**Common Promotions**

**Contents**

1 Introduction 1

1.1 Document Organization 1

1.2 Document Notation and Conventions 1

1.3 Normative References 1

1.4 Informative References 1

2 Promotion Definition 3

2.1 If-Then model 3

2.2 Note on Promotions and Natural Language 4

3 Promotions Structure 5

3.1 Promo element and Promo-type complex type 5

3.1.1 PromoList 5

3.2 Common Types 6

3.2.1 ProductID-type 6

3.2.2 Object-type 6

3.2.3 When-type 7

3.2.4 Terms-type 8

3.3 If-type 9

3.3.1 IfNumberOf-type 9

3.4 Then-type 9

3.4.1 ThenNumberOf-type 10

3.5 Combinations-type 11

3.5.1 IfCombination-type 11

3.5.2 ThenCombination-type 11

4 Patterns and Examples 12

4.1 Format for Templates 12

4.2 Encoding frequently used promotion patterns 12

4.2.1 Buy *x* Get *y* at Terms 12

4.2.2 Offer with no precondition 16

4.2.3 Get *subset of assets* for terms 16

4.3 Additional Patterns 19

4.3.1 Tiered offers 19

4.4 Other 19


This work is licensed under a [Creative Commons Attribution 3.0 Unported License](http://creativecommons.org/licenses/by/3.0/).

**NOTE**: No effort is being made by the Motion Picture Laboratories to in any way obligate any market participant to adhere to Common Metadata. Whether to adopt the Common Metadata in whole or in part is left entirely to the individual discretion of individual market participants, using their own independent business judgment. Moreover, Motion Picture Laboratories disclaims any warranty or representation as to the suitability of the Common Metadata for any purpose, and any liability for any damages or other harm you may incur as a result of subscribing to this Common Metadata. **Revision History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| 1.0 | TBD | Original Version |

# Introduction

This specification provides a means to specify promotions such as “Buy one, get one free” and “Earn 1000 points and get a free T-shirt”. It defines a general model that provides a wide range of promotional opportunities.

The Promotion spec (‘Promo’) is intended to work in conjunction with EMA Avails.

## Document Organization

This document is organized as follows:

1. Introduction—Provides background, scope and conventions
2. Definition of a Promotion
3. Promotions Structure
4. Patterns and Examples

## Document Notation and Conventions

This document uses conventions as defined in [CM]. This is a less formal document, so strict conventions may not expressly apply in all cases.

## Normative References

|  |  |
| --- | --- |
| [CM] | Common Metadata, TR-META-CM, [www.movielabs.com/md/md](http://www.movielabs.com/md/md)  |
| [Manifest] | MovieLabs Common Media Manifest Metadata v1.5, TR-META-MMM, [www.movielabs.com/md/manifest](http://www.movielabs.com/md/manifest)  |
| [Avail] | EMA Content Availability Data (Avails), TR-META-AVAIL, [www.movielabs.com/md/avails](http://www.movielabs.com/md/avails)  |
| [MEC] | Media Entertainment Core, TR-META-MEC, [www.movielabs.com/md/mec](http://www.movielabs.com/md/mec) |
| [GTIN] | GS1, General Specification, Version 15 (issue 2), January 2015 (or later), <http://www.gs1.org/gtin>  |
| [Ad-ID] | Ad-ID advertisement identifier, <http://www.ad-id.org/how-it-works>  |
| [ISRC] | Master recordings, ISO 3901, <http://www.ifpi.org/content/section_resources/isrc.html>  |

## Informative References

|  |  |
| --- | --- |
| [MMC] | Media Manifest Core, TR-META-MMC, [www.movielabs.com/md/mmc](http://www.movielabs.com/md/mmc)  |
| [EIDR-UG] | EIDR 2.0 Registry User’s Guide, [eidr.org/technology/](http://eidr.org/technology/)  |
| [EIDR-ID] | EIDR ID Format, [eidr.org/technology/](http://eidr.org/technology/)  |

# Promotion Definition

## If-Then model

Promotions are structured with the following structure:

**IF <condition> THEN <promotion>**

For example, IF *buy one*, THEN *get one free*. The IF part is optional, as some promotions do not have any preconditions. For example, *get any title for 30% off*.

The <condition> and <promotion> themselves have multiple parts. A fuller enumeration is as follows:

**IF
 <action taken>
 ON <number of>
 <objects>
THEN
 DURING <action timeframe>
 TAKE <action>
 WITH <terms>
 ON <number of>
 <objects>
 AT <result timeframe>**

Where

* Common terms
	+ <objects> is anything that can be bought, sold, earned, acquired, traded or otherwise transacted against. Objects can be physical (e.g., Blu-ray, T-shirt) or digital (e.g., EST license, points).
	+ <number of> is a quantification of the number of objects
* “IF” terms
	+ <action taken> is a verb defining an action on objects (e.g., buy, sell, trade, and earn)
* “THEN” specific terms
	+ <action timeframe> is a timeframe when the <action> can be taken. This indicate a delay (e.g., “one week later”), a deadline (e.g., “offer good for one week”, or until July 5), or a window (e.g., “no earlier than one week from now, but no later than two weeks from now”).
	+ <terms> defines the terms of the offer. This is the most open parameter because this spec strives not to constrain terms. The terms model used in EMA Avails [Avails] is supported.
	+ <action> is an action that may be taken by the consumer or on behalf of the consumer.
	+ <result timeframe> is the period when action takes effect. This can be immediate, delayed or tied to another event.

Generally, not all terms are not relevant to a promotion. The minimal construct is:

**THEN
 TAKE <action>
 WITH <terms>
 <objects>**

For example, if the promotion was “Buy from this list of titles at $5 each.”, <action> is “Buy”, <terms> are “$5 each”, and <objects> is the list of titles.

## Note on Promotions and Natural Language

To this point, the promotion has been presented in language structure. However, promotions are contractual and need to be precise. Also, promotions can be substantially more complex than can be practically captured in natural language. And, a complex promotion wouldn’t even be very readable in natural language (think lots of nested parenthesis).

Our hope is that people will develop tools that support natural language input and output to make it easier to generate and interpret promotions.

# Promotions Structure

Promotions are captured in XML documents using the “promo” namespace, defined here.

“promo” builds on EMA Avails [Avails] (referenced here as “avails:” namespace) and Common Metadata [CM] (referenced here as “md:” namespace).

## Promo element and Promo-type complex type

The Promo element is defined as Promo-type.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **Promo-type** |  |  |  |  |
|  | Type |  |  |  |
| If |  | “If” part of an If-Then construct | promo:If-type | 0..n |
| Then |  | “Then” part of an If-Then construct | promo:Then-type | 1..n |
| Combinations |  | Instructions on how to combine multiple IF or Then instances | promo:Combinations-type | 0..1 |

### PromoList

For multiple promotions, use PromoList defined as PromoList-type.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **PromoList-type** |  |  |  |  |
| Promo |  | Promotion | promo:Promo-type | 1..n |
|  | promoGroup | Label that defines groups of promotions. Promo elements with the same @promoGroup are mutually exclusive. | xs:string | 0..1 |

When multiple Promo instances exist and they have the same @promoGroup label, these promotions are designed to offered exclusively. For example, if the offer is “Buy 1 get 10% off, Buy 2 get 20% off, Buy 3 get 30% off” each offer is encoded in its own Promo. However, they are all given the same promoGroup so they can’t be stacked (consumer gets only one).

## Common Types

### ProductID-type

xxx

Note that this uses the Content type for structure. There is no implication what is referenced is Content.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **ProductID-type** |  |  | md:ContentIdentifier-type |  |

### Object-type

The Object-type complex time defines the objects that can be subject of the IF and THEN clauses.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **Object-type** |  |  |  |  |
| Assets |  | Media assets as defined by ALID | md:id-type | 0..n |
| GroupingEntity |  | Grouping for presentation to consumer. Includes branding. See [MEC]. | md:GroupingEntity-type | 0..n |
| Subscription |  | Subscription offering | promo:Subscription-type | 0..n |
| AddOn |  | Add-on offering, such as game add-ons. | promo:AddOn-type | 0..n |
| GTIN |  | Global Trade Item Number (GTIN) for UPC, EAN and other item numbers; in accordance [GTIN] | xs:string | 0..n |
| ProductID |  | General structure for references to product via IDs. Used when ID is not supported by other elements. | promo:ProductID-type | 0..n |
| ISRC |  | International Standard Recording Code (ISRC) to reference songs; in accordance with [ISRC]. | xs:string, pattern [A-Z0-9]{2}-[A-Z0-9]{3}-[0-9]{2}-[0-9]{5} | 0..n |
| AdID |  | Ad-ID advertising identifier, in accordance with [Ad-ID] | xs:sting, pattern [a-zA-Z1-9][a-zA-Z0-9]{10}[hHdD]? | 0..n |
| URL |  | A URL that references an object | xs:anyURI | 0..n |
| Cash |  | Monetary value | md:Money-type | 0..n |
| Coupon |  | Coupon identification [CHS: This likely needs work. Need more coupon metadata, even if just a name and an ID to create a unique coupon code.] | xs:string | 0..n |

#### Subscription-type

Subscriptions are services or objects delivered over a period of time. For example, a subscription to an online service or to a magazine.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **Subscription-type** |  |  |  |  |
| SubscriptionID |  | Identifier for subscription [CHS: is this sufficent?] | xs:string |  |
| Start |  | Start date and time for subscription. If absent, subscription is immediate or ASAP. | xs:dateTime | 0..1 |
| Duration |  | Duration of subscription. | xs:duration | 0..1 |

#### AddOn-type

Addons are additional related materials, typically plug-ins, extensions or additional material for games. This does not include audiovisual material as those are covered by Assets, ISRC, etc.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **AddOn-type** |  |  |  |  |
| AddOnID |  | Identifier for addon.  | xs:string |  |

### When-type

This type defines when something happens. It’s either in the form of a time or a condition. Note that this can be used for both start and end conditions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **When-type** |  |  |  |  |
| NoEarlierThan |  | The earliest time the event can occur or period can begin or end. | xs:dateTime | 0..1 | Seq. | Choice |
| NoLaterThan |  | The latest time the event can occur or period can begin or end. | xs:dateTime | 0..1 |
| Condition |  | Condition for time. Examples of conditions are found in [Avails]. | xs:string |  |  |
|  | Lag | Length of time before or after Condition that event or period begins or ends. | xs:duration |  |  |

### Terms-type

Terms are business terms associated with an IF or THEN clause.

They are designed to cover a wide range of offerings and adjustments.

UnitPrice is a price for each object is individually. Note that this might be special pricing.

LotPrice provides a price for all objects as single unit.

DiscountMultipler is used for sales, typically between 0 and 1 (non-inclusive). For example, a DiscountMultiplier of 0.5 means 50% off.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **Terms-type** |  |  |  |  |
| UnitPrice |  | Unit price | md:Money-type | choice |
| LotPrice |  | Lot price | md:Money-type |
| DiscountMultiplier |  | Multiple of original price to determine actual price. | xs:float |
| PriceInclusive |  | Objects are included in original [CHS: I think this should be used exclusively of the first three.]  | xs:boolean | 0..1 |

## If-type

The If-type complex type defined the IF clause as described in Section 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **If-type** |  |  |  |  |
| Action |  | Action that must be taken for if condition to be met. | promo:Action-type |  |
| NumberOf |  | Number of Objects associated with Action. If absent, there are no quantity limits. | promo-IfNumberOf-type | 0..1 |
| Objects |  | Objects associated with Action. | promo:Objects-type |  |

### IfNumberOf-type

If provides a range. If there are no constraints, this object is omitted.

There are situations where a consumer might exceed Max, but Max is the maximum number that applies to the Then clause. For example, the offer might be structure, “buy up to 5 from this list and get the same number for free.” In this case Max=5 and the most they can redeem free is 5. However, if the user wishes to buy 6, presumably they won’t be prevented from doing so.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **IfNumberOf-type** |  |  |  |  |
| Min |  | Minimum number of objects required to satisfy if condition. | xs:nonNegativeInteger |  |
| Max |  | Maximum number of objects needed to satisfy if condition. Default is unlimited. | xs:positiveInteger | 0..1 |

## Then-type

The Then-type complex type defined the THEN clause as described in Section 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **Then-type** |  |  |  |  |
| ActionTimeframe |  | Time period where consumer can take action on this offer. | promo:When-type | 0..1 |
| Action |  | Action associated with offer. | promo:Action-type |  |
| Terms |  | Terms associated with offer. | promo:Terms-type |  |
| NumberOf |  | Quantity of Objects associated with offer. If absent, quantities are unlimited. | promo:ThenNumberOf-type | 0..1 |
| Objects |  | Objects associated with offer. | promo:Objects-type |  |
| ResultTimefreame |  | Assuming consumer exercises offer, period when offer take effect.  | promo:When-type | 0..1 |

### ThenNumberOf-type

Min, Max and QuantityMultipler determine the number of Objects associated with the THEN clause offer.

Min and Max specify the range of objects allowed. QuantityMultipler relates the IF clause to the THEN clause. For example, if a user can “buy up to 5 and get as many free” then QuantityMultipler is 1. @roundUp is used to indicate whether to round up or down (e.g., if quantity is 5, QuantityMultiplier is 0.5, and @roundUp=“true” the quantity will be 3).

When Min, Max and QuantityMultiplier are be used together, the most constrained limit applies. For example, if Max is 2 and QuantityMultiplier yields 3, then Max of 2 applies.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **ThenNumberOf-type** |  |  |  |  |
| Min |  | Minimum number of objects | xs:nonNegativeInteger | 0..1 |
| Max |  | Maximum number of objects. | xs:positiveInteger | 0..1 |
| QuantityMultiplier |  | Multiplier for quantity of objects in IF clause.  | xs:float | 0..1 |
|  | roundUp | If ‘true’ round up the product of the original quantity and QuantityMultiplier; otherwise round down. | xs:boolean | 0..1 |

## Combinations-type

Combinations-type applies when multiple IF elements and/or multiple THEN elements are present. It defines how to combine IF clauses and how to combine THEN clauses.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **Combinations-type** |  |  |  |  |
| IfCombination |  |  | promo:IfCombination-type | 0..1 |
|  |  |  | promo:ThenCombination-type | 0..1 |

### IfCombination-type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **IfCombination-type** |  |  |  |  |
| TotalMin |  |  | xs:positiveInteger | 0..1 |
| TotalMax |  |  | xs:positiveInteger | 0..1 |
| AllIfConditionsRequired |  |  | xs:boolean | 0..1 |

### ThenCombination-type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element**  | **Attribute** | **Definition** | **Value** | **Card.** |
| **ThenCombinatinos-type** |  |  |  |  |
| TotalMin |  |  | xs:positiveInteger | 0..1 |
| TotalMax |  |  | xs:positiveInteger | 0..1 |
| AllThenConditionsAllowed |  |  | xs:boolean | 0..1 |

# Patterns and Examples

Promotions come in a wide variety of forms. However, frequently used promotions tend to follow a relatively small number patterns.

The following table lists some frequently used promotions and how they can be generalized:

|  |  |
| --- | --- |
| **Example** | **Generalization** |
| Buy one, get one freeBuy 3 at full price, get 3 more for $5 | Buy n or more of m, price of n is x |
| buy three from this list for $15 | Buy n of m for a fixed price |
| Buy three or more titles for $5 each | Buy n or more of m, pay x/title |
| Buy three or more get 40% off | Price of n or more of m, is y% discounted |
| Buy three at full price, subsequent are 1/2 off | Buy n or more of m, price of 1 through m-n is y% discounted |

## Format for Templates

We’ll be using the following structure for illustration purposes and XML as the actual template.

 **IF
 <action taken>
 ON <number of>
 <objects>
THEN
 DURING <action timeframe>
 TAKE <action>
 WITH <terms>
 ON <number of>
 <objects>
 AT <result timeframe>**

## Encoding frequently used promotion patterns

### Buy *x* Get *y* at Terms

Those will be presented in the following form:

|  |  |  |
| --- | --- | --- |
| Parameter | Value | Example (buy one, get one free) |
| IF |  |  |
|  | <action taken> | EST (license type) | EST |
|  | <number of> | Number of objects to be bought to trigger offer | 1 |
|  | <objects> | Objects or objects to buy | {list of titles} |
| THEN |  |  |
|  | <action timeframe> | optional |  |
|  | <action> | EST (license type) | EST |
|  | <terms> | Terms of purchase | $0 |
|  | <number of> | Number titles to buy | 1 |
|  | <objects> | Object or objects to buy | {list of titles} |
|  | <result timeframe>  | optional |  |

<promo:Promo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:md="http://www.movielabs.com/schema/md/v2.5/md" xmlns:promo="http://www.movielabs.com/schema/promo/v1.1/promo">

 <promo:Type>BOGO</promo:Type>

 <promo:Offer>

 <promo:If>

 <promo:Action>EST</promo:Action>

 <promo:NumberOf>

 <promo:Min>1</promo:Min>

 <promo:Max>1</promo:Max>

 </promo:NumberOf>

 <promo:Objects>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02001/</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02002</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02003/</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02004</promo:Assets>

 </promo:Objects>

 </promo:If>

 <promo:Then>

 <promo:Action>EST</promo:Action>

 <promo:Terms>

 <promo:PriceInclusive>true</promo:PriceInclusive>

 </promo:Terms>

 <promo:NumberOf>

 <promo:Max>1</promo:Max>

 </promo:NumberOf>

 <promo:Objects>

 <promo:IncludeIfObjects exclusive="true">true</promo:IncludeIfObjects>

 </promo:Objects>

 </promo:Then>

 </promo:Offer>

</promo:Promo>

#### Variations on quantities

If the offer were *Buy three-or-more, get one free*, the IF NumberOf would look like this.

<promo:NumberOf>

 <promo:Min>3</promo:Min>

 <promo:Max>3</promo:Max>

</promo:NumberOf>

If it were 3 or more, the Max can be excluded:

promo:NumberOf>

 <promo:Min>3</promo:Min>

</promo:NumberOf>

If it were get ‘up to 3’, the Then NumberOf would look like the following (3 or fewer). Note that by excluding Min, the consumer may ignore free titles.

<promo:NumberOf>

 <promo:Max>3</promo:Max>

</promo:NumberOf>

#### Variations on Objects

If the “If” objects and the “Then” objects were different, the “Then” Objects element would include different titles. For example,

<promo:Objects>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:03005/</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:03006</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:03007/</promo:Assets>

</promo:Objects>

The object received need not be a movie title. Promos can reference benefits in the form of cash, coupon, EAN/UPC or ISRC (music), or more generically an ID (structured) or URL.

This example includes three ALIDs, a UPC code in GTIN format and an ISRC code referring to a song:

 <promo:Objects>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02001/</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02002</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02003/</promo:Assets>

 <promo:GTIN>0000001234554321</promo:GTIN>

 <promo:ISRC>US-S1Z-10-000006</promo:ISRC>

 </promo:Objects>

Sometimes an object could mean any title from a studio or channel. Rather than listing specific each title, the channel can be specified using GroupingEntity. In the following example, the channel is Sofaspud Films:

<promo:Objects>

 <promo:GroupingEntity>

 <md:Type>publisher</promo:Type>

 <md:GroupingIdentity>Sofaspud Films</md:GroupingIdentity>

 <md:DisplayName>Sofaspud Films</md:DisplayName>

 </promo: GroupingEntity >

 </promo:Objects>

#### Variations on Terms

The original shows “free”. Following are some examples of different terms.

If the terms were a fixed price (e.g., $5), Terms would look like the following:

<promo:Terms>

 <promo:UnitPrice currency="USD">5</promo:UnitPrice>

</promo:Terms>

If terms were ½ off (i.e., multiply price by 0.5), it would look like this:

<promo:Terms>

 <promo:DiscountMultiplier>0.5</promo:DiscountMultiplier>

</promo:Terms>

#### Variations on timeframe

If the consumer would not be able to initiate the offer (e.g., buy a discounted title) until after some period (in this example, one week), the following would be added to “Then”. Note that the condition is “Now” which means the moment of the transaction, as opposed to “Today” which means the calendar day.

<promo:ActionTimeframe>

 <promo:Condition lag="P7d">Now</promo:Condition>

</promo:ActionTimeframe>

If the consumer would not receive the title until later, ResultTimeframe would be included, such as the following. Note that this is how a release date would be indicated (actual date included)

<promo:ResultTimeframe>

 <promo:NoEarlierThan>2017-08-16T00:00:00Z</promo:NoEarlierThan>

</promo:ResultTimeframe>

### Offer with no precondition

It is possible to offer the Then part, without any preconditions. This is useful for offering products grouped together. In the simplest case, this is equivalent to an Avails Bundle. However, the Promotion structure allows the inclusion of items (e.g., T-shirts) not otherwise available through an EMA Avail, and it supports additional timing preconditions and postconditions.

In this example, $25 gets the consumer three movies, a t-shirt (UPC as GTIN) and a song (ISRC).

<promo:Promo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:md="http://www.movielabs.com/schema/md/v2.5/md" xmlns:promo="http://www.movielabs.com/schema/promo/v1.1/promo">

 <promo:Type>LOT</promo:Type>

 <promo:Offer>

 <promo:Then>

 <promo:Action>EST</promo:Action>

 <promo:Terms>

 <promo:UnitPrice currency="USD">25</promo:UnitPrice>

 </promo:Terms>

 <promo:Objects>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02001/</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02002</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02003/</promo:Assets>

 <promo:GTIN>0000001234554321</promo:GTIN>

 <promo:ISRC>US-S1Z-10-000006</promo:ISRC>

 </promo:Objects>

 </promo:Then>

 </promo:Offer>

</promo:Promo>

### Get *subset of assets* for terms

In cases above, a consumer can get one Object or all of the Objects. This section describes more complicated scenarios.

#### Any n of m

If the Action applies to a subset of the Objects, this is encoded using Min and Max.

Assuming they can acquire ‘up to’ a number, Min can be excluded. In this example, up to three.

 <promo:NumberOf>

 <promo:Max>3</promo:Max>

 </promo:NumberOf>

If there is an explicit minimum, then both Min and ax are used. In this example, two to five:

 <promo:NumberOf>

 <promo:Min>2</promo:Min>

 <promo:Max>5</promo:Max>

 </promo:NumberOf>

If the quantity is unlimited, Max is excluded. The following is “at least” two:

 <promo:NumberOf>

 <promo:Min>2</promo:Min>

 </promo:NumberOf>

If there are no limits, NumberOf is excluded.

#### Same as number of objects in “IF” (or proportional)

If the Action applies to the same number of Objects as in the IF term (e.g., buy *three* get *three* free), or the number is proportional (but *four*, get *two*), QuantityMultiplier is used.

This would allow the same number objects

 <promo:NumberOf>

 <promo:QuantityMultipler>1</promo:QuantityMultiplier>

 </promo:NumberOf>

If they can buy half as many, a QuantityMultiple of 0.5 is used. This defaults to round down (e.g., 5 becomes 2), but with roundUp=”true”, it rounds up (e.g., 5 becomes 3).

 <promo:NumberOf>

 <promo:QuantityMultipler roundUp=”true”>0.5</promo:QuantityMultiplier>

 </promo:NumberOf>

#### Specific combinations

This previous examples assume quantity choices are from a single lot. However, offers can be in the form, “up to three of these *and* two of those”, or “up to three of these *or* two of those.”)

NOTE: Admittedly, this is more complicated than many will want to implement. However, for completeness a method is included. However, to avoid overly complicating simpler cases, not every imaginable combination is supported (i.e., no parenthesis).

These cases are handled with multiple instances of IF or THEN clauses. These examples show THEN clauses.

An offer of up to two 2 movies, one t-shirt and one song is shown below. AllThenOffersAllowed indicates whether all of the Then clauses apply or just one. The example shows AllThenOffersAllowed=“true” which means the consumer gets all of them. If the choice is for 2 moves, one t-short *or* one song, then AllThenOffersAllowed must be “false”.

<promo:Promo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:md="http://www.movielabs.com/schema/md/v2.5/md" xmlns:promo="http://www.movielabs.com/schema/promo/v1.1/promo">

 <promo:Type>LOT</promo:Type>

 <promo:Offer>

 <promo:If>

…

 </promo:If>

 <promo:Then>

 <promo:Action>EST</promo:Action>

 <promo:Terms>

 <promo:PriceInclusive>true</promo:PriceInclusive>

 </promo:Terms>

 <promo:NumberOf>

 <promo:Min>2</promo:Min>

 </promo:NumberOf>

 <promo:Objects>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02001/</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02002</promo:Assets>

 <promo:Assets>md:alid:mpm.sofaspudfilms.com:02003/</promo:Assets>

 </promo:Objects>

 </promo:Then>

 <promo:Then>

 <promo:Action>EST</promo:Action>

 <promo:Terms>

 <promo:PriceInclusive>true</promo:PriceInclusive>

 </promo:Terms>

 <promo:Objects>

 <promo:GTIN>0000001234554321</promo:GTIN>

 </promo:Objects>

 </promo:Then>

 <promo:Then>

 <promo:Action>EST</promo:Action>

 <promo:Terms>

 <promo:PriceInclusive>true</promo:PriceInclusive>

 </promo:Terms>

 <promo:Objects>

 <promo:ISRC>US-S1Z-10-000006</promo:ISRC>

 </promo:Objects>

 </promo:Then>

 <promo:ThenCombination>

 <promo:AllThenOffersAllowed>true</promo:AllThenOffersAllows>

 </promo:ThenCombination>

 </promo:Offer>

</promo:Promo>

## Additional Patterns

### Tiered offers

Some offers are of forms like this: Spend $100 get 10% off, spend $200 get 20% off, spend $300 get 30% off and a free T-shirt (not all three at once).

These are handled as three offers encoded as three Promo elements in a PromoList. All would have the same PromoList/@promoGroup label indicating these are all part of the same offer. For example

* Promo: IF $100 THEN 10%
* Promo: IF $200 THEN 20%
* Promo: IF $300 THEN 30% and THEN T-Shirt

The key is to structure the Promo instances correctly. For example, an offer such as, “Spend $100, get 10% off, spend an additional $100 get a T-shirt” is more correctly, “Spend $100, get 10% off; and spend $200 get 10% and a T-shirt”.

## Other

[CHS: Check for: Gifting, earning (watching, playing, accumulating, contest/raffle). ]