OPSA Working Group M. Palmero
Internet-Draft F. Brockners
Intended status: Informational Cisco Systems
Expires: 31 December 2023 S. Kumar
NC State University
C. Cardona
NTT
D. Lopez
Telefonica I+D
29 June 2023

# Asset Lifecycle Management and Operations~~, ~~: A Problem Statement
draft-palmero-opsawg-ps-almo-00

## Abstract

This document presents a problem statement for assets lifecycle management and operations. It describes ~~the ~~a framework, the motivation and requirements for asset-centric metrics including, but not limited to, asset adoption, usability, entitlements, supported ~~features and~~ capabilities, and enabled ~~features and~~ capabilities. ~~An~~The document also defines an information model ~~is proposed~~ whose primary objective is to measure and improve the network operators' experience along the lifecycle journey, from technical requirements and technology selection through renewal, including the end of life of an asset.
.

**Commenté [BMI1]:** This may be restrictive as the abstract indicates that also a framework is defined.

**Commenté [BMI2]:** As this can be seen as a capability.

## Status of This Memo



## Copyright Notice

Table of Contents



## 1. Introduction

The virtualization of hardware assets and the development of applications using microservice architectures for cloud-native infrastructures ~~created~~ triggered new utilization and licensing models. For example, a service can be deployed by composing multiple assets together; where an asset refers to hardware, software, application, system, or another service. ~~The~~ Also, services ~~could also~~can be instantiated ~~dynamically~~on-demand. As an example, cloud-native ~~infrastructures~~ services from one vendor may be hosted on the physical server from another vendor or a combination of multiple cloud-native functions from one or more vendors can be combined to execute ~~any~~ a composite service.

~~This~~ Such dynamicity introduces challenges for both lifecycle and adoption management of the assets. For example, an operator may need to identify the capability availability of different assets or measure the usage of each capability (or the combination thereof) from any specific asset to measure, e.g., its optimal potential. Moreover, an operator could pinpoint the reason of ~~this~~ a non-~~optimum~~ nominal use of ~~the~~ an asset, e.g., the software application could not be optimally deployed~~, or~~ because it is not simple to use~~,~~ or the usage is not well documented, etc. The operator may then feed this information back to the support ~~engineers~~ teams and the software developers, so they can focus their work effort only on features that users are adopting, or even determine when the lifecycle of the development could end.

**a mis en forme :** Surlignage

**a mis en forme :** Surlignage

This creates the need to collect and analyze asset-centric lifecycle management and operations data. ~~From now on, t~~This data is referred to as ~~will be referred as~~ Asset Lifecycle Management and Operations (ALMO). ~~for an asset; where~~Note that ALMO is not limited to virtualized or cloud environments, it covers all types of networking environments in which technology assets are deployed. No assumption is made about which technology is used to implement an assess.

ALMO data is the set of data required to measure asset-centric lifecycle metrics including, but not limited to, asset adoption and usability, entitlement, supported features and capabilities, and enabled features and capabilities.

The main challenge in collecting ALMO data, especially in a multivendor environment, ~~relies on~~ is the ability to produce and consume such data in a vendor-agnostic, consistent, and synchronized way. Another challenge is to maintain/cleanup the data (e.g., update it with external data such as End of Support (EoS)).

This document describes the motivation for ALMO, lists sample use cases, followed by an ALMO ~~proposed~~ information model ~~for ALMO~~. The ~~list of~~ use cases ~~describes~~motivate the need for new functional blocks and their interactions. The current version of this document focuses on assets, entitlement information, features, and feature's usage. A more specific document will follow this reference with the specification of the YANG data model for the specific ~~four~~ modules [RFC7950].

~~This document is organized as follows. Section 2 establishes the terminology and abbreviations. In Section 3, the goals and motivation of ALMO are discussed. In Section 4, use cases are introduced. Section 5 proposes the information model for ALMO.~~

## 2. Terminology

The document makes use of the following terms:

* Asset: refers to hardware, software, applications, or services. An asset can be physical or virtual. The granularity of what constitutes an asset is deployment and implementation specific.

* Consumer: refers to an entity that utilizes the ALMO data. A consumer can be an operator, an asset developer, or some other interested third party. Also, network controllers or orchestrators may be considered as consumers under some conditions.

* Developer: refers to the entity that creates or develops an asset or an asset component.

* Features: are options or functional capabilities ~~available~~ offered by ~~in~~ an asset.

**Commenté [BMI3]:** Just pick one, unless there are strong differences between this two. If so, you may clarify the difference.

* Entitlement: also known as license, is issued by an entity such as the developer or the Open Source community. It ~~and allows the~~ provides a consumer to legitimately make ~~operator~~ ~~to~~ use of an ~~the~~ asset. Entitlements determine how ~~the~~ an asset can be leveraged and what is required in cases the asset is changed.

* Event report: refers to record of details of an incident that an operator reports, e.g., a trouble ticket.

* Lifecycle Management and Operations (LMO) connects to: 1. Assets as a generalized entity subject to LCM. 2. Entitlements as the essential policy root for LCM. 3. Metrics as the evidence for LCM transitions. 4. Reports (no incidents) as the way of collecting input from users.

* Optimal Software Version (OSV): refers to the elected software version considered optimal in the user environment.

* Usage: refers to how an asset is being used (e.g., which features are used).

* Operator: refers to the owner or consumer of the asset. An ~~Operator~~operator belongs to an organization. Within the organization there are entities that~~: a)~~ use the assets in their operations~~, b)~~ and those who manage the assets. For example, supply chain and risk management teams will also be operators~~. For example,~~ by running “what-if” scenarios to anticipate decommissioning operations or assess the impact of component exhaustion, etc. This document uses the terms "user" and "operator" interchangeably.

* User Experience: how a user interacts with and experiences a particular asset. It includes a user's perceptions of ease of use, efficiency, and utility of an asset.

Abbreviations:

* ALMO: Asset Lifecycle Management and Operations~~.~~

* EoL: End of Life~~.~~

* EoS: End of Support~~.~~

* LCM: Lifecycle Management~~.~~

* PID: Product Identifier~~.~~

## 3. Motivation

The operator experience with a specific asset can be organized into four classes:

**a mis en forme :** Surlignage

**Commenté [BMI4]:** Or consumer ?

1. Asset characteristic class: covering ~~anything~~ a set of data that characterizes an ~~related to~~ assets, entitlement, features, etc.

2. Utilization class: measuring how the assets (including ~~and~~ the features) are used, duration of usage, uptime, etc.

3. Notification class: covering any security advisory, retirement, etc.

4. Incident class: ~~to~~ recording and reporting ~~any~~ problems that ~~the~~ an operator has faced with ~~the~~ an asset.

The ability to measure, produce, and consume ALMO data could benefit the operator in addressing issues such as:

* Entitlements may not have been obtained at the ~~optimum~~ nominal level of usage for a given asset or feature, where a user might have bought entitlements that are not activated. Defining a nominal level is local to each deployment.

* Features of an asset might not be used as needed in all deployments within the organization and for various reasons.~~.~~

* Resolution of incidents involving an asset and the developer of a technology used within the asset.

* (Virtualized) asset lifecycle events, such as scaling or migration.

* Facilitating DevOps deployment, including automated testing, validation, pre-production, production, and~~/~~ certification procedures.

* A link to supply chain management and/or connect it with the provenance information.

ALMO could ~~allow~~ help developer organizations to optimize their features. For example, they could consider deprecating features that are used infrequently ~~or~~, focus on introducing more features for the assets that are widely deployed in various infrastructures, or adjust the design to better ease usability or ease integration, etc.

ALMO ~~also covers the need of communication between users and the developer. ALMO can~~ provides a structured approach ~~the capability~~ for users to provide feedback about ~~any~~ an asset (e.g., potential deficiency of a feature, feature enhancement request, inadequacy of a usage-based license model ). An administrator in the user organization may define and thus include specific metrics that identify a potential problem of ~~that~~ a specific asset feature ~~or a capability of the asset~~. ~~An engineer in the~~A developer ~~organization~~ can determine the impact of the potential deficiency from the number of users providing feedback or the amount/periodicity of feedback (e.g., enrollment, updates). Note that this channel is different from a "call to a Technical Assistance Center" in which the user may request help in resolving specific operational issues with the asset.

## 4. Sample Use Cases

This section presents some use cases where ALMO is ~~required~~useful.

### 4.1. Entitlement Inventory and Activation

An Ops ~~engineer~~ team would like to understand which entitlements are activated and which are ~~actually~~ being used, ~~and/~~or consumed. It is also important for asset operators to understand which features might need an entitlement and how to activate them.

It is relatively straightforward to have an inventory of existing entitlements when there is only one asset developer (providing the asset) and one asset family.

But complexity grows when there are many different developers, systems and processes involved. New service offerings have introduced new attributes and datasets and require alignment with new business models (pay-per-product, subscription model, pay-as-you-go model, etc.). They might support different entitlement types and models: asset activation keys, trust-based model, systems that act as proxy from the back end owned by the asset developer to support the control of entitlements, etc.

Sometimes it is a challenge to report which entitlements have been bought by the asset user, or who in the user organization owns that entitlement because that information might rely on different asset developers; even within the same asset developer, entitlements may correspond to different types or groups of assets. Asset users often need to interact with different entitlement systems and processes.

Information on how assets are entitled could be delivered from a combination of attributes such as: sales order, purchase order, asset activation key, serial number, etc.

If there is no consistency on how to deal with those data points, complexity increases for the consumer, potentially requiring expensive manual procedures. Automating those ~~manual~~ steps or exceptions becomes time-consuming, eventually leading to higher costs for the asset consumer.

**Commenté [BMI5]:** Not sure what is the intent here.

Having a common ALO reference ~~for ALMO~~ eases the integration between different data sources, processes, and consolidation of the information under a common reference.

The ALMO information model and ~~the future~~companion data models ~~for ALMO~~ will include data to support metrics that might be required by consumption-based charging and entitlement of asset usage.

### 4.2. Risk Mitigation Check

Asset operators ~~would like to be aware~~require tracking ~~of~~ issues known by the asset developer that are causing assets to crash or perform badly so that they can act to remediate the issue quickly, or even prevent the crash if alerts are triggered on time. There are analytics tools that can process memory core dumps and crash-related files, providing the ability to the asset developers to determine the root cause.

Accordingly, asset users can remediate the problem, automate the remedy to enable incident deflection, allowing the support staff to focus on other problems.

Risk Mitigation Check (RMC) could also include the possibility to be aware of current and historical restarts allowing network and software engineers to enhance the service quality to asset users. Also, RMC could run what-if scenarios to identify which assets would be impacted when a certain component is faulty.

### 4.3. Bug Fixes and Errata

Both hardware and software critical issues or Errata need development to automate asset user matching:

* Hardware Errata match on product identifiers (PIDs) + serial numbers along with additional hardware attributes.

* Software Errata match on software type and software version along with some additional device attributes.

Engineering might develop the logic to check whether any critical issue applies to a single serial number or a specific software release.

The information to be correlated includes customer identification, activated features, and asset information that the asset user might own. All this information needs to be correlated with hardware and ~~softwae~~software errata, and EOL information to show which part of the asset inventory might be affected.

### 4.4. Security Advisory

The Security Advisory use case automates the matching of asset user data to security bulletins published by asset developers. Security Advisory logic implemented by developers could apply to a specific software release.

This is a specific use case of ~~t~~Section 4.3.

### 4.5. Optimal Software Version

The objective of the Optimal Software Version (OSV) use case is that consumers can mark software images as OSV for their assets; based on this, it is easier for them to control and align their hardware and software assets to the set of OSVs.

Based on the logic of OSV, use cases like software compliance, risk trend analysis, acknowledge bugs, security advisories, errata, what-if analysis, etc., could be realized.

### 4.5.1. Software Conformance

All the assets should normally be running at their latest recommended software version; notably in case of a required security update for a specific feature. The recommended version may be a decision of the vendor or the operator.

The Software Conformance use case provides a view to the asset users and informs the users whether the assets that belong to a specific group conforms to the OSV. It can provide the users with a report, including a representation of software compliance for the entire network and software applications. This report could include the current software version running on the asset and the recommended software version. The report could enable users to quickly highlight which group of assets might need the most attention to inspire appropriate actions.

The Software Conformance use case uses data that might not be provided by the asset itself. Data needs to be provided and maintained also by the asset developers, through e.g., asset catalog information. Similar logic applies to a feature catalog, where the asset developer maintains the data and updates it adequately based on existing bugs, security advisories, etc.

The Software Conformance process needs to correlate the Software catalog information with the software version running on the asset.

### 4.5.2. What-if Analysis

The What-if Analysis use case allows asset users to plan for new hardware or software, giving them the possibility to change the configuration parameters or model how new hardware or software might change the software suggestions generated by OSV.

OSV and the associated use cases involve dependencies on attributes that might need to be collected from assets directly, including related inventory information (serial numbers, asset identifiers, software versions, etc.), but also dynamic information could be required, like:

* Information on features that might be enabled on the particular asset.

* Catalogs, that might include information related to release notes. For example, consider a feature catalog. This catalog could include software versions that support a specific feature; the software releases that a feature is supported in; or the latest version that a feature is supported in, in case the feature is EOL.

* Data sources to correlate information coming from reports on critical issues or errata, security advisory, End of Life, etc.

Those catalogs and data sources with errata information, EOL, etc. need to be maintained and updated by asset developers, making sure, that the software running on the assets is safe to run and up to date.

## 4.6. Asset Retirement - End of Life

Hardware EOL reports need to map Hardware EOL PIDs, focusing on base PIDs so that bundles, spares, non-base PIDs, etc., do not provide false EOL reporting to asset users.

Software EOL reports are used to automate the matching of user software type and software version to software EOL bulletins.

# 5. ALMO Information Model

The broad metric classes defined in Section 3 that quantify user experience can be modeled as shown in Figure 1. There is a reference to the assets that the user possesses. Each asset may be entitled to one or more entitlements; ~~a~~ an entitlement may contain one or more sub-entitlements. The level of usage for each feature and entitlement associated with the asset is measured.

For example, a user needs to measure the utilization of a specific entitlement for a specific type of asset. The information about the entitlement may reside in ~~a~~ an entitlement server. The state (activated or not) of the entitlement may reside with the asset itself or a proxy. They can be aggregated/correlated as per the information model shown in Figure 1 to give information to the user regarding the utilization of the entitlements. The user experience is thus enhanced by having accurate knowledge about the utility of the given entitlement.

```
 may_be_part_of                                    may_be_part_of
   +------+                entitled_by                +-------+
   |      | +----------------------------------------+ |       |
   |      | |                                        | |       |
   |      V V                                        | V       |
+------------+  entitled_by           tracked_by   +------------+
|Entitlements|<-------------+          +-------------|   Usage    |
+------------+-----------+ | may_be_  | +----------->+------------+
|Entitlement | entitles  | | part_of  | | tracks     |  Features  |
| attributes |           | | +------+ | |            | and usage  |
+------------+           | | |      | | |            | attributes |
                         v | |      v v |            +------------+
                        +-----------------+
                        |     Asset       |
                        +-----------------+    future_association
                        |     Asset       |<-----------------+
                        |   attributes    |----------------+ |
                        +-----------------+                | |
                                                           v |
                                                   +------------+
                                                   |  Future    |
                                                   | Expansion  |
                                                   +------------+
```

Figure 1: Information Model

The model allows for future expansion by new metrics that will quantify user experience. Notice that future association relationship and future expansion might be linked to asset or to one of the other datasets: Feature, Feature Usage or Entitlements.

## 6. Security Considerations

~~The security considerations mentioned in section 17 of [RFC7950] apply.~~

ALMO brings several security and privacy implications because of the various components and attributes of the information model. For example, each functional component can be tampered with to give manipulated data. ALMO when used alone or with other relevant data, can identify an individual, revealing ~~Personal Identifiable Information (PII)~~personal data. Misconfigurations can lead to data being accessed by unauthorized entities.

~~Methods exist to secure the communication of management information.~~ The transport entity of the functional model ~~MUST~~ must implement methods for secure transport. This document also contains an Information model ~~and Data Model~~ in which none of the objects defined are writable. If the objects are deemed sensitive in a particular environment, access to them ~~MUST~~ must be restricted using appropriately configured security and access control rights. The information model contains several optional elements which can be enabled or disabled for the sake of privacy and security. ~~Proper~~ Appropriate authentication and audit ~~trail~~ procedures ~~MUST~~ must be in place ~~included~~ for all the ~~users/processes~~consumers of ~~that access the~~ ALMO.

Commenté [BMI6]: There is no yang in the doc

Commenté [BMI7]: Is US-centric.

Commenté [BMI8]: No sure what entity you are referring to/

Commenté [BMI9]: Shouldn't ALMO data be fed with data shared by the vendors (e.g., planned EoL)?

Commenté [BMI10]: So, these are configurable. This may conflict with the previous sentence.

## Change log

RFC Editor Note: This section is to be removed during the final publication of the document.

version 00

* Initial version.

* Reference to DMLMO draft, looking to simplify the objective and understanding of it, with focus on the framework and the main functional models, reducing scope to assets, features, usage and entitlements; with special highlight to the interrelation between them.

## Acknowledgments

The authors wish to thank Martin Beverley, Ignacio Dominguez Martinez, and Gonzalo Salgueiro (by alphabetical order) for their helpful comments and suggestions.

## Contributors

This document was created by meaningful contributions (by alphabetical order) from Shwetha Bhandari, Yenu Gobena, Nagendra Kumar Nainar, Jan Lindblad, Josh Suhr, Dhiren Tailor, Yannis Viniotis, and ~~Ã‰ric~~Éric Vyncke.

## Informative References

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/rfc/rfc7950>.

## Authors' Addresses

Marisol Palmero
Cisco Systems
Email: mpalmero@cisco.com

Frank Brockners
Cisco Systems
Email: fbrockne@cisco.com

Sudhendu Kumar
NC State University
Email: skumar23@ncsu.edu

Camilo Cardona
NTT
Email: camilo@ntt.net

Diego Lopez
Telefonica I+D
Email: diego.r.lopez@telefonica.com