From: John Paddleford
Sent: Friday, April 11, 2003 11:44 PM
To: Chris Weinstein (LCA)
Cc: Jeffrey Friedberg; Christina Calio; Geoff Harris
Subject: FW: MusicNet use of wmstubdrm

Privileged

---- Original Message ----
From: Jeff Wallace [mailto:jwallace@MusicNet.com]
Sent: Friday, April 11, 2003 5:14 PM
To: John Paddleford; Christina Calio
Subject: MusicNet use of wmstubdrm

John, Christina

Here is the document I promised on how MusicNet is going to use the *wmstubdrm.lib* so we can execute the Windows Media DRM addendum. It covers what interfaces we are using along with how we are going to deploy it in the MusicNet SDK.

<<Using Windows Media FormatSDK.doc>>

Jeff Wallace
Senior Director of Engineering
MusicNet
(206) 269-6040 Direct
(206) 604-5389 Cell
jwallace@musicnet.com

Plaintiff's Exhibit
7214
Comes V. Microsoft

Exhibit: 11
Wit: Paddleford
Date: 5-06-04
Emi Albright, CCR RPR
Esquire Deposition Services

MS-CC-Bu 000000365895
CONFIDENTIAL

Using Windows Media FormatSDK

Author: Sorin Jianu (sorin@musicnet.com)
Date: April 10, 2003

This document describes the features present in the MusicNet Client SDK 3.0 that require the use of Microsoft Windows Media Format SDK Series 9. The interface definitions included in this document are publicly available either in the IDL files or the type libraries delivered with MusicNet Client SDK 3.0.

Overview

The 3.0 release of MusicNet Service offers support for protected WMA content.

MusicNet Client SDK 3.0 exposes high level APIs for player individualization, license acquisition, license registration, and reading of license properties and content metadata. In the current release there is no SDK support for WMA content creation, editing, and playback. Applications built on top of MusicNet Client SDK 3.0 must implement playback functionality either through Windows Media Player Control or Windows Media Player Application.

Mainly, MusicNet Client SDK 3.0 is using the `IRMGetLicense` interface to manage the license acquisition and registration, while the `IWMReader` and `IWMDRMReader` are being used to get DRM properties.

MusicNet Client SDK 3.0 is built and distributed as a set of COM components hosted in several DLLs. There are no other exports from these DLLs other than the 4 standard exports required by COM.

Of all the client SDK modules, only MNWMRM.dll module is statically linking the WMStubDRM library.

The COM interfaces implemented by MNWMRM module which make use of FormatSDK functionality are defined further in this document. There are no other documented or undocumented ways to access code that is otherwise available in the WMStubDRM. In addition, since MNWMRM uses only a small part of the objects, functions, and interfaces available in the WMStubDRM, a significant portion of the library is not linked in the MNWMRM module.

Individualization

The support for individualization is exposed in the following interfaces:

```
interface IMNWMIndividualization : IDispatch
{
    [id(1), helpstring("method Individualize")]
    HRESULT Individualize([in] BSTR fileUrl);
    [id(2), helpstring("method CancelIndividualization")]
    HRESULT CancelIndividualization();
};

interface _IMNWMIndividualizationEvents : IDispatch
{
      [id(1), helpstring("method OnBegin")]
      HRESULT OnBegin();
      [id(2), helpstring("method OnSucceed")]
      HRESULT OnSucceed();
      [id(3), helpstring("method OnFail")]
      HRESULT OnFail(long result);
      [id(4), helpstring("method OnCancel")]
      HRESULT OnCancel();
      [id(5), helpstring("method OnDownload")]
      HRESULT OnDownload(long readTotal, long readProgress);
      [id(6), helpstring("method OnInstall")] HRESULT OnInstall();

};
```

The implementation of `Individualize` method creates a WMReader, queries for `IWMDRMReader` and then calls `IWMDRMReader::Individualize`. During the individualization process, events are generated for an external observer to monitor the individualization.

License Acquisition and Registration

Support for License Acquisition and license registration is available through `IMNDRMPlugIn2::GetFormatedClientInfo` and `IMNWMRM::RegisterLicense` methods, which are part of the interfaces further below:

```
interface IMNDRMPlugIn2 : IMNDRMPlugIn
{
      [id(25), helpstring("method GetFormatedClientInfo")]
      HRESULT GetFormatedClientInfo([out, retval] BSTR* pXML);
};

interface IMNWMRM : IMNDRMPlugIn2
{
      [id(5), helpstring("method RegisterLicense")]
      HRESULT RegisterLicense([in] BSTR license);
      [id(6), helpstring("method IsLicenseValid")]
      HRESULT IsLicenseValid[[in] BSTR fileName, [out,retval]
VARIANT_BOOL* pValid);
      [id(7), helpstring("method GetRemainingTime")]
```

```
      HRESULT GetRemainingTime([in] BSTR fileName, [out,retval]
VARIANT* pTime);
      [id(8), helpstring("method HasPermanentLicense")]
      HRESULT HasPermanentLicense([in] BSTR fileName, [out,retval]
VARIANT_BOOL* pValid);
      [propget, id(9), helpstring("property RemainingBurnCount")]
      HRESULT RemainingBurnCount([in] BSTR fileName, [out, retval]
LONG* pVal);
      [propget, id(10), helpstring("property
      RemainingPDTransferCount")]
      HRESULT RemainingPDTransferCount ([in] BSTR fileName, [out,
      retval] LONG* pVal);
};
```

To request and register licenses, the implementation of `IMNDRMPlugIn2::GetFormatedClientInfo` and `IMNWMRM::RegisterLicense` uses the RMGetLicense object, documented in the Windows Media Rights Manager but actually distributed with the FormatSDK.

`IMNDRMPlugIn2::GetFormatedClientInfo` calls `IRMGetLicense::GetSystemInfo` to get DRM system information from the client computer. This information is formatted and later on used in the license acquisition request.

`IMNWMRM::RegisterLicense` calls `IRMGetLicense::StoreLicense` to store the license on the client computer.

License Properties and Metadata

The `IMNWMRM` has several methods and properties that require reading of DRM properties associated with a WMA files or their corresponding licenses.

The names of the methods and properties of the remaining `IMNWMRM` interface are self-explanatory.