# *ODPP Column Map Exits*
# *Migration Guide from v9.1.0.4 to v11.3.0*

**IBM's Optim Enterprise Solution**

**Version 11 Release 3 Modification 0 (June 2014)**
This edition applies to version 11, release 3, modification 0 of IBM InfoSphere Optim Masking on Demand and to all subsequent releases and modifications until otherwise indicated in new editions.

# 1. Introduction

This guide contains details for migrating an Optim Column Map exit that uses the ODPP Application Programming Interface (API) from v9.1.0.4 to v11.3.0

The current version includes significant changes that make it easier to invoke the ODPP API functions as well as improvements to error reporting.

# 2. Changes in v11.3.0

## *2.1 Changes to invoking the ODPP API functions*

To write an Optim column map exit based on ODPP in Optim 11.3.0:

1. Include header "ODPPAPI.h". This header is provided along with the sample Column Map exits for ODPP (CMExit_ODPP_CCN) and Optim (CMExit).

```
#include "ODPPAPI.h"
```

2. Set the API pointer. A pointer to the ODPP API structure that contains the function pointers is passed as a member of the input parameter structure PST_STRUCT_CM_EXIT_PARM / PST_STRUCT_CM_WEXIT_PARM.

```
ODPP_INIT_API_PTR(pInputParms->pODPPApi);
```

3. Invoke ODPP functions using the defines in the header. The ODPP Framework is initialized and terminated by the Optim Column Services component. Hence functions ODPP_Framework_Init and ODPP_Framework_Term must not be invoked from within the column map exit.

- ODPP_Provider_Init
- ODPP_Provider_Service
- ODPP_Provider_Term
- ODPP_Provider_Get_Error_Count
- ODPP_Provider_Get_Error
- ODPP_Provider_Get_Formatted_Error_Msg
- ODPP_Provider_Term_Errors
- ODPP_Provider_Get_Info

## 2.2 Changes to error reporting

The following table summarizes the changes to the return code values in v11.3.0. Informational messages and row processing errors return code ODPPSUCCESS_WITH_INFO while successful rows return code ODPPSUCCESS. For row processing errors the bHasError member of DP_ROW_DEF is TRUE while for informational messages the bHasError member of DP_ROW_DEF is FALSE.

| | | Prior to v11.3.0 | v11.3.0 |
|---|---|---|---|
| **Successful row** | Return code | ODPPSUCCESS | ODPPSUCCESS |
| | bHasError member of DP_ROW_DEF | FALSE | FALSE |
| | iErrorCode member of DP_FIELD_DATA_DEF | 0 | 0 |
| **Informational messages** | Return code | ODPPSUCCESS | ODPPSUCCESS_WITH_INFO |
| | bHasError member of DP_ROW_DEF | FALSE | FALSE |
| | iErrorCode member of DP_FIELD_DATA_DEF | Non-zero error code | Non-zero error code |
| **Row errors** | Return code | ODPPSUCCESS | ODPPSUCCESS_WITH_INFO |
| | bHasError member of DP_ROW_DEF | TRUE | TRUE |
| | iErrorCode member of DP_FIELD_DATA_DEF | Non-zero error code | Non-zero error code |
| **Process error** | Return code | Code other than ODPPSUCCESS | Code other than ODPPSUCCESS and ODPPSUCCESS_WITH_INFO |

# 3. For ODPP v9.1.0.4 users

The following steps must be followed to migrate the column map exit from v9.1.0.4 to v11.3.0.

1. Update the Optim header files. The v11.3.0 header files are available with the sample column map exits.
    - PSTEXIT.H
    - PSTCMXIT.H
    - PSTEXIT.H

- ODPPAPI.h

2. Update the ODPP header files.
3. Remove the defines for the ODPP function and library names.
4. Remove the type definitions for the ODPP function pointers.
5. Remove the ODPP_PROVIDER_API_DEF structure.
6. Remove the following function definitions:
   - GetODPPProcAddress
   - ODPPLoadLibrary
   - ODPPUnloadLibrary
   - LoadProviderLib
7. Remove the invocation of the above functions from the code.
8. Include header "ODPPAPI.h".
9. Set the ODPP API pointer at the beginning of function PSTColMapExit / PSTColMapWExit:

   ODPP_INIT_API_PTR(pInputParms->pODPPApi);

10. Replace the invocation of the ODPP functions with the corresponding defines.

    For example:

    - Replace (pCcnWa->OdppPrvApi.pInit) with ODPP_Provider_Init
    - Replace (pCcnWa->OdppPrvApi.pService) with ODPP_Provider_Service
    - Replace (pCcnWa->OdppPrvApi.pTerm) with ODPP_Provider_Term

11. If present, remove invocations of ODPP_Framework_Init and ODPP_Framework_Term.
12. After the invocation of ODPP_Provider_Service, modify the error reporting to handle the code ODPPSUCCESS_WITH_INFO for row processing errors and informational messages.

    For example:

```
retVal = ODPP_Provider_Service(pCcnWa->iSvcToken, sMethod, NULL, pCcnWa->pRowSet);
switch(retVal)
{
case ODPPSUCCESS:
  break;
case ODPPSUCCESS_WITH_INFO:
  // Are there ODPP row errors?
  if( pCcnWa->pRowDef->bHasError == TRUE)
  {
    // Yes, Process the ODPP errors and skip this row
    bIsInfoMsg = FALSE;
    ProcessODPPErrors(pCcnWa, bIsInfoMsg);
    rc = PST_CMW_EXIT_REJECT_ROW;
    goto SkipRow;
  }
  else
  {
    // Informational message such as source NULL.
    // Process the the informational messages to free the captured ODPP error control block
    // (ECB). This will ensure that memory consumption remains stable when there are
    // several NULL values.
    bIsInfoMsg = TRUE;
    ProcessODPPErrors(pCcnWa, bIsInfoMsg);
  }
  break;
default:
```

```
    // Process error
    bIsInfoMsg = FALSE;
    ProcessODPPErrors(pCcnWa, bIsInfoMsg);
    rc = PST_CMW_EXIT_ABORT_PROCESS;
    goto Exit;
  }
```

13. Compile the column map exit.

# 4. For ODPP v9.1 users whose exit links to ODPPImpLib.9.1.lib

The following steps must be followed to migrate the column map exit from a v9.1 version where the column map exit links to ODPPImpLib.9.1.lib.

1. Update the Optim header files. The v11.3.0 header files are available with the sample column map exits.

- PSTEXIT.H
- PSTCMXIT.H
- PSTEXIT.H
- ODPPAPI.h

2. Update the ODPP header files.
3. Remove ODPPImpLib.9.1.lib from the project settings under Linker -> Input -> Additional Dependencies.
4. Remove the code that invokes functions Provider_FrmwInit() and Provider_FrmwTerm().
5. Include header "ODPPAPI.h".
6. Set the ODPP API pointer at the beginning of function PSTColMapExit / PSTColMapWExit:

```
ODPP_INIT_API_PTR(pInputParms->pODPPApi);
```

7. Replace the invocation of the ODPP functions with the corresponding defines.

For example:

- Replace Provider_Init with ODPP_Provider_Init
- Replace Provider_Service with ODPP_Provider_Service
- Replace Provider_Term with ODPP_Provider_Term
- Replace Provider_GetErrorCount with ODPP_Provider_Get_Error_Count
- Replace Provider_GetError with ODPP_Provider_Get_Error
- Replace Provider_GetFormattedErrorMsg with ODPP_Provider_Get_Formatted_Error_Msg

8. After the invocation of ODPP_Provider_Service, modify the error reporting to handle the code ODPPSUCCESS_WITH_INFO for row processing errors and informational messages.

For example:

```
retVal = ODPP_Provider_Service(pCcnWa->iSvcToken, sMethod, NULL, pCcnWa->pRowSet);
switch(retVal)
{
case ODPPSUCCESS:
  break;
```

```
case ODPPSUCCESS_WITH_INFO:
  // Are there ODPP row errors?
  if( pCcnWa->pRowDef->bHasError == TRUE)
  {
    // Yes, Process the ODPP errors and skip this row
    bIsInfoMsg = FALSE;
    ProcessODPPErrors(pCcnWa, bIsInfoMsg);
    rc = PST_CMW_EXIT_REJECT_ROW;
    goto SkipRow;
  }
  else
  {
    // Informational message such as source NULL.
    // Process the the informational messages to free the captured ODPP error control block
    // (ECB). This will ensure that memory consumption remains stable when there are
    // several NULL values.
    bIsInfoMsg = TRUE;
    ProcessODPPErrors(pCcnWa, bIsInfoMsg);
  }
  break;
default:
  // Process error
  bIsInfoMsg = FALSE;
  ProcessODPPErrors(pCcnWa, bIsInfoMsg);
  rc = PST_CMW_EXIT_ABORT_PROCESS;
  goto Exit;
}
```

9. Compile the column map exit.

# 5. Compiling exits with Microsoft Visual Studio

The Optim and ODPP binaries are compiled with Visual Studio 2008 SP1 and the dependent MSVC libraries such as msvcr90.dll are provided along with the Optim binaries. If the exit is compiled with a higher version of Visual Studio such as Visual Studio 2012 the corresponding dependent MSVC libraries such as msvcr110.dll must be copied to the rt\bin folder of the Optim installation. Failing to do so may result in a “User Exit Load Error”.