UPS ATRP Interface Reference

11/05/2014

Revision 2.0

Keith Musslewhite

Stan Gordon

PSG SC Engineering and Quality
Hewlett-Packard Company

The information contained herein is subject to change without notice. The only warranties for HP Products and services are set forth in the express warranty statements accompanying such Products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.
Microsoft, Windows, and Windows NT are U.S. registered trademarks of Microsoft Corporation.
Printed in the US

Contents

Purpose

This document describes a machine-to-machine interface between UPS and third party UI applications for the purpose of printing asset tags. It describes what is required to use it, how to make requests and how to process the response. This document should be the main guideline when implementing the UPS ATRP Web Service.

Audience

The indented reader of this document is anyone that wishes to implement and set up machine-to-machine communication with UPS to print asset tag in accordance with HP standards and customer supplied requirements. The reader is required to have a basic knowledge of how web services work and should feel confident using terms like SOAP and XML.

Use of this guide assumes you are familiar with the following:

- XML (for an overview, go to *W3 Schools XML Tutorial)*
- Basic understanding of web services (for an overview, go to *W3 Schools Web Services Tutorial)*
- A programming language for consuming a web service and any related tools

Overview

This document describes the UPS ATRP Interface web service and how to set up machine-to-machine communications.

When using ATRP, all user interface and interactions must be performed by the application consuming the ATRP services. Error handling and messaging of steps / actions to perform must be provided by the application consuming ATRP.

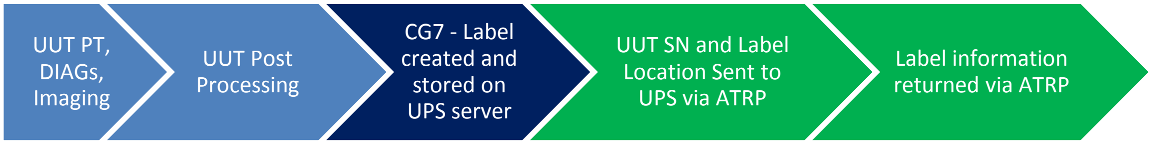
ATRP will provide the Bartender BTW file and artwork for placing the label as part of information returned as part of the service.

Prerequisites

In some cases, use of the ATRP interface process requires that the UPS UUT client be executed on the UUT prior to label printing in order to create the label information for that unit and store it on the UPS site server. This is normally accomplished during the run-in phase via CG7 components.

Timing

General flow (UPS UUT client shown in dark blue, ATRP interface portions shown in green boxes):


UUT PT, DIAGs, Imaging
UUT Post Processing
CG7 - Label created and stored on UPS server
UUT SN and Label Location Sent to UPS via ATRP
Label information returned via ATRP

Interface details

Web Methods

This section describes the WEB Methods available in the WEB ATRP Web service.

[WebMethod]public ATRPStruct UPSGetATRP(string SN, string Location)

Parameters

SN = serial number

Location = target of the asset tag (valid values are 'chassis' or 'shipping'. Case insensitive)

Returns: ATRPStruct as defined below.

```
public struct ATRPStruct
{
   public string version;
   public int retcode;
   public string message;
   public DataSet tagdata;
}
```

version – string containing web service version number

retcode – integer containing a return value

0 = success

-1 = failure

message – string containing error message (if applicable)

tagdata - dataset consisting of four columns and an unspecified amount of rows

- An empty dataset indicates there is no label information associated with that SN / location combination.

Each row of the dataset consists of the following data items:

Column 0: byte string representing the Bartender format file

Column 1: comma delimited string of data to send to Bartender format file

Column 2: byte string representing a .jpeg image showing placement of asset tag

Column 3: HP part number for print stock

[WebMethod]public ATRPStruct UPSGetTags(string SN, string Location)

Parameters

SN = serial number

Location = target of the asset tag (valid values are 'chassis' or 'shipping'. Case insensitive)

Returns: ATRPStruct as defined below.

```
public struct ATRPStruct
{
   public string version;
   public int retcode;
   public string message;
   public DataSet tagdata;
}
```

version – string containing web service version number
retcode – integer containing a return value
 0 = success
-1 = failure
message – string containing error message (if applicable)
tagdata - dataset consisting of two columns and an unspecified amount of rows

- An empty dataset indicates there is no label information associated with that SN / location combination.

Each row of the dataset consists of the following data items:
Column 0: byte string representing the Bartender format file
Column 1: comma delimited string of data to send to Bartender format file

[WebMethod]public ATRPStruct UPSGetImages(string SN, string Location)

Parameters

SN = serial number
Location = target of the asset tag (valid values are ‘chassis’ or ‘shipping’. Case insensitive)

Returns: ATRPStruct as defined below.

```
public struct ATRPStruct
{
   public string version;
   public int retcode;
   public string message;
   public DataSet tagdata;
}
```

version – string containing web service version number

retcode – integer containing a return value

 0 = success

-1 = failure

message – string containing error message (if applicable)

tagdata - dataset consisting of one column and an unspecified amount of rows

- An empty dataset indicates there is no label information associated with that SN / location combination.

Each row of the dataset consists of the following data item:

Column 0: byte string representing a .jpeg image showing placement of asset tag

[WebMethod]public ATRPStruct UPSGetATM (string SerialNumber, string PartNumber, string AssetNumber, string HPPO, string AssetTagNumber, string MAC, string MAC2, string SystemID, string Placement, string TagData)

Parameters

SerialNumber = Serial number of the UUT to be processed

PartNumber = Customer specific AV asset tagging part number found on BOM

AssetNumber = Name of the asset number as defined by CIS Engineering.

HPPO = Order reference number

AssetNumber = Asset Number to be assigned either by Site or by UPS tool

MAC = Mac Address of the UUT to be processed

MAC2 = Mac address (optional) just in case needed

SystemID = System ID of the UUT to be processed, needed to identify SOP image to be shown

Placement = target of the asset tag (valid values are ‘chassis’ or ‘shipping’. Case insensitive)

TagData = String that will contain the BarTender Fields to be printed in the Tag separated by commas

Returns: ATRPStruct as defined below.

```
public struct ATMStruct
{
   public string version;
```

```
  public int retcode;
  public string message;
  public string assetTagNum;
}
```

version – string containing web service version number
retcode – integer containing a return value
 0 = success
-1 = failure
 1 = success, but means the Serial Number was already processed by UPS.
message – string containing error or success message
assetTagNum - string containing the asset tag number that was assigned to the UUT.

[WebMethod]public ATRPStruct UPSGetUSI(string SerialNumber, string PartNumber, string HPPO)

Parameters

SerialNumber = Serial number of the UUT to be processed
PartNumber = Customer specific AV asset tagging part number found on BOM
HPPO = Order reference number

Returns: ATRPStruct as defined below.

```
public struct USIStruct
{
  public string version;
  public int retcode;
  public string message;
  public XmlDocument unattend;
}
```

version – string containing web service version number
retcode – integer containing a return value
 0 = success
-1 = failure
message – string containing error or success message
unattend – Xml document to be retrieved containing all the Unit personalization to be injected in the UUT processed.

[WebMethod]public ATRPStruct UPSGetRange(string PartNumber, string AssetNumber, string HPPO)

Parameters

PartNumber = customer specific AV asset tagging part number found on BOM
AssetNumber = name of the asset number as defined by CIS Engineering.
HPPO = order reference number

Assumptions

- Site has already notified UPS of the order using one of the “SendBOM” web methods available in the UPS Primary Interface (PUPS)

- The Site allows sufficient time for UPS to service the range requests
- HPPO is the same reference number used in the HPPO parameter in previous SendBOM call
- Site has possession of the asset number name of the requested asset number (defined by CIS Engineering and available via UPSConsole application)
- Site will be responsible for generating the actual customer asset number by applying sequential range values to the supplied asset number mask.
- Once processed by this web method, range records in UPS site database will be marked as depleted inventory (0% available).

Returns: ATRPStruct as defined below.

```
public struct ATRPStruct
{
   public string version;
   public int retcode;
   public string message;
   public DataSet tagdata;
}
```

version – string containing web service version number
retcode – integer containing a return value
 0 = success
-1 = failure
message – string containing error message (if applicable)
tagdata - dataset consisting of three columns and an unspecified amount of rows

  - An empty dataset indicates there is no data in the site UPS database associated with the supplied parameter combination.

Each row of the dataset consists of the following data items:
Column 0: integer representing a starting value of the range
Column 1: integer representing an ending value of the range
Column 2: string representing the asset number mask

Risks

- Subsequent calls to this web method with the same parameter values will always result in an empty dataset.

Configuration

Configuration of the ATRP web service is controlled by the file web.config. This file exists in the parent directory of the ATRP directory. The values in the appSettings section need to be configured for each installation site.

Example web.config file:

```
<?xml version="1.0"?>
<configuration>
  <appSettings>
    <add key="UPSServerName" value="<UPSDBServer>"></add>
```

```
      <add key="UPSServerIP" value=" <UPSDBServer IP>"></add>
      <add key="UPSUser" value="<UPS administrator login name>"></add>
      <add key="UPSPwd" value="<Encrypted password for UPSUser>"></add>
  </appSettings>
 <connectionStrings/>
 <system.web>
    <compilation debug="false"/>
    <authentication mode="Windows"/>
    <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
      <error statusCode="403" redirect="NoAccess.htm" />
      <error statusCode="404" redirect="FileNotFound.htm" />
    </customErrors>
   </system.web>
</configuration
```

Using the web service in production

This section describes the UPS ATRP interface web service production environment and how it should be used

Sample web service consumer for UPSGetATRP(c#)

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data;
using System.Data.SqlClient;
using System.IO;


public struct ATRPStruct
{
 public string version;
 public int retcode;
 public string message;
 public DataSet tagdata;
}

namespace UPS_ATRPConsumer
{
 class Program
 {
  static void Main(string[] args)
  {
   byte[] Bartenderbytes = { };
   BinaryWriter bw;
   FileStream fs;

   localhost.ATRP atrp = new localhost.ATRP();

   string serialnumber = "100001";
   string location = "Chassis";

   UPS_ATRPConsumer.localhost.ATRPStruct retval = atrp.UPSGetATRP(serialnumber, location);
   if (retval.retcode == 0)
   {
    if (retval.tagdata.Tables["AssetTags"].Rows.Count > 0)
    {
     foreach (DataRow dr in retval.tagdata.Tables["AssetTags"].Rows)
     {
      // handle bartender file
      Bartenderbytes = (byte[])dr[0];
      if (File.Exists(serialnumber + ".btw")) File.Delete(serialnumber + ".btw");
      fs = new FileStream(serialnumber + ".btw", FileMode.CreateNew);
      bw = new BinaryWriter(fs);
      bw.Write(Bartenderbytes);
      bw.Close();
      fs.Close();
      // read data for sending to Bartender
      string data = Convert.ToString(dr[1]);
      // handle placement image
      if (dr[2].ToString().Length > 0)
      {
       byte[] placement = (byte[])dr[2];
       fs = new FileStream(serialnumber + ".jpg", FileMode.CreateNew);
       bw = new BinaryWriter(fs);
       bw.Write(placement);
       bw.Close();
       fs.Close();
      }
      // get the printer stock part number
      string printstock = Convert.ToString(dr[3]);
      // do the printing
     }
    }
    else
    {
     // no tag data available for SN/Location combination
    }
   }
   else
   {
    // error condition - read retval.message
   }
  }
 }
}
```

Sample code for applying a number to UPS asset number mask (c#)

```
private string GetAssetNumber(string mask, int currvalue)
{
  int len;
  string currstr = String.Empty;

  int ptr1=0, ptr2=0;
  string newval = String.Empty;
  string prefix = String.Empty;
  string suffix = String.Empty;
  string zeros = @"00000000000000000000";

  ptr1 = mask.IndexOf('%', 0);
  ptr2 = mask.IndexOf('d', ptr1);

  if (ptr1 > 0) prefix = mask.Substring(0, ptr1);
  if (ptr2 + 1 < mask.Length) suffix = mask.Substring(ptr2 + 1);

  currstr = Convert.ToString(currvalue);

  if (ptr2 - ptr1 > 1)
    len = Convert.ToInt32(mask.Substring(ptr1 + 1, ptr2 - ptr1 - 1));
  else
    len = currstr.Length;

  newval = zeros.Substring(0, len - currstr.Length) + currstr;


  return prefix + newval + suffix;
}
```

Change History

Keith Musslewhite	5/23/2011	Rev 1.0. Initial draft
Keith Musslewhite / Stan Gordon	8/23/2011	Rev 1.1 – Additional high level info for readability and timing
Keith Musslewhite	2/7/2012	Rev 1.2 – updated web method to return structure instead of dataset. Supports version 1.1 of webservice. Added configuration section.
Keith Musslewhite	9/5/2012	Rev 1.3 Added new web methods to support SFNG print server requirements.
Keith Musslewhite	5/7/2013	Rev 1.4 Added proposed web method to support ODM access to available asset number range(s)
Edgar Sierra	12/9/2013	Rev 1.7 Update table Change History, deleted "proposed" to web method UPSGetRange
Chris López	11/5/2014	Rev 2.0 Added UPSGetATM and UPSGetUSI specifications