OpenCable™ Specifications
Stewardship and Fulfillment Interfaces

Campaign Information Package Specification

OC-SP-SaFI-CIPv3.0-120307

Issued

Document Status Sheet

Document Control Number:	OC-SP-SaFI-CIPv3.0-120307			
Document Title:	Campaign Information Package Specification			
Revision History:	I01 – Released 6/26/09 v1.1 – Released 7/2/10 v2.0 – Released 1/31/11 v3.0 – Released 3/7/12			
Date:	March 7, 2012			
Status:	~~Work in Progress~~	~~Candidate~~	Issued	~~Closed~~
Distribution Restrictions:	~~Author Only~~	~~CL/Member~~	~~CL/ Member/ Vendor~~	Public

Key to Document Status Codes:

Work in Progress	An incomplete document, designed to guide discussion and generate feedback that may include several alternative requirements for consideration.
Draft	A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
Issued	A stable document, which has undergone rigorous member and vendor review and is suitable for product design and development, cross-vendor interoperability, and for certification testing.
Closed	A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

Trademarks:

Contents

Figures

Tables

1 SCOPE

1.1 Introduction and Purpose

This document specifies the data model and protocols that comprise the MSO Campaign Information Package Interface.

1.2 Requirements

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

"SHALL"	This word means that the item is an absolute requirement of this specification.
"SHALL NOT"	This phrase means that the item is an absolute prohibition of this specification.
"SHOULD"	This word means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
"MAY"	This word means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

2 REFERENCES

2.1 Normative References

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

[CONTENT 3.0] CableLabs Content 3.0 Specification, MD-SP-CONTENTv3.0-I01-100812, August 12, 2010, Cable Television Laboratories, Inc.

[CIPWSDL] OC-SP-SaFI-CIP-3.0.0.wsdl, March 7, 2012, Cable Television Laboratories, Inc.

[CIPXSD] OC-SP-SaFI-CIP-3.0.0.xsd, March 7, 2012, Cable Television Laboratories, Inc.

[COMXSD] OC-SaFI-COM-3.0.0.xsd, March 7, 2012, Cable Television Laboratories, Inc.

[MHP 1.1.2] DVB Multimedia Home Platform (MHP) Specification 1.1.2. http://www.mhp.org/mhp_technology/mhp_1_1/mhp_a0068r1.zip

[RFC 4122] IETF RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace, July 2005.

[RFC 4648] IETF RFC 4648, The Base16, Base32, and Base64 Data Encodings, October 2006.

[SCTE 130-2] SCTE 130-2 2008a, Digital Program Insertion–Advertising Systems Interfaces Part 2–Core Data Elements.

[SCTE 130-3] SCTE 130-3 2010, Digital Program Insertion–Advertising Systems Interfaces Part 3–Ad Management Service (ADM) Interface.

[WSDL] Web Services Description Language (WSDL) Version 2.0 Part 0: Primer W3C Recommendation, 26 June 2007.

Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Recommendation, 26 June 2007.

Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts, W3C Recommendation, 26 June 2007.

[XML/SCH] XML Schema Part 0: Primer Second Edition, W3C Recommendation, 28 October 2004.

XML Schema Part 1: Structures Second Edition, W3C Recommendation, 28 October 2004.

XML Schema Part 2: Datatypes Second Edition, W3C Recommendation, 28 October 2004.

2.2 Informative References

This document uses the following informative references:

[CIP EXMPL] OC-SP-SaFI-CIP-3.0.0-example1.xml, March 7, 2012, Cable Television Laboratories, Inc.

[CIP HTML] OC-SP-SaFI-CIP-3.0.0.html, March 7, 2012, Cable Television Laboratories, Inc.

[IAF] Interactive Application Fulfillment Summary Interface Specification, OC-SP-SaFI-IAFv3.0-120307, March 7, 2012, Cable Television Laboratories, Inc.

[IAM] Interactive Application Messaging Specification, OC-SP-SaFI-IAMv3.0-120307, March 7, 2012, Cable Television Laboratories, Inc.

[SCTE 35] ANSI/SCTE 35 2011, Digital Program Insertion Cueing Message for Cable.

[SMS] Service Measurement Summary Interface Specification, OC-SP-SaFI-SMSv3.0-120307, March 7, 2012, Cable Television Laboratories, Inc.

2.3 Reference Acquisition

- Cable Television Laboratories, Inc., 858 Coal Creek Circle, Louisville, CO 80027; Phone +1-303-661-9100; Fax +1-303-661-9199; http://www.cablelabs.com/
- Internet Engineering Task Force (IETF) Secretariat, 48377 Fremont Blvd., Suite 117, Fremont, California 94538, USA, Phone: +1-510-492-4080, Fax: +1-510-492-4001; http://www.ietf.org/
- SCTE - Society of Cable Telecommunications Engineers Inc., 140 Philips Road, Exton, PA 19341 Phone: 610-363-6888 / 800-542-5040; Fax: 610-363-5898; http://www.scte.org/
- W3C, http://www.w3.org/

3 TERMS AND DEFINITIONS

This specification uses the following terms:

Affiliate	An operational entity that performs SaFI operations with one or more MSOs.
Bundle	A structured set of placements in a specific campaign, expressed individually at one or more indicated MSOs and syscodes. The relationship between the placements that forms the basis of a bundle is beyond the scope of this specification, but may be indicated by an included product family value.
Campaign	Provides a set of delivery plans and/or placement directions for one or more MSOs by specific systems (Syscodes) within an MSO's footprint,. A Campaign is negotiated, purchased, and managed as an entity via campaign planning and management tools that are not in scope for the MSO interfaces. The campaign is typically expressed as one or more products from predefined product families that are defined in Bundles for placement, reporting, and operational status updates by MSO delivery and/or processing systems.
Enhanced Program Sequence ID	An integer that is one element of the globally-unique identifier for a Bundle or Bundle component.
GeoCode	Geographic Code: the geographic region that this service measurement message represents. The value in this element may indicate a ZIP Code, MSO syscode, or other encoded regional identifier.
MSO Order	The part of a Campaign Information Package (CIP) that falls within a specific MSO's advertising footprint.
Placement	A specific presentation of one or more advanced advertising assets at some advertising placement opportunity. In CIP, a data structure that supplies the definition of the conditions under which a placement may be executed.
Programmed Event ID	A UUID that is one element of the globally-unique identifier for a Bundle or Bundle component.
Service Measurement	Information about the reach and usage of a campaign.
Stewardship and Fulfillment Interfaces	A collection of interfaces defined by CableLabs to support advanced services on multiple cable systems. The CIP Interface is one of the SaFI interfaces.
Syscode	A four-character, predefined code that represents one or more specific cable plant networks and their corresponding geographical area.
System Order	The part of an MSO Order that falls within a single zone-specific syscode. In simple cases, all the Bundles of the Campaign will appear within each System Order; however, this may not be true due to site capabilities, or when targeting is applied.
Top Level Bundle	A bundle that is immediately contained within a syscode, as opposed to a bundle contained within a parent bundle. With respect to the CIP Interface top level bundles are the independent units of communication and update for campaigns.

4 ABBREVIATIONS AND ACRONYMS

This specification uses the following abbreviations:

AMB	Application Message Block
ARB	Application Report Block
CIP	Campaign Information Package
EPSID	Enhanced Program Sequence ID
EpType	Enhancement Package Type
ETV	Enhanced Television
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol over Secure Sockets Layer (SSL)
PEID	Programmed Event ID
RFI	Request For Information
SaFI	Stewardship and Fulfillment Interfaces
SOAP	Simple Object Access Protocol; as of SOAP 1.2, this no longer represents an acronym
STB	Set-Top Box
WSDL	Web Services Description Language

5 OVERVIEW

5.1 General Context

The MSO Campaign Information Package defines a set of web services implemented between a cable advertising affiliate and one or more MSOs, or between a campaign manager and other system components within an MSO. These web services coordinate the fulfillment of a campaign with all MSO systems that have a direct role in the preparation, delivery, execution, or reporting of the campaign's elements. The primary SaFI interface delivers CIP data using a publish/subscribe model. This data provides a single, self-consistent description to an MSO of the actions they are expected to perform for the campaign, to characterize all the resources required for those actions, and to communicate readiness and results for those actions. In all cases, the actual performance of actions described is solely within the domain of the MSO. In a general sense, the primary MSO functions that utilize the CIP are:

- Placement activities, in which the CIP defines the actions, context, and criteria for placement decisions.
- Reporting activities, in which the CIP characterizes the reporting required.
- Operational status reporting, in which the CIP identifies the content and frequency of the expected status.

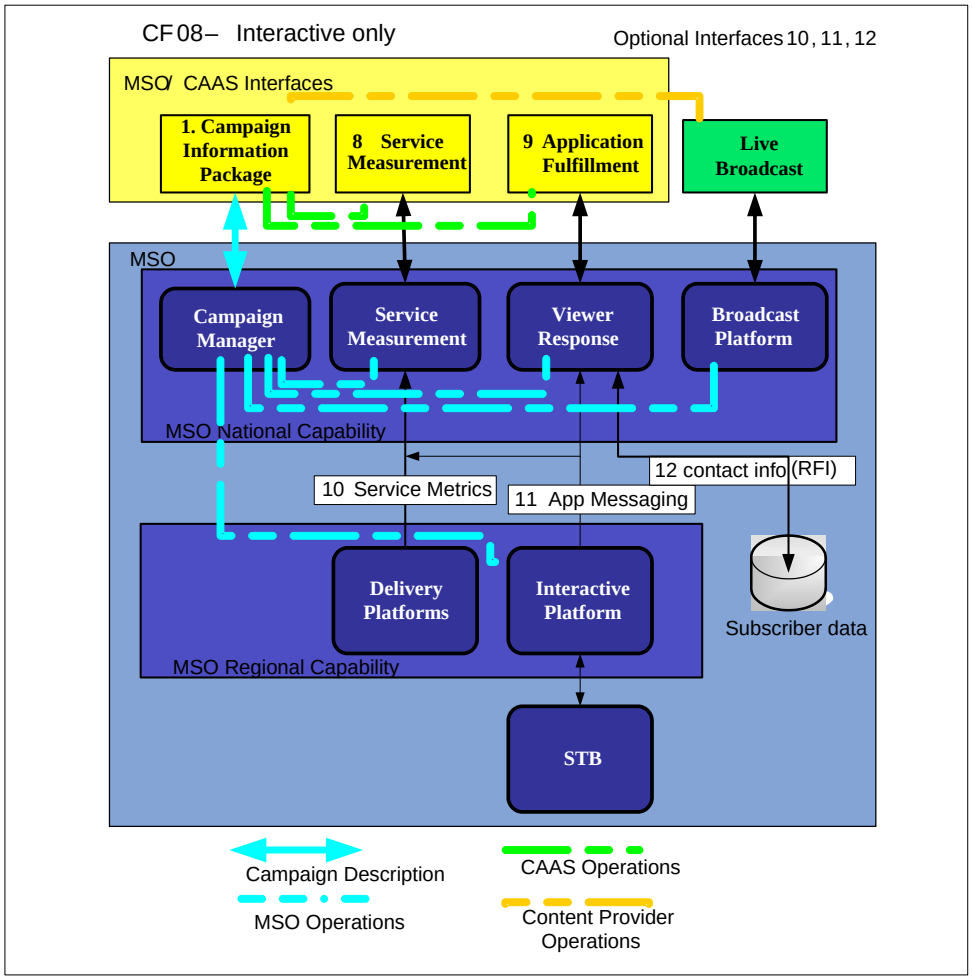

Figure 5-1 - Context of the Campaign Information Package Interface

This specification describes a Campaign that informs the MSO system elements responsible for application filtering, measurement, and fulfillment. The Campaign Information Package (CIP) contains the Campaign details, and is delivered to or within an MSO over the defined interface. This is a subset of the broader campaign information managed between the Affiliate and the MSO's systems, which also includes the operational characterization for the Affiliate and interactions with the agency, advertising customer, programmer, and any national third parties that may be involved. The CIP identifies the MSOs and the syscodes within each MSO that are to participate in a campaign, ensuring that an MSO system receives or processes only the portions of a campaign that are relevant to its operation.

The Campaign Information Package conveys the instructions for content and application placement within Placements, and collects Placements within Bundles. The CIP document consists of a set of top level bundle definitions, where a bundle is an originator-defined container for placement decisions. These bundles are fundamental units of the CIP, and are the primary transaction units, in that they are individually and independently communicated, updated, or deleted. The CIP document is simply a container for one or more top level bundles. This structure permits the web services to operate on individual real-time updates of bundles or on "batches" of bundles in periodic updates.

5.2 Specification Components

This specification consists of the following elements:

1. This specification document, which is normative except as otherwise noted.
2. The associated XML schema file, OC-SP-SaFI-CIP-3.0.0.xsd [CIPXSD], which is **normative** both in terms of XML validation and documentation, except as may be otherwise noted.
3. The associated WSDL file, OC-SP-SaFI-CIP-3.0.0.wsdl [CIPWSDL], which is **normative**.
4. The associated XML schema document set contained in OC-SP-SaFI-CIP-3.0.0.html [CIP HTML], which is **informative**.
5. The associated CIP example file, OC-SP-SaFI-CIP-3.0.0-example1.xml [CIP EXMPL], which is **informative**.
6. The CIP profile set, OC-SP-SaFI-CIP-3.0.0-profiles, an **informative** set of CIP document profiles by use case, which is expected to be published shortly after this specification.

6 MSO CAMPAIGN INFORMATION PACKAGE INTERFACE REQUIREMENTS

NOTE: Most of the requirements outlined in previous versions of the CIP specification are now contained in the CIP XML schema ([CIPXSD]). This section provides additional information, as well as a small number of requirements.

6.1 Transmission Protocol

The CIP coordination protocol uses a subscription-based consumer/producer model for CIP documents. In this model, there are one or more CIP producer sites and one or more CIP consumer sites. (*Site* is used here as an address providing the defined set of web services.) Both the producer and consumer advertise web services in support of the CIP coordination protocol. The physical or logical location of these sites is not defined.

CIP documents are created and modified only at the producer sites. It is expected that there will be a number of these CIP producer sites, each representing separate sets of campaigns. Each producer site receives and retains subscriptions, publishes CIP documents containing changes to the campaign against those subscriptions, and responds to requests for its CIP documents.

Each MSO will have one or more CIP consumer sites. Each consumer site is responsible for CIP processing on behalf of one or more syscodes, and subscribes on behalf of those syscodes. Following registration, the site receives documents that represent new or changed requirements for one or more of its subscribed syscodes. A consumer site can also request a copy of one or more CIP documents.

6.1.1 Messages

The web service implementation between CIP producers and consumers utilizes the message set described here. These messages are defined in the schema.

6.1.1.1 Element SetRegistrationRequest, SetRegistrationResponse

The SetRegistrationRequest message SHALL be issued by a consumer site to a publication site to add or remove that consumer site from the distribution site's subscription list, or to maintain the set of syscodes which the consumer site presents.

The consumer site is represented by the URL to be used for published messages, and the subscription list holds the subscribed syscodes. If the syscode list is empty, the entire set of syscodes predefined for the subscriber URL (via an unspecified out-of-band mechanism) is implied. The message is processed exactly as though it had been issued as containing that implied list of syscodes, except that it performs no operation if the implied list itself is empty.

6.1.1.2 GetRegistrationRequest, GetRegistrationResponse

The GetRegistrationRequest message MAY be issued by the consumer site to request the current list of syscodes that are valid at the producer site for a subscriber URL. The list is the same form as the SetRegistrationRequest message's list.

6.1.1.3 ReadRequest, ReadResponse

The ReadRequest message SHALL be issued by a consumer site to obtain the current copy of a CIP document. The consumer site does not have to be on the producer site's subscription list in order to issue a ReadRequest. The ReadRequest can identify a specific document or supply parameters to identify a set of documents. A parameterized query will return all documents meeting the parameters.

A valid ReadRequest receives a ReadResponse that includes the selected CIP document.

6.1.1.4 UpdateNotice, UpdateResponse

The UpdateNotice message publishes one or more unsolicited updates to campaign data from a CIP publisher to a CIP subscriber. A publication site SHALL issue the UpdateNotice message to deliver to subscribers one or more new or revised top level bundles contained in a Campaign Information Package. The frequency with which UpdateNotice messages are issued, and the scope of updates which are contained in one UpdateNotice, are left to the implementation.

If UpdateNotice messages are received with the isRetry attribute set to true, they may be duplicates. Receivers SHALL NOT be affected by receiving duplicate update messages.

6.2 Data Model

The MSO Campaign Information Package data model is defined by the normative XML schema contained in [CIPXSD]. This section presents additional informative material for various data elements.

6.2.1 Supported Product Families

The CIP data model is intended to provide support for certain products within the following product families:

1. Consumer application advertising products, including Vote/Poll and Request for Information (RFI) application products
2. On-demand advertising insertion products
3. Traditional and enhanced linear advertising insertion products
4. Combinations of the above

In addition, the CIP model is intended to support delivery of any advertising product on any MSO subscriber device platform.

6.2.2 Interactive Products

Figure 6-1 shows a timeline for a typical sports event with a total of three enhancement products applied during the event. These are Active Voting Packages 1 and 2, and an RFI Package. This diagram will be used to illustrate examples in this section.

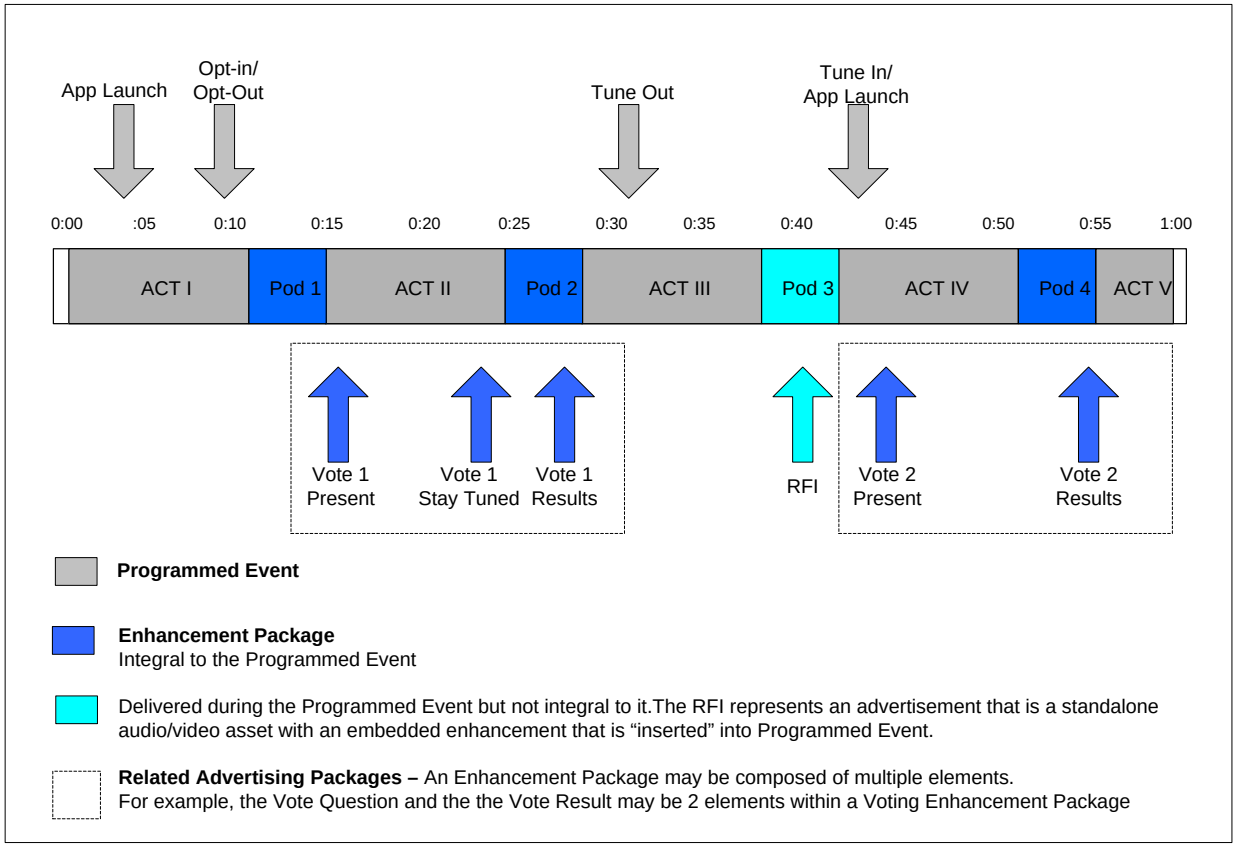

Figure 6-1 - Identification Models and Set-top Message Identification

6.2.3 Application Management Identifiers

This section describes the primary identifiers and their relation to return message processing.

The return message host, nominally a set-top box, supplies a device-specific identifier, but all other context for processing return messages must be supplied within the application. For processing, an arbitrary returned message must be placed in the correct context when many similar messages are arriving from the same or different applications in different programming, or repeats of the same application in one program.

Two identifiers are used to provide deterministic mapping of every message to the business context in the campaign and its parsing and processing instructions. The first is the Programmed Event ID (PEID), a universally-unique identifier for a specific context within the CIP. Within that context, the enhancement components may be identified with a local index that is unique within the context of that specific PEID: this is the Enhancement Package Sequence ID (EPSID).

Return message processing is defined by a structure named a Phase, which is always labeled with a PEID and EPSID and identifies a specific messaging event. In order to bind the message to processing instructions, there must be a CIP Phase such that the return message type matches the Phase event and the message identifiers match the identifier set in the Phase. Within the Phase are one or more processing rules directing the message processing to be applied. The Bundle structure containing the Phase can also contribute context and additional processing.

6.2.4 Ad Insertion Products

This CIP data model provides for placement of one or more content files, nominally 'ads' or 'applications', into a linear or server-based offering of initial content, which is nominally 'entertainment'. Although it is not limited to

SCTE-130 delivery environments, this facility is intended to fully support implementation in an SCTE-130 environment that is performing operations defined for an SCTE-130 Part 3 ADS. In light of this, the CIP data model represents various elements using SCTE-130 compatible concepts and definitions.

6.2.5 Status

The only unit of update in the CIP data model is a top-level bundle. Whenever a change is made within a top-level bundle, the entire structure is republished in its new form, and only that current form is published. Since changes may be made in any child component, and changes might be additions, edits, or deletions, the recipient must be able to correlate the components of the current structure with those of any prior distribution to be able to perform the appropriate local updates.

This correlation is performed at two levels. Each major component contains a RecordIdAttributeGroup, which provides a persistent identifier consisting of PEID plus EPSID. At the first level, this allows correlation of current and prior components. In addition, each major component carries a RecordStatusAttributeGroup which provides revision data for that component. In particular, the revision number and record-revoked flag support updating the matched components at the second level. This revision control applies to the component proper and to any children that do not contain an independent RecordStatusAttributeGroup. A child that has its own RecordStatusAttributeGroup must be processed independently.

There are three update conditions:

1. If the current and prior components have the same revision number and the record-revoked flag is false, there has been no change in the controlled data; that is, the component and its controlled children continue to apply to the current revision. Note that although the component does not change, the recState in the RecordStatusAttributeGroup may be changed to put the component into a new intended state.
2. If there is a change in the revision number, the controlled data in the current revision replaces the corresponding data in the prior revision.
3. If there is no change in the revision number, but the record-revoked flag is true, the component no longer exists in the current revision. The component from the prior revision should be "removed", in the sense that it will not contribute to execution of the top-level bundle in its current revision. If the environment continues to execute the prior revision for some period of time, for example, while other components of the current top-level bundle are staged, those executions are unaffected by the change; it only applies to the current revision.

The ability of a consumer site to request all versions permits the history of an updatable section to be distributed, so a document consumer that has not received or retained a prior state can still compare the current state to the prior state, or to any older state that might be represented in some dependent systems.

Note that the adoption of a revision is assumed to be neither instantaneous nor simultaneous within or across operational sites. Consequently, any CIP update should be expected to result in the receipt of concurrent reporting messages derived from multiple revisions of the CIP. Feedback reporting includes the component revision number, which will allow the operational status of each revision to be tracked independently.

Since older reporting formats as of IAM did not include a CIP revision, and current implementations may not report the CIP revision, these cases can only be disambiguated by changes in the reporting identifiers: the PEIDs and EPSIDs. This requires that any CIP revision incorporating updates that can change the interpretation of reported data must terminate (Windup or Close) any Bundle or Placement containing those changes and create new updatable sections at revision one with distinct identifiers. In this case, the recRevoked flag will be set in the prior revision of the section, and the chosen recState will (eventually) control the permitted activity with respect to that revision.

RecordStatusAttributeGroup is an empty element that SHALL use the following attributes:

- recState (ElementStateType) is a required string with enumerated values, defined in the annotation block of the schema definition [CIPXSD]. This attribute describes the campaign owner's direction with respect to

the updatable section; recState is not intended to reflect real-time changes in campaign execution, but rather the owner's current intent. A newly-created updatable section MAY have any initial value; however, subsequent valid transitions between these states SHALL be as shown in Figure 6-2.

Table 6–1 - Record State Enumerated Values

Enumerated Value	Meaning
pnd (Pending)	Data is present, but may not be an operationally-complete or consistent set.
com (Committed)	Data is present, and is an operationally-complete and consistent set, though not necessarily final (data may be modified, even when active).
act (Active)	The data present is intended to be operational and the line item should be executed according to the contained parameters. The actual time for various executions will vary by operational role (e.g., inserters before presentations). This is a statement of intent, not the result of operational feedback.
pse (Paused)	Presentations must not be made based on this element. This is a temporary state that should return to Active, but may instead go to Windup.
wnd (Windup)	All executions related to the record should have ceased, but post-processing functions may continue. This includes upstream messaging within any pre-defined window.
cls (Closed)	All activity with respect to the record is terminated. No additional revisions of the element may be provided after it is closed.

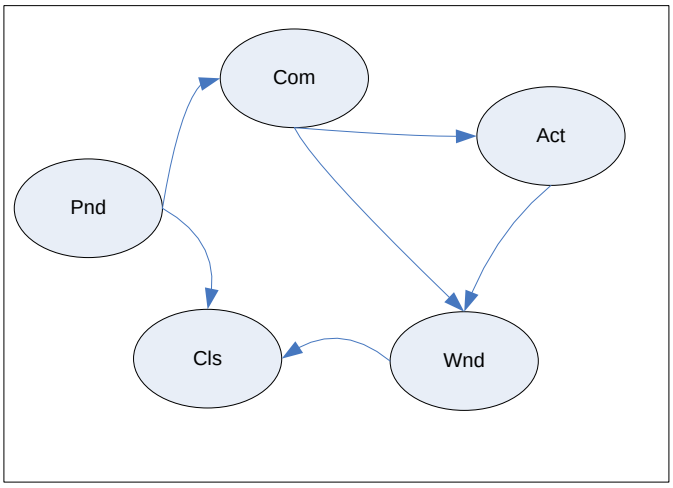

Figure 6-2 - Valid Record State Transitions

- recRevoked (RecordRevokedFlagType) is a required Boolean, and SHALL be set to false initially and modified to true if the updatable section is either deleted or superseded.
- revision (ElementRevisionNumberType) is a required non-negative integer that represents the revision number of this particular instance of the updatable section. The revision SHALL be incremented every time any component of the section with this PEID is modified in any way, excluding changes to the value of this Status element.
- revDat (ElementRevisionDateType) is required, and SHALL be the date and time the updatable section was last modified, including changes to the status element only.

6.2.6 General Organization

This section addresses the characteristics of the individual data elements that participate in the data model.

The overall organization of the MSO Campaign Information Package data is shown in Figure 6-3. This is a generally hierarchical model of a Campaign Information Package containing one or more MSO orders, each containing one or more System Orders, each containing one or more Bundles, each with its operational data contained in Placements. Qualifiers can appear at the Bundle or Placement, with Bundle qualifiers imputed to their placements. Assets are defined as children of Placements.

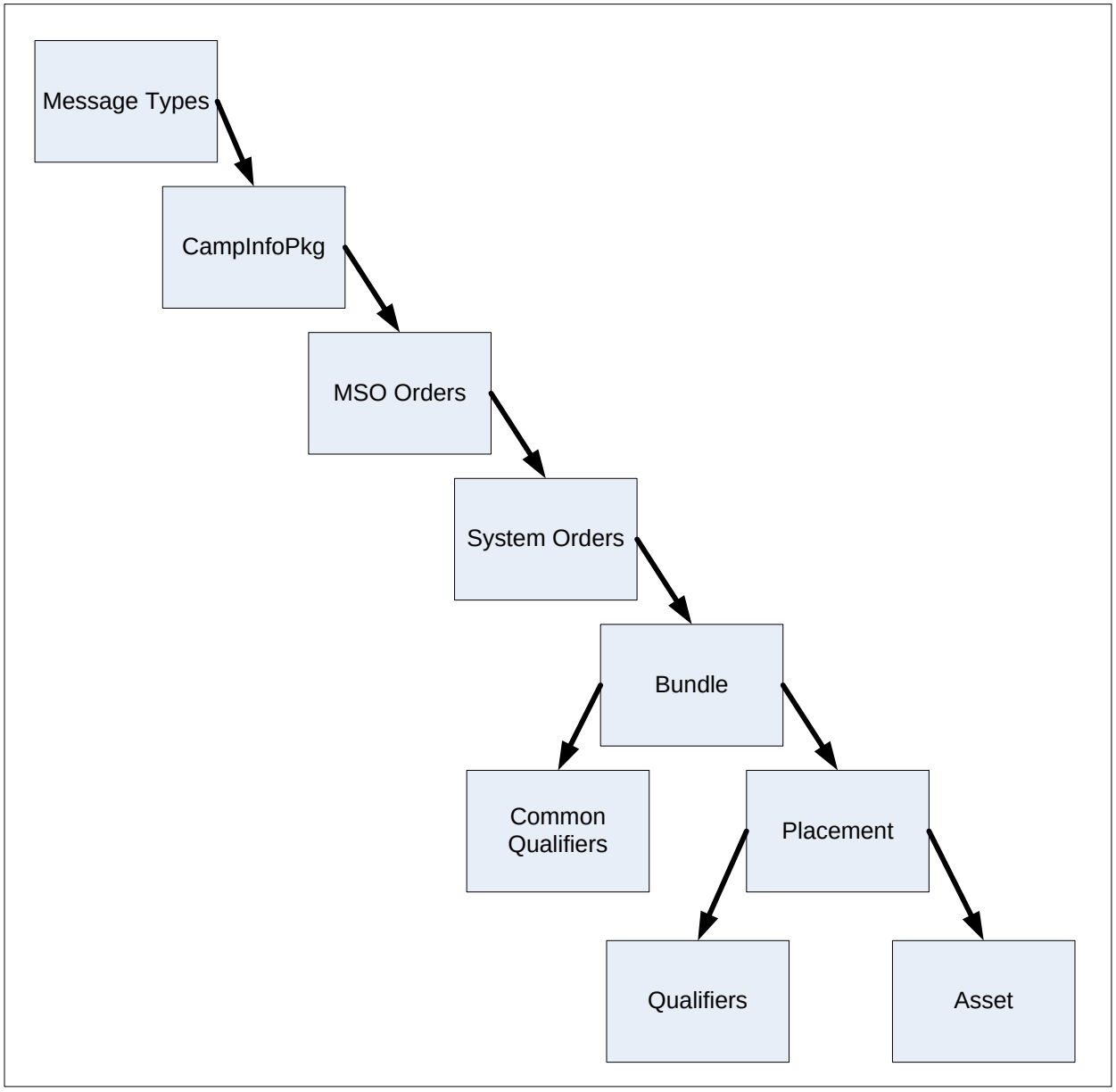

Figure 6-3 - MSO Campaign Information Package Organization

7 CAMPAIGN INFORMATION PACKAGE DATA MODELS (NORMATIVE)

The following data model documents are normative components of this specification.

The formal XML schema data model and data field semantic definitions are found in [CIPXSD].

The formal web service interface data definition is found in [CIPWSDL].

Annex A Qualifier Namespaces

The use of qualifiers namespaces is defined in [CIPXSD] under NamespaceType and TermType. The following qualifier namespaces are normative for execution environments supporting Campaign Information Package version 2 or higher. The values of the prefixes are informative, but recommended.

Figure A–1 - Qualifier Namespace Declarations

Prefix	Namespace	Description
xml	http://www.w3.org/XML/1998/namespace	Used for xml:lang attribute
xs	http://www.w3.org/2001/XMLSchema	XML foundation. See [XML/SCH], parts 1 and 2.
xsi	http://www.w3.org/2001/XMLSchema-instance	
cip	urn:cablelabs:safi:xsd:cip:3.0	[CIPXSD]
common	urn:cablelabs:safi:xsd:com:3.0	[COMXSD]
vod30	http://www.cablelabs.com/namespaces/metadata/xsd/vod30/1	[CONTENT 3.0] Provides VODContainerType and the root ADI3 element
adicore	http://www.cablelabs.com/namespaces/metadata/xsd/core/1	[CONTENT 3.0] Provides the base AssetType and AssetRefType
title	http://www.cablelabs.com/namespaces/metadata/xsd/title/1	[CONTENT 3.0] Provides TitleType
content	http://www.cablelabs.com/namespaces/metadata/xsd/content/1	[CONTENT 3.0] Provides all of the known content types
130core	http://www.scte.org/schemas/130-2/2008a/core	[SCTE 130-2]
adm	http://www.scte.org/schemas/130-3/2008a/adm	[SCTE 130-3]

Annex B Public URN Definitions

The following public URNs are reserved in the schemas:

cablelabs:safi:xsd:cip:3.0
cablelabs:safi:xsd:com:3.0
cablelabs:safi:xsd:iaf:3.0
cablelabs:safi:xsd:iam:3.0
cablelabs:safi:xsd:sms:3.0

Appendix I Implementation Notes (Informative)

This appendix provides an overview of new or substantially modified facilities addressed in this release, along with the expected application of those facilities.

I.1 Return data processing

The RetData and AppPhase elements from the 1.-2 (Dev2) schemas were omitted from the 2.0 release because they were inadequate to support some of the required 2.x use cases. They appear in a revised version in 3.0.

This version provides a Phase element, extended from the 1.-Dev2 AppPhase, which may be included in bundles and placements. Phases provide a facility for a campaign to execute business logic "plugins" within the CIP execution environment. This logic is contained in externally defined rule sets, which are collections of individual rules. Each rule is functionally a runtime method call with the rule set, called method, and call arguments supplied in the phase.

In 3.0 RC1, AppPhase was defined only to be invoked in response to an application IAM message. In 3.0, Phases may be invoked, or triggered, by a number of possible events in the CIP workflow. These are defined in the PhaseEventType in the schema, and represent roughly the creation or deletion of some CIP elements, the beginning or end of a flight window, the discovery or loss of an asset or application, evaluation of a placement during placement candidate processing, the selection of a placement as a result of placement candidate processing, and the receipt of a post-placement message (e.g., IAM or PSN). An occurrence of such a trigger event for a Phase causes the sequential execution of all of its rules. Each rule is essentially a method that:

- is an identified method of a predefined rule set;
- is defined in an interpreted language (without implication for implementations);
- is executed in the "below-the-line" CIP execution context;
- accepts run time arguments supplied in the CIP;
- has access to preconfigured read-only and read-write storage for named variables;
- and is able to generate output message (e.g., invoke web service client operations to send an IAF message).

All rules are predefined in the sense they must be provisioned before they are available for use. Individual rules are members of a rule set, which is identified by a repository URL, a UUID within that repository, and a version. The rule set can contain either zero or one rule for each legal invocation. In the CIP the set of repository URL, rule set UUID, and version serve to uniquely identify a rule.

I.1.1 Asset Phases

Asset phases have only two defined events: for the discovery or loss of an asset. These are to be interpreted as contemporaneous with the discovery or loss of an asset for purposes of State Feedback. When a placement is created, all its assets are defined to be in an unknown state. Any subsequent state change that results in the asset state becoming "ready" triggers a "found" event, and a state change that results in the asset state becoming "unknown" triggers a "lost" event.

I.1.2 Placement Phases

Phases of placements have four possible events; for the definition of the placement, for the destruction of the placement, the action of making a placement, and the receipt of placement result data, such as IAM, PSNs or other future platform elements. In addition, any child phase trigger can be propagated to the Placement.

Because an application is defined at the placement level, allowance for application structure must be provided in the phases of a placement. This goal is met by making Phase recursive, so one placement can define a hierarchy over several levels of phase. A selected phase element within a placement is processed by applying the input event the selected phase element. Then, if that element has an ancestor phase element that has the same event trigger, the ancestor element is processed. If not, the ancestor is skipped. This repeats until a top level phase node of the Placement is encountered. Note that this means phases in placement elements can actually have any event of an asset or application phase. Although they will not be invoked directly, since a placement does not define those events, they may be invoked during traversal of the path to the root node from an asset or application.

The Phase element provides the ID referenced by the execution environment (PEID/EPSID). It also has one or more ProcessRuleSelector elements that hold what was EpType (this is now a UUID named ruleSelectorId, and a set of actual arguments for the rule, passed as call-by-name. As an example, under CIP 1.1, the SMS summary for an application and an IAF report for that same application are both created from the same set of IAM messages. In one possible CIP 3.0 version, the IAM messages would reference the PEID/EPSID of a phase that had two rule selectors. One would characterize the SMS reporting and the other the IAF reporting. Alternatively, the CIP 3.0 phase could have a single rule that acted like the CIP 1.1 rule, and created both outputs. In both cases, the rule arguments would be defined to replace the attributes that were passed in the CIP 1.1 return data structure.

I.1.3 Bundle Phases

Phases of bundles have four possible events; for the creation of the bundle, for the destruction of the bundle, for flight begin within the bundle, and for flight end. These phases represent campaign-defined rollups of execution data. Since bundles represent product structure, these rollups of their associated phases also can represent product structure, if desired.

If a bundle phase element is selected, the processing recursion as described in placement also applies, propagating up any phase tree and bundle tree until a top level phase node of a top level bundle is encountered. Finally, a top-level phase element in a placement may also propagate updates to a node in a parent bundle phase tree using the parentPhasePeid and parentPhaseEpsid attributes. This allows placement events to be processed in bundles by permitting input messages processed against the placement phase to also be applied to that bundle phase.

Note that this means bundle nodes can also have any event of placement phase. Although they will not be invoked directly, since a bundle does not define those events, they may be invoked during traversal of the path to the root node from a placement.

The combination of data for each ProcessRuleSelector and its parent Phase include all reporting fields that were defined for a CIP 1.1 PlacementType + ReturnDataType + (AppMsgs or SmMsgs). This was intentional, in order to minimize the impact on current processing, other than the provision for multiple explicit rules and the propagation to parent (or related) phases.

I.1.4 Extensions

There are two extensions that did not appear in 2.Dev-1, args and named data storage.

Args are name-value pairs defined in the CIP rule element and supplied at run time to the processing rule, where name must match a formal argument of the rule and value replaces it in the execution of the rule. These are analogous to the formal and actual arguments of a method.

A rule may access predefined and named read-only and read-write storage for named variables. The naming uses a hierarchical form with levels separated by ".". This convention supplies scoping for the rule sets, e.g., by a campaign id, a rule set id, and variable id. One form of read-write store that must be available is one shared with term evaluation for the decision system. This permits rules to assign values to variables that are tested in placement decisions. The rule accesses this data store by name and variable name, as it does any data store. The qualifier term

accesses the shared data by identifying the source as a rule store, the namespace as the rule storage name, and the qualifier as the variable name.

I.2 Multiple execution platforms

Various deployment platform architectures for advanced advertising are possible, and two are currently projected. At this time, the primary distinction between platforms is in the business context of the Decision Support System. On what we call the Exe CIP platform (Executable CIP), the only form to date, decisions are made at the MSO level as defined in the CIP. On the PAR CIP platform (Preflight and Reporting), the decisions are made by an ADS at the affiliate level. On this platform, CIP data to support the execution of placements is not needed. Note that reporting here includes both Stewardship Reporting (SMSI) and Operational Reporting (CIP Feedback), described later in this document.

Some elements in a CIP are applicable only to the execution of placements, but not to preflight or reporting, These elements are not required in a PAR CIP, but would be present in a Exe CIP. As a first approximation, the elements that apply only to an Exe CIP are:

- The set of common qualifier and qualifier data elements
- Some attributes of Placement (i.e., priority, category, trick mode)
- Asset metadata data elements

Consequently, it would be possible to revise the CIP to structurally separate the data by platform. After consideration of the needs of PAR CIP sites, it was decided the redundancy of providing reporting and preflight data with the necessary bundle and placement structure did not justify creating distinct CIP structures. Consequently, PAR CIP sites will receive a normal CIP document, although execution information may be absent. Also, since placements and bundles are defined on a syscode basis, there will be no AssetReferences or AppReferences that have “apply” attribute values of “place” or “valAndPlace” in any placement for a PAR CIP syscode.

I.3 Preflight

Support for preflight readiness checking has been added in 3.0. This is through a ContentMgmt element which is now optional in both an AssetReference and AppReference, and so applies to one asset of one placement. The ContentMgmt element provides guidance on preflight requirements for its associated media or application reference. This element identifies a “residence window” during which the asset or application is expected to be available. These are specified by the “requiredBegin” and “requiredEnd” date-time attributes. Note that the begin time would normally lead the flight window by some interval to allow action to be taken on missing assets.

The ContentMgmt element also has a “deliveryDomain” attribute that can provide the CIP subscriber site with a hint about which servers should contain the asset. While usage has not been established, this might, for example, include a linear network name or VOD service name.

While the ContentMgmt defines the criteria for asset readiness, the actual state is reported in the StateFeedback element, covered in the next section.

I.4 Feedback

CIP feedback had been defined to return readiness and execution progress to the CIP originator. This is an operational facility, and is independent of the stewardship functions of the SaFI SMS interface. The FeedbackNotice and FeedbackResponse messages and corresponding services are the mechanism for the CIP subscriber to report to the CIP publisher.

A new StateFeedback element is present at a number of levels, including both bundles and placements. This element is optional in a published CIP. Its inclusion at any point in a CIP document indicates reporting is desired at that level. It may also provide a request for a FeedbackNotice, as described below.

State reporting is performed by making a copy of a published CIP and inserting (or expanding existing) StateFeedback elements where appropriate. The element must be included for any bundle or placement where the state has changed since the prior report.

As defined in the schema, the CipVer attribute on each bundle identifies the version number of the campaign data that generated the data, and will be incremented for CIP updates. This version number is included in the StateFeedback element. Since the life cycle of updates at CIP subscriber sites may extend well past the publication of a new version, two or more CIP versions may be active at the subscriber site at one time. A StateFeedback element identifies the CIP version to which it applies, and several may be present for distinct versions.

A reportInterval value in the StateFeedback requests periodic reporting of the feedback until the end of flight or an option report end time. It may also have a date-time attribute of "reportAsOf", which is reserved for use by the CIP publisher in an outbound CIP document, and indicates that the CIP publisher wishes a state update for the indicated CIP version of the element containing that StateFeedback. If the subscriber site has issued a FeedbackNotice that included that version of the bundle or placement and occurred subsequent to the reportAsOf time, the message may be ignored. Otherwise, a new update should be issued. The CIP publisher will never populate a StateFeedback element with elements or attributes other than cipVer, reportAsOf, reportEndTime, or reportInterval. A CIP subscriber will never populate the reportAsOf, reportInterval, or reportEndTime attributes.

A state-feedback report conveys four kinds of data: the overall record state of the bundle or placement, the readiness of all assets, the readiness of the execution environment, and optional statistics maintained by rules. The record state is reported in the same enumeration as the "recState" of the bundle or placement. *Note, however, that the downstream recState is an indication of a desired state, while the feedback recState is a report of actual state.* For example, a CIP update will not necessarily be issued at the start of a flight window to change the recState from "com" (committed) to "act" (active). However, the StateFeedback would report the change at that time. Once a given CIP version number has been reported with recState equal to "cls" (closed), no further reporting is expected for that version.

The asset readiness, where assets here include application references as well as asset references, is reported in an attribute named "mediaState". For a placement, this is a summary of the readiness of all assets of the placement. For a bundle, it is a summary of the readiness of all child bundles or child placements.

The values, ordered by decreasing precedence, are "unknown", "processing", and "ready". The "unknown" state is the initial state at the CIP publisher. The subscriber site must report "unknown" if any asset state is unknown, "processing" if any asset is undergoing processing (e.g., movement or preparation) and the others are ready, and "ready" *if and only if* all are ready.

Execution readiness is very similar, with the attribute being "executionState" and the same value options. The state applies to all operational systems necessary to execute the placement, or to execute all placements and bundles of the bundle, as appropriate. Operational systems would include at least stewardship reporting subsystems in a PAR CIP site, and also decision systems, carousels and splicers, SIS information, etc., in an EXE CIP site.

The optional statistics element defines a list of requested statistics that are maintained in a rule datamap. Each element in the list references a rule datamap variable, and the runtime implementation is responsible for evaluating that variable and inserting the result in the body of the statistic element.

I.5 Other notes

Reporting must support structured products, with examples being reports for targeting and for certain compound product functionality (e.g., "bookends"). This is met by permitting recursive, multilevel bundles. The specific bundle

structure will be pre-defined in operational guidelines and identified by the "ProductFamily" attribute in the topmost bundle. The role of lower-level bundles might be expressed in the same attribute, but that is undefined at this time. ProductMember is defined in placements for a similar purpose.

App attributes have been moved to an AppReference element inside placement. This is a relocation of these attributes, without change in name or semantics. This was done in recognition that applications may be pre-flight controlled assets as much as media, and also in anticipation of a single placement identifying more than one "application". Note that this is not necessarily limited to STB applications.

The 3.0 version permits an MsoOrder to consist of the choice of a single Orders element (for all syscodes) or a set of individual Syscode orders. This permits MSOs with a centralized service to receive only one Order element for those bundles that have no local differentiation. The data within an Order and SysOrder are identical; the only difference is in the element name and scope as defined above.

The 3.0 Placement adds several constructs either missing or present as placeholders in 2.0. These include flight window as an effective date time group, category management, and trick mode restrictions.

Appendix II Revision Summary from 2.0 (Informative)

II.1 Schema Modifications from 2.0-3.0

CIP version 3.0 is a major version, and as such does not require a deterministic EC history from the prior release. Several major revisions to the CIP schema from the 2.0-3.0 version are recorded here:

1. Introduce strawman recursion in bundle and add cip:childConcurrencyType.
2. Add preflight element in AssetReference.
3. Move ProductFamily to Bundle, add ProductMember at Placement.
4. Added Phase and ProcessRuleSelector, reference in Bundle and Placement.
5. Added StateFeedback reference in bundle and placement.
6. Moved most of PhaseType to ProcessRuleSelectorType.
7. Made product rule selector attributes optional so can deprecate and add anyAttribute, same for feedback attributes.
8. Mso/Sysorders and MsoOrder/Orders made [0,inf] from [1,inf].
9. Renamed DataProductSelector to ProcessRuleSelector.
10. Renamed PreflightData to ContentMgmt.
11. Added Ext element to ReadRequest.
12. Added ContentMgmt and Ext elements to AppReference along with apply attribute.
13. removed iamUrl and dataUrl attributes in ProcessRuleSelectorType.
14. Removed unreferenced GuidType.
15. In MsoOrder added a choice of the existing SysOrder or MSO global Orders.
16. Add documentation and Ext elements to StateFeedbackCountsType. All attributes except cipVer are now optional.
17. Define trial enumeration for TrickModeType.
18. Changed isClosed to recStatus, changed some StateFeedback attribute use.
19. Added FeedbackNotice and FeedbackResponse messages.
20. Added TrickModes element, defined trick mode attributes.
21. Added Notes to InvalidRequest, StateFeedback, FeedbackResponse, ReadResponse, UpdateResponse, SetRegistrationResponse, GetRegistrationResponse.
    Added MsgResultAttributeGroup to GetRegistrationResponse.
22. Add source to TermType.
23. Orders of MsoOrders is max 1, not unbounded.
24. Add AffiliateUrl to ProcessRuleSelector.
25. Add attribute invoked to PhaseType.
26. Add Phase to AppReferenceType, AssetReferenceType.
27. Remove Phase in ApplicationReference.
28. In qualifierSourceType, add ruleStore.
29. In Placement, removed trickMode.
30. Add types EffectiveDatesDayTimeType, EffectiveDatesDayTimeAttributeGroup.
31. In placement and bundle, add Flight.
32. In placement, added choice of (CategoryExclusionA or CategoryExt) for category logic and removed category attribute.
33. Add types CategoryExclusionAType, ExclusionScopeAType.
34. In placement, added separationConstraint.
35. Added SeparationConstraintType.
36. Added types PerformanceGoalAType, GoalAType, GoalAShape.
37. In placement, added choice of (PerformanceGoalA or GoalExt) for performance goal logic.
38. Renamed StateFeedbackCountType to StateFeedbackStatisticsType, added Statistic.
39. As a nameValueType, removed named count attributes.
40. Replaced ProcessRuleSelectorType in-line definition for Arg with new common NameValueType.

41. Removed AssetMetadataKeyType, AssetMetadataValueType.
42. Added durationSeconds attribute to placementType.
43. In RuleSelectorId removed affiliateUrl, fieldFilter, reportBegin, reportEnd, reportInterval.
44. In Bundle make Flight unbounded, defined as "min/max" of begin/end, "or" of days, times.
45. Defined TrickModeExclusionType, SpeedScaleType, TrickModeType.
46. Add TrickModeRestrictionType.
47. Need a way for a rule to disable/enable trick mode restrictions (replaces TrickModeScopeType).
48. Add Canoe's category exclusion using CategoryExclusionA.
49. Add Placement and Bundle separation constraint, values PO, session, none.
50. Add effectiveDatesDayTimeAttributeGroup for flight window to placement and bundle. Note added in element wrapper.
51. Add assetMetadata to AppReference so symmetric with AssetReference.
52. Change metadata type to match NameValueType in rules.
53. Add source to TermType, {ent,|place|ruleStore}.
54. Add operator to qualifier expression for "contains", functionally Java string.contains().
55. Add "private:*" pattern to qualifier namespaces.
56. Feedback counts are a name/value pair, with the value optionally a datamap reference consistent with the Phase processing rule data space.
57. Remove "rotation" from AssetReferenceType.
58. Add trick modes to asset reference.
59. Add order attribute to AssetReference which defines the order of the assets in a placement response.
60. Add back CIP 1.1 context to read request/response.
61. Put enumerations in schema as patterns.
62. Remove duplicate registration prohibition, keep all when multiple.
63. Remove Duration from placement as ability already exists in AssetReference metadata.
64. Extend StateFeedback to an AssetReference and AppReference level.
65. Make AssetMetadata more similar to "Term" (TermType) and include optional Namespace and Qualifier.
66. Added StateFeedback to:
    a. ProcessRuleSelectorType
    b. AssetReferenceType
    c. AppReferenceType
67. Define following at Bundle level as well as Placement:
    a. Placement @SeparationConstraintType
    b. Placement CategoryExclusion
68. AssetReference gets TrickMode element. For AppReference added values from Mux 3.0, at least testFlag, testPriority.
69. In Bundle, removed CipRevDate.
70. Removed Placement@category attribute.
71. Renamed cipVer to campVer.
72. Changed DecisionOwner to apply pattern to { uri:*, private:*}.
73. Placement @productMember references ProductMemberType.
74. Removed ValueType.
75. Removed CipNotificationGroup.
76. Changed invoke to event.
77. Removed PerCentType.
78. Removed ValueType.
79. Removed CipNotificationGroup.
80. The pattern for ExclusionScopeType contained a typo: "Session" has been changed to "session" (all lowercase).
81. PhaseEventType referenced an incorrect recState ("Winddown"). The correct recState is "Windup".
82. The documentation for TrickModeRestrictionRule has been moved into its parent, TrickModeRestrictionType. Also, the final line now reads "the actual PEID and EPSID of the parent Asset or Placement".
83. A new event named "decision" has been added to PhaseEventType. This event is valid only within a Placement, and represents each evaluation of the candidate within the decision process for an opportunity.

84. An optional "sequence" attribute has been added to ProcessRuleSelectorType. This attribute holds an ordinal value (one or greater) that defines rule evaluation order within a Phase or other multi-rule container.
85. The "priority" attribute has been duplicated from PlacementType into BundleType.
86. A RecordStatusAttributeGroup has been added to PhaseType.
87. Eligible PhaseEvents are now consistent for Phases within Bundle and Placement. This allows all events at the placement and bundle levels. NOTE that this means there are now two bundle-level triggers: transfer from a lower-level event, and "found" as a primary bundle trigger.
88. To add an interval trigger mechanism, two new elements have been added to StateFeedbackType: reportInterval and reportEndTime. These are used by the CIP publisher to control state feedback reporting for this CIP version.
89. A "group" option has been added to ExclusionScopeType.
90. A "group" option has been added to SeparationConstraintType.
91. A "creDat" attribute has been added to RecordStatusAttributeGroup.
92. The "cipCreDat" and "cipRevDat" attributes have been removed from BundleType.
93. In BundleType, the following have been renamed:
    cipOrdrOwnr is now campOrderOwner
    cipOrder is now campOrder
94. In BundleSelectorGroup, the following have been renamed to match the corresponding BundleType fields:
    cipGUID is now guid
    cipGname is now gname
    cipOrderId is now campOrder
    cipOrderIdOwner is now campOrderOwner