CPE Application Best Practices

Overview

CONTENTS

REVISION HISTORY

Version	Date	Description

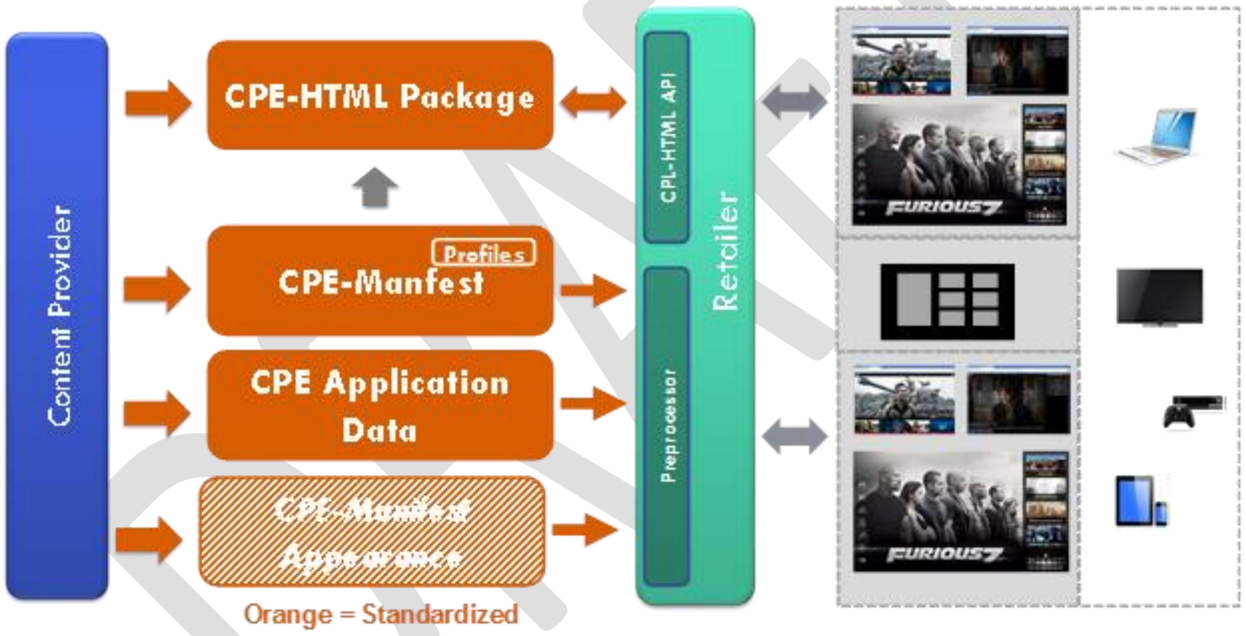
1 INTRODUCTION

This document is the common document in a set of CPE Best Practices for application implementation. As the parent document, it contains conventions, references and other rules commonly used in the individual Best Practices.

This document is part of the Cross-Platform Extras (CPE) collection of documents found at www.movielabs.com/cpe.

1.1 Overview

Cross-Platform Extras defines two models for providing applications: CPE-HTML and CPE-Manifest. The former is an API that allows HTML portability across retailers. The latter is a data-driven user experience with UI supplied by the retailer.



In either case, to allow applications to work across platforms it is important to clearly define specific practices (particularly data) for each application. The application is implemented once per platform and content is authored once. Because of the standardization, the content behaves the same across all platforms.

Consider the CPE Application Data box above. For example, let's say that includes a map coordinates, descriptions and other data. Each retailer can implement a mapping application that displays that map data. If the implementation is HTML/JavaScript then a single implementation could be used across all retailers.

Our intent is to supply sample implementations or reference code for various platforms to make it easier to implement applications. Contributions are welcome.

1.2 Document Notation and 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]. That is:

- “MUST”, “REQUIRED” or “SHALL”, mean that the definition is an absolute requirement of the specification.
- “MUST NOT” or “SHALL NOT” means that the definition is an absolute prohibition of the specification.
- “SHOULD” or “RECOMMENDED” mean that there may be valid reasons to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- “SHOULD NOT” or “NOT RECOMMENDED” mean that there may be valid reasons when the particular behavior is acceptable, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- “MAY” or “OPTIONAL” mean the item is truly optional, however a preferred implementation may be specified for OPTIONAL features to improve interoperability.

Terms defined to have a specific meaning within this specification will be capitalized, e.g. “Track”, and should be interpreted with their general meaning if not capitalized.

Normative key words are written in all caps, e.g. “SHALL”

1.3 Normative References

1.3.1.1 Core CPE Specifications

[CM]	Common Metadata, www.movielabs.com/md/md
[Manifest]	Common Metadata Media Manifest Metadata, www.movielabs.com/md/manifest
[CPE-Manifest]	Cross-Platform Extras, Manifest, www.movielabs.com/cpe/manifest
[CPE-HTML]	Cross-Platform Extras, HTML, www.movielabs.com/cpe/html
[CPE-AppData]	Cross-Platform Extras Application Data, www.movielabs.com/cpe/app

1.3.1.2 Commonly Referenced Specifications

[Ratings]	Common Ratings Metadata, www.movielabs.com/md/ratings
[RFC4646]	Philips, A, et al, *RFC 4646, Tags for Identifying Languages*, IETF, September, 2006. http://www.ietf.org/rfc/rfc4646.txt
[ISO639]	ISO 639-2 Registration Authority, Library of Congress. http://www.loc.gov/standards/iso639-2/
[ISO3166-1]	Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes, 2007.
[ISO3166-2]	ISO 3166-2:2007Codes for the representation of names of countries and their subdivisions -- Part 2: Country subdivision code
[ISO4217]	Currency shall be encoded using ISO 4217 Alphabetic Code. http://www.iso.org/iso/currency_codes_list-1
[ISO8601]	ISO 8601:2000 Second Edition, *Representation of dates and times, second edition*, 2000-12-15.
[TTML]	Timed Text Markup Language (TTML) 1.0, W3C Proposed Recommendation 14 September 2010, http://www.w3.org/TR/ttaf1-dfxp/

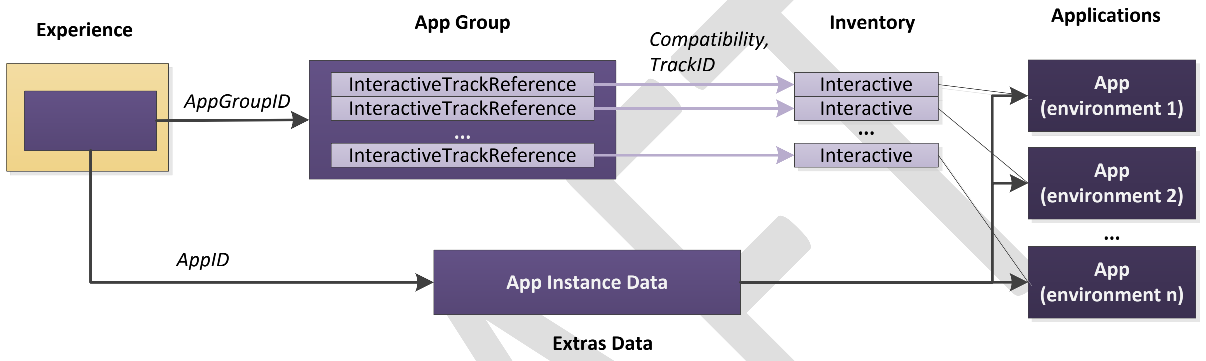
1.4 Informative References

[AdID]	Ad-ID advertisement identifier, www.ad-id.org

2 APPLICATION DATA SOURCES

Data can come from a variety of sources depending on the application. Commonly used sources are Common Metadata [CM], Media Manifest [Manifest] and CPE Application Data [CPE-AppData]. Data might can come from other sources as well. For example, when displaying scripts synchronized with the presentation, information can be self-contained within the script file, including synchronization data.

Application data can be delivered independently of the Manifest file as illustrated in the following picture and defined in [CPE-AppData].



An Experience references an App Group.

The Application Group, documented in [Manifest], Section 7.1, provides the mechanism to signal the same application implemented for different platforms. All applications in the Application Group should offer the same function. Each application in an Application Group supports a particular platform. For example, all applications in the Application Group might be mapping application; one for HTML, one for iOS, one for Android, one for PC, one for Mac, and so forth. This can be more granular if necessary (e.g., iOS 6-8, vs IOS 9).

The App Group references the Inventory that provides more details about the App, including where it can be found.

The App Group and Inventory is not necessarily specific to the Experience. For example, all mapping apps would be in the same App Group. If additional data is needed for the App Instance (i.e., the application behavior associated with that Experience), App Instance Data can be referenced by the Experience as @AppID. For example, while the mapping apps are in the App Group, the location for a map would be in the App Instance Data.

3 SELECTING AND DISPLAYING APPS

Determining which Applications to play requires attention because not all applications can be executed in all environments. The goal is to support a graceful degradation so that at least some functionality is available across all platforms. In the worst case, it should be possible to omit the application without substantial negative impact on the UI.

3.1 Application Groups

It is up to the Player to compare the Compatibility element to its own capabilities and choose the appropriate app.

AppGroup/Interactive/TrackReference should have a Compatibility instance for each environment supported by the application. Compatibility is defined in terms of DigitalAssetInteractiveEncoding-type found in [CM], Section 5.2.9.3.

The list of environments in [CM] is currently incomplete. The following list should be used:

- ‘Android’ – Android operating system native app
- ‘iOS’ – Apple iOS operating system native app [CHS: do we need separate iPhone and iPad?]
- ‘tvOS’ – Apple tvOS
- ‘MacOS’ – Apple MacOS native app
- ‘Windows’ – Microsoft Windows native app
- ‘WindowsPhone’ – Microsoft Windows phone operating system native app
- ‘BrightScript’ – Roku BrightScript native app
- ‘Linux’ – Linux native app
- ‘Flash’ – Adobe Flash
- ‘BD-J’ – Blu-ray Java
- ‘MHEG’ – MHEG-5, or more formally ISO/IEC 13522-5.
- ‘HTML5’ – W3C HTML5
- ‘Default’ – Represents an application that can be played if nothing else can. This is typically an image.
- ‘Other’ – may be used when there is not a type convention.

3.2 Graceful Degradation

Some platforms may be unable to implement certain applications. In this situation, there are two options

- Omit this application – The application disappears from the UI entirely.
- Show an image in lieu of the application

The choice of which one goes with the content author. If the application cannot be implemented and there would be no ‘hole’ in the UI (e.g., a menu list with no items) the application should be omitted. If a menu item has child elements that only consist of applications that cannot be played, that menu item is not shown. Authors should take care to avoid situations where this may occur.

In certain specific cases, an image is a sufficient replacement for the application. For example, if a mapping application could not be implemented in HTML, it might be sufficient to show a map image with the point of interest highlighted. An Interactive element is created in the Inventory as follows:

- Type=‘Image’
- Encoding/RuntimeEnvironment=‘Default’
- PictureID references a Picture that contains the default images. Note that these can be localized—this is why PictureID is referenced instead of ImageID.

3.3 App Display Information

When creating an interactive user interface, the player will display information about each user-accessible Experience. The Manifest is designed to allow a Player to display information about audiovisual, gallery and application Experiences in a nearly identical manner. Specifically, the ContentID element in each object references metadata with the display information.

For information on using metadata, see Manifest Interactivity Best Practices [ManifestBPI].

Note: AppName and Rating should not be used. They are redundant with ContentID-referenced metadata and will likely be deprecated.

3.4 Timed Event Triggered Apps

An app can be trigged by a Timed Event. The Timed Event is in itself context for the application invocation. That is, what happens is directly timed to that TimedEvent.

[CHS: Should probably add AppID to AppGroupID in Timed Event. Also, each event should have its own ID.]

3.5 Adding custom behavior

[CHS: Describe how to use these mechanisms to create functions bilaterally (or just extend this document). Pop-up purchase app is an example of how data can be combined or supplemented.]