=’BM’]"><td valign="top">Bermuda</td></xsl:when>
        <xsl:when test=".[.=’BN’]"><td valign="top">Brunei</td></xsl:when>
        <xsl:when test=".[.=’BO’]"><td valign="top">Bolivia</td></xsl:when>
        <xsl:when test=".[.=’BR’]"><td valign="top">Brazil</td></xsl:when>
        <xsl:when test=".[.=’BS’]"><td valign="top">Bahamas</td></xsl:when>
        <xsl:when test=".[.=’BT’]"><td valign="top">Bhutan</td></xsl:when>
        <xsl:when test=".[.=’BU’]"><td valign="top">Burma</td></xsl:when>
        <xsl:when test=".[.=’BV’]"><td valign="top">Bouvet Island</td></xsl:when>
        <xsl:when test=".[.=’BW’]"><td valign="top">Botswana</td></xsl:when>
        <xsl:when test=".[.=’BY’]"><td valign="top">Byelorussian Ssr (Bielorussia)</td></
xsl:when>
        <xsl:when test=".[.=’BZ’]"><td valign="top">Belize</td></xsl:when>
        <xsl:when test=".[.=’CA’]"><td valign="top">Canada</td></xsl:when>
        <xsl:when test=".[.=’CC’]"><td valign="top">Cocos Islands</td></xsl:when>
        <xsl:when test=".[.=’CF’]"><td valign="top">Central African Republic</td></
xsl:when>
        <xsl:when test=".[.=’CG’]"><td valign="top">Congo</td></xsl:when>
        <xsl:when test=".[.=’CH’]"><td valign="top">Switzerland</td></xsl:when>
        <xsl:when test=".[.=’CI’]"><td valign="top">Ivory Coast</td></xsl:when>
        <xsl:when test=".[.=’CK’]"><td valign="top">Cook Islands</td></xsl:when>
        <xsl:when test=".[.=’CL’]"><td valign="top">Chile</td></xsl:when>
        <xsl:when test=".[.=’CM’]"><td valign="top">United Republic of Cameroon</td></
xsl:when>
        <xsl:when test=".[.=’CN’]"><td valign="top">China</td></xsl:when>
        <xsl:when test=".[.=’CO’]"><td valign="top">Columbia</td></xsl:when>
        <xsl:when test=".[.=’CR’]"><td valign="top">Costa Rica</td></xsl:when>
        <xsl:when test=".[.=’CS’]"><td valign="top">Czechoslovakia</td></xsl:when>
        <xsl:when test=".[.=’CT’]"><td valign="top">Canton and Enderbury Islands</td></
xsl:when>
        <xsl:when test=".[.=’CU’]"><td valign="top">Cuba</td></xsl:when>
        <xsl:when test=".[.=’CV’]"><td valign="top">Cape Verde</td></xsl:when>
        <xsl:when test=".[.=’CX’]"><td valign="top">Christmas Island</td></xsl:when>
        <xsl:when test=".[.=’CY’]"><td valign="top">Cyprus</td></xsl:when>
```

```
         <xsl:when test=".[.=’CZ’]"><td valign="top">Czech</td></xsl:when>
         <xsl:when test=".[.=’DE’]"><td valign="top">Federal Republic of Germany</td></
xsl:when>
         <xsl:when test=".[.=’DJ’]"><td valign="top">Djibouti</td></xsl:when>
         <xsl:when test=".[.=’DK’]"><td valign="top">Denmark</td></xsl:when>
         <xsl:when test=".[.=’DM’]"><td valign="top">Dominica</td></xsl:when>
         <xsl:when test=".[.=’DO’]"><td valign="top">Dominican Republic</td></xsl:when>
         <xsl:when test=".[.=’DZ’]"><td valign="top">Algeria</td></xsl:when>
         <xsl:when test=".[.=’EC’]"><td valign="top">Ecuador</td></xsl:when>
         <xsl:when test=".[.=’EG’]"><td valign="top">Egypt</td></xsl:when>
         <xsl:when test=".[.=’EH’]"><td valign="top">Western Sahara</td></xsl:when>
         <xsl:when test=".[.=’ES’]"><td valign="top">Spain</td></xsl:when>
         <xsl:when test=".[.=’ET’]"><td valign="top">Ethiopia</td></xsl:when>
         <xsl:when test=".[.=’FI’]"><td valign="top">Finland</td></xsl:when>
         <xsl:when test=".[.=’FJ’]"><td valign="top">Fiji</td></xsl:when>
         <xsl:when test=".[.=’FK’]"><td valign="top">Falkland Islands</td></xsl:when>
         <xsl:when test=".[.=’FO’]"><td valign="top">Faeroe Islands</td></xsl:when>
         <xsl:when test=".[.=’FR’]"><td valign="top">France</td></xsl:when>
         <xsl:when test=".[.=’GA’]"><td valign="top">Gabon</td></xsl:when>
         <xsl:when test=".[.=’GB’]"><td valign="top">United Kingdom</td></xsl:when>
         <xsl:when test=".[.=’GD’]"><td valign="top">Grenada</td></xsl:when>
         <xsl:when test=".[.=’GF’]"><td valign="top">French Guiana</td></xsl:when>
         <xsl:when test=".[.=’GG’]"><td valign="top">Georgia</td></xsl:when>
         <xsl:when test=".[.=’GH’]"><td valign="top">Ghana</td></xsl:when>
         <xsl:when test=".[.=’GI’]"><td valign="top">Gibraltar</td></xsl:when>
         <xsl:when test=".[.=’GL’]"><td valign="top">Greenland</td></xsl:when>
         <xsl:when test=".[.=’GM’]"><td valign="top">Gambia</td></xsl:when>
         <xsl:when test=".[.=’GN’]"><td valign="top">Guinea</td></xsl:when>
         <xsl:when test=".[.=’GP’]"><td valign="top">Guadeloupe</td></xsl:when>
         <xsl:when test=".[.=’GQ’]"><td valign="top">Equatorial Guinea</td></xsl:when>
         <xsl:when test=".[.=’GR’]"><td valign="top">Greece</td></xsl:when>
         <xsl:when test=".[.=’GT’]"><td valign="top">Guatemala</td></xsl:when>
         <xsl:when test=".[.=’GU’]"><td valign="top">Guam</td></xsl:when>
         <xsl:when test=".[.=’GW’]"><td valign="top">Guinea-Bisseu</td></xsl:when>
         <xsl:when test=".[.=’GY’]"><td valign="top">Guyana</td></xsl:when>
         <xsl:when test=".[.=’HK’]"><td valign="top">Hong Kong</td></xsl:when>
         <xsl:when test=".[.=’HM’]"><td valign="top">Heard and Mc Donald Islands</td></
xsl:when>
         <xsl:when test=".[.=’HN’]"><td valign="top">Honduras</td></xsl:when>
         <xsl:when test=".[.=’HR’]"><td valign="top">Croatia</td></xsl:when>
         <xsl:when test=".[.=’HT’]"><td valign="top">Haiti</td></xsl:when>
         <xsl:when test=".[.=’HU’]"><td valign="top">Hungary</td></xsl:when>
         <xsl:when test=".[.=’HV’]"><td valign="top">Upper Volta</td></xsl:when>
         <xsl:when test=".[.=’ID’]"><td valign="top">Indonesia</td></xsl:when>
         <xsl:when test=".[.=’IE’]"><td valign="top">Ireland</td></xsl:when>
         <xsl:when test=".[.=’IL’]"><td valign="top">Israel</td></xsl:when>
         <xsl:when test=".[.=’IN’]"><td valign="top">India</td></xsl:when>
         <xsl:when test=".[.=’IO’]"><td valign="top">British Indian Ocean Territory</td></
xsl:when>
         <xsl:when test=".[.=’IQ’]"><td valign="top">Iraq</td></xsl:when>
         <xsl:when test=".[.=’IR’]"><td valign="top">Iran</td></xsl:when>
         <xsl:when test=".[.=’IS’]"><td valign="top">Iceland</td></xsl:when>
         <xsl:when test=".[.=’IT’]"><td valign="top">Italy</td></xsl:when>
         <xsl:when test=".[.=’JM’]"><td valign="top">Jamaica</td></xsl:when>
         <xsl:when test=".[.=’JO’]"><td valign="top">Jordan</td></xsl:when>
         <xsl:when test=".[.=’JP’]"><td valign="top">Japan</td></xsl:when>
         <xsl:when test=".[.=’JT’]"><td valign="top">Johnston Island</td></xsl:when>
         <xsl:when test=".[.=’KE’]"><td valign="top">Kenya</td></xsl:when>
         <xsl:when test=".[.=’KG’]"><td valign="top">Kyrgyzstan (Kirgistan)</td></
xsl:when>
         <xsl:when test=".[.=’KH’]"><td valign="top">Democratic Kampuchea</td></xsl:when>
         <xsl:when test=".[.=’KI’]"><td valign="top">Kiribati</td></xsl:when>
         <xsl:when test=".[.=’KK’]"><td valign="top">Kazakhstan</td></xsl:when>
         <xsl:when test=".[.=’KM’]"><td valign="top">Comoros</td></xsl:when>
         <xsl:when test=".[.=’KN’]"><td valign="top">St. Kitts Nevis Anguilla</td></
xsl:when>
         <xsl:when test=".[.=’KP’]"><td valign="top">Democratic People’s Republic of Ko-
rea</td></xsl:when>
         <xsl:when test=".[.=’KR’]"><td valign="top">Republic of Korea</td></xsl:when>
         <xsl:when test=".[.=’KW’]"><td valign="top">Kuwait</td></xsl:when>
         <xsl:when test=".[.=’KY’]"><td valign="top">Cayman Islands</td></xsl:when>
         <xsl:when test=".[.=’LA’]"><td valign="top">Lao People’s Democratic Republic</
td></xsl:when>
         <xsl:when test=".[.=’LB’]"><td valign="top">Lebanon</td></xsl:when>
         <xsl:when test=".[.=’LC’]"><td valign="top">Saint Lucia</td></xsl:when>
         <xsl:when test=".[.=’LI’]"><td valign="top">Liechtenstein</td></xsl:when>
         <xsl:when test=".[.=’LK’]"><td valign="top">Sri Lanka</td></xsl:when>
         <xsl:when test=".[.=’LR’]"><td valign="top">Liberia</td></xsl:when>
         <xsl:when test=".[.=’LS’]"><td valign="top">Lesotho</td></xsl:when>
         <xsl:when test=".[.=’LT’]"><td valign="top">Lithuania</td></xsl:when>
         <xsl:when test=".[.=’LU’]"><td valign="top">Luxembourg</td></xsl:when>
         <xsl:when test=".[.=’LV’]"><td valign="top">Latvia</td></xsl:when>
         <xsl:when test=".[.=’LY’]"><td valign="top">Libyan Arab Jamahiriya</td></xsl:when>
         <xsl:when test=".[.=’MA’]"><td valign="top">Morrocco</td></xsl:when>
         <xsl:when test=".[.=’MC’]"><td valign="top">Monaco</td></xsl:when>
         <xsl:when test=".[.=’MD’]"><td valign="top">Moldova</td></xsl:when>
         <xsl:when test=".[.=’MG’]"><td valign="top">Madagascar</td></xsl:when>
         <xsl:when test=".[.=’MI’]"><td valign="top">Midway Islands</td></xsl:when>
         <xsl:when test=".[.=’ML’]"><td valign="top">Mali</td></xsl:when>
         <xsl:when test=".[.=’MN’]"><td valign="top">Mongolia</td></xsl:when>
         <xsl:when test=".[.=’MO’]"><td valign="top">Macau</td></xsl:when>
         <xsl:when test=".[.=’MQ’]"><td valign="top">Martinique</td></xsl:when>
         <xsl:when test=".[.=’MR’]"><td valign="top">Mauritania</td></xsl:when>
         <xsl:when test=".[.=’MS’]"><td valign="top">Montserrat</td></xsl:when>
         <xsl:when test=".[.=’MT’]"><td valign="top">Malta</td></xsl:when>
         <xsl:when test=".[.=’MU’]"><td valign="top">Mauritius</td></xsl:when>
         <xsl:when test=".[.=’MV’]"><td valign="top">Maldives</td></xsl:when>
         <xsl:when test=".[.=’MW’]"><td valign="top">Malawi</td></xsl:when>
         <xsl:when test=".[.=’MX’]"><td valign="top">Mexico</td></xsl:when>
         <xsl:when test=".[.=’MY’]"><td valign="top">Malasia</td></xsl:when>
         <xsl:when test=".[.=’MZ’]"><td valign="top">Mozambique</td></xsl:when>
         <xsl:when test=".[.=’NA’]"><td valign="top">Namibia</td></xsl:when>
         <xsl:when test=".[.=’NC’]"><td valign="top">New Calidonia</td></xsl:when>
         <xsl:when test=".[.=’NE’]"><td valign="top">Niger</td></xsl:when>
         <xsl:when test=".[.=’NF’]"><td valign="top">Norfolk Island</td></xsl:when>
         <xsl:when test=".[.=’NG’]"><td valign="top">Nigeria</td></xsl:when>
         <xsl:when test=".[.=’NI’]"><td valign="top">Nicaragua</td></xsl:when>
         <xsl:when test=".[.=’NL’]"><td valign="top">Netherlands</td></xsl:when>
         <xsl:when test=".[.=’NO’]"><td valign="top">Norway</td></xsl:when>
         <xsl:when test=".[.=’NP’]"><td valign="top">Napal</td></xsl:when>
         <xsl:when test=".[.=’NQ’]"><td valign="top">Dronning Maud Land</td></xsl:when>
         <xsl:when test=".[.=’NR’]"><td valign="top">Nauru</td></xsl:when>
         <xsl:when test=".[.=’NT’]"><td valign="top">Neutral Zone</td></xsl:when>
         <xsl:when test=".[.=’NU’]"><td valign="top">Niue</td></xsl:when>
         <xsl:when test=".[.=’NZ’]"><td valign="top">New Zealand</td></xsl:when>
         <xsl:when test=".[.=’OM’]"><td valign="top">Oman</td></xsl:when>
         <xsl:when test=".[.=’PA’]"><td valign="top">Panama</td></xsl:when>
         <xsl:when test=".[.=’PC’]"><td valign="top">Pacific Islands</td></xsl:when>
         <xsl:when test=".[.=’PE’]"><td valign="top">Peru</td></xsl:when>
         <xsl:when test=".[.=’PF’]"><td valign="top">French Polynesia</td></xsl:when>
         <xsl:when test=".[.=’PG’]"><td valign="top">Papua New Guinea</td></xsl:when>
         <xsl:when test=".[.=’PH’]"><td valign="top">Phillipines</td></xsl:when>
         <xsl:when test=".[.=’PK’]"><td valign="top">Pakistan</td></xsl:when>
         <xsl:when test=".[.=’PL’]"><td valign="top">Poland</td></xsl:when>
         <xsl:when test=".[.=’PM’]"><td valign="top">St. Pierre and Miquelon</td></
xsl:when>
         <xsl:when test=".[.=’PN’]"><td valign="top">Pitcairn Island</td></xsl:when>
         <xsl:when test=".[.=’PR’]"><td valign="top">Puerto Rico</td></xsl:when>
         <xsl:when test=".[.=’PT’]"><td valign="top">Portugal</td></xsl:when>
         <xsl:when test=".[.=’PU’]"><td valign="top">United States Miscellaneous Pacific
Islands</td></xsl:when>
         <xsl:when test=".[.=’PY’]"><td valign="top">Paraguay</td></xsl:when>
         <xsl:when test=".[.=’QA’]"><td valign="top">Qatar</td></xsl:when>
         <xsl:when test=".[.=’RE’]"><td valign="top">Reunion</td></xsl:when>
         <xsl:when test=".[.=’RO’]"><td valign="top">Romania</td></xsl:when>
         <xsl:when test=".[.=’RU’]"><td valign="top">Russia</td></xsl:when>
         <xsl:when test=".[.=’RW’]"><td valign="top">Rwanda</td></xsl:when>
         <xsl:when test=".[.=’SA’]"><td valign="top">Saudi Arabia</td></xsl:when>
         <xsl:when test=".[.=’SB’]"><td valign="top">Solomon Islands</td></xsl:when>
         <xsl:when test=".[.=’SC’]"><td valign="top">Seychelles</td></xsl:when>
         <xsl:when test=".[.=’SD’]"><td valign="top">Sudan</td></xsl:when>
         <xsl:when test=".[.=’SE’]"><td valign="top">Sweden</td></xsl:when>
         <xsl:when test=".[.=’SG’]"><td valign="top">Singapore</td></xsl:when>
         <xsl:when test=".[.=’SH’]"><td valign="top">St. Helena</td></xsl:when>
         <xsl:when test=".[.=’SI’]"><td valign="top">Slovenia</td></xsl:when>
         <xsl:when test=".[.=’SJ’]"><td valign="top">Svalbard and Jan Mayen Islands</td></
xsl:when>
         <xsl:when test=".[.=’SL’]"><td valign="top">Sierra Leone</td></xsl:when>
         <xsl:when test=".[.=’SM’]"><td valign="top">San Marino</td></xsl:when>
         <xsl:when test=".[.=’SN’]"><td valign="top">Senegal</td></xsl:when>
         <xsl:when test=".[.=’SO’]"><td valign="top">Somalia</td></xsl:when>
         <xsl:when test=".[.=’SQ’]"><td valign="top">Slovakia</td></xsl:when>
         <xsl:when test=".[.=’SR’]"><td valign="top">Suriname</td></xsl:when>
         <xsl:when test=".[.=’ST’]"><td valign="top">Sao Tome and Principe</td></xsl:when>
         <xsl:when test=".[.=’SU’]"><td valign="top">USSR</td></xsl:when>
         <xsl:when test=".[.=’SV’]"><td valign="top">El Salvador</td></xsl:when>
         <xsl:when test=".[.=’SY’]"><td valign="top">Syran Arab Republic</td></xsl:when>
         <xsl:when test=".[.=’SZ’]"><td valign="top">Swaziland</td></xsl:when>
         <xsl:when test=".[.=’TC’]"><td valign="top">Turks and Caicos Islands</td></
xsl:when>
         <xsl:when test=".[.=’TD’]"><td valign="top">Chad</td></xsl:when>
         <xsl:when test=".[.=’TG’]"><td valign="top">Togo</td></xsl:when>
         <xsl:when test=".[.=’TH’]"><td valign="top">Thailand</td></xsl:when>
         <xsl:when test=".[.=’TJ’]"><td valign="top">Tajikistan</td></xsl:when>
         <xsl:when test=".[.=’TK’]"><td valign="top">Tokelau</td></xsl:when>
         <xsl:when test=".[.=’TM’]"><td valign="top">Turkmenistan</td></xsl:when>
         <xsl:when test=".[.=’TN’]"><td valign="top">Tunisia</td></xsl:when>
         <xsl:when test=".[.=’TO’]"><td valign="top">Tonga</td></xsl:when>
```

```
        <xsl:when test=".[.=’TP’]"><td valign="top">East Timor</td></xsl:when>
        <xsl:when test=".[.=’TR’]"><td valign="top">Turkey</td></xsl:when>
        <xsl:when test=".[.=’TT’]"><td valign="top">Trinidad and Tobago</td></xsl:when>
        <xsl:when test=".[.=’TV’]"><td valign="top">Tuvalu</td></xsl:when>
        <xsl:when test=".[.=’TW’]"><td valign="top">Taiwan</td></xsl:when>
        <xsl:when test=".[.=’TZ’]"><td valign="top">United Republic of Tanzania</td></
xsl:when>
        <xsl:when test=".[.=’UA’]"><td valign="top">Ukrainian SSR</td></xsl:when>
        <xsl:when test=".[.=’UG’]"><td valign="top">Uganda</td></xsl:when>
        <xsl:when test=".[.=’US’]"><td valign="top">United States</td></xsl:when>
        <xsl:when test=".[.=’UY’]"><td valign="top">Uruguay</td></xsl:when>
        <xsl:when test=".[.=’UZ’]"><td valign="top">Uzbekistan</td></xsl:when>
        <xsl:when test=".[.=’VA’]"><td valign="top">Vatican City State</td></xsl:when>
        <xsl:when test=".[.=’VC’]"><td valign="top">Saint Vincent and the Grenadines</
td></xsl:when>
        <xsl:when test=".[.=’VE’]"><td valign="top">Venezuela</td></xsl:when>
        <xsl:when test=".[.=’VG’]"><td valign="top">British Virgin Islands</td></xsl:when>
        <xsl:when test=".[.=’VI’]"><td valign="top">Unites States Virgin Islands</td></
xsl:when>
        <xsl:when test=".[.=’VN’]"><td valign="top">Vietnam</td></xsl:when>
        <xsl:when test=".[.=’VU’]"><td valign="top">Vanuatu</td></xsl:when>
        <xsl:when test=".[.=’WF’]"><td valign="top">Wallis and Futuma Islands</td></
xsl:when>
        <xsl:when test=".[.=’WK’]"><td valign="top">Wake Island</td></xsl:when>
        <xsl:when test=".[.=’WS’]"><td valign="top">Samoa</td></xsl:when>
        <xsl:when test=".[.=’YD’]"><td valign="top">Democratic Yemen</td></xsl:when>
        <xsl:when test=".[.=’YE’]"><td valign="top">Yemen</td></xsl:when>
        <xsl:when test=".[.=’YU’]"><td valign="top">Yugoslavia</td></xsl:when>
        <xsl:when test=".[.=’ZA’]"><td valign="top">South Africa</td></xsl:when>
        <xsl:when test=".[.=’ZM’]"><td valign="top">Zambia</td></xsl:when>
        <xsl:when test=".[.=’ZR’]"><td valign="top">Zaire</td></xsl:when>
        <xsl:when test=".[.=’ZW’]"><td valign="top">Zimbabwe</td></xsl:when>
        <xsl:when test=".[.=’UK’]"><td valign="top">United Kingdom</td></xsl:when>
        <xsl:otherwise><td valign="top" bgcolor="red"><font color="silver"><marquee>Un-
known Country <xsl:value-of select="." /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="InvoiceDate">
    <td><xsl:eval>DateFormat(nodeTypedValue)</xsl:eval><xsl:eval>TimeFormat(nodeTypedVal-
ue)</xsl:eval></td>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="DiscountTreatment">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’UN’]"><td>Line item net unit price</td></xsl:when>
        <xsl:when test="@stdValue[.=’UG’]"><td>Line item gross unit price</td></xsl:when>
        <xsl:when test="@stdValue[.=’TN’]"><td>Line item subtotal net amount</td></
xsl:when>
        <xsl:when test="@stdValue[.=’TG’]"><td>Line item subtotal gross amount</td></
xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Discount
Treatment <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="TaxTreatment">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’NIL’]"><td>Line item amounts are net amounts, and tax is calculated at invoice level</td></xsl:when>
        <xsl:when test="@stdValue[.=’GIL’]"><td>Line item amounts are gross amounts, and tax is calculated at invoice level</td></xsl:when>
        <xsl:when test="@stdValue[.=’NLL’]"><td>Line item amounts are net amounts, and tax is calculated at line level</td></xsl:when>
        <xsl:when test="@stdValue[.=’GLL’]"><td>Line item amounts are gross amounts, and tax is calculated at line level</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Tax Treatment <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="TimeRelation">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’1’]"><td>Reference date</td></xsl:when>
        <xsl:when test="@stdValue[.=’2’]"><td>Before reference date</td></xsl:when>
        <xsl:when test="@stdValue[.=’3’]"><td>After reference date</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Time Relation <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="TypeOfPeriod">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’CD’]"><td>Calendar day</td></xsl:when>
        <xsl:when test="@stdValue[.=’DW’]"><td>Working day</td></xsl:when>
        <xsl:when test="@stdValue[.=’M’]"><td>Month</td></xsl:when>
        <xsl:when test="@stdValue[.=’W’]"><td>Week</td></xsl:when>
        <xsl:when test="@stdValue[.=’Y’]"><td>Year</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Type of Period <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="PaymentTermType">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’1’]"><td>Basic</td></xsl:when>
        <xsl:when test="@stdValue[.=’3’]"><td>Fixed date</td></xsl:when>
        <xsl:when test="@stdValue[.=’8’]"><td>Basic discount</td></xsl:when>
        <xsl:when test="@stdValue[.=’10’]"><td>Instant</td></xsl:when>
        <xsl:when test="@stdValue[.=’22’]"><td>Discount</td></xsl:when>
        <xsl:when test="@stdValue[.=’OTHER’]"><td><xsl:value-of select="." /></td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Payment Term Type <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
```

```
</xsl:template>

<xsl:template match="CardType">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’VS’]"><td>Visa</td></xsl:when>
        <xsl:when test="@stdValue[.=’AMEX’]"><td>American Express</td></xsl:when>
        <xsl:when test="@stdValue[.=’MC’]"><td>Mastercard</td></xsl:when>
        <xsl:when test="@stdValue[.=’DINERS’]"><td>Diners Card</td></xsl:when>
        <xsl:when test="@stdValue[.=’DSCVR’]"><td>Discover</td></xsl:when>
        <xsl:when test="@stdValue[.=’OTHER’]"><td>><xsl:value-of select="." /></td></
xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Payment
Term Type <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="Date">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’STRT’]"><td>Start Date/Time</td></xsl:when>
        <xsl:when test="@stdValue[.=’END’]"><td>End Date/Time</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Date Type
<xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="PartNumDetail">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’BP’]"><td>Buyers Part No</td></xsl:when>
        <xsl:when test="@stdValue[.=’VP’]"><td>Vendors Part No</td></xsl:when>
        <xsl:when test="@stdValue[.=’CC’]"><td>Industry commodity code</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Part Number
Type <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="UnitOfMeasure">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’EA’]"><td>EA</td></xsl:when>
        <xsl:when test="@stdValue[.=’DY’]"><td>DY</td></xsl:when>
        <xsl:when test="@stdValue[.=’C’]"><td>C</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Unit of
Measure <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="TaxFunction">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’5’]"><td>Customs duty</td></xsl:when>
        <xsl:when test="@stdValue[.=’7’]"><td>Tax</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Tax Func-
tion <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="TaxType">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’VAT’]"><td>Value Added Tax</td></xsl:when>
        <xsl:when test="@stdValue[.=’GST’]"><td>Goods and Services Tax</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Tax Type 
<xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="TaxCategory">
    <xsl:choose>
      <xsl:when test="@stdValue[.=’A’]"><td align="right">Mixed</td></xsl:when>
        <xsl:when test="@stdValue[.=’E’]"><td align="right">Exempt</td></xsl:when>
        <xsl:when test="@stdValue[.=’G’]"><td align="right">Free export item</td></
xsl:when>
        <xsl:when test="@stdValue[.=’S’]"><td align="right">Standard</td></xsl:when>
        <xsl:when test="@stdValue[.=’Z’]"><td align="right">Zero</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Tax Catego-
ry <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="SpecialCond">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’6’]"><td>Subject to bonus</td></xsl:when>
        <xsl:when test="@stdValue[.=’7’]"><td>Subject to commission</td></xsl:when>
        <xsl:when test="@stdValue[.=’11’]"><td>Price includes excise</td></xsl:when>
        <xsl:when test="@stdValue[.=’12’]"><td>Price includes tax</td></xsl:when>
        <xsl:when test="@stdValue[.=’18’]"><td>Item subject to national export restric-
tions</td></xsl:when>
        <xsl:when test="@stdValue[.=’97’]"><td>Promotional price</td></xsl:when>
        <xsl:when test="@stdValue[.=’94’]"><td>Service</td></xsl:when>
        <xsl:when test="@stdValue[.=’103’]"><td>Loan</td></xsl:when>
        <xsl:when test="@stdValue[.=’104’]"><td>Rental</td></xsl:when>
        <xsl:when test="@stdValue[.=’105’]"><td>Processing</td></xsl:when>
        <xsl:when test="@stdValue[.=’106’]"><td>Exchange</td></xsl:when>
        <xsl:when test="@stdValue[.=’140’]"><td>Return of goods</td></xsl:when>
        <xsl:when test="@stdValue[.=’OTHER’]"><td><xsl:value-of select="." /></td></
xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Special 
Condition <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="PaymentMean">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’10’]"><td>Cash</td></xsl:when>
        <xsl:when test="@stdValue[.=’20’]"><td>Cheque</td></xsl:when>
        <xsl:when test="@stdValue[.=’30’]"><td>Credit transfer</td></xsl:when>
        <xsl:when test="@stdValue[.=’ZZZ’]"><td>Credit/Debit card</td></xsl:when>
        <xsl:when test="@stdValue[.=’OTHER’]"><td><xsl:value-of select="." /></td></
xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Payment
Method<xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="RefDate">
    <xsl:choose>
        <xsl:when test="@stdValue[.=’5’]"><td>Date of invoice</td></xsl:when>
        <xsl:when test="@stdValue[.=’9’]"><td> Date invoice received</td></xsl:when>
        <xsl:when test="@stdValue[.=’21’]"><td>Goods received by buyer</td></xsl:when>
        <xsl:when test="@stdValue[.=’26’]"><td>Date of arrival of transport</td></
xsl:when>
        <xsl:when test="@stdValue[.=’81’]"><td>Date of shipment (as evidenced by transport
documentation)</td></xsl:when>
        <xsl:when test="@stdValue[.=’82’]"><td>Date payment due</td></xsl:when>
        <xsl:otherwise><td bgcolor="red"><font color="silver"><marquee>Unknown Reference
Date <xsl:value-of select="@stdValue" /></marquee></font></td></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

<xsl:template match="Ref[@stdValue=’STOP’]">
    <xsl:choose>
        <xsl:when test=".[.=’O’]">STOPOVEROK</xsl:when>
        <xsl:when test=".[.=’X’]">STOPOVERNOK</xsl:when>
        <xsl:otherwise></xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates />
</xsl:template>

</xsl:stylesheet>
```