OBS Plug-in for
Avid MediaCentral|UX

Table of Contents

1 Overview

The OBS Plug-in for Avid MediaCentral|UX delivers the functionality previously provided by the OBS Web application. Unlike OBS Web the plug-in delivers functionality that is wholly integrated within the Avid MediaCentral|UX framework.

2 Requirements

- Avid MediaCentral|UX version 2.5 or later
- Avid Interplay Web Services
- Avid iNEWS Web Services version 1.4 or later
- FTP site(s) to receive published rundowns

3 Deployment

The OBS plug-in is delivered as a java archive ".jar" file together with a dependant library. Currently in the absence of an Avid mechanism for installing third party plug-ins these files will need to be deployed manually to the MediaCentral|UX server(s).

4 Installing the Plug-in

As described in the previous section it is necessary to deploy the plug-in manually. This requires the creation of folders on the MediaCentral|UX server(s) and uploading the plug-in resources to them. This can be achieved most conveniently through the use of remote access tools such as PUTTY and PSFTP.

4.1 Create the Destination Folder

The plug-in must be installed in a dedicated directory in the path:

```
/opt/avid/avid-interplay-central/plugins
```

Create a new directory in plugins to hold the OBS plug-in resources.These notes will assume that the directory is named `obs`.

4.2 Upload the OBS Plug-in Resources

Copy the two resource files, commons-net-3.3.jar and obsPlugin-V.V.V.vYYYYMMDD_HHMMSS.jar to the newly created directory. (Note that V.V.V represents the version number of the plug-in and YYYYMMDD etc. the timestamp when it was created).

4.3 Create the OBS Plug-in Profile

In order to be recognised by Avid MediaCentral|UX a profile is required. This is a YAML file having a .yml extension. This file can either be created locally and uploaded to the server(s) or created on the servers using VI or similar.

4.3.1 Profile Location

MediaCentral|UX plug-in profiles are stored in the following location:

```
/opt/avid/avid-interplay-central/profiles
```

4.3.2 OBS Plug-in YAML File

The contents of the file should include the following data:

```
name: "OBS"
modules:
  - obs
optional: true
```

The key parameters here are the name of the plug-in, in this case OBS", and in the modules section the location of the plug-in resources in the plugins directory. This will be the name of the directory created in section 4.1 previously, in this example obs.

Either create a file (e.g. obs.yml) on the server(s) with contents as described above or create a similar file locally and upload it. Note that the install package contains a suitable file for a directory named obs.

4.4 Enabling the Plug-in

The final stage of installing the plug-in is to enable it in MediaCentral|UX. To enable the plug-in execute the following command:

```
/opt/avid/avid-interplay-central/configurator
```

This will launch the MediaCentral|UX configurator:
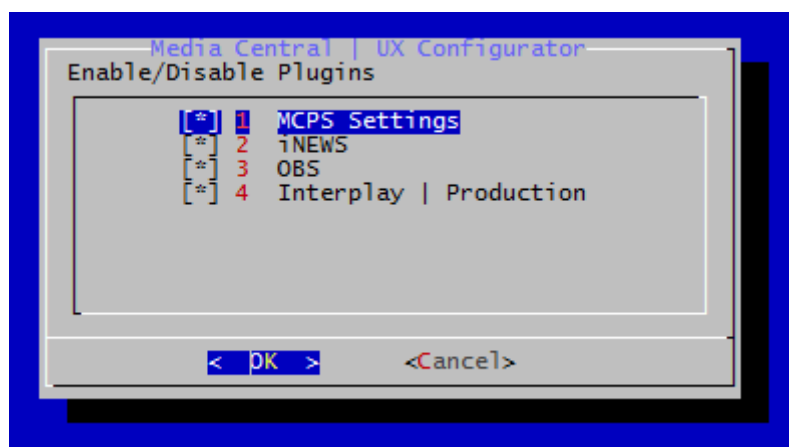
Use the up/down arrow keys to access the OBS plug-in checkbox ([ ]) and press the spacebar to add the enabling asterix. Tab to the OK button and press enter to accept the change. Avid MediaCentral|UX will be restarted and the OBS plug-in activated.

4.5 Clustered Servers

It will be necessary to perform the install tasks on all servers in a MediaCentral|UX cluster.

5 Updating the Plug-in

Updating the plug-in simply involves replacing any changed resource files and restarting the MediaCentral|UX framework.
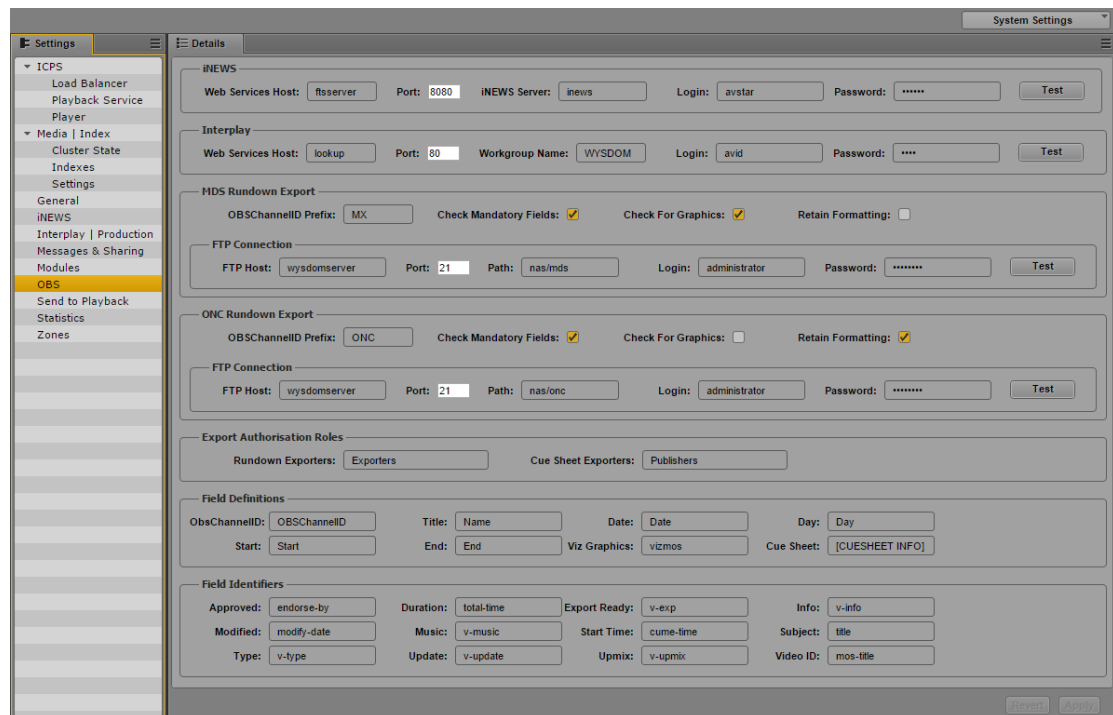
5.1 Restarting the Avid MediaCentral|UX Framework

To restart the framework execute the following command:

```
service avid-interplay-central restart
```

6 Configuring the Plug-in

The plug-in configuration is accessed from the System Settings layout:

The settings pane includes several sections, each managing the configuration of a particular aspect of the OBS plug-in. The sections that follow describe each configuration in more detail. Note that for more information concerning field definitions and identifiers see the document *OBS Lillehammer and Rio XML export Tool V11*.

6.1 iNEWS Configuration

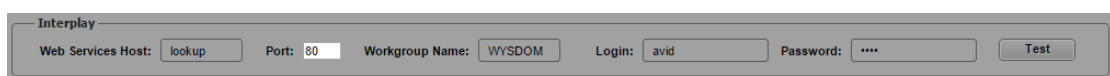This section configures communication with iNEWS web services and hence the iNEWS server.

iNEWS

Web Services Host: ftsserver   Port: 8080   iNEWS Server: inews   Login: avstar   Password: •••••••   Test

Web Services Host:	Hostname or IP Address of the server hosting iNEWS web services
Port:	The http port used by the iNEWS web services server
iNEWS Server:	Hostname or IP Address of the iNEWS server
Login:	The user name used to communicate with iNEWS
Password:	Password used to communicate with iNEWS

6.2 Interplay Configuration

This section configures communication with Interplay web services and hence the Interplay system.
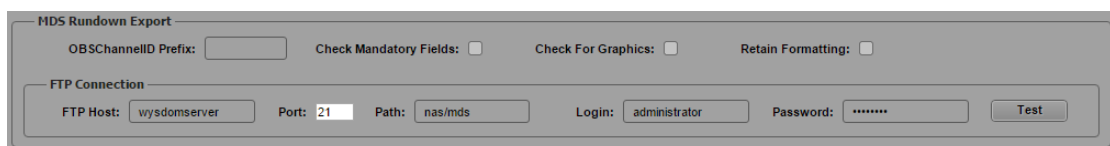
Interplay

Web Services Host: lookup   Port: 80   Workgroup Name: WYSDOM   Login: avid   Password: ••••   Test

Web Services Host:	Hostname or IP Address of the server hosting Interplay web services
Port:	The http port used by the Interplay web services server
Workgroup Name:	The workgroup name used to identify the Interplay System by Interplay web services. Note that this is not necessarily the same as the Avid Service Framework workgroup name.
Login:	The user name used to communicate with Interplay
Password:	Password used to communicate with Interplay

6.3 MDS and ONC Rundown Export

The OBS plug-in exports rundowns as XML formatted documents. These documents are exported to an FTP destination. The MDS and ONC Rundown Export sections configure aspects of the rundowns themselves together with details of the FTP server to which the rundowns will be exported.

MDS Rundown Export

OBSChannelID Prefix: Check Mandatory Fields: ☐   Check For Graphics: ☐   Retain Formatting: ☐

FTP Connection

FTP Host: wysdomserver   Port: 21   Path: nas/mds   Login: administrator   Password: ••••••••   Test

6.3.1 Rundown Identifiers

The OBSChannelID Prefix is used to identify the rundown as being an MDS or ONC rundown. The rundown is first checked to see if it is an MDS rundown, if so the initial characters of the OBSChannelID must match those entered in the MDS OBSChannelID Prefix field.

If the rundown is not identified as an MDS one it is then checked to see if the initial characters of the OBSChannelID match those entered in the ONC OBSChannelID Prefix field, if so the rundown is identified as an ONC rundown.

Note that if the OBSChannelID Prefix field is left empty then *all* rundowns not identified as MDS will be regarded as ONC rundowns.

6.3.2 Export Options

Three options configure the export of rundowns:

Check Mandatory Fields: If *checked* those fields identified as mandatory will be checked and if found to be empty will cause the export to fail. If *unchecked* empty fields will not prevent an export of the rundown.

Check For Graphics: If *checked* and an item does not include a graphics element then the export will fail. If *unchecked* missing graphics will not prevent an export of the rundown.

Retain Formatting: If *checked* the story body and cue sheet (if any) will retain all iNEWS formatting tags (e.g. Closed Caption, text formatting). If *unchecked* the exported script will be "cleaned" of all formatting tags.
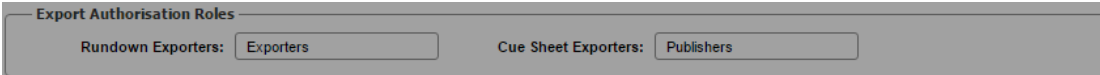
6.3.3 FTP Connection

This section configures the export FTP connection.

FTP Host: Hostname or IP Address of the ONC export FTP server
Port: The FTP port used by the ONC export FTP server
Path: The destination directory path
Login: The FTP user name
Password: The FTP password

6.4   Export Authorisation

Only certain users are permitted to export rundowns and/or cue sheets. Users are authorised by assigning them an appropriate MediaCentral|UX role. This section defines the roles used.
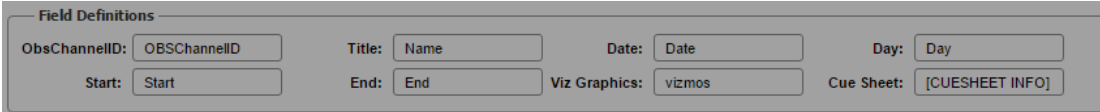

Rundown Exporters: Role to permit the export of rundowns
Cue Sheet Exporters: Role to permit the export of cue sheets

6.5   Field Definitions

This section defines the data that is used to identify key parameters associated with exports.

ObsChannelID: Identifies the story containing the channel identifier
Title: Identifies the story containing the rundown title

Date:	Identifies the story containing the rundown date
Day:	Identifies the story containing the rundown day
Start:	Identifies the story containing the rundown start time
End:	Identifies the story containing the rundown end time
Viz Graphics:	Identifies an embedded production cue as referencing a VizRT graphics item
Cue Sheet:	Identifies the start of cue sheet (aka marker data) published to the story body

Note that with the exception of Viz Graphics and Cue Sheet the definitions are not case sensitive.

6.6   Field Identifiers

This section defines which iNEWS form fields correspond to a given export parameter.

Approved:	Identifies the approval field
Duration:	Identifies the duration field
Export Ready:	Identifies the field indicating the export ready status
Info:	Identifies the information field
Modified:	Identifies the modification timestamp field
Music:	Identifies the music field
Start Time:	Identifies the start time field
Story ID:	Identifies the story ID field
Subject:	Identifies the subject field
Type:	Identifies the type field
Update:	Identifies the update field
Upmix:	Identifies the upmix time field
Video ID:	Identifies the Video ID field

6.7   Saving the Configuration

Once all data has been entered click Apply to save the changes.

6.8   Reverting Changes

If changes have been made but not yet saved they can be "undone" by clicking the Revert button.

6.9   Testing the Configuration

Once the configuration has been saved it is possible to test the important communication aspects. The iNEWS, Interplay and FTP configurations all feature a Test Connection button. For iNEWS and Interplay the test will check that communication with the server is possible. iNEWS also checks that the

required version of web services is available whilst Interplay checks that the designated workgroup name is available.

For the ONC and FTP connections the test will confirm that a test file can be written to the configured destination. Note that as part of the test the file will be deleted.

Note that if any of the parameters is changed the Apply button must be clicked in order to save the configuration. If this is not done then the previously saved values will be used.
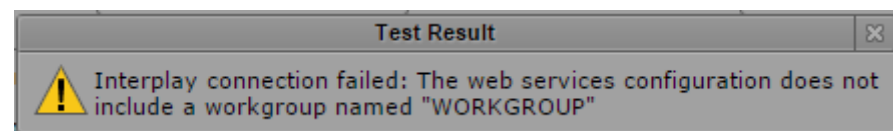
To test a connection, click the appropriate Test Connection button. Particularly if this is the first time that the connection has been used a "Please Wait" dialog may be shown, after which the result of the test will be displayed. For example the result of a successful iNEWS test would be:
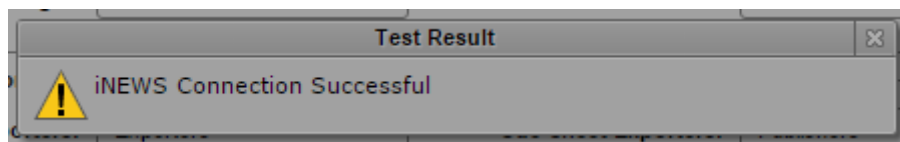
A failed Interplay connection might display results such as:
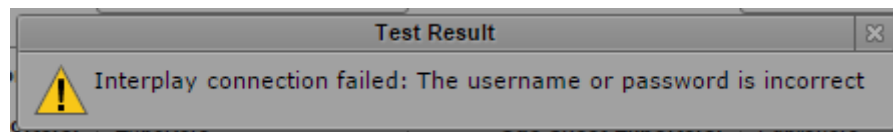
Or

6.10 Saved Settings Location

For this release the configuration settings are saved to a configuration file in the plug-in directory. Within the configured directory (e.g. `plugins/obs`) a settings directory is created by the plug-in. The settings are stored in this directory as a file named obsconfig.xml.

Currently once configured it will be necessary to copy this file and its associated sub-directory to other MediaCentral|UX servers in a cluster.

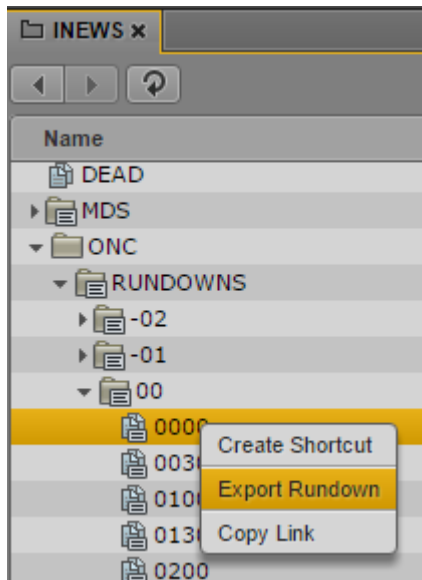It is anticipated that a future release of MediaCentral|UX will remove this requirement.
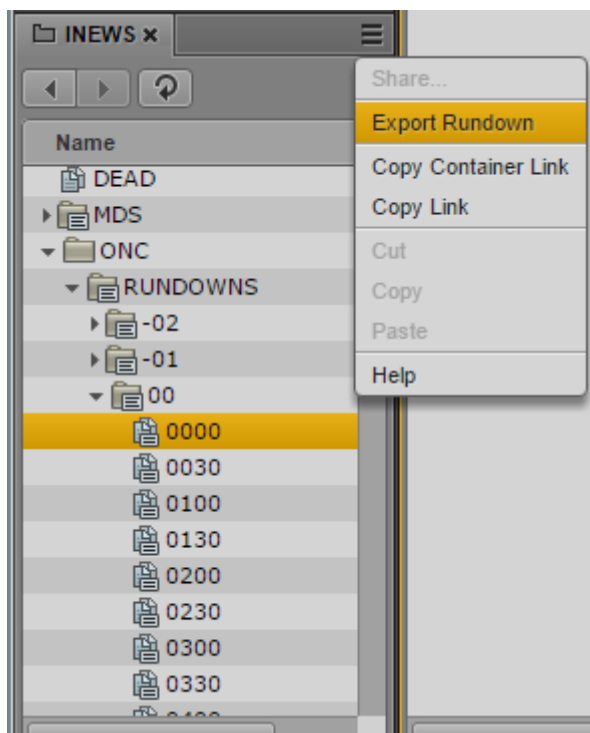
7 Operations

7.1 Exporting a Rundown
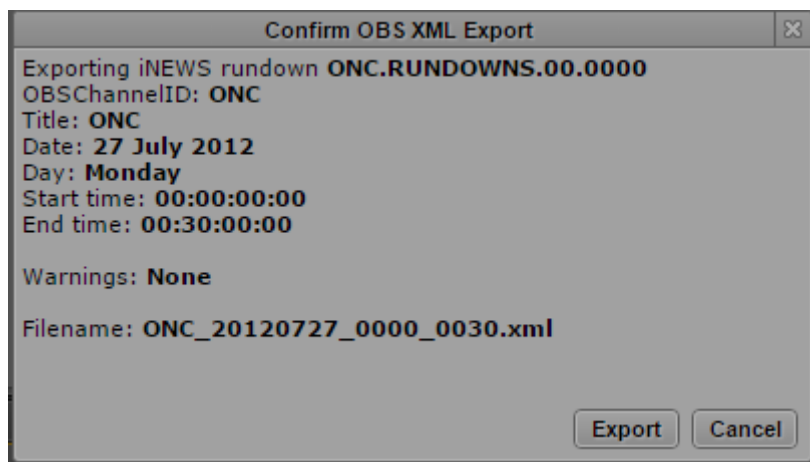
To export a rundown do one of:

- Right click a rundown in an iNEWS Assets tab and select Export Rundown from the pop-up menu
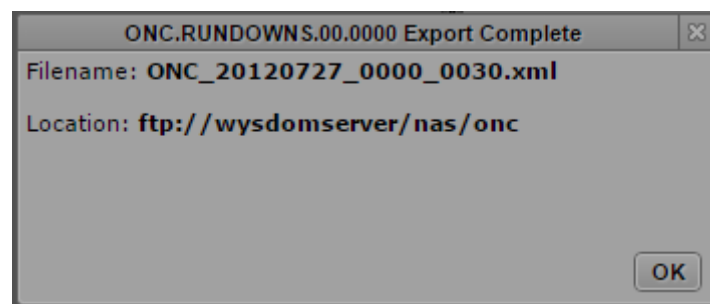

- Select a rundown in an iNEWS Assets tab and choose Export Rundown from the pane menu

The plug-in will format the rundown in preparation for export, in the process making a number of checks to ensure that essential data is present. If preparation succeeds a confirmation dialog will be shown:

```
Confirm OBS XML Export                              [X]

Exporting iNEWS rundown ONC.RUNDOWNS.00.0000
OBSChannelID: ONC
Title: ONC
Date: 27 July 2012
Day: Monday
Start time: 00:00:00:00
End time: 00:30:00:00

Warnings: None

Filename: ONC_20120727_0000_0030.xml


                                       [ Export ]  [ Cancel ]
```
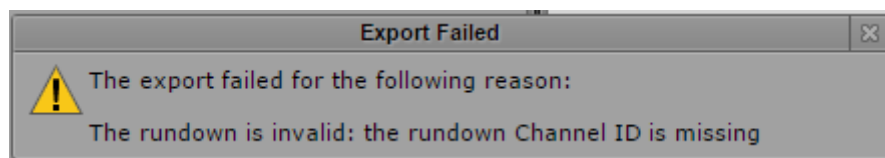
After checking that the result is as expected the rundown can be published to the appropriate FTP destination by clicking the Export button. A dialog confirming the export will then be displayed:

```
ONC.RUNDOWNS.00.0000 Export Complete            [X]

Filename: ONC_20120727_0000_0030.xml

Location: ftp://wysdomserver/nas/onc



                                              [ OK ]
```

Click the OK button to dismiss the dialog.

In the event of a problem with the rundown, e.g. a missing parameter, an error message will be shown such as the example below:

```
Export Failed                                     [X]

  /!\    The export failed for the following reason:

         The rundown is invalid: the rundown Channel ID is missing
```

Some missing parameters may not prevent an export, in such cases a warning is displayed in the export confirmation dialog such as the example below.
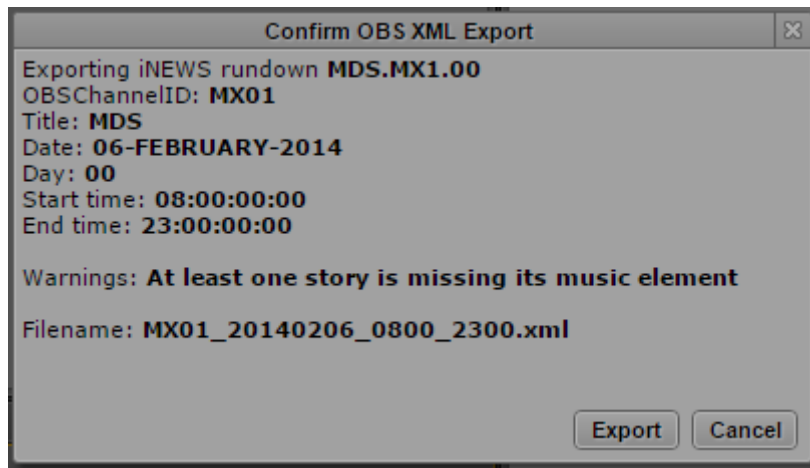
Note that only the first instance of a warning is displayed.
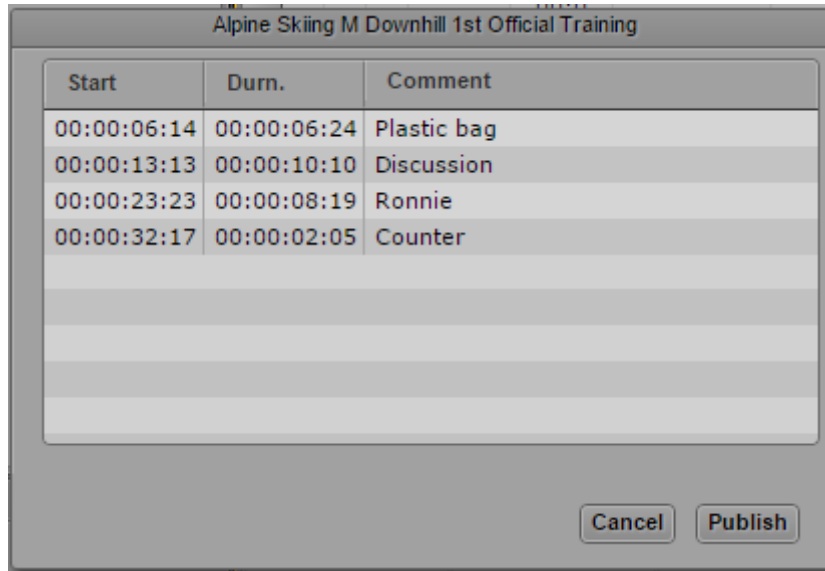
7.2   Publishing a Cue Sheet

Publishing a cue sheet involves the appending of details of any markers (aka locators) present in the video sequence associated with a story. Any markers will be formatted as text and appended to the end of the existing story body. Any previously published marker information will be replaced.

A cue sheet is published by selecting a story in the rundown pane and clicking the Cue Sheet Publish button in the queue pane toolbar:
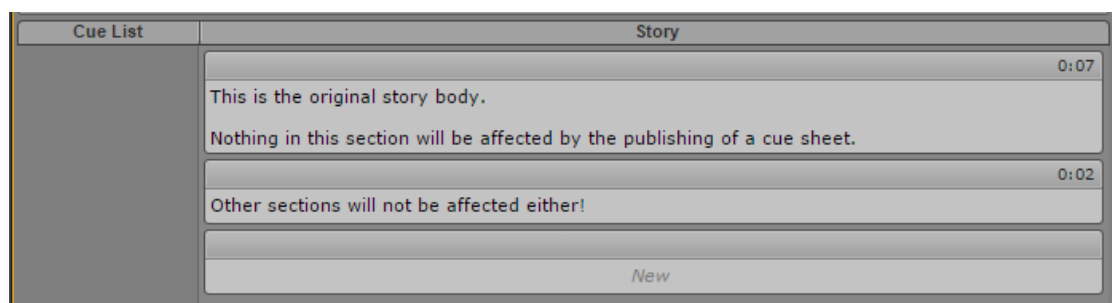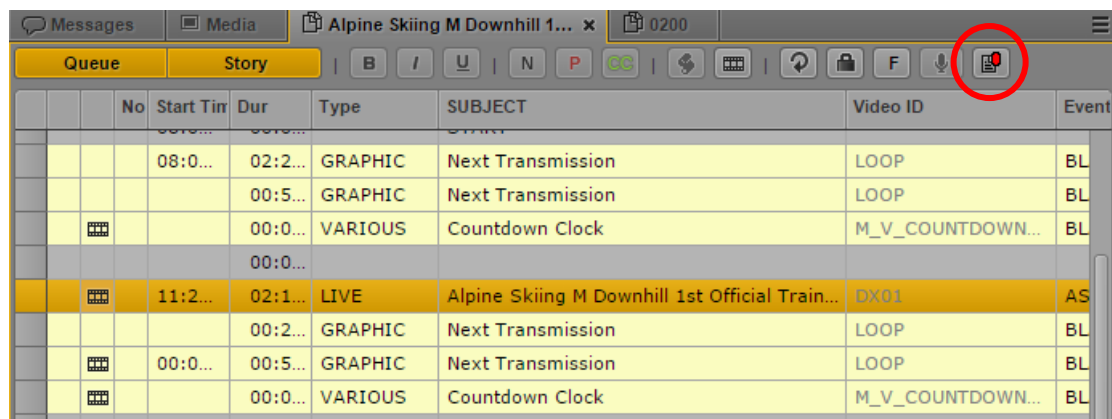

With the desired story selected click the Publish Cue Sheet button to publish the cue sheet. The plug-in will attempt to retrieve details of the associated sequence from the story and then query the sequence for any markers. The result will be shown in a confirmation dialog similar to that shown below:

If the results are as expected click the Publish button to publish the cue sheet to the story. Alternatively click the Cancel button to abort the publishing operation. If the cue sheet is published successfully a confirmation is displayed:
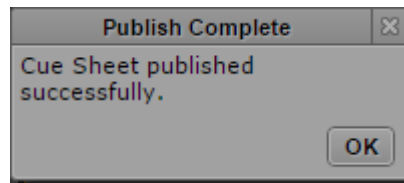
The story body will now contain the published cue sheet information:
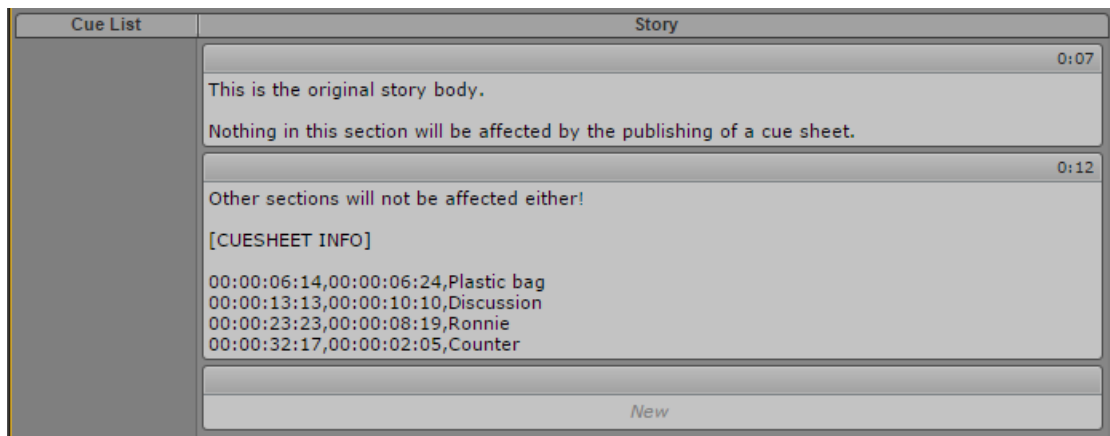
7.2.1 Cue Sheet Publish Fails

It maybe that it is not possible to publish the cue sheet to the story, the most likely reason being that the story body is locked. If the publish action fails an appropriate error message is displayed:

Story Locked

Failed to publish the Cue Sheet because the story is locked
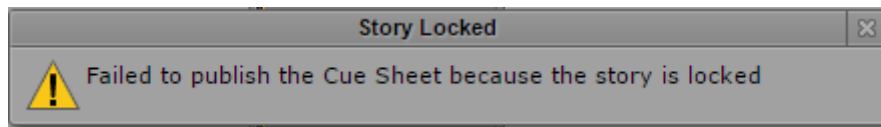
7.3 Export and Publish Authorisation

As mentioned in section 6.5 only certain users may be authorised to export rundowns or publish cue sheets. In order to perform such operations the user must be assigned the appropriate MediaCentral|UX role. In the event that an unauthorised user attempts to perform an export or cue sheet publish an error message is displayed:

Publish Not Allowed

Sorry but you are not permitted to publish cue sheets
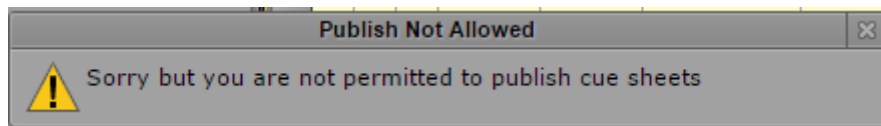
Appendix – Plug-in Architecture

This appendix briefly outlines the architecture of the plug-in.

The plug-in has two distinct components, the user facing elements which are implemented as javascript with backend server processes implemented as java. The backend processes manage the communication with iNEWS and Interplay web services as well as the export of rundowns via FTP. Currently there is another backend process which manages the saving of configuration data, however it is anticipated that a future release will be able to take advantage of the in-built MediaCentral|UX settings database.

Communication between the user facing javascript routines and the backend processes is via RESTful web services typically passing JSON formatted data. This communication mechanism is an inherent part of the MediaCentral|UX framework.

Rundown export is achieved by the following steps:

- The path to the desired rundown (queue path) is passed to the backend by the user facing javascript
- The backend retrieves the stories from iNEWS using iNEWS web services
- The backend processes all stories in the rundown applying the rules defined in the specification document
- If the rundown can be exported it is saved in the backend and key data is returned to the user facing javascript
- If the rundown can be exported the user facing routine displays the confirmation dialog which includes the key data provided by the backend, if it cannot be exported or a failure occurs the appropriate error dialog is displayed
- If the user confirms the export the javascript sends another message to the backend requesting publication of the rundown, if the user cancels it sends a message to the backend instructing it to delete the stored rundown data
- The backend exports the rundown to the appropriate FTP destination and deletes the rundown data. Finally an appropriate response is returned to the user facing routine
- The javascript layer displays the result

Cue Sheet publish is as follows:

- The user facing javascript extracts the sequence MobID from the story data provided by the MediaCentral|UX iNEWS implementation when the user clicks the publish button and sends it to the backend
- The backend uses Interplay web services to retrieve the marker data, this is formatted, stored locally and also returned to the user facing routine
- The javascript layer displays the returned marker data in a confirmation dialog

- If the user confirms the export the javascript layer sends a reference to the stored data to the backend together with details of the story (queue path and locator), if the user cancels the backend is requested to delete the data
- The backend retrieves the story data using iNEWS web services, removes any existing cue sheet data and then appends the new data to the story body. Finally the updated story body is saved back to the story.
- The backend returns the result of the operation to the user facing routine for display to the user.