Core Information Model (CoreModel)

TR-512.3 Foundation (Identifiers, naming and state)

Version 1.3.1
January 2018

ONF Document Type: Technical Recommendation
ONF Document Name: Core Information Model version 1.3.1

Disclaimer

Open Networking Foundation
2275 E. Bayshore Road, Suite 103, Palo Alto, CA 94303
www.opennetworking.org

Important note

This Technical Recommendations has been approved by the Project TST, but has not been approved by the ONF board. This Technical Recommendation is an update to a previously released TR specification, but it has been approved under the ONF publishing guidelines for 'Informational' publications that allow Project technical steering teams (TSTs) to authorize publication of Informational documents. The designation of '-info' at the end of the document ID also reflects that the project team (not the ONF board) approved this TR.

Table of Contents

List of Figures

Document History

Version	Date	Description of Change
1.0	March 30, 2015	Initial version of the base document of the "Core Information Model" fragment of the ONF Common Information Model (ONF-CIM).
1.1	November 24, 2015	Version 1.1
1.2	September 20, 2016	Version 1.2 [Note Version 1.1 was a single document whereas 1.2 is broken into a number of separate parts]
1.3	September 2017	Version 1.3 [Published via wiki only]
1.3.1	January 2018	Addition of text related to approval status.

1 Introduction

This document is an addendum to the TR-512 ONF Core Information Model and forms part of the description of the ONF-CIM. For general overview material and references to the other parts refer to TR-512.1.

1.1 References

For a full list of references see TR-512.1.

1.2 Definitions

For a full list of definition see TR-512.1.

1.3 Conventions

See TR-512.1 for an explanation of:

- UML conventions
- Lifecycle Stereotypes
- Diagram symbol set

1.4 Viewing UML diagrams

Some of the UML diagrams are very dense. To view them either zoom (sometimes to 400%) or open the associated image file (and zoom appropriately) or open the corresponding UML diagram via Papyrus (for each figure with a UML diagram the UML model diagram name is provided under the figure or within the figure).

1.5 Understanding the figures

Figures showing fragments of the model using standard UML symbols and also figures illustrating application of the model are provided throughout this document. Many of the application-oriented figures also provide UML class diagrams for the corresponding model fragments (see TR-512.1 for diagram symbol sets). All UML diagrams depict a subset of the relationships between the classes, such as inheritance (i.e. specialization), association relationships (such as aggregation and composition), and conditional features or capabilities. Some UML diagrams also show further details of the individual classes, such as their attributes and the data types used by the attributes.

2 Introduction to the Foundation Model

The focus of this document is the parts of Core Foundation Model of the ONF-CIM that deal with naming, identifiers and state.

The CoreFoundationModel covers all aspects of naming and addressing for all classes in the ONF-CIM. The focus of this document is:

- Definition of terminology related to naming and identity
- Description of the model for naming and identifiers that is inherited by other models
- Description of the state model

A data dictionary that sets out the details of all classes, data types and attributes is also provided (TR-512.DD).

3 CoreFoundationModel

This Model includes all aspects of the core that are relevant to all other parts of the ONF CIM such as identifiers, naming and states.

3.1 Naming and identifiers

3.1.1 Key considerations

To communicate about a thing it is necessary to have some way of referring to that thing, i.e., to have some reference. Terms such as name and identifier are often used when describing the reference. Unfortunately these terms in general usage have ambiguity in their definition that leads to erroneous system behavior. To ensure that the controller system behavior is not erroneous, the model will adopt the following principal definitions:

- Entity[1]: Has identity, defined boundary, properties, functionality and lifecycle in a global context.
    - Examples: An Equipment (see TR-512.6), an LTP (see TR-512.2)
- Entity Feature[2]: An inseparable, externally distinguishable part of an entity.
    - Examples: A pin in connector (see TR-512.6), the ports of a FC (see TR-512.2), a face of a cube, the handle of a cup.
    - Note that this is important from a modeling perspective as the representation appears similar to that of an Entity
    - Represented using a UML class
- Role: A specific structure of responsibilities, knowledge, skills, and attitudes in the context of some activity or greater structure. The role has an identity and identifier.
- Identifier: A property of an entity/role with a value that is unique within an identifier space, where the identifier space is itself globally unique, and immutable. An identifier carries no semantics with respect to the purpose or state of the entity.
- Universally Unique Identifier[3] (UUID): An identifier that is universally unique.
- Local ID: An identifier that is unique in the context of some scope that is less than the global scope.
- Name: A property of an entity with a value that is unique in some namespace but may change during the life of the entity. A name carries no semantics with respect to the purpose of the entity.

[1] An Entity is represented using a UML class
[2] A Feature of an Entity is represented using a UML class
[3] The term GUID was used in the previous version of the model. The change is in recognition of the more generally applicability of UUID

- Label: A property of an entity with a value that is not expected to be unique and is allowed to change. A label carries no semantics with respect to the purpose of the entity and has no effect on the entity behavior or state.
    - A label can be used to carry a freeform text string for any operator purpose. The contents of a label in one view may happen to be the value of a name or identifier in another view. From the perspective of the view with the label there is no expectation other than the value is a string.
- Place: Where something is located
- Address: A structure of named values[4] in some address space that defines a location (a volume in that address space) where the structure is a nested hierarchy.
    - A named value may be a name or identifier, the name of the value may be a name or identifier
- Route: the way (via specified intermediate locations and paths) to get to one location from another.
- Property: A quality associated with a thing, structure or location.
- Semantics: Meaning.
- Reference: Data in a communication between two applications that allows a shared understanding of the individual things.
    - This could be an identifier (including a UUID), a name, an address, or a route, depending upon the needs

Note:

- An entity may be known to be at a place in some functional or physical structure.
- A role may be known to be at a place in some process or behavioral structure.

The figure below illustrates the naming/identifier-related attributes defined in the ONF-CIM. They are Universally Unique ID (UUID), Local ID, Name and Label.

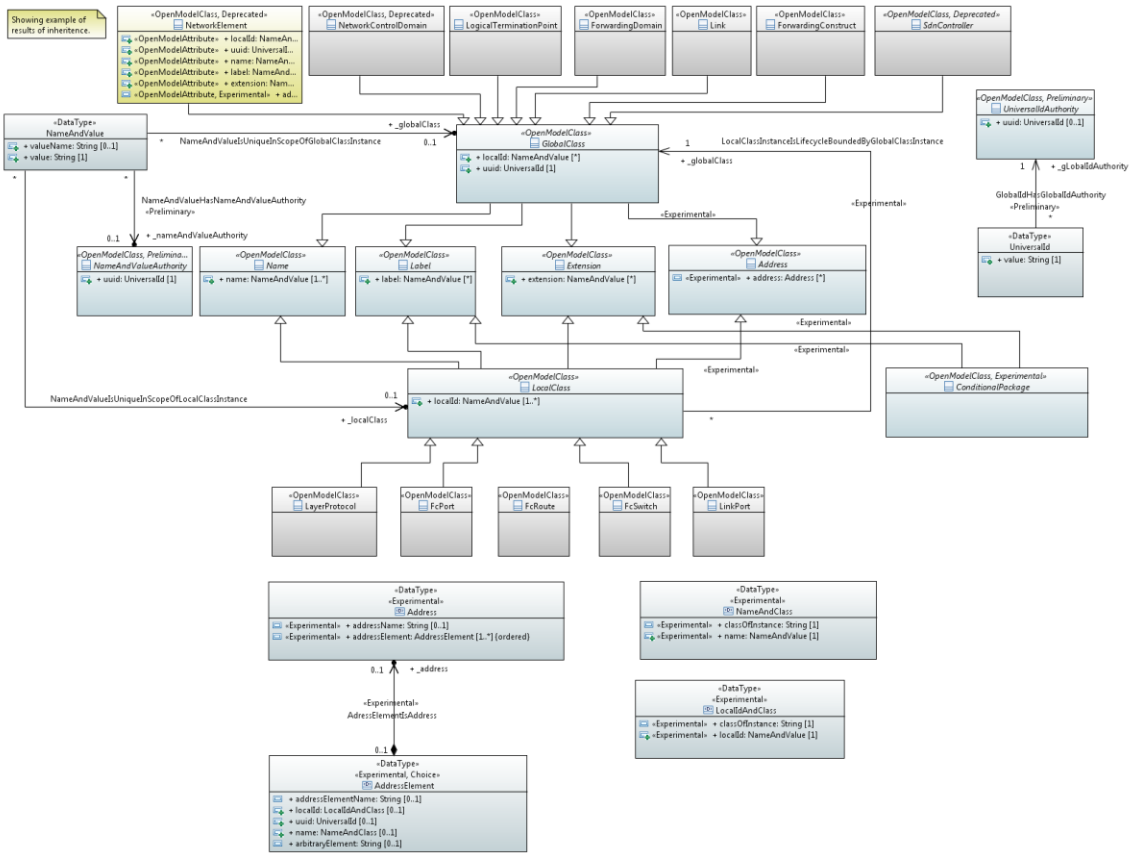
The model includes two abstract classes that provide names and identifiers, the GlobalClass and the LocalClass.[5] A GlobalClass represents a type of thing that has instances which can exist in their own right (independently of any others). A LocalClass represents a type of thing that is inseparable from a GlobalClass, but that is a distinct feature of that GlobalClass such that the instances of LocalClass are able to have associations with other instances. The mandatory LocalId of the LocalClass instance is unique in the context of the GlobalClass instance, from which it is inseparable.

The model also includes Extension which is not related to naming/identification. Extension provides an opportunity to define properties not declared in the class that extend the class enabling a realization with simple ad-hoc extension of standard classes to be conformant.

Note that a UUID is applicable only to global type object classes (i.e., subclass of GlobalClass) that their instances can exist on their own right, e.g., LTP, FD, Link, FC, and NetworkElement.

[4] A named value is simply a tuple with two terms, one being a value and the other being the name of that value. For example in a street address a value may be “London” and the name of that value would be “City”.
[5] The model also provides ConditionalPackage to supply names and identifiers to _Pac classes but this is currently experimental.

The other naming/identifier-related attributes are applicable to both global type object classes and local type object classes (i.e., subclass of LocalClass).[6]



CoreModel diagram: Foundation-CommonPackagesNoNotes

Figure 3-1 Class Diagram for Naming and Identifier of Objects

3.1.2 Classes and attributes

3.1.2.1 Address

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::Address

Provides an opportunity to state the location of the entity via one or more hierarchies of narrowing contexts.

This class is abstract.

[6] The intention is that only classes from the Core Model are shown in the figure. The classes shown are essentially illustrative. There is another figure in the model that captures Core Model inheritance in detail. All classes from all fragments should inherit from GlobalClass, LocalClass or ConditionalPackage. There is no issue with model dependency as the inheritance association is maintained with the class that is inheriting properties. Although not mandatory, it would seem advisable to maintain a figure per fragment that shows all classes from that fragment and their inheritance.

Table 1: Attributes for Address

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
address	Experimental	One or more descriptions of the location.

3.1.2.2 ConditionalPackage

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::ConditionalPackage

The base class for conditional packages.

This class is abstract.

Inherits properties from:

- Label
- Extension

This class is Experimental.

3.1.2.3 Extension

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::Extension

Extension provides an opportunity to define properties not declared in the class that extend the class enabling a realization with simple ad-hoc extension of standard classes to be conformant.

This class is abstract.

Table 2: Attributes for Extension

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
extension		List of simple name-value extensions.

3.1.2.4 GlobalClass

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::GlobalClass

Represents a type of thing (an Entity) that has instances which can exist in their own right (independently of any others).

Entity: Has identity, defined boundary, properties, functionality and lifecycle in a global context.

(This should be considered in the context of a Class: (usage) The representation of a thing that may be an entity or an inseparable Entity Feature).

This class is abstract.

Inherits properties from:

- Address
- Name
- Label
- State_Pac
- Extension

Table 3: Attributes for GlobalClass

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
localId		An identifier that is unique in the context of some scope that is less than the global scope. (This should be considered in the context of Identifier: A property of an entity/role with a value that is unique within an identifier space, where the identifier space is itself unique, and immutable. The identifier therefore represents the identity of the entity/role. An identifier carries no semantics with respect to the purpose of the entity.)
uuid		UUID: An identifier that is universally unique (This should be considered in the context of Identifier: A property of an entity/role with a value that is unique within an identifier space, where the identifier space is itself globally unique, and immutable. An identifier carries no semantics with respect to the purpose or state of the entity) The uuid should be treated as opaque by the user.

3.1.2.5 Label

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::Label

A property of an entity with a value that is not expected to be unique and is allowed to change. A label carries no semantics with respect to the purpose of the entity and has no effect on the entity behavior or state.

This class is abstract.

Table 4: Attributes for Label

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
label		List of labels.

3.1.2.6 LocalClass

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::LocalClass

A LocalClass represents a Feature of an Entity. It is inseparable from a GlobalClass but is a distinct feature of that GlobalClass such that the instances of LocalClass are able to have associations to other instances.
Feature of an Entity: An inseparable, externally distinguishable part of an entity.
The mandatory LocalId of the LocalClass instance is unique in the context of the GlobalClass from which it is inseparable.
(This should be considered in the context of a Class: (usage) The representation of a thing that may be an entity or an inseparable feature of an entity.)

This class is abstract.

Inherits properties from:

- Address
- Name
- Label
- State_Pac
- Extension

Table 5: Attributes for LocalClass

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
localId		An identifier that is unique in the context of some scope that is less than the global scope. (This should be considered in the context of Identifier: A property of an entity/role with a value that is unique within an identifier space, where the identifier space is itself unique, and immutable. The identifier therefore represents the identity of the entity/role. An identifier carries no semantics with respect to the purpose of the entity.)

3.1.2.7 Name

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::Name

Name: A property of an entity with a value that is unique in some namespace but may change during the life of the entity. A name carries no semantics with respect to the purpose of the entity.

This class is abstract.

Table 6: Attributes for Name

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
name		List of names.

3.1.2.8 NameAndValueAuthority

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::NameAndValueAuthority

Represents the authority that controls the legal values for the names and values of a name/value attribute.

This class is abstract.

This class is Preliminary.

Table 7: Attributes for NameAndValueAuthority

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
uuid		The UUID for the NameAndValueAuthority.

3.1.2.9 UniversalIdAuthority

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::ObjectClasses::UniversalIdAuthority

Represents the authority that controls the allocation of UUIDs.

This class is abstract.

This class is Preliminary.

Table 8: Attributes for UniversalIdAuthority

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
uuid		The UUID for the UUID authority.

3.1.3 DataTypes

3.1.3.1 Address

Qualified Name:
CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::TypeDefinitions::Address

A description of location via a hierarchy of narrowing contexts.

Table 9: Attributes for Address

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
addressName	Experimental	The name of the address (to allow the specific hierarchy to be distinguished from others for the same entity).
addressElement	Experimental	The elements of the address that form the recursive scope narrowing.

3.1.3.2 AddressElement

Qualified Name:
CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::TypeDefinitions::AddressElement

One element of a hierarchy of elements.
Note that the element must have one and only one value chosen from a list of potential value types.

Table 10: Attributes for AddressElement

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
addressElementName		The name of the address element (e.g. "shelf" as an element of a shelf/slot/port addressing scheme). The remainder of the structure has the reference for the shelf.
localId		If the element is a localId (where the element above in the hierarchy must be the context in which the specific localId is relevant).
uuid		If the element is a uuid (where this element could be the top of a hierarchy but may also be at some level in the hierarchy where address navigation is considered necessary to assist in location of the UUID).
name		If the element is a name.

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_address		See referenced class
arbitraryElement		Where the element is from some external model that is not formally represented in this model.

3.1.3.3 LocalIdAndClass

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::TypeDefinitions::LocalIdAndClass

The localId and the class of entity that it identifies.

Table 11: Attributes for LocalIdAndClass

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
classOfInstance	Experimental	The class to which the name refers.
localId	Experimental	The localId of the entity.

3.1.3.4 NameAndClass

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::TypeDefinitions::NameAndClass

The name and the class of entity that it names.

Table 12: Attributes for NameAndClass

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
classOfInstance	Experimental	The class to which the name refers.
name	Experimental	If the element is a name.

3.1.3.5 NameAndValue

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::TypeDefinitions::NameAndValue

A scoped name-value pair.

Table 13: Attributes for NameAndValue

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
valueName		The name of the value. The value need not have a name.
value		The value.
_nameAndValueAuthority		The authority that defines the named value.
_globalClass		The scope of the name uniqueness.
_localClass		The scope of the name uniqueness.

3.1.3.6 UniversalId

Qualified Name: CoreModel::CoreFoundationModel::SuperClassesAndCommonPackages::TypeDefinitions::UniversalId

The universal ID value where the mechanism for generation is defined by some authority not directly referenced in the structure.
An example structure is [IETF RFC4122].

Table 14: Attributes for UniversalId

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
value		The specific value of the universal id.

3.1.4 Use of names, identifiers and addresses

The following figure provides various examples of naming and identifier. The figure is currently under development. The figure includes diagram element from TR-512.5 and TR-512.6.

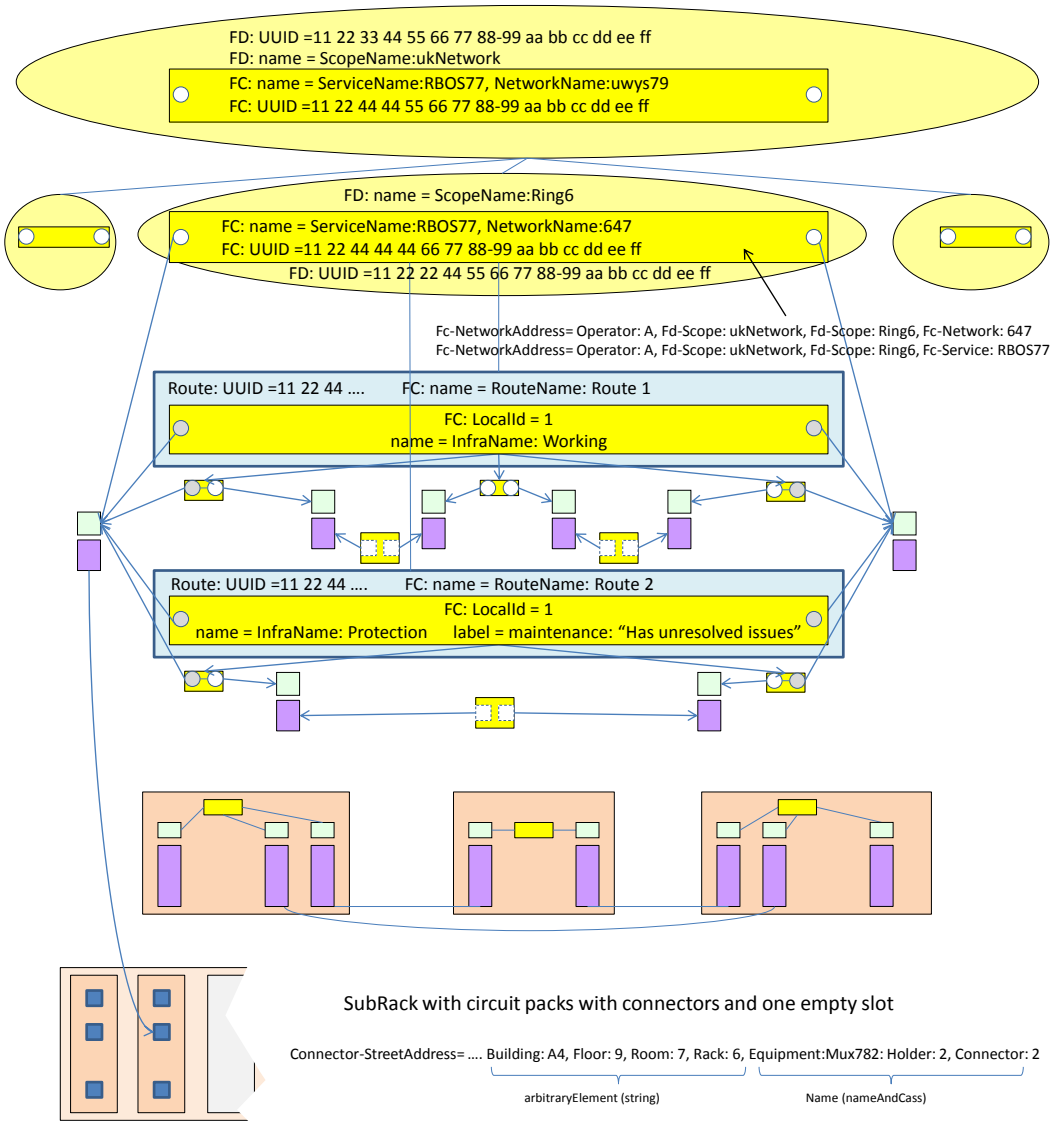
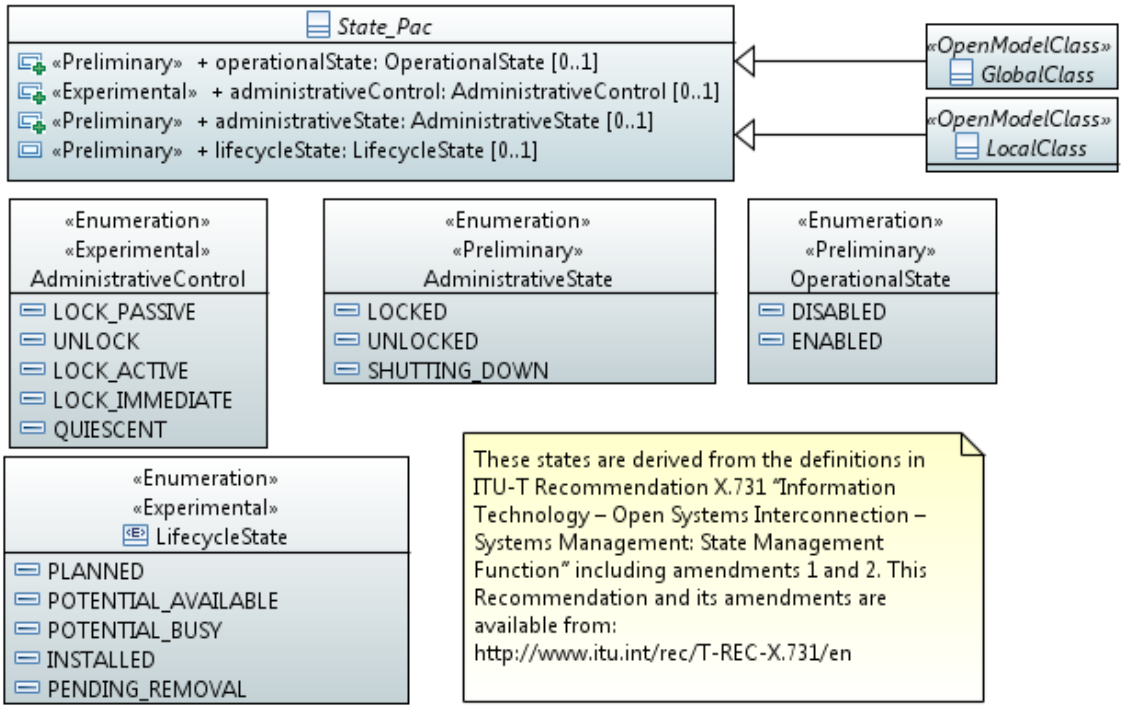


Figure 3-2 Sketch of names, identifiers and addresses for various entities

3.2 States

The Core Foundation Model also defines a State_Pac artifact, which provides state attributes. The work on states is preliminary at this stage (it is derived from [ITU-T X.731]). The State_Pac is inherited by GlobalClass and LocalClass object classes. The use of these states provides a consistent way represent the overall operability, usability and current usage of the resource.

It should be noted that the states are «Preliminary»/«Experimental».



CoreModel diagram: State-FullModel

Figure 3-3 States for all Objects

3.2.1 Classes and attributes

3.2.1.1 AdministrativeState

Qualified Name: CoreModel::CoreFoundationModel::StateModel::StateMachines::AdministrativeState::AdministrativeState

This class is Experimental.

3.2.1.2 LifecycleState

Qualified Name: CoreModel::CoreFoundationModel::StateModel::StateMachines::LifecycleState::LifecycleState

This class is Experimental.

3.2.1.3 OperationalState

Qualified Name: CoreModel::CoreFoundationModel::StateModel::StateMachines::OperationalState::OperationalState

This class is Experimental.

3.2.1.4 State_Pac

Qualified Name: CoreModel::CoreFoundationModel::StateModel::ObjectClasses::State_Pac

Provides general state attributes.

This class is abstract.

This class is Preliminary.

Table 15: Attributes for State_Pac

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
operationalState	Preliminary	The operational state is used to indicate whether or not the resource is installed and working.
administrativeControl	Experimental	The administrativeControl state provides control of the availability of specific resources without modification to the provisioning of those resources. The value is the current control target. The actual administrativeState may or may not be at target.
administrativeState	Preliminary	Shows whether or not the client has permission to use or has a prohibition against using the resource. The administrative state expresses usage permissions for specific resources without modification to the provisioning of those resources.
lifecycleState	Preliminary	Used to track the planned deployment, allocation to clients and withdrawal of resources.

3.2.2 Enumerations

3.2.2.1 AdministrativeControl

Qualified Name: CoreModel::CoreFoundationModel::StateModel::TypeDefinitions::AdministrativeControl

Reflects the current control action when the entity is not in the desired state.

Applied stereotypes:

- Experimental

Contains Enumeration Literals:

- UNLOCK:
  - The intention is for the entity to become unlocked.
    The entity may already be UNLOCKED.
  - Applied stereotypes:

        - Experimental
- LOCK_PASSIVE:
    - The intention is for the entity to become locked but no effort is expected to move to the Locked state (the state will be achieved once all users stop using the resource).
      The entity may be LOCKED.
    - Applied stereotypes:
        - Experimental
- LOCK_ACTIVE:
    - The intention is for the entity to become locked and it is expected that effort will be made to move to the Locked state (users will be actively removed).
      The entity may already be LOCKED.
    - Applied stereotypes:
        - Experimental
- LOCK_IMMEDIATE:
    - The intention is for the entity to become locked and it is expected to move to the Locked state immediately (users will be force removed).
      The entity may already be LOCKED.
    - Applied stereotypes:
        - Experimental
- QUIESCENT:
    - The administrative state is at a stable value (LOCKED/UNLOCKED) and no action is being taken.
    - Applied stereotypes:
        - Experimental

3.2.2.2 AdministrativeState

Qualified Name:
CoreModel::CoreFoundationModel::StateModel::TypeDefinitions::AdministrativeState

The administrative state is used to show whether use of a resource is allowed or prohibited. The administrative state can be observed and directly controlled by a certain operational roles. Typically, only a user (in the provider context) with administrative privileges is allowed to write the administrative state, any other users are restricted to read only.

Applied stereotypes:

- Preliminary

Contains Enumeration Literals:

- LOCKED:
    - Users are administratively prohibited from making use of the resource.
    - Applied stereotypes:
        - Preliminary
- UNLOCKED:
    - Users are allowed to use the resource.
    - Applied stereotypes:

        - Preliminary
- SHUTTING_DOWN:
    - The entity is administratively restricted to existing instances of use only. There may be specific actions to remove existing uses. No new instances of use can be enabled.
      The resource automatically transitions to "locked" when the last user quits.
      The administrative state is not visible in the client context.
      The lifecycle state "pending removal" should be used to indicate to the client that the provider intends to remove the resource.
    - Applied stereotypes:
        - Experimental

3.2.2.3 LifecycleState

Qualified Name:
CoreModel::CoreFoundationModel::StateModel::TypeDefinitions::LifecycleState

This state is used to track the planned deployment, allocation to clients and withdrawal of resources.

Applied stereotypes:
- Experimental

Contains Enumeration Literals:
- PLANNED:
    - The resource is planned but is not present in the network.
      Should include a "time" when the resources are expected to be installed.
    - Applied stereotypes:
        - Experimental
- POTENTIAL_AVAILABLE:
    - The supporting resources are present in the network but are shared with other clients; or require further configuration before they can be used; or both.
      (1) When a potential resource is configured and allocated to a client it is moved to the INSTALLED state for that client.
      (2) If the potential resource has been consumed (e.g. allocated to another client) it is moved to the POTENTIAL BUSY state for all other clients.
    - Applied stereotypes:
        - Experimental
- POTENTIAL_BUSY:
    - The supporting resources are present in the network but have been allocated to other clients.
    - Applied stereotypes:
        - Experimental
- INSTALLED:
    - The resource is present in the network and is capable of providing the service.
    - Applied stereotypes:
        - Experimental

- PENDING_REMOVAL:
    - The resource has been marked for removal. Should include a "time" when the resources are expected to be removed.
    - Applied stereotypes:
        - Experimental

3.2.2.4 OperationalState

Qualified Name:
CoreModel::CoreFoundationModel::StateModel::TypeDefinitions::OperationalState

The operational state is used to indicate whether or not the resource is installed and working.

Applied stereotypes:

- Preliminary

Contains Enumeration Literals:

- DISABLED:
    - The resource is unable to meet the SLA of the user of the resource.
      If no (explicit) SLA is defined the resource is disabled if it is totally inoperable and unable to provide service to the user.
    - Applied stereotypes:
        - Preliminary
- ENABLED:
    - The resource is partially or fully operable and available for use.
    - Applied stereotypes:
        - Preliminary

3.2.3 Relationship between states in Provider context

If the lifecycleState is PLANNED then the operationalState must be DISABLED and the administrativeState should be LOCKED.

In all other circumstances the states are independent.

3.2.4 Relationship between states in the client and provider context

The table below lists the states in the provider context that influence the states in the client context. The client does not have direct visibility of the provider states but does perceive effect through the operationalState.

Table 16: Influence of Provider state on Client state

Provider State	**Value**	**Client State**	**Value**
operationalState	DISABLED	operationalState	DISABLED
administrativeState	SHUTTING_DOWN	operationalState	DISABLED

administrativeState	LOCKED	operationalState	DISABLED
lifecycleState	PLANNED	operationalState	DISABLED
lifecycleState	POTENTIAL_BUSY	operationalState	DISABLED

No other states in the client context have a dependency on the state in the provider context.

None of the states in the client context influence the states in the provider context.

The administrativeState in the provider context is not visible in the client context. The client context may maintain an independent administrativeState.

The provider controls the lifecycleState that is visible to the client context.

3.2.5 State diagrams

These state diagrams are experimental sketches that will be refined in following releases.

3.2.5.1 Administrative State

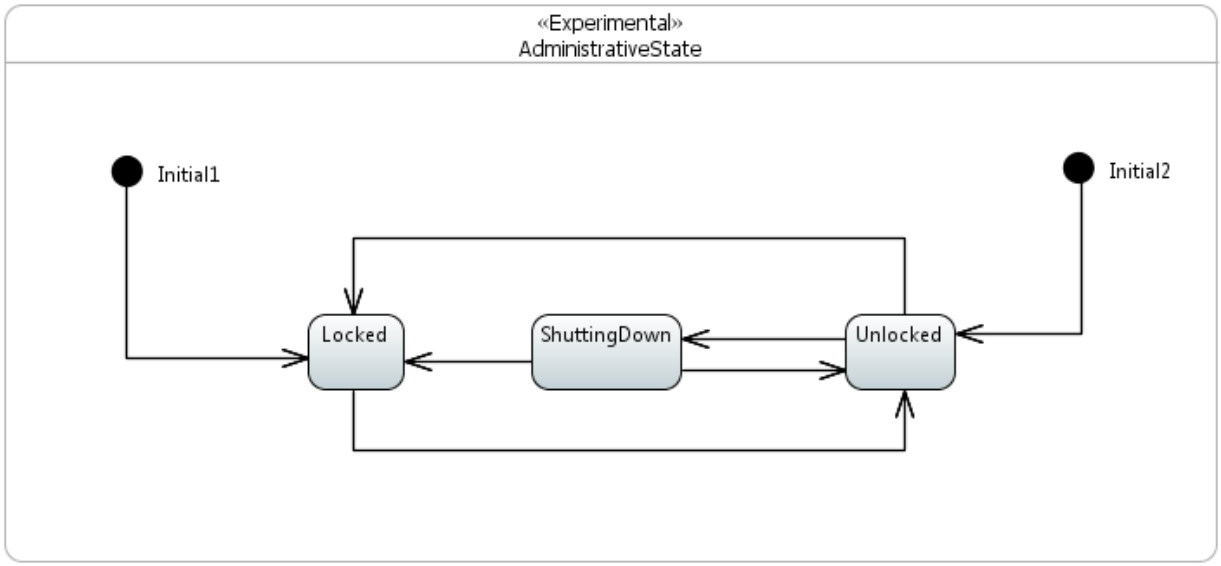


Figure 3-4 Administrative State

3.2.5.2 Operational State

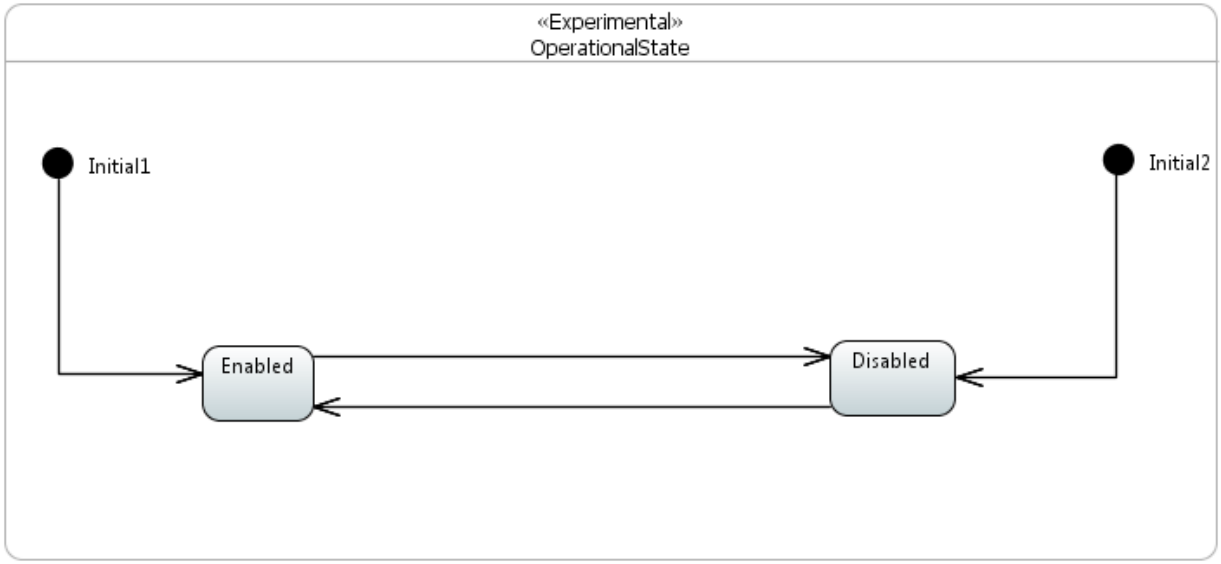


Figure 3-5 Operational State

3.2.5.3 Lifecycle State

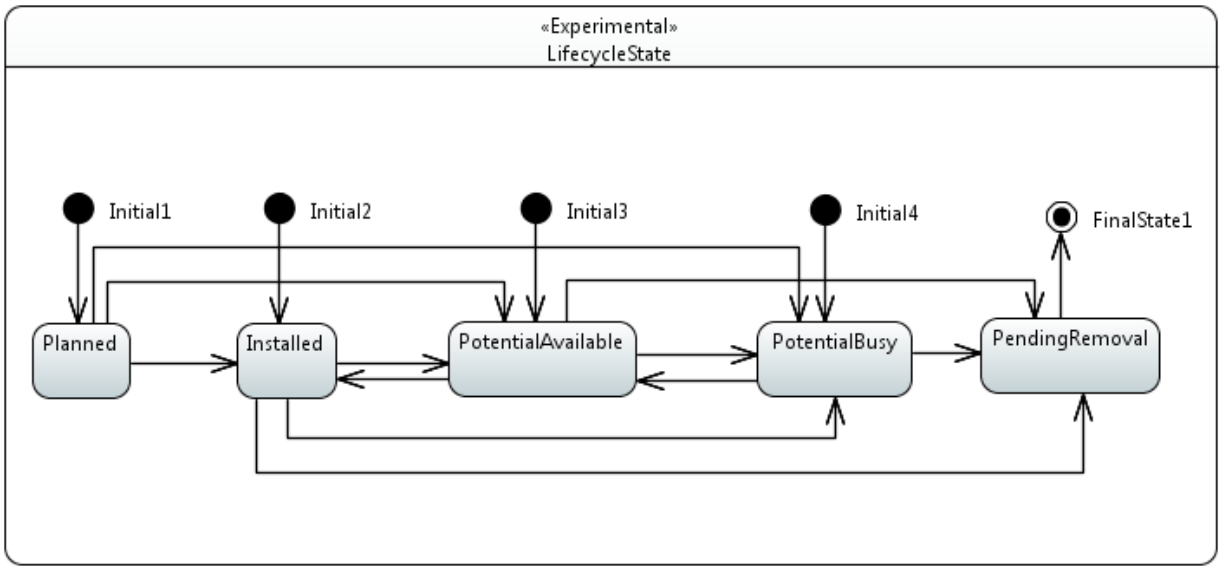


Figure 3-6 Lifecycle State

3.2.6 Use of states

Examples to be added.

End of document