OrderMate API

Menu & Sales integration (Online Ordering)

API Revision:	**14**
Minimum OrderMate Version Required:	**2.8.2 – Filewatcher Interface** 2.8.10 – Webservice Interface 2.8.11 – ETA Webservice 2.8.15m – tablesversion Method 2.8.16 – Account Comments 2.8.22 – Price Adds in Combos 2.8.25 – Loyalty Point and Customer Info 2.8.25 – Examine order 2.8.26 – HTTP Response Migration
Changes in this version:	400 Bad Request responses now include a detailed error message.

Table of Contents

Menu & Sales integration (Online Ordering)

Overview

The OrderMate Menu and Sales Integration provides a way for websites to retrieve menu and pricelist data from the OrderMate system to provide content for a website, and a way for websites to enter an internet order into the OrderMate Point of Sale system to be processed. The integration specification is essentially an XML schema that describes the accepted import and export format of XML files. The sales import references back to ID values of items exported in the menu export.

Integration Type

There are two types of integration available for online orders, a file watcher and a web service integration, both of these are outlined in detail below.

The filewatcher integration is no longer being actively developed, but is being supported for existing users. All new functionality is added to the webservice integration. See below for a list of features:

	Filewatcher	Webservice
Receive Orders on local machine	✓	✓
Receive orders from a remote server		✓
Error feedback	✓	✓
Order status feedback		✓
Order prep time feedback		✓
Publish Menu		✓

ServerMate File Watcher Integration

Online Orders

When using the file watcher integration, the OrderMate POS server (ServerMate) will monitor a directory on the local machine for online orders.

When ServerMate detects an online order, it is processed and then moved into a /success or /failed sub-directory.

If an order is moved to the failed directory, it will be accompanied by a .err file which will contain an error code used for troubleshooting purposes.

When using the file watcher integration, it is the responsibility of the online integrator to develop a windows application that will run on the OrderMate POS server and download the order XML. The application developed should also monitor the success and failed directories to determine whether or not the order has been successfully accepted by the POS.

Menu

The file watcher integration does not offer a way of automatically supplying the menu. The menu is manually exported from OfficeMate (The OrderMate POS administration software) by the POS user/restaurant owner.

Because online orders must contain items that exist within the POS system, it is a requirement when integrating using the file watcher method that you build an import function that will facilitate the automatic importing of an OrderMate menu XML file.

ServerMate Webservice Integration

The webservice integration method allows the submitting of online orders, it also provides the menu.

In order to use this integration method the ServerMate webservice must be enabled and the end consumer must have access to the webservice (via the URL and port setup in OfficeMate).

An example of the URL path if at local host would be:

http://localhost:8090/integration/onlineorder/

Each of the available webservice methods are outlined below with an example of the response.

~~All methods must add the username and password as parameters to the url in order to access the webservice.~~
All methods can currently accept the username and password in the url, however the preferred method is to place the username and password within the headers.

Parameter	**Description**	**Mandatory~~/~~Optional**
username	The username to access the webservice.	Mandatory
password	The password to access the webservice.	Mandatory

menuversion Method (HTTP Get)

This method returns the current version number of the menu. This method can be used to indicate when the menu has been changed i.e. if the consumer records the menu version when the menu was exported, then they can compare this against the current menu version.

Example URL

http://localhost:8090/integration/onlineorder/menuversion?user=test&password=test

Example Response

```
<ns2:OnlineOrderResult resultDesc="Successful" resultCode="200"
menuVersion="1175"/>
```

tablesversion Method (HTTP Get)

This method returns the current version number of the site's table setup. This method can be used to indicate when the table setup has been changed (such as by tables being moved, or table groups being created). If the consumer records the tables version when the menu was exported, then they can compare this against the current tables version.

Example URL

http://localhost:8090/integration/onlineorder/tablesversion?user=test&password=test

Example Response

```
<ns2:OnlineOrderResult resultDesc="Successful" resultCode="200"
tablesVersion="1175"/>
```

order Method (HTTP Post)

This method can be used to process an online order aka OrderMateSale. The OrderMateSale order should be supplied as a HTTP Entity within the post. The OrderMateSale xml is in the exact same format as the file that would be used with the file watcher implementation. The given example in the Sample OrderMateSale section is used in the example below.

Example URL

http://localhost:8090/integration/onlineorder/order?user=test&password=test

Example Response

The example below shows the successful response. However should there have been any errors with processing the orders the resultCode would be 200 and the OnlineOrderResult returned would contain the OnlineErrorResultList element that will give greater details as to the exact cause of the error.

```
<ns2:OnlineOrderResult resultDesc="Successful" resultCode="200" orderID="1"/>
   <ns2:Menu label="TableOrder" ID="1">
      <MenuSection label="Side Dishes" ID="30">
            ........
      </MenuSection>
   </ns2:Menu>
</ns2:OnlineOrderResult>
```

examine Method (HTTP Post)

This method can be used to examine an online order before it is processed. The OrderMateSale order is supplied as a HTTP Entity within the post. The format is the same as for ordering, however the order does not need to be complete: The 'examine' service can check the order and return further details about it before it is committed.
For example, a customer card may be swiped and passed through. If the customer is recognized, the customer's details is returned with the resulting response.
As another example, items may be examined and returned with loyalty point information attached.
The 'examine' webservice is a 'best-effort' service, in that it attempts to create the account as if it were to be posted to the 'order' webservice, only an account will not be created or amended, and further information relating to the order is returned to the caller.
A successful examine implies, but does not guarantee, a successful subsequent order. This is because the order routine actually purchases items, consumes stock, adds / consumes customer points, locks accounts, etc. and any of these may cause an order to fail.
OrderMate Reserves the right to add functionality and parameters to the returned sale in the future to provide more information, such as upsell suggestions, item availability, etc.

Example URL

http://localhost:8090/integration/onlineorder/examine?user=test&password=test

Example Success Response

The example below shows the successful response. However should there have been any errors with processing the orders the HTTP response code would be 400 or 500. See the example error response for the details that are returned in the response entity.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OnlineOrderResult
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/onlineorder"
xmlns:ns3="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
xmlns:ns4="http://www.ordermate.com.au/xmlintegration/public/menu">
<ns2:Order isPaid="false" isDelivery="false" orderDate="2015-10-
13T10:47:00.992+11:00" totalPrice="19.30" amountPaid="0.00" open="false">
<customer ID="23">
      <name title="Mr" firstName="Bill" lastName="Mason"/>
      <address houseNumber="" streetName="" streetType="" suburb="" state=""
postCode=""/>
      <email>bill@masonite.com</email>
```

```
        <primaryPhone>03</primaryPhone>
        <secondaryPhone>03</secondaryPhone>
        <comments>Bob is a frequent customer</comments>
        <currentPoints>0</currentPoints>
        <customerCard>%bill0573648</customerCard>
</customer>
<LineItem>
<SalesItem Qty="1" course="0" redeemPoints="10" earnPoints="1" label="Corona
(Bottle)">
        <MenuUnitDesc unitSalePrice="5.80" portion="Whole" ID="469" label="Corona
(Whole Bottle)">
        <MenuPriceRef menuID="1" name="TableOrder"/>
        </MenuUnitDesc>
</SalesItem>
</LineItem>
                ...
        </ns2:Order>
</ns2:OnlineOrderResult>
```

In the above example, Bill's card of %bill0573648 was passed through, and some items ordered, and the returned response is the sale with extra contextual information attached.

Example Error Response

In the event of a 400 Bad Request response, the response entity will be a JSON object containing three main objects, described below.

Key	Explanation	Example
message	A detailed error message describing the nature of the error. This is not end user-friendly, but is intended to assist integrators and tech support with resolving order issues.	`"Cannot import component Wings Pack(Serve), min/max on options are not satisfied: Dipping Sauce(Serve) min=1 max=2[],"`
details	An array of JSON objects containing key value pairs of IDs, details that are invalid or parameters that have not been satisfied correctly. The possible key-value pairs are explained on the following page of this document.	`{ "min": 1, "max": 2, "optionGroupId": 11 }`
objects	A series of JSON objects containing the invalid (or partially invalid) entities that have been provided to the OrderMate POS system by the integration partner. This, in conjunction with the "details" object, is provided to assist in troubleshooting errors in online ordering.	`"ExportMenuUnitDesc[0]": { "id": 1579, "removes": [], "unitSalePrice": 9.5, "menuSizeRef": { "menuID": 1 }, "adds": [], "menuPriceRef": { "menuID": 7 }, "ingredientRemoves": [], "menuItemRef": { "menuID": 725 }, "portion": "Whole", "options": [] }`

Possible Keys in "details"

The contents of the array held by the "details" key can and will vary depending on the nature of the error. The below table describes each of the possible keys that can be returned, and an explanation of what the value of each will mean. Through this means, it is possible for an integrating software to resolve the nature of an error programmatically and attempt to point the user in the right direction, or make an attempt to rectify the issue in the background.

Key	Explanation
min	Given when an option group hasn't been satisfied, this value dictates the minimum number of items an option group needs to be satisfied.
max	Given when an option group hasn't been satisfied, this value dictates the maximum number of items an option group needs to be satisfied.
salesAddId	The ID (or MenuRef ID) of the failed ExportSaleAdds entity.
salesRemoveId	The ID (or MenuRef ID) of the failed ExportSaleRemoves entity.
optionGroupId	The ID of the option group in the POS that was not satisfied. This is most commonly raised alongside the "min" and "max" keys, when an option is required to be selected and no option (or too many options) has been selected.
menuComboId	The ID (or MenuRef ID) of the failed ExportSalesCombo entity.
priceLevelId	The ID (or MenuRef ID) of the MenuPriceRef on an ExportMenuUnitDesc entity.
itemId	The ID of the failed ExportMenuUnitDesc entity.
optionId	The ID (or MenuRef ID) of the ExportSaleOption entity.
salesItemId	The POS-side ID of the Sales Item resolved from the ExportSalesItem entity.
salesComponentId	The POS-side ID of the Menu Unit resolved from the ExportMenuUnitDesc entity.

orderstatus Method (HTTP Get)

When an online order is received by the OrderMate system, the POS user can select a prep time. This time is then available using the orderstatus method.

vendorid -> parameter that is not optional, it is a text that indicates the name of the client, and is used to tackle the schema changes that are requested per client.

This method will return the status of the order requested. This can be used to display the current order status to the user.

Example URL

http://localhost:8090/integration/onlineorder/orderstatus/?user=test&password=test&orderid=1062&vendorid=myid

Example Response (Valid, but not acknowledged request)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OnlineOrderResult
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/onlineorder"
xmlns:ns3="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
xmlns:ns4="http://www.ordermate.com.au/xmlintegration/public/menu">
 <ns2:StatusList>
  <onlineOrderStatus statusCode="Valid" requestDatetime="2012-10-
05T14:49:27.859+10:00" orderId="1062"/>
 </ns2:StatusList>
</ns2:OnlineOrderResult>
```

Example Response (acknowledged request)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OnlineOrderResult
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/onlineorder"
xmlns:ns3="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
xmlns:ns4="http://www.ordermate.com.au/xmlintegration/public/menu">
 <ns2:StatusList>
  <onlineOrderStatus statusCode="Approved" requestDatetime="2012-10-
05T14:52:29.803+10:00" expectedReadyTime="2012-10-05T15:42:00.000+10:00"
orderId="1062"/>
 </ns2:StatusList>
</ns2:OnlineOrderResult>
```

Example Response (invalid request)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OnlineOrderResult
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/onlineorder"
xmlns:ns3="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
xmlns:ns4="http://www.ordermate.com.au/xmlintegration/public/menu">
 <ns2:StatusList>
  <onlineOrderStatus statusCode="NotAvailable" requestDatetime="2012-10-
05T14:53:32.437+10:00" orderId="1064"/>
 </ns2:StatusList>
</ns2:OnlineOrderResult>
```

orderdetails Method (HTTP Get)

For any account whether Online, Bartab, or Phone delivery, the orderdetails method takes the ID of the account in request, and gets back the details as needed, with some options parameter, the details returned can be customized as whether only account main/basic attributes, or including customer details, or sales items/combos as well.

If the account is of type delivery either a phone or online delivery, the attribute "delivery" that specifies the delivery status will be exported in the response, having one of these values (Not assigned, Assigned to driver, or Completed).
If the account is not a delivery, then this attribute is omitted.

Example URL 1

http://localhost:8090/integration/onlineorder/orderdetails?orderid=1001&vendorid=myid&user=test&password=test&options=

Example Response with options as empty: (Only account main attributes are returned)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OrderMateSale
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
isPaid="false" isDelivery="true" orderDate="2015-07-10T10:24:46.000+10:00"
totalPrice="134.10" amountPaid="0.00" open="true" delivery="Not assigned"
ID="1097"/>
```

Example URL 2
http://localhost:8090/integration/onlineorder/orderdetails?orderid=1001&vendorid=myid&user=test&password=test&options=customer

Example Response with options as "customer": (Account main attributes are returned along with customer details)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OrderMateSale
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
isPaid="false" isDelivery="true" orderDate="2015-07-10T10:24:46.000+10:00"
totalPrice="134.10" amountPaid="0.00" open="true" delivery="Not assigned"
ID="1097">
       <customer ID="48">
              <name title="" firstName="ZErnst Stavro" lastName="ZBlofeld"/>
              <address houseNumber="1800-SPECTRE" streetName="Spectre"
streetType="ave" suburb="Hollow Volcano" state="Poland" postCode="1234"/>
              <email/>
              <primaryPhone>1900-SPECTRE</primaryPhone>
              <secondaryPhone>03</secondaryPhone>
              <comments/>
       </customer>
</ns2:OrderMateSale>
```

Example URL 3
http://localhost:8090/integration/onlineorder/orderdetails?orderid=1001&vendorid=myid&user=test&password=test&options=customer,items

Example Response with options as "customer,items": (Account main attributes are returned along with customer details and sales items/combos)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:OrderMateSale
xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
isPaid="false" isDelivery="true" orderDate="2015-07-10T10:24:46.000+10:00"
totalPrice="134.10" amountPaid="0.00" open="true" delivery="Not assigned"
ID="1097">
       <customer ID="48">
              <name title="" firstName="ZErnst Stavro" lastName="ZBlofeld"/>
              <address houseNumber="1800-SPECTRE" streetName="Spectre"
streetType="ave" suburb="Hollow Volcano" state="Poland" postCode="1234"/>
              <email/>
              <primaryPhone>1900-SPECTRE</primaryPhone>
              <secondaryPhone>03</secondaryPhone>
              <comments/>
       </customer>
       <LineItem>
              <SalesItem Qty="1.00000000" course="0" ID="5089" label="Bruschetta
(Medium)">
                     <MenuUnitDesc unitSalePrice="9.23" portion="Whole" ID="623"
label="Bruschetta (Whole Medium)">
                            <MenuPriceRef menuID="1" name="TableOrder"/>
                     </MenuUnitDesc>
              </SalesItem>
       </LineItem>
```

```
        <LineItem>
              <SalesItem Qty="1.00000000" course="0" ID="5090" label="Chicken
Salad (Main)">
                    <MenuUnitDesc unitSalePrice="10.30" portion="Whole" ID="627"
label="Chicken Salad (Whole Main)">
                          <MenuPriceRef menuID="1" name="TableOrder"/>
                    </MenuUnitDesc>
              </SalesItem>
        </LineItem>
</ns2:OrderMateSale>
```

Example URL 4

http://localhost:8090/integration/onlineorder/orderdetails?orderid=1004&vendorid=myid&user=test&password=test

Example Response with no options: (All details are exported) and not a delivery account.

```
<ns2:OrderMateSale isPaid="false" isDelivery="false" orderDate="2009-04-
09T10:03:12.000+10:00" totalPrice="130.76" accountType="ONLINE_BARTAB"
creditLimit="10000.00" amountPaid="91.40" balance="9960.64" open="true" ID="1001">
  <LineItem>
    <SalesItem Qty="1.00000000" course="0" ID="4763" label="Chilli Prawns (Main)">
      <MenuUnitDesc unitSalePrice="21.22" portion="Whole" ID="791" label="Chilli
Prawns (Whole Main)">
        <MenuPriceRef menuID="1" name="TableOrder"/>
      </MenuUnitDesc>
    </SalesItem>
  </LineItem>
  <LineItem>
    <SalesItem Qty="4.35000000" course="0" extId="558431" ID="5016" label="Cascade
Light (Bottle)">
      <MenuUnitDesc unitSalePrice="3.84" portion="Whole" ID="467" label="Cascade
Light (Whole Bottle)">
        <MenuPriceRef menuID="1" name="TableOrder"/>
      </MenuUnitDesc>
    </SalesItem>
  </LineItem>
</ns2:OrderMateSale>
```

menu Method (HTTP Get)

This method can be called to produce the exported menu. The menu element is an OrderMateMenu (see the Menu Schema section for more detail). A valid profile and price level id must be supplied as parameters. The profile and price level ids can be retrieved using the

allprofiles and allpricelevels methods.

Parameter	Description	Mandatory₩Optional
profileid	The Inventory Profile to export.	Mandatory
pricelevelid	The price level to export.	Mandatory

Example URL

http://localhost:8090/integration/onlineorder/menu?user=test&password=test&profileid=1&pricelevelid=1

Example Response

Note in the example below all contents of the menu sections have been removed, see the OrderMateMenu schema section for an example menu export.

```
<ns2:OnlineOrderResult resultDesc="Successful" resultCode="200">
   <ns2:Menu label="TableOrder" ID="1">
      <MenuSection label="Side Dishes" ID="30">
             .........
      </MenuSection>
   </ns2:Menu>
</ns2:OnlineOrderResult>
```

allpricelevels Method (HTTP Get)

This method returns a list of all the available price levels. The price levels ids can then be used to specify the price level to export within the menu method request.

Example URL

http://localhost:8090/integration/onlineorder/allpricelevels?user=test&password=test

Example Response

```
<ns2:OnlineOrderResult resultDesc="Successful" resultCode="200">
   <ns2:PriceLevelList>
      <ns2:PriceLevel name="Student Special" id="2"/>
      <ns2:PriceLevel name="TableOrder" id="1"/>
   </ns2:PriceLevelList>
</ns2:OnlineOrderResult>
```

allprofiles Method (HTTP Get)

This method returns a list of all the available profiles. The profile ids can then be used to specify the profile to export within the menu method request.

Example URL

http://localhost:8090/integration/onlineorder/allprofiles?user=test&password=test

Example Response

```
<ns2:OnlineOrderResult resultDesc="Successful" resultCode="200"
menuVersion="1175"/>
   <ns2:OnlineOrderResult resultDesc="Successful" resultCode="200">
```

```
        <ns2:ProfileList><ns2:Profile name="TableOrder" id="1"/>
    </ns2:ProfileList>
</ns2:OnlineOrderResult>
```

Method Response Codes

All methods will return an OnlineOrderResponse (see the Online Order Webservice Schema). This will contain at minimum a response code and description attribute that will indicate the result of the method call. The possible response codes are listed in the table below:

Result Code	**Description**
200	Successful.
400	Invalid parameter.
400	Invalid profileId parameter. The plain-text response entity will contain more detailed information on the cause of the error.
400	Invalid pricelevelId parameter. The plain-text response entity will contain more detailed information on the cause of the error.
400	Error processing the order. The plain-text response entity will contain more detailed information on the cause of the error.
500	Internal Server Error. If this error persists then OrderMate support should be called.

Error Reporting for Online Orders

Online Order XML-Files are validated before they will be stored in the database. In case of an error, we simply return a HTTP Response with a 400 Bad Request or 500 Internal Server Error status code and a plain text response containing details as to what went wrong.

The legacy implementation (pre-2.8.26), however, was to generate an Error XML File with one of the following error codes:

So far, there are the following error codes generated:

- ORDER_INVALID (OrderInvalid)
- ORDER_NOT_COMPLETE (OrderNotComplete)
- XML_FORMAT_ERROR (XMLFormatError)
- ORDER_OTHER_ERROR (OrderOtherError)

Example

In case the structure of an "Online Order XML-File" should be wrong, we generate an XML Error file with the type "XML_FORMAT_ERROR", which shows the column and line number where the error appears in the XML-File.

Menu Schema – menu.xsd

Menu Sections, Items and Units

The menu schema provides the export format for a menu in the ordermate system. The root node of the menu is the <OrderMateMenu>, an Ordermate Menu contains a number of Menu Sections. Menu sections have a number of menu items, and menu items can have a number of menu item units. A menu item unit is a specified menu item with a price, a size and a portion. The portion specifies whether an item unit is a 'whole' or 'half' portion.
To give an example, a menu might have a section, Pizzas. Within the pizza menu section would be a number of items, Hawaiian, Mexicana and Supreme. Each pizza may be sold in different portions, either whole or half half, with a number of sizes, small, medium, large. Each instance of an item with a size and portion has a price, this is essentially what a menu item unit is. So a Hawaiian Pizza item may have six units:

Hawaiian (small, whole) @ $9
Hawaiian (medium, whole) @ $11
Hawaiian (large, whole) @ $13
Hawaiian (small, half) @ $5
Hawaiian (medium, half) @ $8
Hawaiian (large, half) @ $9

Modifications

There are a number of modifications that can be performed to menu items and menu item units, these modifications may change the price of a sales item. A menu item has a number of ingredients that can be removed, these are represented by the minus tag, a child of menu item. Menu items units may have a list of additions that can be made, this is represented by the plus tag, the plus is considered an addition to the regular item. Menu item units can also have a number of option groups, a user is forced to choose one option per group from a list of options. To give an example, a customer wants to order a Thick based Pizza Supreme with no anchovies and extra cheese. Anchovies would be a minus, it is something present on the pizza that can be removed. Extra cheese is an addition that can be made, it may cost an extra $0.50. Thick base could be set up as a pizza option, the option may be “thin base” or “thick base”, the ordered pizza must be one or the other.

Combos

Menu Sections can also have MenuCombos. A combo is a combination of menu item units sold at a special price. These are special deals, like two large pizzas and a garlic bread for $25. A

MenuCombo contains a number of combo groups, a combo group specifies what item units can be ordered and how many of that item can be ordered. In the two large pizza and a garlic bread for $25 combo, there would be a combo group arbitrarily called "large pizza", this would have a minimum and maximum quantity of 2, and would list the large pizza item units that are available to form this combo. The combo would have a second group, with garlic bread in it, and a minimum and maximum quantity of 1. For a combo to be valid when it is ordered, it must satisfy the constraints of its combo groups. Price Adds on links to items are optional, with empty or null implying $0 price add. The Third Party Integrator is expected to include the value of the price add if the unit is ordered within a combo.

The 'apportionable' flag in ComboGroups has special significance. If 'true' specifies that an item ordered in that group will have the discount applied to it, false otherwise. For example: If I have a 30% discount combo, then the 30% discount is made and consumed over only those items that are apportionable.

Combo Group "Burger" = not Apportionable

Combo Group "Chips" = Apportionable

Combo Group "Drink" = Apportionable

So Elvis Burger = $7. Drink = $3. Chips = $3 := $13 Combo Price

Thus, Elvis Burger = $7, Drink = $3 – 30% = $2.10, Chips = $2.10 = $11.20 Combo Price

If no Combo Groups are marked as apportionable, then all Combo Groups are considered apportionable.

Sales Schema

Internet orders come into the system following the SalesOrder.xsd schema. The root element of a sale is <OrderMateSale>. The sale has a number of properties, it must have a customer, and has a number of line items. A line item is either a sales item or a sales combo. A sales item has a quantity, course number (optional), item notes (optional) and either a menu unit description or a partial sales items. A partial sales item has two halves.

Menu Unit Descriptions

A menu unit description is where the sales schema references back to the imported menu. A menu item description references a menu item, a size, a unit price, a list of options, a list of additions and a list of minuses. The menu references are matched on the ID attribute of the menu ref, the label of the referenced item is provided for trouble shooting and is not used to match the sales element back to the corresponding menu element.

Sales Modifications

There are three sales modifications corresponding to the menu modifications: options, adds and removes. Each sales modification can only be formed from the corresponding menu modification

type. A sales option can only be formed from a menu reference to a menu option, likewise a sales add is formed from a menu add, and a sales remove is formed from a menu minus.

Partial Items

A partial item has a first half and a second half. Each half references a menu unit description. A partial item can only be formed from menu units that have a portion of "half". The total price of the item is the sum of the price of the halves.

Combos

A sales combo is a collection of sales items sold as a single line item at a special price. A sales combo must meet the requirements of the corresponding menu combo. A combo has a number of groups, those groups specify constraints on the qty and instance of item that can be ordered. The constraints of all combo groups must be met for a combo to be valid.

ExportCustomers

A sales order must have a customer. In the integration, customers are matched on their email addresses. The OrderMateSale element has a boolean attribute, isDelivery, if this is true a customer must have an address, otherwise only a name and phone number are required.

Schemas

The two xml schemas reference Entity.xsd, and many of the elements within the SalesOrder and Menu schemas extend the ExportEntity of this schema. The export entity has two attributes a label and an ID, both of which are optional. The ID field is specifically used to reference an ID of an entity in the ordermate system, this field should be blank where an entity is not in the ordermate database, so generally this attribute will be empty on incoming sale orders.

ENTITY.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:jxb="http://java.sun.com/xml/ns/jaxb" jxb:version="2.0"
  targetNamespace="http://www.ordermate.com.au/xmlintegration/entity"
  xmlns:ent="http://www.ordermate.com.au/xmlintegration/entity">

  <!--*******************************************************-->
  <!-- COPYRIGHT VC Solutions Pty Ltd 2010                   -->
  <!-- The sole purpose of this xml schema is to be used to-->
  <!-- integerate with the OrderMate POS system              -->
  <!--*******************************************************-->

  <!-- COMMON ENTITY BASE TYPE -->
  <xsd:complexType name="ExportEntity">
    <xsd:attribute name="ID" type="xsd:long" use="optional" />
    <xsd:attribute name="label" type="xsd:string" use="optional" />
  </xsd:complexType>

  <!-- PROPERTIED ENTITY ************************************
    -* A propertied entity is an entity with a list of property elements,
    -* a property value can only be a basic datatype, some from java.lang
    -* or a datatype that we support special handling of.
    -*
    -* A Property of a PropertiedEntity is basically a ValueObject,
    -* it is anything that does not have it's own externalID.
  -->
  <xsd:complexType name="PropertiedEntity">
    <xsd:complexContent>
      <xsd:extension base="ent:ExportEntity">
        <xsd:sequence>
         <xsd:element name="Property" type="ent:Prop"
                 minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="XMLRefID" type="xsd:ID"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- PROPERTY ***************************************************
    -* A property represents a property on an object, the type-package of the
    -* object is optional, if it is missing then it is assumed to be a type
    -* from the package java.lang
    -* val is the value of the property and missing the values are null
    -* ************************************************************-->
  <xsd:complexType name="Prop">
    <xsd:attribute name="name" type="xsd:string" use="required"/>
    <xsd:attribute name="pkg" type="xsd:string" default="java.lang" use="optional"/>
    <xsd:attribute name="type" type="xsd:string" use="required"/>
    <xsd:attribute name="val" type="xsd:string" use="optional"/>
    <xsd:attribute name="required" type="xsd:boolean" default="false" use="optional"/>
  </xsd:complexType>
</xsd:schema>
```

SALES.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:jxb="http://java.sun.com/xml/ns/jaxb" jxb:version="2.0"
        xmlns:ent="http://www.ordermate.com.au/xmlintegration/entity"
        targetNamespace="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
        xmlns:sale="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
        xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
        jxb:extensionBindingPrefixes="xjc">

        <!--*****************************************************-->
        <!-- COPYRIGHT VC Solutions Pty Ltd 2010                 -->
        <!-- The sole purpose of this xml schema is to be used to-->
        <!-- integrate with the OrderMate POS system             -->
        <!--*****************************************************-->

        <xsd:annotation>
                <xsd:appinfo>
                <jxb:globalBindings>
        <xjc:simple/>
        </jxb:globalBindings>
                </xsd:appinfo>
        </xsd:annotation>

        <xsd:import namespace="http://www.ordermate.com.au/xmlintegration/entity"
              schemaLocation="Entity.xsd"/>

        <!-- ROOT ELEMENT -->
        <xsd:element name="OrderMateSale" type="sale:ExportSaleOrder" />

        <!-- SALES ORDER TYPE -->
        <xsd:complexType name="ExportSaleOrder" >
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="customer" type="sale:ExportCustomer"
minOccurs="0" maxOccurs="1"/>
                                        <xsd:element name="LineItem" type="sale:LineItem"
minOccurs="0" maxOccurs="unbounded" />
                                        <xsd:element name="payment" type="sale:OnlinePrePayment"
minOccurs="0" maxOccurs="unbounded"/>
                                </xsd:sequence>
                                <xsd:attribute name="isPaid" type="xsd:boolean"/><!-- Defines that
the account has one or many payments -->
                                <xsd:attribute name="isDelivery" type="xsd:boolean"/>
                                <xsd:attribute name="orderDate" type="xsd:dateTime"/>
                                <xsd:attribute name="totalPrice" type="xsd:decimal"/>
                                <xsd:attribute name="extWebTableID" type="xsd:string"/>
                                <xsd:attribute name="extOrderID" type="xsd:string"/>
                                <xsd:attribute name="extHRef" type="xsd:string"/>
                                <xsd:attribute name="extXMLDocketHRef" type="xsd:string"/>
                                <xsd:attribute name="extSourceName" type="xsd:string"/>
                                <xsd:attribute name="timeDue" type="xsd:time"/>
                                <xsd:attribute name="dateDue" type="xsd:date" use="optional"/>
                                <xsd:attribute name="cashdrawerName" type="xsd:string"
use="optional"/> <!-- Legacy to support single payment -->
                                <xsd:attribute name="receiptPrinterName" type="xsd:string"
use="optional"/>
                                <xsd:attribute name="accountType" type="xsd:string"
use="optional"/> <!-- Not for external use -->
                                <xsd:attribute name="accountName" type="xsd:string"
use="optional"/> <!-- Not for external use -->
                                <xsd:attribute name="paymentType" type="xsd:string"
use="optional"/> <!-- Legacy to support single payment -->
                                <xsd:attribute name="comment" type="xsd:string" use="optional"/>
<!-- comments upon the account -->
```

```
                                <xsd:attribute name="creditLimit" type="xsd:decimal"
use="optional"/> <!-- outgoing credit limit upon the account -->
                                <xsd:attribute name="amountPaid" type="xsd:decimal"
use="optional"/> <!-- outgoing amount paid off an account -->
                                <xsd:attribute name="balance" type="xsd:decimal" use="optional"/>
<!-- credit amount left on the account -->
                                <xsd:attribute name="open" type="xsd:boolean" use="optional"/> <!-
- Whether or not the account is open -->
                                <xsd:attribute name="delivery" type="sale:DeliveryStatus"
use="optional"/> <!-- Either completed or whether assigned to a driver or not -->
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- LINE ITEM CONTAINER -->
        <xsd:complexType name="LineItem">
                <xsd:sequence>
                        <xsd:choice>
                                <xsd:element name="SalesItem" type="sale:ExportSalesItem"/>
                                <xsd:element name="SalesCombo" type="sale:ExportSalesCombo"/>
                        </xsd:choice>
                </xsd:sequence>
        </xsd:complexType>

        <!-- SALES ITEM -->
        <!--  -->
        <xsd:complexType name="ExportSalesItem">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:choice>
                                        <xsd:element name="MenuUnitDesc"
type="sale:ExportMenuUnitDesc"/>
                                        <xsd:element name="PartialSalesItem"
type="sale:ExportSalesPartialItem"/>
                                </xsd:choice>
                                <xsd:attribute name="Qty" type="xsd:decimal"/>
                                <xsd:attribute name="earnPoints" type="xsd:integer"
use="optional"/><!-- The number of points earnt if purchased -->
                                <xsd:attribute name="redeemPoints" type="xsd:integer"
use="optional"/>  <!-- The points to redeem this item -->
                                <xsd:attribute name="redeemQty" type="xsd:integer"
use="optional"/>        <!-- The number of items to redeem -->
                                <xsd:attribute name="course" type="xsd:int" use="optional"/> <!--
The number of the course, 0,1,2,3 -->
                                <xsd:attribute name="salesItemNotes" type="xsd:string"
use="optional"/>
                                <xsd:attribute name="extId" type="xsd:long" use="optional" />
                        </xsd:extension>
                </xsd:complexContent>

        </xsd:complexType>

        <!-- SALES COMBO -->
        <!-- The price is optional, if it is missing, the price is taken from the
                sum of the component prices -->
        <xsd:complexType name="ExportSalesCombo">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="menuCombo" type="sale:MenuRef"
minOccurs="1" maxOccurs="1"/>
                                        <xsd:element name="salesComboItem"
type="sale:ExportSalesItem" minOccurs="1" maxOccurs="unbounded"/>
                                </xsd:sequence>
                                <xsd:attribute name="Qty" type="xsd:decimal"/>
                                <xsd:attribute name="earnPoints" type="xsd:integer"
use="optional"/><!-- The number of points earnt if purchased -->
                                <xsd:attribute name="redeemPoints" type="xsd:integer"
use="optional"/>  <!-- The points to redeem this item -->
                                <xsd:attribute name="redeemQty" type="xsd:integer"
use="optional"/>        <!-- The number of items to redeem -->
```

```
                        <xsd:attribute name="unitPrice" type="xsd:decimal"
use="optional"/>
                    </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <xsd:complexType name="ExportMenuUnitDesc">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="MenuItemRef" type="sale:MenuRef"
minOccurs="0" maxOccurs="1"/>
                                        <xsd:element name="MenuPriceRef" type="sale:MenuRef"
minOccurs="0" maxOccurs="1"/>
                                        <xsd:element name="MenuSizeRef" type="sale:MenuRef"
minOccurs="0" maxOccurs="1"/>
                                        <xsd:element name="options" type="sale:ExportSaleOption"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xsd:element name="adds" type="sale:ExportSaleAdds"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xsd:element name="removes" type="sale:ExportSaleRemoves"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xsd:element name="ingredientRemoves"
type="sale:ExportSaleIngredientRemove" minOccurs="0"  maxOccurs="unbounded"/>
                                </xsd:sequence>
                                <xsd:attribute name="unitSalePrice" type="xsd:decimal"
use="required"/>
                                <xsd:attribute name="portion" type="xsd:string"/>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- HALF HALF Sales item -->
        <!-- Each menu unit must have a size of 'half' -->
        <xsd:complexType name="ExportSalesPartialItem">
                        <xsd:all>
                                <xsd:element name="firstHalf" type="sale:ExportMenuUnitDesc"/>
                                <xsd:element name="secondHalf" type="sale:ExportMenuUnitDesc"/>
                        </xsd:all>
                        <xsd:attribute name="totalUnitPrice" type="xsd:decimal" use="required"/>
        </xsd:complexType>

        <!-- SALES OPTION -->
        <xsd:complexType name="ExportSaleOption">
                <xsd:all>
                        <xsd:element name="MenuMod" type="sale:MenuRef" minOccurs="1"/>
                </xsd:all>
                <xsd:attribute name="Price" type="xsd:decimal"/>
                <xsd:attribute name="Exclude" type="xsd:boolean" use="optional"/>
        </xsd:complexType>

        <!-- SALES ADD -->
        <xsd:complexType name="ExportSaleAdds">
                <xsd:all>
                        <xsd:element name="MenuMod" type="sale:MenuRef" minOccurs="1"/>
                </xsd:all>
                <xsd:attribute name="Price" type="xsd:decimal"/>
                <xsd:attribute name="Qty" type="xsd:integer"/>
                <xsd:attribute name="Exclude" type="xsd:boolean" use="optional"/>
        </xsd:complexType>

        <!-- SALES REMOVES -->
        <!-- The price field will be added to the price of the item, so to reduce the
                 price the price attribute must be negative. -->
        <xsd:complexType name="ExportSaleRemoves">
                <xsd:all>
                        <xsd:element name="MenuMod" type="sale:MenuRef" minOccurs="1"/>
                </xsd:all>
                <xsd:attribute name="Price" type="xsd:decimal"/>
        </xsd:complexType>
```

```
        <!-- REMOVED INGREDIENTS -->
        <xsd:complexType name="ExportSaleIngredientRemove">
                <xsd:all>
                        <xsd:element name="MenuIngredient" type="sale:MenuRef" minOccurs="1"
maxOccurs="1"/>
                </xsd:all>
                <xsd:attribute name="Price" type="xsd:decimal"/>
        </xsd:complexType>

        <!-- REF  -->
        <xsd:complexType name="MenuRef">
                <xsd:attribute name="menuID" type="xsd:integer"/>
                <xsd:attribute name="name" type="xsd:string"/>
        </xsd:complexType>

        <!-- CUSTOMER -->
        <xsd:complexType name="ExportCustomer">
                <xsd:all>
                        <xsd:element name="name" type="sale:PersonName" minOccurs="1"/>
                        <xsd:element name="address" type="sale:Address" minOccurs="0"/>
                        <xsd:element name="email" type="xsd:string" minOccurs="1"
nillable="false"/>
                        <xsd:element name="primaryPhone" type="xsd:string" minOccurs="1"/>
                        <xsd:element name="secondaryPhone" type="xsd:string" minOccurs="0"/>
                        <xsd:element name="comments" type="xsd:string" minOccurs="0"/>
                        <xsd:element name="currentPoints" type="xsd:integer" minOccurs="0"/>
                        <xsd:element name="customerCard" type="xsd:string" minOccurs="0"/>
        <!-- The customer card string -->
                </xsd:all>
                <xsd:attribute name="ID" type="xsd:long" use="optional" />
        </xsd:complexType>

        <xsd:complexType name="PersonName">
                <xsd:attribute name="title" type="xsd:string" use="optional"/>
                <xsd:attribute name="firstName" type="xsd:string" use="required"/>
                <xsd:attribute name="lastName" type="xsd:string" use="optional"/>
        </xsd:complexType>

        <xsd:complexType name="Address">
                <xsd:attribute name="houseNumber" type="xsd:string" use="required"/>
                <xsd:attribute name="streetName" type="xsd:string" use="required"/>
                <xsd:attribute name="streetType" type="xsd:string" use="optional"/>
                <xsd:attribute name="suburb" type="xsd:string" use="required"/>
                <xsd:attribute name="state" type="xsd:string" use="optional"/>
                <xsd:attribute name="postCode" type="xsd:string" use="optional"/>
        </xsd:complexType>

        <!-- Online Transaction -->
        <xsd:complexType name="OnlinePrePayment">
                <xsd:attribute name="amount" type="xsd:double" use="required"/>
        <!-- Cannot be blank -->
                <xsd:attribute name="cashdrawerName" type="xsd:string" use="optional"/>   <!-- If
left blank, will go into default cashdrawer -->
                <xsd:attribute name="paymentType" type="xsd:string" use="optional"/>      <!-- If
left blank, will go into default for internet prepayment -->
        </xsd:complexType>

        <!-- Delivery Status -->
        <xsd:simpleType name="DeliveryStatus">
                <xsd:restriction base="xsd:string">
                        <xsd:enumeration value="Assigned to driver" />
                        <xsd:enumeration value="Not assigned" />
                        <xsd:enumeration value="Completed" />
                </xsd:restriction>
        </xsd:simpleType>
</xsd:schema>
```

MENU.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:jxb="http://java.sun.com/xml/ns/jaxb" jxb:version="2.0"
        xmlns:ent="http://www.ordermate.com.au/xmlintegration/entity"
        targetNamespace="http://www.ordermate.com.au/xmlintegration/public/menu"
        xmlns:menu="http://www.ordermate.com.au/xmlintegration/public/menu"
        xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
        jxb:extensionBindingPrefixes="xjc">

        <!--*****************************************************-->
        <!-- COPYRIGHT VC Solutions Pty Ltd 2010                                  -->
        <!-- The sole purpose of this xml schema is to be used to-->
        <!-- integerate with the OrderMate POS system                    -->
        <!--*****************************************************-->

        <xsd:annotation>
                <xsd:appinfo>
                <jxb:globalBindings>
        <xjc:simple/>
        </jxb:globalBindings>
                </xsd:appinfo>
        </xsd:annotation>

        <xsd:import namespace="http://www.ordermate.com.au/xmlintegration/entity"
              schemaLocation="Entity.xsd"/>

    <!-- ROOT ELEMENT -->
        <xsd:element name="OrderMateMenu" type="menu:Menu" />

        <!-- INVENTORY PROFILE -->
        <xsd:complexType name="Menu">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                        <xsd:sequence>
                                <xsd:element name="MenuSection" type="menu:MenuSection"
                                        maxOccurs="unbounded" />
                        </xsd:sequence>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!--  INVENTORY SECTION -->
        <xsd:complexType name="MenuSection">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="MenuItem" type="menu:MenuItem"
minOccurs="0" maxOccurs="unbounded" />
                                        <xsd:element name="MenuCombo" type="menu:MenuCombo"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xsd:element name="MenuSection" type="menu:MenuSection"
        minOccurs="0" maxOccurs="unbounded" />
                                </xsd:sequence>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- MENU ITEM TYPE -->
        <xsd:complexType name="MenuItem">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="Minus" type="menu:Modification"
minOccurs="0" maxOccurs="unbounded" />
                                        <xsd:element name="MenuItemUnit" type="menu:MenuItemUnit"
minOccurs="1" maxOccurs="unbounded"/>
```

```
                                </xsd:sequence>
                                <xsd:attribute name="sequence" type="xsd:int" />
                                <xsd:attribute name="optionMin" type="xsd:int" use="optional"/>
<!-- Old Sticky Options, deprecated -->
                                <xsd:attribute name="optionMax" type="xsd:int" use="optional"/>
<!-- Old Sticky Options, deprecated -->
                                <xsd:attribute name="icon" type="xsd:string" use="optional"/> <!--
An optional url to an image/icon for the item -->
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- MENU ITEM UNIT -->
        <xsd:complexType name="MenuItemUnit">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="Size" type="menu:Size" minOccurs="1"
maxOccurs="1"/>
                                        <xsd:element name="Portion" type="menu:Portion"
minOccurs="1" maxOccurs="1"/>
                                        <xsd:element name="Price" type="menu:UnitPrice"
minOccurs="1" maxOccurs="unbounded"/>
                                        <xsd:element name="Plus" type="menu:Modification"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xsd:element name="Ingredient" type="menu:MenuIngredient"
minOccurs ="0" maxOccurs="unbounded"/>
                                        <xsd:element name="OptionGroup"
type="menu:ModificationGroup" minOccurs="0" maxOccurs="unbounded"/>
                                        <xsd:element name="DefaultMod" type="menu:Modification"
minOccurs="0" maxOccurs="unbounded"/>
                                </xsd:sequence>
                                <xsd:attribute name="XMLRefID" type="xsd:ID"/>
                                <xsd:attribute name="recipe" type="xsd:string" use="optional"/>
        <!-- An optional recipe description of the unit -->
                                <xsd:attribute name="state" type="xsd:string" use="optional"/> <!-
- The state of the item. ACTIVE, INACTIVE or DELETED. None implies ACTIVE -->
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- UNITPRICE -->
        <xsd:complexType name="UnitPrice">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:attribute name="value" type="xsd:decimal" />
                                <xsd:attribute name="priceLevelID" type="xsd:int" />
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!--  SIZE -->
        <xsd:complexType name="Size">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:attribute name="Shortname" type="xsd:string"/>
                                <xsd:attribute name="Sequence" type="xsd:int"/>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- PORTION: one of whole or half -->
        <xsd:complexType name="Portion">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:attribute name="value" default="whole">
                                 <xsd:simpleType>
                                <xsd:restriction base="xsd:string">
                                        <xsd:enumeration value="whole"/>
                                        <xsd:enumeration value="half"/>
                                </xsd:restriction>
```

```
                        </xsd:simpleType>
                        </xsd:attribute>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- MODIFICATION GROUP -->
        <xsd:complexType name="ModificationGroup">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="Option" type="menu:Modification"
minOccurs="1" maxOccurs="unbounded"/>
                                </xsd:sequence>
                                <xsd:attribute name="sequence" type="xsd:int"/>
                                <xsd:attribute name="minQty" type="xsd:int"/>    <!-- New min
quantities orderable on option groups. 0 : max (unless 0) -->
                                <xsd:attribute name="maxQty" type="xsd:int"/>    <!-- New max
quantities orderable on option groups. 0 : unbounded. 0 implies no boundary -->
                                <xsd:attribute name="reducedZone" type="xsd:int" use="optional"/>
<!-- The number of Modifications orderable that will attract a reduced  price. Empty implies 0 or
no reduced zone -->
                                        <!-- Kiosk only setting. 3rd party optional -->
                                <xsd:attribute name="accountType" type="xsd:string"
use="optional"/>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- MODIFICATION -->
        <xsd:complexType name="Modification">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:attribute name="price" type="xsd:decimal"/>
                                <xsd:attribute name="sequence" type="xsd:int"/>
                                <xsd:attribute name="defaultMod" type="xsd:boolean"
use="optional"/> <!-- Defines a Modification that is ordered by default when the item is ordered --
>
                                <xsd:attribute name="reducedFactor" type="xsd:decimal"
use="optional"/> <!-- The multiplication of the price if ordered within the reducedZone. Empty
implies no reduction. -->
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- INGREDIENT -->
        <xsd:complexType name="MenuIngredient">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:attribute name="price" type="xsd:decimal"/>         <!-- The
price of the ingredient if removed -->
                                <xsd:attribute name="removable" type="xsd:boolean"/>     <!-- If
not removable, cannot be removed -->
                                <xsd:attribute name="quantity" type="xsd:decimal" use="optional"/>
<!-- Reserved for future implementation -->
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- MENU COMBO - a menu combo can have a flat price, or if the price is not set
                        then the price is simply the sum of the price of its component items.
        -->
        <xsd:complexType name="MenuCombo">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="ComboGroup" type="menu:ComboGroup"
minOccurs="1" maxOccurs="unbounded"/>
                                        <!-- POS-5051: Advanced combo details will be exported in
online menu XML. -->
                                        <xsd:element name="FreeItemGroup" type="menu:ComboGroup"
```

```
minOccurs="0" maxOccurs="1"/>
                                </xsd:sequence>
                                <xsd:attribute name="price" type="xsd:decimal" use="optional"/>
                                <xsd:attribute name="shortname" type="xsd:string"/>
                                <xsd:attribute name="sequence" type="xsd:int"/>
                                <!-- POS-5051: Advanced combo details will be exported in online
menu XML. -->
                                <xsd:attribute name="discount" type="xsd:decimal" use="optional"/>
                                <xsd:attribute name="discountUnit" type="xsd:string"
use="optional"/>
                                <xsd:attribute name="comboStrategy" type="xsd:string"
use="optional"/>
                                <xsd:attribute name="type" type="xsd:string" use="optional" />

                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>

        <!-- COMBO GROUP elements have a list of menu item units that comprise the combo group,
                        rather than relist the details of the menu item unit, they are referenced
by their
                        XMLRefID value
         -->
        <xsd:complexType name="ComboGroup">
                <xsd:complexContent>
                        <xsd:extension base="ent:ExportEntity">
                                <xsd:sequence>
                                        <xsd:element name="ComboItem" type="menu:MenuItemUnitRef"
minOccurs="1" maxOccurs="unbounded"/>
                                </xsd:sequence>
                                <xsd:attribute name="minQty" type="xsd:decimal"/>
                                <xsd:attribute name="maxQty" type="xsd:decimal"/>
                                <xsd:attribute name="sequence" type="xsd:int"/>
                                <xsd:attribute name="apportionable" type="xsd:boolean"
use=”optional”/>
                        </xsd:extension>
                </xsd:complexContent>
        </xsd:complexType>
        <xsd:complexType name="MenuItemUnitRef">
                <xsd:attribute name="XMLRefID" type="xsd:IDREF">
                        <xsd:annotation>
                <xsd:appinfo>
                        <jxb:property>
                                <jxb:baseType name="MenuItemUnit"/>
                        </jxb:property>
                </xsd:appinfo>
            </xsd:annotation>
                </xsd:attribute>
                <xsd:attribute name="comboPriceAdd" type="xsd:decimal" use="optional"/> <!-- The
price that would be added if ordered in a combo -->
        </xsd:complexType>
</xsd:schema>
```

Menu & Sales integration (Online Ordering)

SAMPLE ORDERMATE SALE

```
<?xml version="1.0" encoding="UTF-8"?>
<ns2:OrderMateSale xmlns:ns2="http://www.ordermate.com.au/xmlintegration/public/SaleOrder"
  label="iTakeaway Order 5"
  isPaid="false"
  isDelivery="false"
  orderDate="2010-06-15T12:00:00"
  totalPrice="9.00"
  extOrderID="99"
  extSourceName="itakeaway"
  extHRef="http://ordermatetest.com/order99.xml"
  extXMLDocketHRef=http://ordermatetest.com/order99Docket.xml
  comment="Please have order waiting on counter when I get there">
  <customer>
    <name title="Mr" firstName="Steve" lastName="Lancashire"/>
    <address houseNumber="59"
      streetName="Fennell"
      streetType="st"
      suburb="Port Melbourne"
      state="Vic"
      postCode="3205"/>
    <email>steve@ordermate.com.au</email>
    <primaryPhone>0466 400 999</primaryPhone>
    <secondaryPhone>1300 667 067</secondaryPhone>
    <comments>Steve is a pretty nice guy</comments>
  </customer>
  <LineItem>
    <SalesItem Qty="1" label="Baci (Main, Whole)">
      <MenuUnitDesc unitSalePrice="9.00" portion="Whole" >
        <MenuItemRef menuID="409" name="Baci"/>
        <MenuPriceRef menuID="1" name="TableOrder"/>
        <MenuSizeRef menuID="1" name="Main"/>
      </MenuUnitDesc>
    </SalesItem>
  </LineItem>
</ns2:OrderMateSale>
```

Menu & Sales integration (Online Ordering)

ONLINE ORDER ERROR SCHEMA

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:jxb="http://java.sun.com/xml/ns/jaxb" jxb:version="2.0"
  xmlns:onlineordererror="http://www.ordermate.com.au/xmlintegration/public/onlineordererror"
  targetNamespace="http://www.ordermate.com.au/xmlintegration/public/onlineordererror">

  <!--*******************************************************-->
  <!-- COPYRIGHT VC Solutions Pty Ltd 2012                   -->
  <!-- The sole purpose of this xml schema is to be used to-->
  <!-- integerate with the OrderMate POS system              -->
  <!--*******************************************************-->

  <!-- ROOT ELEMENT -->
  <xsd:element name="OnlineOrderErrorList"          type="onlineordererror:OnlineOrderErrorList" />

  <xsd:simpleType name="ErrorCodeType">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="OrderInvalid"/>
      <xsd:enumeration value="OrderNotComplete"/>
      <xsd:enumeration value="OrderOtherError"/>
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="OnlineOrderErrorEntry">
    <xsd:attribute name="errorCode"                 type="onlineordererror:ErrorCodeType" />
    <xsd:attribute name="date"                      type="xsd:date"/>
    <xsd:attribute name="time"                      type="xsd:time"/>
    <xsd:attribute name="location"                  type="xsd:string"/>
    <xsd:attribute name="description"               type="xsd:string"/>
    <xsd:attribute name="label"                     type="xsd:string"/>
    <!-- <xsd:attribute name="failedSaleOrder"      type="sale:ExportSaleOrder"/> -->
    <xsd:attribute name="IdFailedSaleOrder"         type="xsd:long"/>
  </xsd:complexType>

  <xsd:complexType name="OnlineOrderErrorList">
    <xsd:sequence>
      <xsd:element name="onlineOrderError" type="onlineordererror:OnlineOrderErrorEntry"
                                           minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```

ONLINE ORDER SCHEMA

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:jxb="http://java.sun.com/xml/ns/jaxb" jxb:version="2.0"
      xmlns:menu="http://www.ordermate.com.au/xmlintegration/public/menu"
      xmlns:onlineordererror="http://www.ordermate.com.au/xmlintegration/public/onlineordererror"
      targetNamespace="http://www.ordermate.com.au/xmlintegration/public/onlineorder"
      xmlns:onlineorder="http://www.ordermate.com.au/xmlintegration/public/onlineorder"
      elementFormDefault="qualified">

  <!-- Imports for reference schemas -->
  <xsd:import namespace="http://www.ordermate.com.au/xmlintegration/public/onlineordererror"
              schemaLocation="OnlineOrderError.xsd"/>

  <xsd:import namespace="http://www.ordermate.com.au/xmlintegration/public/menu"
              schemaLocation="Menu.xsd"/>

    <!-- ROOT ELEMENT -->
  <xsd:element name="OnlineOrderResult" type="onlineorder:OnlineOrderResult" />

        <!-- Main type, contains all other types -->
        <xsd:complexType name="OnlineOrderResult">
                <xsd:sequence>
                        <xsd:element name="ProfileList"           type="onlineorder:ProfileList"

        minOccurs="0" maxOccurs="1"/>

                        <xsd:element name="PriceLevelList"type="onlineorder:PriceLevelList"

        minOccurs="0" maxOccurs="1"/>

                        <xsd:element name="ErrorList"
        type="onlineordererror:OnlineOrderErrorList"

        minOccurs="0" maxOccurs="1"/>

                        <xsd:element name="StatusList"
        type="onlineorderstatus:OnlineOrderStatusList"

        minOccurs="0" maxOccurs="1"/>


                        <xsd:element name="Menu"                  type="menu:Menu"

        minOccurs="0" maxOccurs="1"/>


                </xsd:sequence>
                <xsd:attribute name="resultCode"  type="xsd:integer"        use="required"/>
                <xsd:attribute name="resultDesc"  type="xsd:string"         use="required"/>
                <xsd:attribute name="orderId"             type="xsd:long"   use="optional"/>
                <xsd:attribute name="menuVersion" type="xsd:long"  use="optional"/>
                <xsd:attribute name="tablesVersion"       type="xsd:long"   use="optional"/>
  </xsd:complexType>

  <!-- Profile List -->
  <xsd:complexType name="ProfileList">
    <xsd:sequence>
      <xsd:element name="Profile"          type="onlineorder:Profile"
                                           minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <!-- Profile -->
  <xsd:complexType name="Profile">
    <xsd:attribute name="id"                        type="xsd:long"           use="required"/>
    <xsd:attribute name="name"                      type="xsd:string"/>
  </xsd:complexType>

  <!-- Price Level List -->
  <xsd:complexType name="PriceLevelList">
    <xsd:sequence>
      <xsd:element name="PriceLevel"                type="onlineorder:PriceLevel"
                                                    minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
```

```
  <!-- Price Level -->
  <xsd:complexType name="PriceLevel">
    <xsd:attribute name="id"                    type="xsd:long"          use="required"/>
    <xsd:attribute name="name"                  type="xsd:string"/>
  </xsd:complexType>
</xsd:schema>
```

ONLINE ORDER STATUS SCHEMA

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:jxb="http://java.sun.com/xml/ns/jaxb" jxb:version="2.0"
      xmlns:onlineorderstatus="http://www.ordermate.com.au/xmlintegration/public/onlineorderstatus"
      targetNamespace="http://www.ordermate.com.au/xmlintegration/public/onlineorderstatus">

      <!--*******************************************************-->
      <!-- COPYRIGHT VC Solutions Pty Ltd 2012                  -->
      <!-- The sole purpose of this xml schema is to be used to-->
      <!-- integerate with the OrderMate POS system    -->
      <!--                                                      -->
      <!--
         -->
      <!--*******************************************************-->

      <!-- ROOT ELEMENT -->
      <xsd:element name="OnlineOrderStatusList"    type="onlineorderstatus:OnlineOrderStatusList" />

      <xsd:simpleType name="StatusCodeType">
        <xsd:restriction base="xsd:string">
        <xsd:enumeration value="Valid"/>           <!-- Order was valid but has not been accepted by
the pos user yet -->
        <xsd:enumeration value="Approved"/>                <!-- Order has been accepted and is in the
system -->
        <xsd:enumeration value="Rejected"/>                <!-- Order has been rejected by an operator
-->
        <xsd:enumeration value="NotAvailable"/>    <!-- Account has been removed or is not a valid id
or doesn't belong to external vendor -->
        </xsd:restriction>
      </xsd:simpleType>

      <xsd:complexType name="OnlineOrderStatusEntry">
        <xsd:attribute name="statusCode"
      type="onlineorderstatus:StatusCodeType" />
        <xsd:attribute name="requestDatetime"                          type="xsd:dateTime"/>     <!-- The
date when this status request was received by the POS clock -->
        <xsd:attribute name="accountExtId"                             type="xsd:string"/>
      <!-- The external or vendor specific account id -->
        <xsd:attribute name="accountId"                                        type="xsd:long"/>
      <!-- Ordermate unique id of the account -->
        <xsd:attribute name="creationDatetime"                         type="xsd:dateTime"/>     <!-- The
time when the account was added to the db -->
        <xsd:attribute name="expectedReadyTime"            type="xsd:dateTime" use="optional"/> <!--
Datetime when account should be ready, null if not specified yet -->
      </xsd:complexType>

      <xsd:complexType name="OnlineOrderStatusList">
        <xsd:sequence>
                <xsd:element name="onlineOrderStatus"
      type="onlineorderstatus:OnlineOrderStatusEntry"

      minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:complexType>

</xsd:schema>
```