



# **Exceptions to Real-Time Processing**

July 28, 2008

## Table of Contents

Exceptions to Real-Time Processing .....	1
Scenario Examples .....	3
Typical scenario .....	3
Exceptions .....	3
RWI 2 .....	5
Viewing the list of your queued items .....	5
Acknowledging queued items .....	6
Finding previously acknowledged items .....	7
Changing your queue summary preferences .....	8
API .....	9
Query Queued Order or Inventory Items .....	9
Perl examples for Query Queued Items .....	9
XML examples for Query Queued Items .....	14
Request parameter definitions for Query Queued Items .....	22
Response parameter definitions for Query Queued Items .....	24
Acknowledge Queued Items .....	26
Perl examples for Acknowledge Queued Items .....	26
XML examples for Acknowledge Queued Items .....	29
Request parameter definitions for Acknowledge Queued Items .....	33
Response parameter definitions for Acknowledge Queued Items .....	34
Process Order examples, returning a 310 code .....	35
Perl examples for Process Order, returning a 310 code .....	35
XML examples for Process Order, returning a 310 code .....	36
Error Codes for real-time/asynchronous processing .....	39
Revisions and Updates .....	40

# Exceptions to Real-Time Processing

Orders for the Email, Digital Certificates, Managed DNS, and Website Builder services are processed in real-time, whenever possible. However, in the rare case that the provider for a particular service is temporarily unavailable, orders may not resolve immediately. Such orders would be queued and the OpenSRS system will monitor and/or resubmit those requests until they have resolved with completed or declined state. Please note that queued orders will not be put in pending state, but will be put in a transient-processing state.

Order items that have been processed asynchronously (i.e. processed after a short delay) will be listed in a queue summary so that Resellers can review processed queued items, and determine if any follow-up actions with their customer are needed. The queue summary is a tool that lists updates of all asynchronous actions (currently for only the Email service, with remaining services to be added in the future). Resellers can use the queue summary to obtain updates on all actions for which the status could not be determined in real time.

Once Resellers have reviewed the update for an item, and have taken the appropriate action on their end (reconciling their own system, or contacting customer), they can acknowledge the item to remove it from the list of notifications. Items that have been removed from the queue summary can always be retrieved using the RWI 2 or API by querying the queue history.

This functionality will be particularly useful for natively asynchronous processes such as ordering digital certificates. The queue summary will be the place where Resellers can find the resolutions for all 'delayed-processing' orders.

**Note:** Asynchronous processing happens very rarely for the Email service. In most cases, orders will be processed when submitted.

## How Resellers can use this functionality

In the case of exceptions to real-time processing, you have some options as to how to handle asynchronously-processed items, and how you want to manage communications with your customers.

**Note:** You do not need to reconcile your system with OpenSRS.

### **1 - Reconcile your systems and communicate with customers later if necessary**

If queue summary results indicate that changes to your records or actions with your customers are necessary, you can update your system, and communicate with your customers at that time. You may also need to update your billing system if you bill automatically. Since exceptions to real-time processing are infrequent, this "reconcile later" method should be adequate for most Resellers.

You only will need to take action if the resolution to an asynchronously-processed item is unexpected. For example, an asynchronously-processed item is considered a successful event by the API, so only if you later find out that the order has failed would you need to take corrective action.

We recommend that you create your own processes for monitoring asynchronously-processed items, for example, you should check the queue every day. Regardless of how or how often you monitor the queue, all notifications will be saved, and will be available next time you query the system.

### **2 - Program your system to respond to 300 family errors**

You can use the API to program your system to respond to 300 family errors. Since exceptions to real-time processing are very rare, this approach would be beneficial only if you have a large volume of transactions.

#### **Examples of how you can take advantage of queue information**

**Order has successfully completed after delay** - If you charge customer immediately, no action is required; however, you might want to inform you customer of when their service will be active.

**Order failed due to incorrect data submitted** - If you suspect a data-entry error, you can resubmit the request.

**Order failed due to unavailability** - You can contact OpenSRS support to clarify why it's unavailable. For example, someone else may have purchased that service without rights (e.g. your client owns domain on which you are trying to obtain an email). If you find a failure to be result of genuine unavailability you can refund your customer or offer an alternate service.

# Scenario Examples

## Typical scenario

### Order item completes successfully, in real-time

1. Reseller places an order for a new service.
2. System contacts supplier.
3. Supplier provisions the service for the order item.
4. System returns a response that indicates that the order was successfully received.
5. Reseller charges their customer, and informs them that the order item was processed.
6. System marks the order item as completed, and stores details about the order item.

## Exceptions

### Example 1 – Order item completes successfully after short delay

1. Reseller places an order for a new service.
2. System finds the supplier is unavailable, returns a response that indicates that the order was successfully received, but queued because of supplier unavailability, and places the order item in the queue.
3. Reseller charges their customer for the account, and informs them that we will process the order item when the supplier is available.
4. Supplier becomes available, and the system submits the order item for processing.
5. Supplier provisions the service for the order item.
6. System marks the order item as completed, and stores details about the order item as an unacknowledged item.
7. Reseller reviews the details in the Queue Summary page and determines that no action is required.
8. Reseller checks the checkbox for this order item, to indicate they have reviewed it.
9. System notes the date and time the Reseller acknowledged the order item result, and no longer returns information.

## **Example 2 – Vendor unable to provision new service, order item fails**

1. Reseller places an order for a new service.
2. System finds the supplier is unavailable, returns a response that indicates that the order was successfully received, but queued because of supplier unavailability, and places the order item in the queue.
3. Reseller charges their customer for the account, and informs them that we will process the order item when the supplier is available.
4. Supplier becomes available, and the system submits the order item for processing.
5. Supplier replies that the order item is not available for provisioning.
6. System marks the order item as declined, and stores details about the order item as an unacknowledged item.
7. Reseller reviews the details in the Queue Summary page, determines that the order item failed, and refunds the customer's money.
8. Reseller checks the checkbox for this order item, to indicate they have reviewed it.
9. System notes the date and time the Reseller acknowledged the order item result, and no longer returns information.

# RWI 2

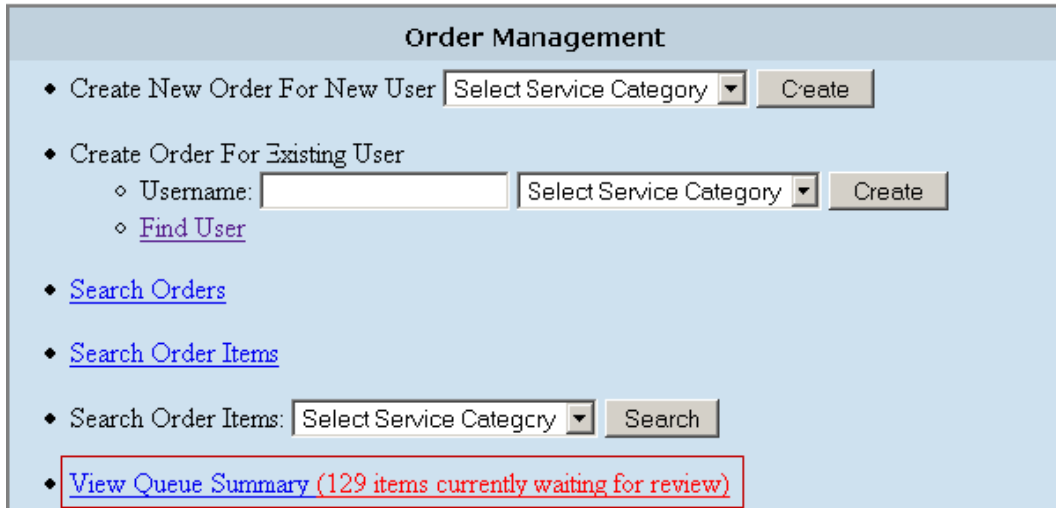
## Viewing the list of your queued items

Items that were processed asynchronously are placed in a queue, and listed on the **Queue Summary** page in the RWI 2. You can access the queue summary from either the **Services Sold Management** page, or the **Order Management** page.

The queue summary is intended to help Resellers manage the asynchronously processed order items that they need to act on. If you've taken action on an order item, e.g., contacted the customer, you can acknowledge it, which will remove it from the queue summary.

### To view the queue summary:

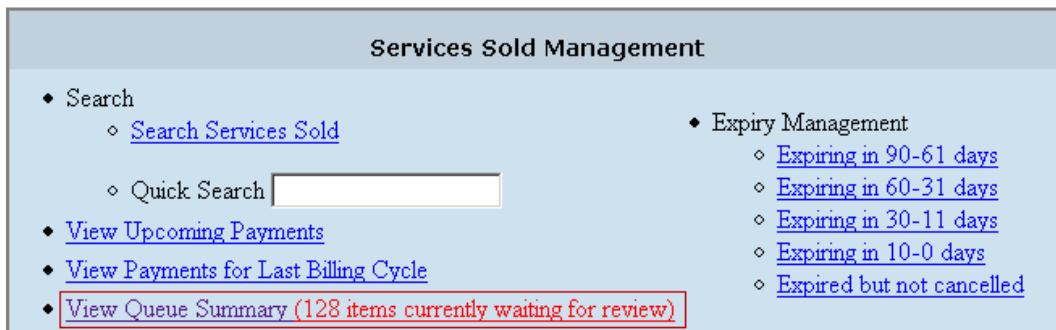
1. In the RWI 2, click either the **Orders** tab or the **Services Sold** tab.



The screenshot shows the 'Order Management' page with a light blue background. It contains several menu items:

- ◆ Create New Order For New User [Select Service Category] [Create]
- ◆ Create Order For Existing User
  - ◊ Username: [input field] [Select Service Category] [Create]
  - ◊ [Find User](#)
- ◆ [Search Orders](#)
- ◆ [Search Order Items](#)
- ◆ Search Order Items: [Select Service Category] [Search]
- ◆ [View Queue Summary \(129 items currently waiting for review\)](#) (highlighted with a red border)

or



The screenshot shows the 'Services Sold Management' page with a light blue background. It contains several menu items:

- ◆ Search
  - ◊ [Search Services Sold](#)
  - ◊ Quick Search [input field]
- ◆ [View Upcoming Payments](#)
- ◆ [View Payments for Last Billing Cycle](#)
- ◆ [View Queue Summary \(128 items currently waiting for review\)](#) (highlighted with a red border)
- ◆ Expiry Management
  - ◊ [Expiring in 90-61 days](#)
  - ◊ [Expiring in 60-31 days](#)
  - ◊ [Expiring in 30-11 days](#)
  - ◊ [Expiring in 10-0 days](#)
  - ◊ [Expired but not cancelled](#)

If you have asynchronously-processed order items that you have not yet acknowledged, and if you have not changed the default settings for the queue summary, the number of items in the queue will be shown in red text.

- Click the **View Queue Summary** link. The summary page opens, listing queued items in two groups: **Total Order Items Found** and **Total Services Sold Found**.

Total Order Items Found: 3							
Pages: 1							
Order Item ID	Service Category	Description	Date Submitted	Date Processed	Action	Result	Reviewed
<a href="#">42850</a>	email	user1@tet-email-aug29.com	29-Aug-2005 09:11:25	29-Aug-2005 09:27:02	Process	Completed	<input type="checkbox"/>
<a href="#">42849</a>	email	ett2@tet-email-aug29.com	29-Aug-2005 09:11:25	29-Aug-2005 09:33:01	Process	Completed	<input type="checkbox"/>
<a href="#">42848</a>	email	tet1@tet-email-aug29.com	29-Aug-2005 09:11:24	29-Aug-2005 09:27:02	Process	Completed	<input type="checkbox"/>

[Select Completed](#)  
[Select Declined](#)  
[Select All](#)

[Search Order Item Queue History](#)

Total Services Sold Found: 4							
Pages: 1							
Product Item ID	Service Category	Description	Date Submitted	Date Processed	Action	Result	Reviewed
<a href="#">31966</a>	email	tet2@tet-email-aug19.com	29-Aug-2005 08:55:15	29-Aug-2005 12:06:00	Activate	Completed	<input type="checkbox"/>
<a href="#">31945</a>	email	tet2@tet-email-aug18-two.com	29-Aug-2005 08:56:15	29-Aug-2005 12:03:01	Activate	Completed	<input type="checkbox"/>
<a href="#">32043</a>	email	tet2@tet-email-aug23.com	29-Aug-2005 09:16:37	29-Aug-2005 09:30:00	Delete	Completed	<input type="checkbox"/>
<a href="#">31851</a>	email	tet1@tet-email-aug17-four.com	22-Aug-2005 10:17:22	22-Aug-2005 10:40:01	Activate	Completed	<input type="checkbox"/>

[Select Completed](#)  
[Select Declined](#)  
[Select All](#)

[Search Services Sold Queue History](#)

**To view the details of a particular item:** click the ID number for that item in the **Order Item ID** column or the **Product Item ID** column.

## Acknowledging queued items

Items that were processed asynchronously are placed in a queue, and listed on the **Queue Summary** page in the RWI 2. The queue summary is intended to help Resellers manage the asynchronously processed order items that they need to act on. If you've taken action on an order item, e.g. contacted the customer, you can acknowledge it, which will remove it from the queue summary.

**To view the queue summary:**

- In the RWI 2, click either the **Orders** tab or the **Services Sold** tab. The **Order Management** page or the **Services Sold Management** page opens.

### Order Management

- ◆ Create New Order For New User
- ◆ Create Order For Existing User
  - Username:
  - [Find User](#)
- ◆ [Search Orders](#)
- ◆ [Search Order Items](#)
- ◆ Search Order Items:
- ◆ [View Queue Summary \(129 items currently waiting for review\)](#)

or

### Services Sold Management

- ◆ Search
  - ◊ [Search Services Sold](#)
  - ◊ Quick Search
- ◆ [View Upcoming Payments](#)
- ◆ [View Payments for Last Billing Cycle](#)
- ◆ [View Queue Summary \(128 items currently waiting for review\)](#)
- ◆ Expiry Management
  - ◊ [Expiring in 90-61 days](#)
  - ◊ [Expiring in 60-31 days](#)
  - ◊ [Expiring in 30-11 days](#)
  - ◊ [Expiring in 10-0 days](#)
  - ◊ [Expired but not cancelled](#)

If you have asynchronously-processed order items that you have not yet acknowledged, and if you have not changed the default settings for the queue summary, the number of items in the queue will be shown in red text.

2. Click the **View Queue Summary** link. The summary page opens, listing queued items in two groups: **Total Order Items Found** and **Total Services Sold Found**.

Total Order Items Found: 3  
Pages: 1

Order Item ID	Service Category	Description	Date Submitted	Date Processed	Action	Result	Reviewed
<a href="#">42850</a>	email	user1@tet-email-aug29.com	29-Aug-2005 09:11:25	29-Aug-2005 09:27:02	Process	Completed	<input type="checkbox"/>
<a href="#">42849</a>	email	ett2@tet-email-aug29.com	29-Aug-2005 09:11:25	29-Aug-2005 09:33:01	Process	Completed	<input type="checkbox"/>
<a href="#">42848</a>	email	tet1@tet-email-aug29.com	29-Aug-2005 09:11:24	29-Aug-2005 09:27:02	Process	Completed	<input type="checkbox"/>

[Select Completed](#)  
[Select Declined](#)  
[Select All](#)

[Search Order Item Queue History](#)

---

Total Services Sold Found: 4  
Pages: 1

Product Item ID	Service Category	Description	Date Submitted	Date Processed	Action	Result	Reviewed
<a href="#">31966</a>	email	tet2@tet-email-aug19.com	29-Aug-2005 08:55:15	29-Aug-2005 12:06:00	Activate	Completed	<input type="checkbox"/>
<a href="#">31945</a>	email	tet2@tet-email-aug18-two.com	29-Aug-2005 08:56:15	29-Aug-2005 12:03:01	Activate	Completed	<input type="checkbox"/>
<a href="#">32043</a>	email	tet2@tet-email-aug23.com	29-Aug-2005 09:16:37	29-Aug-2005 09:30:00	Delete	Completed	<input type="checkbox"/>
<a href="#">31851</a>	email	tet1@tet-email-aug17-four.com	22-Aug-2005 10:17:22	22-Aug-2005 10:40:01	Activate	Completed	<input type="checkbox"/>

[Select Completed](#)  
[Select Declined](#)  
[Select All](#)

[Search Services Sold Queue History](#)

3. Look in the **Description** column to locate the item that you want to acknowledge, then click the checkbox for that item in the **Reviewed** column. Repeat to acknowledge multiple items.
4. Click the **Submit** button(s) when done. The page reloads, and the acknowledged items are removed from the list.

## Finding previously acknowledged items

1. In the RWI 2, click either the **Orders** tab or the **Services Sold** tab. The **Order Management** page or the **Services Sold Management** page opens.

If you have asynchronously processed order items that you have not yet acknowledged, and if you have not changed the default settings for the queue summary, the number of items in the queue will be shown in red text.

2. Click the **View Queue Summary** link. The summary page opens, listing queued items in two groups: **Total Order Items Found** and **Total Services Sold Found**.
3. Click the **Search Order Item Queue History** or the **Search Services Sold Queue History** link. The **Search Order Items Queue History** page or the **Search Services Sold Queue History** page opens.
4. Use the fields and drop-down lists to refine your search. All fields are optional, but more information will return more precise results.
5. Click the **Search Queue History** button. The page reloads, and lists any reviewed (acknowledged) items.

## Changing your queue summary preferences

The **Queue Summary Preferences** page allows you to specify whether completed or declined items, for both order items and services sold, will be listed in the queue summary. By default, all asynchronously processed items are listed in the queue summary.

If you opt-out of a display setting, items of that type will be automatically acknowledged.

These settings also determine whether to display the alert that appears on the **Order Management** and **Services Sold Management** pages. For example, if you set both options for services sold to **No**, the alert will not be shown in the **Services Sold Management** page.

### To change you queue preferences:

1. Click the **Settings** tab.
2. Click the **Manage Queue Summary Preferences** link, located below **Queue Preferences**. The **Queue Summary Preferences** page opens.

**Queue Summary Preferences**

Display Completed Order Items:	<input type="radio"/> Yes <input checked="" type="radio"/> No	<p><b>Note:</b> Changes made to these settings will only apply to orders queued after you save your changes.</p> <p>Any items that have already appeared in the Queue Summary will remain there until you mark them as Reviewed.</p>
Display Declined Order Items:	<input checked="" type="radio"/> Yes <input type="radio"/> No	
Display Completed Service Sold Items:	<input type="radio"/> Yes <input checked="" type="radio"/> No	
Display Declined Service Sold:	<input checked="" type="radio"/> Yes <input type="radio"/> No	

3. Use the radio buttons to determine which queued items will be listed in your queue summary.
4. Click the **Save Changes** button.

# API

## Query Queued Order or Inventory Items

Items that were processed asynchronously are placed in a queue. To list the items in the queue, execute the Query Queued Items command.

The queue is intended to help Resellers manage the asynchronously-processed order items that they need to act on. If you've taken action on an item, e.g. contacted your customer, you can acknowledge it, which will remove it from the queue summary. See the [Acknowledge Queued Items](#) command.

### Topics in this section:

[Perl examples for Query Queued Items](#)

[XML examples for Query Queued Items](#)

[Request parameter definitions for Query Queued Items](#)

[Response parameter definitions for Query Queued Items](#)

## Perl examples for Query Queued Items

### Note:

- The **query\_name** parameter can have either **queued.order\_items** or **queued\_inventory\_item** for its value.
- All conditions are optional and can be used in any combination that is meaningful to you.

### Request 1

```
{
  'protocol' => 'TPP',
  'version' => '1.4.0',
  'action' => 'EXECUTE',
  'object' => 'QUERY',
  'requestor' => {
    'username' => 'default_user'
  },
  'attributes' => {
    'query_name' => 'queued.order_items',
    'page_size' => '5',
    'start_index' => '1',
    'conditions' => [
      {
        'operand' => {
```

```

    'between' => {
      'end' => '02-aug-2005 00:00:00',
      'start' => '01-feb-2005 12:00:00'
    }
  },
  'type' => 'simple',
  'field' => 'submission_date'
},
{
  'link' => 'and',
  'type' => 'link'
},
{
  'operand' => {
    'between' => {
      'end' => '01-jul-2005 00:00:00',
      'start' => '01-jan-2005 00:00:00'
    }
  },
  'type' => 'simple',
  'field' => 'completion_date'
},
{
  'link' => 'and',
  'type' => 'link'
},
{
  'operand' => {
    'eq' => 'email'
  },
  'type' => 'simple',
  'field' => 'service'
},
{
  'link' => 'and',

```

```

        'type' => 'link'
    },
    {
        'operand' => {
            'like' => 'test*.com'
        },
        'type' => 'simple',
        'field' => 'description'
    }
]
}
};

```

## Response 1

```

{
    'protocol' => 'TPP',
    'version' => '1.4.0',
    'action' => 'execute:reply',
    'object' => 'query',
    'session_id' => 'OSRS-5710738',
    'is_success' => '1',
    'response_code' => '200',
    'response_text' => 'Request completed successfully',
    'attributes' => {
        'result_control' => {
            'page_size' => '5',
            'record_count' => '3',
            'start_index' => '1',
            'report_instance_id' => '0'
        },
        'result' => [
            {
                'completion_date' => '28-Jul-2005 17:42:06',
                'acknowledged' => '0',
                'service' => 'email',
                'description' => 'user1@jul28c.com',
            }
        ]
    }
}

```

```
'queued_item_id' => '2411',
'action' => 'process',
'object_id' => '3096479',
'submission_date' => '28-Jul-2005 17:28:52',
'result' => 'completed'
},
{
  'completion_date' => '28-Jul-2005 17:45:03',
  'acknowledged' => '0',
  'service' => 'email',
  'description' => 'user3@jul28b.com',
  'queued_item_id' => '2405',
  'action' => 'process',
  'object_id' => '3096476',
  'submission_date' => '28-Jul-2005 17:20:02',
  'result' => 'completed'
},
{
  'completion_date' => '22-Jul-2005 10:30:13',
  'acknowledged' => '0',
  'service' => 'email',
  'description' => 'user1@jul21c.com',
  'queued_item_id' => '774',
  'action' => 'process',
  'object_id' => '3096287',
  'submission_date' => '21-Jul-2005 14:28:18',
  'result' => 'completed'
}
]
}
};
```

## Request 2

```

<?{
  'protocol' => 'TPP',
  'action' => 'execute',
  'object' => 'query',
  'version' => '1.4.0',
  'requestor' => {
    'password' => 'secret',
    'username' => 'orange'
  },
  'attributes' => {
    'page_size' => '10',
    'query_name' => 'inventory_item.by_id',
    'start_index' => '1',
    'conditions' => [
      {
        'operand' => {
          'eq' => '32001'
        },
        'type' => 'simple',
        'field' => 'inventory_item_id'
      }
    ]
  }
};

```

## Response 2 with 510 error

```

{
  'protocol' => 'TPP',
  'session_id' => 'OSRS-295268',
  'object' => 'QUERY',
  'version' => '1.4.0',
  'response_code' => '510',
  'is_success' => '0',
  'response_text' => 'Supplier is unavailable;',

```

```

'action' => 'EXECUTE:REPLY',
'attributes' => {
  'result_control' => {},
  'result' => []
}
};

```

## XML examples for Query Queued Items

### Note:

- The `query_name` parameter can have either `queued.order_items` or `queued_inventory_item` for its value.
- All conditions are optional and can be used in any combination that is meaningful to you.

### Request 1

```

<?xml version='1.0' encoding="UTF-8" standalone="no" ?>
<!DOCTYPE OPS_envelope SYSTEM "ops.dtd">
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">EXECUTE</item>
        <item key="object">QUERY</item>
        <item key="requestor">
          <dt_assoc>
            <item key="username">default_user</item>
          </dt_assoc>
        </item>
        <item key="attributes">
          <dt_assoc>
            <item key="query_name">queued.order_items</item>
            <item key="start_index">1</item>
          </dt_assoc>
        </item>
      </dt_assoc>
    </data_block>
  </body>
</OPS_envelope>

```

```

<item key="page_size">5</item>
<item key="conditions">
  <dt_array>
    <item key="0">
      <dt_assoc>
        <item key="field">submission_date</item>
        <item key="operand">
          <dt_assoc>
            <item key="between">
              <dt_assoc>
                <item key="start">01-feb-2005
12:00:00</item>
                <item key="end">02-aug-2005 00:00:00</item>
              </dt_assoc>
            </item>
          </dt_assoc>
        </item>
      </dt_assoc>
      <item key="type">simple</item>
    </dt_assoc>
  </item>
  <item key="1">
    <dt_assoc>
      <item key="link">and</item>
      <item key="type">link</item>
    </dt_assoc>
  </item>
  <item key="2">
    <dt_assoc>
      <item key="field">completion_date</item>
      <item key="operand">
        <dt_assoc>
          <item key="between">
            <dt_assoc>
              <item key="start">01-jan-2005
00:00:00</item>
              <item key="end">01-jul-2005 00:00:00</item>
            </dt_assoc>
          </item>
        </dt_assoc>
      </item>
    </dt_assoc>
  </item>
</dt_array>
</item>

```

```

        </dt_assoc>
    </item>
</dt_assoc>
</item>
    <item key="type">simple</item>
</dt_assoc>
</item>
<item key="3">
    <dt_assoc>
        <item key="link">and</item>
        <item key="type">link</item>
    </dt_assoc>
</item>
<item key="4">
    <dt_assoc>
        <item key="operand">
            <dt_assoc>
                <item key="eq">email</item>
            </dt_assoc>
        </item>
        <item key="field">service</item>
        <item key="type">simple</item>
    </dt_assoc>
</item>
<item key="5">
    <dt_assoc>
        <item key="link">and</item>
        <item key="type">link</item>
    </dt_assoc>
</item>
<item key="6">
    <dt_assoc>
        <item key="operand">
            <dt_assoc>
                <item key="like">test*.com</item>
            </dt_assoc>
        </item>
    </dt_assoc>
</item>

```

```

        </dt_assoc>
      </item>
      <item key="field">description</item>
      <item key="type">simple</item>
    </dt_assoc>
    <dt_assoc>
      <item key="operand">
        <dt_assoc>
          <item key="eq">1</item>
        </dt_assoc>
      </item>
      <item key="field">acknowledged</item>
      <item key="type">simple</item>
    </dt_assoc>
  </item>
</dt_array>
</item>
</dt_assoc>
</item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Response 1

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_en<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">execute:reply</item>

```

```

<item key="object">query</item>
<item key="session_id">OSRS-5710738</item>
<item key="is_success">1</item>
<item key="response_text">Request completed successfully</item>
<item key="response_code">200</item>
<item key="attributes">
  <dt_assoc>
    <item key="result_control">
      <dt_assoc>
        <item key="record_count">3</item>
        <item key="page_size">5</item>
        <item key="start_index">1</item>
        <item key="report_instance_id">0</item>
      </dt_assoc>
    </item>
    <item key="result">
      <dt_array>
        <item key="0">
          <dt_assoc>
            <item key="submission_date">28-Jul-2005
17:28:52</item>
            <item key="action">process</item>
            <item key="acknowledged">0</item>
            <item key="completion_date">28-Jul-2005
17:42:06</item>
            <item key="object_id">3096479</item>
            <item key="description">user1@jul28c.com</item>
            <item key="result">completed</item>
            <item key="service">email</item>
            <item key="queued_item_id">2411</item>
          </dt_assoc>
        </item>
        <item key="1">
          <dt_assoc>
            <item key="submission_date">28-Jul-2005
17:20:02</item>

```

```

        <item key="action">process</item>
        <item key="acknowledged">0</item>
        <item key="completion_date">28-Jul-2005
17:45:03</item>

        <item key="object_id">3096476</item>
        <item key="description">user3@jul28b.com</item>
        <item key="result">completed</item>
        <item key="service">email</item>
        <item key="queued_item_id">2405</item>
    </dt_assoc>
</item>
<item key="2">
    <dt_assoc>
        <item key="submission_date">21-Jul-2005
14:28:18</item>

        <item key="action">process</item>
        <item key="acknowledged">0</item>
        <item key="completion_date">22-Jul-2005
10:30:13</item>

        <item key="object_id">3096287</item>
        <item key="description">user1@jul21c.com</item>
        <item key="result">completed</item>
        <item key="service">email</item>
        <item key="queued_item_id">774</item>
    </dt_assoc>
</item>
</dt_array>
</item>
</dt_assoc>
</item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Request 2

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">execute</item>
        <item key="object">query</item>
        <item key="requestor">
          <dt_assoc>
            <item key="password">secret</item>
            <item key="username">orange</item>
          </dt_assoc>
        </item>
        <item key="attributes">
          <dt_assoc>
            <item key="start_index">1</item>
            <item key="query_name">inventory_item.by_id</item>
            <item key="page_size">10</item>
            <item key="conditions">
              <dt_array>
                <item key="0">
                  <dt_assoc>
                    <item key="operand">
                      <dt_assoc>
                        <item key="eq">32001</item>
                      </dt_assoc>
                    </item>
                    <item key="type">simple</item>
                    <item key="field">inventory_item_id</item>
                  </dt_assoc>
                </item>
              </dt_array>
            </item>
          </dt_assoc>
        </item>
      </dt_assoc>
    </data_block>
  </body>
</OPS_envelope>

```

```

        </dt_assoc>
      </item>
    </dt_array>
  </item>
</dt_assoc>
</item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Response 2 with 510 error

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">EXECUTE:REPLY</item>
        <item key="object">QUERY</item>
        <item key="session_id">OSRS-295268</item>
        <item key="is_success">0</item>
        <item key="response_code">510</item>
        <item key="response_text">Supplier is unavailable;</item>
        <item key="attributes">
          <dt_assoc>
            <item key="result">
              <dt_array/>
            </item>
            <item key="result_control">
              <dt_assoc/>
            </item>
          </dt_assoc>
        </item>
      </dt_assoc>
    </data_block>
  </body>
</OPS_envelope>

```

```

        </dt_assoc>
    </item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Request parameter definitions for Query Queued Items

### Standard parameters

**action** = execute

**object** = query

### attributes parameter

Parameters within the **attributes** associative array are described below.

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>query_name</b>	int	name of query or report available in the system	required	<b>attributes</b> assoc. array	<b>queued.order_items</b> or <b>queued_inventory_item</b>
<b>start_index</b>	int	Tells the system which record to start with in the result list. This must be '1' the first time you run the query. On subsequent runs, you can set this to the record you want to start with.	optional	<b>attributes</b> assoc. array	int
<b>page_size</b>	int	Tells the system how many records to return to you in the list. If you do not include	optional	<b>attributes</b> assoc. array	int

Parameter Name	Type	Description	Obligation	Location	Allowed Values
		this parameter, a default page size of 50 will be used. The maximum page size allowed is 50.			
<b>conditions</b>	array of assoc. arrays	contains parameters for the query to be executed	required	<b>attributes</b> assoc. array	values are variable; see parameters below
<b>field</b>	string	name of field being used in the query	optional	<b>conditions</b> array of assoc. arrays	description   state   service   username   submission_date   completion_date   acknowledged
<b>operand</b>	string	indicates which operand to use	optional	<b>conditions</b> array of assoc. arrays	eq = equal to the given value neq = not equal to the given value leq = less than or equal to the given value geq = greater than or equal to the given value between = between the given values
<b>type</b>	string	type of condition	optional	<b>conditions</b> array of assoc. arrays	simple   link
<b>link</b>	string	determines how the conditions are used together	optional	<b>conditions</b> array of assoc. arrays	and   or

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>start</b>	string	start value to query between	required if <b>between</b> parameter submitted	<b>between</b> parameter	string
<b>end</b>	string	end value to query between	required if <b>between</b> parameter submitted	<b>between</b> parameter	string

## Response parameter definitions for Query Queued Items

### Standard parameters

**action** = execute:reply

**object** = query

### attributes parameter

Parameters within the **attributes** associative array are described below.

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>result_control</b>	assoc. array	array that determines how query results will be returned	always returned	<b>attributes</b> assoc. array	see below
<b>record_count</b>	int	the number of records returned that match the query criteria	always returned	<b>result_control</b> assoc. array	int
<b>start_index</b>	int	the number of the record that the results are starting with	returned if specified in request	<b>result_control</b> assoc. array	int
<b>page_size</b>	int	the number of records returned per page	returned if specified in request	<b>result_control</b> assoc. array	int
<b>result</b>	array of associative arrays	contains arrays with details for each queued item	always returned	<b>attributes</b> assoc. array	see below

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>service</b>	string	the service provisioning the product	always returned	<b>result</b> assoc. array	string
<b>description</b>	string	Reseller-supplied reference id for use with Reseller's customer and data management systems	returned if applicable	<b>result</b> assoc. array	string
<b>queued_item_id</b>	int	ID of the queued item (not the order item ID or inventory item ID)	always returned	<b>result</b> assoc. array	int
<b>acknowledged</b>	Boolean	flag that indicates if item has been acknowledged. See <a href="#">Acknowledge Queued Items</a> .	always returned	<b>result</b> assoc. array	1   0
<b>submission_date</b>	string	date that the item was submitted for processing	always returned	<b>result</b> assoc. array	string
<b>completion_date</b>	string	date that the item was processed	always returned	<b>result</b> assoc. array	string
<b>action</b>	string	action that was executed	always returned	<b>result</b> associative array	process   delete   activate   suspend
<b>object_id</b>	int	the ID of the object	always returned	<b>result</b> assoc. array	int, valid object ID
<b>result</b>	string	indicates if the item completed	always returned	<b>result</b> assoc. array	completed   declined

In addition, the state **provisioning-requesting** has been added for inventory items.

## Acknowledge Queued Items

Items that were processed asynchronously are placed in a queue. To list the items in the queue, execute the Query Queued Order Items command.

The queue is intended to help Resellers manage the asynchronously-processed order items that they need to act on. If you've taken action on an order item, e.g. contacted your customer, you can acknowledge it, which will remove it from the queue summary.

**Note:** If you acknowledge an already acknowledged item, a success response will be returned.

### Topics in this section:

[Perl examples for Acknowledge Queued Item](#)

[XML examples for Acknowledge Queued Item](#)

[Request parameter definitions for Acknowledge Queued Item](#)

[Response parameter definitions for Acknowledge Queued Item](#)

## Perl examples for Acknowledge Queued Items

### Request 1

```
{
  'protocol' => 'TPP',
  'version' => '1.4',
  'action' => 'ack',
  'object' => 'queued_item',
  'requestor' => {
    'username' => '10group'
  },
  'attributes' => {
    'user_id' => '5209'
    'queued_items' => [
      {
        'queued_item_id' => '537'
      },
      {
        'queued_item_id' => '538'
      }
    ],
  }
};
```

**Response 1**

```

{
  'protocol' => 'TPP',
  'version' => '1.4.0',
  'action' => 'ack:reply',
  'object' => 'queued_item',
  'session_id' => 'OSRS-5710734',
  'is_success' => '1',
  'response_code' => '200',
  'response_text' => 'Request completed successfully',
  'attributes' => {
    'queued_items' => [
      {
        'response_text' => 'Request completed successfully',
        'response_code' => '200',
        'queued_item_id' => '537'
      },
      {
        'response_text' => 'Request completed successfully',
        'response_code' => '200',
        'queued_item_id' => '538'
      }
    ]
  }
};

```

**Request 2**

```

{
  'protocol' => 'TPP',
  'object' => 'queued_item',
  'version' => '1.4.0',
  'action' => 'ack',
  'requestor' => {
    'username' => 'orange'
  },

```

```

'attributes' => {
  'queued_items' => [
    {
      'queued_item_id' => '23232'
    }
  ],
  'user_id' => '1003'
}
};

```

## Response 2 with 3004 error

```

{
  'protocol' => 'TPP',
  'session_id' => 'OSRS-295263',
  'object' => 'QUEUED_ITEM',
  'version' => '1.4.0',
  'response_code' => '3004',
  'is_success' => '0',
  'response_text' => 'Unable to update queued item',
  'action' => 'ACK:REPLY',
  'attributes' => {
    'queued_items' => [
      {
        'response_text' => 'Unable to update queued item : Queued
item [id=23232] not found.',
        'response_code' => '3004',
        'queued_item_id' => '23232'
      }
    ]
  }
}
};

```

## XML examples for Acknowledge Queued Items

### Request 1

```
<?xml version='1.0' encoding="UTF-8" standalone="no" ?>
<!DOCTYPE OPS_envelope SYSTEM "ops.dtd">
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4</item>
        <item key="action">ack</item>
        <item key="object">queued_item</item>
        <item key="requestor">
          <dt_assoc>
            <item key="username">10group</item>
          </dt_assoc>
        </item>
        <item key="attributes">
          <dt_assoc>
            <item key="user_id">5209</item>
            <item key="queued_items">
              <dt_array>
                <item key="0">
                  <dt_assoc>
                    <item key="queued_item_id">537</item>
                  </dt_assoc>
                </item>
                <item key="1">
                  <dt_assoc>
                    <item key="queued_item_id">538</item>
                  </dt_assoc>
                </item>
              </dt_array>
            </item>
          </dt_assoc>
        </item>
      </dt_assoc>
    </data_block>
  </body>
</OPS_envelope>
```

```

        </item>
      </dt_array>
    </item>
  </dt_assoc>
</item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Response 1

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">ack:reply</item>
        <item key="object">queued_item</item>
        <item key="session_id">OSRS-5710734</item>
        <item key="is_success">1</item>
        <item key="response_text">Request completed successfully</item>
        <item key="response_code">200</item>
        <item key="attributes">
          <dt_assoc>
            <item key="queued_items">
              <dt_array>
                <item key="0">
                  <dt_assoc>
                    <item key="response_text">Request completed
successfully</item>
                    <item key="response_code">200</item>

```

```

        <item key="queued_item_id">537</item>
      </dt_assoc>
    </item>
    <item key="1">
      <dt_assoc>
        <item key="response_text">Request completed
successfully</item>
        <item key="response_code">200</item>
        <item key="queued_item_id">538</item>
      </dt_assoc>
    </item>
  </dt_array>
</item>
</dt_assoc>
</item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Request 2

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">ack</item>
        <item key="object">queued_item</item>
        <item key="requestor">
          <dt_assoc>
            <item key="username">orange</item>

```

```

    </dt_assoc>
  </item>
  <item key="attributes">
    <dt_assoc>
      <item key="user_id">1003</item>
      <item key="queued_items">
        <dt_array>
          <item key="0">
            <dt_assoc>
              <item key="queued_item_id">23232</item>
            </dt_assoc>
          </item>
        </dt_array>
      </item>
    </dt_assoc>
  </item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Response 2 with 3004 error

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="action">ACK:REPLY</item>
        <item key="object">QUEUED_ITEM</item>
        <item key="session_id">OSRS-295263</item>
        <item key="is_success">0</item>
      </dt_assoc>
    </data_block>
  </body>
</OPS_envelope>

```

```

<item key="response_code">3004</item>
<item key="response_text">Unable to update queued item</item>
<item key="attributes">
  <dt_assoc>
    <item key="queued_items">
      <dt_array>
        <item key="0">
          <dt_assoc>
            <item key="queued_item_id">23232</item>
            <item key="response_code">3004</item>
            <item key="response_text">Unable to update queued
item : Queued item [id=23232] not found.</item>
          </dt_assoc>
        </item>
      </dt_array>
    </item>
  </dt_assoc>
</item>
</data_block>
</body>
</OPS_envelope>

```

## Request parameter definitions for Acknowledge Queued Items

### Standard parameters

**action** = ack

**object** = queued\_item

### attributes parameter

Parameters within the **attributes** associative array are described below.

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>user_id</b>	int	ID of the user	required	<b>attributes</b> associative array	valid user ID

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>queued_items</b>	assoc. array	Lists asynchronously processed items that have not been acknowledged	required	<b>attributes</b> assoc. array	<b>queued_item_id</b> parameter
<b>queued_item_id</b>	string	The ID of the queued item to acknowledge	required	<b>queued_items</b> assoc. array	valid item ID

## Response parameter definitions for Acknowledge Queued Items

### Standard parameters

**action** = ack:reply

**object** = queued\_item

### attributes parameter

Parameters within the **attributes** associative array are described below.

Parameter Name	Type	Description	Obligation	Location	Allowed Values
<b>queued_items</b>	assoc. array	Lists asynchronously processed items that have not been acknowledged	required	<b>attributes</b> assoc. array	<b>queued_item_id</b> parameter
<b>queued_item_id</b>	string	ID of queued item that has not been acknowledged	required	<b>queued_items</b> assoc. array	valid item ID

In addition, the state **provisioning-requesting** has been added for inventory items.

## Process Order examples, returning a 310 code

### Perl examples for Process Order, returning a 310 code

#### Request

```
{
  'protocol' => 'TPP',
  'version' => '1.4.0',
  'object' => 'order',
  'action' => 'process',
  'requestor' => {
    'password' => 'secret',
    'username' => 'orange'
  },
  'attributes' => {
    'order_id' => '41621'
  }
};
```

#### Response

```
{
  'protocol' => 'TPP',
  'object' => 'ORDER',
  'session_id' => 'OSRS-295267',
  'version' => '1.4.0',
  'response_code' => '310',
  'is_success' => '1',
  'response_text' => 'Request successfully submitted, and placed in
queue because of supplier unavailability.',
  'action' => 'PROCESS:REPLY',
  'attributes' => {
    'client_reference' => 'order created by create.order.save module',
    'create_items' => [
      {
        'client_reference' => '',
```

```

    'status' => 'provisioning-requesting',
    'major_code' => '310',
    'term' => '1',
    'item_id' => '43181',
    'product_item' => {
      'object_type' => 'email',
      'service' => 'email'
    },
    'contact_set' => {
      'owner' => '102'
    },
    'ancillary_price' => '0',
    'price' => '40',
    'major_text' => 'Request successfully submitted, and placed in
queue because of supplier unavailability.'
  }
],
'status' => 'fulfilled',
'order_id' => '41621',
'price' => '40'
}
};

```

## XML examples for Process Order, returning a 310 code

### Request

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>

```

```

<item key="action">process</item>
<item key="object">order</item>
<item key="requestor">
  <dt_assoc>
    <item key="password">secret</item>
    <item key="username">orange</item>
  </dt_assoc>
</item>
<item key="attributes">
  <dt_assoc>
    <item key="order_id">41621</item>
  </dt_assoc>
</item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Response

```

<?xml version='1.0' encoding='UTF-8'?>
<OPS_envelope>
  <header>
    <version>0.9</version>
  </header>
  <body>
    <data_block>
      <dt_assoc>
        <item key="protocol">TPP</item>
        <item key="version">1.4.0</item>
        <item key="session_id">OSRS-295267</item>
        <item key="action">PROCESS:REPLY</item>
        <item key="object">ORDER</item>
        <item key="is_success">1</item>
        <item key="response_code">310</item>
        <item key="response_text">Request successfully submitted, and
placed in queue because of supplier unavailability.</item>

```

```

<item key="attributes">
  <dt_assoc>
    <item key="status">fulfilled</item>
    <item key="create_items">
      <dt_array>
        <item key="0">
          <dt_assoc>
            <item key="status">provisioning-requesting</item>
            <item key="contact_set">
              <dt_assoc>
                <item key="owner">102</item>
              </dt_assoc>
            </item>
            <item key="major_code">310</item>
            <item key="ancillary_price">0</item>
            <item key="item_id">43181</item>
            <item key="term">1</item>
            <item key="client_reference"/>
            <item key="price">40</item>
            <item key="product_item">
              <dt_assoc>
                <item key="service">email</item>
                <item key="object_type">email</item>
              </dt_assoc>
            </item>
            <item key="major_text">Request successfully
submitted, and placed in queue because of supplier
unavailability.</item>
          </dt_assoc>
        </item>
      </dt_array>
    </item>
    <item key="order_id">41621</item>
    <item key="client_reference">order created by
create.order.save module</item>
    <item key="price">40</item>

```

```

        </dt_assoc>
    </item>
</dt_assoc>
</data_block>
</body>
</OPS_envelope>

```

## Error Codes for real-time/asynchronous processing

### 300 family error codes

These error codes apply to real-time/asynchronous processing, and indicate that items have been queued. These errors apply to all billable APIs, i.e.:

- process/order
- create/order (only if when handling = 'process' which means the order will get processed as well)
- delete/inventory\_item
- suspend/inventory\_item
- activate/inventory\_item

310 = In progress. Supplier is unavailable.

320 = In progress. System is busy.

330 = In progress. Action is temporarily unavailable.

340 = In progress. Unexpected error to be resolved.

350 = In progress. Request successfully submitted and placed in the queue.

### 500 family error codes

These new error codes indicate permanent and immediate failures, and will not result in item queuing. These errors apply to all non-billable APIs.

510 = Supplier is unavailable.

520 = System is busy.

530 = Action is unavailable.

540 = Unexpected error.

### 3000 family error code

This new error code indicates a permanent and immediate failure.

3004 = Unable to update queued item.

# Revisions and Updates

## **July 28, 2008**

Rebranded guide with new OpenSRS logo.

## **December 8, 2005**

Functionality available on the Horizon test system for the Digital Certificates, Managed DNS, Email Defense, and Website Builder services.

## **December 15, 2005**

Exceptions to Real-Time Processing, identified enhanced provisioning of additional TPP products to include additional synchronous processes: Digital Certificates, Managed DNS, Email Defense, and Website Builder.

Horizon test environment availability: December 8, 2005

Live: December 15, 2005.

## **August 31, 2005**

Pre-release of this document; functionality not yet available.

October 13, 2005: Functionality available on the Horizon test system.

October 19, 2005: Functionality available live.

Real-time processing and asynchronous event management will be available for the remaining services in coming months.