Software Test Plan

Document

Of the

Unloaded Shipping Container

Display Pages

Extension of Lasko’s WMS System

Prepared for Lasko Products, Inc.

Version 1.0.0

Prepared by Henry Landry

April 26, 2015

# Table of Contents

[Table of Contents 1](#_Toc417658218)

[1.0 Introduction 3](#_Toc417658219)

[1.1 Goals and Objects 3](#_Toc417658220)

[1.2 Intended Audience 3](#_Toc417658221)

[1.3 Statement of Scope 3](#_Toc417658222)

[1.4 References 4](#_Toc417658223)

[1.5 Major Constraints & Points to Remember 4](#_Toc417658224)

[2.0 Testing Plan 5](#_Toc417658225)

[2.1 Architecture Diagram 5](#_Toc417658226)

[2.2 Testing Strategy 5](#_Toc417658227)

[2.2.1 Content Testing 5](#_Toc417658228)

[2.2.2 Database testing 5](#_Toc417658229)

[2.2.3 User Interface testing 6](#_Toc417658230)

[2.2.4 Component Level testing 6](#_Toc417658231)

[2.2.5 Navigation testing 6](#_Toc417658232)

[2.2.6 Configuration testing 6](#_Toc417658233)

[2.2.7 Performance testing 7](#_Toc417658234)

[2.2.8 Security testing 7](#_Toc417658235)

[2.2.9 Standards testing 7](#_Toc417658236)

[2.2.10 Implementation Testing 7](#_Toc417658237)

[2.3 Software components to be tested 7](#_Toc417658238)

[2.3.1 Web Page Interfaces 7](#_Toc417658239)

[2.3.2 Web service methods 8](#_Toc417658240)

[2.4 Testing Resources and Staffing 8](#_Toc417658241)

[2.4 Test Record Keeping 9](#_Toc417658242)

[3.0 Test Procedure 10](#_Toc417658243)

[3.1 Content & Navigation Testing 10](#_Toc417658244)

[3.2 Database Testing 10](#_Toc417658245)

[3.2.1 GetCntrDataByReferenceAndDate web service method 10](#_Toc417658246)

[3.2.2 GetCntrDataByReferenceNbr web service method 12](#_Toc417658247)

[3.3 User Interface & Component Level Testing 13](#_Toc417658248)

[3.3.1 SMSActivityByRefNbr Web Page 13](#_Toc417658249)

[3.3.2 SMSCntrDetailsByRefNbr Web Page 14](#_Toc417658250)

[3.4 Configuration Testing 15](#_Toc417658251)

[3.5 Implementation Testing 15](#_Toc417658252)

[Appendix A 16](#_Toc417658253)

[SQL Query for testing GetCntrDataByReferenceAndDate method and SMSCntrDetailsByRefNbr web page 16](#_Toc417658254)

[Appendix B 17](#_Toc417658255)

[SQL Query for testing GetCntrDataByReferenceNbr method and SMSCntrDetailsByRefNbr web page 17](#_Toc417658256)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Henry Landry | 3/29/15 | Initial version | 1.0.0 |
|  |  |  |  |

# 1.0 Introduction

## 1.1 Goals and Objects

The purpose of this document is to describe the architecture and component design of the Unloaded Shipping Container Display application extension of Lasko’s WMS system. Thorough testing of the functionality and performance of the new subsystem in a methodical and detailed manner will yield a high quality product

This test specification is a plan to accomplish the required testing that ensures the stakeholders will be happy with their new web pages in a timely manner.

## 1.2 Intended Audience

This document is intended for programmers & testers of the WMS system that will help to implement this subsystem. The document describes the tests that will be required at each level and assign a tester by their job function. Specific individuals can be assigned by management and team leaders when testing is necessary.

## 1.3 Statement of Scope

An overall plan for integration of the software and a description of specific tests are documented in this section. Below are the different kinds of tests that the team will take to ensure the quality of the software.

* Unit Testing
  + Web service methods
  + Web pages are not unit tested as

Unit tests will be performed using black box testing methods.

* Integration Testing
  + Web pages and related web service methods together
  + Web pages and particularly the flow between the pages and other pages in the system
* Validation Testing

We will test software as whole, so all the units of the software will be included

* + The web pages are shared with users of the system for their approval that the system meets their needs.
* High-order Testing
  + Web pages will be tested for general compatibility with the rest of the system with elements such as time outs.

## 1.4 References

This document references and/or should be used in conjunction with the following documents:

* Use case and activities diagrams document for this project. They will help provide clarity to this document. U28475331\_CEN4031\_RequrementsUseCase.docx
* The software requirements specifications document for the project will also help provide clarity. Landry-U28475331-CEN4031-SRS.docx .
* The software design document for the project is Landry-U28475331-CEN4031-SWDesignDoc
* Lasko Custom Programming Guide which is conventions and standards for all development.
* Various standards and conventions documents Lasko IT has assembled about the WMS application web pages and WCF service methods
* Tools – Visual Studio, Internet Information Service, ASP.NET and C#
* Software Engineering: A Practitioner’s Approach, Pressman & Maxim

## 1.5 Major Constraints & Points to Remember

In this section we will talk about the business, technical or resource related constraint that may keep us from performing all tests necessary.

1. This is an extension of an already implemented system that is internal to the business. As a result much of the base functionality is provided by the system and needs only light testing in this plan.
2. As with all web pages in the system, security is handled by IIS integrated domain security. Therefore security is not tested by this plan.
3. This project does not write any data to the database. Naturally this simplifies the testing greatly.
4. Tests for this project are not much automated. The web service methods could be automated but it would not be cost effective to do so with the limited test cases necessary. This is one benefit from not writing to the database. The testing of the web pages is not automated because it largely involves ensuring formatting and control operation meet requirements.

# 2.0 Testing Plan

## 2.1 Architecture Diagram

As a review, here is a high level architecture diagram of the subsystem.

**SMSActivityByRefNbr** Page

**SMSCntrDetailsByRefNbr** Page

**GetCntrData**

**ByReferenceAndDate**

Exposed Web Service Method

**GetCntrData**

**ByReferenceNbr**

Exposed Web Service Method

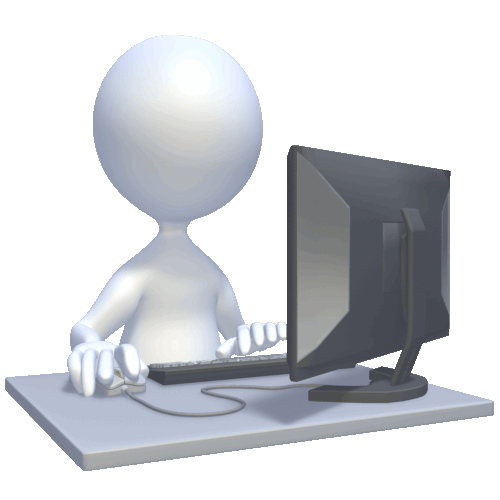
**shipmgmt**

Data Model/Database

**SummarizeCntrData**

**ByRefNbr**

Internal Web Service Method



WCF Net.TCP

HTTP

WCF Net.TCP

Internal Method

Internal Method

EDM Database Queries

**UI Layer**

**Business Layer (WMS Service)**

**Data Layer**

System Desktop user

## 2.2 Testing Strategy

This section describes the testing strategy for the Unloaded Containers project. The strategy follows a generally recommended approach for WebApps. *In order to ensure non-trivial data, these test cases assume a database with considerable is data available – production or a recent copy of production.*

There is an accompanying excel workbook for the sample test result sheets in the specific tests. Individual worksheets in the workbook are for specific tests. Samples can be used as is or enhanced if more detail is needed.

### 2.2.1 Content Testing

These tests will be designed to uncover syntactical errors such as misspellings; semantic errors by reviewing the information presented for completeness and formatting but not accuracy; and errors in the organization of the data and the webpages as they are presented. Non-technical people and managers will help with this testing.

### 2.2.2 Database testing

The database testing for this webapp will consist of unit testing the web service methods. The unit tests conducted will be both black box to exercise the input criteria and also white box by examining the code for risky parts and designing tests around these areas.

More significant database testing is not required for this project because the project never creates data. It only reads data created by other modules. This is extremely detailed testing and developers will handle it.

### 2.2.3 User Interface testing

This phase of testing will put the web pages to real use testing. It will accomplish much of the verification and validation of the site as a whole. Users will be invited to work with a test site connected to their production database so they may compare data and functionality in their day to day work. This unusual benefit arises because the project does not update data. Developers will also participate

The tests will work the controls of the pages to ensure the controls work together as requirements specify. The controls should be colored and otherwise take the look and feel of the design. By default this is considered blank box testing.

As stated in requirements documents this application is only required to be compatible with recent versions of Internet Explorer thus the usual cross browser testing is not needed for this project.

### 2.2.4 Component Level testing

Generally component level testing involves heavy form submission. In this project data is not entered. The extent of component level testing for this plan will be to ensure the criteria selected by the user to see a particular set of data is actually passed to the web service methods. Therefore, in order to save time and effort, this testing is combined with User Interface testing.

### 2.2.5 Navigation testing

Navigation is important although somewhat trivial to this project. The user must start at the SMSActivityByRefNbr web page only. Clicking on a row in the data grid should link to the SMSCntrDetailsByRefNbr page with pallet details about the selected row. If the user clicks on the Return button they should go directly to the same criteria back on the SMSActivityByRefNbr page.

Developers will formally test this functionality for the record keeping while users will certainly test during Use Interface testing. In order to save time this testing will be combined with Content Testing.

### 2.2.6 Configuration testing

This components of this project are a subsystem of an existing larger system. The larger system actually controls the configuration through the base classes and configuration files already in production. However there are two new configuration variables that must be implemented for these pages to work. The test cases will confirm handling if these variables are not set or set to impossible values. This is white box testing since it requires understanding of the details of the system.

Client side configuration is not so much of an issue due to control we have over the user’s browsers.

### 2.2.7 Performance testing

Required performance of these web pages is stated in the requirements document. Performance data will be gathered during Database and User Interface testing as both of these will read data that must meet the performance specifications.

General stress testing will not be conducted as the user population is very small and the use of this subsystem is a small load on servers compared to other existing activity on the system. However, effect on the CPU and memory use of the web service server will be monitored during Database and User Interface testing.

Combining performance testing with the other testing will save time for the project.

### 2.2.8 Security testing

Security is already handled by IIS windows domain integrated security and no further testing is required per SQS-1.

### 2.2.9 Standards testing

The software for this project will be reviewed per normal procedures for style and conformance to standards, re-use of existing method, etc. This will test requirements SQP-02, SQIP-1, SQIP-2, SQIP-03, SDR-01 and SDR-02.

### 2.2.10 Implementation Testing

As the subsystem is implement at each site testing must be performed. The nature of implementation of this WMS system means there is a set of testing done for each rollout. Naturally, there is often minimal risk to selected other parts of the system for rollouts of web services. The standard set of testing for the SMSWMSService web service should be performed after each implementation.

In addition, specific tests will be performed to ensure the new subsystem was correctly and completely implemented.

## 2.3 Software components to be tested

### 2.3.1 Web Page Interfaces

The tests to be carried on these interface web pages are described below. Most cross browser testing is not required as Internet Explorer 9 and above is the only support browser plus versions greater than IE 9 should use compatibility mode.

**SMSActivityByRefNbr Web Page**

This page is the primary entrance into the system. It lists all of the shipping containers unloaded into a warehouse for a certain time period specified in the criteria. Testing on this page will consist of ensuring the requirements for the page are met; including operation/formatting of controls, data accuracy/formatting, error handling and performance.

SMSCntrDetailsByRefNbr Web Page

Users drill down from the SMSActivityByRefNbr page into the palletization details of a selected shipping container using this page. Testing on this page will also consist of ensuring the requirements for the page are met; including operation/formatting of controls, data accuracy/formatting, error handling and performance.

### 2.3.2 Web service methods

The tests to be conducted on the web service methods are described below. Both methods return either a list or a single CntrSummaryDataByReferenceType object and tests on normal, non-exception results will be examining the contents of these objects.

**GetCntrDataByReferenceAndDate Web Service Method**

This method primarily provides the data for the SMSActivityByRefNbr web page. This makes the web page a very good test harness for the web service method during integration testing. However the service method also can provide more data and criteria than the web page supports so much unit testing will be required and handled through non-automated WCF Tester. This method must meet the requirements for data accuracy, error handling and performance.

**GetCntrDataByReferenceNbr Web Service Method**

This method primarily provides the data for the SMSCntrDetailsByRefNbr web page. This makes the web page a good test harness for the web service method. However the service method also can provide more data and criteria than the web page supports so much unit testing will be required and handled through non-automated WCF Tester. This method must meet the requirements for data accuracy, error handling and performance.

**SummarizeCntrDataByReferenceNbr Web Service Method**

This internal helper method primarily does most of the heavy work for the other two methods. This method will have limited individual unit testing of its own because testing of the exposed methods will cover testing of this method.

## 2.4 Testing Resources and Staffing

Lasko has no dedicated testing personnel. Therefore the developers working on the system will be the primary testers while IT managers will act as QA support checking unit tests and exercising some cases of their own. Selected users of the system will also help with testing in an informal way, as a parallel to their existing work methods.

## 2.4 Test Record Keeping

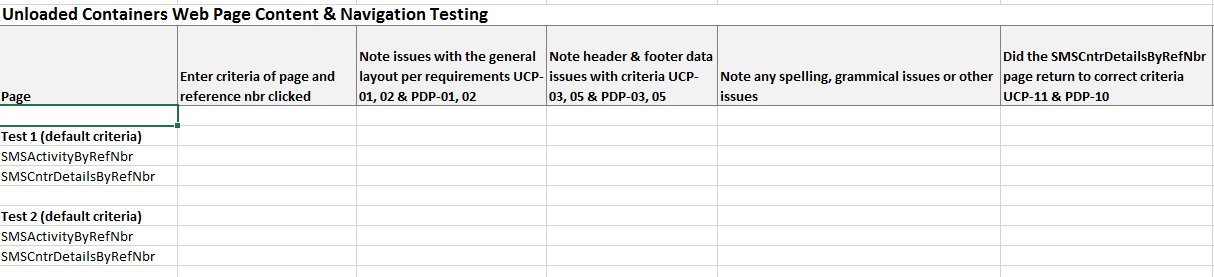
Each set of test cases in section 3 displays an example of a spreadsheet that can be used to log the results of test. Logs should be submitted to the project lead for review and filing.

# 3.0 Test Procedure

This section describes the test procedures in detail.

## 3.1 Content & Navigation Testing

Open the SMSActivityByRefNbr.aspx page on your browser from the main SMS directory http://SMS. Confirm the default criteria is set and some data loads into the grid. Examine each section of the page and the spelling of all controls on the page (do not check spelling of data). Complete spreadsheet rows for this page as displayed below. After that is complete click on any row in the data grid to move to the SMSCntrDetailsByRefNbr Web Page. Confirm the title and palletization data on the page matches row you clicked on. Return to the SMSActivityByRefNbr page using the Return button. Repeat the test several times with different criteria and different reference numbers – confirm the return always sets SMSActivityByRefNbr to the criteria selected when the row was clicked to get to SMSCntrDetailsByRefNbr.



Before completing the tests leave a page for at least 11 minutes to confirm there is no time out on the page. During all the testing above confirm the footer section has correct information – server name & IP, date & time and screen code.

Finally open the SMSDeviceState.aspx page and confirm the entries there for your session match the page you are currently on.

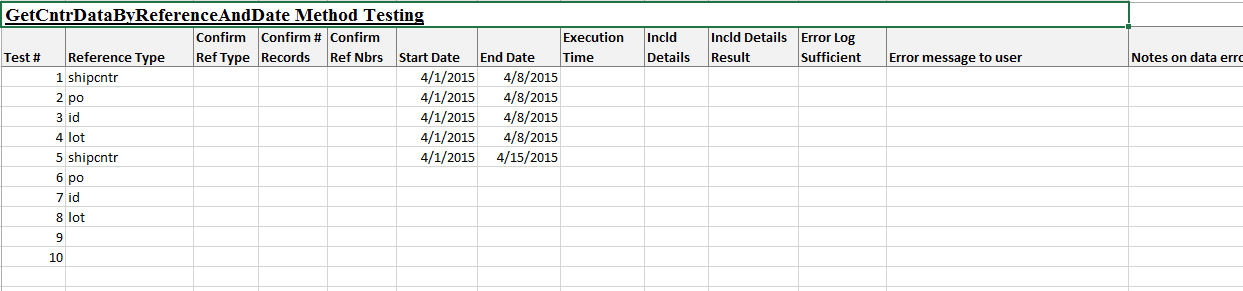
This section tests requirements UCP-01, UCP-02, UCP-03, UCP-05, UCP-07, UCP-11, UCP-14 and PDP-01, PDP-02, PDP-03, PDP-05, PDP-10, SQP-03, SQT-4

## 3.2 Database Testing

### 3.2.1 GetCntrDataByReferenceAndDate web service method

Execute the method using WCF tester and test cases listed below. For test cases that should not return results (errors) ensure the log entries and returned exceptions meet specifications. All tests of the method can test performance using non automated methods to time the returned results.

Sample spreadsheet to log test results:



**Test cases are a combination of white and black box testing**

1. This step finds nontrivial sets of data to test with; that is sizeable sets of criteria which are sets of reference numbers contained in List of CntrSummaryDataByReferenceType objects. Using SQL queries on the S1301\_Cntr table, find non-trivial sets of data for each of the reference types (shipcntr, id, po and lot) with date ranges of 1 year ago through today, 1 month ago through today, 2 weeks ago through today and days ago through today. This is a total of 16 cases. However, the test can be abbreviated by limiting the test to shipcntr and id, which are the most popular reference types.
2. This step tests the larger sets of data with generally successful results. Try each set of data located in step 1 using WCF tester on the GetCntrDataByReferenceAndDate method and time the results, logging the time as described in record keeping above. The IncldDetails argument can be alternated between true and false but results to confirm it worked noted on the log. It is not necessary in this section to test the values in the CntrDetails of the returned results, only that there are either null or a list of results depending on the IncldDetails argument.
3. Look at the number of records returned and the reference type & reference numbers in WCF tester to confirm this data matches the same fields on the SQL query.
4. It is not necessary to confirm the detailed fields on the screen. This will be done in the section testing GetCntrDataByReferenceNbr method. That section will use some of the data from step 2 to validate the data returned at detail level for selected reference numbers. From the results returned in step 2 from the calls select 2-3 individual reference numbers to test in detail and record them on the log for GetCntrDataByReferenceNbr to be used later.
5. This step tests exception handling. Execute the query for impossible criteria such as
   1. Earlier end date than start date
   2. A reference type other than the proper values
   3. Null values for each of the criteria
   4. Change the database connection string to an invalid one
   5. Change the service reference for LaskoERP to an invalid one. This will cause expected quantities to fail but should not fail the entire call. Log entries should be made
   6. Use impossible date ranges such as pre-2010 and post-2015
   7. Log the results and error messages for the users. Confirm the log entries explain the situation and point the correct fault location to a bug correction developer sufficiently.

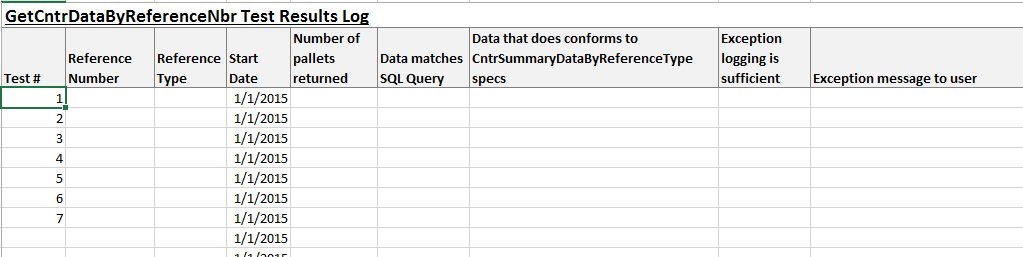
This section tests UCP-12, SQE-01, SQRE-1, SDR-03 and SQT-01

### 3.2.2 GetCntrDataByReferenceNbr web service method

Execute the method using WCF tester and test cases listed below. For test cases that should not return results (errors) ensure the log entries and returned exceptions meet specifications. All tests of the method can test performance using non automated methods to time the returned results.

Tests on this method will also test the SummarizeCntrDataByReferenceNbr method.

Sample spreadsheet to log results



**Test cases are a combination of white and black box testing**

1. This step uses the reference numbers collected in step 3 of the GetCntrDataByReferenceAndDate method tests to validate the data returned at detail level for selected reference numbers.
   1. Execute the SQL query in Appendix A for each of the individual reference numbers logged in the previous section in turn. Then execute the service method in WCF tester for the corresponding reference number. Ensure the IncldDetails parameter is turned on for the web service method.
   2. The data in the top level CntrSummaryDataByReferenceType class should match the data in the SQL query.
      1. Using the design specifications for the CntrSummaryDataByReferenceType class validate the results of each selected reference number to ensure logical accuracy. As examples, check the start time is really the earliest pallet scanned and multiple item containers have ‘multiple’ for item description.
      2. Reference type ‘shipcntr’ is the only type that returns the field ‘Expected Qty’. This value must be looked up in the ERP system by container number to verify its accuracy.
   3. Execute the SQL query in Appendix B for each of the individual reference numbers. Expand the CntrDetailData class in WCF tester.
      1. The results returned in the web service method in CntrDetailData for each selected reference number should match the SQL query row by row.
   4. Log results as described in the record keeping section
2. This step tests exception handling. Execute the query for impossible criteria such as
   1. Null, empty string and impossible reference numbers
   2. Change the database connection string to an invalid one
   3. Use a very old date and a future date such as pre-2010 and post-2015
   4. Log the results and error messages for the users. Confirm the log entries explain the situation and point the correct fault location to a bug correction developer sufficiently.

This section tests SQE-02, SQRE-1, SDR-03 and SQT-01

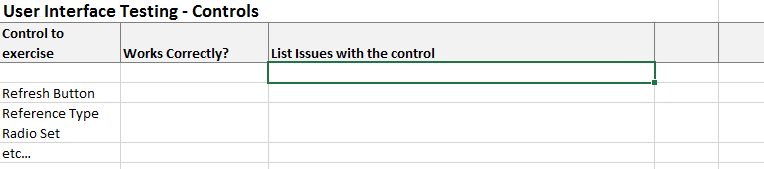
## 3.3 User Interface & Component Level Testing

### 3.3.1 SMSActivityByRefNbr Web Page

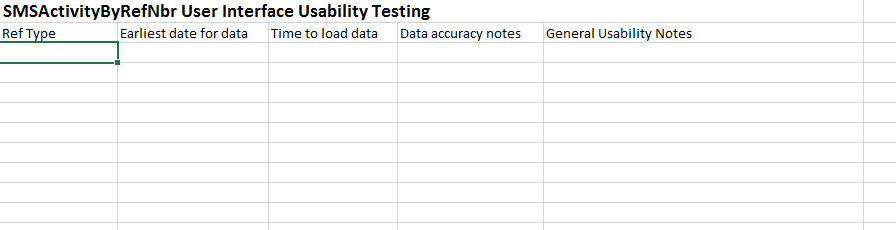
Open the web page up and work each of the controls on the page thoroughly. Make sure to work the controls in combination with each other as they have tight interactions. Requirements specifications UCP-05 and UCP-06 discuss how they should interact.

The primary controls in question are the Refresh button, the Reference Type radio set, and the date criteria. These all affect the screen title so monitor the title changes as each control is used.

As each control is tested make notes about its adherence to specifications on a spreadsheet. A sample is provided below



After controls are confirmed, use the screen with criteria to match criteria from the database tests in 3.2.1 above. This includes the default criteria when the page first opens. The database test confirmed the data returned from web service methods is accurate. The same results can be used to confirm the results displayed on the page are accurate – the screen should display the exact results web methods returned with proper formatting. This also tests the criteria submitted to the web service methods and the performance of the web page. Repeat for various criteria. Log the results on a spreadsheet similar to below.

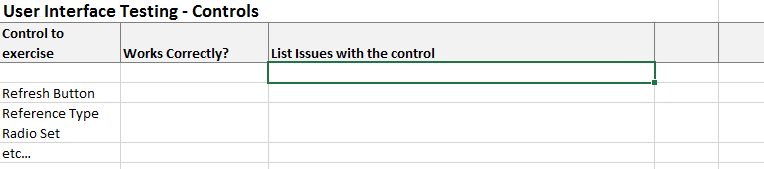


Requirements tested by this section are UCP-03, UCP-05, UCP-06, UCP-08, UCP-09, SQE-01, and SQIP-2.

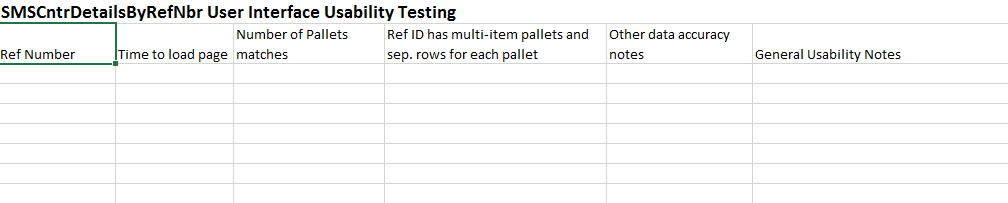
### 3.3.2 SMSCntrDetailsByRefNbr Web Page

Open the SMSActivityByRefNbrPage up and select one of the rows in the grid to get to page being tested. Confirm the title contains the reference number you selected.

If connected to a production database it will be possible to find a reference type ID with today’s date and shift (for example 20150420S01 for first shift). With this information you can test the refresh button is updating with new records since production is almost always creating pallets.



After controls are confirmed, use the screen with criteria to match criteria from the database tests in 3.2.2 above. The database test confirmed the data returned from web service methods is accurate. The same results can be used to confirm the results displayed on the page are accurate – the screen should display the exact results web methods returned with proper formatting. This also tests the criteria submitted to the web service methods and the performance of the web page. Repeat for various criteria. Log the results on a spreadsheet similar to below.



Finally, attempt to open the page with no query string in the URL. The page should fail with an error message.

Requirements tested by this section are PDP-03, PDP-04, PDP-06, PDP-10, SQT-1, SQP-3, SQE-02, and SQIP-2.

## 3.4 Configuration Testing

This test is for confirming the operation of the configuration variables. Alter the values of each of the configuration variables CntrActivityMainPg and CntrActivityDetailPg in web.config for your test site.

If the variable is removed completely, the hardcoded default value should actually work correctly however a log entry should be made. Remove each variable one at a time, then click from the page you left in place to the page you removed. The same results should happen if removing both variables.

If the variable is set to an impossible folder or URL the page should display a graceful error message for the user. Change each variable one at a time, then click from the page you left unaltered to the page you altered. The same results should happen if altering both variables.

Record results on a spreadsheet.

## 3.5 Implementation Testing

After implementation at each site, run the required standard tests for the WMS system to confirm no damage was done to the site’s software.

Run these tests to confirm the new Unloaded Containers page subsystem was correctly implemented.

1. Open the SMSActivityByRefNbr.aspx page for the site that was implemented from the SMS main directory <http://sms>.
2. Check the site’s Device State page entry for your device to ensure accuracy
3. Ensure the page opens with reasonable data and is formatted correctly (confirms all code behind and style sheet files were implemented)
4. Click on a row in the grid to ensure the palletization details for the correct reference number opens up and is formatted correctly.
5. Check the event viewer for SMSWMSService and the device log for your IP address for any errors. There should be none, especially entries about missing configuration variables.
6. Ensure the data is from the correct site.
7. Repeat several times for different criteria.

# Appendix A

### SQL Query for testing GetCntrDataByReferenceAndDate method and SMSCntrDetailsByRefNbr web page

/\*\*\*\*\*\* Script for calculating test results for GetCntrDataByReferenceAndDate method \*\*\*\*\*\*/

/\*\*\*\*\*\* initialize the @refnbr parameter with the reference number to test \*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\* the @dmgloc parameter can be changed to a different warehouse location if necessary \*/

use shipmgmt

DECLARE @refnbr varchar(15) = 'ecmu805984'

DECLARE @dmgloc varchar(15) = '99'

SELECT

S1301\_RefNbr

,[S1301\_RefType]

,[S1301\_ConfirmedOperationCode]

,[S1301\_Customer]

,[S1301\_RefVerified]

,(select min( S1301\_CreateDate ) from S1301\_Cntr where S1301\_RefNbr = @refnbr ) as 'Start Date'

,(select min( S1301\_CreateTime ) from S1301\_Cntr where S1301\_RefNbr = @refnbr ) as 'Start Time'

,(select max( S1301\_CreateDate ) from S1301\_Cntr where S1301\_RefNbr = @refnbr ) as 'End Date'

,(select max( S1301\_CreateTime ) from S1301\_Cntr where S1301\_RefNbr = @refnbr ) as 'End Time'

,(case

when ( select count( distinct( S1302\_ItemNbr ) ) from S1301\_Cntr

join [shipmgmt].[dbo].[S1302\_CntrDetail] as S1302

on S1301\_Guid = S1302\_CntrGuid

where S1301\_RefNbr = @refnbr ) > 1

Then 'Multiple'

else

S1302\_ItemNbr

end

) as 'Item Nbr'

,(case

when ( select count( distinct( S1302\_ItemNbr ) ) from S1301\_Cntr

join [shipmgmt].[dbo].[S1302\_CntrDetail] as S1302 on S1301\_Guid = S1302\_CntrGuid

join [shipmgmt].[dbo].[S0501\_Item] on S1302\_ItemGUID = S0501\_GUID

where S1301\_RefNbr = @refnbr ) > 1

Then 'Multiple'

else

( select top 1 S0501\_ItemDesc1 from S1301\_Cntr

join [shipmgmt].[dbo].[S1302\_CntrDetail] as S1302 on S1301\_Guid = S1302\_CntrGuid

join [shipmgmt].[dbo].[S0501\_Item] on S1302\_ItemGUID = S0501\_GUID

where S1301\_RefNbr = @refnbr )

end

) as 'Item Desc'

,[S1301\_CntrStatus]

,[S1301\_BinNbr]

,(select top 1 S1301\_UserID from S1301\_Cntr where S1301\_RefNbr = @refnbr ) as 'User ID'

,[S1301\_BuildOperationCode]

,( select count('X') from S1301\_Cntr where S1301\_RefNbr = @refnbr and S1301\_CntrStatus <> 'printed' and S1301\_CntrStatus <> 'created' ) as 'Verified'

,( select count('X') from S1301\_Cntr where S1301\_RefNbr = @refnbr and S1301\_WHLocation <> @dmgloc ) as 'Udmge Pallets'

,( select sum(S1302\_OriginalQty) from S1301\_Cntr

join [shipmgmt].[dbo].[S1302\_CntrDetail] as S1302 on S1301\_Guid = S1302\_CntrGuid

where S1301\_RefNbr = @refnbr and S1301\_WHLocation <> @dmgloc ) as 'Udmge Qty'

,( select sum(S1302\_OriginalQty) from S1301\_Cntr

join [shipmgmt].[dbo].[S1302\_CntrDetail] as S1302 on S1301\_Guid = S1302\_CntrGuid

where S1301\_RefNbr = @refnbr and S1301\_WHLocation = @dmgloc ) as 'Damage Qty'

,[S1301\_WHLocation]

FROM [shipmgmt].[dbo].[S1301\_Cntr] as S1301

join [shipmgmt].[dbo].[S1302\_CntrDetail] as S1302 on S1301\_Guid = S1302\_CntrGuid

where S1301\_RefNbr = @refnbr

# Appendix B

### SQL Query for testing GetCntrDataByReferenceNbr method and SMSCntrDetailsByRefNbr web page

/\*\*\*\*\*\* Script for SelectTopNRows command from SSMS  \*\*\*\*\*\*/

use shipmgmt

declare @Refnbr varchar(20) = 'oolu441788'

SELECT [S1301\_CntrID]

      , S1301\_UserID

      ,( select S0901\_User.S0901\_UserName from S0901\_User where S1301\_UserID = S0901\_UserID ) as 'Name'

      ,[S1301\_WHLocation]

      ,(case

            when [S1301\_BinNbr] is null then 'None'

            else

            [S1301\_BinNbr]

            end

            )  as 'Bin Nbr'

      ,[S1301\_CntrStatus]

      ,[S1301\_CreateDate]

      ,[S1301\_CreateTime]

      ,[S1302\_ItemNbr]

      ,(Select S0501\_ItemDesc1 from S0501\_Item where S0501\_itemnbr = S1302\_itemnbr ) as 'Item Desc'

      ,[S1302\_CntrDetail].S1302\_OriginalQty

  FROM [shipmgmt].[dbo].[S1301\_Cntr]

  join [shipmgmt].[dbo].S1302\_CntrDetail on S1301\_GUID = S1302\_CntrGUID

  where S1301\_RefNbr = @Refnbr