Mobile Advertising Architecture

Approved Version 1.0 – 20 Mar 2012

Open Mobile Alliance

OMA-AD-Mobile_Advertising-V1_0-20120320-A

Contents

1. SCOPE (INFORMATIVE) 5
2. REFERENCES 6
2.1 NORMATIVE REFERENCES 6
2.2 INFORMATIVE REFERENCES 6
3. TERMINOLOGY AND CONVENTIONS 7
3.1 CONVENTIONS 7
3.2 DEFINITIONS 7
3.3 ABBREVIATIONS 8
4. INTRODUCTION (INFORMATIVE) 9
4.1 VERSION 1.0 9
5. ARCHITECTURAL MODEL 10
5.1 DEPENDENCIES 10
5.2 ARCHITECTURAL DIAGRAM 10
5.3 FUNCTIONAL COMPONENTS AND INTERFACES/REFERENCE POINTS DEFINITION 11
5.3.1 MobAd Enabler functional Components (Normative) 11
5.3.2 Entities External to the MobAd Enabler (Informative) 13
5.3.3 Interfaces (Normative) 14
5.4 SECURITY CONSIDERATIONS 15
APPENDIX A. CHANGE HISTORY (INFORMATIVE) 16
A.1 APPROVED VERSION HISTORY 16
APPENDIX B. FLOWS (INFORMATIVE) 17
B.1 AD APP REQUEST ADS CALL FLOW 17
B.2 AD APP REPORT AD METRICS DATA CALL FLOW 18
B.3 AD ENGINE AD REQUEST CALL FLOW 19
B.4 PUSH AD DELIVERY TO THE AD ENGINE CALL FLOW 20
B.5 AD ENGINE REPORT AD METRICS DATA CALL FLOW 21
B.6 AD ENGINE REQUESTING MOBAD RULES FROM AD SERVER CALL FLOW 21
B.7 AD SERVER PUSHING MOBAD RULES TO AD ENGINE CALL FLOW 22
B.8 AD ENGINE EVENT NOTIFICATION CALL FLOW 22
B.9 SP APP AD REQUEST CALL FLOW 23
B.10 SP APP REPORT AD METRICS DATA CALL FLOW 24
B.11 INFORMATIVE EXAMPLE: END-TO-END HIGH LEVEL FLOW INITIATED FROM A DEVICE 24
B.12 INFORMATIVE EXAMPLE: END-TO-END HIGH LEVEL FLOW INITIATED FROM AN SP APP 26
B.13 INFORMATIVE EXAMPLE: END-TO-END BROADCAST AD DELIVERY FLOW 27
B.14 AD FORWARDING THROUGH SERVICE PROVIDER CALL FLOW 29
APPENDIX C. CONTEXTUALISATION AND PERSONALISATION IMPLEMENTATION CHOICES (INFORMATIVE) 30
C.1 INFORMATIVE REFERENCE POINTS 31
C.1.1. CP-Network 31
C.1.2. CP-Device 31
C.2 OMA PRESENCE 31
C.3 OMA LOCATION 31
C.4 OMA XML DOCUMENT MANAGEMENT (XDM) 31
C.5 OMA DPE (DEVICE PROFILE EVOLUTION) 32

Figures

Figure 1: MobAd Enabler architecture 10
Figure 2: Ad App request Ads call flow 17

Figure 3: Ad App report Ad Metrics data call flow....18
Figure 4: Ad Engine Ad Request call flow....19
Figure 5: Push Ad Delivery to the Ad Engine Call Flow....20
Figure 6: Ad Engine report Ad Metrics data call flow....21
Figure 7: Ad Engine requesting MobAd Rules from Ad Server....21
Figure 8: Ad Server pushing MobAd Rules to Ad Engine....22
Figure 9: Ad Engine event notification call flow....22
Figure 10: SP App Ad Request Call Flow....23
Figure 11: SP App Report Ad Metrics Data Call Flow....24
Figure 12: Informative example: end-to-end flow initiated from a device for obtaining Ads....25
Figure 13: Informative example: end-to-end flow initiated from an SP App for obtaining Ads....27
Figure 14: Informative example: broadcast-based Ad(s) Delivery....28
Figure 15: Ad Forwarding Call Flow....29
Figure 16: MobAd architecture diagram enhanced with C&PR....30

1. Scope (Informative)

This document defines the architecture of Mobile Advertising (MobAd) Enabler. This architecture is based on the requirements listed in the MobAd Requirement Document [OMA-MOBAD-RD].

The objective of the MobAd Enabler architecture is to define network-side and device-side MobAd Enabler components, the interface(s) between them and the interface(s) exposed by those components to some entities that rely on MobAd Enabler (i.e. device-side Ad Apps as well as network-side SP Apps). The needs of external entities such as Content Providers, Advertisers, etc are not considered when defining interfaces for MobAd Enabler components, because those external entities are out of scope of MobAd AD and TS. The MobAd Enabler architecture is designed to support the core functionalities related to mobile advertising, in particular Ad Selection, Ad Delivery and Ad Metrics data handling.

2. References

2.1 Normative References

[OMA-ARCH-BEST-PRACTICES] "Architecture Best Practices", Open Mobile Alliance™, OMA-ORG-Architecture_Best_Practices-V1_4-20081202-A, URL:http://www.openmobilealliance.org/

[OMA-MOBAD-RD] "Mobile Advertising Requirements", Open Mobile Alliance™, OMA-RD-Mobile_Advertising-V1_0, URL:http://www.openmobilealliance.org/

[OSE] "OMA Service Environment", Open Mobile Alliance™, OMA-AD-Service_Environment-V1_0_4-20070201-A, URL:http://www.openmobilealliance.org/

[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, R.Fielding et al, June 1999, URL:http://www.ietf.org/rfc/rfc2616.txt

2.2 Informative References

[OMA-AD-DPE] "Device Profile Evolution Architecture", Open Mobile Alliance™, OMA-AD-DPE-V1_0, URL:http://www.openmobilealliance.org/

[OMA-BCAST-AD] "Mobile Broadcast Services Architecture", Open Mobile Alliance™, OMA-AD-BCAST_Services-V1_0, URL:http://www.openmobilealliance.org/

[OMA-DCD-AD] "Dynamic Content Delivery Architecture", Open Mobile Alliance™, OMA-AD-DCD-V1_0-20081223-C, URL:http://www.openmobilealliance.org/

[OMADICT] "Dictionary for OMA Specifications", Version 2.7, Open Mobile Alliance™, OMA-ORG-Dictionary-V2.7, URL:http://www.openmobilealliance.org/

[OMA-MLS-AD] "Mobile Location Service Architecture", Open Mobile Alliance™, OMA-AD-MLS-V1_2-20080627-C, URL:http://www.openmobilealliance.org/

[OMA-PRS-SIMPLE-AD] "Presence SIMPLE Architecture", Open Mobile Alliance™, OMA-AD-Presence_SIMPLE-V2_0, URL:http://www.openmobilealliance.org/

[OMA-XDM-AD] "XML Document Management Architecture", Open Mobile Alliance™, OMA-AD-XDM-V2_1, 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”, "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.

This document follows the conventions in the architecture best practices document [OMA-ARCH-BEST-PRACTICES].

For the figures representing architecture diagrams:

- Interfaces are depicted with an arrow line;
- Reference points are depicted with a non-arrow line;
- Interfaces or reference points are depicted with a solid line;
- Components within the scope of MobAd Enabler are depicted with a solid border line;
- Entities external to the MobAd Enabler are depicted with a dashed border line;
- Mandatory components or entities are depicted with a white filling;
- Optional components or entities are depicted with a grey filling.

For the figures representing call flows:

- Components within the scope of MobAd Enabler are depicted with a solid border line;
- Entities external to the MobAd Enabler are depicted with a dashed border line;
- Mandatory components or entities are depicted with a white filling;
- Optional components or entities are depicted with a grey filling;
- Steps within the scope of the MobAd Enabler are depicted with a solid arrow line,
- Steps out of scope of the MobAd Enabler are depicted with a dashed arrow line.

3.2 Definitions

Ad App	An application running on the Device which interacts with the Ad Engine in order to present Ad(s) to the user
Ad Campaign	See definition for Campaign.
Ad Channel	See definition in [OMA-MOBAD-RD].
Ad Engine	The Ad Engine is an application implementing MobAd Enabler functions and running on the Device. The Ad Engine interacts with Ad App and Ad Server. See Section 5.3.1.2 for the details
Ad Impression	See definition for Impression.
Ad Metadata	See definition in [OMA-MOBAD-RD]. Examples of usage are targeted audience, capping rules, Ad expiration, etc.
Ad Metrics	See definition in [OMA-MOBAD-RD].
Ad Selection	See definition in [OMA-MOBAD-RD].
Ad Server	A logical component that refers to MobAd Enabler Entities on the Network as per the definition in [OMA-MOBAD-RD]. See AD section 5.3.1.1 for details.
Advertisement	See definition in [OMA-MOBAD-RD].
Advertiser	See definition in [OMA-MOBAD-RD].

Campaign	See definition in [OMA-MOBAD-RD].
Content Metadata	See definition in [OMA-MOBAD-RD].
Content Provider	See definition in [OMA-MOBAD-RD].
Contextualisation	See definition in [OMA-MOBAD-RD]
Impression	See definition in [OMA-MOBAD-RD].
MobAd Rules	The MobAd Enabler service-related metadata (as opposed to Ad metadata), that define the Service Provider's rules related to e.g. Ad prefetching and caching policies.
Personalisation	See definition in [OMA-MOBAD-RD]
Principal	See definition in [OMADICT].
SP App	See definition in [OMA-MOBAD-RD].
User	See definition in [OMA-MOBAD-RD].
User Context	See definition in [OMA-MOBAD-RD].

3.3 Abbreviations

Ad	Advertisement
BCAST	Mobile Broadcast Services
C&PR	Contextualisation and Personalisation Resources
DCD	Dynamic Content Delivery
DPE	Device Profile Evolution
DRM	Digital Rights Management
ID	Identifier
OMA	Open Mobile Alliance
MLS	Mobile Location Service
MobAd	Mobile Advertising
MMS	Multimedia Messaging Service
SMSC	Short Message Service Centre
SP	Service Provider
URL	Universal Resource Locator
XDM	XML Document Management

4. Introduction (Informative)

This document defines the architecture of the MobAd Enabler based on the MobAd requirements document [OMA-MOBAD-RD].

This Architecture Document defines functional components interfaces and flows related to (list is non-exhaustive):

- Ad selection
- Contextualisation and Personalisation of Ad
- Ad App & SP App support
- Interactive Ads (e.g. click to pay, click to call, click to forward, etc)
- Recording and collection of Ad Metrics data
- Delivery of Ads

This document describes:

a. Functions of the Ad Server
b. Functions of the Ad Engine
c. Interfaces between Ad Engine and Ad Ap
d. Interfaces between Ad Server and Ad Engine.
e. Interfaces between Ad Server and SP App

The MobAd Enabler defines interfaces exposed by entities that are part of the Enabler (i.e.: Ad Server and Ad Engine). The MobAd enabler Architecture Document describes only MobAd intrinsic functional components and related interfaces. Interfaces to Advertiser and Content Provider are out of scope.

The MobAd Enabler does not describe interfaces to entities located outside of the Service Provider environment or the terminal. Ad rendering and definition of Ad formats are out of scope.

4.1 Version 1.0

The MobAd Architecture Document addresses the requirements targeted for this release that can be solved by architecture design. Some features may require architecture document updates in later releases, either because requirements have been deferred to a future release (e.g. in the case of Content Scanning) or because additional use cases or detailed requirements may be needed (e.g. support of Mobile Advertising in roaming scenarios) in order to assess whether the current architecture is adequate.

Please refer to [OMA-MOBAD-RD] for those features that are identified as future releases.

5. Architectural Model

5.1 Dependencies

The MobAd Enabler technical specifications are dependent on the following technologies:

- The HTTP 1.1 [RFC2616] as the default mandatory protocol for MobAd-2 and MobAd-3 interfaces.
- The DCD Enabler for the Ad delivery as described in [OMA-DCD-AD]. DCD is one of the optionally supported delivery mechanisms for which MobAd will describe the adaptation at TS stage for Delv-1 interface.
- The BCAST Enabler for Ad delivery as described in [OMA-BCAST-AD]. BCAST is one of the optionally supported delivery mechanisms for which MobAd will describe the adaptation at TS stage for Delv-1 interface.

5.2 Architectural Diagram

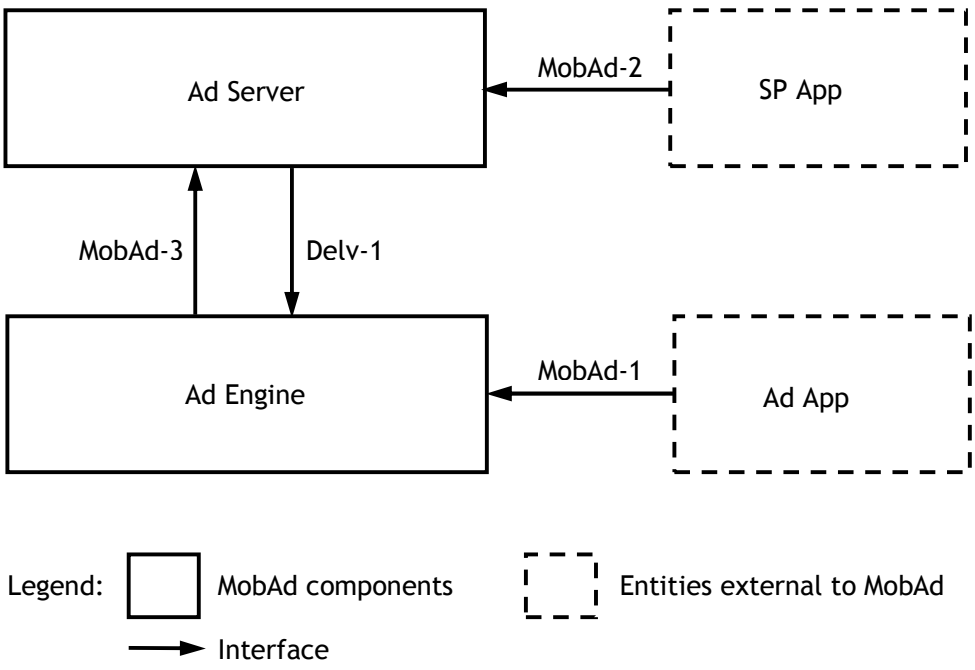

Figure 1: MobAd Enabler architecture

The MobAd Enabler consists of mandatory components (Ad Server and Ad Engine) and interfaces exposed by those components. All other depicted components and interfaces are not specified in this document, but are shown for a better understanding of the interactions with the MobAd Enabler (C&PR).

Both Ad Server and Ad Engine are MobAd Enabler mandatory functional components. There are two supported models:
-Ad App/Ad Engine/Ad Server model

-SP App/Ad Server model

5.3 Functional Components and Interfaces/reference points definition

5.3.1 MobAd Enabler functional Components (Normative)

5.3.1.1 Ad Server

The Ad Server (see section 3.2 Definitions) is a MobAd enabler component resident in the network (outside the device) that performs actions grouped in the following high-level functions:

- Ad selection function
- Ad delivery function
- Ad Metrics data handling function
- User / service data management function

Note: the grouping of the high level functions above is an illustrative example and not normative.

5.3.1.1.1 Ad Selection Function

The Ad selection function selects the most appropriate Ad(s) /Ad Campaign(s) primarily using:

- Contextualisation and Personalisation
- Ad Metadata
- Applicable MobAd Rules

5.3.1.1.2 Ad Delivery Function

The Ad delivery function delivers the following to Ad Engines and SP Apps:

- Ad(s)/Ad Campaign(s) with Ad Metadata
- Reference(s)to Ad(s)/Ad Campaign(s) (e.g. via URL) with Ad Metadata
- An indicator for no suitable result

The delivery of Ad(s)/Ad Campaigns can be done via:

- Pull: responses over a unicast channel to Ad Engine and/or SP App requests. This is mandatory to support.
- Push: over a unicast channel to the Ad Engine. This is optional to support.
- Broadcast: over a broadcast channel to the Ad Engine. This is optional to support.

5.3.1.1.3 Ad Metrics Data Handling Function

The Ad Metrics data handling function consists of primarily the following sub-functions:

- Collect Ad Metrics data about Ad Impression(s) and user's actions with Ad(s) (e.g. clicking after an Ad is presented). This information is collected from SP App and Ad Engine.
- Augment collected Ad Metrics data with data known to the Ad Server (such as sources IDs, time-stamp, context etc).
- Associate Ad Metrics data with specific Ad Campaigns if they are part of any Ad Campaign.
- Process Ad Metrics data, and all other related collected data into a consolidated report, that may be used, e.g.:

- For billing the Advertiser (out of scope)
- As additional input for future Ad selection.

Note: it is up to the implementation and the SP policy whether and if so how the verification of reported Ad Metrics data is done.

5.3.1.1.4 User / Service Data Management Function

The user / service data management function consists of primarily the following sub-functions:

- Handle user's Contextualisation and Personalisation data.
- Manage MobAd Rules.
- Manage (e.g. create) and select user groups for targeting purpose.
- Handle Ad Channels.
- Manage Ad(s), Ad(s) ID(s) and Ad(s) Metadata.

5.3.1.2 Ad Engine

The Ad Engine (see section 3.2 Definitions) is a MobAd Enabler component resident on the device that performs actions grouped in the following high-level functions:

- Ad acquisition and delivery function
- Ad selection function
- Ad Metrics data handling function
- User / service / device data handling function

Note: the grouping of the high level functions above is an illustrative example and not normative.

5.3.1.2.1 Ad Acquisition and Delivery Function

The Ad acquisition and delivery function consists of primarily the following sub-functions:

- Receive Ad Requests from Ad Apps.
- Obtain and/or receive either of the following from the Ad Server:
  - Ad(s)/Ad Campaign(s) with Ad Metadata
  - Reference(s) to Ad(s)/Ad Campaign(s) (e.g. via URL) with Ad Metadata
  - An indicator for no suitable result
- Cache Ad(s) received from the Ad Server.
- Update (e.g. delete) Ad(s) previously received from the Ad Server.
- Deliver the selected result to the Ad Apps

The acquiring of Ad(s)/Ad Campaigns can be done via:

  - Pull: Ad Engine issues requests to the Ad Server and receives Ad(s) from the Ad Server over a unicast channel. This is mandatory to support.
  - Push: Ad Server delivers Ad(s) to the Ad Engine over a unicast channel. This is optional to support.

- Broadcast: Ad Server delivers Ad(s) and/or Ad metadata to the Ad Engine over a broadcast channel. This is optional to support.

5.3.1.2.2 Ad Selection Function

The Ad selection function selects the most appropriate Ad(s) primarily using the following:

- Ad selection criteria including MobAd Rules and available Ad Metadata
- data provided from Ad Apps as input in the request for Ad(s)
- data from Contextualisation and Personalisation Resources (C&PR)

The Ad Engine's cache is the primary source of the Ad(s). Under certain conditions, the Ad Engine can request Ad(s) from the Ad Server.

5.3.1.2.3 Ad Metrics Data Handling Function

The Ad Metrics data handling function consists of primarily the following sub-functions:

- Receive Ad Metrics data from Ad App.
- Augment Ad Metrics data received from Ad App with other metrics-related information known by the Ad Engine.
- Attempt to identify if the Ad Metrics data are inaccurate or not, and handle them accordingly based on the Service Provider policy, e.g.:
  - Either discard inaccurate Ad Metrics data prior to sending the Ad Metrics data report to the Ad Server.
  - Or include this inaccurate Ad Metrics data in the report.
- Report the Ad Metrics data to the Ad Server, optionally combining multiple reports into one report.

5.3.1.2.4 User/Service/Device Data Handling Function

The user / service / device data handling function consists of primarily the following sub-functions:

- Gather and process user's Contextualisation and Personalisation data.
- Optionally monitor and process device static and/or dynamic status (e.g. device resource threshold).
- Apply MobAd Rules.
- Manage cached Ad(s), Ad(s) ID(s) and Ad(s) Metadata.

5.3.2 Entities External to the MobAd Enabler (Informative)

5.3.2.1 SP App

The SP App is an external entity that requests and receives Ad(s)/Ad Campaign(s) from Ad Server, and embeds them in content that is provided to the user. SP App records Ad Metrics data related to the Ad(s) and reports Ad Metrics data to the Ad Server.

Examples of SP App can be Ad-aware web portals, MMS Relay / Server, SMSC, gaming server that uses Ad Server functionality exposed by the Ad Server interface.

5.3.2.2 Ad App

The Ad App is an external entity resident on the device that requests and receives Ad(s) from Ad Engine, and presents them to the user. Ad App also report Ad Metrics data to the Ad Engine.

Examples of Ad App can be Ad-aware messaging client, web browser, gaming client that uses Ad Engine functionality exposed by the Ad Engine interface.

5.3.3 Interfaces (Normative)

Note: it is for further study in the next major release whether additional interfaces are needed and if so how to request / create / modify / delete / retrieve operations for data needed by the MobAd Enabler (e.g. MobAd user preferences).

5.3.3.1 MobAd Enabler Interfaces

5.3.3.1.1 MobAd-1

MobAd-1 is an interface exposed by the Ad Engine to the Ad App. The Ad App uses this interface to request and obtain Ads and their associated Ads identifiers from the Ad Engine, as well as to report Ad Metrics data to the Ad Engine, accompanied by the associated Ads identifiers.

5.3.3.1.2 MobAd-2

MobAd-2 is an interface exposed by the Ad Server to the SP App. The SP App uses this interface to request and obtain Ad(s), reference(s) to Ad(s), associated Ad(s) identifiers and possibly additional information (to be determined in the TS stage), as well as to report Ad Metrics data, accompanied by the associated Ad(s) ID(s).

This interface can also be used by the Ad Server to inform the SP App that some Ad(s) (stored locally by the SP App) are supposed to be deleted. This can be achieved either by attaching Ad deletion information to an Ad Server response following an SP App request for Ad(s), or by returning such information in response to a specific request for updates on Ad's validity.

The message pattern supported by MobAd-2 is a request issued by SP App towards Ad Server followed by a synchronous response from Ad Server to SP App.

Note: it is For Further Study in a next major release whether and if so how to support one or more subsequent asynchronous additional responses from Ad Server to SP App.

5.3.3.1.3 MobAd-3

MobAd-3 is an interface exposed by the Ad Server to the Ad Engine. The Ad Engine uses this interface to request and obtain Ad(s), reference(s) to Ad(s), their associated Ad(s) ID(s) and Ad Metadata from the Ad Server, as well as to report Ad Metrics data to the Ad Server, accompanied by the associated Ad(s) ID(s).

This interface can also be used by the Ad Server to inform the Ad Engine that some Ad(s) (stored locally by the Ad Engine) are supposed to be deleted. This can be achieved either by attaching Ad deletion information to an Ad Server response following an Ad Engine request for Ad(s), or by returning such information in response to a specific request for updates on Ad's validity.

This interface may also be used by the Ad Engine to retrieve MobAd Rules and to provide notification to the Ad Server.

5.3.3.1.4 Delv-1

Delv-1 is an optional interface exposed by the Ad Engine. The Ad Engine receives Ad(s) and/or Ad Metadata over this interface from the Ad Server via underlying push and/or broadcast delivery mechanisms. The Ad Server uses this interface to push either Ad(s) or notification that Ad(s) are available for retrieval.

The Ad Server may also use this interface to provide service notification to the Ad Engine (e.g. information that SP caching and pre-fetching policies have been dynamically updated; An Ad or campaign needs to be cancelled ASAP, MobAd Rules have changed, request for Ad Metrics data reporting, etc).

The DELV-1 is a MobAd interface which may optionally be realized/implemented using other OMA enablers or protocols to which adaptation will be defined in adaptation specification.

5.4 Security Considerations

The security considerations described in this section apply to any MobAd Enabler implementation, and these considerations may result in different deployment models. Any particular security mechanism that proves to be intrinsic to the MobAd Enabler specification will be addressed in the MobAd Technical Specifications.

The SP deploying the MobAd Enabler implementation needs to ensure that all entities requiring access to information provided via the interfaces exposed by the functional components of the MobAd Enabler are subject to the following security considerations:

- The entities (such as SP App, Ad App) which report Ad Metrics data should be authenticated and authorised, but the chosen authentication/authorisation mechanisms are out-of-scope for the MobAd Enabler specification.
    - SP App is considered to be a trusted application deployed by the Service Provider in its Service Provider domain. Whether the SP requires special authentication/authorisation mechanisms between SP App and Ad Server is an implementation and deployment consideration, subject to specific Service Provider security policies.
    - Ad App is considered to be a trusted application deployed on a device belonging to a user. Whether the SP requires special authentication/authorisation mechanisms on the device between the Ad App and the Ad Engine is an implementation and deployment consideration, subject to specific Service Provider security policies.
- Specific recommendations for transport security, authentication/authorisation, data encryption, etc may be required for communication between Ad Server and Ad Engine (components that exchange data over-the-air) and can be considered during the Technical Specification phase.
- The MobAd Enabler implementation shall consider protecting the user data acquired from C&PR, by applying security mechanisms consistent with the C&PR specifications and the applicable Service Provider security policies (e.g. including transport security, user data privacy, data encryption, etc).
- The MobAd Enabler implementation shall consider security mechanisms supporting the anonymisation of personal identification information in the Ad Metrics data collected and consolidated report (e.g. user name and contact information), and/or the encryption of such personal information.
- The MobAd Enabler is agnostic of any protection mechanism, e.g. OMA DRM or others, used to protect the Ad(s).

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-AD-Mobile_Advertising-V1_0	20 Mar 2012	Status changed to Approved by TP: OMA-TP-2012-0116-INP_MobAd_V1_0_ERP_for_Final_Approval

Appendix B. Flows (informative)

This section contains detailed call flows, and high-level end-to-end call flows.

- The detailed call flows define the basic reusable patterns of communication between MobAd entities. In MobAd lifecycle, these patterns can appear in different combinations, in different time ordering. The detailed call flows are the "building blocks" of MobAd behaviour.
- The high-level end-to-end call flows show some examples of how the basic patterns can be applied to yield important end-to-end scenarios. The high-level end-to-end call flows are "constructed" from the detailed call flows.

The purpose of a detailed call flow is:

- to illustrate triggering conditions for a basic pattern;
- to illustrate sequence of steps;
- to illustrate precise meaning of each step;
- to illustrate step conditions (for optional steps);

The purpose of a high-level end-to-end call flow is:

- to provide an illustrative example for an important extended scenario;
- to show relative sequence of basic patterns in this scenario.

The high-level end-to-end call flows use steps defined by the corresponding detailed call flows.

The detailed call flows are described in sections from B.1 to B.9.

The high-level end-to-end call flows are described in sections from B.10 to B.13.

B.1 Ad App Request Ads Call Flow

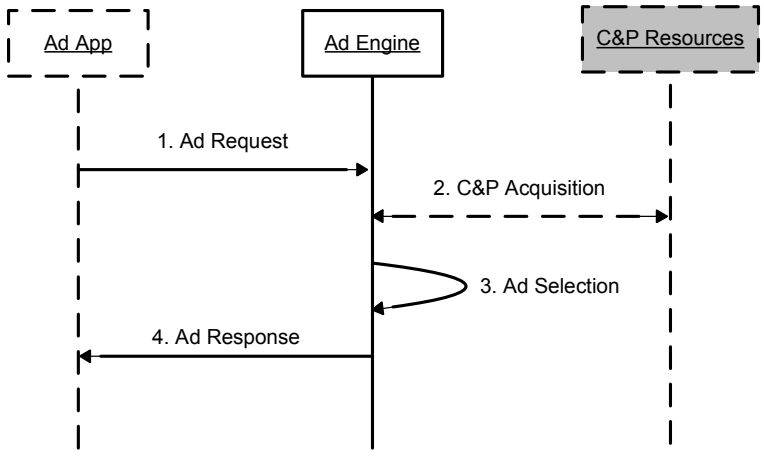

Figure 2: Ad App request Ads call flow

This call flow is triggered by Ad App's internal execution logic.

1. Ad App requests Ads from the Ad Engine. The request may contain some contextual information such as:
   - Ad App ID

- Ad presentation format
- Ad size and positioning
- Content metadata information available to the Ad App (e.g. webpage meta tags indicating user is browsing an automobile web site and would be interested in car Ads, or the topic in case of an Ad-enabled game (soccer))
- An indication of the urgency of the request, e.g. time-to-live (TTL)

2. The Ad Engine may obtain Contextualisation and/or Personalisation information to facilitate the subsequent step 3.
3. The Ad Engine executes the Ad selection logic that analyses Ads stored in the Ad Engine's cache. This Ad selection may result in one, several or no Ads. If needed, the Ad Engine may communicate with the Ad Server, for the purpose of obtaining more Ads (as described in Appendix B.3)
4. The Ad Engine returns the results of the Ad selection. A selected Ad is accompanied by a unique Ad ID.

B.2 Ad App Report Ad Metrics Data Call Flow

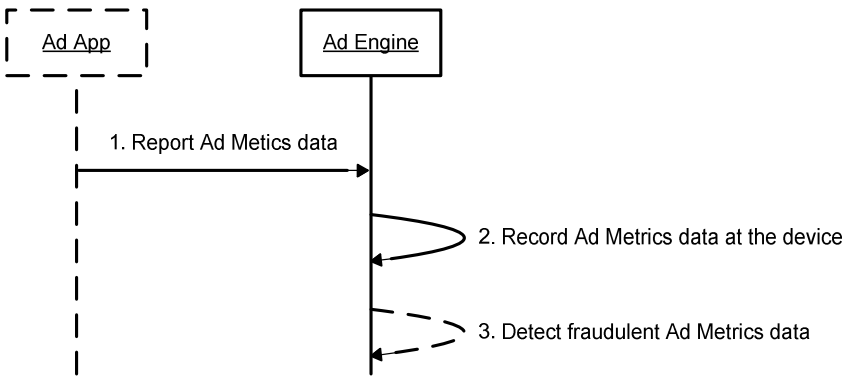

Figure 3: Ad App report Ad Metrics data call flow

This call flow is triggered by Ad App's Ad Metrics data reporting, e.g. after user interaction.

1. Ad App reports Ad Metrics data to the Ad Engine, such as:
    - Ad Impressions
    - Ad related user actions
    - Unique Ad ID
    - Ad App ID
2. Ad Engine records the Ad Metrics data. This data can be further used to refine information for future Ad selection process. Ad Engine complements the Ad Metrics data with information known to Ad Engine (e.g. the user's presence state or location at the time they viewed or interacted with an Ad).
3. Ad Engine attempts to identify if the Ad Metrics data are inaccurate or not, and mark it accordingly. If Ad Metrics data are detected as inaccurate, Ad Engine will:
    - Either discard these marked inaccurate Ad Metrics data prior to sending the Ad Metrics data report to the Ad Server
    - Or include these marked inaccurate Ad Metrics data in the report

B.3 Ad Engine Ad Request Call Flow

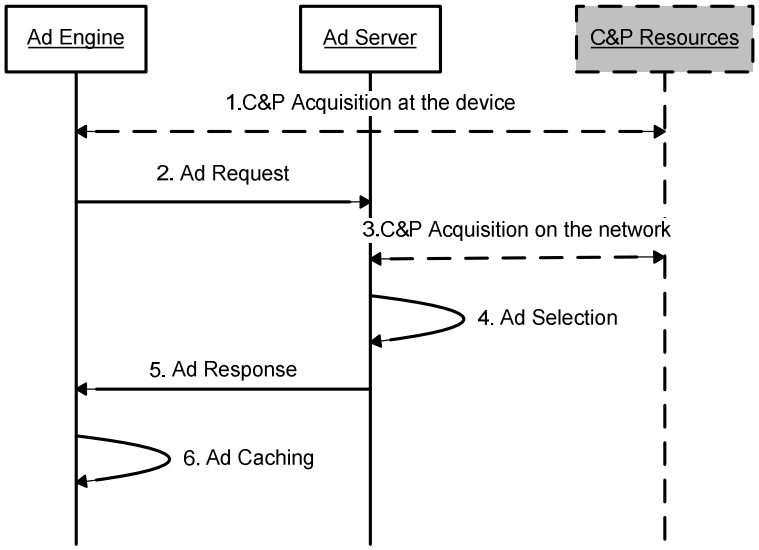

Figure 4: Ad Engine Ad Request call flow

This call flow is triggered by a stimulus-based event such as a periodic timer in the Ad Engine.

1. This step is only applicable if any Contextualisation and/or Personalisation information which resides on the terminal is being used. The Ad Engine obtains personalisation and/or contextualisation information located on the terminal for the purpose of obtaining the appropriate Ad(s) for the end user.
2. Ad Engine requests Ad(s) from the Ad Server. The parameters may be derived from information obtained in Step 1. Or Ad Engine may also request specific Ads.
3. This step is only applicable if any Contextualisation and/or Personalisation information which resides on the network is being used. The Ad Server obtains Contextualisation and/or Personalisation information located on the network for the purpose of obtaining the appropriate Ad(s) for the end user. Information obtained at Step 1 and Step 3 may complement each other.
4. The Ad Server executes the Ad selection logic, which may result in one, several or no Ads.
5. If the Ad selection in step 4 resulted in one or more Ads, the Ad Server delivers the Ad(s) to the Ad Engine. The returned Ad(s) may be selected based on a best-effort match, or represent a default Ad(s) - the rule for such Ad selection is outside the scope of this call flow. If the Ad selection in step 4 resulted in null match, the Ad response may contains no Ads. A unique Ad ID associated with each selected Ad will be returned. In the Ad Response the Ad Server delivers Ads and/or associated Ad Metadata (which may include rules and instructions related to usage of Ads which are being provided).
6. Ad Engine caches and may filter the received Ad(s), if any, for anticipated future delivery to Ad App(s) located on the same device.

B.4 Push Ad Delivery to the Ad Engine Call Flow

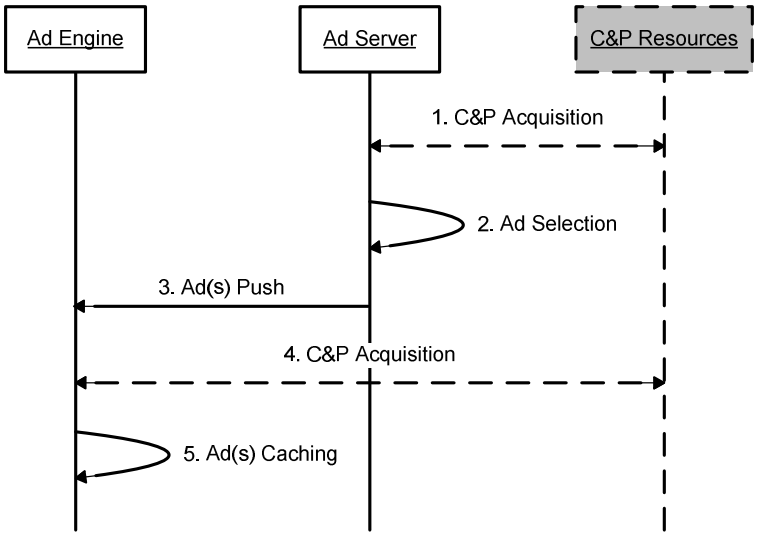

Figure 5: Push Ad Delivery to the Ad Engine Call Flow

This call flow is triggered when a Service Provider has identified a target audience for an Ad Campaign and wishes to deliver the Ad(s) in that Campaign to each targeted user as soon as possible. In this situation, the call flow below is repeated for each user in the Campaign's target audience.

1. If necessary, the Ad Server obtains Contextualisation and/or Personalisation information about the user to whom the Ad Campaign will be sent. This information could be used to select all or a subset of the Ads in the Campaign that best match for the user and their device.
2. The Ad Server performs the Ad Selection process, e.g. matching the C&P information against the Ad Metadata for the Ads in the Ad Campaign.
3. The Ad Server sends the Ad(s) and/or Ad Metadata identified in step 2 to the Ad Engine using a push delivery mechanism. Alternatively, the Ad Server may send only an Ad notification that includes a link that the Ad Engine can use to retrieve the Ad(s) and/or Ad Metadata.
4. If necessary, the Ad Engine obtains Contextualisation and/or Personalisation information relevant to handling the Ad(s).
5. The Ad Engine filters and caches the received Ad(s) for anticipated future delivery to the Ad App(s) located on the same device. If a link to the Ad(s) was sent in step 3, the Ad Engine will download the Ad(s) and/or Ad Metadata.

B.5 Ad Engine Report Ad Metrics Data Call Flow

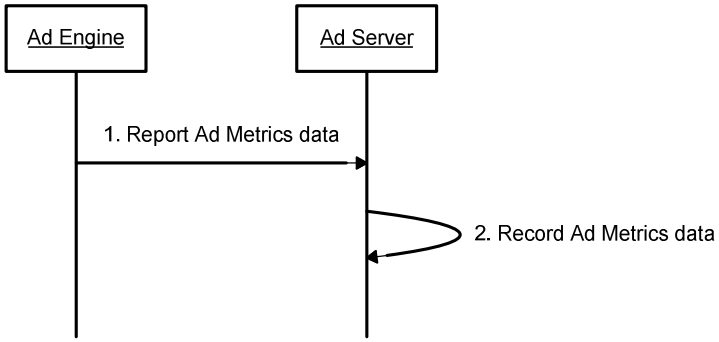

Figure 6: Ad Engine report Ad Metrics data call flow

This call flow is triggered by a stimulus-based event such as a periodic timer in the Ad Engine or when the device is turned on.

1. Ad Engine reports Ad Metrics data related to impressions and interactions with Ads (and other related information) to the Ad Server. Ad Engine Ad Metrics report may contain information about multiple Ads as well as multiple Ad impressions and interactions obtained from different Ad Apps.
2. Ad Server receives the report, and records the obtained Ad Metrics data. Ad Server processes the data (e.g. for providing consolidated reports, for refining information for future Ad selection process). Ad Server may also analyse the obtained Ad Metrics data to identify inaccurate Ad Metrics data and act accordingly.

B.6 Ad Engine Requesting MobAd Rules from Ad Server Call Flow

The following flow relates to the optional functionality of the Ad Server to provide the Ad Engine with MobAd Rules that will be used by the Ad Engine to perform its functions. This flow shows the scenario where the Ad Engine requests MobAd Rules from the Ad Server.

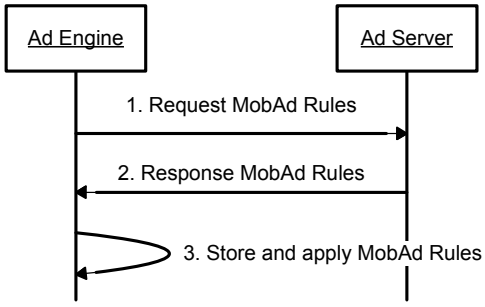

Figure 7: Ad Engine requesting MobAd Rules from Ad Server

This call flow is triggered by the Ad Engine' internal execution logic, for example when the Ad Engine starts up, when the User's Device establishes a connection to the network, or periodically according to SP's policy.

1. Ad Engine requests MobAd Rules or their updates from the Ad Server.

2. Ad Server responds the MobAd Rules or their updates to the Ad Engine.
3. Ad Engine stores and applies the received MobAd Rules.

B.7 Ad Server Pushing MobAd Rules to Ad Engine Call Flow

The following flow relates to the optional functionality of the Ad Server to provide the Ad Engine with MobAd Rules that will be used by the Ad Engine to perform its functions. This flow shows the scenario where the Ad Server pushes MobAd Rules to the Ad Engine.

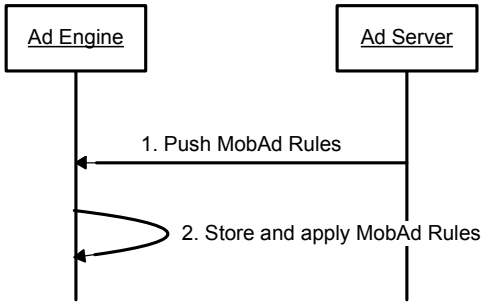

Figure 8: Ad Server pushing MobAd Rules to Ad Engine

This call flow is triggered when a Service Provider has identified need to deliver MobAd Rules or their updates to a certain set of Users.

1. Ad Server pushes MobAd Rules or their updates of them to the Ad Engine.
2. Ad Engine stores and applies the received MobAd Rules.

B.8 Ad Engine Event Notification Call Flow

The following functionality and call flow are optional and could be addressed in a future release of the TS. They relate to the Ad Engine notifying the Ad Server, upon detection of some event(s). After notifying the Ad Server, Ad Engine may suspend and resume its functions, based on SP policy.

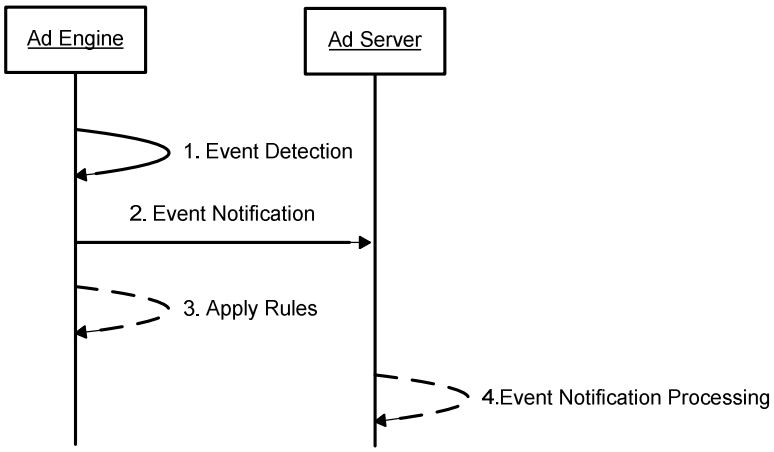

Figure 9: Ad Engine event notification call flow

This call flow is triggered by a number of specific events, which may happen at User Device. The set of events is defined by the Service Provider.

1. Ad Engine detects a given event happened such as device resources thresholds is reached e.g. battery life.

Note: The Ad Engine can be a listener to given device resources. This step is implementation specific.

2. Depending on the type of the event the Ad Engine may notify the Ad Server accordingly.

Note: the type of events and the need for a notification can be defined by or be based on SP rules and policies.

3. Depending on the type of detected events and/or rules applying to it (if any), the Ad Engine may suspend or resume part or all of its functions e.g.:
   - For a given period of time.
   - Until another event is detected.
4. Ad Server processes the notification and may change its behaviour based on that.

B.9 SP App Ad Request Call Flow

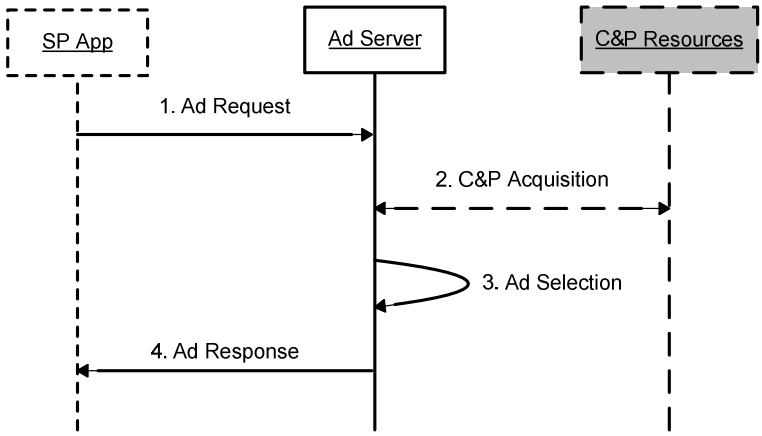

Figure 10: SP App Ad Request Call Flow

This call flow is triggered by SP App's internal execution logic.

1. SP App requests Ad(s) and / or Ad Metadata from the Ad Server.
   The SP App Ad Request from SP App contains at least an SP App ID parameter.

   The SP App may pass the following information in the Ad Request:

   a. Principal identification (e.g. single user or group of users)
      This information may be passed using different sets and/or constructs of identifiers, as to allow for flexibility and ease of integration. The identifiers can represent, among others:
      - User identifiers (anonymised or real, temporary or permanent, encrypted or plain), which can then in turn be used by the Ad Server to determine relevant user information and attributes, be it dynamic or static
      - User group identifiers

   b. Information identifying specific Ad(s) to be obtained (e.g. a specific Ad Campaign)

   c. Application and / or content context information (e.g. webpage is about cars, keywords of different types, application-specific user preference, etc.)

d. Desired Ad expiration time / date

e. Location information (e.g. location information that relates to the content to be sent)

f. Ad presentation information (e.g. information describing supported formats), types (e.g. Image, Video etc.), Width/Height, etc.

Note 1: information identifying specific Ad(s) may have been provided to the SP App through different means (e.g. out-of-scope notifications from Ad Server, out-of-scope provisioning into the SP App, etc)

2. The Ad Server can obtain Contextualisation and/or Personalisation information located on the network for the purpose of obtaining the appropriate Ad(s).
3. The Ad Server processes the Ad Request from the SP App and subsequently performs the Ad Selection. Ad Server processes the Ad Request parameters by using available information or may call external entities to support some of these parameters during the selection step. These steps are not illustrated in this flow.
4. The Ad Server shall provide a response to the SP App.

The Ad Response message contains:

a. Selected Ad(s) (or reference(s) to Ad(s)) and / or Ad Metadata according to the information provided within the request. If not all information can be satisfactorily used, the Ad Server can ignore or apply the information differently (larger geographic radius, etc.).The Ad Server can provide Default Ads (e.g. filler). If no available or applicable Ad(s) can be returned, the Ad Server may return an indication that no Ad is available or applicable.

b. A unique Ad ID associated with each selected Ad that will be used when reporting Ad Metrics data to identify the Ad that was used, assuming Ad(s) were returned.

c. Ad metadata (which may include rules and instructions related to usage of Ads which are being provided).

B.10 SP App Report Ad Metrics Data Call Flow

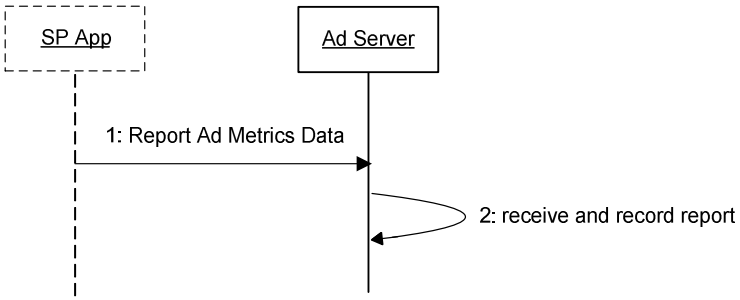

Figure 11: SP App Report Ad Metrics Data Call Flow

This call flow is triggered by SP App's internal execution logic.

1. SP App reports Ad Metrics data to the Ad Server. The report may contain information such as the SP App ID.
2. Ad Server receives the report, processes and records it.

B.11 Informative Example: End-to-End High Level Flow Initiated From A Device

The flow in figure 12 represents at a high level the exchanges between an Ad App, Ad Engine, Ad Server and the Contextualisation and Personalisation Resources, when requests for Ads are initiated from a device.

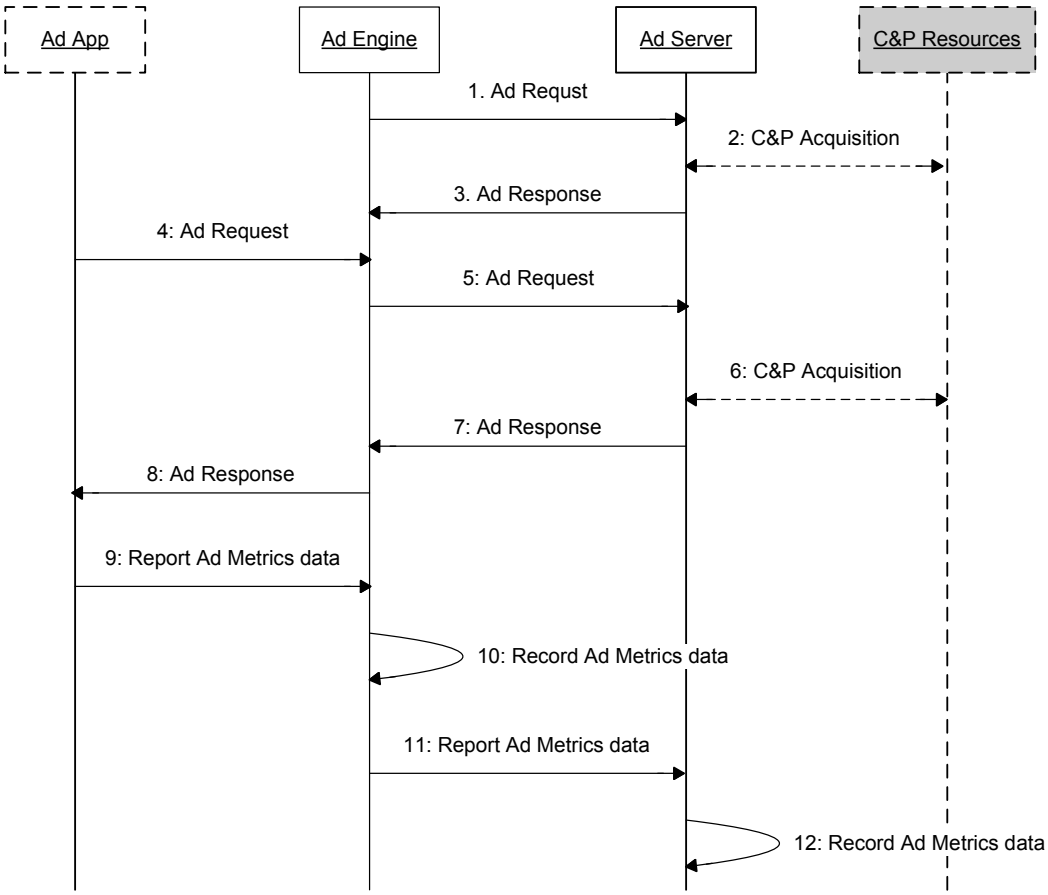

Figure 12: Informative example: end-to-end flow initiated from a device for obtaining Ads

The following steps represent one possible sequence of the main operations:

1. The Ad Engine requests Ad(s) from the Ad Server, in order to cache Ads in the device.
2. The Ad Server processes available personal and contextual data, in order to select the most appropriate Ad(s).
    a. The Ad Server may use internal resources.
    b. Optionally, the Ad Server uses external Contextualisation and Personalisation Resources.
3. The Ad Server usually returns one (or more) Ad(s) to the Ad Engine. The return usually includes Ad(s) or references to Ad(s), but if no Ad is applicable or available, the return could indicate that, or a default Ad could be returned instead.
4. An Ad App requests Ad(s) from the Ad Engine residing on the same device.
5. If needed, the Ad Engine may repeat the Ad Request operation. This may be needed for a refinement (e.g. for specific Ad formats, for specific Ad App context, etc) or because previous cached Ads expired or because of other reasons.
6. C&P Acquisition (see step 2) may be repeated in this step.

7. The Ad Server may repeat the Ad Response operation to return one (or more) Ad(s) to the Ad Engine. The return can include Ad(s) or reference(s) to Ad(s).

8. The Ad Engine provides to the Ad App the Ad(s) (or reference(s) to Ad(s)) it selected, for presentation to the user. If no Ad is available or applicable, the return could indicate that, or a default Ad could be returned instead.

9. The User's impressions/actions with an Ad (e.g. click-through) may be reported by Ad App to Ad Engine. Note that the Ad impressions and the User's impressions/actions can be recorded in the device through other means than, or in addition to an explicit report by Ad App (see next step).

10. The Ad Engine records Ad impression(s) and interaction(s). Ad Engine may analyse the obtained Ad Metrics data based on metrics rules, in order to identify inaccurate Ad Metrics data. Ad Engine reports Ad Metrics data from Ad App with information known to Ad Engine. In some cases the Ad App report is filtered out and contains only the Ad Engine report. This data can be further used to refine information for future Ad selection process.

11. The Ad Engine reports Ad Metrics data related to impressions and interactions with an Ad (and other related information) to the Ad Server. Ad Engine could also collect several Ad Metrics data (e.g. impressions, interactions) and report them at once.

12. The Ad Server processes the received Ad Metrics data (e.g. provide consolidated reports; refine information for future Ad selection process, etc). Ad Server may also analyse the obtained Ad Metrics data based on metrics rules, in order to identify inaccurate Ad Metrics data and filter them out.

B.12 Informative Example: End-to-End High Level Flow Initiated From An SP App

The flow in figure 13 represents at a high level the exchanges between an SP App, Ad Server and the Contextualisation and Personalisation Resources, when requests for Ads are initiated from an SP App.

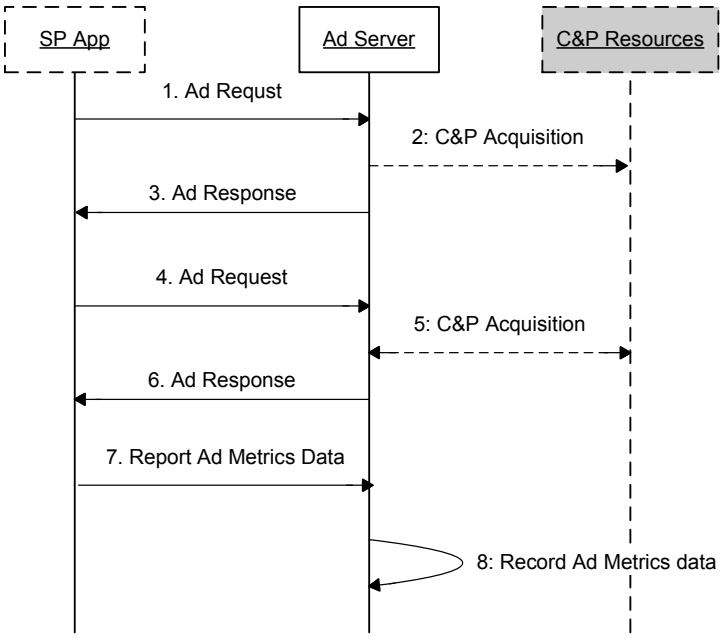

Figure 13: Informative example: end-to-end flow initiated from an SP App for obtaining Ads

The following steps represent one possible sequence of the main operations:

1. The SP App requests Ad(s) from the Ad Server.

2. The Ad Server processes available personal and contextual data, in order to select the most appropriate Ad(s).

    a. The Ad Server may use internal resources.

    b. Optionally, the Ad Server may use External Contextualisation and Personalisation Resources.

3. The Ad Server usually returns one (or more) Ad(s) to the SP App. The returned message usually includes Ad(s) or references to Ad(s), but if no Ad is applicable or available, the returned message could indicate that, or a default Ad could be returned instead.

4. If needed, the SP App may repeat the Ad Request operation. This may be needed for a refinement (e.g. for specific Ad formats, for Ads related to specific context expressed by SP App, etc) or because a previously obtained Ad expired, or for other reasons.

5. C&P Acquisition (see step 2) may be repeated in this step.

6. The Ad Server may repeat the Ad Response operation to return one (or more) Ad(s) to the SP App. The returned message can include Ad(s) or reference(s) to Ad(s).

7. The SP App reports Ad Metrics data related to impressions and interactions with an Ad (and other related information), as well as an identification of the SP App used to present the Ad and sends them to the Ad Server. SP App could also collect several Ad Metrics data (e.g. impressions, interactions) and report them at once.

8. The Ad Server processes the received Ad Metrics data (e.g. provide consolidated reports; refine information for future Ad selection process, etc). Ad Server may also analyse the obtained Ad Metrics data based on metrics rules, in order to identify inaccurate Ad Metrics data and filter them out.

B.13 Informative Example: End-to-end Broadcast Ad Delivery Flow

Broadcast delivery of Ad(s) is an optional feature of the MobAd Enabler.

This flow represents at a high level broadcast delivery of Ad(s) among an Ad App, Ad Engine, Ad Server and the Contextualisation and Personalisation Resources.

This scenario assumes that the users have subscribed to the broadcast service and their devices are already pre-configured to receive broadcast channels. To enable broadcast Ad delivery, the Ad Server arranges to deliver a stream or file of Ad(s) and Ad Metadata over a broadcast bearer (e.g. if DCD is used for Ad delivery, the Ad Server could use DCD Content Provider channel registration to define an Ad Channel). To receive the broadcasted Ad(s) and/or Ad Metadata or files, each device provides a set of parameters to the broadcast client on the device so that the device can “bind to” the Ad delivery process (e.g. for DCD, Channel Metadata would be supplied to the device’s DCD Client). The Ad delivery can be repeated periodically over the broadcast bearer to enable devices to receive the current set of Ads (e.g. when they power on and to obtain subsequent changes to the available set of Ads). Broadcast Ad delivery can also be used on a one-time basis for special events, e.g. to broadcast Ad(s) to everyone attending a sporting event at a stadium.

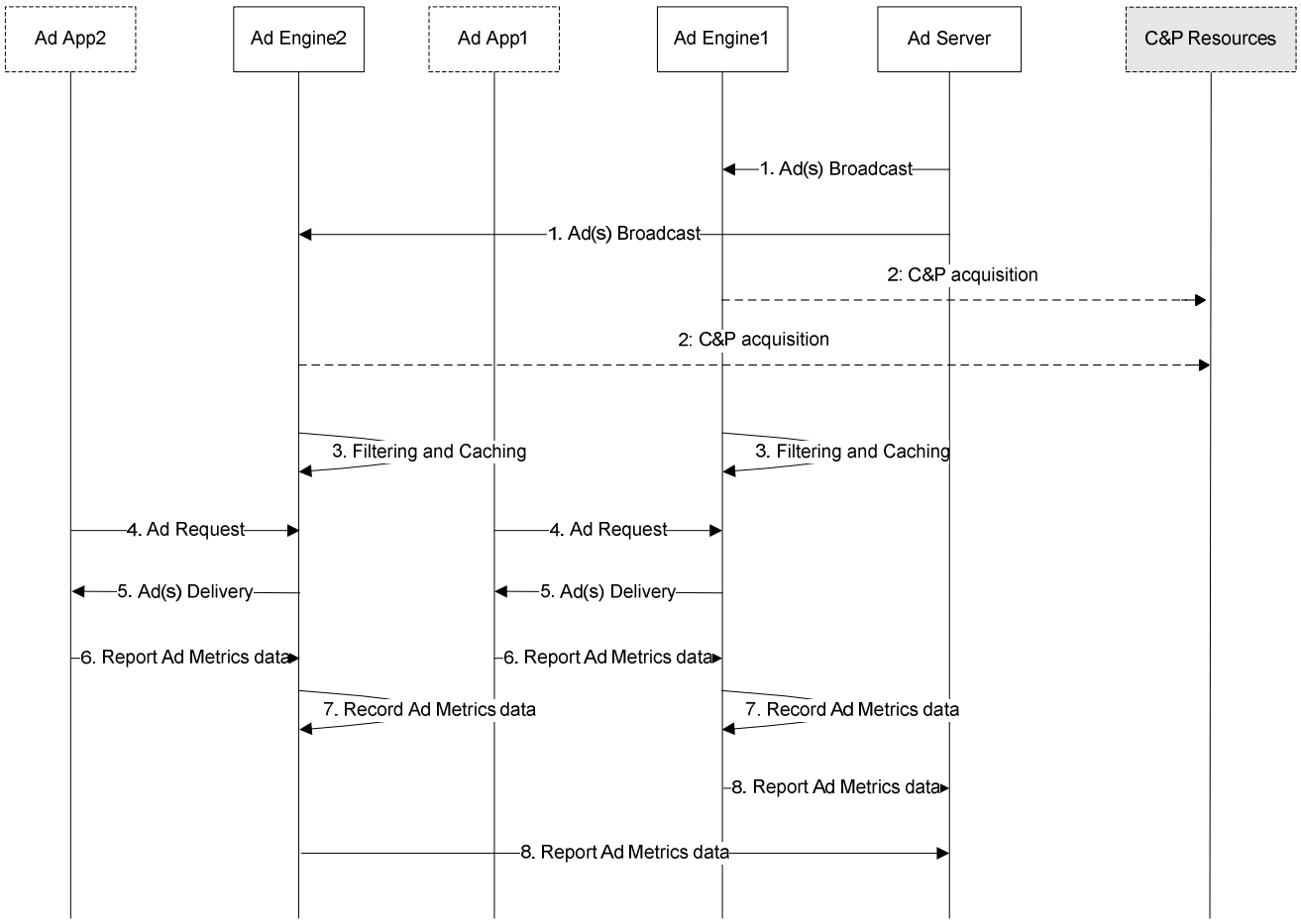

Figure 14: Informative example: broadcast-based Ad(s) Delivery

1. After the Ad Selection, the Ad Server broadcasts the Ad(s) and/or its Ad Metadata to the Ad Engine. This can be done for example by using the services of the OMA DCD enabler (which in turn may rely on OMA BCAST or 3GPP CBS). Alternatively, the Ad Server may broadcast only the Ad Metadata with reference to the Ad(s).
2. The Ad Engine may acquire the User Information and/or User Context from Contextualisation and Personalisation Resource to help filtering the Ad(s). This operation is an optional process and can be omitted.
3. The Ad Engine monitors the broadcast delivery of Ad(s) and / or Ads Metadata. The Ad Engine filters and caches only those matching the User Information and/or User Context. If only the Ads Metadata is passed in Step 1, the Ad Engine needs to acquire the Ad(s) from the Ad Server before caching. (NOTE: Steps 1-3 are repeated for each periodic transmission of the broadcast Ad delivery).
4. The Ad App requests Ad(s) from the Ad Engine.
5. The Ad Engine delivers the Ad(s) to the Ad App.
6. The Ad App report the Ad Metrics data related to the impression and/or interaction with Ad(s).
7. The Ad Engine records Ad Metrics data.
8. The Ad Engine reports Ad Metrics data to the Ad Server.

B.14 Ad Forwarding Through Service Provider Call Flow

This is an optional functionality.

This call flow addresses the mechanism of Ad forwarding which is useful for the purposes of e.g. viral advertising.

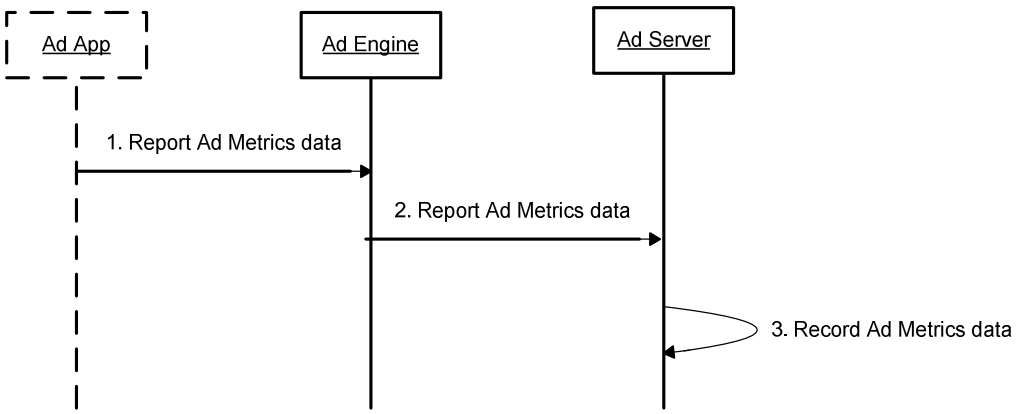

Figure 15: Ad Forwarding Call Flow

1. Following User clicking to forward the Ad to another user (out of scope), the Ad App reports to the Ad Engine either that it has forwarded the Ad or that it requests the Ad to be forwarded.
2. The Ad Engine reports this event to the Ad Server. This event is embedded in the Ad Metrics data.

   Note: This message may contain identification parameters that will be defined in TS.
3. The Ad Server receives and verifies the Ad Forward Event Metrics data and processes it appropriately based on Service Provider policy. This operation is the internal process of the Ad Server.

Appendix C. Contextualisation and Personalisation Implementation Choices (Informative)

Contextualisation and Personalisation Resources (C&PR) are entities external to the MobAd Enabler.

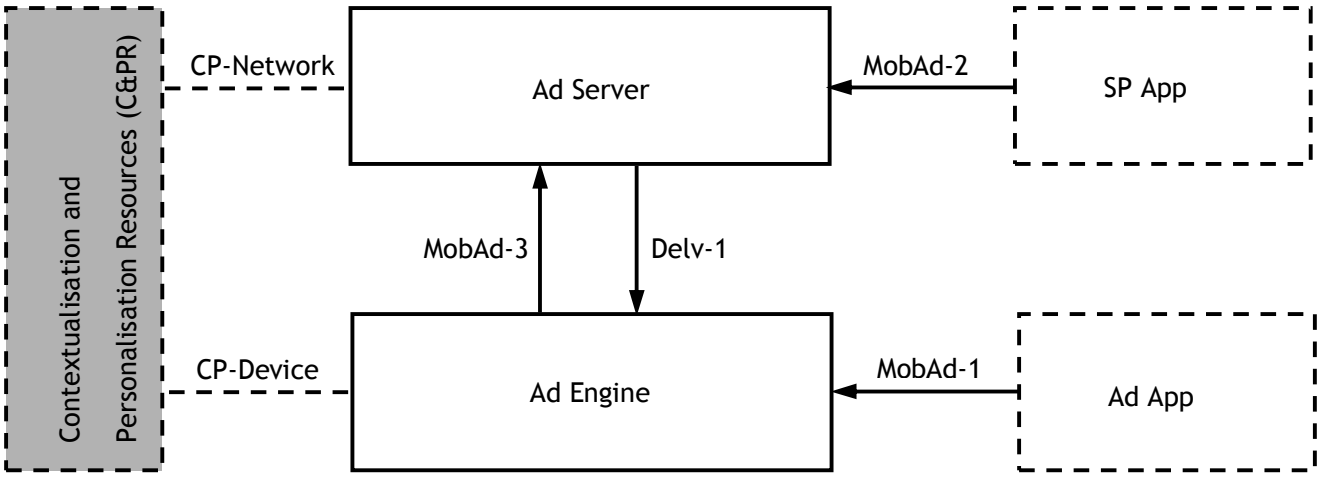

Figure 16: MobAd architecture diagram enhanced with C&PR

The C&PR may reside in either the Service Provider domain or on the Device. In a typical deployment, the C&PR can be accessed by both Ad Server and Ad Engine implementations.

The C&PR is used to have a better understanding of what could best meet the needs of a user at the time of Ad delivery. Using C&PR, the Ad Server or Ad Engine may improve the selection of the Ad(s) to satisfy the given criteria and rules.

Examples of the C&PR used are:

- User Profile
- MobAd User Preferences (e.g. associating User Context information with Ad categories)
- Location
- Presence
- Other User Preferences
- Other User Context Information

Contextualisation and Personalisation Resources external to the MobAd Enabler may be used by MobAd Enabler entities (Ad Server, Ad Engine) to acquire Contextualisation and Personalisation information with the goal of improving the Ads selection process.

Such resources may include OMA enablers, but they cannot be restricted to OMA enablers, neither should such use of OMA enablers be mandated. As a result, the use of OMA enablers to obtain Contextualisation and Personalisation information is a MobAd enabler implementation choice, dictated mainly by the specifics of the environment where a MobAd enabler implementation may be deployed.

A non-exhaustive summary of some OMA enablers that could be used as Contextualisation and Personalisation Resources is provided starting with Appendix C.2.

C.1 Informative Reference Points

The reference points listed in this section are not specified by MobAd Enabler. A MobAd Enabler implementation however may use non-MobAd Enabler interfaces as I2 interfaces (see [OSE]) to get access to information available from Contextualisation and Personalisation Resources, used in the Ad(s) selection process. Both functional components of the MobAd Enabler, the Ad Server and the Ad Engine may use such external resources to acquire Contextualisation and Personalisation information.

Since there could be relatively large and non-exhaustive set of interfaces potentially used, they are represented in the architectural diagram as reference points.

C.1.1. CP-Network

CP-Network is a reference point between the Ad Server and all the combined external Contextualisation and Personalisation Resources that may be accessed by an Ad Server implementation, comprising all I2 interfaces exposed by those resources.

C.1.2. CP-Device

CP-Device is a reference point between the Ad Engine and all the combined external Contextualisation and Personalisation Resources that may be accessed by an Ad Engine implementation, comprising all I2 interfaces exposed by those resources.

C.2 OMA Presence

The OMA Presence Enabler provides principals with the possibility to store and manipulate Presence Information. It also allows principals to obtain the Presence Information of other principals (e.g. users, groups of users). The functionalities provided by the OMA Presence Enabler are specified in [OMA-PRS-SIMPLE-AD].

An implementation of the MobAd Enabler functional components (Ad Server, Ad Engine) may use the capabilities of the Presence Enabler to obtain Presence Information regarding a principal targeted for Ad(s).

If included in an implementation, the MobAd Enabler functional components interact with the OMA Presence Enabler via the reference points defined by the OMA Presence Enabler (i.e. for that particular implementation CP-Network and CP-Device include specific OMA Presence Enabler reference points).

C.3 OMA Location

The OMA Mobile Location Service (MLS) Enabler provides principals with the possibility to obtain location information of other user(s). The functionalities provided by the OMA MLS Enabler are specified in [OMA-MLS-AD].

An implementation of the MobAd Enabler functional components (Ad Server, Ad Engine) may use the capabilities of the OMA MLS Enabler to obtain location information regarding user(s) targeted for Ad(s).

If included in an implementation, the MobAd Enabler functional components interact with the OMA MLS Enabler via the reference point Le defined by the OMA MLS Enabler (i.e. for that particular implementation CP-Network and CP-Device include the OMA MLS Le reference point).

C.4 OMA XML Document Management (XDM)

The OMA XML Document Management Enabler allows users and other Enablers to store and manage XML documents [OMA-XDM-AD]. The functionalities provided by the XDM Enabler are specified in [OMA-XDM-AD].

An implementation of the MobAd Enabler functional components (Ad Server, Ad Engine) may use the capabilities of the OMA XDM Enabler to retrieve information such as User Profile information, MobAd Enabler preferences, MobAd Enabler rules, selected User Groups information, etc. Other SP authorised principals (e.g. a MobAd Enabler administrator) may use the capabilities of the XDM Enabler to store and manage the information mentioned before.

If included in an implementation, the MobAd Enabler functional components and/or other authorised principals interact with the OMA XDM Enabler via the reference points defined by the XDM Enabler (i.e. for that particular implementation CP-Network and CP-Device include specific OMA XDM reference points).

C.5 OMA DPE (Device Profile Evolution)

The OMA DPE Enabler provides means to communicate the static and dynamic device information. It can communicate the device static and dynamic capabilities to the server. The functionalities provided by the OMA DPE Enabler are specified in [OMA-AD-DPE].

An implementation of the MobAd Enabler functional components (Ad Server, Ad Engine) may use DPE Enabler to obtain device capabilities related information.

Where implemented, functional components of MobAd Enabler implementation interact with the OMA DPE Enabler via the interfaces defined by the OMA DPE Enabler (i.e. for that particular implementation CP-Network and CP-Device include specific OMA DPE Enabler interfaces).