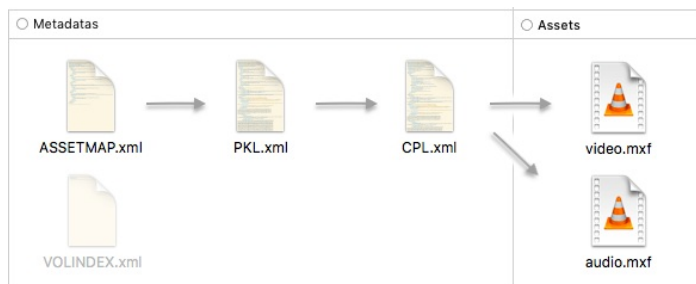


PRÉFACE



L'**AssetMap** est le premier fichier d'un DCP, c'est un index, il sert à identifier tous les fichiers présents d'un DCP - qui sont surnommés **assets** d'où le nom de **AssetMap**.

L'AssetMap est défini dans la norme **SMPTE ST 429-9 - D-Cinema Packaging — Asset Mapping and File Segmentation**

Son format interne :

- Le formatage de l'AssetMap est du texte utilisant la syntaxe **XML** (*Extensible Markup Language*).
- Il est de type **text/xml** avec un codage de caractère en **UTF-8**.
- Il est directement lisible par un humain avec un simple éditeur de texte.

Ces principales règles sont :

- Il n'existe qu'**une seule AssetMap par DCP**.
- Son nom est **obligatoirement** **ASSETMAP.xml** ¹
- L'AssetMap se trouve à la racine du répertoire ²

DESCRIPTION

L'AssetMap est l'équivalent de la table des matières d'un livre à la différence qu'il va lister tous les fichiers présents dans le répertoire contenant le DCP.

L'AssetMap sert d'index, il va donner un listing complet des noms des différents fichiers (assets) présents dans celui-ci. Les assets sont parfois accompagnés d'informations supplémentaires telles qu'une annotation, une taille, un offset et si cet asset est une **Packing List (PKL)** (qu'on étudiera plus tard).

C'est le point d'entrée pour tous logiciels manipulant du DCP.

Notez le "présents" : un DCP peut ne contenir qu'une petite partie des assets d'une oeuvre (par exemple, seulement les sous-titres) et - grâce aux fichiers de métadonnées **Packing List (PKL)** et **Composition Playlist (CPL)** - **faire uniquement des références à d'autres assets venant d'autres DCP**. Dans l'exemple du livre, ce serait les références vers d'autres ouvrages. Dans notre exemple, l'AssetMap ne fera référence qu'au fichier de sous-titres (en plus des deux fichiers de métadonnées).

L'ASSETMAP MINIMALE

Voici l'exemple d'une **AssetMap minimale** intégrant les éléments et les données obligatoires, et 4 assets (PKL, CPL, video, audio) pour un film classique ³ :

```
<?xml version="1.0" encoding="UTF-8"?>
<AssetMap xmlns="http://www.smpte-ra.org/schemas/429-9/2007/AM">
  <Id>urn:uuid:606e9f2d-3a4c-49e2-ba4a-fc213caf1e64</Id>
  <Creator>DCP Inside</Creator>
  <VolumeCount>1</VolumeCount>
  <IssueDate>2022-01-01T00:00:00-00:00</IssueDate>
  <Issuer>DCP Inside</Issuer>
  <AssetList>
    <Asset>
      <Id>urn:uuid:ce5c22c8-a640-428c-9004-2107e1f3c94e</Id>
      <PackingList>true</PackingList>
      <ChunkList>
        <Chunk>
          <Path>PKL.xml</Path>
        </Chunk>
      </ChunkList>
    </Asset>
    <Asset>
      <Id>urn:uuid:d2e32d20-1f85-4d86-b09f-2ee40ac097a0</Id>
      <ChunkList>
        <Chunk>
          <Path>CPL.xml</Path>
        </Chunk>
      </ChunkList>
    </Asset>
    <Asset>
      <Id>urn:uuid:af457798-e34e-4233-9fe1-3cc974ca33e9</Id>
      <ChunkList>
        <Chunk>
          <Path>picture.mxf</Path>
        </Chunk>
      </ChunkList>
    </Asset>
    <Asset>
      <Id>urn:uuid:6e007158-b706-4168-964d-53f35e7d2b74</Id>
      <ChunkList>
        <Chunk>
          <Path>audio.mxf</Path>
        </Chunk>
      </ChunkList>
    </Asset>
  </AssetList>
</AssetMap>
```

Rapidement, nous voyons les quatres assets : **PKL.xml**, **CPL.xml**, **picture.mxf** et **audio.mxf**. Nous y reviendrons en détails plus tard.

L'ASSETMAP COMPLÈTE

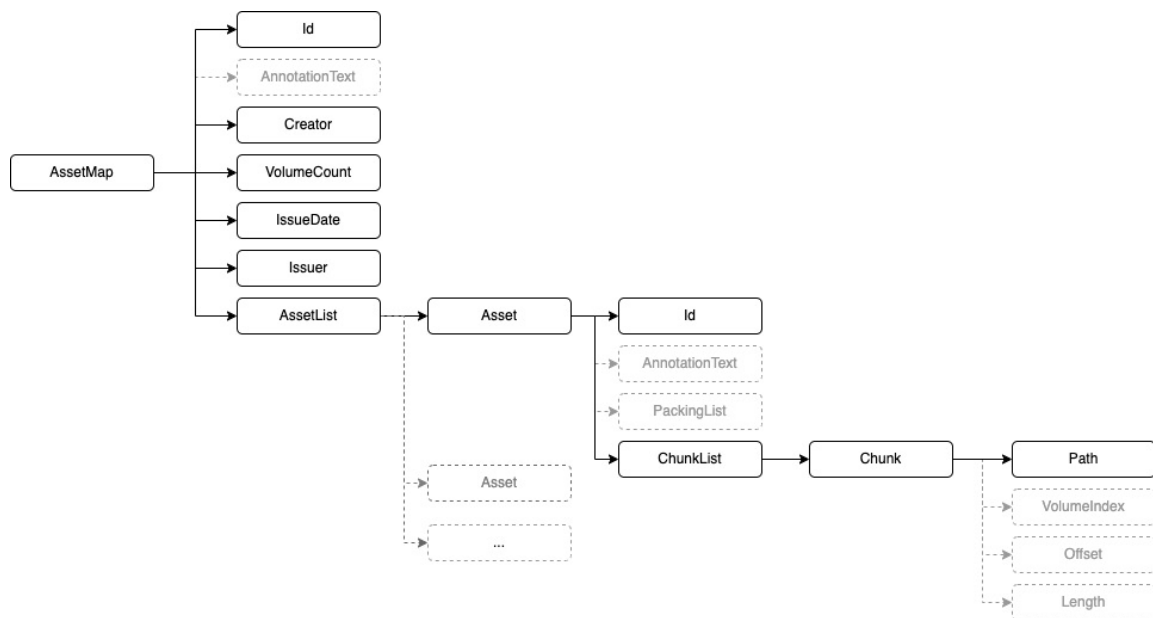
Voici l'exemple d'une [AssetMap complète](#) comprenant l'ensemble des éléments, des attributs et des données possibles :

```
<?xml version="1.0" encoding="UTF-8"?>
<AssetMap xmlns="http://www.smpte-ra.org/schemas/429-9/2007/AM">
  <Id>urn:uuid:606e9f2d-3a4c-49e2-ba4a-fc213caf1e64</Id>
  <AnnotationText language="en">DCP-INSIDE_FTR-2D-24_S_FR-FR_FR_71-FR_4K_UP_20220101_SMPTE_OV</AnnotationText>
  <Creator language="en">DCP Inside</Creator>
  <VolumeCount>1</VolumeCount>
  <IssueDate>2022-01-01T00:00:00-00:00</IssueDate>
  <Issuer language="en">DCP Inside</Issuer>
  <AssetList>
    <Asset>
      <Id>urn:uuid:ce5c22c8-a640-428c-9004-2107e1f3c94e</Id>
      <AnnotationText language="en">Packing List (PKL)</AnnotationText>
      <PackingList>true</PackingList>
      <ChunkList>
        <Chunk>
          <Path>PKL.xml</Path>
          <VolumeIndex>1</VolumeIndex>
          <Offset>0</Offset>
          <Length>1690</Length>
        </Chunk>
      </ChunkList>
    </Asset>
    <Asset>
      <Id>urn:uuid:d2e32d20-1f85-4d86-b09f-2ee40ac097a0</Id>
      <AnnotationText language="en">Composition Playlist (CPL)</AnnotationText>
      <ChunkList>
        <Chunk>
          <Path>CPL.xml</Path>
          <VolumeIndex>1</VolumeIndex>
          <Offset>0</Offset>
          <Length>1901</Length>
        </Chunk>
      </ChunkList>
    </Asset>
    <Asset>
      <Id>urn:uuid:af457798-e34e-4233-9fe1-3cc974ca33e9</Id>
      <AnnotationText language="en">Asset Picture</AnnotationText>
      <ChunkList>
        <Chunk>
          <Path>picture.mxf</Path>
          <VolumeIndex>1</VolumeIndex>
          <Offset>0</Offset>
          <Length>1280112340</Length>
        </Chunk>
      </ChunkList>
    </Asset>
    <Asset>
      <Id>urn:uuid:6e007158-b706-4168-964d-53f35e7d2b74</Id>
      <AnnotationText language="en">Asset Sound</AnnotationText>
      <ChunkList>
        <Chunk>
          <Path>audio.mxf</Path>
          <VolumeIndex>1</VolumeIndex>
          <Offset>0</Offset>
          <Length>56172026</Length>
        </Chunk>
      </ChunkList>
    </Asset>
  </AssetList>
</AssetMap>
```

On sera bien d'accord que ceci est un simple exemple et que le nombre d'assets peut être plus important.

SCHÉMA VISUEL DE LA STRUCTURE XML

Voici un schéma visuel simplifié de la structure interne d'une AssetMap.



Une explication rapide sur comment lire le schéma :

Chaque case représente un élément (*tag*), les flèches indiquent une structure enfante existante (*child*), les cases grises représentent l'optionnalité de l'élément et les pointillés représentent plusieurs éléments potentiels, ainsi :

- **Id** est un élément à la racine et contient un identifiant
- **AssetList** va contenir un ou plusieurs **Asset**
- **Asset** possède un élément **Id** et un élément **ChunkList**
- **ChunkList** contient un seul élément **Chunk**
- Et ainsi de suite jusqu'à **Path**

Important : Les éléments doivent suivre cet ordre. ⁴

EXPLICATION DE LA STRUCTURE XML

LA STRUCTURE RACINE :

À la racine, nous avons 7 éléments de base, dont un optionnel :

Nom	Format	Exemple	
Id	UUID	urn:uuid:49d8371e-393b-48d8-a3d7-059f43bc637c	Obligatoire
AnnotationText	Texte	DCP-INSIDE_FTR-2D-24_S_FR-FR_FR_71-FR_4K_UP_20220101_SMPTE_OV	Optionnel
Creator	Texte	DCP Inside	Obligatoire
VolumeCount	Integer	1	Obligatoire
IssueDate	DateTime	2022-01-01T00:00:00-00:00	Obligatoire
Issuer	Texte	DCP Inside	Obligatoire
AssetList	xml:AssetList	<AssetList><Asset/></AssetList>	Obligatoire

LA STRUCTURE **ASSETLIST** :

La structure **AssetList** contient uniquement un ou plusieurs **Asset** :

LA STRUCTURE **ASSET** :

Nom	Format	Exemple	Obligatoire
Id	UUID	urn:uuid:86a13dd1-b9a7-4ab2-82f7-d36907d47745	Obligatoire
AnnotationText	Texte	Asset Picture	Optionnel
PackingList	Boolean	true	Optionnel
ChunkList	xml:ChunkList	<ChunkList><Chunk/></ChunkList>	Obligatoire

LA STRUCTURE **CHUNKLIST** :

Dans **ChunkList**, vous avez un seul **Chunk**.

LA STRUCTURE **CHUNK** :

Nom	Format	Exemple	Obligatoire
Path	Texte (URI)	CPL.xml	Obligatoire
VolumeIndex	Integer	1	Optionnel
Offset	Integer	0	Optionnel
Length	Integer (Bytes)	56172026	Optionnel

EXPLICATIONS DES DIFFÉRENTS ÉLÉMENTS

L'AssetMap a une sorte d'en-tête (header) intégrant des informations comme son identifiant unique, son titre descriptif, son créateur, sa date de création.

Cet en-tête est composé de :

- Id
- AnnotationText
- Creator
- VolumeCount
- IssueDate
- Issuer

Et finalisé par un AssetList

ID

Type de présence : Obligatoire

Description : Ce champ définit un identifiant unique appelé UUID.

Au format UUID - normé par la RFC 4122 - il est obligatoirement préfixé par un urn:uuid suivi de son UUID.

Exemple d'un Id :

```
<Id>urn:uuid:606e9f2d-3a4c-49e2-ba4a-fc213caf1e64</Id>
```

Et son champ de version sera toujours à 4 (pour tous les UUID utilisant la RFC 4122, que ce soit pour les assets ou les identifiants de clés de chiffrage (KeyId) :

```
<Id>urn:uuid:606e9f2d-3a4c-9e2-ba4a-fc213caf1e64</Id>
```

ANNOTATIONTEXT

Type de présence : Optionnel

Description : Ce champ définit titre descriptif du DCP.

Au format texte, la composition de ce champ est libre. Cependant, il est fortement recommandé d'appliquer la syntaxe Digital Cinema Naming Convention. Notez que le non respect de ce formalisme ne vaut pas l'invalidation du DCP.

L'AnnotationText peut s'écrire de cette façon:

```
<AnnotationText>AssetMap</AnnotationText>
```

Ou encore en utilisant la convention de nommage :

```
<AnnotationText>DCP-INSIDE_FTR-2D-24_S_FR-FR_FR_71-FR_4K_UP_20220101_SMPTE_OV</AnnotationText>
```

Ou encore avec l'attribut optionnel "language" permettant de définir la langue de ce contenu :

```
<AnnotationText language="en">DCP-INSIDE_FTR-2D-24_S_FR-FR_FR_71-FR_4K_UP_20220101_SMPTE_OV</AnnotationText>
```

CREATOR

Type de présence : Obligatoire

Description : Ce champ définit le matériel ou logiciel qui a généré ce DCP.

Au format texte, la composition de ce champ est libre.

Le Creator peut s'écrire de cette façon:

```
<Creator>DCP Inside</Creator>
```

Ou encore avec l'attribution optionnel "*language*" permettant de définir la langue de ce contenu :

```
<Creator language="en">DCP Inside</Creator>
```

Sur différents DCP, vous pourrez avoir des noms de "Creator" comme :

- orca_wrapping (Doremi)
- EasyDCP
- CipherAssetMapGen
- ClipsterDCI
- Colorfront Transkoder
- OpenDCP
- Autres...

VOLUMECOUNT

Type de présence : Obligatoire

Description : Ce champ est censé définir un nombre de volume mais la fonctionnalité n'a jamais été normalisée et donc jamais utilisée, vous pouvez donc l'ignorer sans état d'âme. Sa valeur sera toujours 1

ISSUEDATE

Type de présence : Obligatoire

Description : Ce champ définit la date de création du DCP. Elle doit respecter la normalisation DateTime ISO-8601.

La syntaxe habituelle est la suivante :

```
YYYY-MM-DDThh:mm:ss+hh:mm
```

Ce qui correspond :

Symbole	Description
YYYY	Année sur 4 chiffres
MM	Mois sur 2 chiffres - de 01 à 12
DD	Jour sur 2 chiffres - de 01 à 31
T	Séparateur de temps (time)
hh	Heure sur 2 chiffres - de 01 à 23
mm	Minute sur 2 chiffres - de 01 à 59
ss	Seconde sur 2 chiffres - de 01 à 59
+/-/Z	Séparateur fuseau horaire. Ce séparateur peut être soit + , soit - , soit Z . Si Z est utilisé, les 2 champs suivants ne sont pas utilisés
hh	Heure de décalage - de 01 à 23
mm	Minute de décalage - de 01 à 59

L'IssueDate peut s'écrire de cette façon :

```
<IssueDate>2022-01-15T09:30:00+00:00</IssueDate>
```

ISSUER

Type de présence : Obligatoire.

Description : Ce champ définit le créateur (personne ou société) du DCP.

Au format texte, la composition de ce champ est libre.

Le Issuer peut s'écrire de cette façon:

```
<Issuer>DCP Inside</Issuer>
```

Ou encore avec l'attribution optionnel "*language*" permettant de définir la langue de ce contenu :

```
<Issuer language="en">DCP Inside</Issuer>
```

Sur différents DCP, vous pourrez des noms de "Issuer" comme :

- Doremi Labs, Inc.
- Qube
- Dolby Laboratories Inc.

- `Clipster DCI`
- `ColorFront`
- Le nom ou l'acronyme d'un studio
- Le nom ou l'acronyme d'un laboratoire
- Le nom, pseudonyme ou acronyme d'une personne physique (rare)
- Autres..

ASSETLIST

Type de présence : Obligatoire.

Description : Il n'est qu'un container, il ne fait que contenir une liste d'éléments de type `Asset` . Il ne contient aucun autre élément.

Les règles principales sont :

- Il faut au **minimum** un élément `Asset` dans `AssetList`
- Il n'y a pas d'ordre dans l'ordre des éléments `Asset`

Schématiquement, une `AssetList` ressemble à cela, rien de plus :

```
<AssetList>
  <Asset>(…)</Asset>
  <Asset>(…)</Asset>
  <Asset>(…)</Asset>
  <Asset>(…)</Asset>
  <Asset>(…)</Asset>
</AssetList>
```

ASSETLIST → ASSET

Un exemple d'un `Asset` :

```
<Asset>
  <Id>urn:uuid:ce5c22c8-a640-428c-9004-2107e1f3c94e</Id>
  <AnnotationText language="en">Packing List (PKL)</AnnotationText>
  <PackingList>true</PackingList>
  <ChunkList>
    <Chunk>
      <Path>PKL.xml</Path>
      <VolumeIndex>1</VolumeIndex>
      <Offset>0</Offset>
      <Length>1690</Length>
    </Chunk>
  </ChunkList>
</Asset>
```

Type de présence : Obligatoire (au minimum 1)

Description : `Asset` est un container intégrant les informations de l'asset (fichier)

Il faut un élément `Asset` par fichier présent dans le DCP (à l'exception de l'`AssetMap`)

L'élément `Asset` possède 4 sous-éléments, dont 2 optionnels :

- Le **sous-élément** `Id` se comporte comme précédent vu, il définit un identifiant unique par asset qui les suivront sur l'ensemble du DCP. Les identifiants ont un impact et une importance dans l'ensemble du DCP. Ils sont utilisés et sont présents dans les autres fichiers. Par exemple, pour une `PKL` ou pour la `CPL`, l'identifiant sera à la racine de leur structure. Pour les `MXF`, l'identifiant sera présent dans le `MXF` (dans la partie [Source Package](#) → champ `PackageUID`).

Selon les fichiers, il faudra donc les récupérer :

Type d'asset	Lien chapitre
Pour une PKL	Récupération de l'identifiant (ID) d'une PKL
Pour une CPL	Récupération de l'identifiant (ID) d'une CPL
Pour un MXF	Récupération de l'identifiant (PackageUID) d'un MXF

- Le **sous-élément** `AnnotationText` se comporte comme précédent vu. Il indique des informations sur l'asset en lui-même. Le champ est libre, vous pouvez même l'ignorer si vous n'avez pas plus d'informations à communiquer dessus.
- Le **sous-élément** `PackingList` permet de définir si l'asset est une [Packing List \(PKL\)](#) ou non.

A noter que la bonne syntaxe est :

```
<PackingList>true</PackingList>
```

Les autres syntaxes doivent être considérées comme nulle ⁵ :

```
<PackingList/>
<PackingList></PackingList>
<PackingList>false</PackingList>
```

- Et enfin, le sous-élément `ChunkList` qui intègre ... un seul `Chunk` (*morceau*). A l'origine, il était prévu d'intégrer plusieurs "morceaux" d'un

fichier. Cela n'a jamais été fait. Vous ne verrez donc qu'un seul élément `Chunk`.

Les éléments `Id`, `AnnotationText` (si présent), `PackingList` (si présent) et `ChunkList` doivent suivre cet ordre.

ASSETLIST → ASSET → CHUNKLIST

Dans `ChunkList`, il n'existe qu'un seul `Chunk`.

Tout comme d'autres éléments prévus mais jamais utilisés, il était prévu de pouvoir découper les fichiers en plusieurs petits morceaux (chunks).

ASSETLIST → ASSET → CHUNKLIST → CHUNK

Type de présence : Obligatoire (un seul)

Description : Est un container intégrant les informations de la partie ou de la totalité de l'asset (fichier) - dans notre cas, c'est la totalité.

Le `Chunk` n'a qu'un seul élément obligatoire : c'est le `Path` : le chemin d'accès vers le fichier. C'est le minimum vital pour l'AssetMap.

Le `Path` est soumis à quelques règles :

- Il doit respecter la RFC 3986 et le type XML anyURI.
- Le chemin vers le fichier doit être relatif et non absolu
- Le nom du fichier est lui aussi codifié, il ne peut intégrer que des lettres (A à Z) en majuscules et minuscules, des chiffres (0 à 9) et des tirets '-' ou '_' et le point.
- Il existe aussi une restriction concernant la taille des segments. Un segment est la partie entre deux '/' : il est limité à 100 caractères par segment avec un maximum de 10 segments. Autrement dit: vous ne pouvez avoir un fichier placé dans le 11ème sous-repertoire d'un DCP⁶.
- Et enfin, il doit respecter la sensibilité à la casse typographique (case sensitive) : "fichier.mxf" sera différent de "FICHIER.MXF"

Voici des exemples de valeurs de `Path` valides ou invalides :

Chemin	Statut	Notes
fichier.mxf	Valide	
dossier/fichier.mxf	Valide	
./dossier/fichier.mxf	Valide	
dossier/../fichier.mxf	Valide	Est dans la structure du dossier racine
/fichier.mxf	Invalide	Les chemins absolus sont interdits
../fichier.mxf	Invalide	N'est pas dans la structure du dossier racine, il est en dehors
PKL.xml	Valide	
pAckInGLiSt.xml	Valide	
PKL_58c333cb-11dc-44e8-b7ae-fdd5b8514d66.xml	Valide	
subtitles/Arial.ttf	Valide	
subtitles/Reel01.xml	Valide	
english-version_1234/subtitles-English-V2.xml	Valide	
f0e3118a-567d-465b-bdd1-f0cb84012dfe/subs.xml	Valide	
subs/subtitles_fr.mxf	Valide	
vidéo-reel-1.mxf	Invalide	Présence d'un caractère non valide ("é")
audio (piste).mxf	Invalide	Présence d'un espace et des parenthèses
CPL[42].xml	Invalide	Présence des crochets

Vous retrouverez les 666.795 Paths respectant cette règle et 684 Paths ne la respectant pas - mais aperçu dans des DCP (ces derniers pouvant avoir été créés mais pouvant avoir été rejetés par la suite).

Sur plus de 80.000 AssetMap étudiés, dans tous les cas, les fichiers se trouvent à la racine du DCP et s'il y a un répertoire, il ne sert qu'à stocker et séparer certains types d'assets, comme des sous-titres par exemple. Jamais constaté : une cascade de sous-répertoires, même sur des gros DCP (plusieurs PKL, plusieurs CPL).

Les autres éléments sont totalement facultatifs :

- VolumeIndex** : Doit être toujours à 1, c'est aussi un reliquat d'une possible évolution mais qui n'a jamais été mis en place, vous pouvez ignorer cet élément.
- Offset** : Doit être toujours à 0, c'est aussi un reliquat d'une possible évolution mais qui n'a jamais été mis en place, vous pouvez ignorer cet élément aussi.
- Length** : Doit être la taille en octets (bytes) du fichier. Notez que si `Length` est défini, les logiciels/matériels peuvent être amenés à vérifier la taille effective du fichier. Si cette taille n'est pas correcte, ils peuvent rejeter le DCP.

Vous devez respecter cet ordre. Si vous utilisez des éléments optionnels, il suffit de suivre l'ordre, par exemple : `Path` puis `Length`.

CRYPTOGRAPHIE : LE GRAND MANQUANT

A contrario des autres éléments du DCP, l'AssetMap est le **seul élément du DCP ne contenant aucune forme de cryptographie** : Aucun [hash](#) (ou empreinte numérique), aucune [signature ou authentification](#), ni aucun [chiffrement](#).

INTEROP VS SMPTE

Les différences entre une AssetMap SMPTE et une AssetMap Interop :

- Le nom du fichier :
 - `ASSETMAP.xml` (SMPTE)
 - `ASSETMAP` (Interop)
- Le Namespace XML :
 - `xmlns="http://www.smpste-ra.org/schemas/429-9/2007/AM"` (SMPTE)
 - `xmlns="http://www.digicine.com/PROTO-ASDCP-AM-20040311#"` (Interop)
- Attributs XML "language" et "lang" :
 - `lang="en"` (Interop)
 - `language="en"` (SMPTE)
- `<PackingList/>`
 - `<PackingList/>` et `<PackingList>true</PackingList>` acceptés en Interop
 - `<PackingList>true</PackingList>` seulement accepté en SMPTE

NOTES

- Voir **Interop vs SMPTE** ↩
- La norme SMPTE accepte le fait d'avoir des AssetMaps dans des sous-répertoires. Dans ce cas, l'AssetMap présente à la racine sera ignorée. Considérez ce cas de figure comme inutile : si plusieurs AssetMaps sont dans différents sous-répertoires, ce sont juste plusieurs DCP dans un répertoire commun. Rien de plus. ↩
- Selon la norme SMPTE, il faut au minimum une PKL et un asset (la CPL étant considérée comme un asset) ↩

« A DCP shall consist of one Packing List and one or more assets (i.e., Composition Playlists and/or Track Files), referenced by the Packing List.-- SMPTE ST 429-2-2013 - DCP Operational Constraints - Chap. "Minimum Contents" ».

Ici, nous avons pris l'exemple d'un film classique le plus minimal possible - avec qu'une piste d'image et qu'une piste sonore seulement, donc :

- une AssetMap
- une PackingList (PKL)
- une CompositionPlaylist (CPL)
- un MXF Picture
- un MXF Audio.

Si on essaye de faire un DCP le plus minimal possible, nous pourrions avoir que la combinaison "AssetMap + PKL + CPL" sans aucun asset. L'AssetMap ne faisant que référence à la PKL et la CPL. La CPL ne faisant références qu'à des assets dits "virtuels" donc qui proviennent d'autres DCP.

- Malgré l'utilisation du XML qui permet de placer les éléments dans n'importe quel ordre. Ici, on ne peut changer la position des différents éléments dans une AssetMap. Par exemple, à la racine, vous devez respecter que les éléments `Id`, `AnnotationText`, `Creator`, `VolumeCount`, `IssueDate`, `Issuer` et `AssetList` se suivent dans cet ordre. De même que pour d'autres éléments à d'autres endroits. Vérifiez dans le [schéma XSD de l'AssetMap](#): si les éléments sont entourés du tag `xs:sequence`, il faut respecter un ordre. ↩
- La forme `<PackingList/>` est acceptée sur des DCP Interop. ↩
- À noter que même le nom du répertoire contenant le DCP, où se trouve l'AssetMap, doit être limité à 100 caractères. ↩