SCIDM Technical Specification

Candidate Version 1.0 – 09 Feb 2010

Open Mobile Alliance
OMA-TS-SCIDM-V1_0-20100209-C

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Use of this document is subject to all of the terms and conditions of the Use Agreement located at http://www.openmobilealliance.org/UseAgreement.html.

Unless this document is clearly designated as an approved specification, this document is a work in process, is not an approved Open Mobile Alliance™ specification, and is subject to revision or removal without notice.

You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. Information contained in this document may be used, at your sole risk, for any purposes. You may not use this document in any other manner without the prior written permission of the Open Mobile Alliance. The Open Mobile Alliance authorizes you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services. The Open Mobile Alliance assumes no responsibility for errors or omissions in this document.

Each Open Mobile Alliance member has agreed to use reasonable endeavors to inform the Open Mobile Alliance in a timely manner of Essential IPR as it becomes aware that the Essential IPR is related to the prepared or published specification. However, the members do not have an obligation to conduct IPR searches. The declared Essential IPR is publicly available to members and non-members of the Open Mobile Alliance and may be found on the “OMA IPR Declarations” list at http://www.openmobilealliance.org/ipr.html. The Open Mobile Alliance has not conducted an independent IPR review of this document and the information contained herein, and makes no representations or warranties regarding third party IPR, including without limitation patents, copyrights or trade secret rights. This document may contain inventions for which you must obtain licenses from third parties before making, using or selling the inventions. Defined terms above are set forth in the schedule to the Open Mobile Alliance Application Form.

NO REPRESENTATIONS OR WARRANTIES (WHETHER EXPRESS OR IMPLIED) ARE MADE BY THE OPEN MOBILE ALLIANCE OR ANY OPEN MOBILE ALLIANCE MEMBER OR ITS AFFILIATES REGARDING ANY OF THE IPR’S REPRESENTED ON THE “OMA IPR DECLARATIONS” LIST, INCLUDING, BUT NOT LIMITED TO THE ACCURACY, COMPLETENESS, VALIDITY OR RELEVANCE OF THE INFORMATION OR WHETHER OR NOT SUCH RIGHTS ARE ESSENTIAL OR NON-ESSENTIAL.

THE OPEN MOBILE ALLIANCE IS NOT LIABLE FOR AND HEREBY DISCLAIMS ANY DIRECT, INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR EXEMPLARY DAMAGES ARISING OUT OF OR IN CONNECTION WITH THE USE OF DOCUMENTS AND THE INFORMATION CONTAINED IN THE DOCUMENTS.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms set forth above.

Contents

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Figures

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Tables

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

1. Scope

This document presents the complete Technical Specification of Secure Content Identification Mechanism (SCIDM) Version 1.0, including the transactions, message types and their information elements, status and error codes, and other details.

The following items are within the scope of this Technical Specification of the SCIDM 1.0 Enabler:

- Transactions
- Definition of the transaction message types and their information elements, status and error codes.
- Definition of the formats of the content identity, and content ID certificates
- Guidelines for the use of metadata in SCIDM
- Guidelines for the use of content fingerprinting algorithms in SCIDM
- Guidelines for the deployment of SCIDM (i.e., Deployment Model), including whether SCIDM Server (CIM) is deployed as a platform for identification or embedded in a specific application and is dedicated to that application, the deployment location of SCIDM Client and its interoperation with the application that uses the SCIDM for content identification.
- Guidelines on the performance issues, including some guidelines about how to address the limitations of identification of real-time, large-scale content, and how to reduce the delay experienced by the User during content access.
- The adaptation of SCIDM messages to some commonly used transport protocols

The following items fall outside the scope of this Technical Specification of the SCIDM v1.0 Enabler:

- The definition of Content Metadata
- The formal specification of content fingerprinting algorithms
- The formal specification of interoperation between CIM/SCIDM Client and the application.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

2. References

2.1 Normative References

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt

[RFC2616] "Hypertext Transfer Protocol – HTTP/1.1". Fielding R.; Gettys J.; Mogul J.; Frystyk H.; Masinter L.; Leach P.; Berners-Lee T., June 1999. URL: URL:http://www.ietf.org/rfc/rfc2616.txt

[RFC2617] "HTTP Authentication – Basic and Digest Access Authentication". J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart, June 1999. URL: URL:http://www.ietf.org/rfc/rfc2617.txt

[RFC4234] "Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell. October 2005, URL:http://www.ietf.org/rfc/rfc4234.txt

[RFC4346] "The Transport Layer Security (TLS) Protocol Version 1.1", T. Dierks, E. Rescorla. April 2006, URL:http://www.ietf.org/rfc/rfc4346.txt

[RFC4347] "Datagram Transport Layer Security", E. Rescorla, N. Modadugu. April 2006, URL:http://www.ietf.org/rfc/rfc4347.txt

[SCIDM-AD] "Secure Content Identification Mechanism Architecture", Open Mobile Alliance™, OMA-AD-SCIDM-V1_0, URL:http://www.openmobilealliance.org/

[ISO-MPEG-7] ISO/MPEG N4674, Overview of the MPEG-7 Standard, v 6.0, J.M. Martínez, ed., MPEG Requirements Group, Jeju, Mar. 2002

[MPEG-7-IMG-SIG] "ISO/IEC 15938-3:2002/Amd.3 Image Signature Tools," Mar. 2009

[OMA-SEC_CF-AD] "Application Layer Security Common Functions Architecture", Open Mobile Alliance™, OMA-AD-SEC_CF-V1_0, URL:http://www.openmobilealliance.org/

[PUSH-OTA] "Push Over The Air", Version 25-April-2001, Open Mobile Alliance™, WAP-235-PushOTA-20010425-a, URL:http://www.openmobilealliance.org/

[PUSH-PAP] "Push Access Protocol", Version 29-Apr-2001, Open Mobile Alliance™, WAP-247-PAP-20010429-a, URL:http://www.openmobilealliance.org/

[SCIDM-XSD] "OMA SCIDM XML Schema", Version 1.0, Open Mobile Alliance™, OMA-SUP-XSD_SCIDM-V1_0, URL:http://www.openmobilealliance.org/

[SCRRULES] "SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

[SEC_CF] "OMA Application Layer Security Common Functions V1.0" , Open Mobile Alliance™, OMA-ERP-SEC_CF-V1_0-20080902-A.zip, URL:http://www.openmobilealliance.org/

2.2 Informative References

[IEEE-802.3-2008] "LAN/MAN CSMA/CD (Ethernet) Access Method", IEEE Standard for Information technology-Specific requirements - Part 3: Carrier Sense Multiple Access with Collision Detection (CMSA/CD) Access Method and Physical Layer Specifications, URL: http://standards.ieee.org/getieee802/802.3.html

[ISO 639-1] "Codes for the Representation of Names of Languages Part 1: Alpha-2 Code", Jul 2002, ISO, URL: http://www.iso.org

[ISO 639-2] "Codes for the Representation of Names of Languages Part 2: Alpha-3 Code", Oct 1998, ISO, URL: http://www.iso.org

[OMADICT] “Dictionary for OMA Specifications”, Version 2.3, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_7, URL:http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

Compound Content A content item that consists of multiple individually recognizable content objects.

Content Metadata Information about a content item, such as content attributes (e.g. the title, ID, associated category, description, or perspectives) or data associated with the content (e.g. geo-information about where the content was produced).

Content Fingerprint A short “summary” derived from a content item that can uniquely identify the content item.

Content ID A symbol (e.g. number or string) that establishes the identity of the content of the services to be used during its lifecycle (e.g., the assignment, the registration, the query, the verification, etc.).

Content ID Certificate A certificate used to verify the identity of a content item.

Content Provider Entity that provides content for user consumption, usually in exchange for profit. This includes traditional content providers such as label companies, as well as individuals.

Content Registrant Entity that initiates the content registration to the CIM. Registration Client is run by this entity.

Digital Watermark Auxiliary data that is imperceptibly and persistently embedded into an original content such as image, video and audio. This auxiliary data can subsequently be recovered from the watermarked content. Digital Watermark can be used to identify a content item, to verify its integrity, to authenticate the content with embedded copyright mark, to include meta data, etc.

SCIDM Client An entity that makes requests to the CIM for content registration, content identification, content query and content verification.

SCIDM Server An entity that performs the content registration, content identification, content query and content verification in response to the request from SCIDM Client. It also issues Content ID Certificates. It is also referred to as the Content Identity Manager.

Registration Client A SCIDM Client that makes requests to the CIM for content registration

Identification Client A SCIDM Client that makes requests to the CIM for content identity query and content verification

3.3 Abbreviations

AD	Architecture Document
AFP	Audio FingerPrint
CIM	Content Identity Manager
ID	Identifier
IFP	Image FingerPrint
ISO	Internaltional Standards Organization
OMA	Open Mobile Alliance
SCIDM	Secure Content IDentification Mechanism
UGC	User Generated Content
VFP	Video FingerPrint

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

XFP	eXtensible FingerPrint

4. Introduction

This document defines the technical framework and specifies globally interoperable technologies for secure content identification mechanism over different distribution systems based on the SCIDM requirement documents [SCIDM-RD] and the SCIDM architecture document [SCIDM-AD].

Section 5 describes the lifecycles/transactions in the SCIDM Enabler. In Section 6, the normative behaviours of SCIDM Client and SCIDM Server are defined, followed by the definition of the transaction messages and the information elements of each interface in Section 7. Metadata packaging schemas are defined in Section 8. Section 9 defines the representation scheme for content fingerprinting algorithms and their values, as well as the fingerprint containter file format. Aspects of SCIDM security are covered in Section 10. Error handling of all SCIDM operations and transactions are defined in Section 11. Adaptations of SCIDM to transport bearers are defined in Section 12.

4.1 Version 1.0

The OMA SCIDM Enabler v1.0 supports the following functionalities:

- Registration of content and its Content Metadata
- Identification of various content items that have been registered
- Flexible ways to query for the content item and Content Metadata by
  - Content ID
  - Content Fingerprint
  - Other Content Metadata

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

5. SCIDM processes/lifecycle

This section describes the SCIDM processes.

5.1 Content Registration，Update and De-registration

Content Registration is the process in which the Registration Client registers content on which some identification and management actions will be performed to the CIM.Content Registration is the premise of Content Identification and only registered content can be identified. The Content Registration is done via the interface SCIDM-1.

Content Registration is initiated by the Content Registrant. Content Registrant varies depending on the specific applications. For example, in copyrighted content detection application, Content Registrant can be the Content Provider, or UGC owner; in the spam filtering application, Content Registrant can be the network operator or the spam filtering service provider.

In Content Registration, the Registration Client SHALL send the content itself or the Content Fingerprint to the CIM for registration. In the former case, the CIM SHALL extract the Content Fingerprint of the content and store it. Whether the CIM stores the content itself is out of the scope of this enabler. In the latter case, the Registration Client SHALL send the information about the algorithm used to extract the Content Fingerprint to the CIM together with the Content Fingerprint.

Content Registration SHOULD provide necessary metadata, such as the registration purpose. For more detailed information needed in the Content Registration, please see Section 7.1.1.

Upon the registration request from the Registration Client, the CIM SHALL judge the trustworthiness of the Content Metadata submitted. The CIM SHOULD leverage reasonable mechanisms to judge the trustworthiness. The CIM SHALL differentiate different principals, such as traditional Content Provider, individual user, proxy agent, the operator of the Enabler, etc. For more detailed information about how to perform the differentiation, please see [SCIDM-AD].

The SCIDM Enabler SHALL support the registration of User Generated Content. For the UGC registration, the SCIDM Enabler SHALL support credential mechanisms to improve the trustworthiness of the provided metadata. With the credential mechanisms, the Content Registration SHALL provide the CIM with some credentials of the registered metadata provided by a proxy entity (e.g. acting on behalf of a trustable website).

The CIM SHALL allocate a globally unique ID for each registered content item. Unless otherwise specified, Content ID refers to the globally unique ID allocated by the CIM upon registration.

After the CIM completes the registration, it can create a Content ID Certificate for the registered content item. The certificate SHALL contain the content's global unique ID allocated by the CIM, the Content Fingerprint, and metadata that is necessary in some application. The certificate can also be used to indicate that the content is successfully registered. The certificate is used for offline identification (See Section 5.4).

When the Content Registrant needs to modify the registered Content Fingerprint or metadata of the content, the Registration Client can initiate the Content Update process to the CIM. The Content Update SHALL send the Content Fingerprint or metadata to be changed to the CIM. The Content Update SHALL NOT cause the CIM to change the global unique ID allocated before by the CIM since the Content ID corresponds to the content. If a content item is changed, it MAY be registered to the CIM as a new content item.

The SCIDM Enabler SHALL support the deregistration of a registered content item, which is done via the Content Deregistration process.

Mutual authentication between the Registration Client and the CIM SHALL be performed before the Content Registration, Content Update, and Content Deregistration. For more information about the authentication mechanism, please see Section 10.

Figures 1-3 illustrate the process flows of Content Registration, Content Update, and Content Deregistration.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

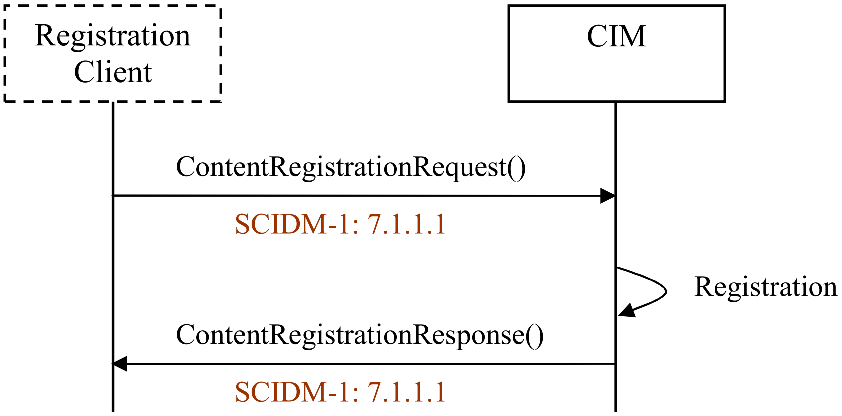


Figure 1: Content Registration

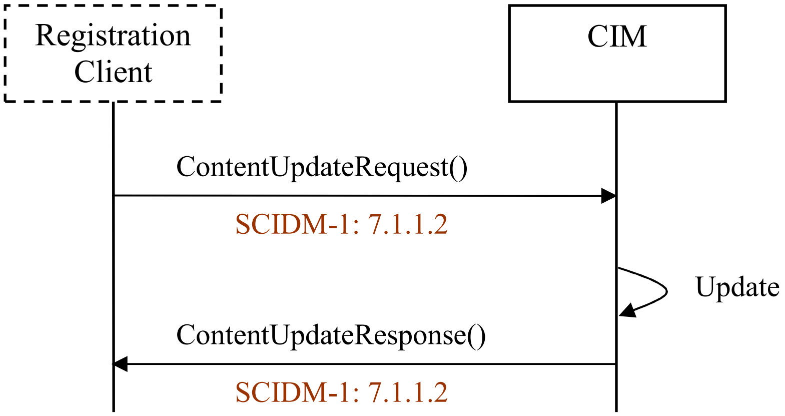


Figure 2: Content Update

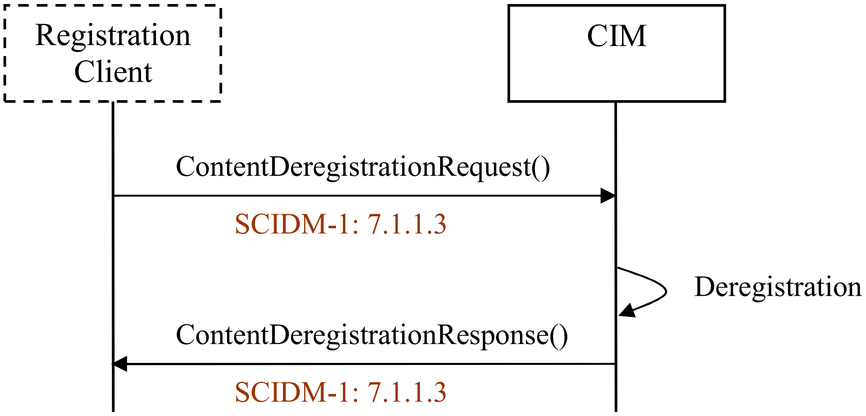


Figure 3: Content Deregistration

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

5.2 The Content Identification Mechanisms

The basic identification mechanism that the SCIDM Enabler SHALL support is identification by Content Fingerprint. For the efficiency consideration, the SCIDM Enabler SHALL support multiple identification mechanisms in addition to identification by Content Fingerprint. The identification mechanisms to be supported are defined as follows.

- **Identification by ID**. Here the ID is the globally unique Content ID allocated by the CIM upon Content Registration. To improve the accuracy, some auxiliary information such as the content size, etc. SHOULD be provided when using this identification mechanism. This identification mechanism SHALL be support by the SCIDM Enabler. Identification by ID is abbreviated as ByID in this document.

- **Identification by history.** The CIM SHALL record the identification history information which can be used for later identification. The identification history information SHOULD include the source/destination information (e.g., the IP address, domain name, subscriber ID, etc) of the content delivery, previous identification results, etc. This identification mechanism SHALL be supported by the SCIDM Enabler and SHOULD be used together with other identification mechanisms. Identification by history is abbreviated as ByHistory in this document.

- **Identification by metadata**. This identification mechanism identifies the content by its metadata, where the metadata mainly refers to the hash value (e.g., MD5, SHA) and other content attributes. This mechanism SHALL be supported by the SCIDM Enabler. Identification by metadata is abbreviated as ByMetadata in this document.

- **Identification by watermark**. Inside the SCIDM Identification Client and the CIM, identification by watermark and by Content Fingerprint refer to different algorithms, different security/accuracy level and different efficiency. Yet the protocol and interfaces between the Identification Client and the CIM can be unified. In this Technical Specification, identification by watermark and identification by Content Fingerprint are treated as the same in the interface except the ID of the identification mechanism. For more detailed definition, please see Section 7. This identification mechanism SHOULD be supported by the CIM and MAY be supported by the Identification Client. Identification by watermark is abbreviated as ByWatermark in this document.

- **Identification by Content Fingerprint**. In the context of the SCIDM Enabler, Content Fingerprint is a short "summary" derived from a content item that can uniquely identify the content item. Usually the "summary" refers to a value calculated using some features that extracted from the content. The Content Fingerprints can be used to identify perceptually identical copies of content and match them. Even if the content has been manipulated, as long as it is still similar to the original, identification by Content Fingerprint may still recognize it and match it to its original source. The types and degrees of manipulation that identification by Content Fingerprint can tolerate are dependent on the application scenario and the specific Content Fingerprint algorithm. They are out of the scope of this enabler. The identification by Content Fingerprint is abbreviated as ByFingerprint in this document.

These five identification mechanisms, in the order defined above, are from simple to complex, from high efficiency to low efficiency, while from low accuracy to high accuracy. If identification using low accuracy mechanisms fails to find a matched content item, identification using higher accuracy mechanisms may be successful. But a successful identification using low accuracy mechanisms does not mean that identification using higher accuracy mechanisms will succeed, as the former identification may be incorrect or insecure. The application SHOULD decide which mechanisms to use based on its network environment and the cost introduced by the potentially incorrect identification.

5.3 Online Content Identification and Query

Online identification and query is the process in which the Identification Client communicates with the CIM for content identification or Content Metadata query. Online Identification is to acquire the globally unique ID of the content based on the identification mechanisms defined in Section 5.2. Online Query is to acquire some metadata of the content based on the identification mechanisms or directly based on the content's globally unique ID. The Identification Client SHALL specify what it wishes to acquire via a request type parameter in its request to the CIM. The Online Identification and Query is accomplished via the interface SCIDM-2.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Upon receiving an identification request for a content item from the application, the Identification Client first selects an appropriate identification mechanism (see Section 5.2), mainly according to the load of the Identification Client, application scenarios or the preset security level demanded by the application.

For different content types, the Identification Client SHALL select the corresponding algorithm appropriate for the type. For more information about the identification algorithms, please refer to Section 9.

After some identification mechanism and algorithm are selected, the Identification Client SHALL collect the information needed for the selected mechanism. For information needed by each identification mechanism please refer to Section 5.2. As to the ByFingerprint mechanism, the Identification Client MAY extract the Content Fingerprint and send the Content Fingerprint to the CIM in the identification request or send the content itself to the CIM. In the latter case, the CIM SHALL extract the Content Fingerprint and then perform the identification operation.

After selecting the identification mechanism and collecting the necessary information, the Identification Client sends an identification request to the CIM. Upon receiving the identification request, the CIM SHALL use the identification mechanism specified by the Identification Client to search for the matched content. If a matched content is found, the CIM SHALL return the Content ID to the Identification Client if the request type is Online Identification and return the requested Content Metadata if the request type is Online Query. If no matched content is found, the CIM SHALL respond to the Identification Client with a failure message.

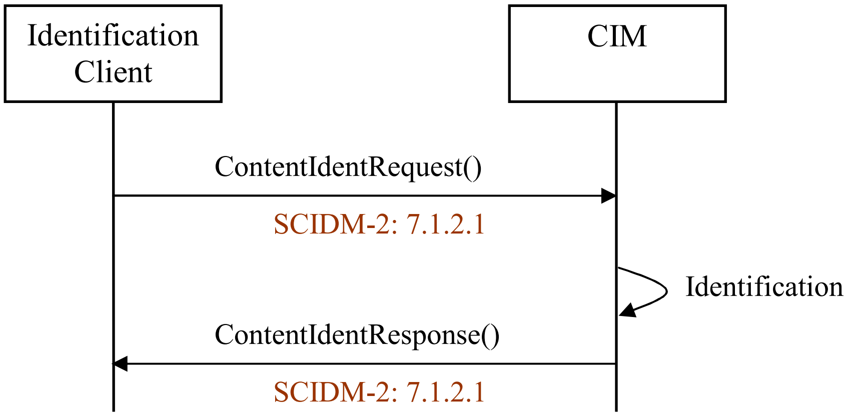
If one identification mechanism fails, the CIM MAY send the Identification Client some recommendation about the identification mechanism to be used that has higher probability to succeed. The Identification Client MAY decide by itself whether to choose another mechanism according to its load, pre-configuration, and CIM's recommendation. If the failed mechanism is ByFingerprint, the Identification Client SHALL NOT select another mechanism.

With the ByFingerprint mechanism, the SCIDM Enabler SHALL support the identification of content based on his partial piece. This is to reduce the network traffic and improve the efficiency. If the identification based on the partial piece succeeds, the CIM SHOULD return the match information between the partial piece and the whole content, such as the position at which the piece is in the content, the matched percentage. These kinds of information are for the Identification Client to do some judgement related to the application.

The SCIDM Enabler SHALL support the identification of Compound Content. For Compound Content, the Identification Client SHALL select appropriate identification mechanism for each member object individually and collect the information of each one. The CIM SHALL do the identification and return the result for each object.

The SCIDM Enabler MAY support the Identification Client to set the time limitation used to accomplish the identification task. The Identification Client MAY decide the time limitation based on the application configuration and set this parameter in the identification request if needed. The CIM MAY choose to notify the Identification Client that it can not finish the identification task before the time limitation. If the Identification Client does not receive the response before the time limitation, it MAY treat the identification as failure.

The Identification Client MAY request the CIM to return the registration information of the content to be identified for later offline identification. If the identification succeeds, the CIM SHOULD return the registration information as requested. The registration information SHOULD be sent to the Identification Client in the form of Content ID Certificate. For more information about offline identification, please refer to Section 5.4.



© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Figure 4: Online Content Identification and Query

5.4 Offline Content Identification

In Offline Content Identification, the Identification Client does not connect to the CIM for content identification and query, but uses the content registration information acquired earlier. Offline Content Identification is to reduce the delay of the network transmission and the burden of the CIM.

The way in which the Identification Client obtains the content registration information includes:

- Acquire the Content ID Certificate attached to the content
- Request the CIM to return them after the successful identification in the Online Content Identification and Query process (Section 5.3)
- Request the CIM to return them through the registration information synchronization process via the interface SCIDM-3 (Section 5.4.1).

When the Identification Client needs to identify a content item, it SHOULD first search its registration information locally and perform Offline Content Identification if found. Otherwise, the Identification Client SHALL connect to the CIM for Online Content Identification and Query.

Offline Content Identification MAY also use different identification mechanisms (Section 5.2) determined by the application.

The registration information is usually acquired in the form of Content ID Certificate. Before the certificate is used, the Identification Client SHALL first verify it, including verifying the validity of the CIM’s signing, the validity of date/time. For the definition of Content ID Certificate, please see Section 5.4.2.

5.4.1 Registration Information Synchronization

Registration information synchronization is the process of synchronizing content registration information from CIM to Identification Client. The synchronization MAY be initiated by either the Identification Client or the CIM. The initiation of the synchronization may be based on some policies preconfigured in the CIM or the Identification Client, both of which are out of the scope of this enabler.

If the Identification Client intends to initiate a synchronization process, it SHALL send a synchronization request to the CIM. In the request, the Identification Client SHALL provide the date and time of last synchronization, and SHOULD provide the application type for which it will use the registration information. Upon receiving the request from the Identification Client, the CIM SHALL retrieve all the registration information relevant to the request and return them to the Identification Client. The registration information returned to the Identification Client SHOULD be in the form of Content ID Certificate.

If the CIM intends to initiate a synchronization process, it SHALL send a notification message to the Identification Client. Upon receiving the notification, the Identification Client SHALL issue a request to the CIM which is the same as the case where the Identification Client initiates the request except a tag showing that the request is initiated by the CIM is set. Upon receiving the request, the CIM SHALL send what it has prepared to synchronize to the Identification Client.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

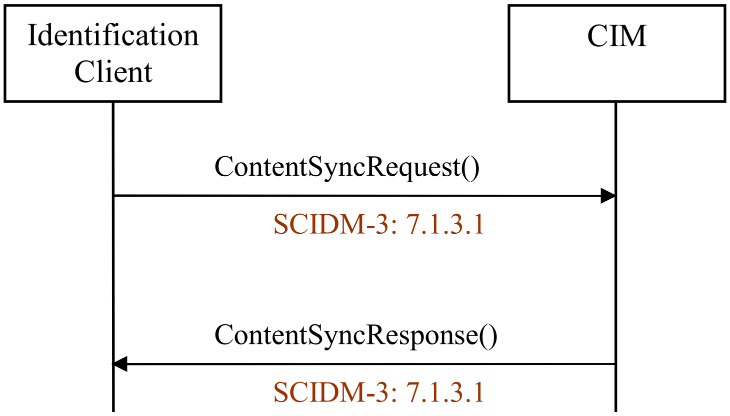


Figure 5: Identification Client Initiated Registration Information Synchronization

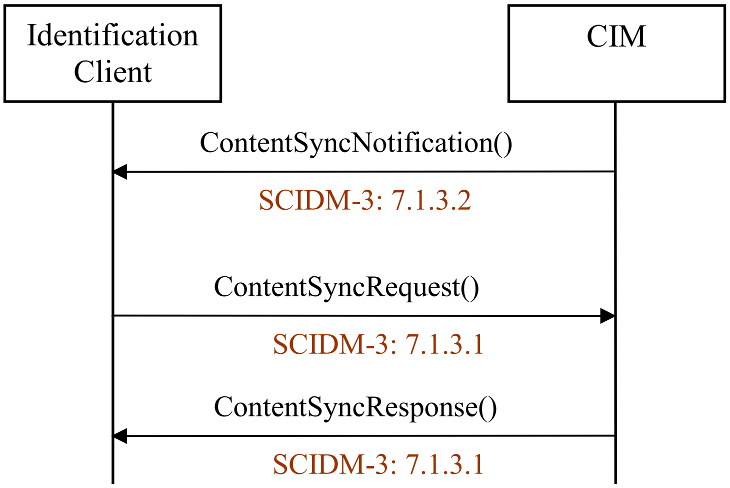


Figure 6: Server Initiated Registration Information Synchronization

5.4.2 Content ID Certificate

The following ASN.1 data type can be used to represent Content ID Certificate:

```
Certificate  ::=  SIGNED { ContentIDCertificate }


ContentIDCertificate ::= SEQUENCE {
        version                     Version DEFAULT v1,
        certificateSerialNumber     CertificateSerialNumber
        contentID                   ContentIdentifier,
        signature                           AlgorithmIdentifier{{SupportedAlgorithms}},
        issuer                              Name,
        validity                            Validity OPTIONAL,
```

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

```
        appType                                 AppType OPTIONAL,
        ContentFingerPrintInfo                  ContentFingerPrintInfo,
        extensions                              Extensions OPTIONAL
}

Version  ::=  INTEGER { v1(0)}
CertificateSerialNumber  ::=  INTEGER
ContentIdentifier  ::=  PrintableString

AppType  ::= INTEGER


AlgorithmIdentifier{ALGORITHM:SupportedAlgorithms}  ::=  SEQUENCE {
        algorithm                   ALGORITHM.&id ({SupportedAlgorithms}),
        parameters                  ALGORITHM.&Type ({SupportedAlgorithms}{ @algorithm}) OPTIONAL }
SupportedAlgorithms  ALGORITHM  ::=  { ... }

Validity  ::=  SEQUENCE {
        notBefore              Time,
        notAfter               Time }

ContentFingerPrintInfo  ::=  SEQUENCE {
        algorithm                   AlgorithmIdentifier{{SupportedAlgorithms}},
        ContentFingerPrint     BIT STRING }

Time  ::=  CHOICE {
        utcTime                     UTCTime,
        generalizedTime             GeneralizedTime }

Extensions ::= SEQUENCE OF Extension

Extension ::= SEQUENCE {
        metadata               PrintableString
}
```

version is the version of the encoded Content ID Certificate.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

serialNumber is an integer assigned by the CIM to each certificate. The value of **serialNumber** shall be unique for each certificate issued by a given CIM (i.e., the issuer name and serial number identify a unique certificate).

signature contains the algorithm identifier for the algorithm and hash function used by the CIM in signing the certificate (e.g., md5WithRSAEncryption, sha-1WithRSAEncryption, id-dsa-with-sha1, etc.)

issuer identifies the entity that has signed and issued the Content ID Certificate.

validity is the time interval during which the CIM warrants that it will maintain information about the status of the Content ID Certificate.

AppType is the type of the application scenario where the content should be managed,

contentFingerPrintInfo is used to carry the Content Fingerprint being certified and to identify the algorithm by which this Content Fingerprint is extracted.

Extension can contain the metadata needed for the AppType

6. SCIDM Operations

6.1 SCIDM Client Operations

The SCIDM Registration Client is out of the scope of the SCIDM Enabler. Only operations of the Identification Client are defined in this section.

6.1.1 Content Identification and Query

Upon receiving an identification request for a content item from the application, the Identification Client SHALL select an appropriate identification mechanism (Section 5.2), mainly according to the load of the Identification Client, application scenarios or the preset security level requirements. Then, the Identification Client SHALL collect the information needed for the selected mechanism. For information needed by each identification mechanism please refer to Section 5.2.

If the selected identification mechanism is ByFingerprint, the Identification Client MAY extract the Content Fingerprint from the content item. The Identification Client SHALL select the corresponding fingerprinting algorithm appropriate for different content types and different application scenarios. In different application scenarios, the service entities have different management purpose to the content. The appropriate algorithm for different content type and different application scenario SHOULD be preconfigured in the Identification Client. If no algorithm is preconfigured in the Identification Client, the Identification Client SHALL package the content itself in the ContentIdentRequest message (Section 7.1.2.1).

If the content contains multi-part, i.e., is compound, the Identification Client SHALL select appropriate identification mechanism for each member object individually and collect the information of each one.

According to the local policy, the Identification Client MAY first perform offline identification before online identification. Based on the selected mechanism and other relevant information, the Identification Client searches for matched registration information locally. The Identification Client SHOULD take sufficiently strong measure to prevent the locally stored registration information from being tampered. The integrity of the locally stored registration information SHOULD be verified based on the signature of the CIM from which the information is acquired.

The identification of each part of a Compound Content can be done separately. For example, the Identification Client MAY perform offline identification on some parts and online identification on other parts; it MAY also perform identification on some parts and ignore other parts.

If the Identification Client needs to contact the CIM for identification, it initiates the Online Identification and Query transaction [Section 7.1.2.1]. The Identification Client SHALL clearly understand if it wants the Content ID or the Content Metadata and set the parameter RequestType correspondingly in the ContentIdentRequest [Section 7.1.2.1.1]. This is specific to the application scenario and MAY be preconfigured.

The Identification Client MAY set the time limitation used to accomplish the identification task in the ContentIdentRequest [Section 7.1.2.1] according to the application scenario and pre-configuration.

According to the pre-configuration, The Identification Client MAY request the CIM to return the registration information of the content for later offline identification. The Identification Client sets the parameter RequestSync of the ContentIdentRequest as defined in Section 7.1.2.1.

Upon receiving the ContentIdentResponse from the CIM, the Identification Client SHALL validate the parameters of the message as defined in Section 7.

If the identification fails and recommendation of the identification mechanism to be used that has higher probability to succeed is presented by the CIM, the Identification Client SHOULD initiate another Online Identification and Query transaction based on the recommendation. If no recommendation is presented, the client SHOULD decide by itself whether to choose another mechanism according to its load, the accuracy of the failed mechanism and the pre-configuration.

If the CIM returns the registration information in the ContentIdentResponse, the Identification Client SHALL verify the information based on the signature of the CIM and store it in a secure region locally.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

6.1.2 Registration Information Synchronization

Based on the application specific policy, the Identification Client MAY need to initiate the synchronization process. The synchronization transaction is defined in Section 7.1.3.1. The Identification Client SHALL set the parameter Initiator to be the Identification Client and clearly inform the CIM what information it needs, as defined in Section 7.1.3.1.

Upon receiving the synchronization notification from the CIM, the Identification Client SHALL issue a request as defined in Section 7.1.3.2. The Identification Client SHALL set the parameter Initiator to be the server.

Upon receiving the returned registration information from the CIM, the Identification Client SHALL verify the information based on the signature of the CIM and store it in a secure region locally.

6.2 SCIDM Server Operations

6.2.1 Content Registration

Upon receiving the content registration request, the CIM SHALL first authenticate the Registration Client. Then the CIM SHALL extract the Content Fingerprint either directly from the request message or from the content embedded in the request message, and search locally to ensure that no duplicated registered content exists in the CIM. The fingerprinting algorithm SHALL be selected based on the registration purpose and the content type. The search SHOULD be based on the extracted Content Fingerprint. If authentication of the Registration Client is successful and no duplicated content is found, the CIM SHALL verify the trustworthiness of the submitted Content Metadata as described in [SCIDM-AD]. After that, CIM SHOULD verify if the content is applicable for registration within this CIM domain. If it is applicable, the CIM performs the registration process and allocates the content ID. How to verify the applicability is out of scope of this specification.

The CIM then SHALL allocate a globally unique ID for the content item. The ID format is defined in Section 7. The CIM SHOULD create a Content ID Certificate for the registered content. The Content ID Certificate is defined in Section 7.

If the content is compound, the CIM SHALL split it into individual objects and extract the Content Fingerprint of each object. The CIM SHOULD allocate one ID and create a certificate for the whole content and the certificate contains the Content Fingerprints of every member object.

Finally the CIM SHALL return a response to the Registration Client. If the registration succeeds, the CIM SHOULD return the Content ID Certificate to the Registration Client.

6.2.2 Online Identification and Query

Upon receiving a content identification and query request (Section 7.1.2.1) from the Identification Client, the CIM SHALL first authenticate the Identification Client. If the authentication succeeds, the CIM SHALL validate the parameters of the message as defined in Section 7.

If the identification and query request only specifies one identification mechanism, the CIM SHALL perform the identification based on the specified mechanism. If the request specifies more than one mechanisms, the CIM SHALL try the mechanisms from simple to complex (as described in Section 5.2). If the identification succeeds with one mechanism, then the CIM SHALL NOT try more complex mechanisms. If the identification fails by all the mechanisms and the ByFingerprint (Section 5.2) mechanism is not specified in the request, the CIM MAY recommend (Section 7.1.2.1.1) some mechanism that has higher probability to succeed in the response to the Identification Client.

If the identification and query request does not specify any identification mechanism, but only contain the content itself, the CIM SHALL use the ByFingerprint mechanism to perform the identification and select the appropriate fingerprinting algorithm according to the application type and content type presented by the request.

If the content to be identified is compound, the CIM SHALL split it into individual objects, and select a fingerprinting algorithm for each object based on its content type.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

If a matched content is found, the CIM SHALL return the Content ID to the Identification Client if the request type is Online Identification and return the requested Content Metadata if the request type is Online Query. If no matched content is found, the CIM SHALL respond to the Identification Client with a failure message with the corresponding error code defined in Section 11.

If matched content is found and the match is partial, the CIM SHOULD return the match information between the presented content from the request and the registered content in the CIM, which MAY include the position at which the piece is in the content, and the matched percentage (Section 7.1.2.1).

If the parameter RequestSync of the request is set (Section 7.1.2.1) and matched content is found, the CIM SHALL validate whether the registration information is allowed to be sent to the Identification Client which MAY be determined based on the policy of the CIM or the permission of the registrant which is out of the scope of the SCIDM Enabler. If allowed, the CIM SHALL populate it in the response (Section 7.1.2.1).

If the time limitation is set in the identification and query request, the CIM SHOULD return a response within the required time period. If no result is obtained in time, the CIM SHALL give up the identification since the Identification Client has treated the request as a failure.

6.2.3 Registration Information Synchronization

Based on the policy, the CIM MAY initiate the registration information synchronization. The CIM SHALL send a notification when such synchronization is needed and set the parameter SessionID in the notification (Section 7.1.3.1.1) which identifies what registration information the CIM would synchronize to the Identification Client.

Upon receiving a registration information synchronization request (Section 7.1.3.1) from the Identification Client, the CIM SHALL first authenticate the Identification Client.

If the authentication succeeds, the CIM SHALL determine whether the initiator of the request is the Identification Client or the CIM. If the Identification Client intiates the request, the CIM SHALL retrieve the registration information specified by the request (Section 7.1.3.1.1) and validate whether the registration information is allowed to be sent to the Identification Client. If yes, the CIM SHALL return them to the Identification Client with a response (Section 7.1.3.1.1).

If the initiator is the CIM, the CIM SHALL retrieve the registration information based on the parameter SessionID (Section 7.1.3.1.1) in the request and return them to the Identification Client.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

7. SCIDM Interfaces

7.1 Client-Server Interfaces

7.1.1 Interface SCIDM-1

7.1.1.1 Content Registration

The Content Registrant registers content to the CIM using the Content Registration transaction (see Figure 7: Content Registration message flow).

This section defines the details of the messages within the Content Registration transaction.

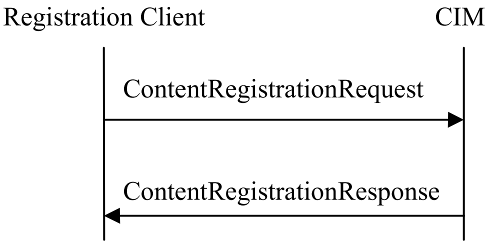


Figure 7: Content Registration message flow

7.1.1.1.1 Message and Information Elements

Message	Implementation	Direction
ContentRegistrationRequest	Mandatory	Registration Client → CIM
ContentRegistrationResponse	Mandatory	CIM →Registration Client

Table 1 Messages for Content Registration

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The Session-ID is unique within the service provider domain.
MessageID	Mandatory	String	Identifies this message. The Message-ID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. Transaction identifier offset is unique within the current session and identical for all messages within the transaction.
RegistrantID	Mandatory	String	Identifier of the Content Registrant. The identifier is allocated by CIM when the Content Registrant registers itself to the CIM, and the ID is unique within a CIM. How the Content Registrant registers itself to the CIM is implementation dependent and is out of the scope of the SCIDM Enabler. For example, it can be done via the web portal provided by the CIM.
RegistrantName	Optional	String	Name or short description of the Content Registrant
ContentRegInfo	Mandatory	List of Data Structure	One or more content registration information packages. The ContentRegInfo structure is defined in Table 3. The Content Registrant can register more than one content item, so there can by more than one instances of ContentRegInfo.

Table 2 Information elements in the ContentRegistrationRequest message

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Information Element	Req	Type	Description
ContentRegInfoID	Mandatory	String	The ID of this instance of ContentRegInfo.
ContentName	Optional	String	The Name of the content. It is not assigned by the CIM but assigned by and used by the Content Registrant for such purposes as content management
RegistrationPurpose	Optional	String	The purpose of the registration. The value of RegistrationPurpose is identical to the parameter ApplicationType in Table 17.
Content	Optional	Opaque Data	Opaque content that is sent to the CIM for registration. This is optional because the Content Registration can also only send the Content Fingerprint to the CIM.
ContentFingerpint	Optional	Opaque Data	Opaque Content Fingerprint that is sent to the CIM for registration. If the ContentInfo structure do not contain the content itself, then it is mandatory to contain this parameter.
ContentFingerprintAlgID	Optional	String	The ID of the algorithm that is used to extract the Content Fingerprint in the parameter ContentFingerprint. The name SHOULD be understandable to the CIM. If the parameter ContentFingerprint is present here, this parameter is mandatory. For the definition of the algorithm ID, please see Section 9.1.
ContentMetadata	Mandatory	Structure	Content metadata that is used to describe the content and used as the base for content management for which the Content Registrant registers the content. What metadata should be contained in this structure should be based on the RegistrationPurpose but this is out of scope. The format of this structure adopts the MPEG7 standard and is described in details in Section 8.

Table 3 Information elements in the ContentRegInfo structure

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is identical to the SessionID in ContentRegistrationRequest.
MesssageID	Mandatory	String	Identifier of a message. The MessageID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. Transaction identifier offset is unique within the current session and identical for all messages within the transaction.
RegistrationResult	Mandatory	List of Data Structure	The result of the registration in CIM. It contains the code, ID of the content and optionally the certificate of the registered content. The RegistrationResult structure is defined in Table 5.

Table 4 Information elements in the ContentRegistrationResponse message

Information Element	Req	Type	Description
ResultID	Mandatory	String	The ID of this RegistrationResult. The ResultID value MUST be identical to the corresponding ContentRegInfoID value.
Code	Mandatory	String	The result code of the registration in CIM. The codes are defined in Section 11.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

ContentID	Mandatory	String	The globally-unique ID of the registered content allocated by the CIM. About the format of the ID please see Section 8.1.
Certificate	Optional	Opaque Data	The Content ID Certificate created by the CIM after the registration, contains the Content Fingerprint, some of its metadata, and the signature of the CIM. About the format of the Content ID Certificate, please see Section 5.4.2.

Table 5 Information elements in the RegistrationResult structure

7.1.1.2 Content Update

The Content Registrant updates content registration information to the CIM using the Content Update transaction (see Figure 8: Content Update message flow).

This section provides the details of the transaction messages.

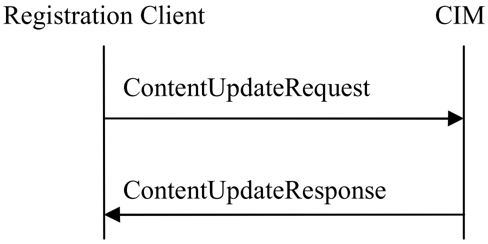


Figure 8: Content Update message flow

7.1.1.2.1 Message and Information Elements

Message	Implementation	Direction
ContentUpdateRequest	Mandatory	Registration Client → CIM
ContentUpdateResponse	Mandatory	CIM →Registration Client

Table 6 Messages for Content Update

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is unique within the service provider domain.
MessageID	Mandatory	String	Message identifier. The MessageID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. The transaction identifier offset is unique within the session and identical for all messages within the transaction.
RegistrantID	Mandatory	String	Identifier of the content registrant. The identifier is allocated by CIM when the Content Registrant registers itself to the CIM, and the ID is unique within a CIM. How the Content Registrant registers itself to the CIM is implementation dependent and is out of the scope of the SCIDM Enabler. For example, it can be done via the web portal provided by the CIM.
RegistrantName	Optional	String	Name or short description of the content registrant
ContentUpdateInfo	Mandatory	List of Data Structure	One or more content registration information packages. The Content Registrant can update more than one content item, so there can be more than one instances of ContentUpdateInfo. This structure is defined in Table 8.

Table 7 Information elements in ContentUpdateRequest message

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Information Element	Req	Type	Description
ContentUpdateInfoID	Mandatory	String	The ID of an instance of ContentUpdateInfo.
ContentID	Mandatory	String	The globally-unique Content ID allocated by the CIM.
RegistrationPurpose	Optional	String	The purpose of the registration. This element is optional, and only when the purpose of the content registration is changed, this element is populated. The value of Registration Purpose is identical to the parameter ApplicationType in Table 17.
ContentFingerpint	Optional	Opaque Data	Opaque Content Fingerprint that is sent to the CIM for update. Only when content registrant uses a different fingerprinting algorithm, this element needs to be populated to replace the old value by the new Content Fingerprint value.
ContentFingerprintAlgID	Optional	String	The name of the algorithm that is used to extract the new Content Fingerprint in the parameter ContentFingerprint. The name SHOULD be understandable to the CIM. If the parameter ContentFingerprint is present here, this parameter is mandatory. For the definition of the algorithm ID, please see Section 9.1.
ContentMetadata	Optional	Structure	Contain the Content Metadata that needs to be updated. What metadata should be contained in this structure is out of scope. The format of this structure adopts the MPEG7 standard and is described in detail in Section 8.

Table 8 Information elements in the ContentUpdateInfo structure

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is identical to the SessionID in ContentUpdateRequest.
MesssageID	Mandatory	String	Message identifier. The Message-ID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. The transaction identifier offset is unique within the session and identical for all messages within the transaction.
UpdateResult	Mandatory	List of Data Structure	The result of the registration in CIM. It contains the code and optionally the Content ID Certificate of the registered content. The UpdateResult structure is defined in Table 10.

Table 9 Information elements in the ContentUpdateResponse message

Information Element	Req	Type	Description
ResultID	Mandatory	String	The ID of this UpdateResult. The ID value MUST be identical to the corresponding ContentUpdateInfoID value.
Code	Mandatory	String	The result code of the content update in CIM. The codes are defined in Section 11.
Certificate	Optional	Opaque Data	The Content ID Certificate created by the CIM after the registration. It contains the Content Fingerprint, some of its metadata, and the signature of the CIM. About the format of the Content ID Certificate, please see Section 5.4.2.

Table 10 Information elements in the UpdateResult structure

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

7.1.1.3 Content Deregistration

The Content Registrant de-registers the content information to the CIM using the Content Deregistration transaction (see Figure 9: Content Deregistration message flow).

This section provides the details of the transaction messages.

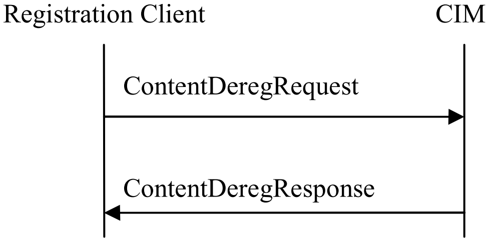


Figure 9: Content Deregistration message flow

7.1.1.3.1 Message and Information Elements

Message	Implementation	Direction
ContentDeregRequest	Mandatory	Registration Client → CIM
ContentDeregResponse	Mandatory	CIM →Registration Client

Table 11 Message for Content De-registration

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is unique within the service provider domain.
MessageID	Mandatory	String	Message identifier. The MessageID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. The transaction identifier offset is unique within the session and identical for all messages within the transaction.
RegistrantID	Mandatory	String	Identifier of the content registrant. The identifier is allocated by CIM when the Content Registrant registers itself to the CIM, and the ID is unique within a CIM. How the Content Registrant registers itself to the CIM is implementation dependent and is out of the scope of the SCIDM Enabler. For example, it can be done via the web portal provided by the CIM.
RegistrantName	Optional	String	Name or short description of the content registrant
ContentDeregInfo	Mandatory	List of Data Structure	One or more content deregistration information packages. The ContentDeregInfo structure is defined in Table 13.

Table 12 Information elements in ContentDeregRequest message

Information Element	Req	Type	Description
ContentDeregInfoID	Mandatory	String	The ID of this instance of ContentDeregInfo.
ContentID	Mandatory	String	The globally-unique Content ID allocated by the CIM.
Description	Optional	String	The reason for deregistration.

Table 13 Information elements in ContentDeregInfo structure

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is identical to the SessionID in ContentDeregRequest.
MesssageID	Mandatory	String	Message identifier. The Message-ID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. The transaction identifier offset is unique within the session and identical for all messages within the transaction.
DeregResult	Mandatory	List of Data Structure	The result of the deregistration in CIM. It contains the code. The DeregResult structure is defined in Table 15.

Table 14 Information elements in ContentDeregResponse message

Information Element	Req	Type	Description
ResultID	Mandatory	String	The ID of this DeregResult. The ID value MUST be identical to the corresponding ContentDeregInfo ID .
Code	Mandatory	String	The result code of the deregistration in CIM. The codes are defined in Section 11.

Table 15 Information elements in the DeregResult structure

7.1.2 Interface SCIDM-2

7.1.2.1 Online Identification and Query

The Identification Client contacts the CIM for content identification and query using the Online Identification and Query transaction (see Figure 10: Online Identification and Query message flow).

This section defines the details of the transaction messages.

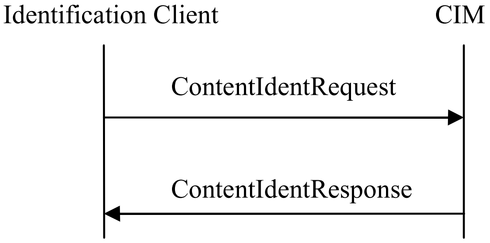


Figure 10: Online Identification and Query message flow

7.1.2.1.1 Message and Information Elements

Message	Implementation	Direction
ContentIdentRequest	Mandatory	Identification Client → CIM
ContentIdentResponse	Mandatory	CIM →Identification Client

Table 16 Messages for Online Identification and Query

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The Session-ID is unique within the service provider domain.
MessageID	Mandatory	String	Identifies this message. The Message-ID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. Transaction identifier offset is unique within a session and identical for all messages within the transaction.
ClientID	Mandatory	String	Identifier of the Identification Client. The identifier is allocated by the CIM when the Identification Client registers itself to the CIM, and the ID is unique within a CIM. How the Identification Client registers itself to the CIM is implementation dependent and is out of the scope of the SCIDM Enabler. For example, it can be done via the web portal provided by the CIM.
ClientName	Optional	String	Name or short description of the Identification Client.
RequestType	Optional	Integer	The type of the request. There are two types for the ContentIdentRequest, being 0 for Online Identification request, and 1 for Online Query request. For the description of the two types, please see Section 5.3. If this element is not present, the CIM treats the request as Online Identification request by default.
ApplicationType	Optional	String	The type of the application for which the identification is done. If the request type is Online Identification, this element SHALL NOT be populated. Otherwise, it SHOULD be populated for the CIM to choose what metadata will be returned to the Identification Client after successful identification.
ContentIdentInfo	Mandatory	List of Data Structure	One or more content identification information packages. Each package contains information for identification of one content item. If a Compound Content is to be identified, information of each member object is packaged in a ContentIdentInfo structure. The Compound Content also can be treated as a single content item whose information is packaged in one ContentIdentInfo. The ContentIdentInfo structure is defined in Table 18.
RequestSync	Optional	Boolean	When presented, this element indicates that the Identification Client requests the CIM to return the registration information of the content to be identified after successful identification.
TimeLimit	Optional	Integer	Time limitation that the Identification Client requests the CIM to accomplish the identification. It is expressed as a number in seconds, such as 30 for 0.5 minutes.

Table 17 Information elements in the ContentIdentRequest message

Information Element	Req	Type	Description
ContentIdentInfoID	Mandatory	String	The ID of this instance of ContentIdentInfo.
IDInfo	Optional	Structure	The identification information for the ByID mechanism. When the Identification Client already knows the content item’s globally unique content ID allocated by the CIM, it can use this structure to carry the Content ID information for Online Query operation, that is, to acquire some Content Metadata based on the Content ID.
HistoryInfo	Optional	Structure	The identification information for the ByHistory mechanism.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

MetadataInfo	Optional	Structure	The identification information for the ByMetadata mechanism.
FingerprintInfo	Optional	Structure	The identification information for the ByWatermark or ByFingerprint mechanism.

Table 18 Information elements in the ContentIdentInfo structure

Information Element	Req	Type	Description
ID	Optional	String	The globally unique Content ID allocated by the CIM. Only used when the Identification Client performs the Online Query operation based on the ID.
Name	Optional	String	The content name, or the ID used in an application (not the Content ID allocated by the CIM). SHALL be presented when the Identification Client wants to do the Online Identification or Online Query via the ByID mechanism.
Size	Optional	Integer	The size of the content. It is expressed as a number in bytes. This is auxiliary information of the element Name.
Ext	Optional	N/A	This is for extension usage. More auxiliary information can be presented inside this tag.

Table 19 Information elements in the IDInfo structure

Information Element	Req	Type	Description
SourceIP	Optional	String	The source IP address of the content delivery in which the content to be identified is being delivered. The Identification Client SHOULD try to get the information and set it in this tag.
SourceDomain	Optional	String	The source domain name of the content delivery in which the content to be identified is being delivered. The Identification Client SHOULD try to get the information and set it in this tag.
DestIP	Optional	String	The destination IP address of the content delivery in which the content to be identified is being delivered. The Identification Client SHOULD try to get the information and set it in this tag.
DestDomain	Optional	String	The destination domain name of the content delivery in which the content to be identified is being delivered. The Identification Client SHOULD try to get the information and set it in this tag.
SubscriberID	Optional	String	The ID of the subscriber who is the initiator of the content delivery. The Identification Client SHOULD try to get the information and set it in this tag.

Table 20 Information elements in the HistoryInfo structure

Information Element	Req	Type	Description
Digest	Mandatory	String	The digest value (e.g., MD5, SHA) of the content item.
DigestAlgID	Mandatory	String	The ID of the digest algorithm in use.
Ext	Optional	N/A	This is for extension usage. More metadata can be presented inside this tag.

Table 21 Information elements in the MetadataInfo structure

Information Element	Req	Type	Description
ContentFingerprint AlgID	Optional	String	ID of the content fingerprinting algorithm. This element MUST be populated if the Identification Client extracts the Content Fingerprint and send it to the CIM. Otherwise, if the Identification Client sends the content itself to the CIM, this parameter SHALL NOT be populated. For the definition of the

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

			algorithm ID, please see Section 9.1.
ContentFingerprint	Optional	Opaque Data	Opaque Content Fingerprint which is extracted by the Identification Client using the algorithm identified by the element ContentFingerprintAlgID. This element MUST be populated if the Identification Client extracts the Content Fingerprint and sends it to the CIM.
Content	Optional	Opaque Data	Opaque content that is sent to the CIM. This element MUST be populated If the Identification Client sends the content itself to the CIM.

Table 22 Information elements in the FingerprintInfo structure

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is identical to the SessionID in ContentIdentRequest.
MesssageID	Mandatory	String	Identifier of a message. The MessageID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. Transaction identifier offset is unique within a session and identical for all messages within the transaction.
IdentResult	Mandatory	List of Data Structure	The result of the content identification and query request. Each IdentResult corresponds to a ContentIdentInfo structure in the ContentIdentRequest. The IdentResult structure is defined in Table 24.

Table 23 Information elements in the ContentIdentResponse message

Information Element	Req	Type	Description
ContentIdentInfoID	Mandatory	String	This value SHALL be identical to the corresponding ContentIdentInfoID in the request message.
Code	Mandatory	String	The result code of the identification and query. The codes are defined in Section 11.
ContentID	Optional	String	The globally unique Content ID of the identified content. If the ContentIdentRequest type is Online Identification, this element MUST be populated.
MatchInfo	Optional	Structure	The match information between the submitted content from the Identification Client and the content in the CIM. If the match is partial, this structure SHOULD be populated. For the detailed definition of this structure, please see Table 25.
ContentMetadata	Optional	Structure	The Content Metadata returned from the CIM to the Identification Client after successful identification and query if the request type is Online Query. If the request type is Online Identification, this element SHALL NOT be populated. For the definition of the format of metadata, please see Section 8.
RegistrationInfo	Optional	Opaque Data	The complete registration information of the identified content. If the element RequestSync is set to 1, the CIM SHOULD populate this element with the registration information, which SHOULD be in the form of Content ID Certificate.

Table 24 Information elements in the IdentResult structure

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Information Element	Req	Type	Description
MatchPosition	Optional	Integer	The position of the submitted content piece in the whole content. It is defined as number of bytes apart from the beginning of the whole content.
MatchPercent	Optional	Integer	The matched percentage. It is defined as a number in percentage, such as 30 for 30% match.
MatchType	Optional	Integer	The type ID of the match. The type definition is: 00 exact match 01 different version 02 partial match 03 clips of original content 04 edit of multiple content

Table 25 Information elements in the MatchInfo structure

7.1.3 Interface SCIDM-3

7.1.3.1 Client-initiated Registration Information Synchronization

The Identification Client makes request to the CIM for registration information synchronization using the Client-initiated Registration Information Synchronization transaction (see Figure 11: Client-initiated Registration Information Synchronization message flow).

This section defines the details of the transaction messages.

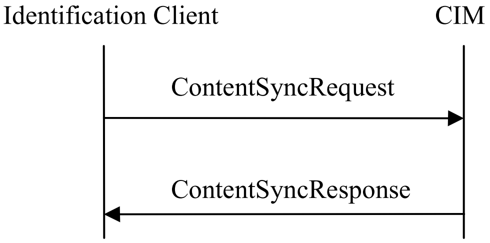


Figure 11: Client-initiated Registration Information Synchronization message flow

7.1.3.1.1 Message and Information Elements

Message	Implementation	Direction
ContentSyncRequest	Mandatory	Identification Client → CIM
ContentSyncResponse	Mandatory	CIM →Identification Client

Table 26 Messages for Client-initiated Registration Information Synchronization

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The Session ID is unique within the service provider domain.
MessageID	Mandatory	String	Identifies this message. The Message ID is unique within a session. Message identifier consists of the transaction identifier suffixed by

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

			two numeric characters for message index within the transaction. Transaction identifier offset is unique within a session and identical for all messages within the transaction.
ClientID	Mandatory	String	Identifier of the Identification Client. The identifier is allocated by CIM when the Identification Client registers itself to the CIM, and the ID is unique within a CIM. How the Identification Client registers itself to the CIM is implementation dependent and is out of the scope of the SCIDM Enabler. For example, it can be done via the web portal provided by the CIM.
ClientName	Optional	String	Name or short description of the Identification Client.
Initiator	Mandatory	Boolean	To indicate if the request is initiated by the Identification Client (the value of this element is 0) or by the CIM (the value is 1).
ContentIDList	Optional	List of Data Structure	List of Content IDs identifying which registration information the Identification Client intends to synchronize. This is optional and used when the Identification Client knows which registration information it wants to synchronize. The structure is defined in Table 28.
SyncRule	Optional	Data Structure	To hold the rule of synchronization, i.e., what registration information the Identification Client intends to synchronize. This is used when the Identification Client does not know what should be synchronized. The structure is defined in Table 29.

Table 27 Information elements in the ContentSyncSRequest message

Information Element	Req	Type	Description
ContentID	Mandatory	String	The globally unique ID of a content item whose registration information the Identification Client intends to synchronize.

Table 28 Information elements in the ContentIDList structure

Information Element	Req	Type	Description
Start	Optional	DateTime	The start date and time of the window of time within which all Registration Client registrations that were made with the CIM are requested by the Identification Client.
End	Optional	DateTime	The end date and time of the window of time within which all Registration Client registrations that were made with the CIM are requested by the Identification Client.
ApplicationIDList	Optional	List of Data Structure	The List of IDs of applications for which the content is registered. By setting this element, the Identification Client shows it intends to synchronize the registration information of the content registered for the specified application.
RegistrantIDList	Optional	List of Data Structure	The List of IDs of registrants who register content to the CIM. By setting this element, the client shows it intends to synchronize the registration information of the content that the specified registrants registered.
ContentTypeList	Optional	List of Data Structure	The List of content types. By setting this element, the client shows it intends to synchronize the registration information of the content of the specified types.

Table 29 Information elements in the SyncRule structure

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Information Element	Req	Type	Description
ApplicationID	Mandatory	String	The application ID.

Table 30 Information elements in the ApplicationIDList structure

Information Element	Req	Type	Description
RegistrantID	Mandatory	String	The registrant ID.

Table 31 Information elements in the RegistrantIDList structure

Information Element	Req	Type	Description
ContentType	Mandatory	String	The content type.

Table 32 Information elements in the ContentTypeList structure

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The SessionID is identical to the SessionID in ContentSyncSRequest.
MesssageID	Mandatory	String	Identifier of a message. The MessageID is unique within a session. Message identifier consists of the transaction identifier suffixed by two numeric characters for message index within the transaction. Transaction identifier offset is unique within the current session and identical for all messages within the transaction.
Code	Mandatory	String	The result code. The codes are defined in Section 11.
RegInfoList	Optional	List of Data Structure	List of registration information.

Table 33 Information elements in the ContentSyncResponse message

Information Element	Req	Type	Description
RegistrationInfo	Mandatory	Opaque Data	One instance of registration information which SHOULD be in the form of Content ID Certificate.

Table 34 Information elements in the RegInfoList structure

7.1.3.2 Server-initiated Registration Information Synchronization

The CIM initiates the registration information synchronization by the Server-initiated Registration Information Synchronization transaction (see Figure 12: Server-initiated Registration Information Synchronization message flow).

For the definition of ContentSyncRequest and ContentSyncResponse, please see Section 7.1.3.1. This section defines the details of the message ContentSyncNotification.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

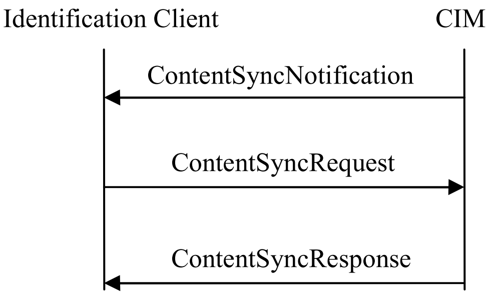


Figure 12: Server-initiated Registration Information Synchronization message flow

7.1.3.2.1 Message and Information Elements

Message	Implementation	Direction
ContentSyncNotification	Mandatory	CIM →Identification Client
ContentSyncRequest	Mandatory	Identification Client → CIM
ContentSyncResponse	Mandatory	CIM →Identification Client

Table 35 Messages for Server-initiated Registration Information Synchronization

Information Element	Req	Type	Description
SessionID	Mandatory	String	Session identifier. The Session ID is unique within the service provider domain.
ServerID	Mandatory	String	The ID of the CIM used by the Identification Client
Informative	Mandatory	Integer	Used to tell the Identification Client the purpose of this notification message. For notification of registration information synchronization, the value is defined as 00. Other value code can be defined based on the requirement.
Ext	Optional	N/A	For future use or vendor specific use.

Table 36 Information elements in the ContentSyncNotification message

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

8. SCIDM Metadata

This section defines the metadata used in the SCIDM Enabler.

8.1 The SCIDM Content ID

The SCIDM Content ID uniquely identifies the content globally and is allocated by the CIM when a content item is registered to the CIM. In the SCIDM Enabler, the format of the Content ID adopts the URI scheme of email address style as follow:

Local_Content_ID@CIM_ID

The Local_Content_ID is the Content ID allocated by the CIM according to the local ID scheme. The CIM_ID is the CIM's ID in the global scope and normally is the domain name or the IP address of the CIM.

8.2 Application Specific Metadata

The application specific metadata includes the content information used by the application for the purpose of content management. In the SCIDM Interfaces, the application specific metadata is conveyed by the structure *ContentMetadata* in the Content Registration transaction, Content Update transaction and Content Identification and Query transaction (Section 7).

Since MPEG-7 [ISO-MPEG-7] defines the technologies to that describe the content metadata for the digital items, it is RECOMMENDED that the SCIDM Enabler uses the standard metadata schemes of MPEG-7 as the description language of the SCIDM Content Metadata.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

9. SCIDM Content Fingerprint

The SCIDM Enabler supports the use of various content fingerprinting algorithms.

There are two classes of Content Fingerprints: fragile and robust. Fragile Content Fingerprints are extremely sensitive to bit errors in the content. Examples include cryptographic hash values of a content item. Robust Content Fingerprints can tolerate some distortion to the original content, typically introduced as a result of common content signal processing such as compression, filtering etc.

9.1 Fingerprinting Algorithm Identifier

A fingerprinting algorithm identifier is introduced in the SCIDM Enabler to enable the use of various fingerprinting algorithms. The fingerprinting algorithm ID identifies the particular fingerprinting algorithm used to generate the Content Fingerprint.

A recent standardized robust fingerprinting algorithm for image is the MPEG-7 Image Signature Tools [MPEG-7-IMG-SIG].

Examples of the fingerprinting algorithm identifier are: MD5, SHA-1, SHA-256, MPEG7-IMG-SIG.

9.2 Fingerprint Container File Format

This section introduces a container file format, called XFP for "eXtensible FingerPrint", to contain Content Fingerprint that is an important element in SCIDM. The detailed syntax specification of this file format is provided in Appendix B.

9.2.1 Introduction

SCIDM is designed to cover various types of media (e.g., audio, video, still images) and allows the use of different fingerprinting algorithms. Because of such broad coverage and diversity in content types and fingerprinting algorithms, it is essential to have a uniform file format to facilitate interchange and management of Content Fingerprint data of different types. The XFP file format is a generic and extensible container that is designed to contain various types of Content Fingerprint as an opaque binary object. The file format can also contain "instance metadata" that may include attributes of and information about the copy of the content from which the Content Fingerprint is computed.

The structure of the XFP File Format is similar to that of the ISO Base Media File Format [ISO-BMFF] that is designed to contain multimedia content, particularly video and audio, along with related metadata. An XFP file consists of a series of objects called "boxes". All boxes use a Key-Length-Value (KLV) data structure, that is, it starts with Key identifying the type of the box, followed by Length specifying the size of the box in bytes, and then followed by Value that is the "payload" or data contained in the box. A box can be nested, which means a box can be the container of another box or be contained in another box. The size specified by Length is the entire size of the box, including the Key and Length fields and the payload data and all contained boxes. The object-based XFP file structure allows any opaque data object to be contained in an XFP file, and the file can be parsed without specific knowledge about the contained data object.

Unlike the ISO Base Media File Format, an XFP file is a standalone Content Fingerprint container. It does not contain the media content from which the Content Fingerprint is generated. In many SCIDM applications, Content Fingerprint is exchanged and processed (e.g., registration or query) without the content. The XFP File Format is designed specifically for containing Content Fingerprint without the content. For applications where it is appropriate and feasible to store Content Fingerprint along with the content, a different file format should be used. For example, it is possible to use the ISO Base Media File Format to store Content Fingerprint in a time-based metadata track along with the corresponding audio or video track. Considerations are taken in designing the XFP File Format such that a video or audio Content Fingerprint contained in an XFP file can be transcribed to a time-based metadata track that is compliant to the ISO Base Media File Format.

9.2.2 Overview of the XFP File Structure

At the highest level, an XFP file begins with an XFP Start Code followed by an XFP CRC Box, an XFP Meta Box and an XFP Data Box, as illustrated in Figure 13. The XFP CRC Box contains CRC codes for data integrity check. The XFP Meta

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Box contains mandatory boxes that describe the structure of Content Fingerprint data, and optional boxes that describe the attributes of the source file and media data. The XFP Data Box contains data of layered Content Fingerprint streams. The hierarchy of the boxes that can be contained in the XFP Meta Box and XFP Data Box is shown in Figure 13. Mandatory and optional boxes are labelled by (M) and (O), respectively. These boxes are specified in detail in Appendix B.

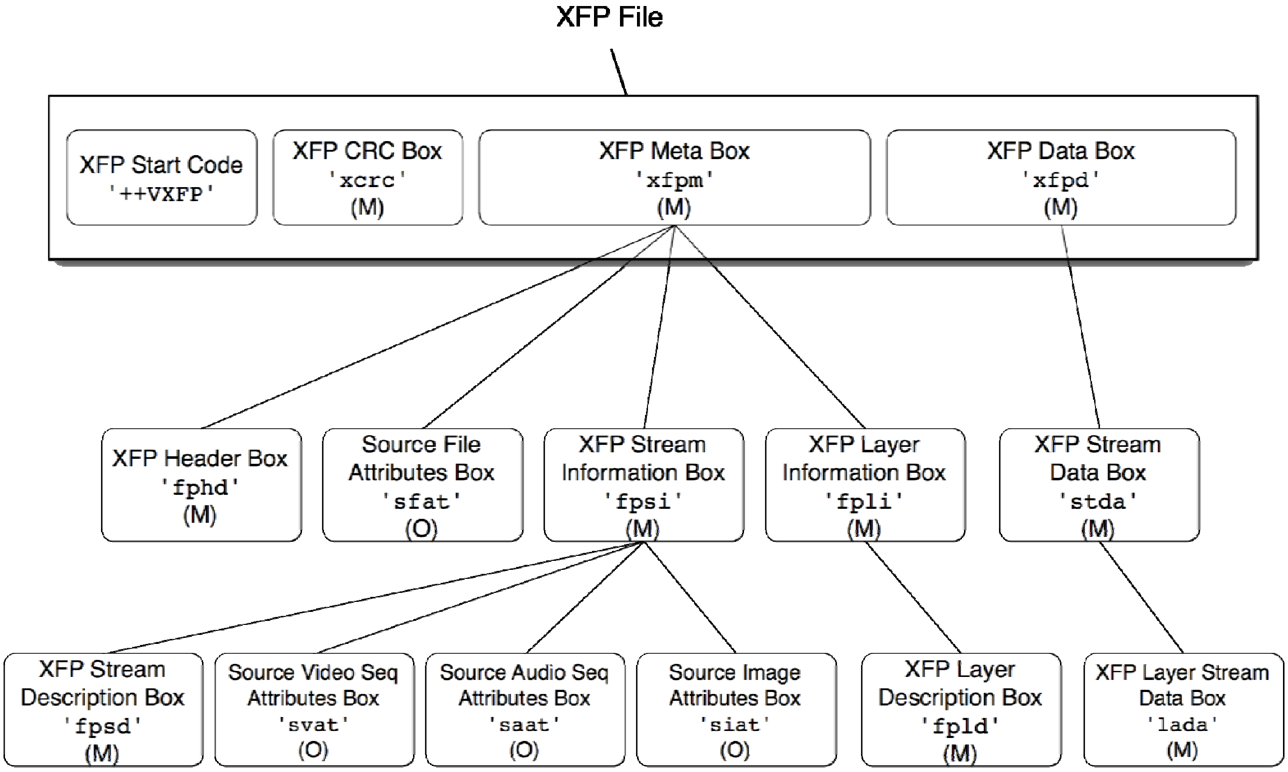

XFP File
XFP Start Code '++VXFP'
XFP CRC Box 'xcrc' (M)
XFP Meta Box 'xfpm' (M)
XFP Data Box 'xfpd' (M)
XFP Header Box 'fphd' (M)
Source File Attributes Box 'sfat' (O)
XFP Stream Information Box 'fpsi' (M)
XFP Layer Information Box 'fpli' (M)
XFP Stream Data Box 'stda' (M)
XFP Stream Description Box 'fpsd' (M)
Source Video Seq Attributes Box 'svat' (O)
Source Audio Seq Attributes Box 'saat' (O)
Source Image Attributes Box 'siat' (O)
XFP Layer Description Box 'fpld' (M)
XFP Layer Stream Data Box 'lada' (M)

Figure 13: XFP File Structure

An important design consideration for the XFP File Format is to support layering of XFP streams. Conceptually, Content Fingerprint or XFP data may be divided by content units and further grouped by layers. For example, video fingerprint or VFP data may be divided into frame fingerprints such that each frame fingerprint describes a corresponding video frame. A frame fingerprint may be further divided into layers such that each layer contains different types of video signature data (e.g., spatial or temporal).

An XFP stream that contains only a specific layer of the Content Fingerprint is called a layered XFP substream. Support for layered XFP substreams in the XFP File Format provides flexibility in grouping and packaging XFP streams, and extensibility for including additional fingerprint data in the future. The XFP Layer Information Box and XFP Stream Information Box are designed to describe the vertical (layer) and horizontal (stream) structures of XFP data.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

10. SCIDM Security

10.1 Authentication

In the interface SCIDM-1, SCIDM-2 and SCIDM-3, the SCIDM Client acts as a subscriber agent for authentication purposes, and the SCIDM Server (CIM) acts as an agent for SCIDM Service Providers for authentication purposes. This section refers to authentication of the SCIDM Client and the CIM acting in those roles.

Authentication ensures that the SCIDM Client and the CIM can trust each other. The necessity of establishing this trust relationship may vary for different SCIDM Service Providers or SCIDM Clients. This necessity is assumed to be known through pre-configuration, or through discovery mechanisms that are unspecified.

The SCIDM Client SHALL supply authentication when the SCIDM Client sends request to the CIM via SCIDM-1 , according to the security policy of the CIM. The SCIDM Client SHOULD supply authentication when the SCIDM Client sends request to the CIM via SCIDM-2 and SCIDM-3, according to the security policy of the CIM. The CIM SHALL supply authentication according to the security policy of the SCIDM Client when returning the response to the SCIDM Client. The SCIDM Client and SCIDM Server SHALL accept valid authentication, when supplied per their security policy.

Three modes of the SCIDM Client authentication are specified for use by the SCIDM Enabler:

- The SCIDM Client authentication provided by functions external to the SCIDM Enabler. The CIM SHALL support this method of SCIDM Client authentication.

- The SCIDM Client authentication via HTTP Digest Authentication [RFC2617]. The CIM SHALL support this method of SCIDM Client authentication. The SCIDM Client SHALL support this method of SCIDM Client authentication.

- The SCIDM Client authentication via a lower-layer security facility, e.g. TLS-based delivery of a client certificate. The CIM SHOULD support this method of the SCIDM Client authentication. The SCIDM Client SHOULD support this method of SCIDM Client authentication.

If The SCIDM Client authentication is not recognized or otherwise not accepted by the CIM, the CIM SHALL provide an “authentication error” response to the SCIDM Client. The SCIDM Client behavior in this case (e.g. request retry) is out of the scope of the enabler.

One mode of the CIM authentication is specified for use by the SCIDM Enabler:

- The CIM credentials-based authentication via a lower-layer security facility, e.g. delivery of a server certificate. The CIM SHALL support this method of CIM authentication. The SCIDM Client SHOULD support this method of CIM authentication.

If the CIM authentication is not recognized or otherwise not accepted by a SCIDM Client (e.g. the server certificate has expired or cannot be validated), the SCIDM Client SHALL abandon the response message from the CIM.

In addition, unauthenticated service may be provided if allowed per the security policy of the CIM and/or the SCIDM Client. The CIM SHALL provide service without the SCIDM Client authentication, if allowed according to SCIDM Service Provider security policy. The SCIDM Client SHALL trust the message from the CIM without CIM authentication, if allowed per the SCIDM Client security policy.

10.1.1 The SCIDM Client Authentication Provided by Functions External to the SCIDM Enabler

In this mode, The SCIDM Client authentication is provided by non-SCIDM network elements, e.g. as operated by the SCIDM Service Provider. This is a typical approach to client authentication in cases that client-provided authentication is not available or not trusted. The details are unspecified, but may include such options as:

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

- Proxy-inserted subscriber identification, e.g.
    - The SCIDM Client is configured to use a proxy for the SCIDM interfaces, and sends requests via the proxy instead of directly to the CIM.
    - The proxy inserts a subscriber identity indication (e.g. HTTP header) based upon network information into the request, and forwards it to the CIM. The subscriber identity may be in any form supported/permitted to be inserted by the network proxy for delivery to all or specific destinations.

10.1.2 The SCIDM Client Authentication via HTTP Digest Authentication

In this mode, the SCIDM Client authentication is provided by the SCIDM Client from pre-configured or subscriber-provided credentials. The mechanisms for pre-configuration or subscriber entry of credentials are unspecified.

HTTP Digest Authentication [RFC2617] is used to deliver the credentials to the CIM:

1. The SCIDM Client sends a request message to the CIM. If the SCIDM Client supports Digest Authentication per the GBA Profile of [SEC_CF], it discloses it [SEC_CF].
2. The CIM indicates the need for SCIDM Client authentication via Digest Authentication by responding to the first request message with a digest access authentication challenge, including a generated nonce value. If the CIM supports Digest Authentication per the GBA Profile of [SEC_CF], it authenticates the Client using GBA-based shared secret as described in [SEC_CF], per the policy established by the SCIDM Service Provider.
3. The SCIDM Client sends a challenge response with its credentials, hashed per the supplied nonce value. Note the credentials may be derived from a variety of sources, e.g. user-provided/stored or provisioned “username:password”, or 3GPP GBA-provided credentials as described for the GBA Profile of [SEC_CF].
4. The CIM retrieves the credentials from the hashed digest-response value.

10.1.3 The SCIDM Client and the CIM Authentication via Lower-layer Security Facilities

In this mode, the SCIDM Client and/or CIM authentication is provided by facilities of an underlying security layer, e.g. TLS-provided client or server certificates. The methods of certificate management and notification of error conditions to the SCIDM Client or the CIM are assumed to be addressed by the underlying transport security, and are unspecified.

10.2 Connection Security

Connection security ensures that the SCIDM Client and the CIM can trust the privacy and integrity of the operations and data exchanged with the other entity. The necessity of connection security during session establishment may vary for different SCIDM Service Providers or SCIDM Clients. This necessity is assumed to be known through preconfiguration, or through discovery mechanisms that are unspecified.

The SCIDM Client SHALL provide connection security, per the security policy of the CIM. The CIM SHALL provide connection security, per the security policy of the SCIDM Client.

One mode of connection security for session establishment is specified for use by the SCIDM Enabler:

Use of transport layer security provided by TLS [RFC4346] or DTLS [RFC4347]. The CIM and the SCIDM Client SHALL support TLS or DTLS to provide connection security for the operations and data of the three interfaces, as applicable to the underlying transport protocol in use.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

11. Error Handling

If a transaction is not successfully completed, an error notification is sent to the requesting entity.

Only the first detected error is indicated in the error notification, and additional errors are not reflected in the error notification.

11.1 Error Conditions

The tables in the following sections summarize the interpretation and handling of error conditions based upon information in the SCIDM transactions. Normative requirements describing the expected SCIDM entity behaviour follow the tables.

The table columns represent:

- Error condition: the type of error that has been detected
- Detected by: the SCIDM entity detecting the error
- Response: the proper response to the error. If more than one entity responses are shown in the table, the first one is the expected reaction of the error-detecting entity, and the second is the expected reaction of another entity when it receives notice of the error from the error-detecting entity. Examples of responses:
    - Fail: terminate the handling of the message, and respond to the requesting entity with the error code.
    - Retry: the action should be retried as indicated.
    - Discard: the error message/element should be discarded with no response to the requesting entity.

Description: scenario in which the error may be detected

11.1.1 Generic Errors

This section addresses errors that may occur in a variety of situations.

Error Condition	Detected by	Response	Description
Malformed message	server, client	Discard message, log error	Fail to parse the message
Invalid parameter	server	server: fail client: retry with correct parameter	Message with invalid parameter
	client	client: fail server: retry with correct parameter	Message with invalid parameter
Authentication error	server	server: fail client: retry with correct authentication	Invalid authentication
	client	client: fail server: retry with correct authentication	Invalid authentication

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Error Condition	Detected by	Response	Description
Temporary failure	server	server: fail client: retry request after delay	Any “temporary failure” condition local to the server
Request not allowed	server	server: fail client: fail to request	Generic error for denied requests
Unregistered client	server	server: fail client: fail to request or notify	The client has not registered with the server.
Unsupported operation	server	server: fail client: fail to request or notify	Reception of an unsupported operation from the client

Table 37 Generic Errors

11.1.2 Registration Errors

This section addresses errors that may occur during content registration.

Error Condition	Detected by	Response	Description
Already exist	server	server: fail client: fail	The content to be registered has been registered before.
Metadata untrustworthy	server	server: fail client: retry with trustworthy metadata.	The metadata submitted by the Registration Client is not trusted by the CIM.

Table 38 Registration Errors

11.1.3 Identification Errors

This section addresses errors that may occur during content identification.

Error Condition	Detected by	Response	Description
No matched content	server	server: fail, MAY respond with further recommendation. client: fail or try with other identification mechanisms	No matched content is found and the server SHALL return the result to the client.
Over time limitation	server	server: discard	No identification result is obtained before the time limitation specified by the client expires.
Unsupported	server	server: fail	The server does not support content identification in the application type specified

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Error Condition	Detected by	Response	Description
application type		client: fail	by the client.
Unsupported content type	server	server: fail client: fail	The server does not support the identification of the type of content specified by the client.
Disallowed synchronization	server	server: respond with no registration information to the client. client: give up the synchronization.	The requested registration information is not allowed to be sent to the client.

Table 39 Identification Errors

11.2 Error Codes

This section provides the numeric values and text description for the error codes.

Error Code	Error Description
000	Identification success
001	Malformed message
002	Invalid parameter
003	Authentication error
004	Temporary failure
005	Request not allowed
006	Unregistered client
007	Unsupported operation
008	Already exist
009	Metadata untrustworthy
010	No matched content
011	Over time limitation
012	Unsupported application type
013	Unsupported content type
014	Disallowed synchronization

Table 40 Error Codes

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

12. SCIDM Adaptation with Transport Protocol

12.1 SCIDM XML Schema

The SCIDM XML schema is based on the definitions of SCIDM messages (Section 7).

The XML schema defines SCIDM messages as distinct XML elements with message structure enforced according to the definitions in Section 7. This approach ensures better interoperability, built-in message validation, and parsing efficiency.

See [SCIDM-XSD] for XML schema definition.

12.2 SCIDM Media Type

The following Media Types are supported by SCIDM and are carried by the relevant mechanisms depending on the transport bearer that is used, e.g., HTTP headers.

Textual form (XML): application/vnd.oma.scidm.messages+xml

12.3 Transport in Push Mode

The SCIDM Server (CIM) SHOULD support either the Push Access Protocol (PAP) [PUSH-PAP] or the Push-OTA protocol over HTTP (OTA-HTTP) [PUSH-OTA] for notification push from the CIM to the SCIDM Client via the SCIDM-3 interface.

For SCIDM Push messages, the CIM SHALL include the Push Application ID header “X-Wap-Application-Id: x-oma-application: scidm.ua”.

When using PAP or OTA-HTTP, the CIM SHALL submit SCIDM messages for delivery using the MIME content type “application/vnd.oma.scidm.messages+xml”.

Push Clients in SCIDM supporting terminals SHALL support routing of Push messages with the Push Application ID header “X-Wap-Application-Id: x-oma-application: scidm.ua” to the SCIDM Client.

12.4 Transport in Pull Mode

12.4.1 HTTP Transport Binding

SCIDM Clients and SCIDM Servers (CIMs) SHALL support Hypertext Transfer Protocol version 1.1 (HTTP [RFC2616]) for the SCIDM-1, SCIDM-2 and SCIDM-3 interfaces.

CIMs and SCIDM Clients SHALL support SCIDM-1, SCIDM-2, and SCIDM-3 interface messages formatted as entity-bodies with the application/vnd.oma.scidm.messages+xml media type.

SCIDM Clients SHALL send all SCIDM-1 SCIDM-2 and SCIDM-3 interface messages as HTTP POST method requests, including:

- The CIM’s address in the request line
- the Host request-header set to the hostname or IP address of the CIM
- the User-Agent request-header set to identify the host device (e.g. “vendor-model/version”), and the name and version of the SCIDM Client as user agent initiating the request
- the Accept request-header with value “application/vnd.oma.scidm.messages+xml”
- the Accept-Encoding request-header with value per the supported HTTP compression encodings, i.e. deflate and / or gzip
- the Content-Length entity-header set to the length of the entity-body

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

- the Content-Type entity-header with value “application/vnd.oma.scidm.messages+xml”
- the SCIDM-1, SCIDM-2, or SCIDM-3 message as message-body

CIMs SHALL send all SCIDM-1, SCIDM-2, and SCIDM-3 interface messages as responses to SCIDM Client messages as the entity-body of HTTP 200 OK responses, including:

- the ETag entity-header set to a unique value within the scope of the CIM
- the Content-Encoding entity-header set to the type of HTTP compression applied, if any
- the Content-Length entity-header set to the length of the entity-body
- the Content-Type entity-header with value “application/vnd.oma.scidm.messages+xml”
- the SCIDM-1, SCIDM-2, or SCIDM-3 message as message-body

When there is no SCIDM message to send in response to a SCIDM Client request, the SCIDM Server SHALL send a “204 No Content” response.

The following table provides an overview of the HTTP binding for each transaction of SCIDM-1, SCIDM-2, and SCIDM-3.

HTTP Method	Request Body	Response Code	Response Body
POST	ContentRegistrationRequest	200 OK	ContentRegistrationResponse
POST	ContentUpdateRequest	200 OK	ContentUpdateResponse
POST	ContentDeregRequest	200 OK	ContentDeregResponse
POST	ContentIdentRequest	200 OK	ContentIdentResponse
POST	ContentSyncRequest	200 OK	ContentSyncResponse

Table 41 Message Relations between HTTP requests and responses

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version –or- No previous version within OMA

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-SCIDM-V1_0	01 May 2009	all 1	Initial document to address the basic starting pointOMA-SEC-SCIDM-2009-0028-INP_SCIDM_TS_template.doc Incorporate agreed INP: OMA-SEC-SCIDM-2009-0030-INP_SCIDM_TS_Scope.doc
	10 June 2009	all	Incorporate agreed INP: OMA-SEC-SCIDM-2009-0029R01-INP_SCIDM_TS_Structure.doc
	17 July 2009	7 4 5.3 5.4 7.1.2 7.1.3 5.1 5.2	Incorporate agreed CRs: OMA-ARC-SCIDM-2009-0056-CR_SCIDM_TS_Registration_Interface.doc OMA-ARC-SCIDM-2009-0058-CR_TS_Intro_Section.doc OMA-ARC-SCIDM-2009-0059R01-CR_TS_Process_Online_Identification.doc OMA-ARC-SCIDM-2009-0060R01-CR_TS_Process_Offline_Identification.doc OMA-ARC-SCIDM-2009-0061R01-CR_TS_Interface_SCIDM_2.doc OMA-ARC-SCIDM-2009-0062R01-CR_TS_Interface_SCIDM_3.doc OMA-ARC-SCIDM-2009-0049R02-CR_TS_Processes_Registration.doc OMA-ARC-SCIDM-2009-0050R01-CR_TS_Processes_Identification.doc
	10 Sep 2009	All 3.2,3.3 2.1,2.2 4.1 Appendix C 6 B.1 2.2,9 11 7.1.3.2.1 2.2,8 B.2 B.3 2.1,10	Incorporate agreed CRs: OMA-ARC-SCIDM-2009-0064R01-CR_TS_Client_Clarification.doc OMA-ARC-SCIDM-2009-0065R01-CR_TS_Defs_Abbreviations.doc OMA-ARC-SCIDM-2009-0066-CR_Ref.doc OMA-ARC-SCIDM-2009-0067R01-CR_version_desc OMA-ARC-SCIDM-2009-0069R01-CR_Implementation_Deployment_Guidelines.doc OMA-ARC-SCIDM-2009-0070R01-CR_TS_Client_Server_Operations.doc OMA-ARC-SCIDM-2009-0071R01-CR_SCR_for_SCIDM1.doc OMA-ARC-SCIDM-2009-0072R01-CR_Fingerprinting_Algorithms.doc OMA-ARC-SCIDM-2009-0073-CR_TS_Error_Handling.doc OMA-ARC-SCIDM-2009-0074-CR_TS_Sync_Notification_Bugfix.doc OMA-ARC-SCIDM-2009-0075-CR_Content_Metadata_Def.doc OMA-ARC-SCIDM-2009-0077R01-CR_SCR_for_SCIDM2.doc OMA-ARC-SCIDM-2009-0078R01-CR_SCR_for_SCIDM3.doc OMA-ARC-SCIDM-2009-0079-CR_TS_Security_Mechanisms.doc
	9 Oct 2009	12,2.1,App 7.1.1.1.1,7.1.1.2.1,7.1.1.3.1,7.1.2.1.1,B.2	Incorporate agreed CR: OMA-ARC-SCIDM-2009-0076R02-CR_Protocol_Adaptation.doc OMA-ARC-SCIDM-2009-0080-CR_TS_Interfaces_Improvements.doc
	22 Oct 2009	5.4.2 B.1 9.2,Annex A, Appendix E,2.1,2.2 6.1.1 B.1	Incorporate agreed CR: OMA-ARC-SCIDM-2009-0068-CR_Content_ID_Certificate.doc OMA-ARC-SCIDM-2009-0083-CR_Changes_SCIDM_SCR.doc OMA-ARC-SCIDM-2009-0085R02-CR_Fingerprint_Container_File_Format.doc OMA-ARC-SCIDM-2009-0086R02-CR_bugfix_for_sec6.1.1.doc OMA-ARC-SCIDM-2009-0087-CR_SCIDM3_PUSH_SCR.doc OMA-ARC-SCIDM-2009-0084-CR_Clerical_Fix_Figure_n_Table_Num.doc
	04 Nov 2009	All	Minor editorial fixes (styles) History moved to App A Cross-references fixed.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Document Identifier	Date	Sections	Description

	07 Jan, 2010	All	Incorporate agreed CR:
			OMA-ARC-SEC-2009-0088R01-CR_SCIDM_CR_0091;
			OMA-ARC-SEC-2009-0090-CR_SCIDM_CR_0093R01;
			OMA-ARC-SCIDM-2009-0090-CR_CONR_Comments_Resolution_Clarifications;
			OMA-ARC-SCIDM-2009-0089-CR_CONRR_Comments_Resolution_Editor_Notes;

Candidate Version OMA-TS-SCIDM-V1_0	09 Feb 2010	All	Status changed to Candidate by TP: OMA-TP-2010-0013-INP_SCIDM_V1_0_ERP_for_Candidate_Approval

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Appendix B. Syntax of XFP File Format (Normative)

This Appendix provides syntax specifications and descriptions of the XFP fingerprint container file format. An introduction to the file format and an overview of the XFP file structure are provided in Section 9.2.

The syntax described herein assumes little-endian byte order. A class-like pseudo-declaration is used to define the structural elements.

B.1 XFP Start Code

B.1.1 Definition

Key:	'++VXFP' (0x2B2B56584650)
Container:	XFP file
Mandatory:	YES
Quantity:	Exactly one

XFP Start Code is a 48-bit magic number that identifies the beginning of an XFP file.

B.2 XFP CRC Box

B.2.1 Definition

Key:	'xcrc'
Container:	XFP file
Mandatory:	YES
Quantity:	Exactly one

XFP CRC Box is a mandatory box that follows XFP Start Code immediately. It contains CRC codes for the XFP Meta Box and XFP Data Box.

B.3 Syntax

```
aligned(8) class XFP_CRC_Box {
                const unsigned int (32) key = 'xcrc';
                unsigned int (32) size;
                unsigned int (32) meta_crc;
                unsigned int (32) data_crc;
}
```

meta_crc is the 32-bit CRC code using IEEE 802.3 polynomial [IEEE-802.3-2008] for the data contained in the XFP Meta Box. Zero value indicates that meta_crc is not known.

data_crc is the 32-bit CRC code using IEEE 802.3 polynomial for the data contained in the XFP Data Box. Zero value indicates that data_crc is not known.

B.4 XFP Meta Container Structure

B.4.1 XFP Meta Box

B.4.1.1 Definition

Key:	'xfpm'

Container:	XFP file
Mandatory:	YES
Quantity:	Exactly one

XFP Meta Box is the container for all metadata related to the Content Fingerprint or XFP data and the media data from which the Content Fingerprint is computed.

B.4.1.2 Syntax

```
aligned(8) class XFP_Meta_Box {
             const unsigned int (32) key = ‘xfpm’;
             unsigned int (32) size;
             unsigned int (8) contained_box [];
}
```

The XFP Meta Box is a container box that contains other boxes.

B.4.2 XFP Header Box

B.4.2.1 Definition

Key:	‘fphd’
Container:	XFP Meta Box (‘xfpm’)
Mandatory:	YES
Quantity:	Exactly one

The XFP Header Box contains information about the XFP file.

B.4.2.2 Syntax

```
aligned(8) class XFP_Header_Box {
             const unsigned int (32) key = ‘fphd’;
             unsigned int (32) size;
             unsigned int (32) version;
             unsigned int (32) creator;
             unsigned int (32) flag;
             unsigned int (32) creation_time;
             unsigned int (32) modification_time;
             unsigned int (32) stream_count;
             const unsigned int (32) reserved = 0;
}
```

`version` is an integer specifying the version of XFP Header Box.

`creator` is a 4-character code identifying the application that created this XFP file. An unknown application is represented by '????'.

`flag` is a set of bit masks that indicate the properties of the file or its content.

`creation_time` is an integer that declares the creation time of this XFP file (in seconds since midnight, January 1, 1904, in UTC time).

`modification_time` is an integer that declares the most recent time when this XFP file was modified (in seconds since midnight, January 1, 1904, in UTC time).

`stream_count` is an integer that indicates the number of XFP streams that are stored in this XFP file. It should be equal to or greater than 1. When the media that the Content Fingerprint corresponds to is video or audio, each XFP stream shall correspond to one and only one unique video or audio track in the original media file.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

B.4.3 Source File Attributes Box

B.4.3.1 Definition

Key:	‘sfat’
Container:	XFP Meta Box (‘xfpm’)
Mandatory:	NO
Quantity:	Zero or one

The Source File Attributes Box contains information about the source file from which the Content Fingerprint is generated.

B.4.3.2 Syntax

```
aligned(8) class Source_File_Attributes_Box {
             const unsigned int (32) key = ‘sfat’;
             unsigned int (32) size;
             unsigned int (32) creation_time;
             unsigned int (32) modification_time;
             unsigned int (64) file_size;
             unsigned int (32) file_format;
             unsigned int (8) hash_code[64];
             unsigned int (8) hash_type;
}
```

`creation_time` is an integer that stores the creation time of the source file (in seconds since midnight, January 1, 1904, in UTC time).

`modification_time` is an integer that stores the most recent time when the source file was modified (in seconds since midnight, January 1, 1904, in UTC time).

`file_size` is the size of the source file in bytes.

`file_format` is a 4-character code identifying the format of the source file. Since there is no global file format registry, it is recommended that the file extension be used to identify the file, e.g., '.avi'.

`hash_code` contains the binary hash value of the source file. The hash function used to generate the `hash_code` is indicated by `hash_type`.

`hash_type` indicates the hash function that is used to generate the `hash_code`. Currently defined hash functions are: 1 (MD5), and 2 (SHA-1).

B.4.4 XFP Stream Information Box

B.4.4.1 Definition

Key:	‘fpsi’
Container:	XFP Meta Box (‘xfpm’)
Mandatory:	YES
Quantity:	One or more

The XFP Stream Information Box contains information about an XFP stream. There shall be at least one such box contained in the XFP Meta Box.

B.4.4.2 Syntax

```
aligned(8) class XFP_Stream_Information_Box {
             const unsigned int (32) key = ‘fpsi’;
             unsigned int (32) size;
```

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

```
                unsigned int (8) contained_box [];
}
```

The XFP Stream Information Box is a container box that contains other boxes.

B.4.5 XFP Stream Description Box

B.4.5.1 Definition

Key:	‘fpsd’
Container:	XFP Stream Information Box (‘fpsi’)
Mandatory:	YES
Quantity:	Exactly one

The XFP Stream Description Box describes an XFP stream. This box is contained in the XFP Stream Information Box.

B.4.5.2 Syntax

```
aligned(8) class XFP_Stream_Description_Box {
                const unsigned int (32) key = ‘fpsd’;
                unsigned int (32) size;
                unsigned int (32) version;
                unsigned int (32) stream_id;
                unsigned int (32) stream_type;
                unsigned int (32) XFP_type;
                unsigned int (32) offset;
                unsigned int (32) time_scale;
                unsigned int (32) duration;
                unsigned int (32) layer_count;
}
```

`version` is an integer specifying the version of XFP Stream Description for the corresponding type of stream.

`stream_id` is an integer that identifies the XFP stream.

`stream_type` is an integer that indicates the type of the stream. Currently defined types are: 1 (VFP - video fingerprint), 2 (AFP - audio fingerprint), 3 (IFP - image fingerprint).

`XFP_type` is a 4-character code identifying the type of XFP data corresponding to a specific content fingerprinting algorithm.

`offset` is an integer that defines the offset in bytes from the beginning of the file to the start of this stream in the XFP file.

`time_scale` is an integer that specifies the time-scale for this XFP stream; this is the number of time units that pass in one second. For example, a time coordinate system that measures time in sixtieths of a second has a time scale of 60. For XFP streams of non-time-based type such as IFP, this value should be set to 0.

`duration` is an integer that declares the duration of this XFP stream (in the scale of `timescale`). If an XFP stream consists of multiple layered XFP streams, the `duration` is the duration of the longest layered stream. For XFP streams of non-time-based type such as IFP, this value should be set to 0.

`layer_count` is an integer that indicates the number of layers that appear in this XFP stream. It should be equal to or greater than 1.

B.4.6 Source Video Sequence Attributes Box

B.4.6.1 Definition

Key:	‘svat’

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Container:	XFP Stream Information Box (‘fpsi’)
Mandatory:	NO
Quantity:	Zero or one

The Source Video Sequence Attributes Box contains information about a video track in the source file from which the VFP is generated.

B.4.6.2 Syntax

```
aligned(8) class Source_Video_Sequence_Attributes_Box {
             const unsigned int (32) key = ‘svat’;
             unsigned int (32) size;
             unsigned int (32) video_frame_rate;
             unsigned int (32) video_frame_width;
             unsigned int (32) video_frame_height;
             unsigned int (32) video_codec;
             unsigned int (32) video_bitrate;
             unsigned int (32) video_duration;
}
```

`video_frame_rate` is an integer that represents the frame rate of the video track. It is in 16.16 fixed-point representation, i.e., (x) << 16.

`video_frame_width` is an integer that represents the width of the video frame in pixels.

`video_frame_height` is an integer that represents the height of the video frame in pixels.

`video_codec` is a 4-character code identifying the codec that is used to encode the video track, e.g., ‘h264’.

`video_bitrate` is an integer that represents the average bitrate in bits/second of the video track.

`video_duration` is an integer that represents the duration in seconds of this video track.

B.4.7 Source Audio Sequence Attributes Box

B.4.7.1 Definition

Key:	‘saat’
Container:	XFP Stream Information Box (‘fpsi’)
Mandatory:	NO
Quantity:	Zero or one

The Source Audio Sequence Attributes Box contains information about an audio track in the source file from which the AFP is generated.

B.4.7.2 Syntax

```
aligned(8) class Source_Audio_Sequence_Attributes_Box {
             const unsigned int (32) key = ‘saat’;
             unsigned int (32) size;
             unsigned int (32) audio_channels;
             unsigned int (32) audio_sample_rate;
             unsigned int (32) audio_sample_size;
             unsigned int (32) audio_sample_format;
             unsigned int (32) audio_codec;
             unsigned int (32) audio_bitrate;
             unsigned int (32) audio_duration;
             unsigned int (8) audio_language[3];
}
```

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

audio_channels is an integer that indicates the number of channels in the audio track.

audio_sample_rate is an integer that represents the sample rate of the audio track. It is in 16.16 fixed-point representation, i.e., (x) << 16.

audio_sample_size is an integer that represents the number of the audio samples in an audio frame.

audio_sample_format is an integer that indicates the format of audio samples.

audio_codec is a 4-character code identifying the codec that is used to encode the audio track, e.g., ‘faac’.

audio_bitrate is an integer that represents the average bitrate in bits/second of the audio track.

audio_duration is an integer that represents the duration in seconds of this audio track.

audio_language is a 3-byte code indicating the language of the audio track. The encoding of audio_language shall be compliant to ISO 639-1 [ISO 639-1] and ISO 639-2 [ISO-639-2], with ISO 639-1 preferred. This means the 2-letter language code in ISO 639-1 is preferred unless the language to be represented is not available in ISO 639-1; in that case, the 3-letter language code in ISO 639-2 shall be used.

B.4.8 Source Image Attributes Box

B.4.8.1 Definition

Key:	‘siat’
Container:	XFP Stream Information Box (‘fpsi’)
Mandatory:	NO
Quantity:	Zero or one

The Source Image Attributes Box contains information about the image from which the IFP is generated.

B.4.8.2 Syntax

```
aligned(8) class Source_Image_Attributes_Box {
             const unsigned int (32) key = ‘siat’;
             unsigned int (32) size;
             unsigned int (32) image_width;
             unsigned int (32) image_height;
             unsigned int (32) image_codec;
             unsigned int (32) image_color_component;
             unsigned int (32) image_pixel_bitrate;
}
```

image_width is an integer that represents the width of the image in pixels.

image_height is an integer that represents the height of the image in pixels.

image_codec is a 4-character code identifying the codec that is used to encode the image, e.g., ‘jpeg’.

image_color_component is an integer that represents the number of color components in the image.

image_pixel_bitrate is an integer that represents the average bitrate in bits/pixel of the image.

B.4.9 XFP Layer Information Box

B.4.9.1 Definition

Key:	‘fpli’
Container:	XFP Meta Box (‘xfpm’)
Mandatory:	YES

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Quantity: One or more

The XFP Layer Information Box contains information about an XFP layer. Layer is a vertical structure of XFP data. An XFP stream can be layered such that each layered XFP substream contains XFP data that belongs to the same, unique, non-overlapping layer. There shall be at least one such box contained in the XFP Meta Box.

B.4.9.2 Syntax

```
aligned(8) class XFP_Layer_Information_Box {
             const unsigned int (32) key = ‘fpli’;
             unsigned int (32) size;
             unsigned int (8) contained_box [];
}
```

The XFP Layer Information Box is a container box that contains other boxes.

B.4.10 XFP Layer Description Box

B.4.10.1 Definition

Key: ‘fpld’

Container: XFP Layer Information Box (‘fpli’)

Mandatory: YES

Quantity: Exactly one

The XFP Layer Description Box describes a layer of XFP data. This box is contained in the XFP Layer Information Box.

B.4.10.2 Syntax

```
aligned(8) class XFP_Layer_Description_Box {
             const unsigned int (32) key = ‘fpld’;
             unsigned int (32) size;
             unsigned int (32) version;
             unsigned int (32) XFP_type;
             unsigned int (32) layer_id;
             unsigned int (32) layer_type;
             int (32) layer_size;
}
```

`version` is an integer specifying the version of XFP Layer Description. Note that when Content Fingerprint data is subdivided into multiple layers, each layer will be defined using an instance of XFP Layer Description Box with different `layer_id` and `layer_type`, but the `version` number and `XFP_type` value should be identical for all these layers.

`XFP_type` is a 4-character code identifying the type of XFP data corresponding to a specific content fingerprinting algorithm.

`layer_id` is an integer that identifies the XFP layer.

`layer_type` is a 4-character code that identifies the layer. The `layer_type` may be registered along with `XFP_type` or defined privately. A `layer_type` = 'DFLT' is reserved for each Content Fingerprint that is represented by an `XFP_type` to denote a default layer partition that is a single layer.

`layer_size` is an integer that indicates the size of a layer in bytes. A negative value indicates the layer has a variable size.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

B.5 XFP Data Container Structure

B.5.1 XFP Data Box

B.5.1.1 Definition

Key: ‘xfpd’
Container: XFP file
Mandatory: YES
Quantity: Exactly one

The XFP Data Box is a mandatory box that follows XFP Meta Box. It contains the data of one or more XFP streams.

B.5.1.2 Syntax

```
aligned(8) class XFP_Data_Box {
             const unsigned int (32) key = ‘xfpd’;
             unsigned int (32) size;
             unsigned int (8) contained_box [];
}
```

The XFP Data Box is the container box that contains other boxes.

B.5.2 XFP Stream Data Box

B.5.2.1 Definition

Key: ‘stda’
Container: XFP Data Box (‘xfpd’)
Mandatory: YES
Quantity: One or more

The XFP Stream Data Box contains data of one or more layered XFP substreams.

B.5.2.2 Syntax

```
aligned(8) class XFP_Stream_Data_Box {
             const unsigned int (32) key = ‘stda’;
             unsigned int (32) size;
             unsigned int (8) contained_box [];
}
```

The XFP Stream Data Box is a container box that contains other boxes.

B.5.3 XFP Layer Stream Data Box

B.5.3.1 Definition

Key: ‘lada’
Container: XFP Stream Data Box (‘stda’)
Mandatory: YES
Quantity: One or more

The XFP Layer Stream Data Box contains data of a layered XFP substream that corresponds to one VFP or AFP layer.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

B.5.3.2 Syntax

```
aligned(8) class XFP_Layer_Stream_Data_Box {
             const unsigned int (32) key = 'lada';
             unsigned int (32) size;
             unsigned int (32) stream_id;
             unsigned int (32) layer_id;
             unsigned int (8) stream_data [];
}
```

`stream_id` is an integer that identifies the XFP stream. It must be valid `stream_id` that is defined in an XFP Stream Information Box.

`layer_id` is an integer that identifies the XFP layer. It must be a valid `layer_id` that is defined in an XFP Layer Information Box.

`stream_data[]` is a block of binary data.

B.6 Miscellaneous Syntax Elements

B.6.1 Padding Data Box

B.6.1.1 Definition

Key:	'padd'
Container:	XFP file or any XFP container box
Mandatory:	NO
Quantity:	Zero, one or more

The Padding Data Box contains a block of binary data that is used solely for padding a container to certain size, usually for the purpose of alignment. It can be contained in an XFP file inside or outside an XFP container box.

B.6.1.2 Syntax

```
aligned(8) class Padding_Data_Box {
             const unsigned int (32) key = 'padd';
             unsigned int (32) size;
             unsigned int (8) padding_data [];
}
```

`padding_data []` is a block of binary data that is not to be used. Although not required, it is recommended to use zero-valued padding data.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Appendix C. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

C.1 SCR for SCIDM-1

C.1.1 SCR for SCIDM-1

C.1.1.1 SCR for SCIDM-1 Request-Response

Item	Function	Reference	Requirement
SCIDM - SCIDM1-REQ_RESP-001-M	Request-response	7.1.1	SCIDM-SCIDM1-REQ- 001-M AND SCIDM-SCIDM1-RESP-001-M AND SCIDM-SCIDM1-HTTP-001-M

C.1.1.2 SCR for SCIDM-1 Request

SCIDM-SCIDM1-REQ-001-M	Request	7.1.1	SCIDM- SCIDM1-REQ- 002-O OR SCIDM- SCIDM1-REQ- 003-O
SCIDM- SCIDM1-REQ-002-O	Request carrying Content	7.1.1	SCIDM- SCIDM1-REQ- 004-M AND SCIDM- SCIDM1-REQ- 005-O
SCIDM- SCIDM1-REQ-003-O	Request carrying ContentFingerprint	7.1.1	SCIDM- SCIDM1-REQ- 004-M AND SCIDM- SCIDM1-REQ- 006-O
SCIDM- SCIDM1-REQ-004-M	Support of ContentMetadata	7.1.1	
SCIDM- SCIDM1-REQ-005-O	Support of Content	7.1.1	
SCIDM- SCIDM1-REQ-006-O	Support of ContentFingerprint & ContentFingerprintAlgID	7.1.1	

C.1.1.3 SCR for SCIDM-1 Response

Item	Function	Reference	Requirement
SCIDM-SCIDM1-RESP-001-M	Response to SCIDM-1 request	7.1.1	SCIDM- SCIDM1-RESP- 002-M AND SCIDM- SCIDM1-RESP-003-M
SCIDM- SCIDM1-RESP-002-M	Support of status code	7.1.1	
SCIDM- SCIDM1-RESP-003-M	Support of ContentID	7.1.1	
SCIDM- SCIDM1-RESP-004-O	Support of Content ID Certificate	7.1.1	

C.1.1.4 SCR for SCIDM-1 Protocol Binding

Item	Function	Reference	Requirement
SCIDM-SCIDM1-HTTP-001-M	HTTP protocol binding	12, 12.1, 12.2, 12.4	

C.2 SCR for SCIDM-2

C.2.1 SCR for SCIDM-2

C.2.1.1 SCR for SCIDM-2 Request-Response

Item	Function	Reference	Requirement
SCIDM – SCIDM2-REQ_RESP-001-M	Request-response	7.1.2	SCIDM-SCIDM2-REQ-001-M AND SCIDM-SCIDM2-RESP-001-M AND SCIDM-SCIDM2-HTTP-001-M

C.2.1.2 SCR for SCIDM-2 Request

SCIDM-SCIDM2-REQ-001-M	Request	7.1.2	SCIDM- SCIDM2-REQ-002-M
SCIDM- SCIDM2-REQ-002-M	Request carrying ContentIdentInfo	7.1.2	SCIDM- SCIDM2-REQ-003-O OR SCIDM- SCIDM2-REQ-004-O OR SCIDM- SCIDM2-REQ-005-O OR SCIDM- SCIDM2-REQ-006-O
SCIDM- SCIDM2-REQ-003-O	Support of ByID mechanism	7.1.2	
SCIDM- SCIDM2-REQ-004-O	Support of ByHistory mechanism	7.1.2	
SCIDM- SCIDM2-REQ-005-O	Support of ByMetadata mechanism	7.1.2	
SCIDM- SCIDM2-REQ-006-O	Support of ByFingerprint mechanism	7.1.2	SCIDM- SCIDM2-REQ-007-O OR SCIDM- SCIDM2-REQ-008-O
SCIDM- SCIDM2-REQ-007-O	Support of Content	7.1.2	
SCIDM- SCIDM2-REQ-008-O	Support of ContentFingerprint & ContentFingerprintAlgID	7.1.2	

C.2.1.3 SCR for SCIDM-2 Response

Item	Function	Reference	Requirement
SCIDM-SCIDM2-RESP-001-M	Response to SCIDM-2 request	7.1.2	SCIDM- SCIDM2-RESP-002-M AND (SCIDM- SCIDM2-RESP-003-O OR SCIDM- SCIDM2-RESP-004-O OR SCIDM-SCIDM2-RESP-005-O)
SCIDM- SCIDM2-RESP-002-M	Support of status code	7.1.2	
SCIDM- SCIDM2-RESP-003-O	Support of ContentID	7.1.2	
SCIDM- SCIDM2-RESP-004-O	Support of ContentMetadata	7.1.2	
SCIDM- SCIDM2-RESP-005-O	Support of RegistrationInfo	7.1.2	

C.2.1.4 SCR for SCIDM-2 Protocol Binding

Item	Function	Reference	Requirement
SCIDM-SCIDM2-HTTP-001-M	HTTP protocol binding	12, 12.1, 12.2, 12.4	

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

C.3 SCR for SCIDM-3

C.3.1 SCR for SCIDM-3

C.3.1.1 SCR for SCIDM-3 Request-Response

Item	Function	Reference	Requirement
SCIDM – SCIDM3-REQ_RESP-001-M	Request-response	7.1.3	SCIDM-SCIDM3-REQ-001-M AND SCIDM-SCIDM3-RESP-001-M AND SCIDM-SCIDM3-BIND-001-M

C.3.1.2 SCR for SCIDM-3 Request

SCIDM-SCIDM3-REQ-001-M	Request	7.1.3	SCIDM- SCIDM3-REQ-002-M AND (SCIDM- SCIDM3-REQ-003-O OR SCIDM- SCIDM3-REQ-004-O)
SCIDM- SCIDM3-REQ-002-M	Request carrying Initiator information	7.1.3	
SCIDM- SCIDM3-REQ-003-O	Request carrying ContentIDList	7.1.3	SCIDM- SCIDM3-REQ-005-O
SCIDM- SCIDM3-REQ-004-O	Request carrying SyncRule	7.1.3	
SCIDM- SCIDM3-REQ-005-O	Support of ContentIDList	7.1.3	

C.3.1.3 SCR for SCIDM-3 Response

Item	Function	Reference	Requirement
SCIDM-SCIDM3-RESP-001-M	Response to SCIDM-3 request	7.1.3	SCIDM- SCIDM3-RESP-002-M
SCIDM- SCIDM3-RESP-002-M	Support of status code	7.1.3	
SCIDM- SCIDM3-RESP-003-O	Support of RegInfoList	7.1.3	

C.3.1.4 SCR for SCIDM-3 Notification

Item	Function	Reference	Requirement
SCIDM-SCIDM3-NOTIF-001-M	SCIDM-3 push notification	7.1.3.2	SCIDM-SCIDM3- BIND-002-M

C.3.1.5 B.3.2 SCR for SCIDM-3 Protocol Binding

Item	Function	Reference	Requirement
SCIDM-SCIDM3-BIND-001-M	HTTP protocol binding	12, 12.1, 12.2, 12.4	
SCIDM-SCIDM3- BIND-002-M	Push protocol binding	12.3	SCIDM-SCIDM3- BIND-003-O OR SCIDM-SCIDM3- BIND-004-O
SCIDM-SCIDM3- BIND-003-O	Push Access Protocol binding	12.3	
SCIDM-SCIDM3- BIND-004-O	Push-OTA protocol over HTTP binding	12.3	

Appendix D. Implementation and Deployment Guidelines (Normative)

D.1 Deployment of SCIDM Server and Client

The SCIDM Enabler aims to provide the basic function of content identification and can be used by many content management applications, such as copyright verification and usage statistics, content filtering, content tracing, remote triggering of ads, etc. The deployment of the SCIDM Enabler is very flexible.

The possible deployment modes of the SCIDM Server are:

- As an independent platform to provide content identification and metadata query services.
- As an embedded module in some application system

The deployment of the SCIDM Client is more flexible and in most cases is related to some application system.

For example, in the copyright verification and usage statistics application, the SCIDM Client can be deployed in an independent machine in the network which targets at scanning some websites or monitoring some gateways. The SCIDM Client can also be located in a website and is dedicated to verify the copyright of the content submitted to the website. In some other scenarios, the SCIDM Client can be deployed in the user device to account for the upload or download of copyright protected content.

In the spam filtering application, the SCIDM Client can be implemented in the SMS/MMS Server and used to detect the spam messages. The SCIDM Client can also be deployed in the user device and perform content detection specific to the user’s interest.

Depending on the application, the SCIDM Client can acquire the Content Metadata from the CIM or only acquire the globally unique ID of the content which is then forwarded to the application system for application specific usage.

D.2 Guidelines on Performance Issues

In the SCIDM Enabler, the performance of the identification process is an important issue. In a practical system, measure should be taken to address the performance problem. Some measures are recommended here.

- Non-real-time identification. In some application scenarios, non-real-time identification is permitted. For example, if the copyrighted content will not be blocked as soon as it is uploaded to a website based on some policy, the identification can be done in non-real-time.
- Identification by partial content. To identify audio and video content, the identification can be based on partial content which can significantly improve the speed.

The SCIDM Enabler supports multiple identification mechanisms (Section 5.2). The actual system should select appropriate mechanisms based on the system load and security requirements, and realtime/non-realtime conditions. For more detailed description, please refer to Section 5.2 and 5.3.

D.3 The CIM Interoperation

When a CIM does not have the registration information of the content to be identified, it can interoperate with other CIMs to query the registration information. For the general description of the CIM Interoperation, please refer to Section C.2 in [SCIDM-AD]. Two modes of CIM interoperation are recommended including the Central CIM mode and the CIM List mode. For the flow of the two modes, please refer to Section B.4 in [SCIDM-AD].

In the Central CIM mode, the transaction of building the index data (Section B.4 in [SCIDM-AD]) reuses the Content Registration transaction (Section 7.1.1.1), with the local CIM as the Registration Client and the Central CIM as the SCIDM Server. In the message ContentRegistrationRequest, the parameter RegistrantID is set to be the ID of the local CIM and the parameter RegistrantName is set to be the domain name of the local CIM or as null. The structure ContentRegInfo only needs to contain the parameter ContentFingerprint.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.

Similarly, the local CIM uses the Content Update transaction (Section 7.1.1.2) and the Content Deregistration transaction (Section 7.1.1.3) to update or un-register the index data in the Central CIM.

The query transaction to the Central CIM reuses the SCIDM Online Identification and Query transaction (Section 7.1.2.1) with the local CIM as the Identification Client and the Central CIM as the SCIDM Server. After the local CIM gets the index data from the Central CIM, it needs to interoperate with the registration information hosting CIM to query the metadata. This transaction also reuses the Online Identification and Query transaction, with the local CIM as the Identification Client and the registration information hosting CIM as the SCIDM Server.

In the CIM List mode, the transaction between the local CIM and the remote CIM reuses the Online Identification and Query transaction (Section 7.1.2.1), with the local CIM as the Identification Client and the remote CIM as the SCIDM Server.

Appendix E. Media-Type Registrations (Informative)

E.1 Media-Type Registration Request for application/vnd.oma.scidm.messages+xml

This section provides the registration request, as per [RFC 2048], to be submitted to IANA.

Type name:	application
Subtype name:	vnd.oma.scidm.messages+xml
Required parameters:	none
Optional parameters:	none
Encoding considerations:	binary

Security considerations:

Client-server request and respond messages are passive, meaning they do not contain executable or active content which may represent a security threat. The format does not include confidential fields. As client-server request and respond messages convey information which services a user accesses, there is some risk that unintentional information may be exposed. The information present in this media format is used to support content registration, identification and query. Thus, the usage of the format may be vulnerable to attacks modifying or spoofing the content of this format. Depending on the system architecture, it is recommended to use source authentication and integrity protection.

Interoperability considerations:

This content type carries client-server messages within the scope of the OMA SCIDM Enabler. The OMA SCIDM Enabler specification includes static conformance requirements and interoperability test cases for this content.

Published specification:

OMA SCIDM 1.0 Enabler Specification – Secure Content Identification Mechanism, especially Section 7.1. Available from http://www.openmobilealliance.org

Applications, which use this media type:

OMA SCIDM Services

Additional information:

Magic number(s):	none
File extension(s):	none
Macintosh File Type Code(s):	none

Person & email address to contact for further information:

Wenjun Zeng

zengw@huawei.com

Intended usage: Limited use.

Only for usage with client-server request and respond messages for SCIDM Services, which meet the semantics given in the mentioned specification.

Author/Change controller: OMNA – Open Mobile Naming Authority, OMA-OMNA@mail.openmobilealliance.org

Appendix F. XFP File Containing MPEG-7 Image Fingerprint (Informative)

The ISO has recently published an amendment of MPEG-7 to specify the algorithm and data format for image fingerprint [MPEG-7-IMG-SIG]. An example is given below to show how to wrap the MPEG-7 image fingerprint data in XFP.

F.1 MPEG-7 Image Signature

The MPEG-7 image fingerprint, known as Image Signature, uses the following binary representation syntax:

ImageSignature {	Number of bits	Mnemonics
GlobalSignatureA	512	bslbf
GlobalSignatureB	512	bslbf
FeaturePointCount	8	uimsbf
for(k=0; k<NumberOfPoints; k++) {		
Xcoord	8	uimsbf
Ycoord	8	uimsbf
Direction	4	uimsbf
LocalSignature	60	bslbf
}		
}		

where NumberOfPoints = FeaturePointCount + 32; the value of FeaturePointCount is restricted to the range 0-48.

For the purpose of wrapping MPEG-7 Image Signature in XFP, it is worth noting that the MPEG-7 Image Signature consists of two types of signatures: Global Signature and Local Signature. The Global Signature data is fixed in size, totalling 1024 bits for two Global Signature data blocks, while the Local Signature data has a variable size, depending on the number of feature points.

F.2 Wrapping MPEG-7 Image Signature in XFP File

An XFP file containing MPEG-7 image fingerprint data has a simple structure because the image fingerprint is not time-based and therefore the stream data structure in XFP reduces to a simpler form. Most mandatory boxes in the XFP file can be filled straightforwardly.

F.2.1 MPEG-7 Image Signature in Single Layer

Following the XFP Start Code, the XFP CRC Box ('xcrc') contains the CRC codes of the XFP Meta Box and XFP Data Box, respectively. Both CRC codes can be computed after the corresponding boxes are filled or left null to indicate that the CRC codes are unknown.

The XFP Meta Box ('xfpm') that follows the XFP CRC Box contains a number of mandatory and optional boxes. In the XFP Header Box ('fphd'), `version` is 1; `creator` is '????' if not known; `flag` should be left 0; `stream_count` is normally 1 for an image unless the image fingerprints correspond to a compound image composed of multiple sub-images.

It is optional to include a Source File Attributes Box ('sfat'). Because it is not possible to reconstruct the original content from its Content Fingerprint, it can be very helpful to save some information about the file from which the Content Fingerprint is computed. The attributes including `file_size`, `file_format`, and `hash_code` can be derived directly from the file.

In the XFP Stream Description Box ('fpsd') that is contained in the XFP Stream Information Box ('fpsi'), `version` is 1; `stream_id` is 1 and can be sequentially incremented if the XFP file contains more than one stream; `stream_type` is 3 for IFP or image fingerprint; `XFP_type` is a 4-character code defined by SCIDM for MPEG-7 Image Signature; `offset` is the offset in bytes from the beginning of the file to the start of the stream data, and can be filled after stream data is laid out; `time_scale` and `duration` should be set to 0 for image fingerprint; `layer_count` is 1 if no partitions of MPEG-7 Image Signature are defined. The option of partitioning MPEG-7 Image Signature into two layers is discussed in the next section.

It is optional to include a Source Image Attributes Box ('siat'). Because it is generally not possible to derive image attributes from an image fingerprint, it can be very helpful to save some important image attributes along with the image fingerprint data. The attributes including `image_width`, `image_height`, `image_codec`, `image_color_component`, and `image_pixel_bitrate` can be derived from the image data directly.

In the XFP Layer Description Box ('fpld') that is contained in the XFP Layer Information Box ('fpli'), `version` is 1; `XFP_type` is a 4-character code defined by SCIDM for MPEG-7 Image Signature; `layer_id` is 1 and can be sequentially incremented if the XFP stream contains more than one layer; `layer_type` is 'DFLT' for a single layer that contains entire MPEG-7 Image Signature. `layer_size` is -1 that indicates the layer has a variable size. The option of partitioning the MPEG-7 Image Signature into two layers is discussed in the next section.

The XFP Data Box ('xfpd') that follows the XFP Meta Box contains a single XFP Stream Data Box ('stda') that further contains an XFP Layer Stream Data Box ('lada'). Because there is only one IFP stream that has only one layer, `stream_id` is 1 and `layer_id` is 1, corresponding to the settings in the XFP Stream Description Box and XFP Layer Description Box, respectively. The `stream_data[]` is the MPEG-7 Image Signature data in its native binary representation.

F.2.2 MPEG-7 Image Signature in Two Layers

As is noted in Section F.1, the MPEG-7 Image Signature consists of fixed-sized Global Signature and variable-sized Local Signature. It can be useful to partition the MPEG-7 Image Signature into two layers containing Global Signature and Local Signature, respectively. In the following the required changes are described for constructing an XFP file that contains the MPEG-7 Image Signature in two layers.

In the XFP Stream Information Box ('fpsi'), there is still only one XFP Stream Description Box ('fpsd'), but `layer_count` should be set to 2.

In the XFP Layer Information Box ('fpli'), there are two instances of the XFP Layer Description Box ('fpld') to describe the Global Signature layer and Local Signature layer, respectively. `XFP_type` in both instances is the 4-character code defined by SCIDM for the MPEG-7 Image Signature. `layer_id` is 1 or 2 for Global Signature or Local Signature layer, respectively. `layer_type` in each instance is a 4-character code defined by SCIDM or privately for the MPEG-7 Global Signature or Local Signature, respectively. `layer_size` is 128 for the Global Signature layer and -1 (variable) for the Local Signature layer.

In the XFP Data Box ('xfpd'), there is still only one XFP Stream Data Box ('stda'), but the latter contains two instances of XFP Layer Stream Data Box ('lada'). `stream_id` is 1 in both instances; `layer_id` is 1 or 2 followed by the `stream_data`[] for Global Signature or Local Signature, respectively.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.
Used with the permission of the Open Mobile Alliance Ltd. under the terms as stated in this document.