MediaCentral® | Newsroom Management

Version 2020.9 HTML5 Plugin Specification

Date Revised	Changes Made
23 August 2020	Initial Version for 2020.9

Important Information

Avid recommends that you thoroughly read all information in this document before installing or using any new software release.

Note: Search the Avid Knowledge Base (http://www.avid.com/onlinesupport) for the most up-to-date documentation related to the latest release, including the latest ReadMe file, which contains information that might have become available after the product guides were published.

This document describes the architectural specifications and configuration needed in relation to integration between the MediaCentral | Newsroom Management system and HTML-based plugins. This document also lists any known limitations.

Table of Contents

Introduction

iNEWS Workstation adds the support for HTML based plugins, allowing for an alternative to the currently ActiveX plugin support. With ActiveX being expected to be phased out over time, this gives creators of device plugins the opportunity to transition to a modernized stack. HTML5 is a powerful technology which can ensure long term and full-featured plugin support inside of the MediaCentral Newsroom Management system.

For this purpose, the iNEWS Workstation utilizes a Chromium container that launches HTML-based, 3rd-party plugins, such as those for Maestro graphics plugin. Basically, the Chromium container supports similar interaction with HTML-based plugins as it already does for ActiveX, which means plugins are available to users via the Tools->Plugins menu at the workstation.

One difference is that while ActiveX must be installed on a workstation, and is therefore isolated to those workstations with ActiveX, HTML5, once configured, may be used system-wide without any additional installation.

Note: It should be noted that ActiveX takes precedence over that of HTML5; therefore, on workstations where both are configured, double-clicking on a production cue in a scripted news story will launch the ActiveX plugin over that of an HTML5 plugin if they have the same progID.

The communication between the iNEWS Workstation (NCS client) and the device plugins is based on web messaging methods as defined in "Web Controls as a Replacement for ActiveX in the MOS Protocol, Revision 2.0"[1]. It is assumed that the reader is familiar with the content of this document.

This document describes the individual interactions as supported by the respective version of the iNEWS Workstation and provides architectural context, a configuration overview, and tips for testing plugins.

[1] Snider, Shawn: *Web Controls as a Replacement for ActiveX in the MOS Protocol – Enhancements to the MOS protocol to deprecate ActiveX based plugins in place of web controls*, Revision 2.0, Ross Video Limited, August 13, 2013. http://mosprotocol.com/wp-content/MOS-Meeting-Docs/MOS-Protocol-Web-Control-Proposal-Rev-2.pdf

iNEWS Workstation and HTML-based plugin

Architecture

To embed the plugin into a native application while still using web messages, iNEWS Workstation uses a layered model as show in Figure 1. The workstation embeds Chromium to host a wrapper page. The wrapper page performs the required translation between the web messages and functions used by the native application. Inside the wrapper frame the plugin is hosted in an IFRAME.

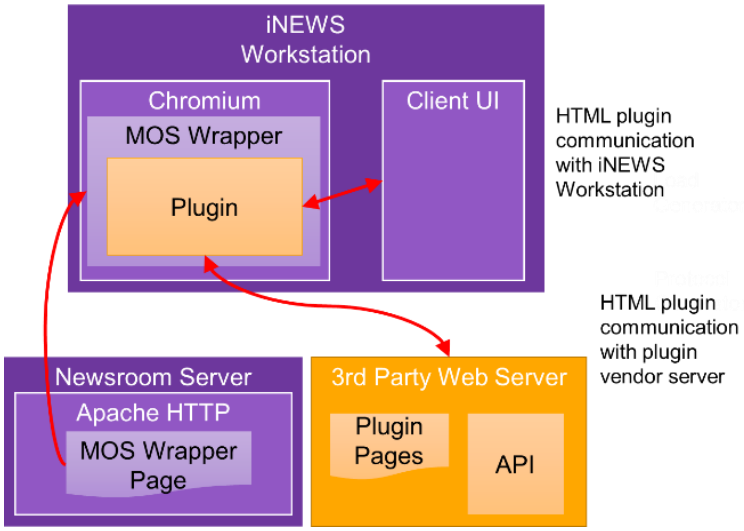

Figure 1:Interactions between Newsroom System and HTML plugin.

The architecture does not require any installation or configuration on the iNEWS Workstation, other than installing the client application itself. The wrapper page is loaded from the MediaCentral | Newsroom Management server and the plugin is loaded from the 3rd party server. All communication of the plugin to the (device) server is to the 3rd party server. The plugin should know its origin as the NCS does not pass further configuration data to the plugin.

Interactions

iNEWS Workstation supports the following types of interaction with plugin:

- Drag and drop a media objects from the plugin to NCS (no specific to HTML5 plugin).
- Open a production cue (media object) from the NCS to the plugin.
- Replace a selected production cue (media object) with another media object in the plugin through the iNEWS context menu item “Replace from Plugin”.
- Apply changes for (media object) items edited inside of the plugin back to the iNEWS client for Editor Plugins.

Each interaction has related user interactions in either the iNEWS Workstation or the plugin. The following subsections detail the interactions.

Initialization

The NCS ORIGIN[2] is provided to the device as a parameter in the invoked URL called "origin". This should be validated whenever a plugins receives a message to ensure it came from the expected window (as shown in the browser example).

Drag and Drop (from plugin to NCS)

When the plugin is open, the user may create an item (media objects) in the plugin. To add the item to a story, the user will drag the item from the plugin and drop it to the story to create a production cue.

The plugin needs to provide an <*ncsItem*> as payload of drag & drop.

Note: The ProgID section needs to match the system setting for it to be able to launch using the plugin.

```
<mos>
<ncsItem>
   <item>
      <itemID>0</itemID>
      <itemSlug>SAMPLE STORY</itemSlug>
      <objID>12345</objID>
      <mosID>iNEWSMOS1</mosID>
      <mosItemBrowserProgID>iNEWS.MosItemBrowser.1</mosItemBrowserProgID>
      <mosAbstract>SOME MOS ABSTRACT</mosAbstract>
   </item>
</ncsItem>
</mos>
```

For more <*ncsItem*> examples refer to the MOS standard documentation.

Open Item (from NCS to Plugin)

iNEWS will launch the plugin if it is not already open and then send a MOS <*ncsItem*> message with the origin information. The plugin uses the content of the item to display it in its UI.

Replace the current Item (from NCS to Plugin)

iNEWS will send an <*ncsItemRequest*> message to the Plugin and the Plugin should package the replacement item as an *ncsItem*(as shown in the *mosMessageFromHost* in the sample)

Applying Changes for Editor Plugins(from Plugin to NCS)

With the Editor plugin it is possible to receive a message from the device. You will get three buttons (OK, Cancel, Apply, placed into the hosting window by iNEWS) at the bottom of the plugin window when you launch a MOS item into the plugin. After modifying the item, you can push it back into iNEWS using the Apply button, where iNEWS Workstation will send an <*ncsItemRequest*> message to plugin and expects to get an <*ncsItem*> with updated data.

[2] According to the "Origins" as described in the MOS-Protocol-Web-Control-Proposal-Rev-2.pdf

Plugin Configuration

The configuration of HTML plugins in MediaCentral | Newsroom Management allows you to configure the access URL, supported program identifiers, and access control for each plugin. The below sections provide a quick overview of the configuration. For a full description of the configuration please consult the Setup and Configuration Guide.

Server-side configuration

The information for each plugin is stored as a story to a dedicated queue. The server-side configuration allows you to specify an alternative queue-name in `/site/dict/queues`.

```
Q_HTML_PLUGINS          /system.html-plugins
```

Client-side configuration

Each plugin is configured using a story in the queue SYSTEM.HTML-PLUGINS (default location). The content of each story follows the below schema:

```
# mandatory
# Plugin URL
URL = https://222.22.222.222:2222/

# mandatory
mosItemBrowserProgID = someMOSPlugin.Browser
mosItemPlayerProgID = someMOSPlugin.Player
mosItemEditorProgID = someMOSPlugin.Editor

# optional
# common user name for login to plugin (for all users)
# if not present, the current iNEWS username will be used
genericUser = Admin

# optional
# name of iNEWS group which is permitted to access plugin
# if not present, no access limitations are applied
permissions = someGroup
```

The MOS Item Program IDs will control the available interactions and menu elements that the iNEWS Workstation provides to the user. They are used to locate the plugin for a MOS item. The title of the story will appear as display value in the menu when the user clicks Tools > Plugins at an iNEWS Workstation.

Testing a Plugin

Workflows

After a plugin is built, the following tests should be performed to verify it is working properly.

- ☐ Basic (browser plugin support)
  - The plugin should be configured properly in the SYSTEM and it should be available in the Tools > Plugins menu.
  - The user should be able to launch the plugin with the menu item.
  - The user should be able to drag & drop a MOS item into the form of a story.
  - The user should be able to drag & drop a MOS item into the body of a story to create a production cue.
  - The user should be able to launch the plugin through its MOS item, by menu or by double-clicking.
- ☐ Player plugin additional support
  - If the configuration of the plugin defines the *mosItemPlayerProgID*, additional menu items should be enabled for MOS items of this plugin.
  - Under the menu Story > Production or under the context menu Play Production Cue three new items—Play, Stop and Pause—should be available.
- ☐ Editor plugin additional support
  - Launching the plugin through its MOS item should display additional buttons on the bottom—Cancel, OK, Apply—which will enable modification of the selected MOS item, using the Apply button should get the production cue updated.

Debugging using Chromium Developer Tools

During development of a plugin it can be very useful to access the plugin code for debugging purposes. The iNEWS Workstation allows access to the Developer Tools of Chromium for the currently loaded window. The access must be actively enabled and is limited to the local workstation.

- ☐ Enabling Developer Tools:
  - Open the Windows registry editor.
  - Navigate to: *Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Environment*
  - If not present, add the registry key *UseChromiumRemoteDebug* (DWORD).
  - Set the registry key to a value of 1 to enable access to the Developer Tools.
    Note: setting the registry key to a value of zero or removing it will disable access to the Developer Tools.
- ☐ Access to Developer Tools:
  - Open Google Chrome web browser.
  - While the plugin is active inside iNEWS Workstation, navigate to the URL: http://localhost:8088/

Sample Plugin

This section guides you through the creation and testing process using a sample Basic plugin. The plugin is a Browser plugin (compared to a Player or Editor plugin, see MOS Item Program IDs in section "Client-side configuration") that lets you:

- Initiate the plugin through the menu selection in the iNEWS Workstation.
- Launch of the selected MOS item into the plugin.
- Drag and drop MOS items into an iNEWS story.
- Replace the current item with the selection in the sample plugin.

Setup

Configure the device as well as the plugin in the SYSTEM area of Newsroom Management:

- In the story SYSTEM.MOS-MAP, add a line, such as:

```
iNEWSMOS1      iNEWSTest1
```

- In the queue SYSTEM.HTML5-PLUGINS, add the following iNEWSBrowser story.

Note: The IP in the URL needs to be the address of the server that is hosting the plugin file.

```
URL = https://192.168.1.123/iNEWSBrowser.html

# data from MOS item content
mosItemBrowserProgID = iNEWS.MOSItemBrowser.1
mosItemPlayerProgID = <value>
mosItemEditorProgID = <value>

;genericUser = demo
;permissions = someGroup
```

- After saving the story and relaunching the iNEWS Workstation, you should see an entry iNEWSBrowser in the list under the menu Tools > Plugins.

Plugin Code

Create a file iNEWSBrowser.html with the below content.

The initialization of the plugin consists of registering the event listener method *mosMsgFromHost*. The plugin then uses the methods to communicate with the NCS:

- *mosMsgFromPlugIn*: sends a message to the NCS via the parent window.
- *mosMsgFromHost*: receives a message from the NCS.

The helper method *getNewsroomOrigin* is used when sending a message or in do verification when receiving a message.

The initialization of the plugin is done in the HTML onload event method *loadStart* to populate the MOS Item area of the plugin.

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta charset="utf-8" />
    <title>iNEWS HTML5 Browser Plugin Sample</title>
    <script type="text/javascript">

    var itemID;

    function getNewsroomOrigin() {
        var qs = document.location.search.split("+").join(" ");
        var params = {};
        var regex = /[?&]?([^=]+)=([^&]*)/g;
        while (tokens = regex.exec(qs)) {
           params[decodeURIComponent(tokens[1])] =
decodeURIComponent(tokens[2]);
        }
        return params['origin'];
    }

    function mosMsgFromPlugIn(message) {
        window.parent.postMessage(message, getNewsroomOrigin());
    }

    function mosMsgFromHost(event) {
        var message = event.data;
        var div = document.getElementById("mosMsgFromHostDiv");
        div.textContent = message;
        var text = div.textContent;
        var parser, xmlDoc;
        parser = new DOMParser();
        xmlDoc = parser.parseFromString(message,"text/xml");

        // Check the Origin in event.origin to ensure it
        // matches our expected NCS origin parameter.
        if (event.origin != getNewsroomOrigin()) {
            return;
        }

        // Handle the Message
        // To Reply, issue a postMessage on the event source.
        if (message.indexOf('<ncsAck>') === -1){
            var reply = "<mos><ncsAck><status>ACK</status></ncsAck></mos>";
            event.source.postMessage(reply, event.origin);
        }

        // handle ncsItemRequest by sending a ncsItem
        if (message.indexOf('<ncsItemRequest>') === -1){
            var reply =
"<mos><ncsItem><item><itemID>0</itemID><objID>55555</objID><mosID>iNEWSMOS1</
mosID> mosItemBrowserProgID>iNEWS.MosItemBrowser.1</mosItemBrowserProgID>
<mosAbstract>REPLACED</mosAbstract><objSlug>REPLACEMENT</objSlug></item></ncs
Item></mos>";
```

```
            event.source.postMessage(reply, event.origin);
        }
    }

    // Register the Event Handler - Cross Browser
    if (window.addEventListener) {
        window.addEventListener('message', mosMsgFromHost, false);
    } else if (window.attachEvent) {
        window.attachEvent('onmessage', mosMsgFromHost, false);
    }

    // create a sample for MOS item
    function loadStart()
    {
        var div = document.getElementById("mosItem");
        div.textContent =
"<mos><ncsItem><item><itemID>0</itemID><itemSlug>SAMPLE
STORY</itemSlug><objID>12345</objID><mosID>iNEWSMOS1</mosID><mosItemBrowserPr
ogID>iNEWS.MosItemBrowser.1</mosItemBrowserProgID><mosAbstract>SOME MOS
ABSTRACT</mosAbstract></item></ncsItem></mos>"
    }

</script>

<style>
  body {background-color:Blue}
  div  {color:Black}
  input.mosInput {
    width:200px;  height:25px;
}
</style>
</head>
<body height="100%" onload="loadStart()">

<table valign="top">

<tr>
<th align="left" width="500"  valign="top">
         <div contenteditable="false" style="color:#FFFFFF"
id="NCSMessage">mosMsgFromHost: <i>(Message received from iNEWS)</i></div>
         <div style="width:500px;height:200px;overflow:auto;background-
color:lightgray" contenteditable="false" id="mosMsgFromHostDiv"></div>
        <p><div contenteditable="false" style="color:#FFFFFF"
id="MOSMessage">MOS item: <i>(Select all of the text to Drag&Drop to
iNEWS)</i></div>
         <div style="width:500px;height:200px;overflow:auto;background-
color:gray" contenteditable="true" id="mosItem"></div>
</th>
</tr>
</table>
</body>
</html>
```

Hosting the file (making sure that the IP is correct, see "Setup"), and you should see the following when launching the plugin inside iNEWS.

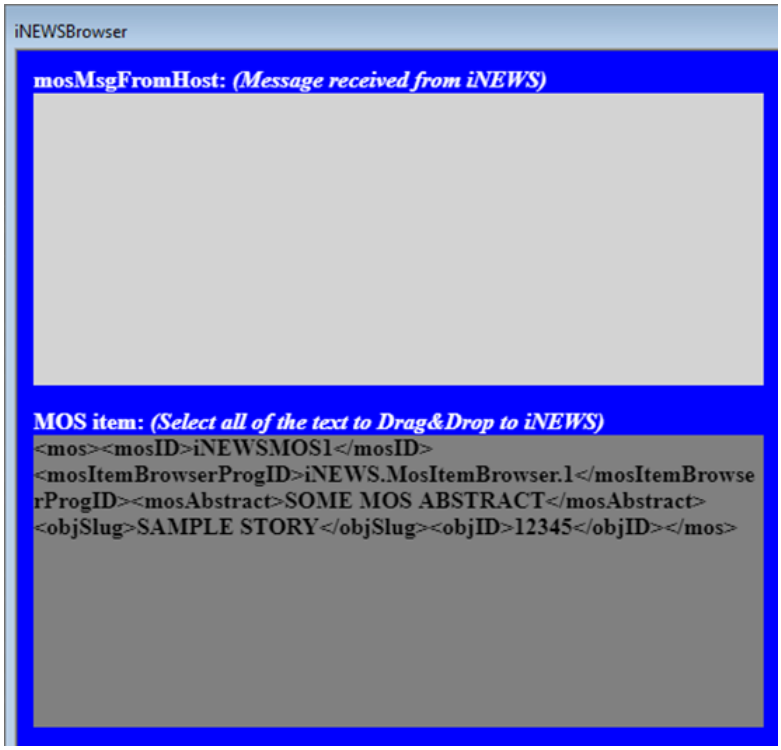

Figure 2: Sample Plugin in the iNEWS Workstation.

Using the Plugin

With the sample plugin running, you should be able to perform the following operations:

- Create a MOS item using the predefined data (You could modify the data but make sure that the *mosItemBrowserProgID* information is not changed; otherwise, you will not be able to open the MOS item back into this plugin.)
  - In the FORM:
    - Use the menu item to launch the plugin.
    - Select all text in the "MOS item" and then drag and drop either into the Story Form of an iNEWS story or into the selected story in the Queue panel.
  - As production cues:
    - Use the menu item to launch the plugin.
    - Select all text in the "MOS item" and then drag and drop into the body of an iNEWS story.
- Launch the selected MOS item into the plugin – iNEWS sends the plugin an *ncsItem* the plugin will display.
  - If the item is in the Story Form, you must use the main menu item *Story > Production Cue > Edit.*
  - If the item is in the body of the story as a production cue, put the cursor in the production cue and double-click.
- Replacing of a selected MOS item – iNEWS sends the plugin an *ncsItemRequest* message and the plugin replies with an *ncsItem* to replace the selected item.
  - If the item is in the Story Form:

        - Set the focus in the Story Form and right-click to display the context menu.
        - Select “Replace from Plugin” menu item.
    - If the item is in the story body as a production cue:
        - Put the cursor in the production cue and right-click to display the context menu.
        - Select “Replace from Plugin” menu item.

Known Limitations

There are no known limitations for HTML-based plugins at this time.

Copyright and Disclaimer

Product specifications are subject to change without notice and do not represent a commitment on the part of Avid Technology, Inc.

The software described in this document is furnished under a license agreement. You can obtain a copy of that license by visiting Avid's Web site at www.avid.com. The terms of that license are also available in the product in the same directory as the software. The software may not be reverse assembled and may be used or copied only in accordance with the terms of the license agreement. It is against the law to copy the software on any medium except as specifically allowed in the license agreement.

Attn. Government User(s). Restricted Rights Legend

U.S. GOVERNMENT RESTRICTED RIGHTS. This Software and its documentation are “commercial computer software” or “commercial computer software documentation.” In the event that such Software or documentation is acquired by or on behalf of a unit or agency of the U.S. Government, all rights with respect to this Software and documentation are subject to the terms of the License Agreement, pursuant to FAR §12.212(a) and/or DFARS §227.7202-1(a), as applicable.

Trademarks

Adrenaline, AirSpeed, ALEX, Alienbrain, Archive, Archive II, Assistant Avid, Avid Unity, Avid Unity ISIS, Avid VideoRAID, CaptureManager, CountDown, Deko, DekoCast, FastBreak, Flexevent, FXDeko, iNEWS, iNEWS Assign, iNEWS ControlAir, Instinct, IntelliRender, Intelli-Sat, Intelli-sat Broadcasting Recording Manager, Interplay, ISIS, IsoSync, LaunchPad, LeaderPlus, ListSync, MachineControl, make manage move | media, MediaCentral, Media Composer, NewsCutter, NewsView, OMF, OMF Interchange, Open Media Framework, Open Media Management, SIDON, SimulPlay, SimulRecord, SPACE, SPACEShift, Sundance Digital, Sundance, Symphony, Thunder, Titansync, Titan, UnityRAID, Video the Web Way, VideoRAID, VideoSPACE, VideoSpin, and Xdeck are either registered trademarks or trademarks of Avid Technology, Inc. in the United States and/or other countries. All other trademarks contained herein are the property of their respective owners.

MediaCentral | Newsroom Management v2020.9 HTML5 Plugin Specification (v7.7) • 10 September 2020

This document is distributed by Avid in online (electronic) form only, and is not available in printed form